

TelePACE Ladder Logic

User and Reference Manual

CONTROL MICROSYSTEMS

SCADA products... for the distance

48 Steacie Drive	Telephone:	613-591-1943
Kanata, Ontario	Facsimile:	613-591-1022
K2K 2A9	Technical Support:	888-226-6876
Canada		888-2CONTROL

©2008 Control Microsystems Inc.
All rights reserved.

Printed in Canada.

Trademarks

TelePACE, SCADASense, SCADAServer, SCADALog, RealFLO, TeleSAFE, SCADAPack, TeleBUS, SCADAWave and ClearSCADA are registered trademarks of Control Microsystems.

All other product names are copyright and registered trademarks or trade names of their respective owners.

Table of Contents

1	TELEPACE LADDER LOGIC OVERVIEW	21
1.1	System Requirements	21
1.2	Installation on a Hard Disk	21
1.3	Running TelePACE Ladder Logic Editor	21
2	TELEPACE LADDER EDITOR ENVIRONMENT	22
2.1	Introduction	22
2.2	Ladder Editor Display	22
2.2.1	Title Bar	22
2.2.2	Menu Bar	22
2.3	Tool Bar	22
2.3.1	Network Title	24
2.3.2	Comment Editor	24
2.3.3	Splitter Bar	24
2.3.4	Status Bar	24
2.3.5	Network Display	24
2.3.6	Register List	24
2.3.7	Cursor	24
2.4	TelePACE Major Components	25
2.4.1	Networks	25
2.4.2	Network Elements	25
2.4.3	Subroutines	25
2.4.4	Program Execution Order	26
2.4.5	Ladder Logic Memory Usage	26
2.5	TelePACE I/O Database Registers	27
2.5.1	16-bit Controller I/O Database	27
2.5.1.1	I/O Database register types	28
2.5.1.1.1	Coil Registers	28
2.5.1.1.2	Status Registers	29
2.5.1.1.3	Input Registers	29
2.5.1.1.4	Holding Registers	29
2.5.2	32-bit Controller I/O Database	29
2.5.2.1	I/O Database register types	30
2.5.2.1.1	Coil Registers	30
2.5.2.1.2	Status Registers	30
2.5.2.1.3	Input Registers	31
2.5.2.1.4	Holding Registers	31

2.5.3	SCADAPack 100: 256K I/O Database.....	31
2.5.3.1	I/O Database register types	32
2.5.3.1.1	Coil Registers	32
2.5.3.1.2	Status Registers	32
2.5.3.1.3	Input Registers	32
2.5.3.1.4	Holding Registers	32
3	TELEPACE PROGRAM DEVELOPMENT	33
3.1	Introduction	33
3.2	Configuration of the TelePACE Program	33
3.2.1	PC Communication Settings.....	33
3.2.2	Options.....	33
3.3	Initializing the Controller	33
3.4	Define Register Assignment	34
3.5	Create Ladder Logic Program.....	34
3.5.1	Inserting Elements.....	34
3.5.2	Inserting Networks.....	35
3.5.3	Editing Elements	35
3.5.4	Deleting Elements	35
3.5.5	Selecting Elements.....	35
3.6	Setting Outputs On Stop.....	35
3.7	Controller Serial Port Settings.....	36
3.8	Write program to controller	36
3.9	Monitor Program On Line	36
3.9.1	Contact Monitoring	36
3.10	Edit Program On Line	37
3.11	Force Registers	37
3.12	Preventing Unauthorized Changes	37
4	TELEPACE PROGRAM REFERENCE	38
4.1	File Menu	38
4.1.1	New	38
4.1.2	Open	38
4.1.3	Save	38
4.1.4	Save As	38
4.1.5	Page Setup.....	39
4.1.6	Print.....	39

4.1.7	Select Print Items	39
4.1.8	Quick File List.....	40
4.1.9	Exit	41
4.2	Edit Menu.....	41
4.2.1	Undo.....	41
4.2.2	Cut Selected	41
4.2.3	Copy Selected	41
4.2.4	Copy Networks	41
4.2.5	Cut Networks.....	41
4.2.6	Paste	42
4.2.7	Insert	42
4.2.7.1	Insert Vertical Shunt.....	42
4.2.7.2	Insert Element.....	43
4.2.7.3	Address Types	44
4.2.7.4	Empty Column	44
4.2.7.5	Empty Row.....	44
4.2.7.6	Network Before	44
4.2.7.7	Network After	44
4.2.8	Delete	44
4.2.8.1	Vertical Shunt.....	45
4.2.8.2	Element.....	45
4.2.8.3	Selected Elements	45
4.2.8.4	Empty Column	45
4.2.8.5	Empty Row.....	45
4.2.8.6	Network.....	45
4.2.8.7	Delete All Networks.....	45
4.2.9	Toggle Vertical Shunt	45
4.2.10	Tag Names.....	46
4.2.11	Erase All Tags	46
4.2.12	Export Tag Names	47
4.2.13	Import Tag Names.....	47
4.2.14	Network Title	48
4.2.15	Element Configuration	48
4.2.16	Registers	48
4.2.16.1	Register Editor	48
4.2.16.1.1	Groups	49
4.2.16.1.2	Update Rate	50
4.2.16.1.3	Register List	50
4.2.16.2	Add Registers Dialog	51
4.2.16.2.1	Registers In Use.....	53
4.2.16.2.2	Range.....	53
4.2.16.2.3	Tags.....	53
4.2.16.3	Edit Register Dialog	54

4.2.16.4	Register Data File	56
4.2.16.4.1	Register Data File Format.....	56
4.2.16.5	Editing the Register Data File	57
4.3	Search Menu	58
4.3.1	Next Network.....	58
4.3.2	Previous Network	58
4.3.3	Go To Network	58
4.3.4	Find Address	59
4.3.5	Find Device	59
4.3.6	Find Tag Name.....	60
4.3.7	Repeat Last Find	60
4.3.8	Replace Address	61
4.3.9	Multiple Coils	61
4.4	Controller Menu	61
4.4.1	Type	61
4.4.2	Serial Ports.....	62
4.4.3	IP Configuration.....	75
4.4.3.1	LAN Port	76
4.4.3.2	com1 Port.....	77
4.4.3.3	com2 Port.....	79
4.4.3.4	com3 Port.....	79
4.4.3.5	com4 Port.....	79
4.4.3.6	PPP Login	79
4.4.3.6.1	Add PPP Username dialog	80
4.4.3.6.2	Edit PPP Username dialog	81
4.4.3.7	Modbus Common.....	81
4.4.3.8	Modbus/TCP	83
4.4.3.9	Modbus RTU in UDP.....	84
4.4.3.10	Modbus ASCII in UDP.....	84
4.4.3.11	DNP in TCP	85
4.4.3.12	DNP in UDP.....	87
4.4.3.13	Friendly IP List	87
4.4.3.13.1	Add Friendly IP Address Range Dialog.....	89
4.4.4	Register Assignment	89
4.4.4.1.1	Edit Register Assignment Dialog	92
4.4.5	Outputs On Stop.....	93
4.4.6	Store and Forward.....	94
4.4.6.1.1	Add/Edit Store and Forward Dialog	96
4.4.7	DNP.....	97
4.4.8	DNP Status.....	97
4.4.9	DNP Master Status.....	97
4.4.10	Initialize	97

4.4.10.1	SCADAPack and SCADAPack 32 Controllers	98
4.4.10.2	SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203 Series of Controllers	99
4.4.10.3	Actions Performed when Controller is Initialized	100
4.4.11	Real Time Clock	101
4.4.12	Monitor Element	102
4.4.13	List Force Registers	103
4.4.14	Remove All Forces	104
4.4.15	Lock Controller	104
4.4.16	Unlock Controller	105
4.4.17	Override Controller Lock.....	105
4.4.18	Show Lock Status.....	106
4.4.19	C/C++ Program Loader	106
4.4.19.1	SCADAPack and SCADAPack 32 Controllers	106
4.4.19.1.1	Add C/C++ Program Dialog	107
4.4.19.2	SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203 Series Controllers	107
4.4.20	Flash Loader	108
4.4.20.1	Writing Programs to Flash.....	110
4.4.21	Program Status	111
4.4.21.1	SCADAPack and SCADAPack 32 Controllers	111
4.4.21.2	SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203 Series of Controllers	113
4.5	Communications Menu	114
4.5.1	Read from Controller	114
4.5.2	Write to Controller	114
4.5.3	PC Communications Settings.....	114
4.5.3.1	ClearSCADA	115
4.5.3.1.1	General Parameters	115
4.5.3.1.2	Advanced Parameters	116
4.5.3.1.3	Information	117
4.5.3.2	DNP	118
4.5.3.2.1	General Parameters	118
4.5.3.2.2	Flow Control Parameters	120
4.5.3.2.3	Dial Up Parameters	122
4.5.3.2.4	Advanced Parameters	123
4.5.3.2.5	Information	124
4.5.3.3	DNP/TCP	125
4.5.3.3.1	General Page	125
4.5.3.3.2	Advanced Page	127
4.5.3.3.3	Information Page	128
4.5.3.4	DNP/UDP	129
4.5.3.4.1	General Page	129
4.5.3.4.2	Advanced Page	130

4.5.3.4.3	Information Page	131
4.5.3.5	Modbus ASCII	132
4.5.3.5.1	General Parameters	133
4.5.3.5.2	Modbus ASCII Configuration (Flow Control)	134
4.5.3.5.3	Modbus ASCII Configuration (Dial Up)	136
4.5.3.5.4	Advanced Parameters	137
4.5.3.5.5	Information	138
4.5.3.6	Modbus ASCII in TCP	139
4.5.3.6.1	General Parameters	140
4.5.3.6.2	Advanced Parameters	142
4.5.3.6.3	Information	143
4.5.3.7	Modbus ASCII in UDP	143
4.5.3.7.1	General Parameters	144
4.5.3.7.2	Advanced Parameters	145
4.5.3.7.3	Information	146
4.5.3.8	Modbus RTU	147
4.5.3.8.1	Introduction	147
4.5.3.8.2	General Parameters	147
4.5.3.8.3	Modbus RTU Configuration (Flow Control)	149
4.5.3.8.4	Modbus RTU Configuration (Dial Up)	150
4.5.3.8.5	Advanced Parameters	152
4.5.3.8.6	Information	153
4.5.3.9	Modbus RTU in TCP	153
4.5.3.9.1	General Parameters	154
4.5.3.9.2	Advanced Parameters	156
4.5.3.9.3	Information	157
4.5.3.10	Modbus RTU in UDP	157
4.5.3.10.1	General Parameters	158
4.5.3.10.2	Advanced Parameters	159
4.5.3.10.3	Information	160
4.5.3.11	Modbus/TCP	161
4.5.3.11.1	General Parameters	161
4.5.3.11.2	Advanced Parameters	162
4.5.3.11.3	Information	163
4.5.3.12	Modbus/UDP	164
4.5.3.12.1	General Parameters	165
4.5.3.12.2	Advanced Parameters	166
4.5.3.12.3	Information	167
4.5.3.13	Modbus/USB	168
4.5.3.13.1	General Page	168
4.5.3.13.2	Information Page	170
4.5.3.14	SCADA Server	170
4.5.3.14.1	General Parameters	171
4.5.3.14.2	Advanced Parameters	172
4.5.3.14.3	Information	173
4.6	Activity Menu	173
4.6.1	Edit Off Line	173
4.6.2	Edit On Line	174

4.6.3	Monitor On Line	174
4.7	Operation Menu	174
4.7.1	Stop	174
4.7.2	Debug.....	174
4.7.3	Run.....	174
4.8	Options Menu	174
4.8.1	Screen Font.....	174
4.8.2	Colors.....	175
4.8.3	Floating-Point Settings	175
4.8.4	Tool Bar.....	175
4.8.5	Title Bar	175
4.8.6	Status Bar.....	175
4.8.7	Single Tag Names.....	175
4.8.8	Double Tag Names	176
4.8.9	Tag and Address	176
4.8.10	Numeric Address.....	176
4.8.11	Allow Multiple Coils	176
4.8.12	Warning Messages.....	176
4.9	Help Menu	176
4.9.1	Contents	177
4.9.2	About Program	177
5	TELEPACE LADDER LOGIC FUNCTIONS.....	178
5.2	ABS – Absolute Value	179
5.3	ABSF - Floating-Point Absolute Value	180
5.4	ADD – Add Signed Values.....	182
5.5	ADDF - Add Floating-Point Values	184
5.6	ADDU – Add Unsigned Values	186
5.7	AND – And Block.....	188
5.8	CALL - Execute Subroutine	190
5.9	Coil	192
5.10	CMPB – Compare Bit	193
5.11	CMP – Compare Signed Values	195
5.12	CMPU – Compare Unsigned Values	197
5.13	DCTR – Down Counter	199
5.14	DEVT – Generate DNP Event.....	201

5.15	DIAL – Control Dial-Up Modem	202
5.16	DIV – Divide Signed Values.....	206
5.17	DIVF - Divide Floating-Point Values	208
5.18	DIVU – Divide Unsigned Values	210
5.19	DLGF – Data Log to File.....	212
5.20	DLOG - Data Logger.....	223
5.21	DPOL – Trigger a DNP class poll	228
5.22	DSYC – Trigger a DNP clock synchronization	230
5.23	DUNS – Trigger a DNP unsolicited response message.....	231
5.24	FIN – FIFO Queue Insert	232
5.25	FOUT – FIFO Queue Remove.....	234
5.26	FLOW – Flow Accumulator	237
5.27	FTOS - Floating-Point to Signed Integer.....	241
5.28	FTOU - Floating-Point to Unsigned Integer	243
5.29	GETB – Get Bit from Block	245
5.30	GETL - Data Logger Extract	247
5.31	GTEF – Floating-Point Greater Than or Equal.....	249
5.32	HART – Send HART Command	251
5.33	INIM – Initialize Dial-Up Modem	257
5.34	L→L – List to List Transfer	260
5.35	L→R – List to Register Transfer.....	262
5.36	LTEF - Floating-Point Less Than or Equal.....	264
5.37	MOD – Modulus of Signed Values	266
5.38	MODU – Modulus of Unsigned Values	268
5.39	MOVE – Move Block.....	270
5.40	MSTR – Master Message	272
5.41	MSIP – Master IP Message	278
5.42	MUL – Multiply Signed Values.....	284
5.43	MULF - Multiply Floating-Point Values.....	286
5.44	MULU – Multiply Unsigned Values	288
5.45	Normally Closed Contact	290
5.46	Normally Open Contact	291
5.47	NOT – Not Block.....	292

5.48	One Shot Coil	294
5.49	OR – Or Block.....	294
5.50	OVER – Override Block of Registers	297
5.51	PID Controller	300
5.52	PIDA – Analog Output PID	301
5.53	PIDD – Digital Output PID	305
5.54	POWR – Floating-Point Raised to Power	309
5.55	PULM - Pulse Minutes	311
5.56	PULS - Pulse Seconds	312
5.57	PUTB – Put Bit into Block	313
5.58	PUT – Put Signed Value into Registers	315
5.59	PUTF - Put Floating-Point Value.....	317
5.60	PUTU – Put Unsigned Value into Registers	319
5.61	R→L – Register to List Transfer	321
5.62	ROTB – Rotate Bits in Block.....	323
5.63	SCAL – Scale Analog Value	325
5.64	Shunts	328
5.65	SLP – Put Controller into Sleep Mode	329
5.66	SQRF - Square Root of Floating-Point Value	332
5.67	STOF - Signed Integer to Floating-Point.....	334
5.68	SUB – Subtract Signed Values	336
5.69	SUBF – Subtract Floating-Point Values	338
5.70	SUBR - Start of Subroutine.....	340
5.71	SUBU –Subtract Unsigned Values	341
5.72	Timers.....	343
5.73	TOTL – Analog Totalizer.....	345
5.74	UCTR – Up Counter	349
5.75	UTOF - Unsigned Integer to Floating-Point	351
5.76	XOR – Exclusive Or Block	353
6	TELEPACE REGISTER ASSIGNMENT REFERENCE	355
6.1	Register Assignment Specification	357
6.2	Register Assignment Example.....	357

6.3	Analog Input I/O Modules	360
6.3.1	AIN Controller RAM Battery V	361
6.3.2	AIN Controller temperature.....	362
6.3.3	AIN 5501 Module.....	363
6.3.4	AIN 5502 Module.....	364
6.3.5	AIN 5503 Module.....	365
6.3.6	AIN 5504 Module.....	366
6.3.7	AIN 5505 Module.....	367
6.3.8	AIN 5506 Module.....	372
6.3.9	AIN 5521 Module.....	377
6.3.10	AIN Generic 8 Point Module	378
6.4	Analog Output I/O Modules.....	379
6.4.1	AOUT 5301 Module.....	380
6.4.2	AOUT 5302 Module.....	380
6.4.3	AOUT 5304 Module.....	382
6.4.4	AOUT Generic 2 Point Module	383
6.4.5	AOUT Generic 4 Point Module	384
6.5	Configuration I/O Modules	385
6.5.1	CNFG 5904 HART Interface Module	386
6.5.2	CNFG Clear Protocol Counters	388
6.5.3	CNFG Clear Serial Port Counters	389
6.5.4	CNFG Device Configuration	390
6.5.5	CNFG DTR Off	393
6.5.6	CNFG IP Settings.....	395
6.5.7	CNFG LED Power Settings	397
6.5.8	CNFG Modbus IP Interface	398
6.5.9	CNFG Modbus IP Protocols	400
6.5.10	CNFG Modbus/TCP Settings	402
6.5.11	CNFG PID Control Block	404
6.5.12	CNFG Power Mode	406
6.5.13	CNFG Protocol Settings Method 1	407
6.5.14	CNFG Protocol Settings Method 2	409
6.5.15	CNFG Protocol Settings Method 3	411
6.5.16	CNFG Real Time Clock and Alarm.....	414
6.5.17	CNFG Save Settings to EEPROM.....	416
6.5.18	CNFG Serial Port Settings.....	417
6.5.19	CNFG Store and Forward.....	419
6.6	Counter I/O Modules.....	421

6.6.1	CNTR Controller Counter Inputs	422
6.6.2	CNTR Controller Interrupt Input.....	423
6.6.3	CNTR 5410 input module	424
6.7	Diagnostic I/O Modules.....	425
6.7.1	DIAG Controller Status Code	426
6.7.2	DIAG DNP Connection Status.....	428
6.7.3	DIAG DNP Event Status Module	429
6.7.4	DIAG DNP Port Status Module.....	431
6.7.5	DIAG DNP Station Status.....	433
6.7.6	DIAG Force	435
6.7.7	DIAG IP Connections	436
6.7.8	DIAG Logic Status.....	437
6.8	DIAG Modbus Protocol Status	438
6.8.1	DIAG Serial Port Comm. Status	440
6.8.2	DIAG Serial Port Protocol Status.....	441
6.9	Digital Input I/O Modules	443
6.9.1	DIN Controller Digital Inputs	444
6.9.2	DIN Controller Interrupt Input	445
6.9.3	DIN Controller Option Switches.....	446
6.9.4	DIN SCADAPack 32 Option Switches	447
6.9.5	DIN 5401 Module	448
6.9.6	DIN 5402 Module	449
6.9.7	DIN 5403 Module	450
6.9.8	DIN 5404 Module	451
6.9.9	DIN 5405 Module	452
6.9.10	DIN 5414 Module	454
6.9.11	DIN 5421 Module	456
6.9.12	DIN Generic 16 Point Module.....	456
6.9.13	DIN Generic 8 Point Module.....	458
6.10	Digital Output I/O Modules.....	459
6.10.1	DOUT 5401 Module	460
6.10.2	DOUT 5402 Module	461
6.10.3	DOUT 5406 Module	462
6.10.4	DOUT 5407 Module	463
6.10.5	DOUT 5408 Module	464
6.10.6	DOUT 5409 Module	464
6.10.7	DOUT 5411 Module	466
6.10.8	DOUT 5415 Module	468

6.10.9	DOUT Generic 16 Point Module	470
6.10.10	DOUT Generic 8 Point Module	472
6.11	SCADAPack and SCADASense Series I/O Modules.....	473
6.11.1	4202 DR Extended/4203 DR IO	474
6.11.2	4202 DR I/O	476
6.11.3	4202/4203 DS I/O.....	478
6.11.4	SCADAPack AOUT Module	480
6.11.5	SCADAPack 5601 I/O Module.....	482
6.11.6	SCADAPack 5602 I/O Module.....	485
6.11.7	SCADAPack 5604 I/O Module.....	488
6.11.8	SCADAPack 5606 I/O Module.....	494
6.11.9	SCADAPack 5607 I/O Module.....	499
6.11.10	SCADAPack LP I/O	504
6.11.11	SCADAPack 100 I/O	508
6.11.12	SCADAPack 330 I/O	510
6.11.13	SCADAPack 334 I/O	512
6.11.14	SCADAPack 350 I/O	516
6.12	Controller Default Register Assignments	520
6.12.1	4202 DR Extended/4203 DR I/O Register Assignment	521
6.12.2	SCADASense 4202/4203 DS I/O Default Register Assignment	522
6.12.3	Micro16 Default Register Assignment (Backwards Compatible Modules)	523
6.12.3.1	Default Analog Input Module Assignment	523
6.12.3.2	Default Analog Output Module Assignment	523
6.12.3.3	Default Configuration Module Assignment	524
6.12.3.4	Default Counter Input Module Assignment.....	525
6.12.3.5	Default Diagnostic Module Assignment.....	526
6.12.3.6	Default Digital Input Module Assignment	526
6.12.3.7	Default Digital Output Module Assignment.....	526
6.12.4	Micro16 Default Register Assignment (Controller I/O Only)	527
6.12.5	SCADAPack (5601 I/O Module) Default Register Assignment	528
6.12.6	SCADAPack (5604 I/O Module) Default Register Assignment	530
6.12.7	SCADAPack LIGHT Default Register Assignment	532
6.12.8	SCADAPack PLUS (5601 I/O Module) Default Register Assignment.....	534
6.12.9	SCADAPack Plus (5604 I/O Module) Default Register Assignment	537
6.12.10	SCADAPack LP Default Register Assignment	539
6.12.11	SCADAPack 330 Default Register Assignment.....	541
6.12.12	SCADAPack 334 Default Register Assignment.....	542
6.12.13	SCADAPack 350 Default Register Assignment.....	543

6.12.14	SCADAPack 32 (5601 I/O Module) Default Register Assignment ...	544
6.12.15	SCADAPack 32 (5604 I/O Module) Default Register Assignment ...	546
6.12.16	SCADAPack 32P Default Register Assignment	548
6.12.17	SCADAPack 100 Default Register Assignment.....	549
7	TELEBUS PROTOCOLS OVERVIEW	551
7.1	Compatibility	551
7.2	Serial Port Configuration.....	551
7.2.1	Communication Parameters	551
7.2.1.1	RTU Protocol Parameters	552
7.2.1.2	ASCII Protocol Parameters	552
7.2.1.3	Baud Rate	553
7.2.1.4	Duplex.....	553
7.2.2	Protocol Parameters.....	554
7.2.2.1	Protocol Type	554
7.2.2.2	Station Number	554
7.2.2.3	Task Priority	555
7.2.2.4	Store and Forward Messaging	555
7.3	I/O Database.....	555
7.3.1	Accessing the I/O Database	556
7.3.1.1	Coil and Status Registers.....	556
7.3.1.2	Input and Holding Registers.....	556
7.3.1.3	Exception Status	556
7.3.1.4	Slave ID	557
7.4	Extended Station Addressing.....	557
7.4.1	Theory of Operation	557
7.5	Slave Mode.....	558
7.5.1	Broadcast Messages	558
7.5.2	Function Codes	558
7.5.2.1	Read Coil Status	559
7.5.2.2	Read Input Status	559
7.5.2.3	Read Holding Register	559
7.5.2.4	Read Input Register	559
7.5.2.5	Force Single Coil.....	559
7.5.2.6	Preset Single Register	559
7.5.2.7	Read Exception Status.....	559
7.5.2.8	Force Multiple Coils.....	559
7.5.2.9	Preset Multiple Registers	560
7.5.2.10	Report Slave ID.....	560
7.6	Modbus Master Mode	560
7.6.1	Modbus Function Codes.....	560

7.6.1.1	Read Coil Status	561
7.6.1.2	Read Input Status	561
7.6.1.3	Read Holding Register	561
7.6.1.4	Read Input Register	561
7.6.1.5	Force Single Coil.....	561
7.6.1.6	Preset Single Register	562
7.6.1.7	Force Multiple Coils.....	562
7.6.1.8	Preset Multiple Registers	562
7.6.2	Enron Modbus Master Mode	562
7.6.2.1	Variable Types	562
7.6.2.1.1	Boolean Registers	562
7.6.2.1.2	Short Integer Registers.....	563
7.6.2.1.3	Long Integer Registers	563
7.6.2.1.4	Floating Point Registers	563
7.6.2.2	Enron Modbus Function Codes.....	563
7.6.3	Sending Messages	564
7.7	Store and Forward Messaging.....	565
7.7.1	Translation Table.....	565
7.7.1.1	Table Size	566
7.7.2	Invalid Translations	566
7.7.3	Store and Forward Configuration	567
7.7.3.1	SCADAPack Controller	567
7.7.3.1.1	TelePACE Ladder Logic	567
7.7.3.1.2	TelePACE C Tools	567
7.7.3.1.3	ISaGRAF IEC61131	568
7.7.3.1.4	ISaGRAF IEC61131 C Tools	568
7.7.3.2	SCADAPack Light Controller	568
7.7.3.2.1	TelePACE Ladder Logic	569
7.7.3.2.2	TelePACE C Tools	569
7.7.3.2.3	ISaGRAF IEC61131	569
7.7.3.2.4	ISaGRAF IEC61131 C Tools	570
7.7.3.3	SCADAPack Plus Controller	570
7.7.3.3.1	TelePACE Ladder Logic	570
7.7.3.3.2	TelePACE C Tools	570
7.7.3.3.3	ISaGRAF IEC61131	571
7.7.3.3.4	ISaGRAF IEC61131 C Tools	571
7.7.3.4	SCADAPack LP Controller	572
7.7.3.4.1	TelePACE Ladder Logic	572
7.7.3.4.2	TelePACE C Tools	572
7.7.3.4.3	ISaGRAF IEC61131	572
7.7.3.4.4	ISaGRAF IEC61131 C Tools	573
7.7.3.5	SCADAPack 100 Controller	573
7.7.3.5.1	TelePACE Ladder Logic	573
7.7.3.5.2	TelePACE C Tools	574
7.7.3.5.3	ISaGRAF IEC61131	574
7.7.3.5.4	ISaGRAF IEC61131 C Tools	574

7.7.3.6	SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 and 32P Controller	575
7.7.3.6.1	TelePACE Ladder Logic	575
7.7.3.6.2	TelePACE C++ Tools	575
7.7.3.6.3	ISaGRAF IEC61131	576
7.7.3.6.4	ISaGRAF IEC61131 C++ Tools	576
7.7.4	Diagnostics Counters	576
7.8	Point-To-Point Protocol (PPP).....	577
7.8.1	PPP Client Setup in Windows 2000	577
7.8.1.1	Direct Serial PPP Connection using Windows 2000	577
7.8.1.1.1	Connection Setup	577
7.8.1.1.2	Making a PPP Connection to the SCADAPack 32	585
7.8.1.1.3	Disconnecting a PPP Connection	587
7.8.1.2	Dial-up PPP Connection using Windows 2000	588
7.8.1.2.1	Connection Setup using Windows 2000	588
7.8.1.2.2	Making a PPP Dial-up Connection to the SCADAPack 32 using Windows 2000	595
7.8.1.2.3	Disconnecting a PPP Connection using Windows 2000	597
8	DNP3 USER AND REFERENCE MANUAL	599
8.1	DNP3 Protocol Overview	599
8.1.1	DNP Architecture.....	599
8.1.1.1	Object Library.....	600
8.1.1.1.1	Internal Indication (IIN) Flags.....	600
8.1.1.2	Application Layer	602
8.1.1.3	Pseudo-Transport Layer	602
8.1.1.4	Data Link Layer	602
8.1.1.5	Physical Layer.....	602
8.1.2	Modbus Database Mapping.....	603
8.1.3	SCADAPack DNP Operation Modes	603
8.2	SCADAPack DNP Outstation.....	603
8.2.1	How to Configure SCADAPack DNP Outstation.....	604
8.2.1.1	Tasks to Complete	605
8.2.1.2	Enable DNP on Communication Interface.....	605
8.2.1.3	Configure DNP Outstation.....	605
8.2.1.4	Confirm Successful Configuration	609
8.3	SCADAPack DNP Master	610
8.3.1	SCADAPack DNP Master Concepts	611
8.3.1.1	SCADAPack DNP Mimic Master	611
8.3.1.2	SCADAPack DNP Address Mapping	613
8.3.2	How to Configure SCADAPack DNP Master	615
8.3.2.1	Tasks to Complete	615
8.3.2.2	Configuration Steps.....	615
8.3.2.3	Confirm Successful DNP Master Configuration	618

8.3.3	How to Configure SCADAPack Address Mapping.....	619
8.3.4	How to Configure SCADAPack DNP Mimic Master.....	620
8.4	SCADAPack DNP Router	620
8.4.1	How to Configure a SCADAPack DNP Router	622
8.4.1.1	Tasks to Complete	622
8.4.1.2	Configuration Steps.....	622
8.5	Design Considerations.....	625
8.5.1	Considerations of DNP3 Protocol and SCADAPack DNP Driver....	625
8.5.1.1	Unsolicited Messages always request for a Confirmation	625
8.5.1.2	Master shall never request for Application Layer Confirmation ..	625
8.5.1.3	DNP Write Messages always request for a Confirmation.....	625
8.5.1.4	Only one DNP3 transaction can be pending at a time	625
8.5.1.5	SCADAPack controllers buffer 3 DNP messages	625
8.5.1.6	Output points in DNP Address Mapping issue DNP Write	626
8.5.2	Typical Configuration Malpractices and Recommendations	626
8.5.2.1	Multiple High Priority Unsolicited Messages	627
8.5.2.2	Master not polling frequently causing event buffer overflows....	627
8.5.2.3	Outstation reports to Multiple Masters with Poor Comms	628
8.5.2.4	Insufficient Use of Input Deadband or Debounce.....	628
8.5.2.5	Master Confirmation and Retries	628
8.5.2.6	Outstation Confirmations and Retries	628
8.5.2.7	Setting relatively large Application Layer timeouts.....	630
8.5.2.8	DNP Address mapping contains multiple output points.....	630
8.5.3	Configuration FAQ.....	630
8.6	DNP Configuration Menu Reference	632
8.6.1	Application Layer Configuration.....	634
8.6.2	Data Link Layer Configuration	639
8.6.3	Master	642
8.6.4	Master Poll	643
8.6.4.1	Add/Edit Master Poll Dialog	644
8.6.4.2	Poll Offset Example.....	648
8.6.5	Address Mapping	648
8.6.5.1	Add/Edit Address Mapping Dialog	650
8.6.6	Routing.....	651
8.6.6.1	Add/Edit DNP Route Dialog	653
8.6.6.2	Dynamic Routing.....	654
8.6.7	Binary Inputs Configuration	654
8.6.7.1	Adding Binary Inputs.....	656
8.6.8	Binary Outputs Configuration	657
8.6.8.1	Adding Binary Outputs	658
8.6.9	16-Bit Analog Inputs Configuration	660
8.6.9.1	Adding 16-Bit Analog Inputs.....	661

8.6.10	32-Bit Analog Inputs Configuration	663
8.6.10.1	Adding 32-Bit Analog Inputs.....	664
8.6.11	Short Floating Point Analog Inputs	666
8.6.11.1	Adding Short Floating Point Analog Inputs.....	668
8.6.12	16-Bit Analog Outputs Configuration	669
8.6.12.1	Adding 16-Bit Analog Outputs.....	670
8.6.13	32-Bit Analog Outputs Configuration	672
8.6.13.1	Adding 32-Bit Analog Outputs.....	672
8.6.14	Short Floating Point Analog Outputs	674
8.6.14.1	Adding Short Floating Point Analog Outputs.....	675
8.6.15	16–Bit Counter Inputs Configuration	677
8.6.15.1	Adding 16-Bit Counter Inputs	678
8.6.16	32-Bit Counter Inputs Configuration	680
8.6.16.1	Adding 32-Bit Counter Inputs	681
8.7	DNP Diagnostics.....	683
8.7.1	DNP Status.....	684
8.7.1.1	Overview Tab.....	684
8.7.1.2	Point Status Tabs.....	686
8.7.2	DNP Master Status.....	687
8.7.2.1	All Stations Tab	688
8.7.2.2	Remote Overview Tab	689
8.7.2.3	Remote Point Status Tabs	690
8.8	DNP Master Device Profile Document.....	691
8.9	DNP Slave Device Profile Document.....	700
9	TELEBUS DF1 PROTOCOL OVERVIEW	711
9.1	Compatibility	711
9.2	Serial Port Configuration.....	712
9.2.1	Communication Parameters	712
9.2.1.1	Protocol Parameters	712
9.2.1.2	Baud Rate	713
9.2.1.3	Duplex.....	713
9.2.2	Protocol Parameters.....	714
9.2.2.1	Protocol Type.....	714
9.2.2.2	Station Number	714
9.2.2.3	Task Priority	714
9.2.2.4	Store and Forward Messaging	714
9.2.3	Default Parameters	715
9.3	I/O Database.....	715
9.3.1	Coil and Status Registers	716
9.3.2	Input and Holding Registers	716

9.3.3	Accessing the I/O Database Using TelePACE	717
9.3.3.1	Modbus Addressing	717
9.3.3.1.1	SCADAPack, SCADAPack 100: 1024K and SCADASense 4202 Series	717
9.3.3.1.2	SCADAPack 100: 256K Controllers.....	717
9.3.3.1.3	SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32, and SCADASense 4203 Series.....	717
9.3.3.2	DF1 Addressing	718
9.3.3.3	Converting Modbus to DF1 Addresses	718
9.3.3.3.1	SCADAPack, SCADAPack 100: 1024K and SCADASense 4202 Series	718
9.3.3.3.2	SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADAPack 32 Controllers.....	719
9.3.4	Accessing the I/O Database Using ISaGRAF.....	720
9.3.4.1	Modbus Addressing	720
9.3.4.2	DF1 Addressing	721
9.3.4.3	Converting Modbus to DF1 Addresses	721
9.3.4.3.1	Coil Registers	721
9.3.4.3.2	Status Registers	721
9.3.4.3.3	Input Registers	722
9.3.4.3.4	Holding Registers	722
9.3.4.3.5	Example	722
9.4	Slave Mode.....	722
9.4.1	Broadcast Messages.....	722
9.4.2	Function Codes	723
9.4.2.1	Protected Write	723
9.4.2.2	Unprotected Read	723
9.4.2.3	Protected Bit Write	723
9.4.2.4	Unprotected Bit Write	724
9.4.2.5	Unprotected Write	724
9.5	Master Mode.....	724
9.5.1	Function Codes	724
9.5.1.1	Protected Write	725
9.5.1.2	Unprotected Read	725
9.5.1.3	Protected Bit Write	725
9.5.1.4	Unprotected Bit Write	725
9.5.1.5	Unprotected Write	725
9.5.2	Sending Messages.....	726
9.5.2.1	Polling DF1 PLCs.....	726
9.5.2.1.1	Notes.....	726
10	MODEM COMMANDS	728
10.1	Modem Settings.....	728
10.2	Generic Modem	728

10.3	5901 High Speed Dial-up Modem.....	729
10.4	ATI 14400 ETC-Express.....	729
10.5	ATI 14400 ETC-E, ETC-I.....	729
10.6	Hayes Smartmodem 1200.....	729
10.7	Hayes ACCURA 96, 144 and 288 Modems.....	730
10.8	Kama 2400 EI.....	730
10.9	Megahertz XJ4288 28.8 PC Card Modem.....	730
10.10	Multitech 224E7B.....	730
10.11	TeleSAFE 6901 Bell 212 Modem.....	731
10.11.1.1	String.....	731
10.12	US Robotics Sportster 28,800.....	731

1 TelePACE Ladder Logic Overview

TelePACE ladder logic is ideal for electricians, engineers, and programmers who program sequencing and process control. The ladder logic editor is a powerful tool for writing, debugging and documenting ladder logic programs. The SCADAPack and TeleSAFE family of controllers executes ladder logic and C application programs simultaneously, providing you with maximum flexibility in implementing your control strategy.

This manual provides full documentation on the TelePACE program including the Ladder Network Editor and the ladder logic programming language. We strongly encourage you to read it, and to notify us if you identify any items that you feel should be clarified, or included in future documentation releases.

We sincerely hope that the reliability and flexibility afforded by this fully programmable controller enable you and your company to solve your automation problems in a cost effective and efficient manner.

To use TelePACE, you need to install the program on your system. The automated installation takes only a few minutes.

Some virus checking software may interfere with Setup. If you experience problems with Setup, disable your virus checker and run Setup again.

1.1 System Requirements

The TelePACE ladder editor requires the following minimum system configuration.

- Microsoft Windows NT, Windows 2000 or Windows XP operating systems. Note that Windows 95, 98 and ME operating systems are no longer supported.
- Mouse or compatible pointing device; and
- Hard disk with approximately 2.5 Mbytes of free disk space.

1.2 Installation on a Hard Disk

To install the TelePACE Ladder Logic Editor:

1. Click **Start**, then select **Run**.
2. From the **Run** dialog box, select **Browse** and select the **setup.exe** file in the TelePACE Demo folder on the CD.
3. Follow the setup directions on the screen.

1.3 Running TelePACE Ladder Logic Editor

To run the ladder logic editor:

- Click Start, then select the TelePACE icon grouping, and select **TelePACE Ladder Editor**.

2 TelePACE Ladder Editor Environment

2.1 Introduction

The TelePACE Ladder Network Editor is a powerful programming and monitoring tool that enables the user to create, edit, document, save, and read or write Ladder Logic programs for the controller.

This section of the manual outlines the features of the Ladder Logic Editor. A full description of the Editor is described in the **TelePACE Program Reference** section of this manual. The user is encouraged to read the following sections to gain an understanding of the many programming and monitoring features available.

2.2 Ladder Editor Display

The following figure illustrates the TelePACE Ladder Editor:

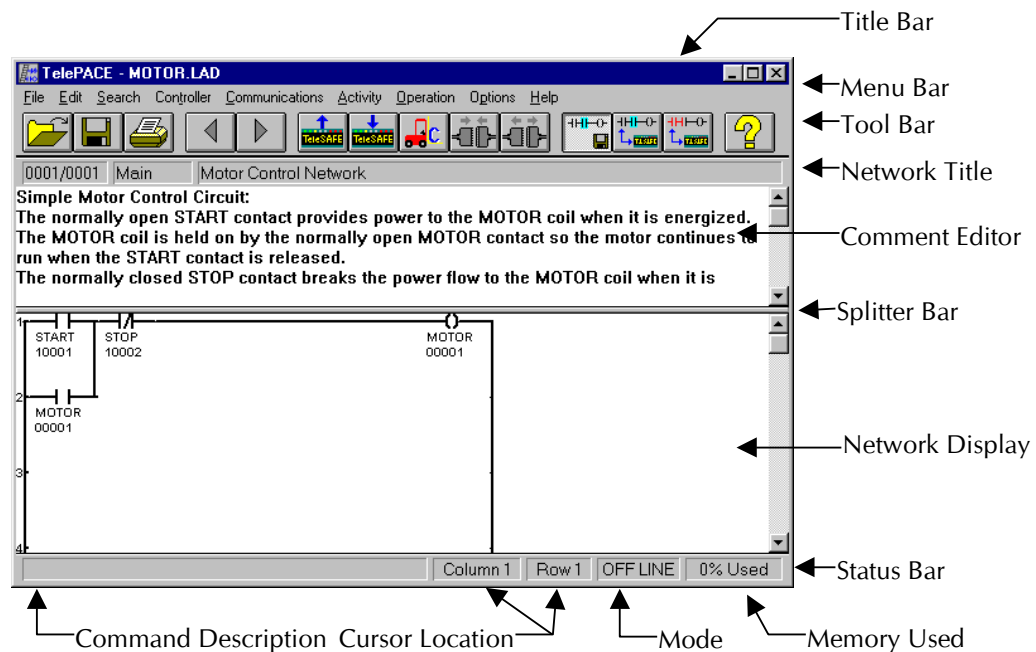


Figure 1: TelePACE Ladder Editor Display

2.2.1 Title Bar

The title bar displays the TelePACE program title and the name of the ladder logic file. The file name area is blank if there is no file in use or the file is new and has not been saved.

2.2.2 Menu Bar

The menu bar displays the programming, editing, monitoring and utility functions available within the Editor. Menu commands can be displayed by clicking the mouse on the menu item or by pressing the alt key and the underlined letter of the menu item.

2.3 Tool Bar

The Tool Bar contains buttons for commonly used commands.

Icon	Command	Description
	Open File	Opens an existing ladder program file. This is the same as the Open command in the File menu.
	Save File	Saves the currently open ladder program file. This is the same as the Save command on the File menu.
	Print	Prints the currently open ladder program file. This is the same as the Print command on the File menu. The Print dialog box appears. This option is disabled when the editor is in Monitor On Line mode.
	Previous Network	Displays the previous network in the currently open ladder program. This is the same as selecting Previous from the Go To Network command in the Search menu.
	Next Network	Displays the next network in the currently open ladder program. This is the same as selecting Next from the Go To Network command in the Search menu.
	Read From Controller	The ladder logic program, register assignment and serial port settings in the controller are read into TelePACE memory. This is the same as the Read From Controller command on the Communication menu.
	Write To Controller	The ladder logic program, register assignment and serial port settings in the TelePACE program are written to the controller. This is the same as the Write To Controller command on the Communication menu.
	C Program Loader	The C Program Loader dialog box appears. This is the same as the C Program Loader command on the Controller menu.
	Connect to Controller	Establishes a connection to a remote unit using a dial up modem. This button is disabled if the current controller is not a dial up connection. This is the same as the Connect To Controller command on the Communication menu.
	Disconnect from Controller	Ends a connection to a remote unit using a dial up modem. This button is disabled if the current controller is not a dial up connection. This is the same as the Disconnect From Controller command on the Communication menu.
	Edit Off Line	Places the editor in Off Line editing mode. This mode is used for creation and editing of a ladder program file. This is the same as the Edit Off Line command on the Activity menu.
	Edit On Line	Places the editor in On Line editing mode. In this mode, any change to the ladder program is immediately written to the controller. This is the same as the Edit On Line command on the Activity menu.
	Monitor On Line	Places the TelePACE program in Monitor On Line mode. This mode is used to monitor ladder program execution in real time. Color is used to indicate power flow within the displayed network. The contents of a user-defined list of registers (see Monitor List) are updated in real time. This is the same as the Monitor On Line command on the Activity menu.
	Help	Displays the TelePACE program on-line help file. This is the same as selecting Contents from the Help menu.

2.3.1 Network Title

The network title has three sections. The left section displays the number of the network being edited and the total number of networks (current/total). Clicking the left mouse button here opens the **Go To Network** dialog.

The center section displays Main if the current network is in the main program and Sub x if it's in a subroutine.

The right section of the Network Title shows the network title. Clicking here opens the **Edit Network Title** pop up dialog box.

2.3.2 Comment Editor

Each network in the ladder program can have approximately three pages of text documentation. The text in the comment editor can be cut or copied to the Windows clipboard or pasted from the clipboard.

The comment editor is selected for editing by positioning the mouse pointer anywhere in the comment editor pane and left clicking the mouse button. A vertical bar indicates the cursor position within the comment editor pane. The scroll bar to the right of the comment editor pane scrolls the editor text within the pane.

2.3.3 Splitter Bar

The splitter bar divides the Comment Editor and the Ladder Editor Panes. Positioning the mouse pointer over the bar and dragging the bar to a new location changes the position of the splitter bar.

2.3.4 Status Bar

The status bar displays command and programming status of the Network Editor. The Status Bar is divided into five panels.

The leftmost panel describes the Toolbar or Menu command pointed to by the mouse.

The second and third panels show the column and row position of the cursor in the Ladder Editor pane.

The fourth panel displays the execution mode of the attached controller, or OFFLINE if the Network Editor is not communicating with a controller.

The last panel displays the percentage of ladder logic memory used in the target controller.

2.3.5 Network Display

The Ladder Editor Pane displays the current network of the open program. This area is used for creation and editing of the ladder logic. When in Monitor On Line mode, this area also contains the Monitor list.

2.3.6 Register List

The register list is a display window containing a list of selected I/O database registers and their real time contents. The I/O database registers to be included in monitor list display window are defined by selecting **Registers** from the **Edit** Menu.

2.3.7 Cursor

To position the cursor in the Ladder editor pane move the mouse pointer to any position in the displayed Ladder network and click the left mouse button. The cursor highlights the element or function where it is positioned. The cursor can be moved about the network using the arrow keys on the keyboard.

2.4 TelePACE Major Components

The major components of a ladder program are ladder networks, function elements, subroutines, and comments. Understanding how these components work and how a program is executed is important to writing an effective program.

NOTE: Program execution order (described below) must be considered when organizing the ladder logic program.

2.4.1 Networks

A network is a diagrammatic representation of control logic similar to a wiring schematic, showing interconnection of relays, timers, contacts and other control elements.

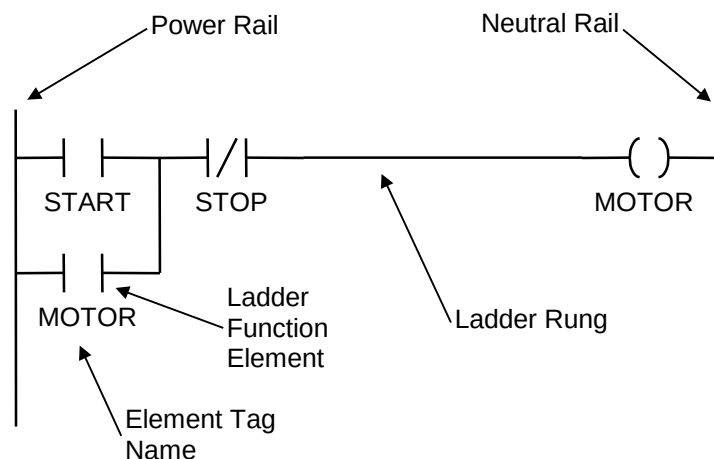


Figure 2: A Ladder Network

Networks in the TelePACE Ladder Editor contain up to eight rungs. Each rung can contain a maximum of 10 logic elements. The highlighted cursor in the Ladder Editor pane occupies one logic element position.

The number of networks in a program is only limited by the memory available.

2.4.2 Network Elements

Network elements are contacts, coils, and function blocks. Shunts are used to interconnect elements. Coils are always found connected to the neutral rail and represent either physical outputs in the controller or internal (memory only) outputs in the I/O database. Contacts represent physical status inputs from the controller or internal (memory only) status inputs in the I/O database. Function blocks are used to perform specific functions, such as moving data, manipulating data or communicating data.

2.4.3 Subroutines

Subroutines permit conditional execution of parts of a Ladder Logic program. Subroutines can be used to reduce the scan time of a program by only scanning code when it is absolutely needed. Reduced scan time will increase the frequency of program execution.

Subroutines can also reduce the size of the program by placing frequently used or repeated code into subroutines that are called from various locations in the main program.

The Ladder Logic program consists of the main program followed by a number of subroutines.

The main program is defined as all the logic networks up to the start of the first subroutine, or until the end of the program if no subroutines exist.

A subroutine is defined as all the logic networks from a subroutine element until the next subroutine element, or the end of the program if there are no more subroutines.

The program is executed from the start of the main program to the end of the main program. If a subroutine call function block is encountered, execution transfers to the start of the subroutine and continues until the end of the subroutine. Execution then returns to the element after the subroutine call element.

Subroutine execution can be nested. This allows subroutines to call other subroutines.

Subroutine execution cannot be recursive. This prevents potential infinite loops in the ladder logic program.

Two elements control the definition and execution of subroutines. The **SUBR** element defines the start of a subroutine. The **CALL** element executes a subroutine.

2.4.4 Program Execution Order

The controller evaluates each element, or function block, in the network in a sequence that starts at the top left hand corner; or ROW 1, COLUMN 1.

The ladder evaluation moves down column 1 until it reaches row 8. The evaluation then continues at the top of column 2 and moves down to row 8 again.

This process continues until the entire network has been evaluated. If there is more than one network, evaluation continues to the next sequential network in the program until the entire program has been evaluated.

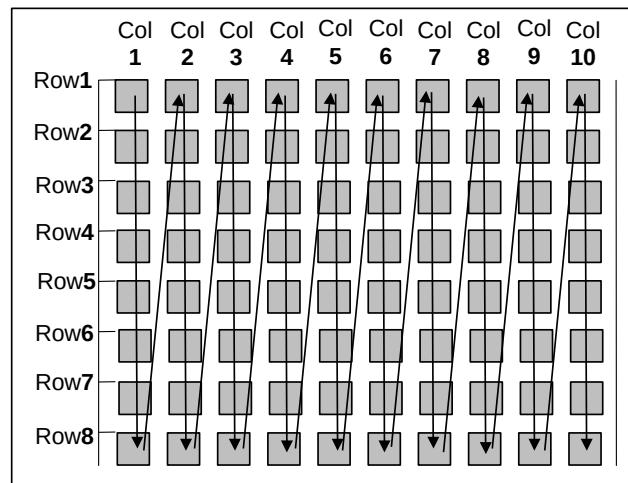


Figure 3: TelePACE Ladder Editor Network Execution

2.4.5 Ladder Logic Memory Usage

Memory usage in a Ladder Logic application program is based on the number of networks and number of elements used by a program.

Networks	Each network in an application requires one word of memory, whether the network is used or not.
Columns	Each column that is occupied in a network requires one word of memory.

Single Elements	Each single element requires one word of memory.
Double Elements	Each double element requires two words of memory.
Triple Elements	Each triple element requires three words of memory.

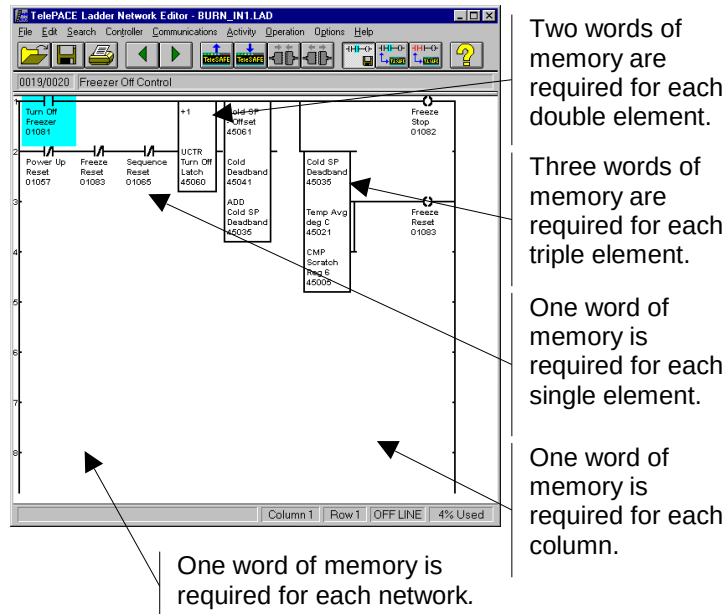


Figure 4: Ladder Logic Memory Usage

2.5 TelePACE I/O Database Registers

The I/O Database is different on the 16 and 32-bit controllers.

The 16-bit controllers comprise of SCADAPack (Light and Plus), SCADAPack 100, SCADAPack LP, SCADASense 4202 DR and SCADASense 4202 DS. The SCADAPack 100: 1024K controller differs from the SCADAPack 100: 256 K controller in that it has the same I/O database as other SCADAPack controllers. The SCADAPack 100: 1024K controller has firmware version 1.80 or newer and a controller ID that is greater than or equal to A182922.

The 32-bit controllers comprise SCADAPack 330, SCADAPack 350, SCADAPack 32, SCADASense 4203 DR and SCADASense 4203 DS.

Note: In this manual, the SCADASense 4202 DR and DS controllers are collectively referred to as the SCADASense 4202 Series controller. The SCADASense 4203 DR and DS are collectively referred to as the SCADASense 4203 Series controllers. The 4202 and 4203 controllers are collectively referred to as the SCADASense controllers.

2.5.1 16-bit Controller I/O Database

The 16-bit controllers which comprise of SCADAPack (Light and Plus), SCADAPack 100:1024K, SCADAPack LP, and SCADASense 4202 Series controllers all have the same IO database. The IO database in the SCADAPack 100: 256K is different as indicated in section **2.5.3-SCADAPack 100: 256K I/O Database**.

The I/O database allows data to be shared between C programs, Ladder Logic programs and communication protocols. A simplified diagram of the I/O Database is shown in **Figure 5**:

SCADAPack, SCADAPack 100: 1024K, SCADAPack LP, SCADASense 4202 Series I/O Database Block Diagram.

The I/O database contains general purpose and user-assigned registers. Ladder Logic and C application programs to store processed information and to receive information from a remote device may use general-purpose registers. Initially all registers in the I/O Database are general-purpose registers.

User-assigned registers are mapped directly from the I/O database to physical I/O hardware, or to controller system configuration and diagnostic parameters. The Register Assignment performs the mapping of registers from the I/O database to physical I/O hardware and system parameters.

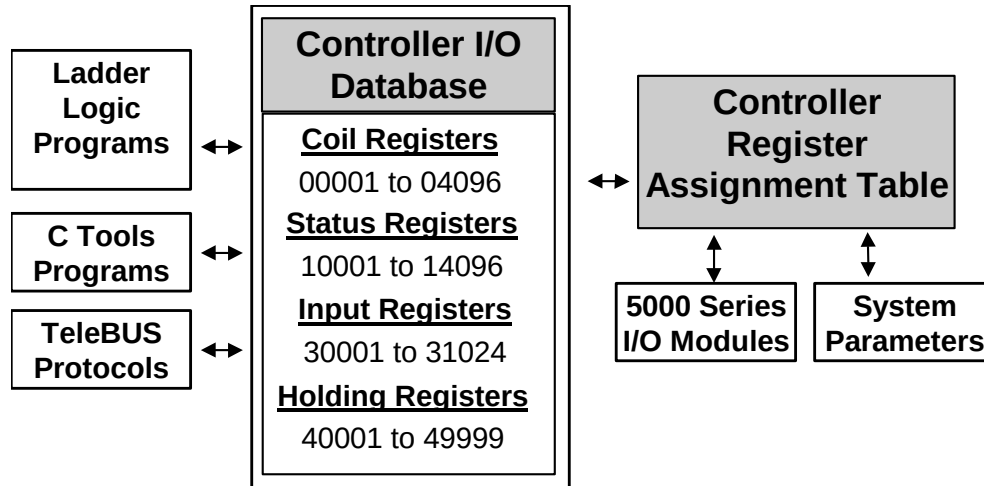


Figure 5: SCADAPack, SCADAPack 100: 1024K, SCADAPack LP, SCADASense 4202 Series I/O Database Block Diagram

User-assigned registers are initialized to the default hardware state or system parameter when the controller is reset. Assigned output registers do not maintain their values during power failures. Assigned output registers do retain their values during application program loading.

General-purpose registers retain their values during power failures and application program loading. The values change only when written by an application program or a communication protocol.

The TeleBUS communication protocols provide a standard communication interface to the controller. The TeleBUS protocols are compatible with the widely used Modbus RTU and ASCII protocols. They provide full access to the I/O database in the controller.

2.5.1.1 I/O Database register types

The I/O database is divided into four types of I/O registers. Each of these types is initially configured as general purpose registers by the controller.

2.5.1.1.1 Coil Registers

Coil registers are single bit registers located in the digital output section of the I/O database. Coil, or digital output, database registers may be assigned to 5000 Series digital output modules or SCADAPack I/O modules through the Register Assignment. Coil registers may also be assigned to controller on board digital outputs and to system configuration modules.

There are 4096 coil registers numbered 00001 to 04096. Ladder logic programs, C language programs, and the TeleBUS protocols can read from and write to these registers.

2.5.1.1.2 Status Registers

Status registers are single bit registers located in the digital input section of the I/O database. Status, or digital input, database registers may be assigned to 5000 Series digital input modules or SCADAPack I/O modules through the Register Assignment. Status registers may also be assigned to controller on board digital inputs and to system diagnostic modules.

There are 4096 status registers numbered 10001 to 14096. Ladder logic programs and the TeleBUS protocols can only read from these registers. C language programs can read data from and write data to these registers.

2.5.1.1.3 Input Registers

Input registers are 16 bit registers located in the analog input section of the I/O database. Input database registers may be assigned to 5000 Series analog input modules or SCADAPack I/O modules through the Register Assignment. Input registers may also be assigned to controller internal analog inputs and to system diagnostic modules.

There are 1024 input registers numbered 30001 to 31024. Ladder logic programs and the TeleBUS protocols can only read from these registers. C language programs can read data from and write data to these registers.

2.5.1.1.4 Holding Registers

Holding registers are 16 bit registers located in the analog output section of the I/O database. Holding, or analog output, database registers may be assigned to 5000 Series analog output modules or SCADAPack analog output modules through the Register Assignment. Holding registers may also be assigned to system diagnostic and configuration modules.

There are 9999 holding registers numbered 40001 to 49999. Ladder logic programs, C language programs, and the TeleBUS protocols can read from and write to these registers.

2.5.2 32-bit Controller I/O Database

The 32-bit controllers, which comprise of SCADAPack32, SCADAPack 330, SCADAPack 350, SCADASense 4203 Series (SCADASense 4203 DR and SCADASense 4203 DS) , all have the same IO database.

The I/O database allows data to be shared between C programs, Ladder Logic programs and communication protocols. A simplified diagram of the I/O Database is shown in **Figure 6: SCADASense 4203 Series, SCADAPack 330, SCADAPack 350 and SCADAPack 32 I/O Database Block Diagram.**

The I/O database contains general purpose and user-assigned registers. Ladder Logic and C application programs to store processed information and to receive information from a remote device may use general-purpose registers. Initially all registers in the I/O Database are general-purpose registers.

User-assigned registers are mapped directly from the I/O database to physical I/O hardware, or to controller system configuration and diagnostic parameters. The Register Assignment performs the mapping of registers from the I/O database to physical I/O hardware and system parameters.

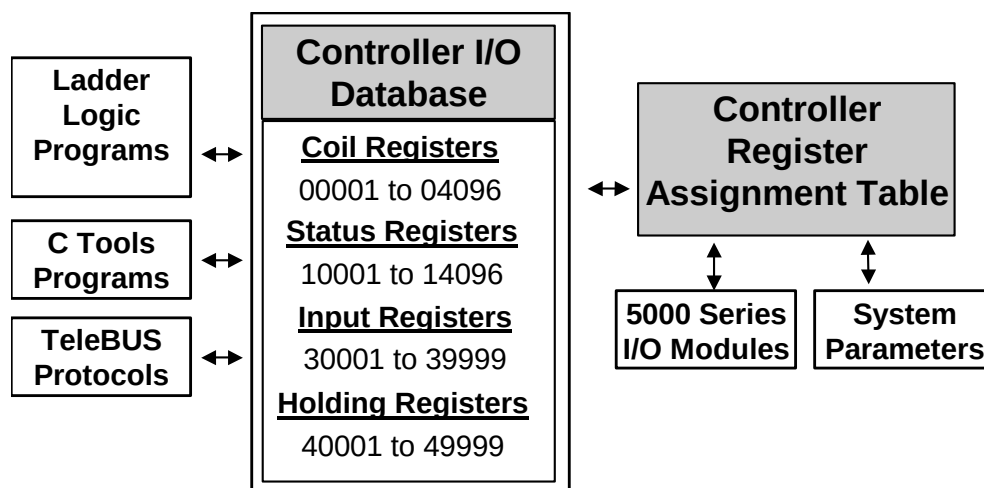


Figure 6: SCADASense 4203 Series, SCADAPack 330, SCADAPack 350 and SCADAPack 32 I/O Database Block Diagram

User-assigned registers are initialized to the default hardware state or system parameter when the controller is reset. Assigned output registers do not maintain their values during power failures. Assigned output registers do retain their values during application program loading.

General-purpose registers retain their values during power failures and application program loading. The values change only when written by an application program or a communication protocol.

The TeleBUS communication protocols provide a standard communication interface to the controller. The TeleBUS protocols are compatible with the widely used Modbus RTU and ASCII protocols. They provide full access to the I/O database in the controller.

2.5.2.1 I/O Database register types

The I/O database is divided into four types of I/O registers. Each of these types is initially configured as general purpose registers by the controller.

2.5.2.1.1 Coil Registers

Coil registers are single bit registers located in the digital output section of the I/O database. Coil, or digital output, database registers may be assigned to 5000 Series digital output modules or SCADAPack I/O modules through the Register Assignment. Coil registers may also be assigned to controller on board digital outputs and to system configuration modules.

There are 4096 coil registers numbered 00001 to 04096. Ladder logic programs, C language programs, and the TeleBUS protocols can read from and write to these registers.

2.5.2.1.2 Status Registers

Status registers are single bit registers located in the digital input section of the I/O database. Status, or digital input, database registers may be assigned to 5000 Series digital input modules or SCADAPack I/O modules through the Register Assignment. Status registers may also be assigned to controller on board digital inputs and to system diagnostic modules.

There are 4096 status registers numbered 10001 to 14096. Ladder logic programs and the TeleBUS protocols can only read from these registers. C language programs can read data from and write data to these registers.

2.5.2.1.3 Input Registers

Input registers are 16 bit registers located in the analog input section of the I/O database. Input database registers may be assigned to 5000 Series analog input modules or SCADAPack I/O modules through the Register Assignment. Input registers may also be assigned to controller internal analog inputs and to system diagnostic modules.

There are 9999 input registers numbered 30001 to 39999. Ladder logic programs and the TeleBUS protocols can only read from these registers. C language programs can read data from and write data to these registers.

2.5.2.1.4 Holding Registers

Holding registers are 16 bit registers located in the analog output section of the I/O database. Holding, or analog output, database registers may be assigned to 5000 Series analog output modules or SCADAPack analog output modules through the Register Assignment. Holding registers may also be assigned to system diagnostic and configuration modules.

There are 9999 holding registers numbered 40001 to 49999. Ladder logic programs, C language programs, and the TeleBUS protocols can read from and write to these registers.

2.5.3 SCADAPack 100: 256K I/O Database

The SCADAPack 100: 256K controller differs from the SCADAPack 100: 1024K controller in that it has a limited I/O database. The SCADAPack 100: 256K controller has firmware version older than 1.80 and a controller ID that is less or equal to A182921.

The SCADAPack 100: 256K I/O database allows data to be shared between C programs, Ladder Logic programs and communication protocols. A simplified diagram of the I/O Database is shown in **Figure 7: SCADAPack 100 – 256K I/O Database Block Diagram**.

The I/O database contains general purpose and user-assigned registers. Ladder Logic and C application programs to store processed information and to receive information from a remote device may use general-purpose registers. Initially all registers in the I/O Database are general-purpose registers.

User-assigned registers are mapped directly from the I/O database to physical I/O hardware, or to controller system configuration and diagnostic parameters. The Register Assignment performs the mapping of registers from the I/O database to physical I/O hardware and system parameters.

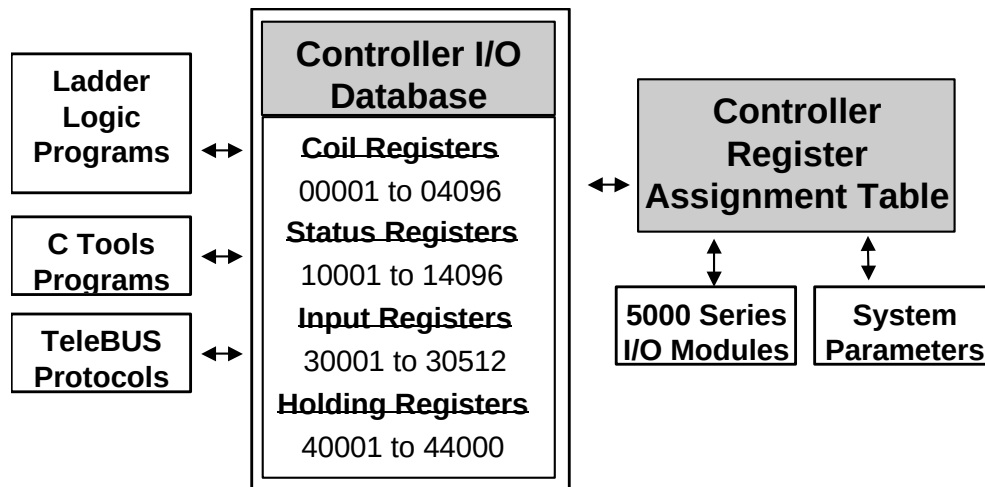


Figure 7: SCADAPack 100 – 256K I/O Database Block Diagram

User-assigned registers are initialized to the default hardware state or system parameter when the controller is reset. Assigned output registers do not maintain their values during power failures. Assigned output registers do retain their values during application program loading.

General-purpose registers retain their values during power failures and application program loading. The values change only when written by an application program or a communication protocol.

The TeleBUS communication protocols provide a standard communication interface to the controller. The TeleBUS protocols are compatible with the widely used Modbus RTU and ASCII protocols. They provide full access to the I/O database in the controller.

2.5.3.1 I/O Database register types

The I/O database is divided into four types of I/O registers. Each of these types is initially configured as general purpose registers by the controller.

2.5.3.1.1 Coil Registers

Coil registers are single bit registers located in the digital output section of the I/O database. Coil, or digital output, database registers may be assigned to 5000 Series digital output modules or SCADAPack I/O modules through the Register Assignment. Coil registers may also be assigned to controller on board digital outputs and to system configuration modules.

There are 4096 coil registers numbered 00001 to 04096. Ladder logic programs, C language programs, and the TeleBUS protocols can read from and write to these registers.

2.5.3.1.2 Status Registers

Status registers are single bit registers located in the digital input section of the I/O database. Status, or digital input, database registers may be assigned to 5000 Series digital input modules or SCADAPack I/O modules through the Register Assignment. Status registers may also be assigned to controller on board digital inputs and to system diagnostic modules.

There are 4096 status registers numbered 10001 to 14096. Ladder logic programs and the TeleBUS protocols can only read from these registers. C language programs can read data from and write data to these registers.

2.5.3.1.3 Input Registers

Input registers are 16 bit registers located in the analog input section of the I/O database. Input database registers may be assigned to 5000 Series analog input modules or SCADAPack I/O modules through the Register Assignment. Input registers may also be assigned to controller internal analog inputs and to system diagnostic modules.

The I/O database for the SCADAPack 100: 256K controller has 512 input registers numbered 30001 to 30512. Ladder logic programs and the TeleBUS protocols can only read from these registers. C language programs can read data from and write data to these registers.

2.5.3.1.4 Holding Registers

Holding registers are 16 bit registers located in the analog output section of the I/O database. Holding, or analog output, database registers may be assigned to 5000 Series analog output modules or SCADAPack analog output modules through the Register Assignment. Holding registers may also be assigned to system diagnostic and configuration modules.

The I/O database for the SCADAPack 100: 1924K controller has 4000 holding registers numbered 40001 to 44000. Ladder logic programs, C language programs, and the TeleBUS protocols can read from and write to these registers.

3 TelePACE Program Development

3.1 Introduction

A Ladder Logic program may be developed in any sequence. Best results, particularly for new users, will be obtained using the following sequence:

Configure **TelePACE**.

Create the ladder program.

Create the Register Assignment.

Select Outputs-On-Stop settings.

Define serial port settings.

Create Ladder Logic Program.

Initialize controller.

Write program to the controller.

Run and test program.

These steps are described in more detail in the sections that follow.

3.2 Configuration of the TelePACE Program

The configuration of the **TelePACE** program involves setting the serial port parameters for the target controller and selecting options to customize the Ladder Editor environment.

3.2.1 PC Communication Settings

Configure **TelePACE** communications to satisfy the requirements of the communication media between the **TelePACE** program and the target controller. Refer to the **PC Communication Settings** in the **TelePACE Program Reference**.

For **SCADAPack**, **SCADAPack Light** and **SCADAPack Plus** controllers, com3 is supported only when the **SCADAPack 5601 or 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the **TelePACE** program.

3.2.2 Options

Set the Ladder Editor environment options to suit the display resolution and color capability of your computer. From the **Options** menu select **Screen Font** to change the Editor screen font and **Colors** to change the Ladder Logic display color.

The format of floating point numbers may be changed using the **Floating-Point Settings** option.

The **Tool Bar**, **Title Bar** and the **Status Bar** may be removed from the Editor Display by selecting or de-selecting the option in the **Option** menu.

Select tag names display options to meet your personal preference. The selections are **Single Tag Names**, **Double Tag Names**, **Tag and Address** and **Numeric Address**.

In most cases it is recommended that the **Allow Multiple Coils** and **Warning Messages** be selected.

3.3 Initializing the Controller

The controller should be initialized before a new program is loaded into the controller. The **Initialize Controller** dialog appears when **Initialize** is selected in the **Controller** menu. Ladder Logic programs C programs and the Register Assignment may be erased from this dialog.

WARNING: If the controller is initialized, using the Initialize command in the Controller menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller. Element Configuration may be used for the following functions:

- DIAL – Control Dial-Up Modem;
- INIM – Initialize Dial-Up Modem;
- MSTR – Master Message;
- SLP – Put Controller into Sleep Mode;
- HART – Protocol Driver;
- DLOG – Data Logger;
- FLOW – Flow Accumulation;
- TOTL – Flow Totalizer;
- PIDA – PID controller for analog output; and
- PIDD – PID controller for digital output.

Refer to the **Ladder Logic Function Reference** for a description of these functions.

3.4 Define Register Assignment

All I/O hardware that is used by the controller must be assigned to I/O database registers in order for the I/O points to be used by the ladder program. Ladder logic programs may read data from, or write data to, the I/O hardware through user-assigned registers in the I/O database.

The Register Assignment assigns I/O database registers to user-assigned registers using I/O modules. An I/O Module can refer to an actual I/O hardware module (e.g. *5401 Digital Input Module*) or it may refer to a set of controller parameters, such as serial port settings.

The **Register Assignment Reference** section describes the purpose of each module and the register assignment requirements for the module.

Register assignments are stored in the user configured Register Assignment and are downloaded with the ladder logic application program.

3.5 Create Ladder Logic Program

3.5.1 Inserting Elements

To insert an element in the Ladder network, position the cursor at the network position where the element is to be inserted. Double click the left mouse button. The **Insert/Edit Network Element** dialog box pops up and an element or function can be selected to insert.

The **Insert/Edit Network Element** dialog box only displays the elements and functions that will fit from the cursor position.

Elements can be inserted using the right mouse button and selecting **Edit Element** from the drop down menu.

Elements can be inserted from the keyboard by positioning the cursor at the desired network position and pressing the Insert key.

3.5.2 Inserting Networks

To insert another network, select **Insert** from the **Edit** menu. The **Insert** dialog pops up. Selections are made in this dialog by moving the mouse pointer to the required radio button and clicking the left mouse button. Networks may be inserted before or after the current network.

3.5.3 Editing Elements

To edit an element in the Ladder network, position the cursor on the element to be edited. Double click the left mouse button. The **Insert/Edit Network Element** dialog box pops up and the element can be edited.

Elements can be edited using the right mouse button and selecting **Edit Element** from the drop down menu.

Elements can be edited from the keyboard by positioning the cursor at the desired element then pressing the Insert key.

3.5.4 Deleting Elements

To remove an element from a network, press the Delete key, or select **Delete** from the **Edit** menu. The element at the cursor position or selected elements, columns, rows or networks can be selected for deleting.

Selected elements can be deleted using the right mouse button and selecting **Delete** from the drop down menu.

3.5.5 Selecting Elements

To select multiple elements position the cursor on the first element or function of the selection and press the left mouse button. Hold the button down and drag the pointer to the last element or function. The elements selected will be highlighted in the same way as the cursor. Releasing the mouse button does not change the highlighting.

Individual disconnected elements may also be selected and copied to the clipboard. To select disconnected elements position the cursor on the first element. Hold the keyboard shift key down. Using the left mouse button, click on elements to be copied to the clipboard.

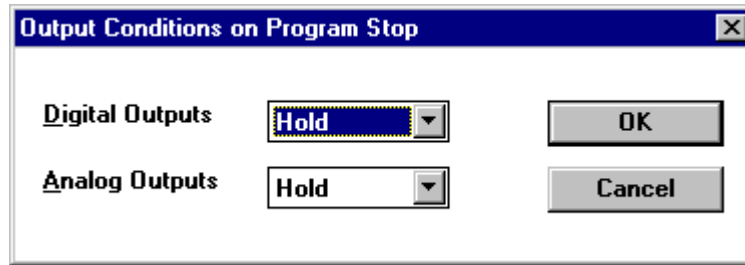
To select elements or functions with the keyboard move the cursor to the first element or function. Hold the shift key down while using the directional arrow keys to move the cursor to the last element or function. The selected elements are highlighted.

3.6 Setting Outputs On Stop

Select **Outputs on Stop** from the **Controller** menu and the **Output Conditions on Program Stop** dialog appears. This dialog controls the state of the controller analog and digital outputs when the ladder logic program is stopped.

The state of the digital outputs may be set to **hold** their last value or to turn **off** when the ladder program is stopped.

The state of the analog outputs may be set to **hold** their last value or to go to **Zero** when the ladder program is stopped.



3.7 Controller Serial Port Settings

Select the **Serial Ports** command from the **Controller** menu to configure the controller serial ports. The **Controller Serial Ports Settings** dialog box pops up when the Serial Ports command is selected.

For **SCADAPack**, **SCADAPack Light**, and **SCADAPack Plus** controllers, Com3 is supported only when the **SCADAPack 5601 or 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the **TelePACE** program.

3.8 Write program to controller

The **Write to Controller** command in the **Communications** menu writes the Ladder Logic program to the controller. The program replaces the program in the controller.

If the program in the controller is executing a dialog box will request whether to stop execution of the new program when the write is complete or to continue execution of the new program after the write is complete.

WARNING: Exercise caution when selecting the **Continue** option. The program will execute with the changes you make, even if the changes are not complete. This may cause undesired operation. Select **Stop** if you are making multiple changes.

3.9 Monitor Program On Line

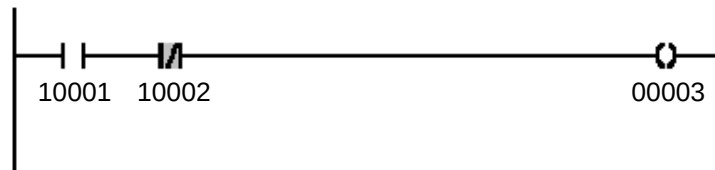
The **Monitor On Line** selection in the **Activity** menu enables the real time monitoring of a program executing in a controller. The editor shows the power flow through the network on the screen. No changes can be made to the program in monitor mode. The **Register Editor** window is displayed when **Monitor On Line** is selected.

3.9.1 Contact Monitoring

It is often necessary, while in Monitor On line mode, to determine whether a contact would pass power if power were supplied to it. This feature is extremely useful when testing the operation of ladder logic programs.

In Monitor On line mode, **TelePACE** shows the power flow through the network. It also colors contacts that are not powered to show how power would flow if they were.

The following diagram illustrates how power flow information is displayed in Monitor On line mode:



In the above example, normally open contact 10001 and normally closed contact 10002, are not energized. The shaded background, in contact 10002, indicates power would flow through the contact if the input side were powered.

- The background of a normally open contact is colored when it is energized and its input is not powered.
- The background of a normally closed contact is colored when it is not energized and its input is not powered.

The background is only displayed on contacts if the input side of the contact is not powered.

The color used for contact monitoring can be changed using the *Colors* command.

3.10 Edit Program On Line

The **Edit On Line** selection in the **Activity** menu is used to edit a ladder logic program that is executing in the controller. All editing commands affect the program in the controller and the program in the **TelePACE Ladder Network Editor**.

3.11 Force Registers

The Register Editor dialog allows the user to modify registers in the memory of the controller. This dialog is only available in the on-line mode. Data modified by the dialog is stored only in the memory of the controller. It does not form part of the ladder logic program. Refer to the **Program Reference** section for detailed information on the Register Editor.

3.12 Preventing Unauthorized Changes

A **TeleSAFE Micro16** or **SCADAPack** controller can be locked to prevent unauthorized changes. A password is required to unlock the controller to make changes.

The controller will reject commands sent to the unit when it is locked. A controller that is unlocked operates without restriction. Three levels of locks are provided.

- Locking the programming commands prevents modifying or viewing the program in the controller. Communication protocols can read and write the I/O database.
- Locking programming and database write a command prevent modifying or viewing the program; and prevents writing to the I/O database. Communication protocols can read data from the I/O database, but cannot modify any data.
- Locking programming and database commands prevents modifying or viewing the program and prevents reading and writing the I/O database. Communication protocols cannot read or write the I/O database.

Refer to the sections Lock Controller, Unlock Controller, Override Controller Lock and Show Lock Status for details on using this feature.

4 TelePACE Program Reference

The TelePACE program is a powerful programming and monitoring tool that enables the user to create, edit, document and save Ladder Logic programs for the controller. TelePACE also loads and runs C programs in the controller. Ladder Logic programs can be written to a controller or read from a controller using the TelePACE Ladder Logic Editor. The on-line monitoring and editing features enable efficient debugging of program and control problems.

The TelePACE program can be used locally – connected directly to the controller – or remotely – connected to the controller through a communication network or dial up modem. TelePACE supports connections to remote controllers using SCADA Server and Modbus/IP. All features of TelePACE are available locally and remotely.

This section of the manual describes each of the TelePACE program commands available from the TelePACE menu bar. The user is encouraged to read through the following sections to gain an understanding of the many programming and monitoring features available.

The menu bar selections and the functions available for each selection are described in the following sections.

4.1 File Menu

The File menu contains commands to create, open and save Ladder Logic files; utility commands to import and export tag names; and commands to configure printer options and print Ladder Logic programs.

4.1.1 New

The **New** command starts a new TelePACE Ladder Logic program. The comment editor window and the ladder editor window are cleared and the network display shows network one of one. The TelePACE Ladder Network Editor retains all information from the previous editor settings with the exception that the I/O Assignments are defaulted to inputs for all channels.

If the current file has not been saved, the editor warns you to save the current file. You may save the changes, discard the changes or cancel the New command.

4.1.2 Open

The **Open** command loads a ladder logic program from disk into the editor. The file replaces the existing file in the editor. If the current file has not been saved, the editor warns you to save the current file. You may save the changes, discard the changes or cancel the Open command.

The Open command will cause the Open File dialog box to appear. This dialog box is used to select the file to Open. Ladder logic files normally end with the file extension LAD. The dialog box displays these files by default. Press **Ctrl-O** to execute the **Open** command.

4.1.3 Save

The **Save** command stores the current file on disk. If the current file has not yet been named, the Save As File dialog box will appear. The file name and working directory can be selected from this dialog box. The title bar at the top of the editor window shows the current file name. Press **Ctrl-S** to execute the **Save** command.

4.1.4 Save As

The **Save As** command stores a file with a different name, or location, than the current file. The Save As File dialog box is used to specify the file name and directory in which to save the file. Ladder

logic file names normally end with the extension LAD. The editor uses the extension LAD automatically, if you do not type it. Press **Ctrl-A** to execute **Save As** command.

4.1.5 Page Setup

The Page Setup dialog controls the appearance of printed documentation. The items to be documented are selected with the **Select Print Items** command.

The **Margins** section defines the printing margins for the page. Use the margins to adjust the look of the printed output. The **Measurement Units** box selects the units of measurement for the margins.

The **Page Headings** section selects headers for each page of printed documentation.

- Check the **Page Header** box to print headings on all pages. The heading includes the file name, a page description, and the items described below.
- Check the **Date and Time** box to print the time and date on each page.
- Check the **Page Numbers** box to print page numbers on each page. Page numbers normally begin with page 1. You can specify the starting page.

4.1.6 Print

The Print command is used to print documentation on your printer. The Print dialog box is used to select the printer and printer options such as page size and orientation for the print job.

The items to be documented are selected with the **Select Print Items** command. The page layout and margins are selected with the **Page Setup** command.

4.1.7 Select Print Items

The **Select Print Items** command is used to select the program documentation and print topics to be printed. Through the selections in the Select Print Items dialog box, shown below, a customized program print-out can be created.

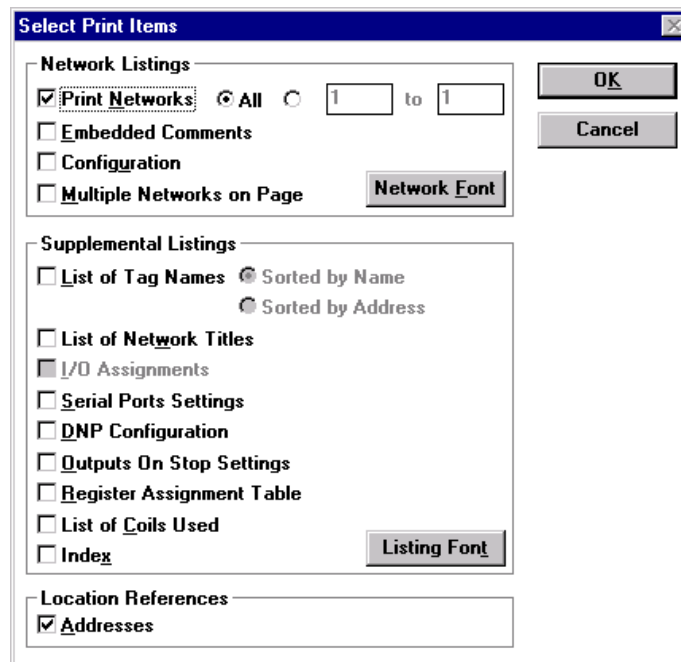


Figure 8: Select Print Items Dialog Box

The **Network Listings** section contains selections for defining the networks and comments to be printed.

The **Print Networks** selection defines the networks to print. All networks in the ladder program or a range of networks may be selected for printing.

The **Print Embedded Comments** selection enables network comments to be printed with the networks.

The **Configuration** selection enables the printing of the element configuration parameters for elements that use the element configuration dialog.

Select **Multiple Networks on Page** to allow multiple unbroken, networks and embedded comments to be printed on each page.

Select **Network Font** to modify the font used in printing the embedded comments and network functions, addresses, tag names and network titles.

The **Supplemental Listings** section contains selections for detailed program information that can be printed. Each item selected in the supplemental listings section is printed as a separate section with page header, if selected in page setup.

The **List of Tag Names** selection adds a complete list of all tag names used to the print out. Tag names can be sorted by their name or address.

A list of all network titles and network numbers used in the program are printed when the **List of Network Titles** is selected.

Selecting the **DNP Configuration** prints the complete configuration for the DNP protocol if it is used in the application.

Selecting the **Serial Port Settings** selection prints complete serial port settings for each serial port.

The **Outputs On Stop Settings** for the analog and digital outputs are printed when this item is selected.

Register Assignment selection enables the complete register assignment used in the application to be printed.

A list of destination coils and tag names in use is added to the print-out when **List of Coils Used** is selected.

An index of all print topics selected is added to the print-out by selecting **Index**.

The font used for printing the supplemental listings is can be modified by selecting **Listing Font**.

Selecting **Addresses** in the **Location References** section adds a complete cross-reference of all addresses used in the ladder logic program. The addresses are cross-referenced to Network / Row / Column locations for all occurrences of each address.

- Click **OK** to accept the selected print items and close the dialog box.
- Click **Cancel** to discard changes.

4.1.8 Quick File List

The quick file list area displays the names of the four most recently used Ladder Logic files. Select a file from this list by clicking the left mouse button on the file name or by pressing the number of the file.

4.1.9 Exit

The Exit command terminates the TelePACE program. The editor warns you to save the current file, if it changed since the last save. You may save the changes, throw away the changes or cancel the Exit command. Press **Ctrl-Q** to execute the **Exit** command.

Selecting Close from the system menu also terminates the program. The editor prompts you to save changes to your program.

4.2 Edit Menu

The Edit menu commands are used to insert, delete and modify the logic elements, function blocks, comments and networks used in the Ladder Logic program. Clicking the left mouse button on the desired command, or, from the keyboard by pressing the underlined letter accesses all Edit menu commands.

4.2.1 Undo

The Undo command reverses the last change made in the Ladder editor or Comment editor. This is a single level command and only the last change made before selecting Undo is reversible. Selecting Undo again reverses the action of the last Undo.

- Press **Ctrl-Z** to execute the **Undo** command from the keyboard.
- Right click your mouse and select **Undo** from the command list.

4.2.2 Cut Selected

The Cut Selected command removes the selected elements in the Ladder Editor or the Comment editor and puts them on the Clipboard. Items on the clipboard remain there until they are replaced by other selected elements.

- Press **Ctrl-X** to execute the **Cut Selected** command from the keyboard.
- Right click your mouse and select **Cut Selected** from the command list.

4.2.3 Copy Selected

The Copy Selected command copies the selected elements in the Ladder editor to the clipboard. Items on the clipboard remain there until they are replaced by other selected elements.

- Press **Ctrl-C** to execute the **Copy Selected** command from the keyboard.
- Right click your mouse and select **Copy Selected** from the command list.

4.2.4 Copy Networks

The Copy Networks command copies a network or block of networks to the clipboard. Comments for the network are also copied to the clipboard.

Networks to copy are selected from the Copy Networks dialog box. Continuous networks from the First Network selection to the Last Network selection may be copied to the clipboard.

4.2.5 Cut Networks

The Cut Networks command cuts a network or block of networks to the clipboard. Comments for the network are also cut to the clipboard.

Networks to cut are selected from the Cut Networks dialog box. Continuous networks from the First Network selection to the Last Network selection may be cut to the clipboard.

4.2.6 Paste

The Paste command is used to paste items on the clipboard to the current cursor position. The paste command in the Edit menu box is grayed out when there are no items on the clipboard or the cursor is in the wrong pane for the clipboard items. For example text cannot be pasted into the Ladder Editor pane and network elements cannot be pasted into the Comment Editor pane.

Text cannot be pasted into the Comment Editor if the clipboard text will cause the Comment Editor to exceed 16 Kbytes of text for the network. In the Ladder Editor pane if there is not enough room between the cursor and the edge of the network to paste the selected items an error message is displayed.

When a network or block of networks are pasted from the clipboard the Paste Block of Networks dialog box appears. This dialog box is used to determine if the block of networks is to be pasted before or after the current network and whether network titles and comments are pasted with the block of networks.

- Press **Ctrl-V** to execute the **Paste** command from the keyboard.
- Right click your mouse and select **Paste** from the command list.

4.2.7 Insert

The Insert command is used to insert changes to the current ladder logic network. When the Insert command is selected the Insert dialog box pops up. Items that can be inserted into the ladder logic program are displayed in the dialog box. Items in the dialog are inaccessible (grayed) if they cannot be used from the current cursor position. Items to be inserted are selected by selecting the radio button associated with each selection.

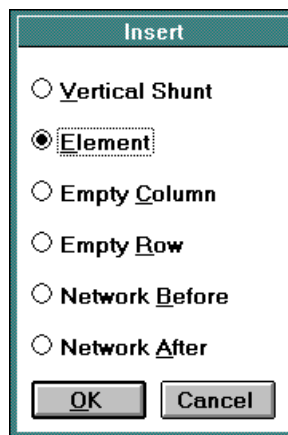


Figure 9: Insert Dialog Box

4.2.7.1 Insert Vertical Shunt

The insert Vertical Shunt selection inserts a vertical shunt on the right side of the element highlighted by the cursor. Where a multiple cell element is selected, the vertical shunt is inserted in the cursor position shown on the status bar. This command is not available in the Comment Editor pane.

- Press **F5** to insert a vertical shunt.
- Right click your mouse and select **Insert Vertical Shunt** from the command list.

4.2.7.2 Insert Element

The insert Element selection inserts or edits an element at the current cursor position. A dialog box appears displaying the Ladder Editor function block list. This command is not available in the Comment Editor pane.

- Press **Insert** to execute this function from the keyboard.
- Right click your mouse and select **Insert/Edit Element** from the command list.

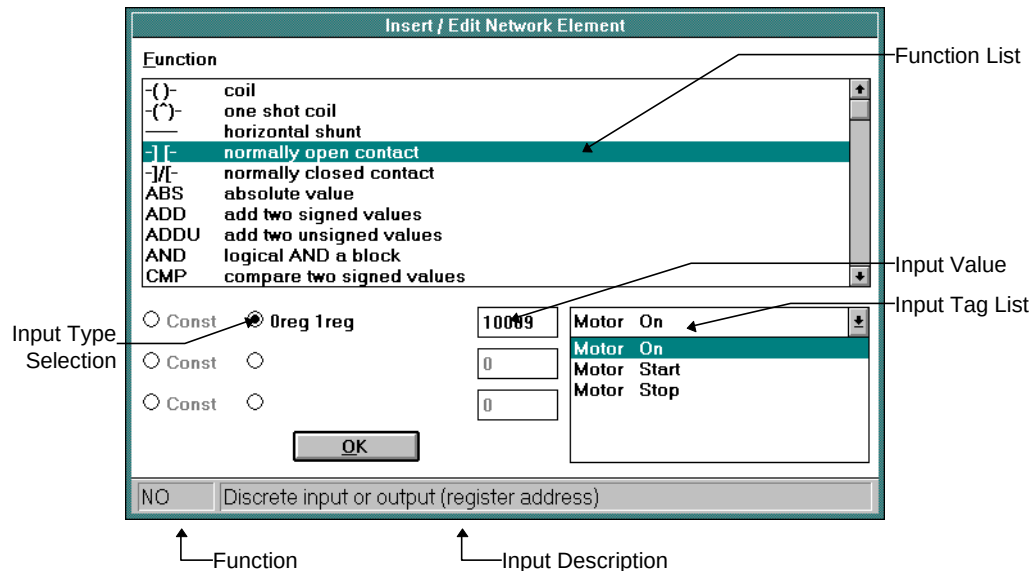


Figure 10: Insert / Edit Network Element Dialog Box

The function block list shows the function blocks that fit in the current position. If the cursor is over an existing function block, the list shows only function blocks of the same size. If the cursor is over a blank space, the list shows all function blocks that fit. Function blocks occupy one, two, or three cells in a network. The left side of the status bar at the bottom of the dialog shows the selected function block name.

Selecting a function activates the input boxes at the bottom of the dialog. There are one, two, or three inputs depending on the function. A description of the input is shown on the status bar, when an input is selected for editing.

Some function blocks allow either addresses or constants as inputs. The radio buttons on the left of the input box indicate if the value in the input box is an address or a constant. Click the left radio button to input a constant. Click the right radio button to input an address of the type displayed. The radio button defaults to the address type selection.

There are two ways to edit an input:

When an address type is selected the register address can be typed into the input edit box. By default the first register address, of the type required, is displayed in the edit box. Some functions require a range of registers to define an input. In these cases the first and last register of the range are displayed. Edit the displayed address to meet the requirements of your application. The upper range limit is recalculated automatically as soon as any valid start register is entered.

When a constant type is selected the constant value can be typed into the input edit box. By default the number of the first register address, of the type required, is displayed in the edit box. Edit the displayed constant to meet the requirements of your application.

A tag name can be selected from the tag selection list. Click the tag list and choose the tag for the address or constant desired. The tag selection list will show only tags that are allowed on the selected input.

4.2.7.3 Address Types

The text beside the address button indicates the register types that are valid for the current function.

Type	Description	Registers
0reg	Discrete output register address	Single bit register
0blk	Discrete output register block address	16 single bit registers
1reg	Discrete input register address	Single bit register
1blk	Discrete input register block address	16 single bit registers
3reg	Analog input register address	Sixteen bit register
4reg	Analog output register address	Sixteen bit register

Table 1: Address Types

Note: Discrete input and output register blocks (blk types) must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033, etc.

4.2.7.4 Empty Column

The Empty Column selection inserts an empty column to the left of the current cursor position. This selection is greyed if an empty column already exists to the left of the cursor or if there is no room for an empty column in the network. This command is not available in the Comment Editor pane.

4.2.7.5 Empty Row

The Empty Row selection inserts an empty row above the current cursor position. This selection is greyed if an empty row already exists to the left of the cursor or if there is no room for an empty row in the network. This command is not available in the Comment Editor pane.

4.2.7.6 Network Before

The Network Before selection inserts an empty network before the current network. The cursor moves to column 1 Row 1 of the new network. This command is available in the Comment Editor pane.

4.2.7.7 Network After

The Network After selection inserts an empty network after the current network. The cursor moves to column 1 Row 1 of the new network. This command is available in the Comment Editor pane.

4.2.8 Delete

The Delete command removes items from the current Ladder Editor network. Press **Delete** to execute this function. A dialog box showing items that can be deleted appears. Items in the dialog box may be inaccessible (grayed) if they do not apply to the selected elements. The delete command is grayed out in the Edit menu when the cursor is in the Comment Editor pane. Items to delete are selected by clicking on the radio button associated with each selection.

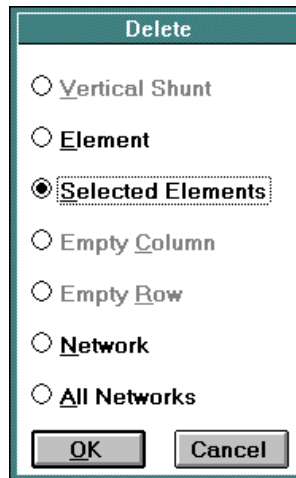


Figure 11: Delete Dialog Box

4.2.8.1 Vertical Shunt

The Vertical Shunt selection deletes the vertical shunt on the element highlighted by the cursor. This command is greyed if there is no vertical shunt on the element.

4.2.8.2 Element

The Element selection deletes the element at the current cursor position. This will delete a single element only, even if multiple elements are selected.

4.2.8.3 Selected Elements

The Selected Element selection deletes the elements highlighted in the network. Elements are highlighted by dragging the network cursor to highlight more than a single element.

4.2.8.4 Empty Column

The Empty Column selection deletes a column that contains no elements, and moves the elements in the columns to the right of it one column to the left.

4.2.8.5 Empty Row

The Empty Row selection deletes a row that contains no elements, and moves the elements in the rows below it up one row.

4.2.8.6 Network

The delete Network command deletes the current network and all the elements in it. Deleting a network does not remove the Tag names from the program.

4.2.8.7 Delete All Networks

The delete All Networks command deletes all ladder logic networks in the current file. You are asked to confirm that you wish to delete the networks. Deleting all networks does not change tag names, I/O assignments or the file name.

4.2.9 Toggle Vertical Shunt

The Toggle Vertical Shunt command toggles (inserts or deletes) a vertical shunt on the right side of the element highlighted by the cursor. The F5 key also performs this command.

4.2.10 Tag Names

The Tag Names selection is used to edit names that can be used in place of addresses and constants. Tag names may be used once only for a constant or an address. Numbers can occur once for a constant and once for an address.

- Tag names must conform to the .CSV file format naming convention.
- Tag names may be up to 16 characters long and up to 2500 tag names can be created.
- The Edit Tag Names dialog box appears allowing the editing of Tag names.

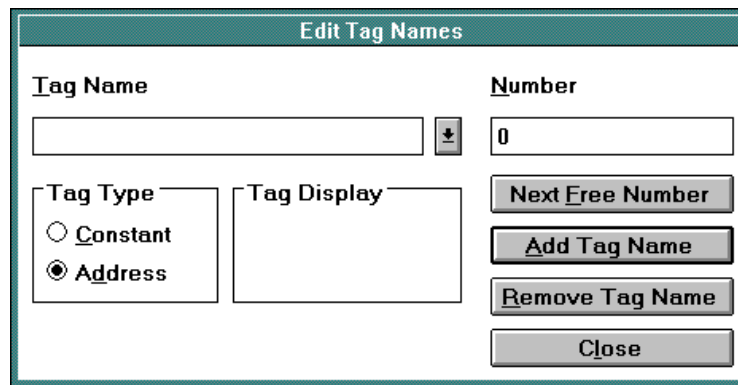


Figure 12: Edit Tag Names Dialog Box

The **Edit Tag Names** edit box is used to enter tag names. The drop down menu to the right of the box displays tag names already created.

The **Number** edit box is used to enter the number associated with the tag.

The **Tag Type** radio buttons select the type of tag. A Tag Type can be an address or a constant.

The **Tag Display** edit box displays the Tag Name entered in the Tag Name edit box. Tag names are a maximum of sixteen characters with eight characters on each line. The tag display box shows the tag name as it will be displayed in the Ladder Editor.

The **Next Free Number** button will cause the value in the Number edit box to be replaced by the next unused address or number of the Tag Type that is not in use.

The **Add Tag Name** button adds the tag to the list of tag names. An error dialog will appear if the tag name exists with a different type or number, or if a number exists with the same type and a different name.

The **Remove Tag Name** button removes the selected tag from the list of tag names. An error dialog will appear if the name is in the list but the other data does not match the type button or number box.

The **Close** button closes the Edit Tag Names dialog.

4.2.11 Erase All Tags

The Erase All Tags command is used to erase all tag names in a TelePACE Ladder Logic application program. This command can be used before importing tags from another program or a CSV file to eliminate errors when there are conflicts between existing tag names and imported tag names.

The command opens the following message box:

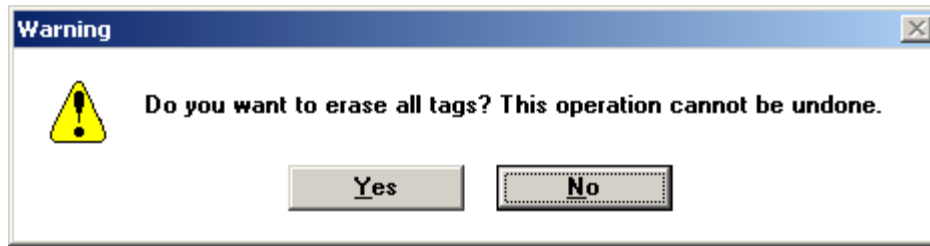


Figure 13: Erase All Tags Dialog Box

- Click on **Yes** to erase all tags.
- Click on **No** to abort the command.

Note: Erasing all tags cannot be undone. If you are unsure you should save your file before executing this command.

4.2.12 Export Tag Names

The Export Tag Names command is used to write the program tag names into a CSV (comma separated value) file. A CSV file can be opened, and edited in most spreadsheet programs. Note that the modified tag names must conform to the .CSV format.

The Export Tag Names to File dialog box is used to specify the file name and directory of the export file. Tag name files normally end with the extension CSV. The editor uses the extension CSV automatically, if you do not type it.

The CSV file contains a title line followed by multiple data lines containing the type_specifier, address and tag_name fields. Commas separate these fields. The type_specifier is A for an address tag, and C for a constant tag. The tag name can be enclosed in quotes. If the tag name contains a comma, it must be enclosed in quotes.

Note that if a type and register value is specified, a corresponding tag name must be entered otherwise an error will result.

Title line: Type, Value, Tag
Data line: type_specifier, address, tag_name

Note: A blank tag name is not allowed. If a type and register value is specified, a corresponding tag name must be entered otherwise an error will result.

4.2.13 Import Tag Names

The Import Tag Names command is used to read tag names into the editor. Tag names may be read from another ladder logic program, or from a CSV file. New tags are added to the existing tag list.

The *Import Tag Names From File* dialog box is used to select the file containing the tag names. Files with the extensions LAD and CSV are displayed by default.

As the tags are imported, they are checked for errors. If a tag is imported that that already exists in the program an error will occur. If an error occurs, you may continue to process the input file to look for more errors, or you may abort the import. If an error occurs, no tags are imported.

A CSV file consists of a title line followed by multiple data lines containing the type_specifier, address and tag_name fields. Commas separate these fields. The type_specifier is A for an address tag, and C for a constant tag. The tag name can be enclosed in quotes. If the tag name contains a comma, it must be enclosed in quotes.

Title line: Type, Value, Tag
Data line: type_specifier, address, tag_name

4.2.14 Network Title

The edit Network Title selection is used to edit the title of the current network. When selected the Edit Network Title dialog box appears and the title can be edited. The title may be a maximum of 28 characters long.

4.2.15 Element Configuration

The element configuration command edits initial values for 40000 series registers used by the MSTR, DIAL, INIM, HART, FLOW, SCAL, TOTL, PID, PIDA and SLP elements. The register values are written to the controller with the program. The register values may be modified later by other elements of the program.

Right click your mouse and select **Element Configuration** from the command list.

WARNING

If the controller is initialized, using the **Initialize** command in the **Controller** menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.

Any change made to an element configuration in the Edit On Line mode will be downloaded to the controller immediately.

4.2.16 Registers

The Registers command replaces, and enhances, the Monitor List command from previous versions of TelePACE. The Registers command allows the online monitoring and control and the offline editing of registers used in a TelePACE application. These registers may be saved in a single group, or in multiple groups for monitoring. Individual registers may be selected for on-line and off-line editing of their data.

When the Registers command is selected the **Register Editor** is opened. The Register Editor is used to add registers to the Register list; edit registers in the list and to display register data.

All registers in the Register list are automatically added to the Register data file see the **Register Data File** section for complete details on this file. Through using the Register Editor and the associated Register Data File users are able to easily:

Save a list, or lists, of registers for monitoring and control that is specific for each TelePACE application program.

Preload registers used in a TelePACE application with data.

Save a 'snap shot' of data in registers used in a TelePACE application.

Create meaningful groups of registers for monitoring and control.

4.2.16.1 Register Editor

The Register Editor displays registers and allows the editing of register content. The dialog is divided into three main areas. The **Groups** section allows the user to create and manage groups of registers. The **Register List** section displays the registers, and register attributes, for each register in a group. The **Update Rate** section allows the user to control the refresh frequency for the registers in a displayed group.

The Register Editor may be opened in the off-line and on-line editing modes. When opened the Register Editor is displayed on top of the Ladder Editor. Ladder Logic cannot be modified while the Register Editor is open. The Register Editor dialog is opened automatically in the Monitor on-line mode.

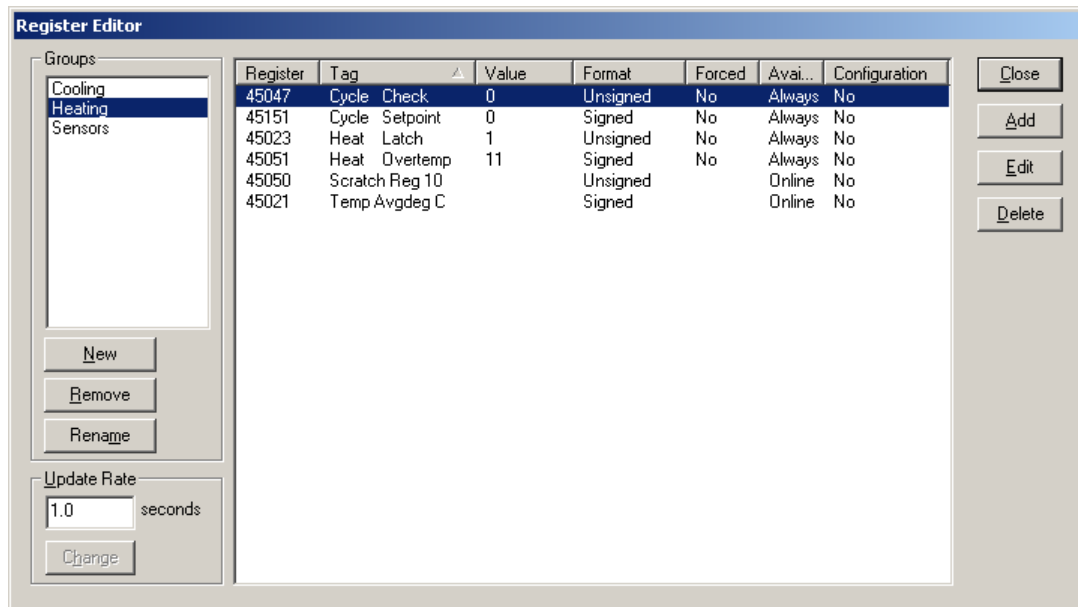


Figure 14: Register Editor Dialog Box

4.2.16.1.1 Groups

Register groups allows users to manage how registers are viewed in the Register Editor. Registers that are used in various parts of an application can be grouped together to provide an easy to view, and edit, list of registers. When in the on-line mode, only the registers in the current group are updated. This improves response time as large groups of registers can take a long time to update. Registers may be assigned to more than one group.

The **Groups** list box displays the register groups that have been created. When a group is selected the registers in the group are displayed in the Registers window. The group that was last selected the previous time the Register Editor was opened is displayed. Only one group can be selected at a time.

The **New** button creates a new group. It opens the New Group dialog.

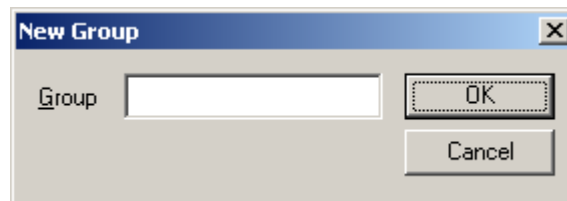


Figure 15: New Group Dialog Box

The **Group** edit box specifies the name of the new group. The group name may contain letters, numbers, and spaces. The group name is 1 to 16 characters long. Group names are case insensitive.

The **OK** button creates the new group and closes the dialog. If the group name already exists an error message is displayed and the dialog remains open.

The **Cancel** button closes the dialog.

The **Remove** button deletes a group from the group list. All the registers in the group are deleted. The button is disabled (greyed) if no group is selected.

The **Rename** button renames a group. It opens the Rename Group dialog.

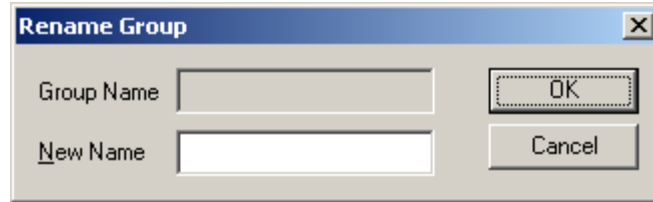


Figure 16: Rename Group Dialog Box

The **Group Name** field shows the current name of the group.

The **New Name** edit box specifies the new name of the group. The group name may contain letters, numbers, and spaces. The group name is 1 to 16 characters long. Group names are case insensitive.

The **OK** button renames the group and closes the dialog. If the group name already exists an error message is displayed and the dialog remains open.

The **Cancel** button closes the dialog.

4.2.16.1.2 Update Rate

The **Update Rate** edit box determines how often the register editor updates values when on-line. Valid values are 0.1 to 1000 seconds. The default value is 1 second. This value is used for all files opened by TelePACE and is saved when TelePACE is closed.

Note: The Update Rate must be set to a value that ensures the controller has enough time to execute C/C++ applications, including RealFLO flow computer. Setting the update rate to a very short time may affect the execution of C/C++ applications.

The **Change** button sets the current update rate. The modification in the **Update Rate** edit box will not take effect until the **Change** button is pressed. This button is disabled when the Register Editor is offline.

4.2.16.1.3 Register List

The Register list is displayed in the center of the Register Editor dialog. The Register List displays the registers for the selected group. Each register in the list has seven fields, or columns associated with it. The columns can be resized by dragging the edge of the column heading. The columns may be sorted by clicking the heading of each column.

The **Register** column displays the register number.

The **Tag** column displays the tag name associated with the register.

The **Value** column displays the value of the register in the selected **format**. Coil and status registers are shown as OFF or ON.

The **Format** column displays the register format. The register format is one of the following:

Register Format	Description
Unsigned	Displays a single register as an unsigned integer
Signed	Displays a single register as a signed integer

Register Format	Description
Unsigned Double	Displays two consecutive registers as an unsigned double integer
Signed Double	Displays two consecutive registers as a signed double integer
Floating Point	Displays two consecutive registers as a floating point number
Hexadecimal	Displays a single register as a hexadecimal number
Binary	Displays a single register as a binary number
ASCII	Displays a single register as two ASCII characters
Boolean	Displays coil and status registers which have no format

Table 2: Register Formats

The **Forced** column indicates if the register value is forced. Valid selections are **No** and **Forced**.

The **Available** column indicates if the register value is available always or only when TelePACE is on-line with the controller.

The **Configuration** column indicates if the register is part of an element configuration. It is recommended that registers that are part of an element configuration not be modified in the register editor. Use the element configuration feature instead.

The **Add** button adds registers to the selected group. It opens the **Add Registers Dialog**. This button is disabled if no group is selected.

The **Delete** button deletes the selected registers from the group. This button is disabled if no register is selected. This button is disabled if no group is selected.

The **Edit** button opens the **Edit Register Dialog** to edit the selected register value, format, force status, and register availability. This button is disabled if no register is selected or more than one register is selected. This button is disabled if no group is selected.

The **Close** button closes the dialog. This button is disabled in the monitor on-line mode.

The dialog is resizable. Click on the lower right corner and drag to change the size of the dialog.

4.2.16.2 Add Registers Dialog

The Add Registers dialog adds registers to the current group in the Register Editor dialog. The group name is displayed in the window title.

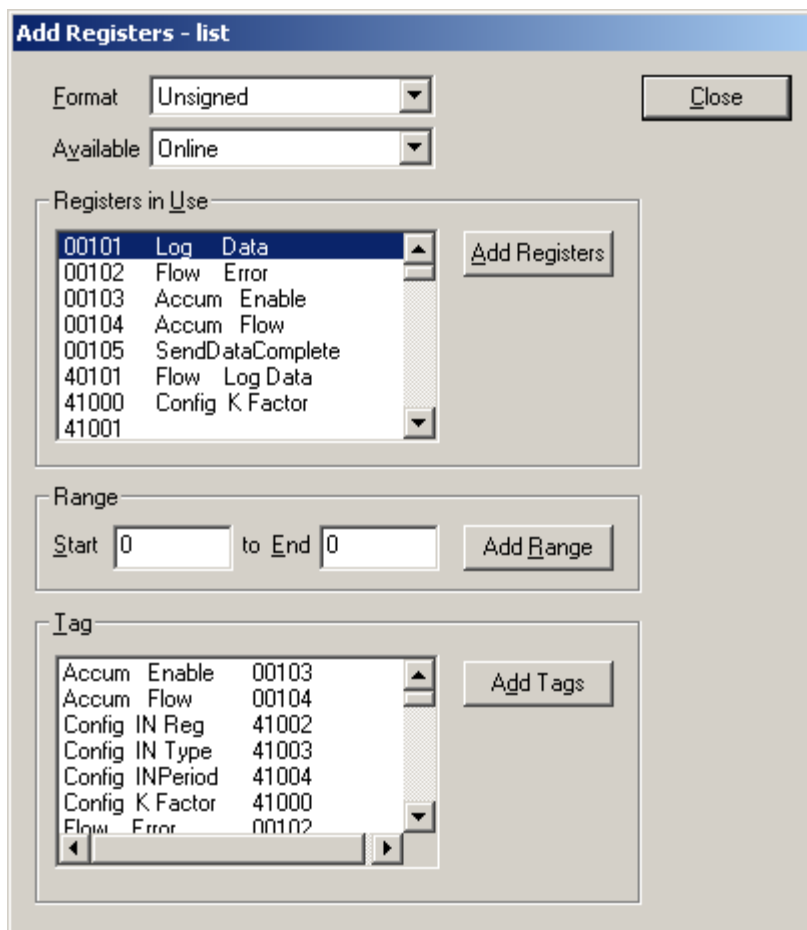


Figure 17: Add Registers – List Dialog Box

The **Format** drop-down list box specifies how the registers will be displayed. Select the format before adding registers.

Display	Description
Unsigned	Displays a single register as an unsigned integer
Signed	Displays a single register as a signed integer
Unsigned Double	Displays two consecutive registers as an unsigned double integer
Signed Double	Displays two consecutive registers as a signed double integer
Floating Point	Displays two consecutive registers as a floating point number
Hexadecimal	Displays a single register as a hexadecimal number
Binary	Displays a single register as a binary number
ASCII	Displays a single register as two ASCII characters
Boolean	Displays coil and status registers which have no format

Table 3: Register Display Properties

If a register or tag is used by element configuration its format cannot be changed. The register cannot be merged with adjacent registers or split if the format of an adjacent register is changed.

If a register or tag is already in the any register editor group but has a different format, the format of the register is changed in all groups.

The Unsigned Double, Signed Double, and Floating Point formats add two registers to the register editor group. The pair of registers is displayed as a single number.

If a single register is added that is part of an existing pair, the existing pair will become invalid and only new single register is displayed.

If a pair of registers is added that overlaps an existing single register, the single register and the following register will be made part of the register pair. If the following register is a part of the existing pair, that existing pair will become invalid.

If a pair of registers is added that overlaps one register of an existing pair, or two registers of two existing pairs, the new pair will be added and the existing overlapped pair will become invalid.

The **Available** list box specifies when register values can be viewed and edited. Valid values are Always and Online. The Always selection makes the register values available in both online and offline editing modes. The Online selection makes the register values available only in the online editing modes. The default is Online.

4.2.16.2.1 Registers In Use

The **Registers in Use** list box displays all registers that are used in the Ladder Logic program. One or more registers may be selected.

- To select a register, click on it.
- To select more than one register, hold the Ctrl key and click on each register to be added.
- To select a range of registers, click on the first register and hold down the Shift key while clicking on the last register. A second method is to hold the Shift key and use the Down Arrow key or Up Arrow key to select the registers.

The **Add Registers** button adds the selected registers to the register editor group. The registers are added in the format selected.

4.2.16.2.2 Range

The **Start** and **End** edit boxes specify a range of registers to add.

- To select a range, enter the Start and End registers.
- To select a single register, enter it in the Start edit box and enter 0 in the End edit box.

The **Add Range** button adds the selected registers to the register editor group. The registers are added in the format selected.

4.2.16.2.3 Tags

The **Tags** list box displays all tags and the corresponding register number that are defined in the Ladder Logic program. One or more tags may be selected.

- To select a tag, click on it.
- To select more than one tag, hold the Ctrl key and click on tags.
- To select a range of tags, click on the first tag and hold down the Shift key while clicking on the last tag or use Down Arrow key or Up Arrow key.

- To select a range of tags, click on the first tag and hold down the Shift key while clicking on the last tag. A second method is to hold the Shift key and use the Down Arrow key or Up Arrow key to select the tags.

The **Add Tags** button adds the selected tags to the register editor group. The tags are added in the format selected.

The **Close** button closes the dialog.

4.2.16.3 Edit Register Dialog

The **Edit Register** dialog modifies values for a register. There are two forms of the dialog. The following dialog is used for editing input registers (3xxxx type) and holding registers (4xxxx type).

Figure 18: Edit Register Dialog Box

The following dialog is used for editing coil registers (0xxxx type) and status registers (0xxxx type).

The image shows a dialog box titled "Edit Register". It contains several input fields and buttons. The fields are: "Register" with the value "01057", "Tag" with the value "Power UpReset", "Value" with a dropdown menu showing "OFF", "Format" with a dropdown menu showing "Boolean", "Force" with a dropdown menu showing "No", and "Available" with a dropdown menu showing "Always". To the right of these fields are three buttons: "OK", "Cancel", and "Apply". Below the fields is a section labeled "Warning" with a large empty text area.

Figure 19: Edit Register Dialog Box- Value Off

The **Register** field displays the register number. This field cannot be edited.

The **Tag** field displays the tag for the register. This field cannot be edited.

The **Value** edit box displays and allows the editing of register values.

For input and holding registers the **Value** edit box displays the current value of the register. Valid values depend on the format of the register. The edit box is disabled if the register is not available Always and the editor is in the off-line mode.

For coil and status registers the **Value** drop-down list box shows the value of the register. Valid values are ON and OFF. The list box is disabled if the register is not available Always and the editor is in the off-line mode.

The **Format** list box selects the format of the register. The list box is disabled and shows none for coil and status registers. The list box is disabled if the register is used for Element Configuration - the format of these registers cannot be changed. The following formats are available.

Format	Description
Unsigned	Displays a single register as an unsigned integer
Signed	Displays a single register as a signed integer
Unsigned Double	Displays two consecutive registers as an unsigned double integer
Signed Double	Displays two consecutive registers as a signed double integer
Floating Point	Displays two consecutive registers as a floating point number
Hexadecimal	Displays a single register as a hexadecimal number
Binary	Displays a single register as a binary number
ASCII	Displays a single register as two ASCII characters
Boolean	Displays coil and status registers which have no format

Table 4: Register Formats

The **Force** drop-down list box indicates if the value is forced. Valid values are No and Forced. The list box is disabled if the register is not available Always and the editor is in the off-line mode.

The **Available** drop-down list box indicates when register values can be viewed and edited. Valid values are Always and Online. The Always selection makes the register values available in both online and offline editing modes. The Online selection makes the register values available only in the online editing modes. The default is Always if the editor is off-line and Online if the editor is on-line.

The **Warning** is displayed if the register is used by element configuration. It is recommended Element Configuration be used to change the register value. The Element Configuration function performs additional error checking specific to the Ladder Logic element.

The **OK** button saves changes and closes the dialog. If the Editor is on-line, changes are written to the controller.

The **Cancel** button closes the dialog.

The **Apply** button writes the changes to the controller, when the editor is on-line. The Apply button is disabled if the editor is off-line.

4.2.16.4 Register Data File

The register data file is created, saved and opened with the Ladder Logic file. The register data file contains the information for any registers that have been added to the Register Editor and any registers used in an element configuration. The Register Data file will contain only the header record (see **Register Data File Format**) if there are no Register Editor entries or element configuration.

Users can easily edit the register data file providing a rapid way to add or remove registers or modify their values or format.

4.2.16.4.1 Register Data File Format

The register data file is saved as a CSV (comma separated value) format file. The file is saved in the format: **filename - Registers.csv**. Where *filename* is the TelePACE Ladder Logic file name.

Each line in the CSV file is one record. The following is an example of a Register Data file.

```
Register,Value,Format,Forced,Available,Groups
01057,0,BO,No,Always,Sensors
30002,9440,S,No,Online,Sensors
30004,312,S,No,Online,Sensors
40238,1826,U,No,Always,all
45021,606,U,No,Online,Heating;Cooling
45035,+23.6250,F,No,Always,Cooling
45041,32,B,No,Always,Cooling
48016,1074,U,No,Always,*Element
```

The first line in the file is a header record and it contains the names of each of the fields. The following table describes each of the fields in the file.

Field	Description
Register	The address of the register or first register of a register pair. Register pairs are used for unsigned double, signed double and floating point registers. Valid register addresses are: 00001 to 09999 Digital Output registers. 10001 to 19999 Digital Input registers. 30001 to 39999 Analog Input registers. 40001 to 49999 Analog Output registers.
Value	Value of the register or registers. Boolean: 0 to 1 Unsigned: 0 to 65535 Signed: -32768 to 32767 Unsigned double: 0 to $2^{32}-1$ Signed double: -2^{31} to $2^{31}-1$ Hexadecimal: same as unsigned Binary: same as unsigned ASCII: same as unsigned
Format	Display format for the register or registers. BO Boolean U Unsigned S Signed UD Unsigned double SD Signed double F Floating point H Hexadecimal B Binary A ASCII
Force	Indicates if register value is forced. No Indicates register is not forced Forced Indicates register is forced
Available	Indicates if register is available always or online only. Always Indicates that the register may be edited in either the Online or Off line mode. Online Indicates that the register may be edited in the Online mode only.
Groups	List of groups that include this register. If the register is included in more than one group each group is listed with semicolons separating the group names. The special value *Element is reserved for registers used by element configuration.

Table 5: Insert Table Caption Here

4.2.16.5 Editing the Register Data File

The CSV file may be edited using a text editor, such as Notepad or a spreadsheet program such as Excel. The following operations can be done easily with a text editor or spreadsheet: adding, inserting, replacing, removing or editing many registers, searching for registers, and sorting.

Care must be exercised when editing the CSV file. Improper changes may result in loss of element configuration or register data. The following list gives some suggestions when editing a CSV file.

- Make a backup copy of the file before changing it. The file can be restored if you make invalid changes.
- A register can occur only once in the CSV file. Check that a register is listed only once after copying lines.
- Do not change the header record.
- Do not change the order of the columns.
- Do not use empty fields.
- Avoid including the header record when sorting column.
- Avoid sorting only some of the columns.
- Avoid overlapped registers with different register formats.
- Group names may contain spaces, letters, and numbers only. Don't use other characters.
- Separate group names with semicolons.
- Group names are case insensitive the case does not matter.
- If registers containing part of an element configuration are deleted, the value for that register will be missing. This may cause unexpected operation of your ladder logic program.

4.3 Search Menu

Search Menu commands are used to locate networks, devices, addresses and tag names in the current Ladder Editor program. Addresses may be located, or located and replaced from the Search menu.

4.3.1 Next Network

Selecting the Next Network command from the Search Menu moves the Ladder Editor to the next Network in the program. The next network command is grayed out if the current network is the last network. Press **F8** to move the Ladder Editor to the next network.

4.3.2 Previous Network

Selecting the Previous Network command from the Search Menu moves the Ladder Editor to the previous Network in the program. The Previous Network command is grayed out if the current network is the first network. Press **F7** on the keyboard to move the Ladder Editor to the previous network.

4.3.3 Go To Network

The Go To Network command is used to move the Ladder Editor to a desired network. When the Go To Network command is selected the Go To Network dialog box pops up. The network titles or number to go to is entered in this dialog box. The Title box selects the title of the network. A drop down list shows multiple network titles in use. Networks that have no title are displayed showing the network number only. The Network box selects the number of the network. The Previous button moves backward through the list of networks. The Next button moves forwards through the list of networks.

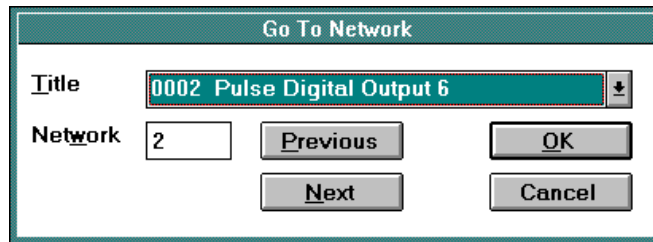


Figure 20: Go To Network Dialog Box

4.3.4 Find Address

The Find Address command searches the program for an address, or range of addresses, and moves to the first element containing any address in the range. The search proceeds down the rows from the present column, to column ten, row eight in the current network, then moves to the next network.

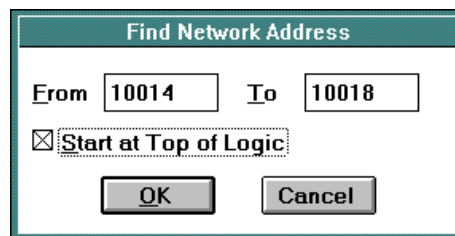


Figure 21: Find Network Address Dialog Box

The Find Network Address dialog box is used to enter the address, or range of addresses to find. The From box specifies the first address to locate. The initial value comes from the element at the current cursor position. The To box specifies the last address in a range. A value of 0 indicates that only the address in the From field is to be located. Checking the Start at Top of Logic box starts the search at the beginning of the program. The search starts at the current cursor position when this box is unchecked.

4.3.5 Find Device

The Find Device command is used to search the program for a device, and move to the first element containing that device. The search proceeds down rows from the present column to column ten row eight in the current network, then moves to the next network.

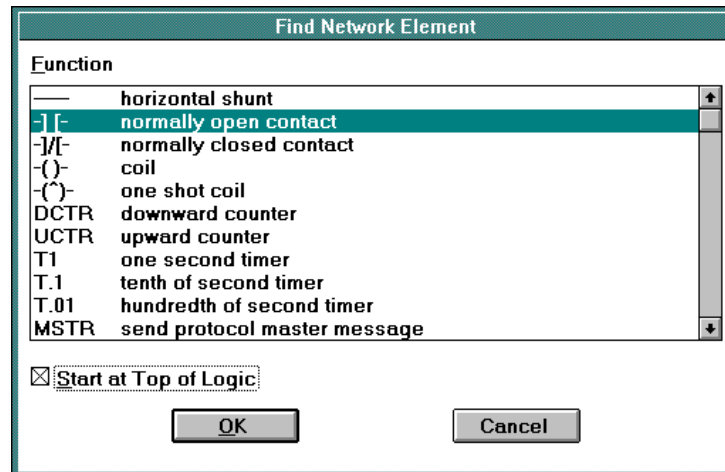


Figure 22: Find Network Element Dialog Box

The Find Network Element dialog box pops up when the Find Device command is selected. The Function list specifies the device to locate. The initial value comes from the element at the current cursor position. Checking the Start at Top of Logic box starts the search at the beginning of the program. Leaving this box unchecked starts the search at the current cursor position.

4.3.6 Find Tag Name

The Find Tag Name command is used to search the program for a tag name, and move to the first element containing the tag name. The search proceeds down rows from the present column to column ten, row eight in the current network, then moves to the next network.

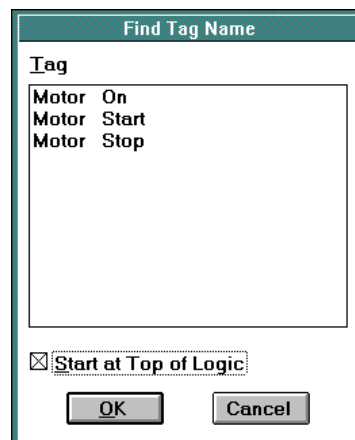


Figure 23: Find Tag Name Dialog Box

The Find Tag Name dialog box, displaying all the tag names used in the program, pops up when the Find Tag Name command is selected. Select a tag name in the Tag list by left clicking the mouse on the desired tag name. Checking the Start at Top of Logic box starts the search for the selected tag name at the beginning of the program. If left unchecked, the search starts at the current cursor position.

4.3.7 Repeat Last Find

The Repeat Last Find command repeats the last search performed. Press F3 to repeat the last find.

4.3.8 Replace Address

The Replace Address command is used to search the program for an address and then replace the address with another address. The search proceeds down rows from the present column to column ten, row eight in the current network, then moves to the next network.

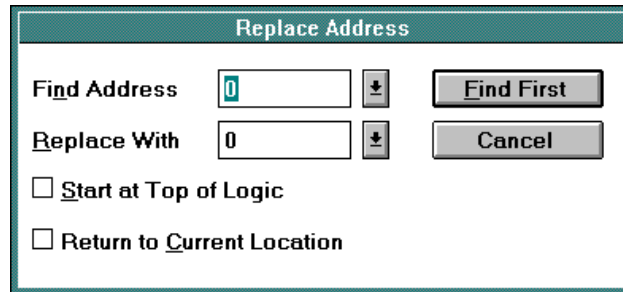


Figure 24: Replace Address Dialog Box

When the Replace Address command is selected the Replace Address dialog box pops up. The address to find and the replacement address are selected from this dialog box.

The Find Address box specifies the address to be replaced. The Replace With box specifies the replacement address.

Checking the Start at Top of Logic box starts the search at the beginning of the program. If left unchecked, the search starts at the current cursor position. Checking the Return to Current Location box returns the cursor to the current network after the command is executed. If left unchecked, the cursor is in the last program network after the command is executed.

The Find First button, when selected, will cause the Change Address dialog box to appear each time the search finds the address to be replaced. The With box allows changing of the replacement address. Selecting the Replace button replaces the address. The Find Next button continues the search to the next occurrence of the address to find.

4.3.9 Multiple Coils

Selecting the Multiple Coils command causes the Multiple Coils dialog box to pop up. This dialog box displays all coils that are used more than once in the current program.

4.4 Controller Menu

The Controller menu commands are used to configure the Micro16 and SCADAPack controllers. Controller selection, serial port configuration, I/O assignments, register assignment, and outputs on stop are defined in this menu selection. Controller initialization, I/O forcing and the C Program Loader are also selected from this menu.

4.4.1 Type

The TelePACE program is used to program and configure the Micro16, SCADAPack and SCADASense controllers. The selected controller determines controller specific options for other menus and communication functions.

TelePACE verifies the controller type when writing to the target controller and when entering the on-line modes. If the controller type selected in the application is different than the target controller, the following message appears:

"Cannot write to a different type or model of controller. Do you want to switch the controller type to xxxx?"

Where xxxx is the detected controller type.

The various controller types available in TelePACE are:

- SCADASense 4202 DR
- SCADASense 4202 DS
- SCADASense 4203 DR
- SCADASense 4203 DS
- Micro16
- SCADAPack
- SCADAPack 100: 1024K: This controller type can only be used when the controller firmware is version 1.80 or newer and the controller ID is greater than or equal to A182922.
- SCADAPack 100: 256K: This controller type can only be used when the controller firmware is version older than 1.80 and the controller ID is less than or equal to A182921.
- SCADAPack 32
- SCADAPack 32P
- SCADAPack LIGHT
- SCADAPack LP
- SCADAPack PLUS
- SCADAPack 330
- SCADAPack 334
- SCADAPack 350

Select the appropriate target controller this application is intended for.

4.4.2 Serial Ports

The Serial Ports command configures the serial ports on the flow computer. The command opens the Controller Serial Port Settings dialog.

Note: The SCADASense 4202 series controllers support Sensor protocol only on com1. The serial port settings for com1 cannot be edited. The settings are described in the following sections.

Controller Serial Port Settings

Port: **COM1** [OK] [Cancel]

Port Settings

Protocol: **Modbus RTU** [Default]

Addressing: **Standard**

Station: **1**

Duplex: **Full**

Baud Rate: **9600**

Data Bits: **8 Bits**

Parity: **None**

Stop Bits: **1 Bit**

Rx Flow: **None**

Tx Flow: **None**

Port Type: **RS-232**

Store and Forward: **Disabled**

Enron Modbus: **Disabled**

Enron Station: **1**

The **Port** drop down menu selects the controller serial port to configure. The settings for the port are displayed in the Port Settings controls section of the dialog. The valid serial ports depend on the controller type. The default serial port is com1.

Controller Type	com1	com2	com3	com4
Micro16	X	X		
SCADAPack	X	X	X	
SCADAPack Plus	X	X	X	X
SCADAPack Light	X	X		X
SCADAPack LP	X	X	X	
SCADAPack 100	X	X		
SCADAPack 32	X	X	X	X
SCADAPack 32P	X	X		X
SCADAPack 330/334	X	X	X	
SCADAPack 350	X	X	X	

Controller Type	com1	com2	com3	com4
SCADASense 4202 DR	X	X	X	
SCADASense 4202 DS	X	X	X	
SCADASense 4203 DR	X	X	X	
SCADASense 4203 DS	X	X	X	
SOLARPack 410	X	X	X	

The **Protocol** drop down menu selects the communication protocol type. Valid protocols depend on the controller type as shown in the following table.

Controller Type	Valid Protocols	Default Protocol
Micro16 SCADAPack SCADAPack Plus SCADAPack Light SCADAPack LP SCADAPack 100	None Modbus RTU Modbus ASCII DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC DNP * Note that SCADAPack 100 controllers with firmware older than version 1.80 do not support DF1 protocols.	Modbus RTU
SCADAPack 32 SCADAPack 32P	None Modbus RTU Modbus ASCII DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC DNP PPP	Modbus RTU
SCADASense family of programmable controllers (4202 DR, 4202 DS, 4203 DR and 4203 DS)	Com 1 fixed as Sensor. Com 2 and Com 3: None Modbus RTU Modbus ASCII DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC DNP	Com 1 fixed as Sensor. Com 2 and Com 3 default is Modbus RTU.
SCADAPack 330/334 SCADAPack 350	None Modbus RTU Modbus ASCII DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC	Modbus RTU

Controller Type	Valid Protocols	Default Protocol
	DNP	
SOLARPack 410	Com 1 fixed as Sensor. Com 2 and Com 3: None Modbus RTU Modbus ASCII DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC DNP	Com 1 fixed as Sensor. Com 2 and Com 3 default is Modbus RTU.

The **Addressing** drop down menu selects the addressing mode for the selected protocol. The control is disabled if the protocol does not support it. Valid addressing modes depend on the selected protocol as shown in the following table.

Protocol	Valid Mode	Default Mode
Modbus RTU	Standard	Standard
Modbus ASCII	Extended	
DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC	Control is disabled	N/A
DNP	Control is disabled	N/A
PPP	Control is disabled	N/A
None	Control is disabled	N/A

The **Station** entry sets the station address for the selected controller serial port. Valid addresses depend on the protocol and addressing mode selected, as shown in the table below.

Protocol	Valid Addresses	Default Address
Modbus RTU Modbus ASCII	Standard addressing: 1 to 255 Extended addressing: 1 to 65534	1
DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC	0 to 254	N/A
DNP	Control is disabled	N/A
PPP	Control is disabled	N/A
None	Control is disabled	N/A

The **Duplex** drop down menu selects full or half-duplex operation for the selected Port. Valid and default duplex settings depend on the serial port and controller type, as shown in the table below. Note that the duplex is forced to Half if a MVT transmitter is configured on the port.

Controller Type	com1		com2		com3		com4	
	Valid	Def.	Valid	Def.	Valid	Def.	Valid	Def.
Micro16	Full Half	Full	Full Half	Full				
SCADAPack	Full Half	Full	Full Half	Full	Half	Half		
SCADAPack Plus	Full Half	Full	Full Half	Full	Half	Half	Half	Half
SCADAPack Light	Full Half	Full	Full Half	Full			Half	Half
SCADAPack LP	Half	Half	Full Half	Full	Half	Half		
SCADAPack 100	Full Half	Full	Full Half	Full				
SCADAPack 32	Full Half	Half	Full Half	Half	Half	Half	Full Half	Half
SCADAPack 32P	Full Half	Full	Full Half	Full			Full Half	Full
SCADASense 4202 DR SCADASense 4202 DS	Full	Full	Half	Half	Half	Half		
SCADASense 4203 DR SCADASense 4203 DS	Half	Half	Full Half	Half	Full Half	Half		
SCADAPack 330/334	Full Half	Half	Full Half	Half	Half Full	Full		
SCADAPack 350	Half	Half	Full Half	Half	Half Full	Full		
SOLARPack 410	Half	Half	Full Half	Half	Full	Full		

The **Baud Rate** drop down menu selects the communication speed for the selected serial port. Valid baud rates depend on the serial port and controller type, as shown in the table below. The default value is always 9600 baud.

Baud Rate										
Controller	300	600	1200	2400	4800	9600	19200	38400	57600	115200
Micro 16										
Com 1, Com 2	X	X	X	X	X	X	X	X		
SCADAPack										
Com 1, Com 2	X	X	X	X	X	X	X	X		
Com 3			X	X	X	X	X	X	X	X
SCADAPack Plus										
Com 1, Com 2	X	X	X	X	X	X	X	X		
Com 3, Com 4			X	X	X	X	X	X	X	X

Baud Rate										
Controller	300	600	1200	2400	4800	9600	19200	38400	57600	115200
SCADAPack Light										
Com 1, Com 2	X	X	X	X	X	X	X	X		
Com 4			X	X	X	X	X	X	X	X
SCADAPack LP										
Com 1, Com 2	X	X	X	X	X	X	X	X		
Com 3			X	X	X	X	X	X	X	X
SCADAPack 100: 1024K										
Com 1, Com 2	X	X	X	X	X	X	X	X		
SCADAPack32										
Com 1, Com 2, Com 4	X	X	X	X	X	X	X	X	X	X
Com 3			X	X	X	X	X	X	X	X
SCADAPack32P										
Com 1, Com 2, Com 4	X	X	X	X	X	X	X	X	X	X
SCADAPack 330/334										
Com 1, Com 2, Com 3	X	X	X	X	X	X	X	X	X	X
SCADAPack 350										
Com 1, Com2, Com3	X	X	X	X	X	X	X	X	X	X
SCADASense Series of Programmable Controllers										
Com1					X					
Com2	X	X	X	X	X	X	X	X		
Com3			X	X	X	X	X	X	X	X
SOLARPack 410										
Com1 (disabled)							X			
Com2	X	X	X	X	X	X	X	X	X	X
Com3										X

The **Data Bits** drop down menu selects the number of data bits. Valid selections are 7 and 8 bits. This parameter is forced to 8 bits when the protocol type is Modbus RTU, PPP or any DF1 protocol. The default selection is 8 bits.

The **Parity** drop down menu selects the parity for the selected port. Valid selections depend on the serial port, controller type and data bits, as shown in the table below. The default selection is always none.

Controller Type	com1	com2	com3		com4	
			7 bits	8 bits	7 bits	8 bits
Micro16	None even odd	none even odd	N/A		N/A	
SCADAPack	None even odd	none even odd	even odd space mark	none even odd mark	N/A	
SCADAPack Plus	None even odd	none even odd	even odd space mark	none even odd mark	even odd space mark	none even odd mark
SCADAPack Light	none even odd	none even odd	N/A		even odd space mark	none even odd mark
SCADAPack LP	none even odd	none even odd	even odd space mark	none even odd mark	N/A	
SCADAPack 100	none even odd	none even odd				
SCADAPack 32	none even odd	none even odd	even odd space mark	none even odd mark	none even odd	
SCADAPack 32P	none even odd	none even odd	N/A		none even odd	
SCADASense 4202 DR SCADASense 4202 DS	none	none even odd	none even odd mark	N/A		
SCADASense 4203 DR SCADASense 4203 DS	none	none even odd	none even odd	N/A		

Controller Type	com1	com2	com3		com4	
			7 bits	8 bits	7 bits	8 bits
SCADAPack 330/334 SCADAPack 350	none even odd	none even odd	None even odd		N/A	
SOLARPack 410	none	none even odd	none		N/A	

The **Stop Bits** drop down menu selects the number of stop bits for the selected serial port. Valid selections are 1 and 2. Valid selection for com3 is 1 stop bit. The default selection is 1.

The **Rx Flow** drop down menu selects the receiver flow control for the selected port. Valid selections depend on the protocol, controller type, and serial port, as shown in the table below. If there is only one valid value the control is disabled. If there is more than one possible value, the default selection is none.

Protocol	Controller	com1	com2	com3	com4
DF1 Full BCC DF1 Full CRC DF1 Half BCC DF1 Half CRC DNP	Micro16	None	None	N/A	N/A
	SP	None	None	None	N/A
	SP Plus	None	None	None	None
	SP Light	None	None	N/A	none
	SP LP	None	None	None	N/A
	SP 100	None	None	N/A	N/A
	SP 32	None	None	None	None
	SP 32P	None	None	N/A	None
	SCADASense 4202 DR SCADASense 4202 DS	N/A	None	None Ignore CTS	N/A
	SCADASense 4203 DR SCADASense 4203 DS	N/A	None	None	N/A
	SCADAPack 330/334 SCADAPack 350	None	None	None	N/A
	SOLARPack 410	None	None	None	N/A
Modbus RTU	Micro16	None	None	N/A	N/A
	SP	None	None	Modbus RTU	N/A
	SP Plus	None	None	Modbus RTU	Modbus RTU
	SP Light	None	None		Modbus RTU
	SP LP	None	None	Modbus RTU	
	SP 100	None	None	N/A	N/A
	SP 32	Modbus RTU	Modbus RTU	Modbus RTU	Modbus RTU
	SP 32P	Modbus RTU	Modbus RTU		Modbus RTU

Protocol	Controller	com1	com2	com3	com4
	SCADASense 4202 DR SCADASense 4202 DS	None	None	Modbus RTU	
	SCADASense 4203 DR SCADASense 4203 DS	None	Modbus RTU	Modbus RTU	N/A
	SCADAPack 330/334 SCADAPack 350	Modbus RTU	Modbus RTU	Modbus RTU	N/A
	SOLARPack 410	None	Modbus RTU	Modbus RTU	N/A
None Modbus ASCII	Micro16	None Xon/Xoff	none Xon/Xoff		N/A
	SP	None Xon/Xoff	none Xon/Xoff	None	N/A
	SP Plus	None Xon/Xoff	none Xon/Xoff	None	None
	SP Light	None Xon/Xoff	none Xon/Xoff	N/A	None
	SP LP	None Xon/Xoff	none Xon/Xoff	None	N/A
	SP 100	None Xon/Xoff	none Xon/Xoff	None	None
	SP 32	None Modbus RTU	None Modbus RTU	None Modbus RTU	None Modbus RTU
	SP 32P	None Modbus RTU	None Modbus RTU	None Modbus RTU	None Modbus RTU
	SCADASense 4202 DR SCADASense 4202 DS	None	None	None Modbus RTU	N/A
	SCADASense 4203 DR SCADASense 4203 DS	None	None	None	N/A
	SCADAPack 330/334 SCADAPack 350	None	None	None	N/A
	SOLARPack 410	None	None	None	N/A
PPP	SP 32	Queued	Queued	Queued	Queued
	SP 32P	Queued	Queued	Queued	Queued

The **Tx Flow** drop down menu selects the transmitter flow control for the selected port. Valid selections depend on the protocol, controller type, and serial port, as shown in the table below. The default selection is none.

Protocol	Controller	com1	com2	com3	com4
Modbus RTU	Micro16	None	None	N/A	N/A
DF1 Full BCC DF1 Full CRC	SP	None	None	None Ignore CTS	N/A

Protocol	Controller	com1	com2	com3	com4
DF1 Half BCC DF1 Half CRC DNP	SP Plus	None	None	None Ignore CTS	None Ignore CTS
	SP Light	None	None	N/A	None Ignore CTS
	SP LP	None	None	None Ignore CTS	N/A
	SP 100	None	None	N/A	N/A
	SP 32	None Ignore CTS	None Ignore CTS	None Ignore CTS	None Ignore CTS
	SP 32P	None Ignore CTS	None Ignore CTS	N/A	None Ignore CTS
	SCADASense 4202 DR SCADASense 4202 DS	None	None	None Ignore CTS	N/A
	SCADASense 4203 DR SCADASense 4203 DS	None	None	None	N/A
	SCADAPack 330/334 SCADAPack 350	None	None	None	N/A
	SOLARPack 410	None	None	None	N/A
None Modbus ASCII	Micro16	None Xon/Xoff	None Xon/Xoff	N/A	N/A
	SP	None Xon/Xoff	None Xon/Xoff	None Ignore CTS	N/A
	SP Plus	None Xon/Xoff	None Xon/Xoff	None Ignore CTS	None Ignore CTS
	SP Light	None Xon/Xoff	None Xon/Xoff	N/A	None Ignore CTS
	SP LP	None Xon/Xoff	None Xon/Xoff	None Ignore CTS	N/A
	SP 100	None Xon/Xoff	None Xon/Xoff	N/A	N/A
	SP 32	None Ignore CTS	None Ignore CTS	None Ignore CTS	None Ignore CTS
	SP 32P	None Ignore CTS	None Ignore CTS	N/A	None Ignore CTS
	SCADASense 4202 DR SCADASense 4202 DS	N/A	None	None Ignore CTS	N/A
	SCADASense 4203 DR SCADASense 4203 DS	N/A	None	None	N/A
	SCADAPack 330/334 SCADAPack 350	None	None	None	N/A
	SOLARPack 410	None	None	None	N/A

Protocol	Controller	com1	com2	com3	com4
PPP	SP 32	None Ignore CTS	None Ignore CTS	None Ignore CTS	None Ignore CTS
	SP 32P	None Ignore CTS	None Ignore CTS	None Ignore CTS	None Ignore CTS

The **Port Type** drop down menu selects the type of serial port. Valid selections depend on the serial port and controller type as shown in the table below. The default selection is RS-232. The options are as follows:

- RS-232: for a regular RS-232 connection.
- RS-232 Dial-up modem: If an external dial-up modem is used on the RS-232 connection.
- RS-232 Collision Avoidance: RS-232 connection with collision avoidance based on the CD signal is available **only when the DNP protocol type is selected on the serial port, and the serial port supports handshaking.**

When this flow control is enabled, the protocol uses the Carrier Detect (CD) signal provided by the serial port to detect if the communication medium is in use. If it is, it waits until the medium is free before transmitting.

Prior to transmitting each Data Link (DL) frame, the controller will test the CD line. If it is active, a countdown equal to the DL timeout will be set and CD will be monitored every 100 ms throughout this countdown period. If the Data Link timeout is set to the minimum of 100 ms, the CD line will be tested once.

If the CD line reports inactive (line not in use), a frame will be transmitted immediately, and a new DL timeout is started as normal. On the other hand, if CD remains active during the DL timeout, the transmission attempt will fail. If a non-zero retry is configured in the Data Link layer, the test will be repeated until the number of retries has been exhausted.

Note: RS-232 Collision Avoidance is supported only on serial ports which support handshaking and whose protocol type is set for DNP.

- RS-485: for a regular RS-485 connection.

Controller Type	com1	com2, com4	com3
Micro16 SCADAPack SCADAPack Plus SCADAPack Light	RS-232 RS-232 dial –up modem RS-232 Collision Avoidance RS-485	RS-232 RS-232 dial –up modem RS-232 Collision Avoidance Note: com4 is available on SCADAPack Plus and SCADAPack Light only.	RS-232 RS-232 dial –up modem

SCADAPack 32 SCADAPack 32P	RS-232 RS-232 dial –up modem RS-232 Collision Avoidance Note: Port type RS-232 applies for RS-232 or RS-485 operation on Com1. Jumper J9 on the controller board must be installed to configure com1 for RS-485 operation.	RS-232 RS-232 dial –up modem RS-232 Collision Avoidance	

Controller Type	com1	com2, com3
Micro16 SCADAPack SCADAPack Plus SCADAPack Light	RS-232 RS-232 dial –up modem RS-232 Collision Avoidance RS-485	
SCADAPack 32 SCADAPack 32P		
SCADASense Programmable Controllers (4202 DR, 4202 DS, 4203 DR and 4203 DS).	N/A (RS-232)	RS-232 RS-232 Dialup Up Modem (com 2 only) NOTE: RS-232 Port Type applies for RS-232 or RS-485 operation.
SCADAPack 100	RS-232 RS-232 Collision Avoidance NOTE: Port type RS-232 applies for RS-232 or RS-485 operation on Com 1.	RS-232 RS-232 dial-up modem RS-232 Collision Avoidance Com 3: not available
SCADAPack LP	RS-485	RS-232 RS-232 dial-up modem RS-232 Collision Avoidance

SCADAPack 330/334	RS-232 RS-232 dial-up modem RS-232 Collision Avoidance RS-485 NOTE: Port type RS-232 applies for RS-232 or RS-485 operation on COM2. Jumper J8 on the controller board must be installed to configure COM1 for RS-485 operation.	RS-232 RS-232 dial-up modem RS-232 Collision Avoidance NOTE: Port type RS-232 applies for RS-232 or RS-485 operation on COM2. Jumper J10 on the controller board must be installed to configure COM2 for RS-485 operation.
SCADAPack 350	RS-485	RS-232 RS-232 dial-up modem RS-232 Collision Avoidance NOTE: Port type RS-232 applies for RS-232 or RS-485 operation on COM2. Jumper J13 on the controller board must be installed to configure COM2 for RS-485 operation.
SOLARPack 410	RS-232	RS-232

The **Store and Forward** drop down menu selects whether store and forward messaging is enabled for the port. Valid selections are enabled and disabled. If this option is enabled, messages will be forwarded according to the settings in the store and forward routing table. The default selection is disabled. This control is disabled when PPP protocol is selected for a serial port, or if any of the DF1 protocols are selected and for com 1 on the SCADASense 4202 or 4203 Series of controllers.

The **Store and Forward** menu selection changes to **Routing** menu selection when DNP protocol is selected for a serial port. Valid selections are enabled and disabled. Routing must be enabled on a serial port to enable routing of DNP messages.

The **Enron Modbus** drop down menu selects whether Enron Modbus is enabled for the port. If this option is enabled, the controller, in addition to regular Modbus messages, will handle Enron Modbus messages. Valid selections depend on the protocol as shown in the table below. This control is disabled when PPP protocol is selected for a serial port and for com 1 on the 4202 series controllers.

Protocol	Valid Selections	Default Selection
Modbus RTU Modbus ASCII	Enabled Disabled	Disabled
DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC	Control is disabled	N/A
DNP	Control is disabled	N/A
None	Control is disabled	N/A

The **Enron Station** entry selects the Enron Modbus station address for the serial port. Valid entries depend on the protocol. The Enron station must be different from the Modbus station set in the **Station** control. This ensures Enron Modbus and Modbus communication can occur on the same port. This entry is greyed out if Enron Modbus is not enabled.

Protocol	Valid Values	Default Value
Modbus RTU Modbus ASCII	Standard addressing: 1 to 255 Extended addressing: 1 to 65534	2
DF1 Full Duplex BCC DF1 Full Duplex CRC DF1 Half Duplex BCC DF1 Half Duplex CRC	Control is disabled	N/A
DNP	Control is disabled	N/A
None	Control is disabled	N/A

The **OK** button saves the settings for all serial ports and closes the dialog.

The **Cancel** button closes the dialog without saving.

The **Default** button sets the parameters for the port to their default values.

4.4.3 IP Configuration

When the IP Configuration menu item is clicked under the Controller menu the IP Configuration dialog is opened. This dialog is available only when the controller type is set to SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P.

The IP Configuration dialog has a tree control on the left side of the window. The SCADAPack 32 and SCADAPack 32P support Point-To-Point Protocol (PPP) on the serial ports. The tree control displays headings for com 1 Port through com 4 Port and PPP Login are displayed for configuring the serial ports for PPP. The SCADAPack 330, SCADAPack 334 and SCADAPack 350 controllers do not support PPP on the serial ports and the headings for com 1 Port through com 4 Port and PPP Login are not displayed.

Each of the tree control selections is explained in the following sections of this user manual.

This tree control for the SCADAPack 32 and SCADAPack 32P contains headings for:

- LAN Port
- com 1 Port
- com 2 Port
- com 3 Port
- com 4 Port
- PPP Login
- Modbus Common
- Modbus/TCP
- Modbus RTU in UDP
- Modbus ASCII in UDP
- DNP in TCP

- DNP in UDP
- Friendly IP List

This tree control for the SCADAPack 330, SCADAPack 334 and SCADAPack 350 controllers contains headings for:

- LAN Port
- Modbus Common
- Modbus/TCP
- Modbus RTU in UDP
- Modbus ASCII in UDP
- DNP in TCP
- DNP in UDP
- Friendly IP List

When a tree control is selected by clicking the mouse on a heading, a property page is opened for the header selected. From the property page the IP configuration parameters for the selected header is displayed.

The **Default** button selects the default values for the current property page.

The **OK** button saves the configuration and closes the Controller IP Configuration dialog.

The **Cancel** button closes the Controller IP Configuration dialog without saving any changes.

4.4.3.1 LAN Port

The LAN Port property page is selected for editing by clicking LAN Port in the tree control section of the Controller IP Configuration dialog. When selected the LAN Port property page is active.

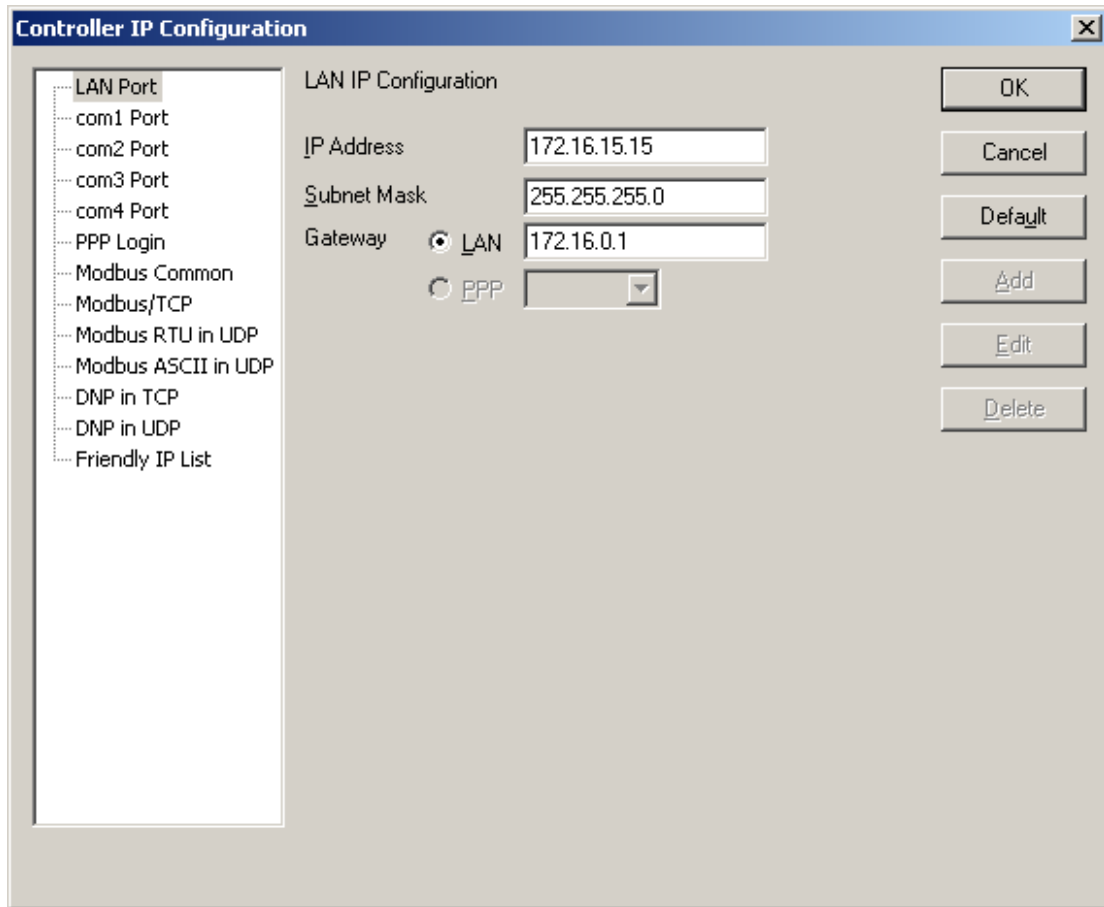


Figure 25: Controller IP Configuration – LAN Port Dialog Box

The **IP Address** is the address of the controller LAN port. The IP address is statically assigned. Contact your network administrator to obtain an IP address for the controller. The default value is 0.0.0.0.

The **Subnet Mask** determines the subnet on which the controller LAN port is located. The subnet mask is statically assigned. Contact your network administrator to obtain the subnet mask for the controller. The default value is 255.255.255.0.

The **Gateway** determines how your controller communicates with devices outside its subnet. The **LAN** radio button selects the gateway specified in the **LAN** edit box. Enter the IP address of the gateway. The gateway is located on the LAN port subnet. The gateway is statically assigned. Contact your network administrator to obtain the gateway IP address. The default value is 0.0.0.0.

The **PPP** radio button selects the serial port where the gateway is located. The **PPP** drop down menu displays only those serial ports currently configured for the PPP protocol. Select a serial port from this menu to select its remote IP address as the gateway. The gateway is automatically assigned to the remote IP address of the selected serial port.

4.4.3.2 com1 Port

The com1 Port property page is selected for editing by clicking com1 Port in the tree control section of the Controller IP Configuration dialog. When selected the com1 Port property page is active. This page configures the IP settings for com1 when the PPP protocol is selected for this serial port.

Figure 26: Controller IP Configuration – com1 Port Dialog Box

The **Enable Auto Answer (PPP Server)** checkbox enables the PPP Server on this serial port. Check this box if you want to allow a remote PPP client to connect to this port. This checkbox enables the remaining settings in the page.

The **IP Address** is the address of this serial port. The IP address is statically assigned. Contact your network administrator to obtain an IP address for this serial port.

The **Subnet Mask** determines the subnet on which this serial port is located. The subnet mask is statically assigned. Contact your network administrator to obtain the subnet mask for this serial port. In a standard PPP configuration, a subnet mask of 255.255.255.255 is used to restrict routing on this serial port to a single host (i.e. the **Remote IP Address**).

If another subnet mask is used, all packets on that subnet will be forwarded to this serial port. Any address on that subnet in addition to the **Remote IP Address** can be used for the remote host in this case.

The **Remote IP Address** is the address that will be assigned to the remote PPP client connected to this serial port. The **Automatic** radio button automatically selects the address to be the serial port's IP address + 1. The second radio button selects the address specified in the edit box. Enter the IP address to assign to the remote client.

The **Allow remote to specify its own IP address** checkbox allows the remote PPP client to assign its own IP address. Check this box if you want to allow this option. Note that the client may or may not request its own IP address. If the client does not make this request, the PPP Server will assign the IP address selected.

The **Authentication** determines the login protocol used at the start of every PPP connection. The **None** radio button removes the login step. The **PAP** radio button selects the Password Authentication Protocol (PAP). The **CHAP** radio button selects the Challenge-Handshake Authentication Protocol (CHAP). PAP and CHAP usernames and passwords are configured on the **PPP Login** page.

The **Inactivity Timeout** is the inactivity timeout for this serial port. If there has been no activity on an existing PPP connection for the selected number of minutes, then the connection is automatically closed. If there is a modem connected it is hung up. Setting this value to zero disables the timeout.

4.4.3.3 com2 Port

The com2 Port property page is selected for editing by clicking com2 Port in the tree control section of the IP Configuration dialog. When selected the com2 Port property page is active. This page configures the IP settings for com2 when the PPP protocol is selected for this serial port.

The com2 Port property page provides the same options as the com1 Port page. See the com1 Port page for a description of these options.

4.4.3.4 com3 Port

The com3 Port property page is selected for editing by clicking com3 Port in the tree control section of the IP Configuration dialog. When selected the com3 Port property page is active. This page configures the IP settings for com3 when the PPP protocol is selected for this serial port.

The com3 Port property page provides the same options as the com1 Port page. See the com1 Port page for a description of these options.

4.4.3.5 com4 Port

The com4 Port property page is selected for editing by clicking com4 Port in the tree control section of the IP Configuration dialog. When selected the com4 Port property page is active. This page configures the IP settings for com4 when the PPP protocol is selected for this serial port.

The com4 Port property page provides the same options as the com1 Port page. See the com1 Port page for a description of these options.

4.4.3.6 PPP Login

The PPP Login property page is selected for editing by clicking PPP Login in the tree control section of the IP Configuration dialog. When selected the PPP Login property page is active.

This page configures the username and password list for PPP login authentication. The list is used only by those serial ports configured for the PPP protocol using PAP or CHAP authentication.

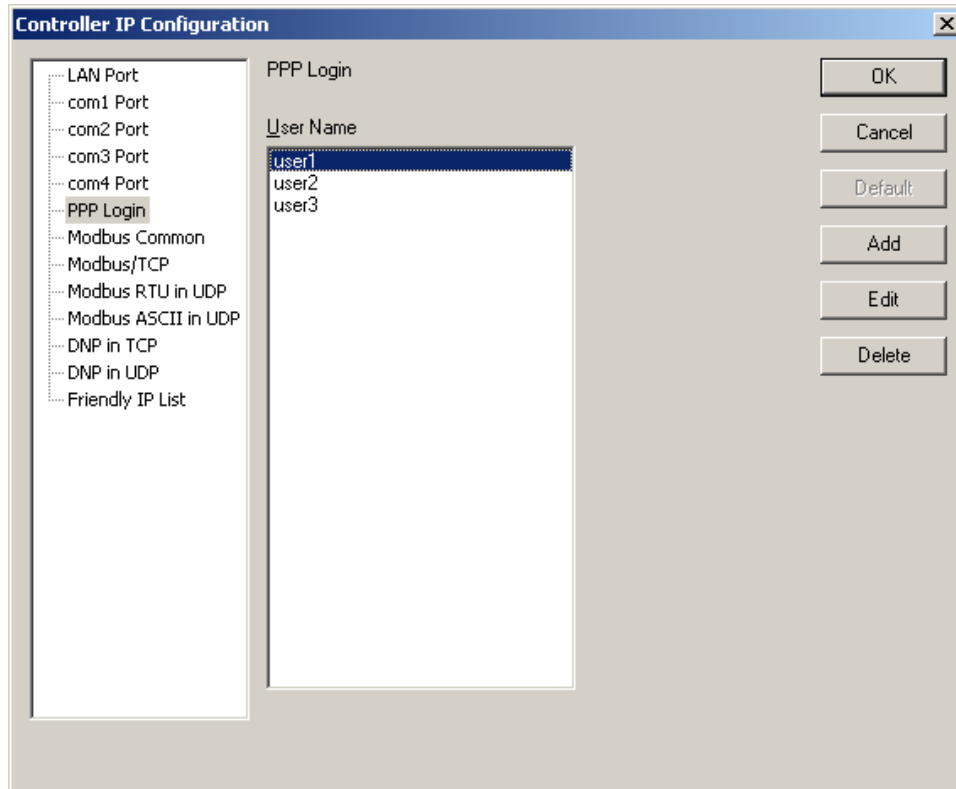


Figure 27: Controller IP Configuration – PPP Login Dialog Box

- Click **Add** to enter a new username to the list. Selecting the Add button opens the Add PPP Username dialog.
- Click **Edit** to edit the username highlighted in the list. Selecting the Edit button opens the Edit PPP Username dialog. This button is disabled if there are no entries in the list.
- Click **Delete** to remove the selected usernames from the list. This button is disabled if there are no entries in the list.

4.4.3.6.1 Add PPP Username dialog

This dialog selects a new PPP username and password.

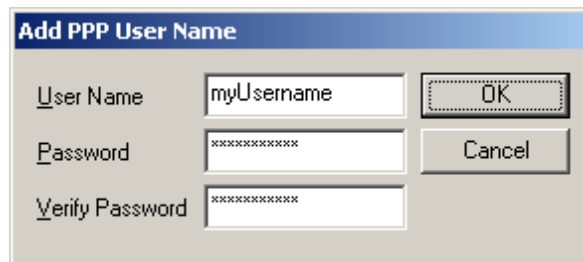


Figure 28: Add PPP User Name Dialog Box

The **Username** edit box selects the username. A username is any alphanumeric string 1 to 16 characters in length, and is case sensitive.

The **Password** edit box selects the password. A password is any alphanumeric string 1 to 16 characters in length, and is case sensitive.

The **Verify Password** edit box selects the verify password. Enter the same string entered for the password.

The **Cancel** button discards any changes made to this dialog and exits the dialog.

The **OK** button to accepts all changes made to this dialog and exits the dialog.

4.4.3.6.2 Edit PPP Username dialog

This dialog edits a PPP username and password selected from the list.

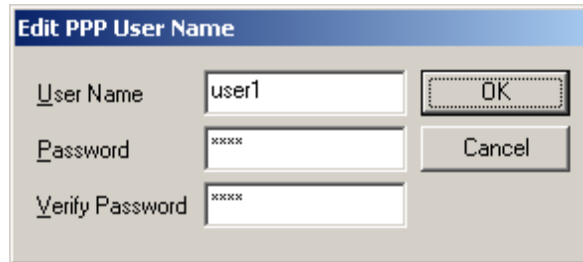


Figure 29: Edit PPP User Name Dialog Box

The **Username** edit box selects the username. A username is any alphanumeric string 1 to 16 characters in length, and is case sensitive.

The **Password** edit box selects the password. A password is any alphanumeric string 1 to 16 characters in length, and is case sensitive.

The **Verify Password** edit box selects the verify password. Enter the same string entered for the password.

The **Cancel** button discards any changes made to this dialog and exits the dialog.

The **OK** button to accepts all changes made to this dialog and exits the dialog.

4.4.3.7 Modbus Common

The Modbus Common property page is selected for editing by clicking Modbus Common in the tree control section of the IP Configuration dialog. When selected the Modbus Common property page is active.

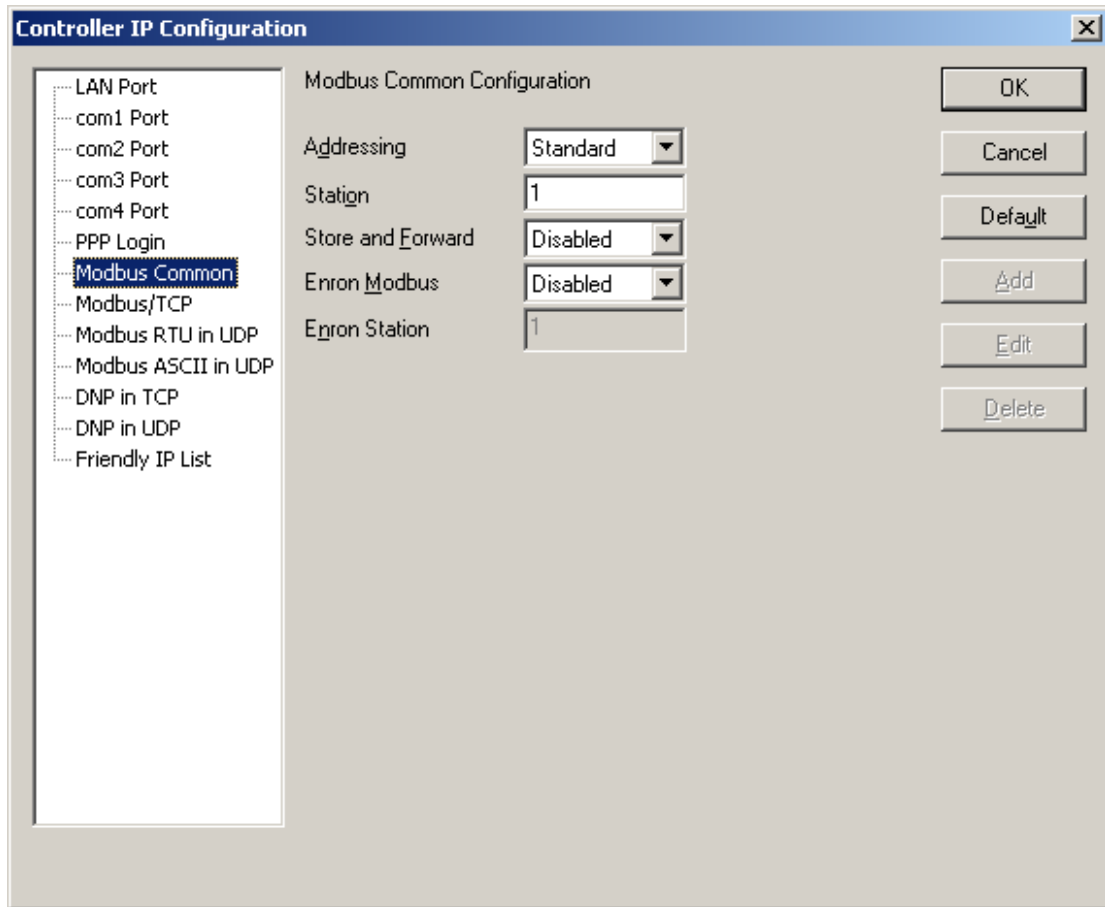


Figure 30: Controller IP Configuration - Modbus Common Dialog Box

The **Addressing** menu selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default value is standard.

The **Station** menu sets the station number of the controller. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default value is 1.

The **Store and Forward** selection controls forwarding of messages using IP based protocols. If this option is enabled, messages will be forwarded according to the settings in the store and forward routing table. The default value is disabled.

The **Enron Modbus** box selects whether Enron Modbus is enabled for the port. If this option is enabled, the controller, in addition to regular Modbus messages, will handle Enron Modbus messages.

The **Enron Station** box selects the Enron Modbus station address. The valid range for Enron Station is 1 to 255 if the Addressing control is set to Standard. The valid range for Enron Station is 1 to 65534 if the Addressing control is set to Extended. The Enron station must be different from the Modbus station set in the **Station** edit box. This ensures Enron Modbus and Modbus communication can occur on the same port.

4.4.3.8 Modbus/TCP

The Modbus/TCP property page is selected for editing by clicking Modbus/TCP in the tree control section of the IP Configuration dialog. When selected the Modbus/TCP property page is active.

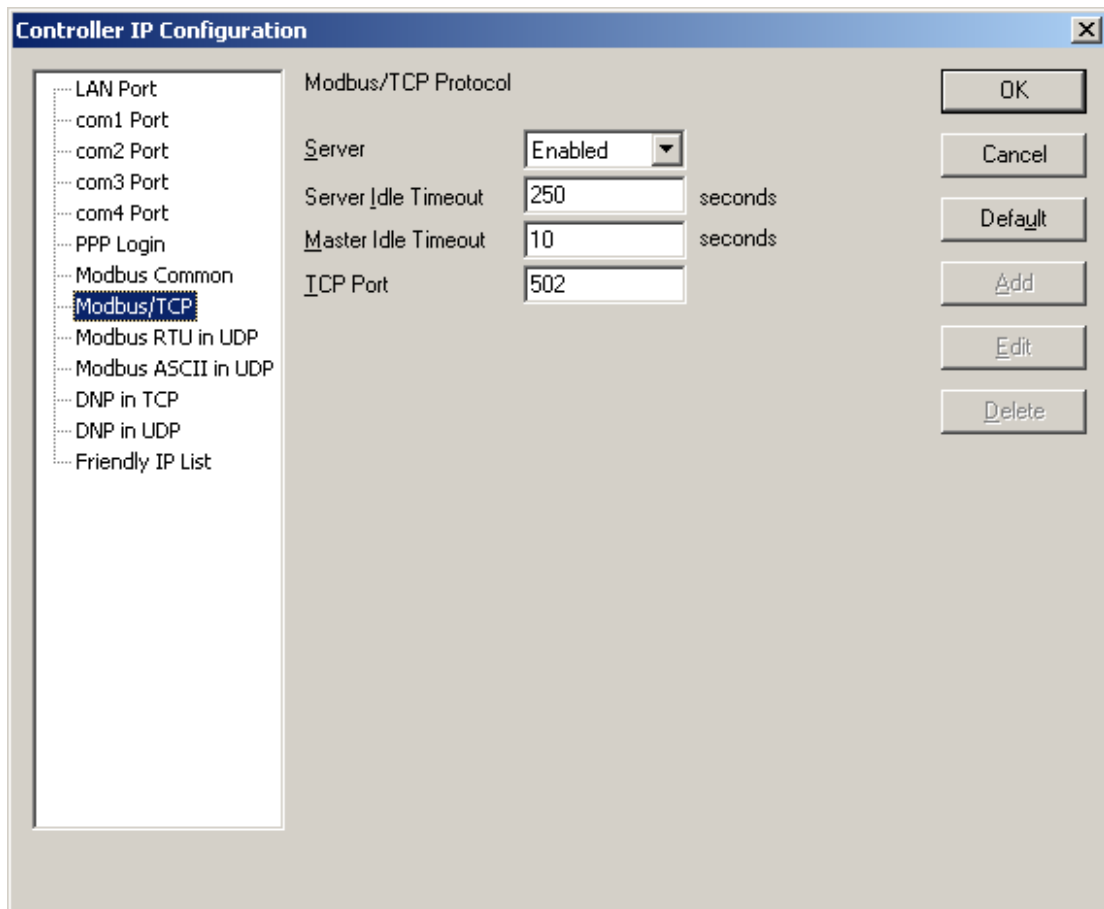


Figure 31: Controller IP Configuration - Modbus/TCP Dialog Box

The **Server** selection selects whether the server is enabled. If this option is enabled the controller supports incoming slave messages. Disabling this option prevents the controller from processing slave messages. Master messaging is always enabled.

The **Master Idle Timeout** determines when connections to a slave controller are closed. Setting this value to zero disables the timeout; the connection will be closed only when your program closes it. Any other value sets the timeout in seconds. The connection will be closed if no messages are sent in that time. This allows the slave device to free unused connections. Valid timeout range is 0 to 4294967295 seconds. The default value is 10 seconds.

The **Server Idle Timeout** determines when connections from a remote device are closed. Setting this value to zero disables the timeout; the connection will be closed only when the remote device closes it. Any other value sets the timeout in seconds. The connection will be closed if no messages are received in that time. This allows the controller to free unused connections. Valid timeout range is 0 to 4294967295 seconds. The default value is 250 seconds.

The **TCP Port** sets the port used by the Modbus/TCP protocol. In almost all cases this should be set to 502. This is the well-known port number for Modbus/TCP. Modbus/TCP devices use 502 by default, and on many devices the value cannot be changed. It is suggested that you change this value

only if this port is used by another service on your network. Valid port number range is 1 to 65534. Consult your network administrator to obtain a port if you are not using the default.

4.4.3.9 Modbus RTU in UDP

The Modbus RTU in UDP property page is selected for editing by clicking Modbus RTU in UDP in the tree control section of the IP Configuration dialog. When selected the Modbus RTU in UDP property page is active.

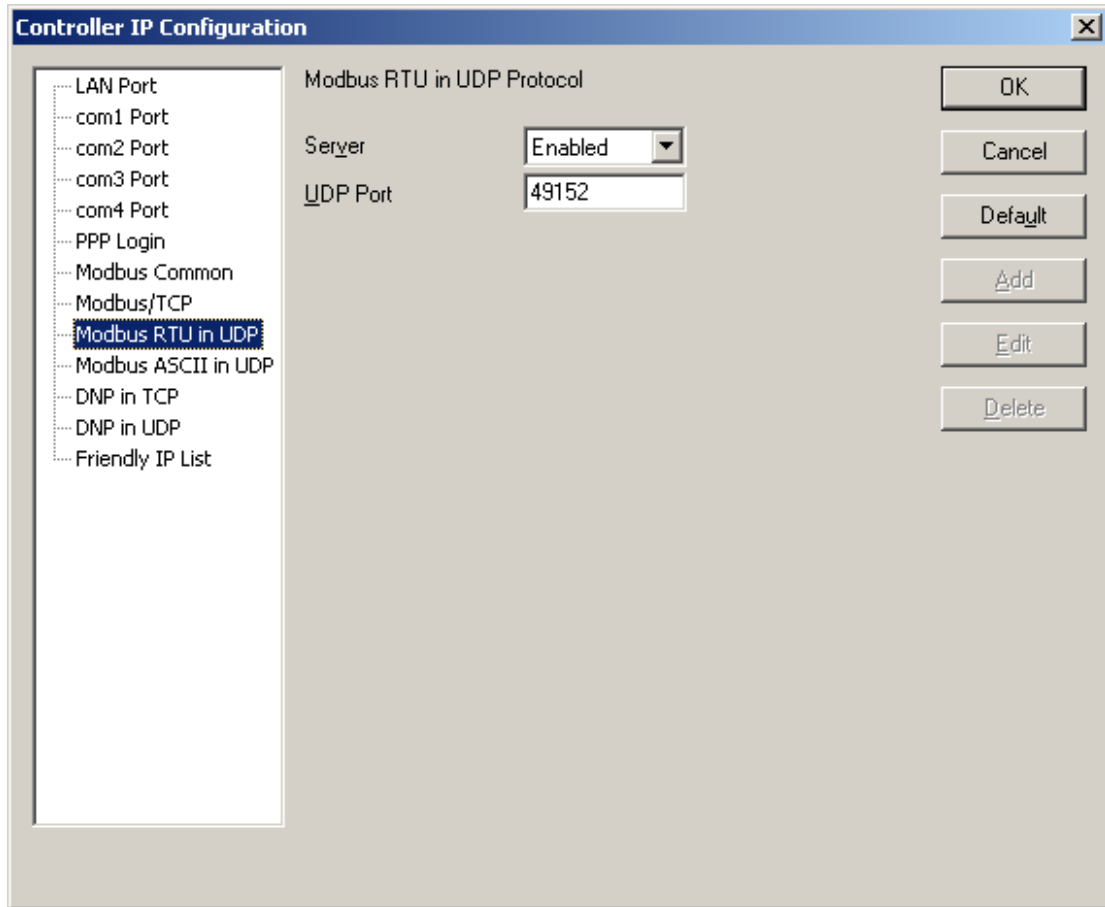


Figure 32: Controller IP Configuration Modbus RTU in UDP Dialog Box

The **Server** selection selects whether the server is enabled. If this option is enabled the controller supports incoming slave messages. Disabling this option prevents the controller from processing slave messages. Master messaging is always enabled.

The **UDP Port** sets the port used by the protocol. Valid port number range is 1 to 65534. The default value is 49152. This is a recommendation only. Consult your network administrator to obtain a port if you are not using the default.

4.4.3.10 Modbus ASCII in UDP

The Modbus ASCII in UDP property page is selected for editing by clicking Modbus ASCII in UDP in the tree control section of the IP Configuration dialog. When selected the Modbus ASCII in UDP property page is active.

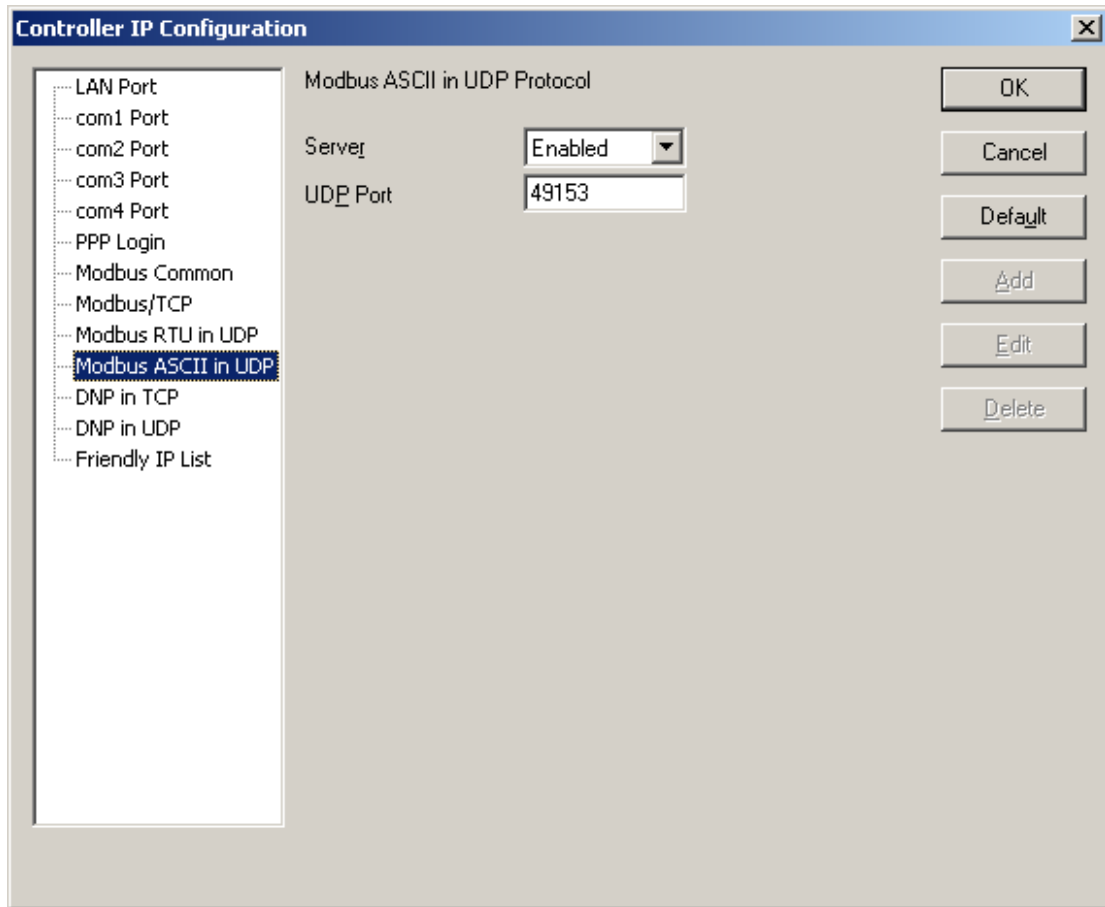


Figure 33: Controller IP Configuration – Modbus ASCII in UDP Dialog Box

The **Server** selection selects whether the server is enabled. If this option is enabled the controller supports incoming slave messages. Disabling this option prevents the controller from processing slave messages. Master messaging is always enabled.

The **UDP Port** sets the port used by the protocol. Valid port number range is 1 to 65534. The default value is 49153. This is a recommendation only. Consult your network administrator to obtain a port if you are not using the default.

4.4.3.11 DNP in TCP

The DNP in TCP property page is selected for editing by DNP in TCP in the tree control section of the IP Configuration dialog. When selected the DNP in TCP property page is active.

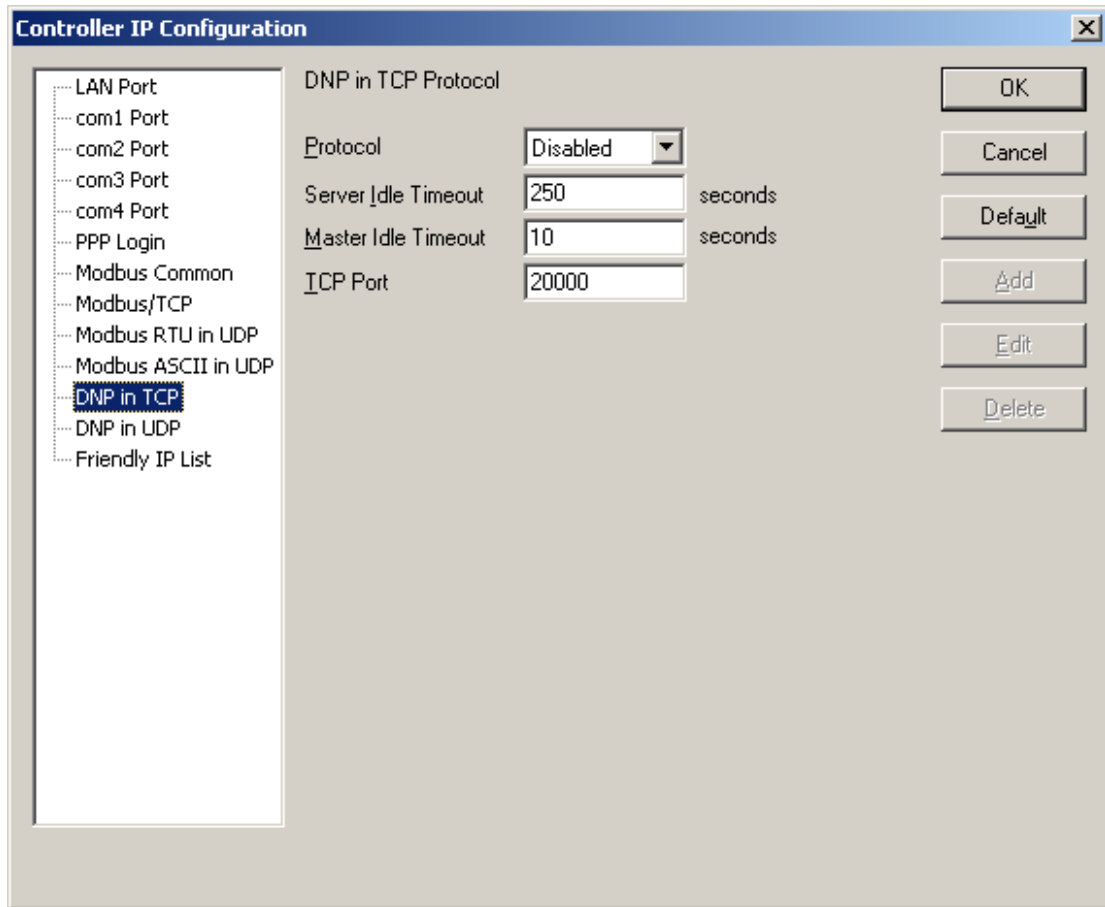


Figure 34: Controller IP Configuration – DNP in TCP Dialog Box

The **Protocol** selection selects whether the DNP in TCP protocol is enabled. If this option is enabled the controller supports DNP in TCP protocol. Disabling this option prevents the controller from processing DNP in TCP protocol messages. Master messaging is always enabled. The default selection is disabled.

The **Server Idle Timeout** determines when connections from a remote device are closed. Setting this value to zero disables the timeout; the connection will be closed only when the remote device closes it. Any other value sets the timeout in seconds. The connection will be closed if no messages are received in that time. This allows the controller to free unused connections. Valid timeout range is 0 to 4294967295 seconds. The default value is 250 seconds.

The **Master Idle Timeout** determines when connections to a slave controller are closed. Setting this value to zero disables the timeout; the connection will be closed only when your program closes it. Any other value sets the timeout in seconds. The connection will be closed if no messages are sent in that time. This allows the slave device to free unused connections. Valid timeout range is 0 to 4294967295 seconds. The default value is 10 seconds.

The **TCP Port** sets the port used by the DNP in TCP protocol. Valid port number range is 1 to 65534. The default value is 20000. Consult your network administrator to obtain a port if you are not using the default.

4.4.3.12 DNP in UDP

The DNP in UDP property page is selected for editing by DNP in UDP in the tree control section of the IP Configuration dialog. When selected the DNP in UDP property page is active.

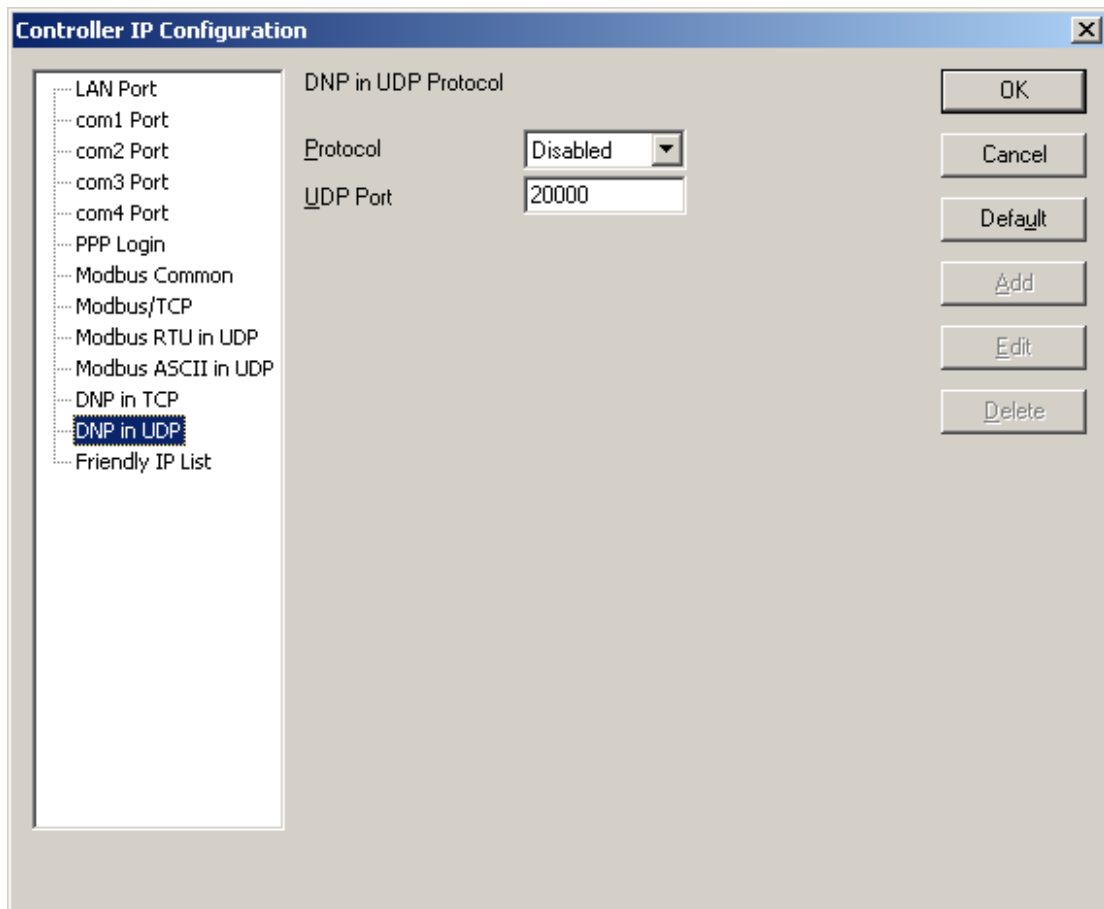


Figure 35: Controller IP Configuration – DNP in UDP Dialog Box

The **Protocol** selection selects whether the DNP in UDP protocol is enabled. If this option is enabled the controller supports DNP in UDP protocol. Disabling this option prevents the controller from processing DNP in UDP protocol messages and sending DNP in UDP master messages. The default selection is disabled.

The **UDP Port** sets the port used by the DNP in UDP protocol. Valid port number range is 1 to 65534. The default value is 20000. Consult your network administrator to obtain a port if you are not using the default.

4.4.3.13 Friendly IP List

The Friendly IP property page is selected for editing by Friendly IP in the tree control section of the IP Configuration dialog.

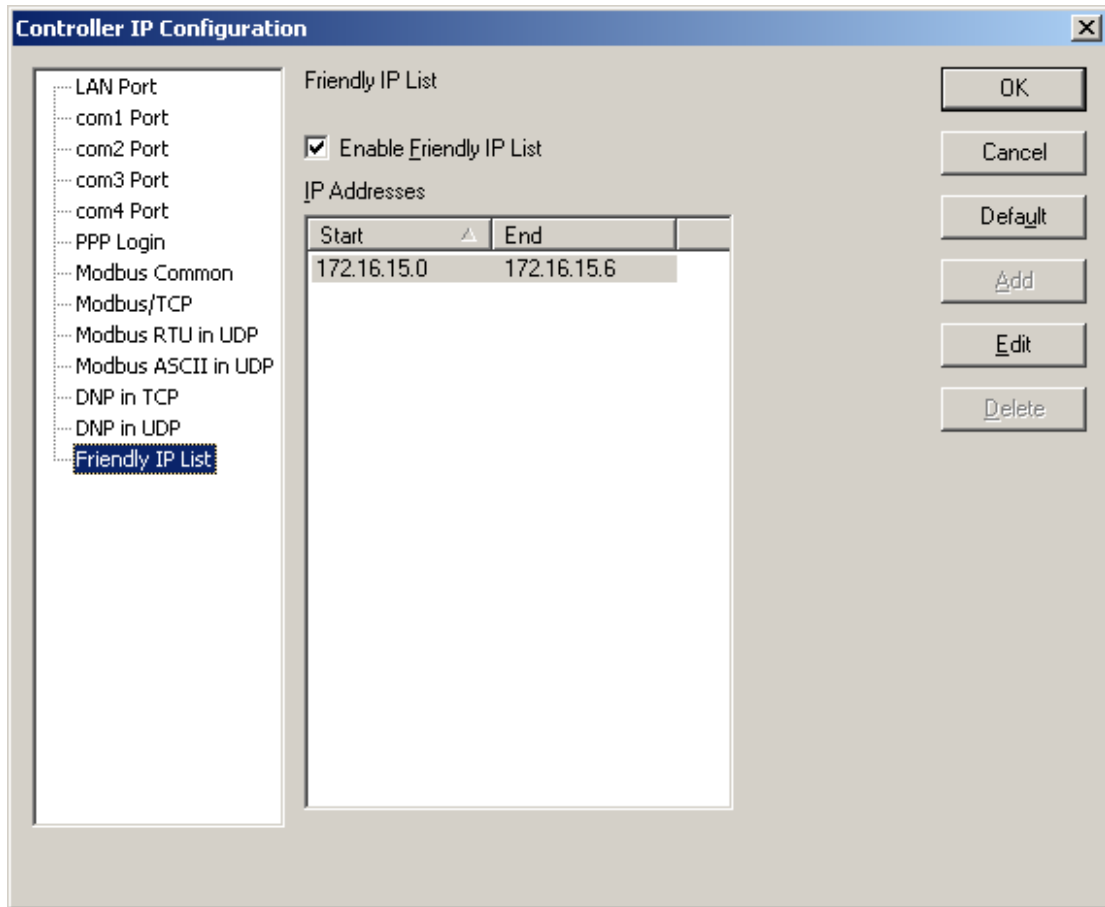


Figure 36: Controller IP Configuration – Friendly IP List Dialog Box

The **Enable Friendly IP List** checkbox enables or disables the friendly IP list. Check this box to accept messages from only the IP addresses in the list. Uncheck this to accept message from all IP addresses.

Click **Add** to enter a new row in the Friendly IP list. Selecting the Add button opens the **Add Friendly IP address** dialog. The button is disabled if the **Enable Friendly IP List** control is not checked. The button is disabled if the table is full. Up to 32 entries can be added to the table.

Click **Edit** to edit range in the Friendly IP list. Selecting the Edit button opens the **Edit Friendly IP address** dialog. The button is disabled if the **Enable Friendly IP List** control is not checked.

The **Delete** button removes the selected rows from the list. This button is disabled if there are no entries in the list. The button is disabled if the **Enable Friendly IP List** control is not checked.

Click on the column headings to sort the list by that column. Click a second time to reverse the sort order. The order is indicated by the triangle next to the text.

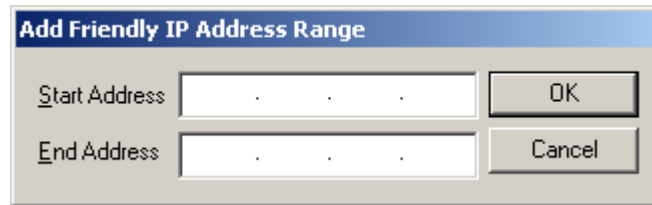
The settings are verified when the OK button is pressed or another settings page is selected.

An error message is displayed if the friendly IP list is enabled and the list is empty.

A warning message is displayed if the IP address of the PC is not in the friendly IP table.

4.4.3.13.1 Add Friendly IP Address Range Dialog

The Add Friendly IP Address Range dialog specifies an IP address range to add to the Friendly IP list.



The dialog box has a title bar "Add Friendly IP Address Range". It contains two input fields: "Start Address" and "End Address", each followed by a button. The "Start Address" field has a button labeled "OK" and the "End Address" field has a button labeled "Cancel".

Figure 37: Add Friendly IP Address Range Dialog Box

Start Address specifies the starting IP address in the range. Enter any valid IP address.

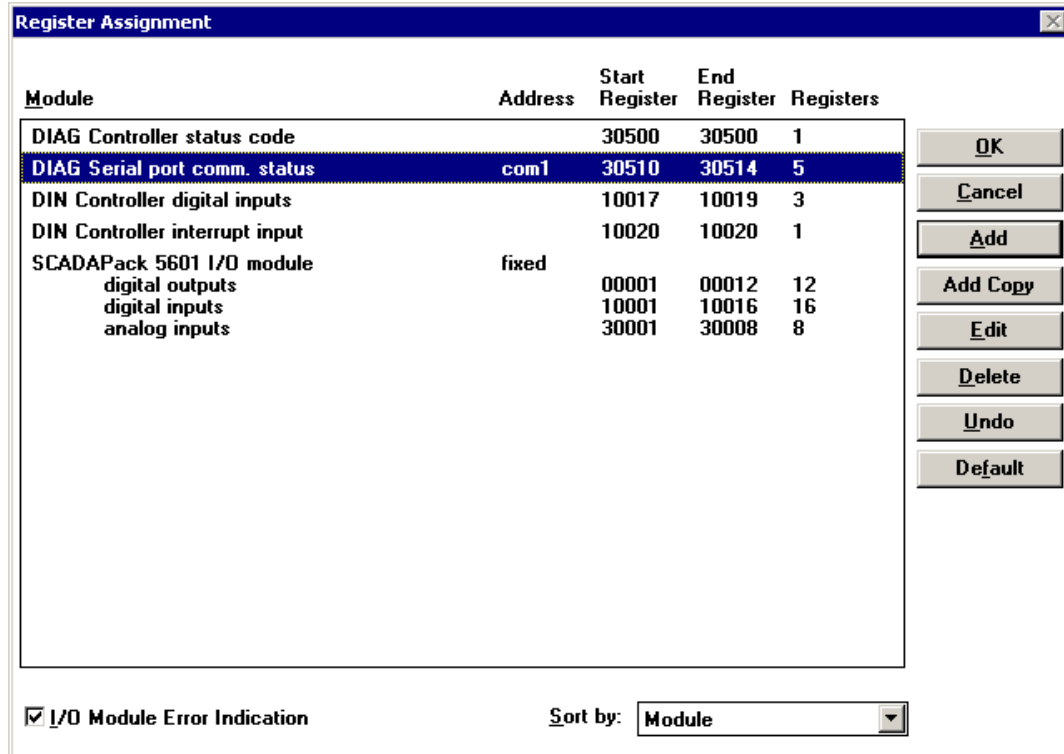
End Address specifies the ending IP address in the range. Enter a valid IP address that is numerically greater than or equal to the IP Start Address. This field can be left blank if only a single IP address is required.

The **OK** button adds the IP address range to the list and closes the dialog. An error is displayed if the address range is invalid.

The **Cancel** button closes the dialog without making any changes.

4.4.4 Register Assignment

The Register Assignment dialog displays the current register assignment list. The user may edit the list or the entries in the list.



The dialog box has a title bar "Register Assignment". It contains a table with the following columns: Module, Address, Start Register, End Register, and Registers. The table has the following rows:

Module	Address	Start Register	End Register	Registers
DIAG Controller status code		30500	30500	1
DIAG Serial port comm. status	com1	30510	30514	5
DIN Controller digital inputs		10017	10019	3
DIN Controller interrupt input		10020	10020	1
SCADAPack 5601 I/O module	fixed			
digital outputs		00001	00012	12
digital inputs		10001	10016	16
analog inputs		30001	30008	8

On the right side of the table are buttons: OK, Cancel, Add, Add Copy, Edit, Delete, Undo, and Default. At the bottom left is a checkbox labeled "I/O Module Error Indication" which is checked. At the bottom right is a label "Sort by:" followed by a dropdown menu showing "Module".

Figure 38: Register Assignment Dialog Box

The main portion of the dialog is a list showing the modules in the register assignment list. The module list displays the Module, Module Address, Start Register, End Register and the number of Registers for the module.

The **Module** field displays the type and name of I/O modules that have been added to the Register Assignment. For modules that support more than one type of I/O, there are multiple lines in the row of the table, one for each input or output I/O type.

The **Address** field displays the unique module address of the physical hardware, such as a 5000 Series 5401 Digital I/O module. Some module types have no address that can be set by the user. The address is blank for these modules.

The **Start Register** field displays the first register address in the I/O database where the module data is stored. A start register is required for each type of input or output on the module.

The **End Register** field displays the last address in the I/O database used by the module. An end register is required for each type of input or output on the module.

The **Registers** field displays the number of registers used by the module. A size is required for each type of input or output on the module.

The **I/O Module Error Indication** check box determines if the controller displays I/O module communication errors. If enabled, the controller will blink the Status LED if there is an I/O error. See the DIAG Controller Status Code diagnostic module for information on the controller status code. If disabled, the controller will not display the module communication status. The module communication status is always checked. This option controls only the indication on the Status LED.

The **Sort by:** window selects how the Register Assignment List is sorted. The selections are: **Module**, **Start Register**, or **Address**.

Click **OK** to update the register assignment list and close the dialog.

If the register assignment is changed it is written to the controller as follows.

If TelePACE is Off Line the changes are not written.

If TelePACE is On Line, the changes are downloaded automatically.

Selecting **Cancel** exits the dialog without saving changes. If changes were made, the user is prompted for confirmation before exiting. This protects against accidentally discarding a large number of changes. Press **ESC** to **Cancel**.

If changes were made the following dialog appears.

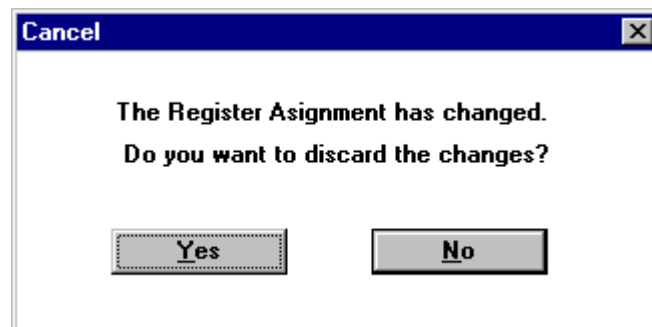


Figure 39: Cancel Register Assignment Dialog Box

Click **Yes** to close the Register Assignment dialog and the register assignment list is not changed.

Click **No** to return to the Register Assignment dialog. **No** is the default selection. Press **Enter** or **ESC** to select **No**.

Click **Add** to open the **Edit Register Assignment** dialog. The Add function adds a new module to the register assignment.

Click **Add Copy** to open the **Edit Register Assignment** dialog. The Address field is set to the first unused address for the currently selected module type. All other fields are set to the values from the currently selected I/O module. The Add Copy button is grayed if:

- the table is empty,
- all I/O modules of the given type are in use (i.e. modules are already defined for all possible addresses for the selected module type),

Selecting **Edit** opens the Edit Register Assignment dialog. All fields are set to the values from the currently selected I/O module. The Edit button is grayed if:

- the table is empty,
- no module is selected in the table.

The **Delete** function removes the selected module from the register assignment. The **Delete** button is grayed if:

- the table is empty,
- no module is selected in the table.

Selecting the **Undo** button undoes the last change to the register assignment list. A single level of undo is provided. Selecting the Undo function a second time, restores the list to its state before the first Undo.

The **Default** function replaces the current Register Assignment with the Default Register Assignment for the controller type selected.

If the controller is a SCADAPack or SCADAPack 32 the following prompt is displayed.

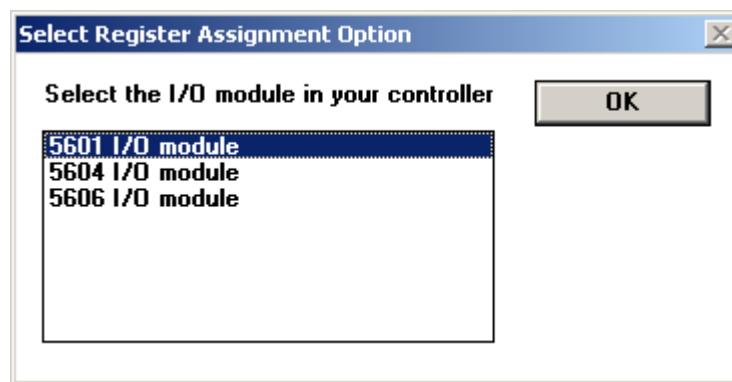


Figure 40: Select Register Assignment Option Dialog Box

Select the I/O module, 5601, 5604 or 5606 that you are using in your controller and click OK.

If the controller type is a SCADASense 4202 DR, the following dialog appears.

The Register Assignment dialog box displays a table with the following data:

Module	Address	Start Register	End Register	Registers
4202GFC I/O	fixed	10001	10001	1
digital inputs		30001	30005	5
analog and counter inputs		40500	40500	1

Buttons on the right: OK, Cancel, Add, Add Copy, Edit, Delete, Undo, Default.

At the bottom: ☒ I/O Module Error Indication, Sort by: Module (dropdown).

Figure 41: Register Assignment Dialog Box

4.4.4.1.1 Edit Register Assignment Dialog

The Edit Register Assignment dialog modifies an entry in the register assignment. The following example shows the dialog editing a module with more than one I/O type.

The Add Register Assignment dialog box contains the following fields and table:

Module: SCADAPack 5602 I/O module (dropdown)
 Address: fixed (dropdown)

Type	Start	End	Registers	Description
0xxxx	1	2	2	digital outputs
1xxxx	10001	10005	5	digital inputs
3xxxx	30001	30005	5	analog inputs

Buttons: OK, Cancel.

Extended Parameters: None

Figure 42: Edit Register Assignment Dialog Box

The **Module** drop-down list box shows the current module type. The drop-down list displays all available modules. If the maximum number of modules of a particular I/O module type is already configured, no modules of this class are shown in the drop-down list.

The **Address** drop-down list box shows the current module address. The drop-down list displays all addresses valid for the current module type that are not already used. If all addresses are in use, the list will be empty.

There are up to four **Type** description fields. The text displays the type of the input or output register. The type descriptions are 0xxxx, 1xxxx, 3xxxx and 4xxxx.

- Digital output data is read from coil (**0xxxx**) registers. The digital outputs are updated continuously with data read from the coil registers.
- Digital input data is stored in status (**1xxxx**) registers. The status registers are updated continuously with data read from the digital inputs.
- Analog input data is stored in input (**3xxxx**) registers. The input registers are updated continuously with data read from the analog inputs.
- Analog output data is stored in holding (**4xxxx**) registers. The analog output registers are updated continuously with data read from the holding registers.

The **Start** edit box holds the starting register in the I/O database for the I/O type. The edit box allows any number to be entered.

The **End** field shows the last register used by the module for the I/O type. TelePACE automatically fills this field.

The **Registers** field shows the number of registers in the I/O module. TelePACE automatically fills this field.

The **Description** field displays the I/O type for multiple I/O modules. TelePACE automatically fills this field.

The **Extended Parameters** window will contain I/O module extended parameters if they are used. Extended parameters may include such items as analog input type for example. Refer to the Register Assignment Reference for information on all I/O modules.

Selecting **OK** checks the data entered. If the data is correct the dialog is closed and the Register Assignment dialog returns with the changes made. If any data is incorrect, a beep will sound and the cursor will select the field containing the first error found. Press **ENTER** to select **OK**.

Click **Cancel** to exit the dialog without saving changes. Press **ESC** to select **Cancel**.

4.4.5 Outputs On Stop

Select Outputs on Stop command and the Output Conditions on Program Stop dialog appears. This dialog controls the state of the controller analog and digital outputs when the ladder logic program is stopped.

The state of the digital outputs may be set to Hold their last value or to turn Off when the ladder program is stopped.

The state of the analog outputs may be set to Hold their last value or to go to Zero when the ladder program is stopped.

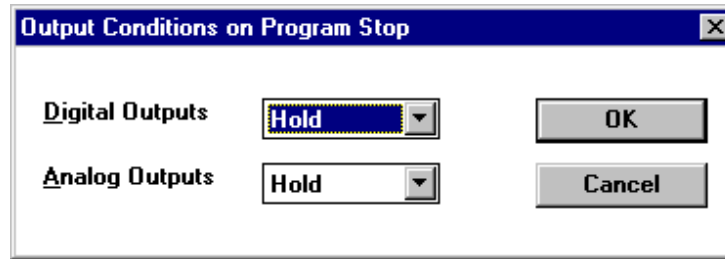


Figure 43: Output Conditions on Program Stop Dialog Box

The **Digital Output** drop-down list box specifies the state of the digital outputs. It has two options: Hold and Off.

The **Analog Output** drop-down list box specifies the state of the analog outputs. It has two options: Hold and Zero.

4.4.6 Store and Forward

The Store and Forward command is used to configure store and forward messaging. A controller configured for store and forward operation receives messages destined for a remote Slave Station on the Slave Interface. The controller forwards the message on the Forward Interface to the Forward Station.

Note: This command is available only when the controller type is set to SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P.

Refer to the following diagram as a reference for the terminology used in the following Store and Forward command reference.

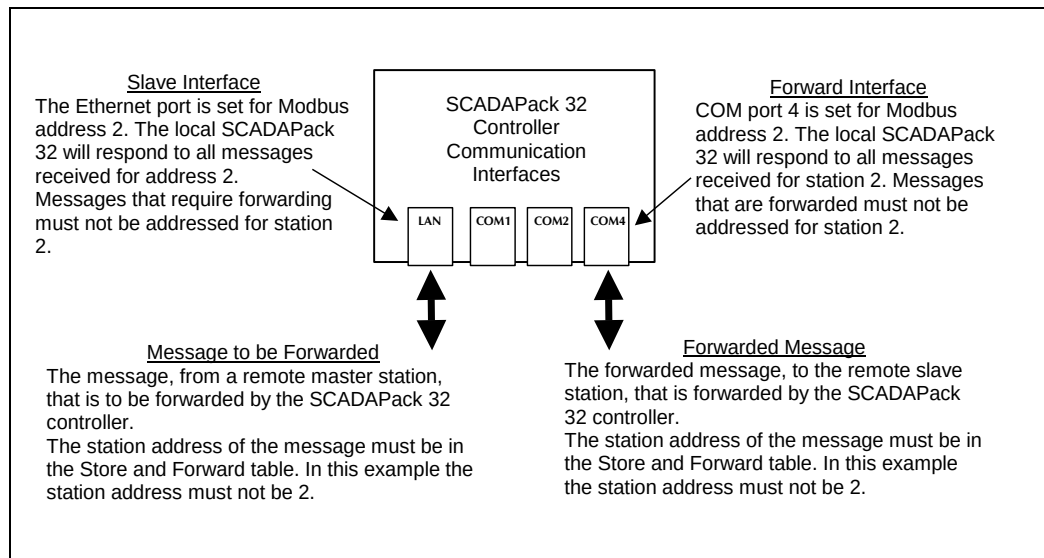


Figure 44: Store and Forward Command Reference

When the Store and Forward command is selected the Store and Forward dialog appears. This dialog displays the Store and Forward table for the controller.

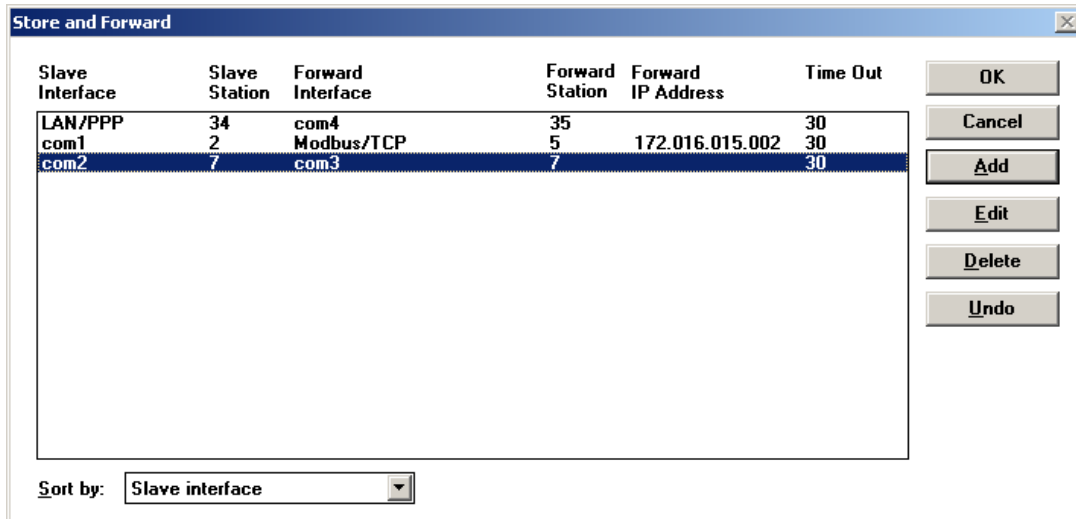


Figure 45: Store and Forward Dialog Box

The Store and Forward table displays each Store and Forward translation as a row, with column headings, in the table. The table may have up to 128 entries. A vertical scroll bar is used if the list exceeds the window size.

The **Slave Interface** heading displays the receiving slave interface the message is received from for each translation.

The **Slave Station** heading displays the Modbus station address of the slave message.

The **Forward Interface** heading displays the interface the message is forwarded from. When forwarding to a Modbus TCP or UDP network, the protocol type is selected for the Forward Interface. The IP Stack automatically determines the exact interface (e.g. LAN/PPP) to use when it searches the network for the Forward IP Address. If a serial port is selected for the Forward Interface, and the serial port is configured for PPP protocol, the message will not be forwarded.

The **Forward Station** heading displays the Modbus station address of the forwarded message.

The **Forward IP Address** heading displays the IP address of the Forward Station. This field is blank unless a TCP or UDP network is selected for Forward Interface.

The **Time Out** heading displays the maximum time (in tenths of seconds) the forwarding task waits for a valid response from the Forward Station. The time out should be equal to or less than the time out set for the master message received on the Slave Interface.

The **OK** button saves the table data. No error checking is done on the table data.

The **Cancel** button closes the dialog without saving changes.

Click **Add** to enter a new row in the store and forward table. Selecting the Add button opens the **Add/Edit Store and Forward** dialog.

Click **Edit** to modify the selected row in the store and forward table. Selecting the Edit button opens the **Add/Edit Store and Forward** dialog containing the data from the selected row. This button is disabled if more than one row is selected. This button is disabled if there are no entries in the table.

The **Delete** button removes the selected rows from the table. This button is disabled if there are no entries in the table.

The **Undo** button undoes the action performed by the last button selection since the dialog was opened. This applies to the buttons Add, Edit, Delete and Undo. This button is disabled when the dialog is opened, and is enabled as soon as any of the applicable buttons are selected.

The **Sorted by** menu box lists each of the five column headings. The rows are sorted according to the selected heading. Headings in the table are, by default, sorted by the Slave Interface heading.

4.4.6.1.1 Add/Edit Store and Forward Dialog

This dialog is used to edit an entry or add a new entry in the store and forward table.

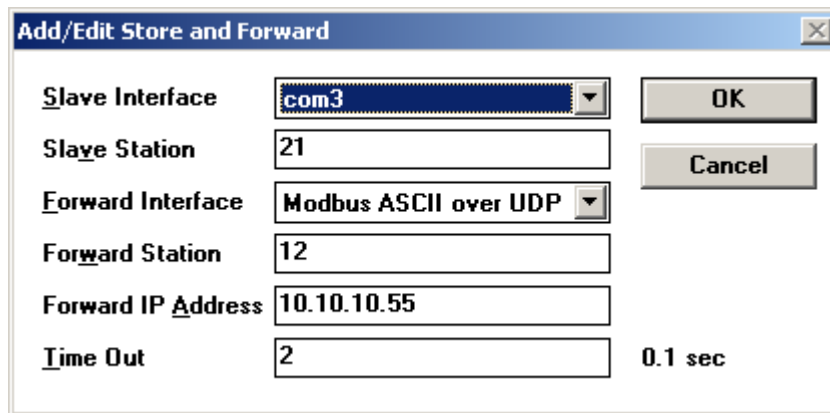
The image shows a Windows-style dialog box titled "Add/Edit Store and Forward". It contains several input fields and two buttons. The fields are: "Slave Interface" (a dropdown menu showing "com3"), "Slave Station" (a text box with "21"), "Forward Interface" (a dropdown menu showing "Modbus ASCII over UDP"), "Forward Station" (a text box with "12"), "Forward IP Address" (a text box with "10.10.10.55"), and "Time Out" (a text box with "2"). To the right of the "Time Out" field is the text "0.1 sec". On the right side of the dialog are two buttons: "OK" and "Cancel".

Figure 46: Add/Edit Store and Forward Dialog Box

The **Slave Interface** is the receiving slave interface the message received from. The drop down list allows the following selection:

- com1
- com2
- com3
- com4
- LAN/PPP

The **Slave Station** is the Modbus station address of the slave message. This address must be different from the Modbus address assigned to the Slave Interface. Valid range for Slave Station is:

1 to 255 when standard addressing is selected for the interface.

1 to 65534 when extended addressing is selected for the interface.

The **Forward Interface** is the interface the message is forwarded from. The drop down list allows the following selection:

- com1
- com2
- com3
- com4
- Modbus/TCP
- Modbus RTU in UDP

- Modbus ASCII in UDP

The **Forward Station** is the Modbus station address of the forwarded message. This address must be different from the Modbus address assigned to the Forward Interface. Valid range for Forward Station is:

1 to 255 when standard addressing is selected for the interface.

1 to 65534 when extended addressing is selected for the interface.

The **Forward IP Address** edit box is disabled and the address is forced to “0.0.0.0” whenever the Forward Interface is set to com1, com2, com3 or com4. The Forward IP Address edit box is enabled only when the Forward Interface is set to a TCP or UDP network. Valid entries are 0 to 255 for each byte in the IP address.

The **Time Out** is the maximum time the forwarding task waits for a valid response from the Forward Station, in tenths of second. Valid entries are 0 to 65535. The time out should be equal to or less than the time out set for the master message received on the Slave Interface.

The **OK** button checks the data for this table entry. If the data is valid the dialog is closed. If the table data entered is invalid, an error message is displayed and the dialog remains open. The table entry is invalid if any of the fields is out of range. The data is also invalid if it conflicts with another entry in the table.

The **Cancel** button closes the dialog without saving changes.

4.4.7 DNP

The DNP command is used to configure the DNP protocol settings for the controller. When selected the DNP Settings window is opened, as shown below.

For complete information on DNP configuration refer to the ***DNP 3 User and Reference Manual***.

4.4.8 DNP Status

The DNP Status command opens the DNP Status dialog. This dialog shows the run-time DNP diagnostics and current data values for the local DNP points.

For complete information on the DNP Status command refer to the ***DNP 3 User and Reference Manual***.

4.4.9 DNP Master Status

The DNP Status command opens the DNP Master Status dialog. This dialog shows the run-time DNP diagnostics and status of the DNP outstations defined in the Master Poll table and current data values for the DNP points in these outstations.

For complete information on the DNP Status command refer to the ***DNP 3 User and Reference Manual***.

4.4.10 Initialize

The initialize command is used to restore a controller to default settings. This is typically done when starting a new project with a controller. The Initialize controller dialog is displayed when this command is selected. The Initialize Controller dialog presented depends on the type of controller selected.

The dialog displayed depends on the controller type. Each dialog is described below.

4.4.10.1 SCADAPack and SCADAPack 32 Controllers

The Initialize Controller dialog shown below appears when the command is selected and the Controller Type is SCADAPack or SCADAPack 32 (not SCADAPack 330, SCADAPack 334, SCADAPack 350 or SCADASense 4203).

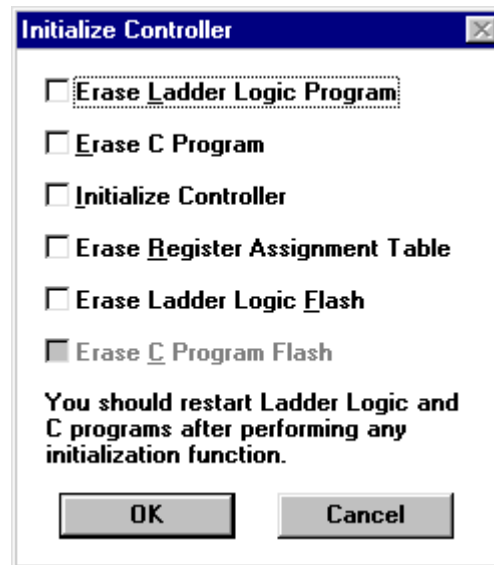


Figure 47: Initialize Controller Dialog Box SCADAPack and SCADAPack 32 Controllers

Check the **Erase Ladder Logic Program** selection to erase the ladder logic program in the controller RAM memory.

Check the **Erase C Program** selection to erase the C application program in the controller RAM memory.

Check the **Initialize Controller** selection to reset all values in the I/O database to default settings. The Initialize controller command will do the following:

- The default communication parameters are set for all serial ports. If you are communicating using settings other than the default, the PC Communication Settings will have to be changed once the command is complete. Note that the Ethernet port on SCADAPack 32 controllers is not changed by the initialize controller command.
- The registers in the I/O database are initialized to their default values.
- All forcing is removed.
- The controller is unlocked.

Check the **Erase Register Assignment** selection to erase the controller register assignments. The Register Assignment is cleared and must be reconfigured.

Check the **Erase Ladder Logic Flash** selection to erase the Ladder Logic program in Flash. This control is greyed if the controller firmware does not support programs in Flash or if the Flash is used by the operating system.

Check the **Erase C Program Flash** selection to erase the C program in Flash. This control is greyed if the controller firmware does not support programs in Flash or if the Flash is used by the operating system.

- Click **OK** to perform the requested initializations.
- Click **Cancel** to exit without performing any action.

Erasing Programs in Flash

Erasing the Flash memory requires the Ladder Logic and C programs be stopped. The following message is displayed if the OK button is selected.

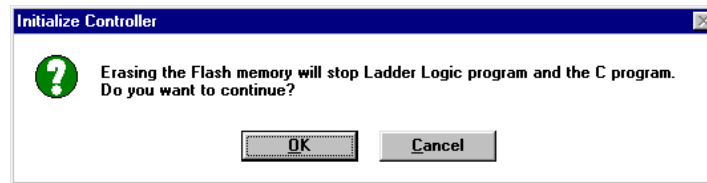


Figure 48: Initialize Controller Dialog Box

- Click **OK** to stop the execution and erase the flash memory. The following actions occur.

The Ladder Logic and C programs in the controller are stopped.

If it was selected, the Ladder Logic program is erased.

If it was selected, the C program is erased.

Programs that were stopped to allow Flash to be erased are not re-started. These programs must be re-started.

4.4.10.2 SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203 Series of Controllers

For SCADAPack 350 and SCADASense 4203 Series of controllers (4203 DR and 4203 DS), the Initialize dialog shown below appears when the command is selected.



Figure 49: Initialize Dialog for SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203 Series Controllers

Select the **Initialize All to Factory Settings** radio button to erase all programs in the controller memory and initialize the controller to factory settings. Please see *Actions Performed when Controller is Initialized* below for details. Erasing all programs includes all C/C++ programs, the Ladder Logic program in RAM, and the Ladder Logic program in ROM, if applicable.

Select the **Initialize** radio button to initialize only the selected items. The items that may be selected are as follows:

- Check the **Erase Ladder Logic Program and Register Assignment** selection to erase the ladder logic program and register assignment in the controller RAM memory.
- Check the **Erase All C/C++ Programs** selection to erase all C/C++ programs in the controller.
- Check the **Erase Ladder Logic Flash** selection to erase the Ladder Logic program in Flash (ROM).
- Check the **Initialize Controller** selection to initialize the controller to default settings. Please see *Actions Performed when Controller is Initialized* below for details. The default settings used depends on the radio button selected below this selection:
- Click on the **OK** button to perform the requested initializations.
- Click on the **Cancel** button to exit without performing any action.

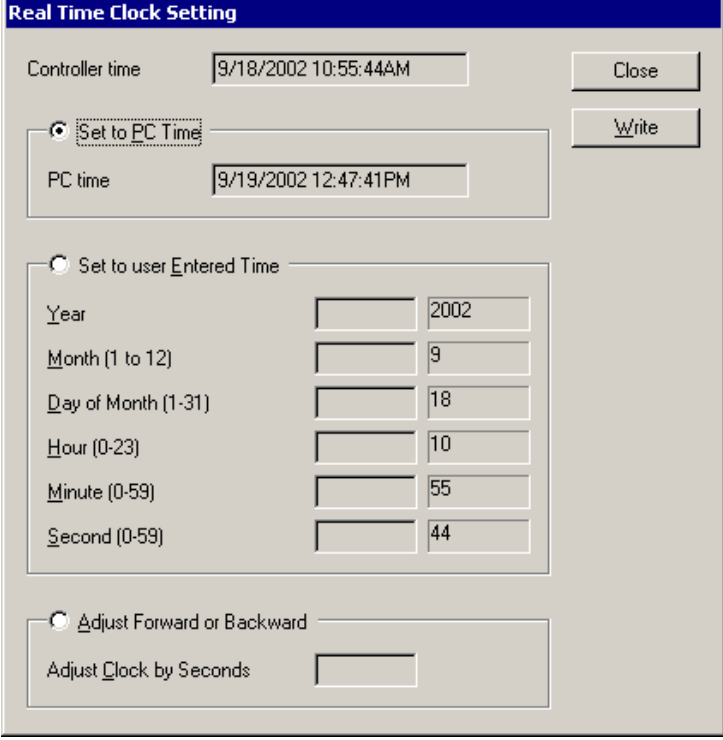
4.4.10.3 Actions Performed when Controller is Initialized

The following actions are performed when the controller is initialized:

- The controller settings below are set to default values. If you are communicating using settings other than the default, the PC Communication Settings will have to be changed once the command is complete. Note that the Ethernet port is not changed by the initialize controller command.
 - Serial port communication parameters
 - Modbus/IP and DNP/IP protocol settings
 - Outputs on Stop settings
 - HART modem configurations
 - LED power control
- All forcing is cleared.
- All Modbus registers are set to zero.
- If there is a register assignment, the assigned registers are initialized to their default values. For example, if a configuration module for a group of controller settings is in the register assignment, the assigned registers are initialized to the default values for those controller settings.
- All digital outputs and analog outputs are cleared.
- Data logged by all DLOG functions is erased.
- Alarm clock is cleared.
- Serial port event counters are cleared.
- Store and forward table is cleared.
- Friendly IP List is cleared.
- The power mode changes to full power.

4.4.11 Real Time Clock

The Real Time Clock command is used to set the controller Real Time Clock. The user may set the clock to the PC time, to a user specified time or adjust the clock forward or back by a number of seconds. The Real Time Clock Setting dialog shown below appears when the command is selected.



The dialog box is titled "Real Time Clock Setting" and has a blue header bar. It contains several controls for setting the controller's real-time clock. At the top, there is a text field labeled "Controller time" showing "9/18/2002 10:55:44AM" and a "Close" button. Below this is a radio button labeled "Set to PC Time" which is selected. To its right is a "Write" button. Below the radio button is a text field labeled "PC time" showing "9/19/2002 12:47:41PM". Further down is another radio button labeled "Set to user Entered Time". Below this radio button are six text fields for "Year", "Month (1 to 12)", "Day of Month (1-31)", "Hour (0-23)", "Minute (0-59)", and "Second (0-59)", with values 2002, 9, 18, 10, 55, and 44 respectively. At the bottom is a radio button labeled "Adjust Forward or Backward". Below this radio button is a text field labeled "Adjust Clock by Seconds".

Figure 50: Real Time Clock Setting Dialog Box

The following controls are available from the Real Time Clock Setting dialog.

Controller time shows the current time and date in the controller. It is updated continuously while the dialog is open. The time and date are displayed in the short time format as defined in the Windows Control Panel.

The **Set to PC Time** radio button selects setting the controller time to match the PC time. The current PC time and date are shown to the right of the button. The time and date are displayed in the short format as defined in the Windows Control Panel.

The **Set to User Entered Time** radio button selects setting the time and date to the values specified by the user in the **Year**, **Month**, **Day**, **Hour**, **Minute** and **Second** controls. The valid values for entry are shown in the dialog. If the Set to User Entered Time radio button is not selected these controls are grayed.

The **Adjust Forward or Backward** radio button selects adjusting the time by the number of seconds specified in the **Adjust Clock by Seconds** edit box. The value can be negative or positive. The edit box is grayed if the *Adjust by* radio button is not selected.

- The **Close** button closes the dialog.
- The **Write** button writes the selected time to the controller.

4.4.12 Monitor Element

The Monitor Element command is used to add registers used by selected elements to the Register Editor. All registers used by the currently selected element, or elements, are added to the Register Editor. This command is available only if an element, or multiple elements, is selected.

This command may be selected by right click your mouse button on a highlighted element, or group of elements, and selecting **Monitor Element** from the list of commands.

To add registers to the Register Editor using the Monitor Registers command first highlight an element by clicking the left mouse button on an element or dragging the cursor over a group of elements. Then select the Monitor Element command from the Controller menu or from the mouse right click menu.

When the Monitor Element command is selected the Monitor Element dialog is displayed.

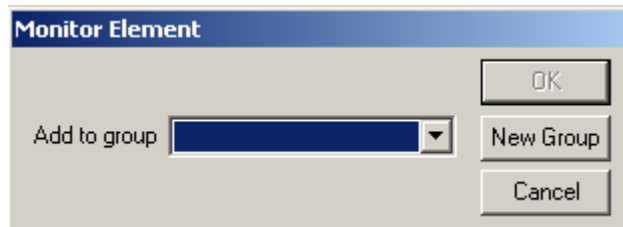


Figure 51: Monitor Element Dialog Box

The **Add to Group** list box selects which group the registers are added to. A drop down selection displays all created groups. If no groups have been created use the New Group button to add a Register Editor group.

The **OK** button adds the registers and closes the dialog. The registers are added to the Register Editor in the format used by the selected element or elements.

The Available status is set to **Online** for all registers except registers contained in elements that use Element Configuration.

The Available status is set to **Always** for all registers contained in elements that use Element Configuration.

The **Cancel** button closes the dialog without adding the registers to the Register Editor.

The **New Group** button opens New group dialog. This dialog is used to create a new group in the Register Editor.

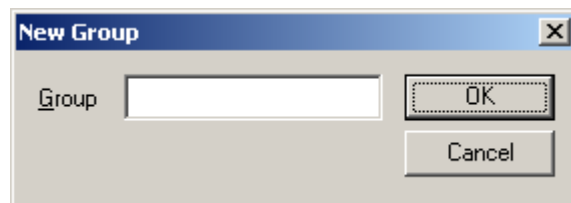


Figure 52: New Group Dialog Box

The **Group** edit box is used to enter the name of the new group. The group name may contain letters, numbers, and spaces. The group name may be 1 to 16 characters long and the name is not case sensitive.

The **OK** button creates the new group and closes the dialog. If the group name already exists an error message is displayed and the dialog remains open.

The **Cancel** button closes the dialog.

4.4.13 List Force Registers

The **List Forced Registers** command displays the Forced Registers dialog. This dialog displays a list of registers that are forced. The user may view and edit the registers, add registers to the list, remove registers from the list and clear all forces. The dialog appears as shown below.

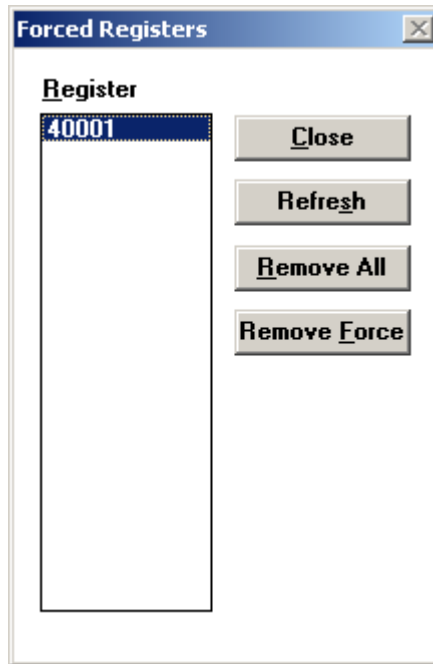


Figure 53: Forced Registers Dialog Box

The **Register** list box shows all registers in the controller that are currently forced.

Click **Close** to close the dialog box. Press **ESC** to select **Close**.

Selecting the **Remove All** button clears the force status for all registers. All registers are removed from the register list box. When selected the user is prompted for confirmation.

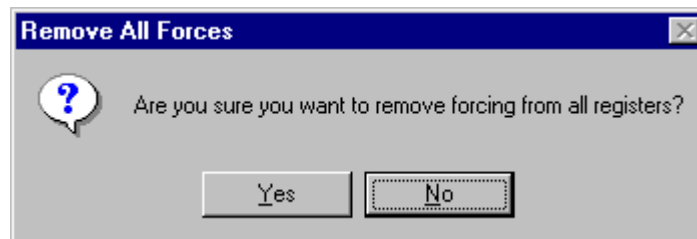


Figure 54: Remove All Forces Dialog Box

Selecting **Yes** removes all forcing. Selecting **No** aborts the command. The **No** button is the default selection. Press **ENTER** or **ESC** to select **No**. This button is grayed when there are no registers in the register list box.

The **Remove Force** button clears the force status for the selected register. The register is removed from the register list box. This button is grayed when there are no registers in the register list box.

4.4.14 Remove All Forces

Selecting the Remove All Forces command clears the force status for all forced registers. This command is only available in the on-line mode.

Selecting **Remove All Forces** prompts the user for confirmation:

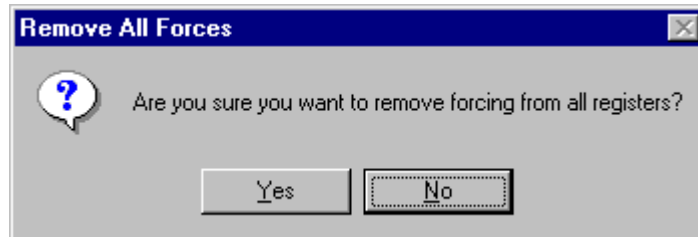


Figure 55: Remove All Forces Dialog Box

- Selecting **Yes** removes all forcing.
- Selecting **No** aborts the command. The **No** button is the default selection. Press **ENTER** or **ESC** to select **No**.

4.4.15 Lock Controller

Locking a controller prevents unauthorized access. Commands sent to the controller when it is locked will be rejected. A controller that is unlocked operates without restriction.

The Lock Controller dialog specifies a password to be used to lock the controller and the commands that are locked.

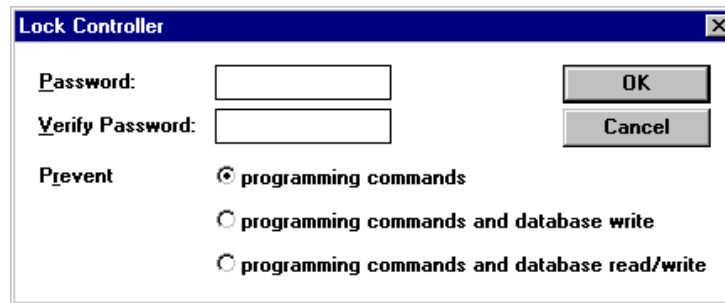


Figure 56: Lock Controller Dialog Box

Enter a password in the Password edit box. Re-enter the password in the Verify Password edit box. Any character string up to eight characters in length may be entered. Typing in these edit boxes is masked. An asterisk is shown for each character typed.

The Prevent radio buttons select the commands that are locked.

Locking the programming commands prevents modifying or viewing the program in the controller. Communication protocols can read and write the I/O database.

Locking programming and database write commands prevents modifying or viewing the program and prevents writing to the I/O database. Communication protocols can read data from the I/O database, but cannot modify any data.

Locking programming and database commands prevents modifying or viewing the program and prevents reading and writing the I/O database. Communication protocols cannot read or write the I/O database.

The **OK** button verifies the passwords are the same and sends the lock controller command to the controller. The dialog is closed. If the passwords are not the same an error message is displayed. Control returns to the dialog.

The **Cancel** button closes the dialog without any action.

If the controller is already locked, a message indicating this is shown instead of the dialog.

4.4.16 Unlock Controller

The Unlock Controller dialog prompts the user for a password to be used to unlock the controller. If the controller is locked, the following dialog is displayed.

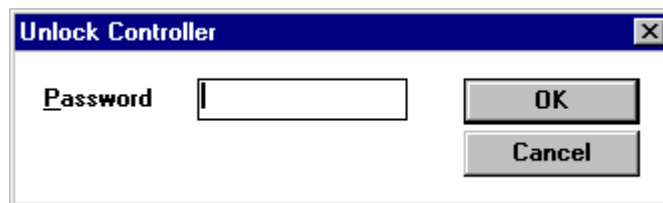


Figure 57: Unlock Controller Dialog Box

Enter the password that was used to lock the controller in the Password edit box. Typing in this edit box is masked. An asterisk is shown for each character typed.

The **Cancel** button closes the dialog without any action.

The **OK** button sends the *Unlock Controller* command to the controller. If the password is correct the controller will be unlocked. If the password is not correct, the controller will remain locked.

If you forget the controller password, the Override Controller Lock command can be used to unlock the controller. It will erase all programs in the controller.

4.4.17 Override Controller Lock

The Override Controller Lock dialog allows the user to unlock a controller without knowing the password. This can be used in the event that the password is forgotten.

To prevent unauthorized access to the information in the controller, the C and Ladder Logic programs are erased. Use this command with caution, as you will lose the programs in the controller.

Selecting the Override Controller Lock command displays the following dialog.

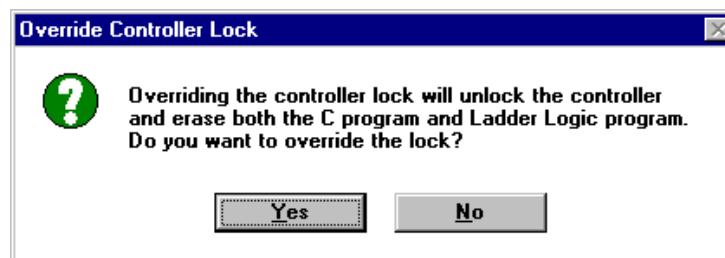


Figure 58: Override Controller Lock

- The **Yes** button unlocks the controller and erases all programs.

- The **No** button closes the dialog without any action.

4.4.18 Show Lock Status

The Show Lock Status command displays the controller lock state. It opens a dialog showing one of the following states:

- unlocked
- locked against programming commands
- locked against programming commands and database write
- locked against programming commands and database read/write
- The **OK** button closes the dialog.

4.4.19 C/C++ Program Loader

The C/C++ Program Loader command opens the C/C++ Program Loader dialog. This dialog allows the user to load, run, stop and delete C/C++ programs.

The C/C++ Program Loader dialog presented is the same for all controller types. For all controllers except the SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203, this dialog may be used to load, run, stop or delete just one C/C++ program.

4.4.19.1 SCADAPack and SCADAPack 32 Controllers

The dialog shown below appears when the C/C++ Program Loader command is selected from the controller menu and the Controller Type is either a SCADAPack32 or any one of the SCADAPack series of controllers (Micro16, SCADAPack, SCADAPack Light, SCADAPack Plus, SCADAPack 100 and SCADASense 4202 DR and DS) but **not** a SCADAPack 330, SCADAPack 334, SCADAPack 350 or SCADASense 4203.

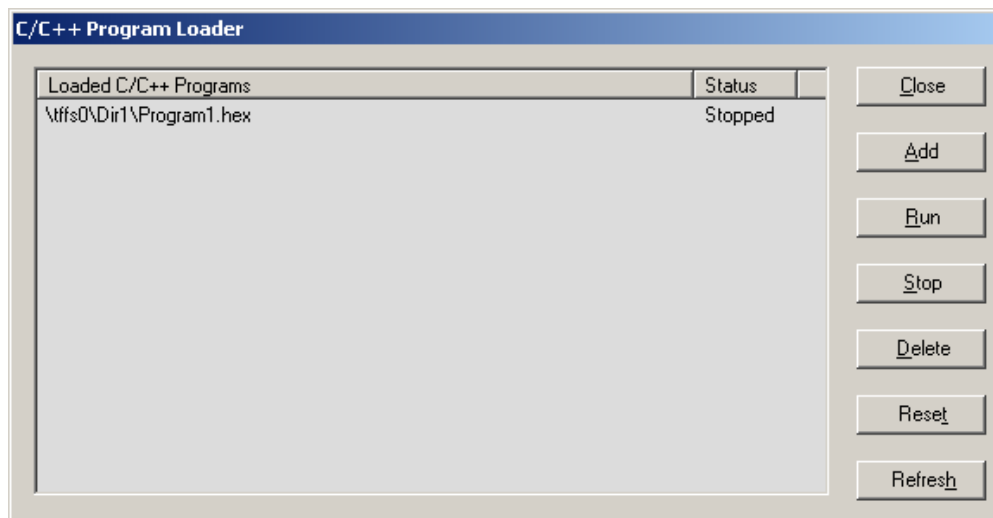


Figure 59: C/C++ Program Loader dialog for SCADAPack Series

The dialog displays the status of the C/C++ Program if one has been loaded in the controller. The status of the program is indicated as **Running** or **Stopped**. The list is empty if there is no C/C++ Program loaded in the controller.

The **C**lose button closes the dialog.

The **Add** button is used to write a C/C++ program to the controller. Selecting the Add button opens the **Add C/C++ Program** dialog.

Note: When using DNP communication between TelePACE and the target controller the DNP Application Layer timeout may need to be increased if a large C/C++ application is added. The default Application Layer timeout of 5 seconds may not be long enough.

The **Run** button stops and restarts the C/C++ program in the controller.

The **Stop** button stops the selected C/C++ program in the controller.

The **Delete** button stops and erases the selected C/C++ program in the controller.

The **Run**, **Stop** and **Delete** buttons are disabled if there is no C/C++ program loaded in the controller.

The **Reset** button resets the controller. A reset restarts the controller processor, the C/C++ program and the Ladder Logic program.

The **Refresh** button refreshes the status of the loaded C/C++ program.

4.4.19.1.1 Add C/C++ Program Dialog

The Add C/C++ Program dialog writes a C/C++ Program to the controller.

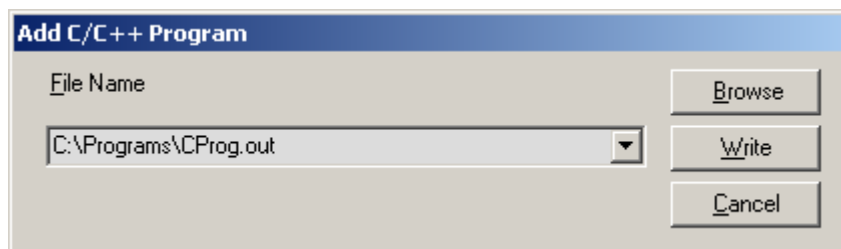


Figure 60: Write C/C++ Program dialog

File Name specifies the C/C++ Program to write to controller. The file name may be selected in a number of ways.

- Click on the **Browse** button to open a standard file open dialog.
- Use the drop-down menu to select the file from a list of previously written files.
- Type the path and file name directly into the edit box.

The **Write** button writes the selected file to the controller. The communication progress dialog box displays information about the write in progress, and allows you to cancel the write. If a C/C++ program is already loaded, it is stopped and erased before the selected file is written to the controller.

Note: When using DNP communication between TelePACE and the target controller the DNP Application Layer timeout may need to be increased if a large C/C++ application is added. The default Application Layer timeout of 5 seconds may not be long enough.

The **Cancel** button exits the dialog without writing to the controller.

4.4.19.2 SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203 Series Controllers

If the controller type is a SCADAPack 330, SCADAPack 334, SCADAPack 350 or a SCADASense 4203 Series controller, the C/C++ Program Loader dialog displayed is shown below. In contrast to

all other SCADAPack controllers, this dialog allows multiple C/C++ programs to be loaded, monitored, and controlled. The number of C/C++ programs that can be loaded into a SCADAPack 330, SCADAPack 334, SCADAPack 350 or a SCADASense 4203 Series controller depends on the memory capacity of RAM or Flash.

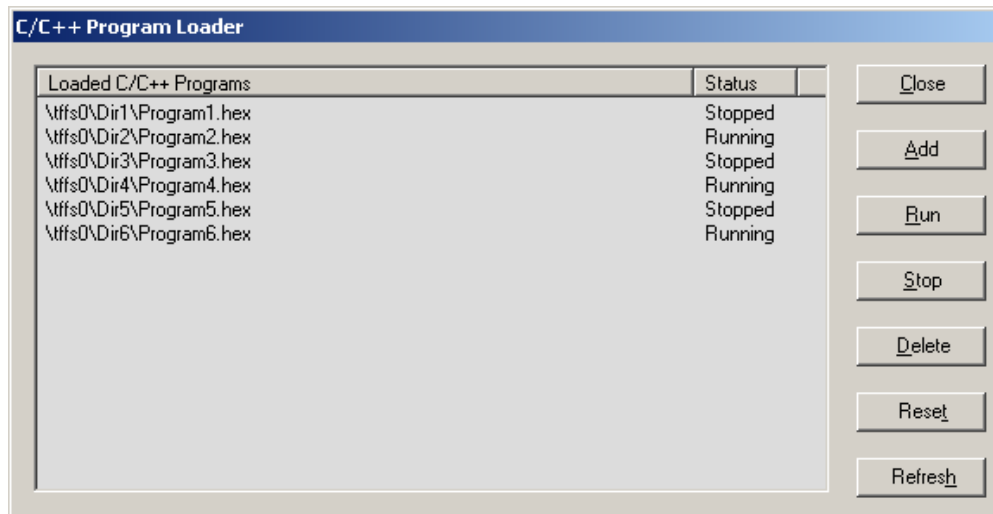


Figure 61: C/C++ Program Loader dialog

The dialog displays the C/C++ Programs that have been loaded in the controller. The status of each program is indicated as **Running** or **Stopped**.

The **C**lose button closes the dialog.

The **A**dd button writes a new C/C++ program to the controller. Selecting the Add button opens the **Add C/C++ Program** dialog. Refer to **4.4.19.1.1-Add C/C++ Program Dialog** for a description of the Add C/C++ program dialog.

The **R**un, **S**top and **D**ele~~t~~e buttons apply to the C/C++ program selected from the list of loaded C/C++ programs. These buttons are disabled when there are no C/C++ programs loaded.

The **R**un button stops and restarts the selected C/C++ program in the controller.

The **S**top button stops the selected C/C++ program in the controller.

The **D**ele~~t~~e button stops and erases the selected C/C++ program in the controller.

The **R**eset button resets the controller. A reset restarts the controller processor, all C/C++ programs and the Ladder Logic program.

The **R**efresh button refreshes the list of loaded programs and their status.

Click on the column headings to sort the list by that column. Click a second time to reverse the sort order.

4.4.20 Flash Loader

The Flash Loader command opens the Flash Loader dialog. This dialog is used to write Ladder Logic and C/C++ programs into Flash memory. This memory is permanently programmed and does not require power or a backup battery. The programs in Flash can be written through all supported communication media.

When this command is selected TelePACE checks if the controller supports Flash memory and, if a Flash memory chip is installed.

If the controller does not support Flash memory, the message “Controller does not support Flash memory” is displayed.

If the controller supports flash memory but there is no Flash memory installed in the application ROM socket, the message “Flash memory is not installed in the application socket (U14)” is displayed. Flash memory will have to be installed in the controller.

If there is Flash memory installed in the application program socket U14, the Flash Loader dialog is displayed as shown below.

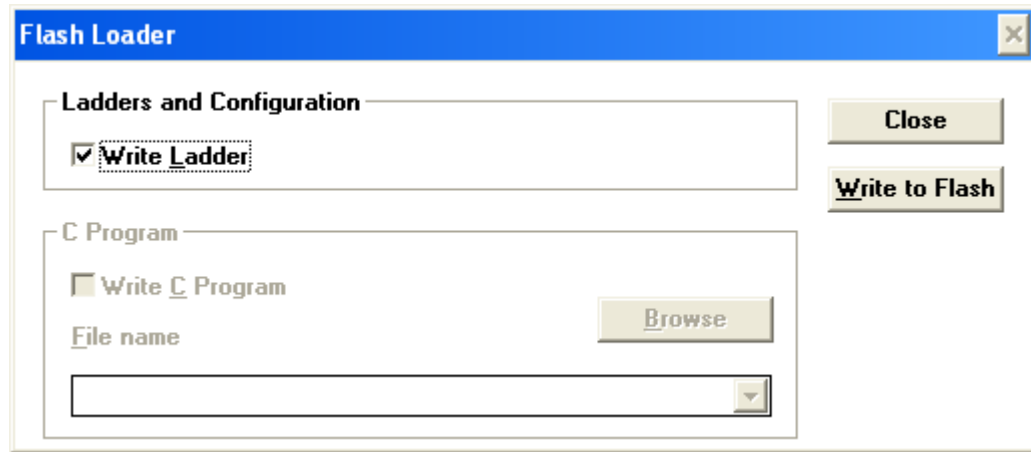


Figure 62: Flash Loader Dialog Box

The **Write Ladder** check-box controls writing of the Ladder Logic program and configuration data. Configuration data includes serial port settings, register assignment, outputs on stop settings and, element configuration. The file currently open Ladder Logic program in TelePACE will be written. This control is disabled if the controller has operating system code in the Ladder Logic section of Flash.

The **Write C Program** check-box controls writing of the C/C++ programs. This area is greyed out if the controller type is one of the following:

- SCADAPack 330
- SCADAPack 334
- SCADAPack 350
- SCADAPack 32 or SCADAPack 32P
- SCADAPack
- SCADASense 4202 DR and DS with firmware 1.65 or newer
- SCADASense 4203 DR and DS

The following controls select which file will be written. These controls are disabled if the controller has operating system code in the C Program section of Flash.

The **File name** edit-box contains the full path name of the file to be written.

The **Browse** button opens standard file open dialog. It displays files of type *.abs. Selecting a file fills in the File name edit-box.

The **Message Length** edit-box selects the maximum message length to be used while writing the programs to the controller. Control over message length is needed when writing large amounts of data over certain communication networks. The C program is written more quickly with a longer message length. The allowable range is 26 to 255 bytes. The default value is 255.

- The **Close** button closes the dialog.
- The **Write to Flash** button writes the selected programs to Flash memory in the controller.

4.4.20.1 Writing Programs to Flash

Programming the Flash memory requires the Ladder Logic and C programs be stopped. The following message is displayed when the Write to Flash button is selected.

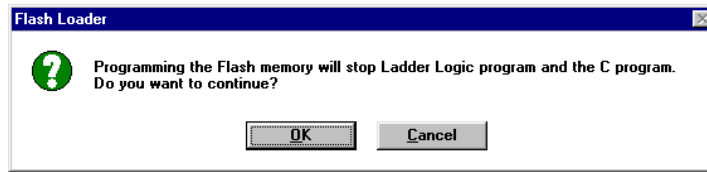


Figure 63: Flash Loader Dialog Box

- Click **OK** to stop the program execution and proceed with writing the program to Flash.
- Click **Cancel** to stop the write to Flash operation.

Ladder Logic and C application programs can both be programmed into RAM and into Flash.

If there is a Ladder Logic program in RAM, the following message is displayed.

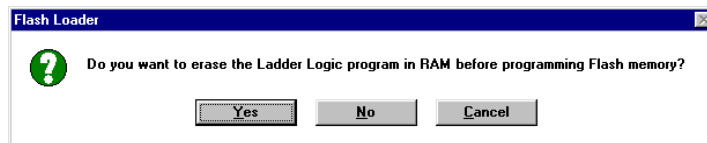


Figure 64: Flash Loader Dialog Box

- Click **Yes** to erase the RAM program and continue with the write to Flash.
- Click **No** to leave the RAM program in its place and continue with the write to Flash.
- Click **Cancel** to stop the entire operation.

If there is a C program in RAM, the following message is displayed.

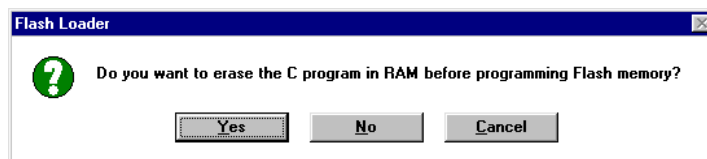


Figure 65: Flash Loader Dialog Box

- Click **Yes** to erase the RAM program and continue with the write to Flash.
- Click **No** to leave the RAM program in its place and continue with the write to Flash.

- Click **Cancel** to stop the entire operation

During the write to Flash operation a communication progress dialog is opened displaying the status of the write operation. Click on the Cancel button to abort the write to Flash at any time. The following actions occur during the write to Flash.

- The programs in the controller are stopped.
- The programs in RAM are erased if required.
- The Flash is erased if required.
- If it was selected, the Ladder Logic program is written to the controller.
- If it was selected, the C program is written to the controller.

If a Ladder Logic program is loaded in RAM or Flash in the controller, a prompt is displayed asking if the program should be started. If there is a Ladder Logic program in RAM the prompt asks if the Ladder Logic program in RAM should be started. If there is a Ladder Logic program in Flash the prompt asks if the Ladder Logic program in Flash should be started.

- Click **Yes** to run the Ladder Logic program. This will happen immediately.
- Click **No** to leave the program stopped.

If a C program is loaded in RAM or Flash in the controller, a prompt is displayed asking if the program should be started. If there is a C program in RAM the prompt asks if the C program in RAM should be started. If there is a C program in Flash the prompt asks if the C program in Flash should be started.

- Click **Yes** to run the C Program. This will happen immediately.
- Click **No** to leave the C Program stopped.

4.4.21 Program Status

The Program Status dialog displays information about programs loaded in the controller memory. The Program Status dialog is displayed when this command is selected. The Program Status dialog displayed depends on the controller type.

4.4.21.1 SCADAPack and SCADAPack 32 Controllers

Programs in RAM memory always take precedence over programs in Flash or EPROM memory. When a command to run a program is received, the controller first checks for a program in RAM, then for one in Flash or EPROM.

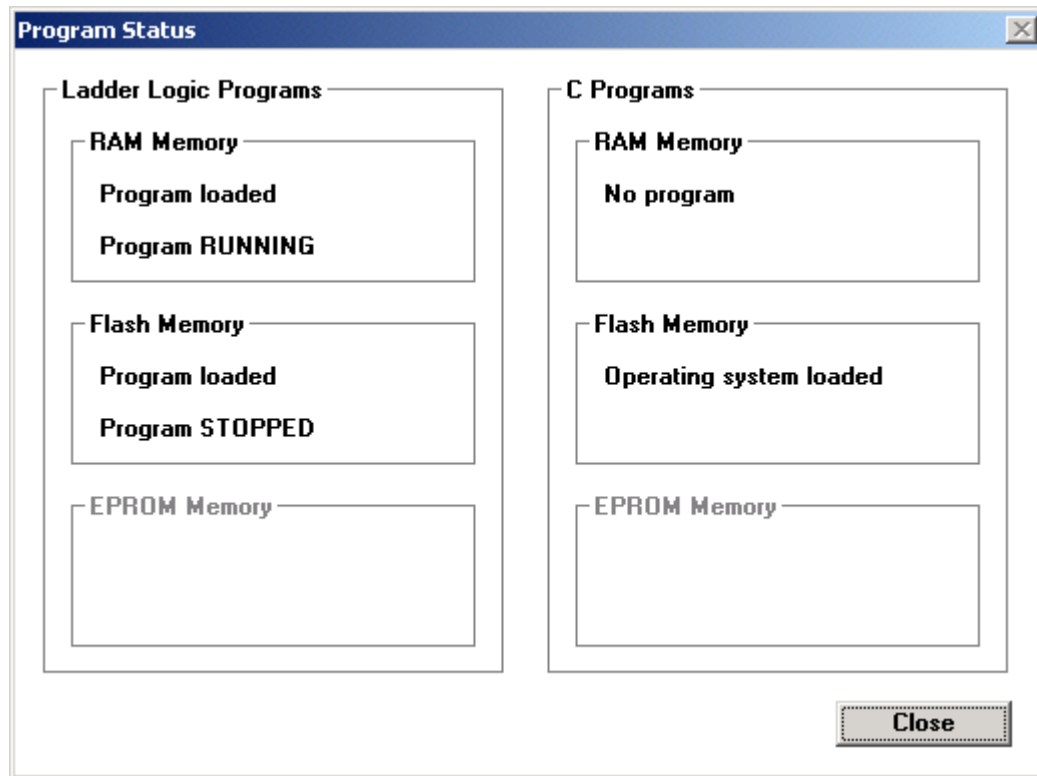


Figure 66: Program Status Dialog Box SCADAPack and SCADAPack 32 Controllers

The **Ladder Logic Programs** section shows information about the Ladder Logic programs.

The **RAM Memory** section shows if a program is loaded and the execution state of the program.

The **Flash Memory** section shows if a program is loaded and the execution state of the program. The program in Flash will be running only if there is no program in RAM. This section is greyed if the controller does not have Flash memory installed.

If the controller has operating system code in the Ladder Logic section of Flash then the Flash Memory or EPROM Memory section displays “Operating System”.

The **EPROM Memory** section shows if a program is loaded and the execution state of the program. This section is greyed if the controller has Flash memory installed.

The **C Programs** section shows information about the C programs.

The **RAM Memory** section shows if a program is loaded and the execution state of the program.

The **Flash Memory** section shows if a program is loaded. This section is greyed if the controller does not have Flash memory installed. If the controller type is SCADAPack 32 or SCADAPack 32P the section displays “Operating System Loaded”.

If the controller has operating system code in the C Program section of Flash then the Flash Memory or EPROM Memory section displays “Operating System”.

The **EPROM Memory** section shows if a program is loaded. This section is greyed if the controller has Flash memory installed.

The **Close** button closes the dialog.

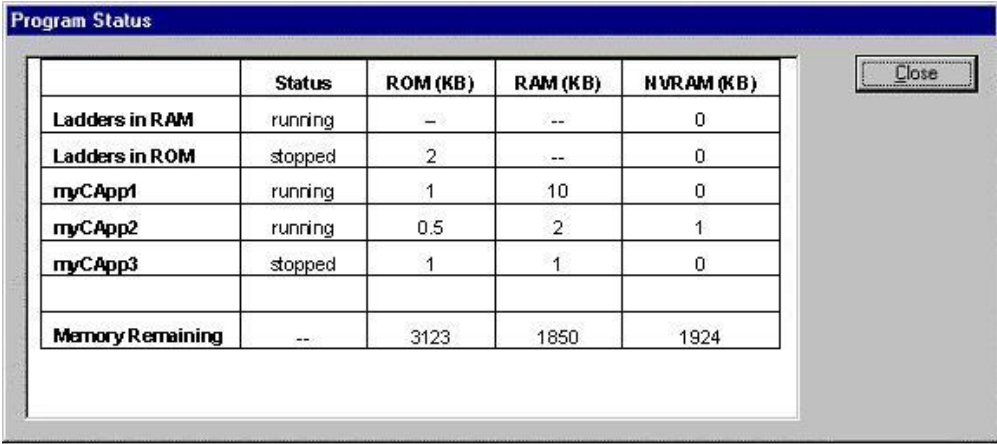
The table below shows the messages that are displayed for the various conditions.

Condition	Message
A program was found in the memory	Program loaded
No program was found in the memory	No program
Ladder Logic mode is running	Program RUNNING
Ladder Logic mode is debug	Program RUNNING in DEBUG
Ladder Logic mode is stopped	Program STOPPED

Table 6: Program Status Settings

4.4.21.2 SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203 Series of Controllers

When the controller type is SCADAPack 330, SCADAPack 334, SCADAPack 350 or a SCADASense 4203 series of controllers, the Program Status dialog displays the current status of the Ladder Logic program and all C/C++ programs loaded in controller as shown in the diagram below.



	Status	ROM (KB)	RAM (KB)	NVRAM (KB)
Ladders in RAM	running	--	--	0
Ladders in ROM	stopped	2	--	0
myCApp1	running	1	10	0
myCApp2	running	0.5	2	1
myCApp3	stopped	1	1	0
Memory Remaining	--	3123	1850	1924

Figure 67: Program Status Dialog for SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203 Series Controllers

The status may be one of the following:

- **Stopped** indicates the program has been downloaded and is currently stopped.
- **Running** indicates the program has been downloaded and is currently running.
- **Debug** indicates the program has been downloaded and is currently running in debug mode (Ladder Logic program only).
- **No Program** indicates that there is no program in the controller.

The memory used by each program is displayed in addition to the program status.

- The **ROM (KB)** column describes the amount of memory used in ROM by each program.
- The **RAM (KB)** column describes the amount of memory used in RAM by each program.
- The **NVRAM (KB)** column describes the amount of memory used in non-volatile RAM by each program.
- The **Memory Remaining** row describes the unused memory of each type available to all programs.

The Ladder Logic program in RAM memory always takes precedence over a Ladder Logic program in ROM. When a command to run a program is received, the controller first checks for a program in RAM, then for one in ROM. A Ladder Logic program in ROM may only be run if there is no Ladder Logic program in RAM.

The **Close** button closes the dialog.

4.5 Communications Menu

Communication menu commands are used to transfer programs between the Host PC and the controller. PC Communication settings for the host PC are configured in this menu selection.

4.5.1 Read from Controller

The Read from controller command is used to read the ladder logic program, register assignment and serial port settings from a controller into the TelePACE program. The program replaces the current program in the ladder editor. The communication progress dialog box displays information about the read in progress, and allows you to Start or Cancel the read. Click **Cancel** to abort a read in progress.

4.5.2 Write to Controller

The Write to controller command writes the ladder logic program, register assignment and serial port settings in the TelePACE program to the controller. The program replaces the current program in the controller. The communication progress dialog box displays information about the write progress, and allows you to Start or Cancel the write. Click **Cancel** to stop a write in progress.

If the program in the controller is executing, a dialog box appears to request the next action. Press **Stop** to stop execution of the ladder logic program when the write is complete. Press **Continue** to continue execution of the new program after the write is complete. Press **Cancel** to abort the write. TelePACE stops the program in the controller before writing the new program to the controller.

Programs that are executing in the target controller are paused momentarily when commands are written. The length of the pause depends on the command written.

4.5.3 PC Communications Settings

The PC Communication Settings command defines the communication protocol and communication link used for communication between the personal computer (PC) and SCADAPack or 4202 series controllers.

When the command is select the Communication Protocols Configuration dialog is displayed as shown below.

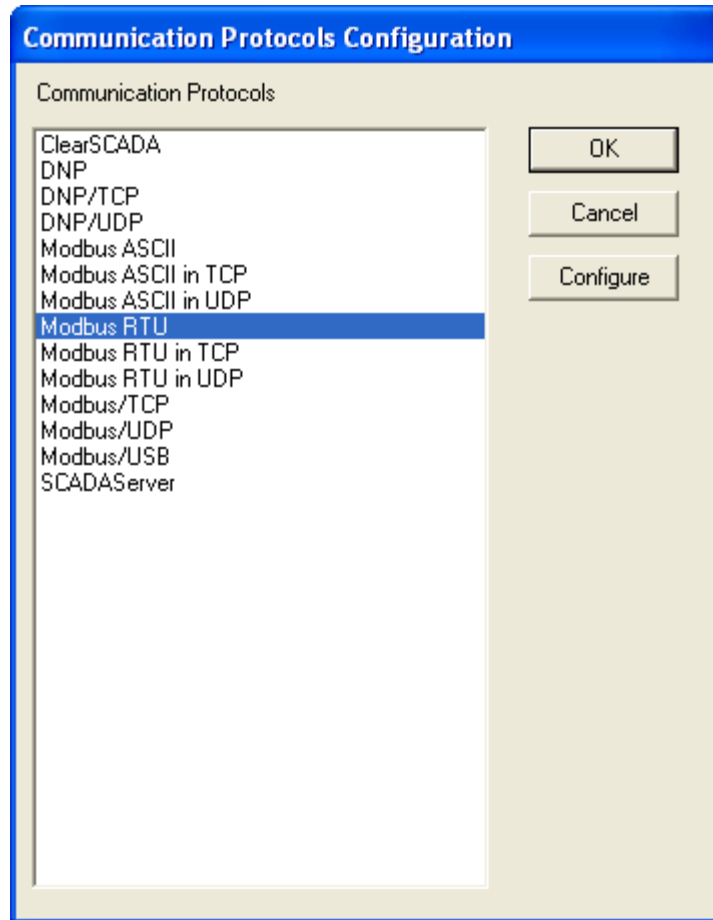


Figure 68: Communication Protocols Configuration dialog.

4.5.3.1 ClearSCADA

The ClearSCADA protocol driver is used for communicating with a local or remote ClearSCADA server. The ClearSCADA server will then, in turn, communicate with devices as per its configuration. The ClearSCADA protocol driver communicates with the ClearSCADA server using a TCP connection.

- To configure a ClearSCADA protocol connection, highlight **ClearSCADA** in the Communication Protocols window and click the **Configure** button. The ClearSCADA Configuration window is displayed.
- To select a configured ClearSCADA protocol connection, highlight **ClearSCADA** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.1.1 General Parameters

When ClearSCADA protocol is selected for configuration the ClearSCADA Configuration dialog is opened with the General tab selected as shown below.

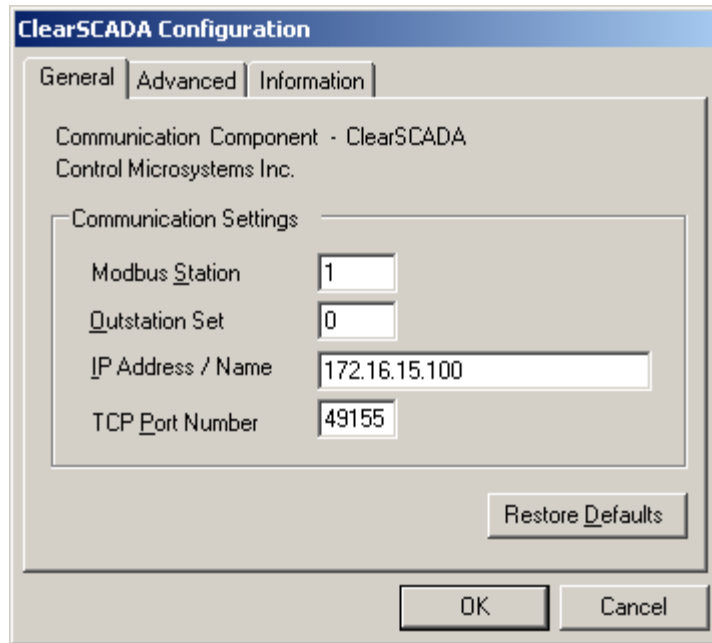


Figure 69: ClearSCADA Configuration (General) Dialog Box

The General tab component information section contains the name of Communication Component and the author, Control Microsystems.

The **Communications Settings** grouping contains all the essential details necessary to establish communication to a device through a local or remote ClearSCADA installation.

The **Modbus Station** entry specifies the station address of the target device. Valid values are 1 to 65534.

The **Outstation Set** entry specifies the ClearSCADA outstation set to which the target device is attached. The valid range is 0 to 65535. The default value is 0.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **TCP Port Number** entry specifies the TCP port on the **ClearSCADA** server. The valid range is 0 to 65535. The default value is 49155

- Click **Restore Defaults** to restore default values to all fields on this page, except for the **IP Address / Name** field. The contents of this field will remain unchanged.

4.5.3.1.2 Advanced Parameters

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A

smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

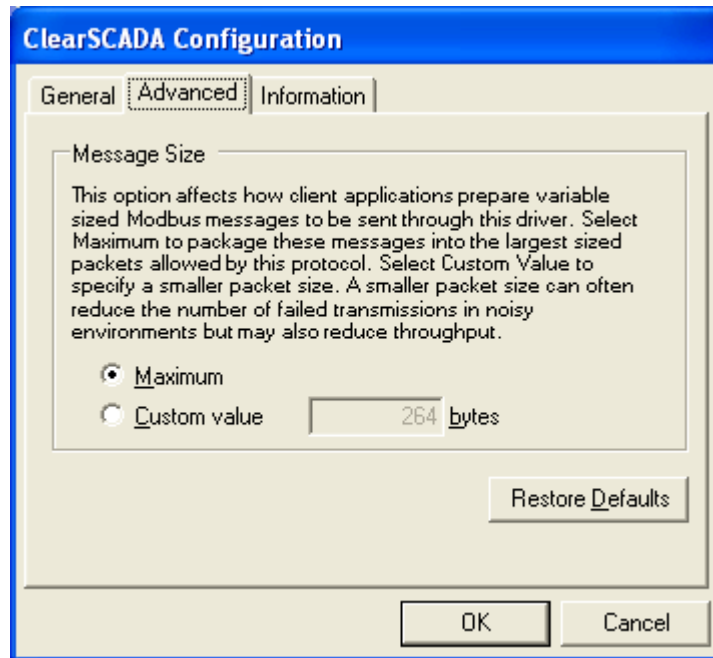


Figure 70: ClearSCADA Configuration (Advanced) Dialog Box

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value will indicate to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 264. The default value is 264.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.1.3 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

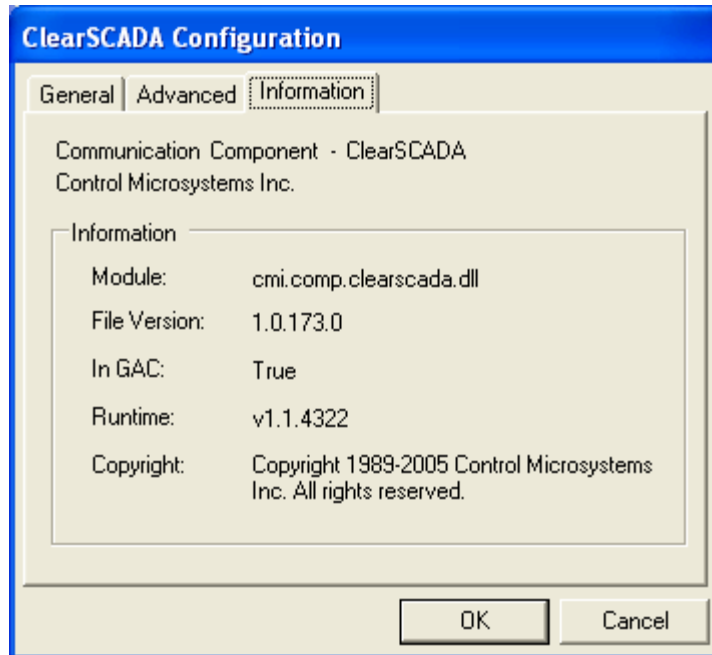


Figure 71: ClearSCADA Configuration (Information) Dialog Box

4.5.3.1.3.1 Information

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

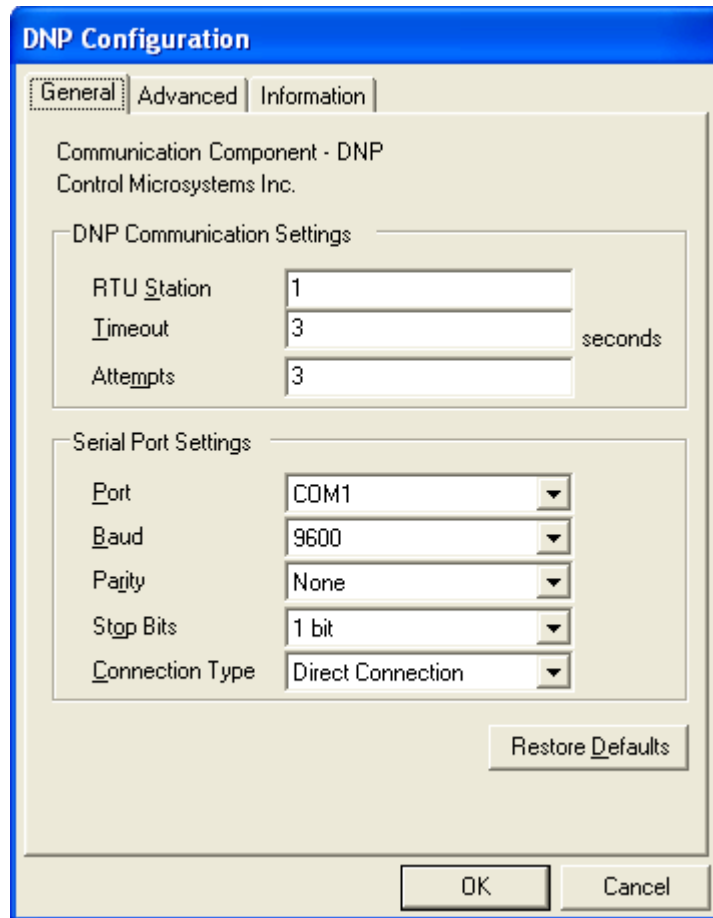
4.5.3.2 DNP

The DNP protocol driver is used to communicate over a serial DNP network to SCADAPack controllers configured for DNP communication.

- To configure a DNP protocol connection, highlight **DNP** in the Communication Protocols window and click the **Configure** button. The DNP Configuration window is displayed.
- To select a configured DNP protocol connection, highlight **DNP** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.2.1 General Parameters

When DNP is selected for configuration the DNP Configuration dialog is opened with the General tab selected as shown below.



The image shows a 'DNP Configuration' dialog box with a blue title bar. It has three tabs: 'General' (selected), 'Advanced', and 'Information'. The 'General' tab contains the following sections:

- Communication Component - DNP**
Control Microsystems Inc.
- DNP Communication Settings**
 - RTU Station: 1
 - Timeout: 3 seconds
 - Attempts: 3
- Serial Port Settings**
 - Port: COM1
 - Baud: 9600
 - Parity: None
 - Stop Bits: 1 bit
 - Connection Type: Direct Connection

At the bottom right of the dialog is a 'Restore Defaults' button. At the very bottom are 'OK' and 'Cancel' buttons.

Figure 72: DNP Configuration (General) Dialog Box

The General tab component information section contains the name of Communication Component and the author, Control Microsystems.

The **DNP Communication Settings** logical grouping contains DNP specific communication settings including the DNP Station address, the timeout interval as well as the number of attempts.

The **RTU Station** parameter sets the target **DNP** station number. Valid entries are 0 to 65519. The default address is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts), or ultimately failing. Valid entries are 1 to 255. The default is 3.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid entries are 1 to 20. The default is 3.

This **Serial Port Settings** grouping contains details directly related to the PC's communication port including the port number, the baud rate, parity and stop bit settings.

The **Port** parameter specifies the PC serial port to use. The DNP driver determines what serial ports are available on the PC and presents these in the drop-down menu list. The available serial ports list will include any USB to serial converters used on the PC. The default value is the first existing port found by the driver.

The **Baud** parameter specifies the baud rate to use for communication. The menu list displays selections for 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, and 57600. The default value is 9600.

The **Parity** parameter specifies the type of parity to use for communication. The menu list displays selections for none, odd and even parity. The default value is None.

The **Stop Bits** parameter specifies the number of stop bits to use for communication. The menu list displays selections for 1 and 2 stop bits. The default value is 1 bit.

The **Connection Type** parameter specifies the serial connection type. The DNP driver supports direct serial connection with no flow control, Request-to-send (RTS) and clear-to-send (CTS) flow control and PSTN dial-up connections. The menu list displays selections for Direct Connection, RTS/CTS Flow Control and Dial Up Connection. The default selection is Direct Connection.

- Select **Direct Connection** for RS-232 for RS-485 connections that do not require the hardware control lines on the serial ports.
- Select **RTS/CTS Flow Control** to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking. Selecting RTS/CTS Flow Control adds a new tab, Flow Control, to the DNP Configuration dialog. Refer to the Flow Control Parameters section below for configuration details.
- Select **Dial Up Connection** to communication over dial up modems. Selecting Dial Up Connection adds a new tab, Dial Up, to the DNP Configuration dialog. Refer to the Dial Up Parameters section below for configuration details.
- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.2.2 Flow Control Parameters

Flow Control parameters are used to configure how RTS and CTS control is used. When RTS/CTS Flow Control is selected for Connection Type the Flow Control tab is added to the DNP Configuration dialog. When the Flow Control tab heading is clicked the Flow Control dialog is opened as shown below.

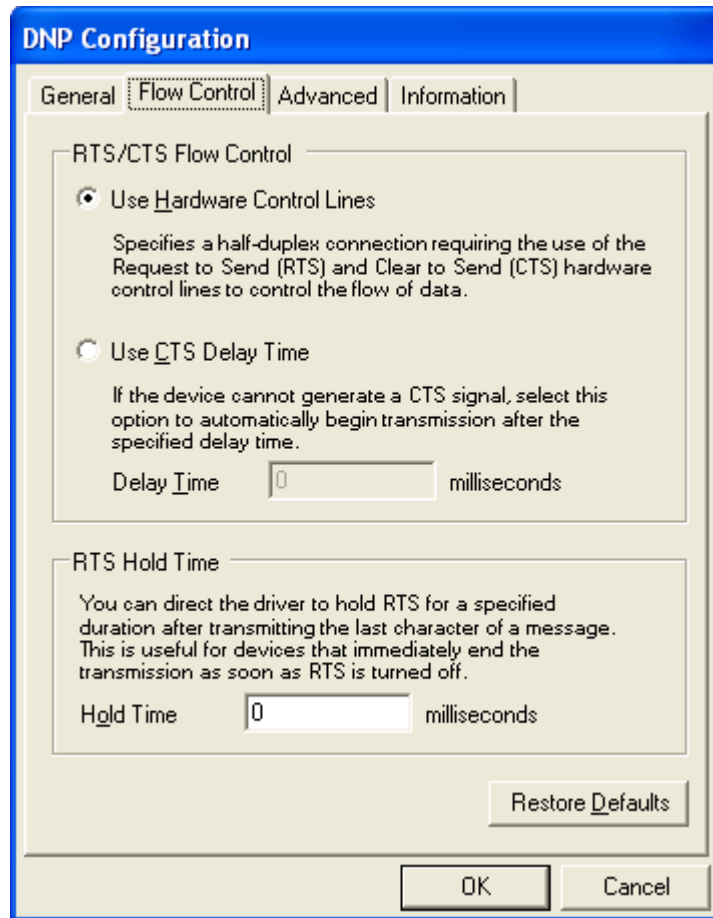


Figure 73: DNP Configuration (Flow Control) Dialog Box

4.5.3.2.2.1 RTS/CTS Flow Control

The **RTS/CTS Flow Control** grouping contains two mutually exclusive options, **Use Hardware Control Lines** and **Use CTS Delay Time**. These options enable the driver to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking.

The **Use Hardware Control Lines** option specifies a half-duplex connection requiring the use of the Request to Send (RTS) and Clear to Send (CTS) hardware control lines to control the flow of data. This selection is used with radios and dedicated telephone line modems. The driver turns on the RTS signal when it wants to transmit data. The modem or other device then turns on CTS when it is ready to transmit. The driver transmits the data, and then turns off the RTS signal. This selection is mutually exclusive of the **Use CTS Delay Time** selection described below. This is the default selection.

The **Use CTS Delay Time** option is selected if the device cannot generate a CTS signal. The driver will assert RTS then wait the specified Delay Time, in milliseconds, before proceeding. This option is mutually exclusive with the **Use Hardware Control Lines** selection described above.

The **Delay Time** parameter sets the time in milliseconds that the driver will wait after asserting RTS before proceeding. The value of this field must be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

The **Hold Time** parameter specifies the time, in milliseconds, that the driver will hold RTS after the last character is transmitted. This is useful for devices that immediately end transmission when RTS is turned off. The value of this field must be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.2.3 Dial Up Parameters

Dial Up parameters are used to configure a dial up connection. When Dial Up is selected for Connection Type the Dial Up tab is added to the DNP Configuration dialog. When the Dial Up tab heading is clicked the Dial Up dialog is opened as shown below.

Figure 74: DNP Configuration (Dial Up) Dialog Box

The **Dialing Prefix** parameter specifies the commands sent to the modem before dialing. A maximum of 32 characters can be entered. All characters are valid. The default value is “&F0 &K0 S0=1 &W0 &Y0”.

The **Phone Number** parameter specifies the telephone number of the remote controller. A maximum of 32 characters can be entered. All characters are valid. This field’s default value is blank.

The **Dial Type** parameter specifies the dialing type. Valid values are Pulse and Tone. The default value is Tone.

The **Dial Attempts** parameter specifies how many dialing attempts will be made. Valid values are 1 to 10. The default value is 1.

The **Connect Time** parameter specifies the amount of time in seconds the modem will wait for a connection. Valid values are 6 to 300. The default value is 60.

The **Pause Time** parameter specifies the time in seconds between dialing attempts. Valid values are 6 to 600. The default value is 30.

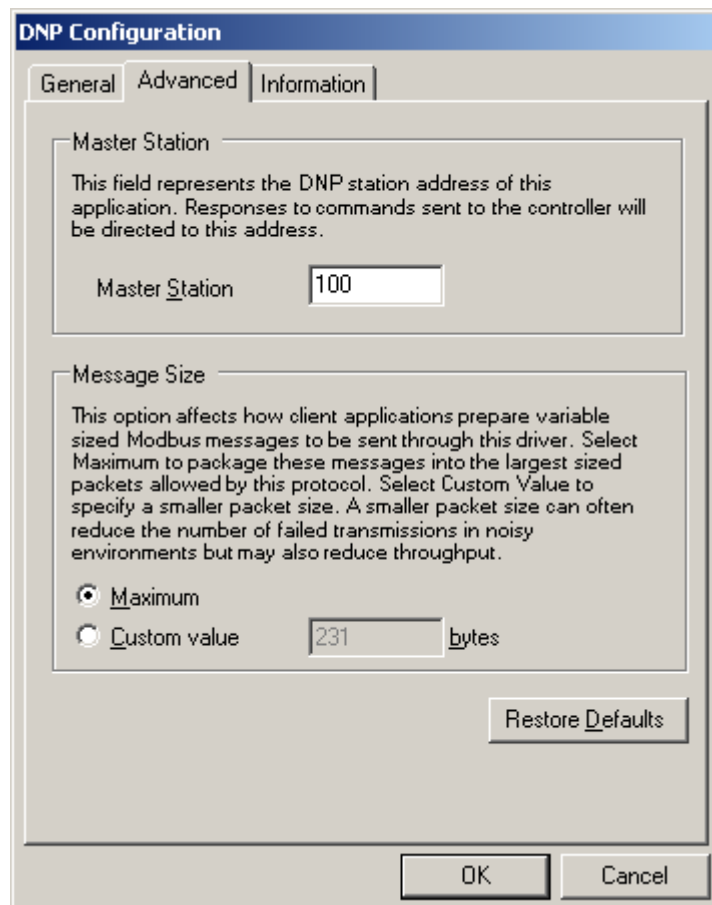
Check the **Inactivity Timeout** check box to automatically terminate the dialup connection after a period of inactivity. The Inactivity Time edit box is enabled only if this option is checked. The default state is checked.

Enter the inactivity period, in minutes, in the **Inactivity Timeout** box. The dialup connection will be terminated automatically after the specified number of minutes of inactivity has lapsed. This option is only active if the Inactivity Timeout box is checked. Valid values are from 1 to 30 minutes. The default value is 1.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the Phone Number field. The content of this field will remain unchanged.

4.5.3.2.4 Advanced Parameters

DNP Configuration Advanced parameters set the DNP master station address and message size control. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.



The image shows a dialog box titled "DNP Configuration" with three tabs: "General", "Advanced", and "Information". The "Advanced" tab is selected. The dialog contains two main sections: "Master Station" and "Message Size".

Master Station: A text box labeled "Master Station" contains the value "100". Above the text box is a description: "This field represents the DNP station address of this application. Responses to commands sent to the controller will be directed to this address."

Message Size: A text box labeled "Message Size" contains the value "231" followed by "bytes". Above the text box is a description: "This option affects how client applications prepare variable sized Modbus messages to be sent through this driver. Select Maximum to package these messages into the largest sized packets allowed by this protocol. Select Custom Value to specify a smaller packet size. A smaller packet size can often reduce the number of failed transmissions in noisy environments but may also reduce throughput."

Below the "Message Size" section are two radio buttons: "Maximum" (selected) and "Custom value".

At the bottom right of the dialog is a button labeled "Restore Defaults". At the very bottom are "OK" and "Cancel" buttons.

Figure 75: DNP Configuration (Advanced) Dialog Box

The **Master Station** parameter is the DNP station address assumed by this communication component. When this driver sends out commands, responses from the controller will be directed to this address. The default value is 100.

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 231. The default value is 231.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.2.5 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

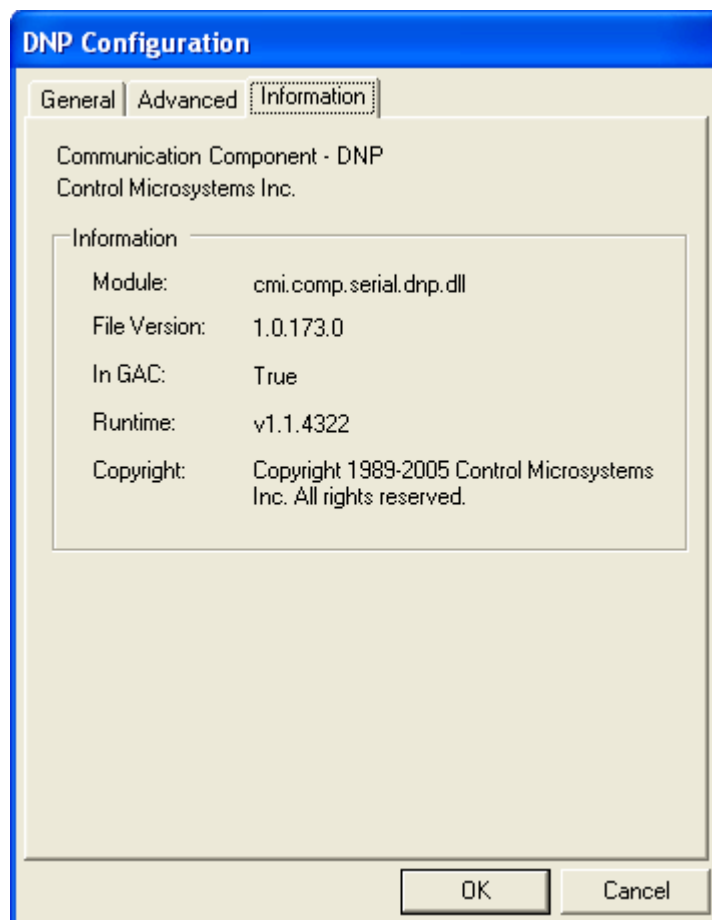


Figure 76: DNP Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

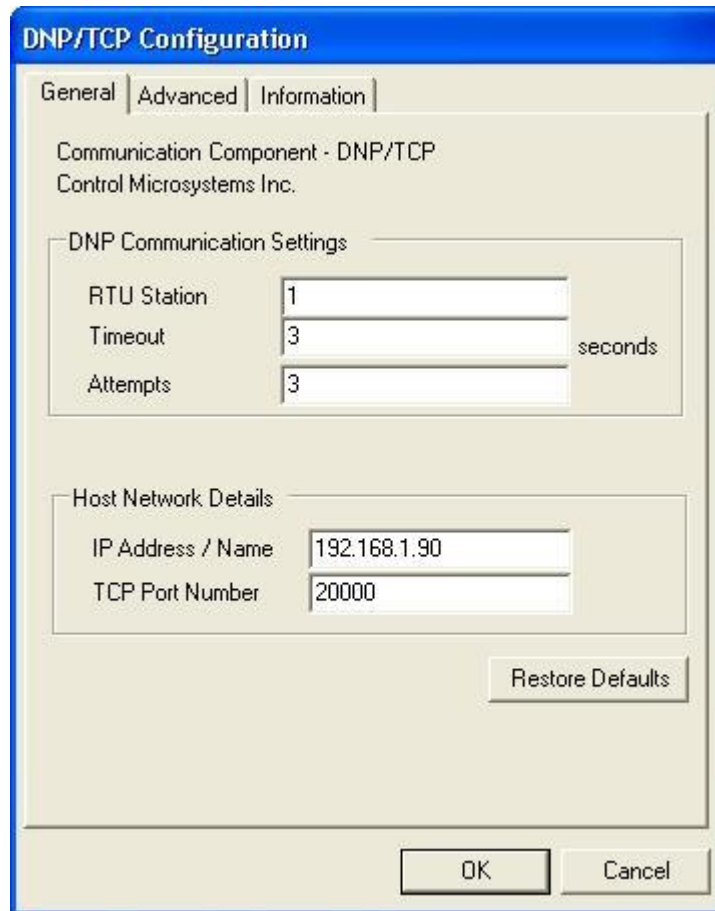
4.5.3.3 DNP/TCP

The DNP/TCP protocol driver is used to communicate over an Ethernet DNP network to SCADAPack controllers configured for DNP/TCP communication.

- To configure a DNP/TCP protocol connection, highlight **DNP/TCP** in the Communication Protocols window and click the **Configure** button. The DNP/TCP Configuration window is displayed.
- To select a configured DNP/TCP protocol connection, highlight **DNP/TCP** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.3.1 General Page

When DNP/TCP protocol is selected for configuration the DNP/TCP Configuration dialog is opened with the General tab selected as shown below.



The dialog box is titled "DNP/TCP Configuration" and has three tabs: "General", "Advanced", and "Information". The "General" tab is selected. Below the tabs, it says "Communication Component - DNP/TCP" and "Control Microsystems Inc.". There are two main sections: "DNP Communication Settings" and "Host Network Details".

DNP Communication Settings:

- RTU Station: 1
- Timeout: 3 seconds
- Attempts: 3

Host Network Details:

- IP Address / Name: 192.168.1.90
- TCP Port Number: 20000

At the bottom right of the dialog box is a "Restore Defaults" button. At the very bottom are "OK" and "Cancel" buttons.

Figure 77: DNP in TCP Configuration (General) Dialog Box

The **DNP Communication Settings** grouping contains DNP specific communication settings including the DNP Station address, the timeout interval as well as the number of attempts.

The **RTU Station** parameter specifies the DNP station number of the target device. The valid range is 0 to 65519. The default is station 1.

The **Timeout** parameter specifies the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts) or ultimately failing. Valid values are 1 to 255. The default value is 3 seconds.

The **Attempts** parameter specifies the number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid values are 1 to 20. The default value is 3 attempts.

The **Host Network Details** grouping contains information about the IP network including the target's IP address or name, and the TCP port number on which it is listening. More details on these below.

4.5.3.3.1.1 IP Address / Name

The IP Address / Name parameter specifies the Ethernet IP address of the target RTU, or a DNS name that can be resolved to an IP address. The default value is blank. The following IP addresses are not supported and will be rejected:

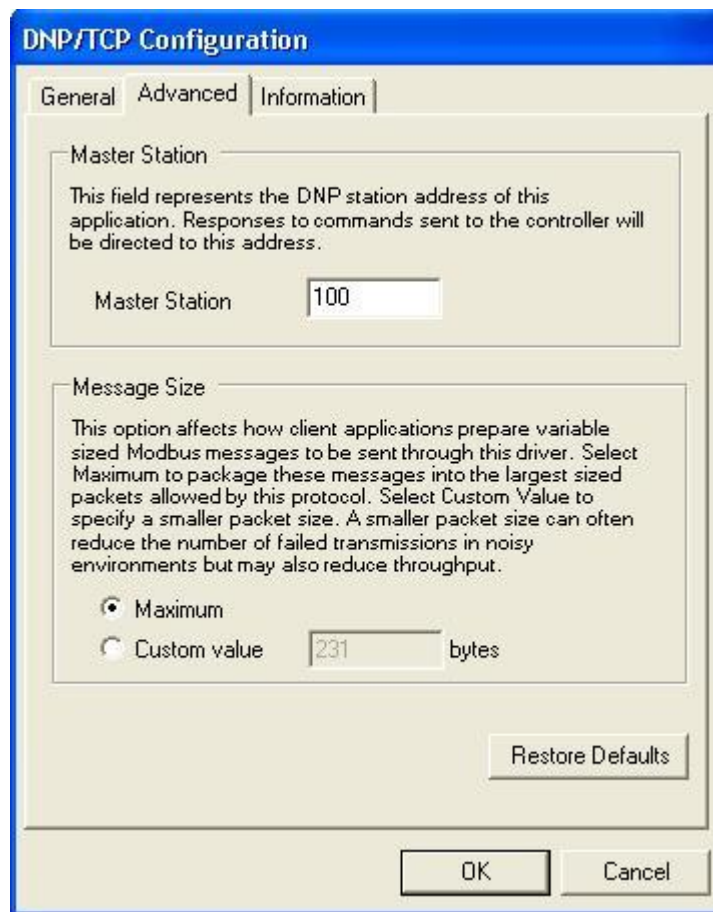
- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **TCP Port No.** field specifies the TCP port of the remote device. Valid values are 0 to 65535. The default value is 20000.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

4.5.3.3.2 Advanced Page

Advanced parameters are used to set the Master Station address and control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.



The image shows a dialog box titled "DNP/TCP Configuration" with three tabs: "General", "Advanced", and "Information". The "Advanced" tab is selected. Inside the dialog, there are two main sections. The first section is titled "Master Station" and contains a text box with the value "100". The second section is titled "Message Size" and contains a text box with the value "231" and the unit "bytes". There are two radio buttons: "Maximum" (selected) and "Custom value". A "Restore Defaults" button is located at the bottom right of the dialog. At the very bottom, there are "OK" and "Cancel" buttons.

Figure 78: DNP n TCP Configuration (Advanced) Dialog Box

The **Master Station** parameter specifies the DNP station address of the RealFLO application. When RealFLO sends out commands, responses from the target controller will be directed to this address. The valid range is 0 to 65519, except that this value cannot be the same as the target RTU Station number. The default value is 100.

The **Maximum** selection indicates that you want the host application to package messages using the maximum size allowable by the protocol.

The **Custom value** selection specifies a custom value for message size. This value indicates to the host application to package messages to be no larger than what is specified if possible. The valid range for the Custom value field is from 2 to 231. *Maximum* is selected by default.

- Click **Restore Defaults** to restore default values to all fields on this page

4.5.3.3.3 Information Page

The Information page displays detailed driver information. When the Information tab is clicked the Information dialog is opened as shown below.



Figure 79: DNP in TCP Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

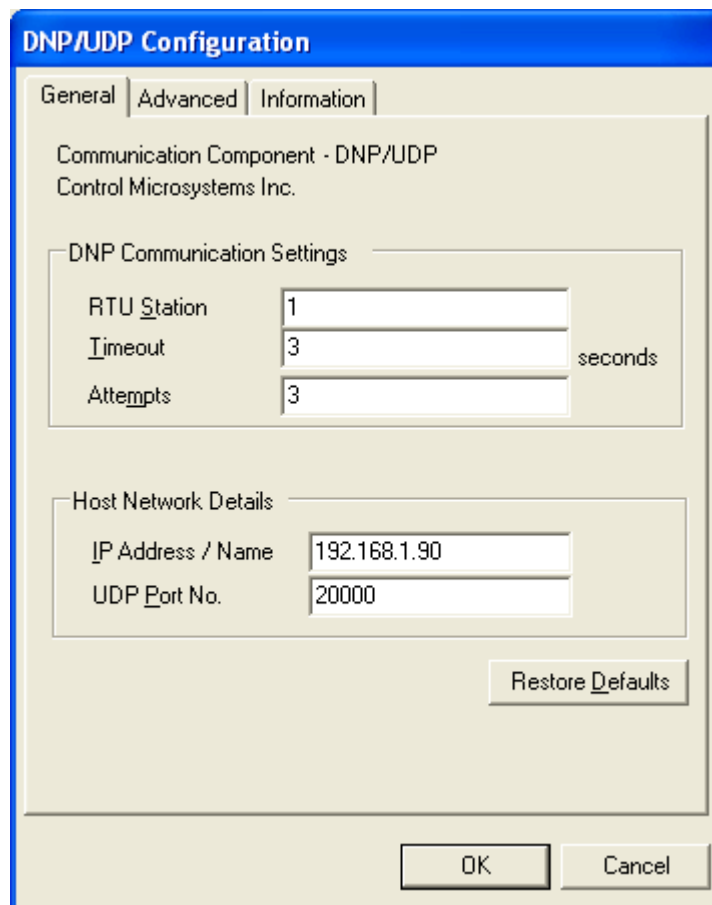
4.5.3.4 DNP/UDP

The DNP/UDP protocol driver is used to communicate over an Ethernet DNP network to SCADAPack controllers configured for DNP/UDP communication.

- To configure a DNP/UDP protocol connection, highlight **DNP/UDP** in the Communication Protocols window and click the **Configure** button. The DNP/UDP Configuration window is displayed.
- To select a configured DNP/UDP protocol connection, highlight **DNP/UDP** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.4.1 General Page

When DNP/UDP protocol is selected for configuration the DNP/UDP Configuration dialog is opened with the General tab selected as shown below.



The image shows a Windows-style dialog box titled "DNP/UDP Configuration". It has three tabs: "General", "Advanced", and "Information", with "General" currently selected. The dialog is for the "Communication Component - DNP/UDP" by "Control Microsystems Inc.". It contains two main sections: "DNP Communication Settings" and "Host Network Details". In the "DNP Communication Settings" section, there are three input fields: "RTU Station" with the value "1", "Timeout" with the value "3", and "Attempts" with the value "3". A label "seconds" is positioned to the right of the "Timeout" field. In the "Host Network Details" section, there are two input fields: "IP Address / Name" with the value "192.168.1.90" and "UDP Port No." with the value "20000". At the bottom right of the dialog is a "Restore Defaults" button. At the very bottom are "OK" and "Cancel" buttons.

DNP Communication Settings	
RTU Station	1
Timeout	3 seconds
Attempts	3

Host Network Details	
IP Address / Name	192.168.1.90
UDP Port No.	20000

Figure 80: DNP in UDP Configuration (General) Dialog Box

The **DNP Communication Settings** grouping contains DNP specific communication settings including the DNP Station address, the timeout interval as well as the number of attempts.

The **RTU Station** parameter specifies the DNP station number of the target device. The valid range is 0 to 65519. The default is station 1.

The **Timeout** parameter specifies the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts) or ultimately failing. Valid values are 1 to 255. The default value is 3 seconds.

The **Attempts** parameter specifies the number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid values are 1 to 20. The default value is 3 attempts.

The **Host Network Details** grouping contains information about the IP network including the target's IP address or name, and the UDP port number on which it is listening. More details on these below.

4.5.3.4.1.1 IP Address / Name

The IP Address / Name parameter specifies the Ethernet IP address of the target RTU, or a DNS name that can be resolved to an IP address. The default value is blank. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **UDP Port No.** field specifies the UDP port of the remote device. Valid values are 0 to 65534. The default value is 20000.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

4.5.3.4.2 Advanced Page

Advanced parameters are used to set the Master Station address and control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

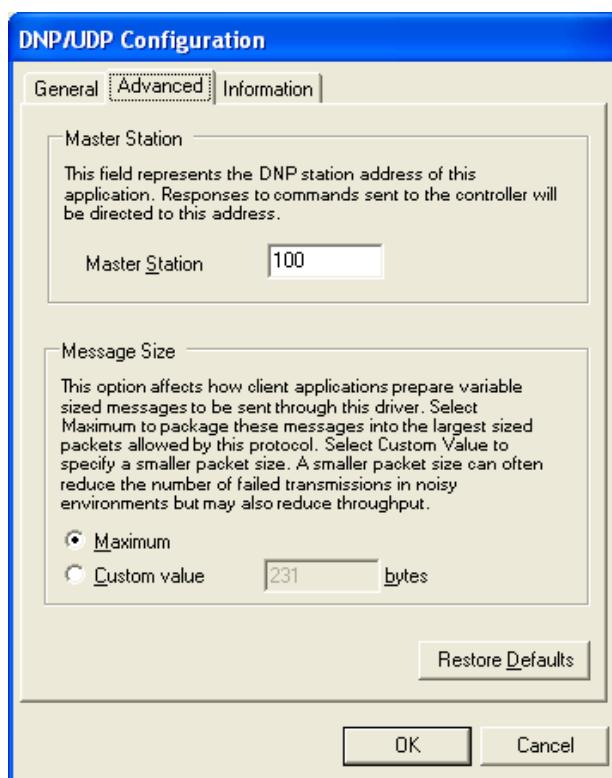


Figure 81: DNP in UDP Configuration (Advanced) Dialog Box

The **Master Station** parameter specifies the DNP station address of the RealFLO application. When RealFLO sends out commands, responses from the target controller will be directed to this address. The valid range is 0 to 65519, except that this value cannot be the same as the target RTU Station number. The default value is 100.

The **Maximum** selection indicates that you want the host application to package messages using the maximum size allowable by the protocol.

The **Custom value** selection specifies a custom value for message size. This value indicates to the host application to package messages to be no larger than what is specified if possible. The valid range for the Custom value field is from 2 to 231. *Maximum* is selected by default.

- Click **Restore Defaults** to restore default values to all fields on this page

4.5.3.4.3 Information Page

The Information page displays detailed driver information. When the Information tab is clicked the Information dialog is opened as shown below.

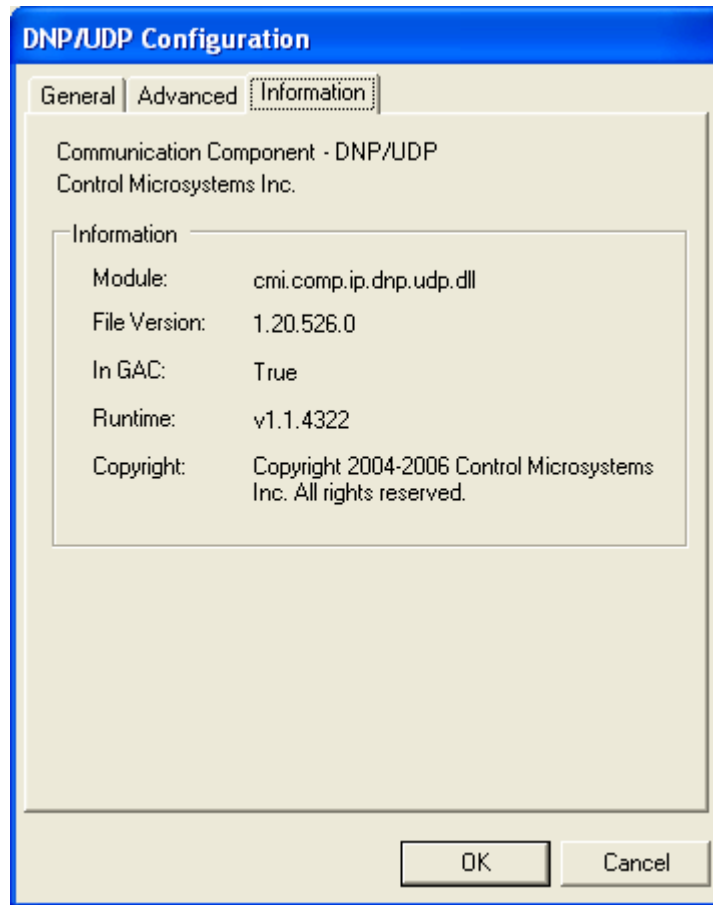


Figure 82: DNP in UDP Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

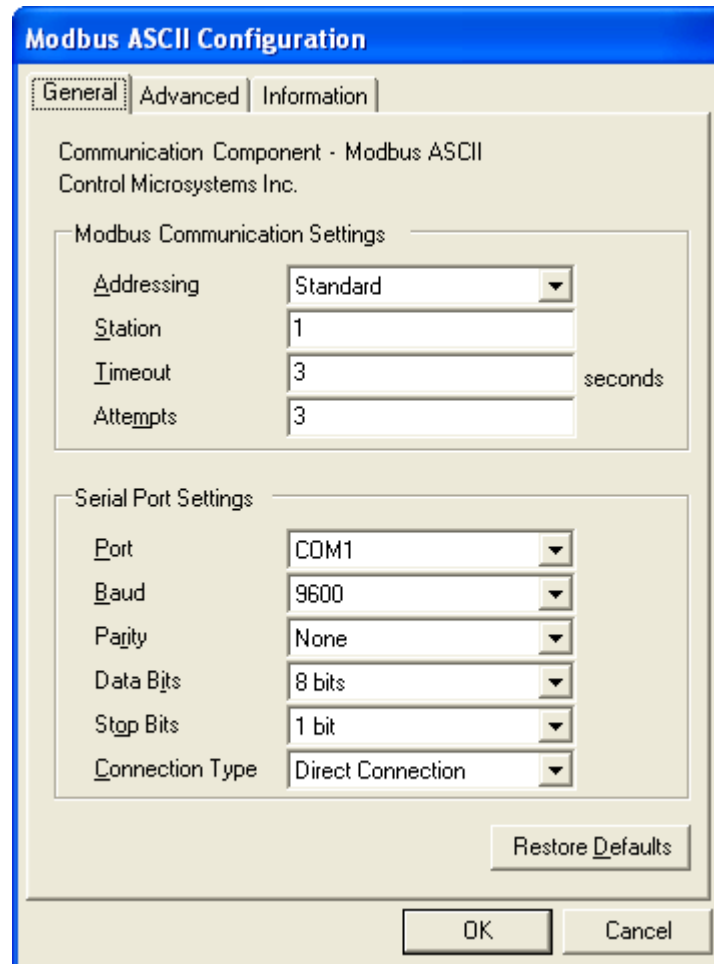
4.5.3.5 Modbus ASCII

The Modbus ASCII protocol driver is used to communicate over a serial network, using Modbus ASCII framing, to SCADAPack controllers configured for Modbus ASCII protocol.

- To configure a Modbus ASCII protocol connection, highlight **Modbus ASCII** in the Communication Protocols window and click the **Configure** button. The Modbus ASCII Configuration window is displayed.
- To select a configured Modbus ASCII protocol connection, highlight **Modbus ASCII** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.5.1 General Parameters

When Modbus ASCII is selected for configuration the Modbus ASCII Configuration dialog is opened with the General tab selected as shown below.



The image shows a Windows-style dialog box titled "Modbus ASCII Configuration". It has three tabs: "General" (selected), "Advanced", and "Information". Below the tabs, it says "Communication Component - Modbus ASCII" and "Control Microsystems Inc.". The dialog is divided into two main sections: "Modbus Communication Settings" and "Serial Port Settings".

Modbus Communication Settings:

- Addressing:** A dropdown menu set to "Standard".
- Station:** A text box containing the number "1".
- Timeout:** A text box containing the number "3", with the unit "seconds" to its right.
- Attempts:** A text box containing the number "3".

Serial Port Settings:

- Port:** A dropdown menu set to "COM1".
- Baud:** A dropdown menu set to "9600".
- Parity:** A dropdown menu set to "None".
- Data Bits:** A dropdown menu set to "8 bits".
- Stop Bits:** A dropdown menu set to "1 bit".
- Connection Type:** A dropdown menu set to "Direct Connection".

At the bottom right of the dialog is a "Restore Defaults" button. At the very bottom are "OK" and "Cancel" buttons.

Figure 83: Modbus ASCII Configuration (General) Dialog Box

The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts), or ultimately failing. Valid entries are 1 to 255. The default is 3.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid entries are 1 to 20. The default is 3.

This **Serial Port Settings** grouping contains details directly related to the PC's communication port including the port number, the baud rate, parity and stop bit settings.

The **Port** parameter specifies the PC serial port to use. The DNP driver determines what serial ports are available on the PC and presents these in the drop-down menu list. The available serial ports list will include any USB to serial converters used on the PC. The default value is the first existing port found by the driver.

The **Baud** parameter specifies the baud rate to use for communication. The menu list displays selections for 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, and 57600. The default value is 9600.

The **Parity** parameter specifies the type of parity to use for communication. The menu list displays selections for none, odd and even parity. The default value is None.

The **Data Bits** parameter specifies the number of data bits contained in the character frame. Valid values are for this field is 7 and 8 bits. The default value is 8 bits.

The **Stop Bits** parameter specifies the number of stop bits to use for communication. The menu list displays selections for 1 and 2 stop bits. The default value is 1 bit.

The **Connection Type** parameter specifies the serial connection type. The Modbus ASCII driver supports direct serial connection with no flow control, Request-to-send (RTS) and clear-to-send (CTS) flow control and PSTN dial-up connections. The menu list displays selections for Direct Connection, RTS/CTS Flow Control and Dial Up Connection. The default selection is Direct Connection.

- Select **Direct Connection** for RS-232 for RS-485 connections that do not require the hardware control lines on the serial ports.
- Select **RTS/CTS Flow Control** to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking. Selecting RTS/CTS Flow Control adds a new tab, Flow Control, to the Modbus ASCII Configuration dialog. Refer to the Flow Control Parameters section below for configuration details.
- Select **Dial Up Connection** to communication over dial up modems. Selecting Dial Up Connection adds a new tab, Dial Up, to the Modbus ASCII Configuration dialog. Refer to the Dial Up Parameters section below for configuration details.
- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.5.2 Modbus ASCII Configuration (Flow Control)

Flow Control parameters are used to configure how RTS and CTS control is used. When RTS/CTS Flow Control is selected for Connection Type the Flow Control tab is added to the Modbus ASCII Configuration dialog. When the Flow Control tab heading is clicked the Flow Control dialog is opened as shown below.

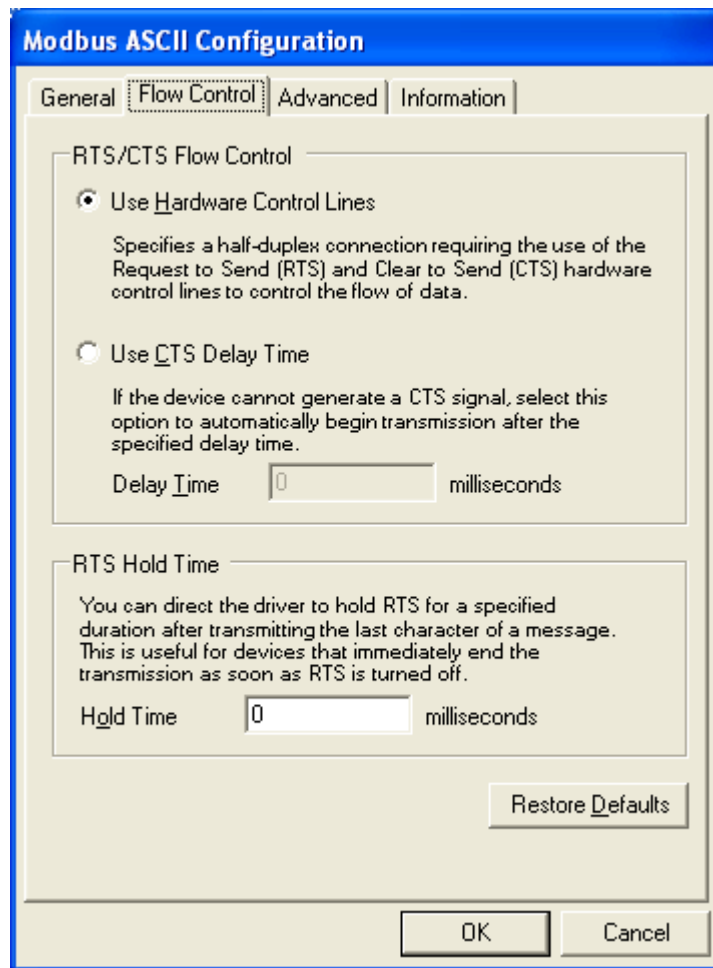


Figure 84: Modbus ASCII Configuration (Flow Control)

The **RTS/CTS Flow Control** grouping contains two mutually exclusive options, **Use Hardware Control Lines** and **Use CTS Delay Time**. These options enable the driver to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking.

The **Use Hardware Control Lines** option specifies a half-duplex connection requiring the use of the Request to Send (RTS) and Clear to Send (CTS) hardware control lines to control the flow of data. This selection is used with radios and dedicated telephone line modems. The driver turns on the RTS signal when it wants to transmit data. The modem or other device then turns on CTS when it is ready to transmit. The driver transmits the data, and then turns off the RTS signal. This selection is mutually exclusive of the **Use CTS Delay Time** selection described below. This is the default selection.

The **Use CTS Delay Time** option is selected if the device cannot generate a CTS signal. The driver will assert RTS then wait the specified Delay Time, in milliseconds, before proceeding. This option is mutually exclusive with the **Use Hardware Control Lines** selection described above.

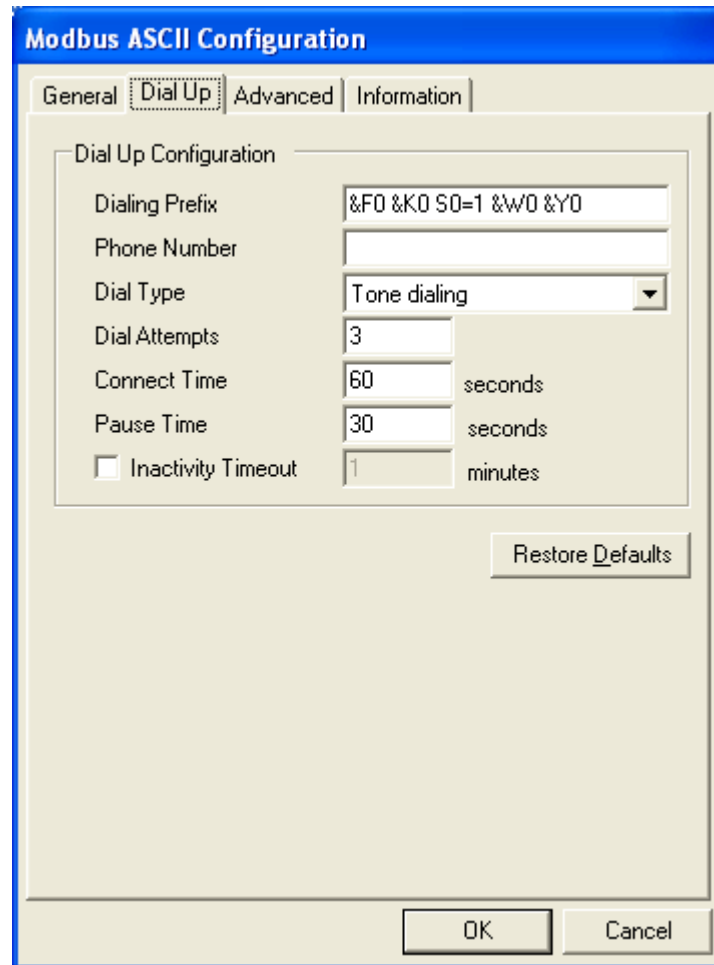
The **Delay Time** parameter sets the time in milliseconds that the driver will wait after asserting RTS before proceeding. The value of this field must be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

The **Hold Time** parameter specifies the time, in milliseconds, that the driver will hold RTS after the last character is transmitted. This is useful for devices that immediately end transmission when RTS is turned off. The value of this field must be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.5.3 Modbus ASCII Configuration (Dial Up)

Dial Up parameters are used to configure a dial up connection. When Dial Up is selected for Connection Type the Dial Up tab is added to the Modbus ASCII Configuration dialog. When the Dial Up tab heading is clicked the Dial Up dialog is opened as shown below.



The image shows a Windows-style dialog box titled "Modbus ASCII Configuration". It has four tabs: "General", "Dial Up", "Advanced", and "Information". The "Dial Up" tab is currently selected. Inside the dialog, there is a section titled "Dial Up Configuration" containing several input fields and a checkbox. The "Dialing Prefix" field contains the text "&F0 &K0 S0=1 &W0 &Y0". The "Phone Number" field is empty. The "Dial Type" dropdown menu is set to "Tone dialing". The "Dial Attempts" field contains the number "3". The "Connect Time" field contains "60" with the unit "seconds" to its right. The "Pause Time" field contains "30" with the unit "seconds" to its right. There is a checkbox labeled "Inactivity Timeout" which is currently unchecked, followed by a field containing "1" with the unit "minutes" to its right. At the bottom right of the "Dial Up Configuration" section is a button labeled "Restore Defaults". At the very bottom of the dialog box are two buttons: "OK" and "Cancel".

Figure 85: Modbus ASCII Configuration (Dial Up)

The **Dialing Prefix** parameter specifies the commands sent to the modem before dialing. A maximum of 32 characters can be entered. All characters are valid. The default value is "&F0 &K0 S0=1 &W0 &Y0".

The **Phone Number** parameter specifies the telephone number of the remote controller. A maximum of 32 characters can be entered. All characters are valid. This field's default value is blank.

The **Dial Type** parameter specifies the dialing type. Valid values are Pulse and Tone. The default value is Tone.

The **Dial Attempts** parameter specifies how many dialing attempts will be made. Valid values are 1 to 10. The default value is 1.

The **Connect Time** parameter specifies the amount of time in seconds the modem will wait for a connection. Valid values are 6 to 300. The default value is 60.

The **Pause Time** parameter specifies the time in seconds between dialing attempts. Valid values are 6 to 600. The default value is 30.

Check the **Inactivity Timeout** check box to automatically terminate the dialup connection after a period of inactivity. The Inactivity Time edit box is enabled only if this option is checked. The default state is checked.

Enter the inactivity period, in minutes, in the **Inactivity Timeout** box. The dialup connection will be terminated automatically after the specified number of minutes of inactivity has lapsed. This option is only active if the Inactivity Timeout box is checked. Valid values are from 1 to 30 minutes. The default value is 1.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the Phone Number field. The content of this field will remain unchanged.

4.5.3.5.4 Advanced Parameters

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

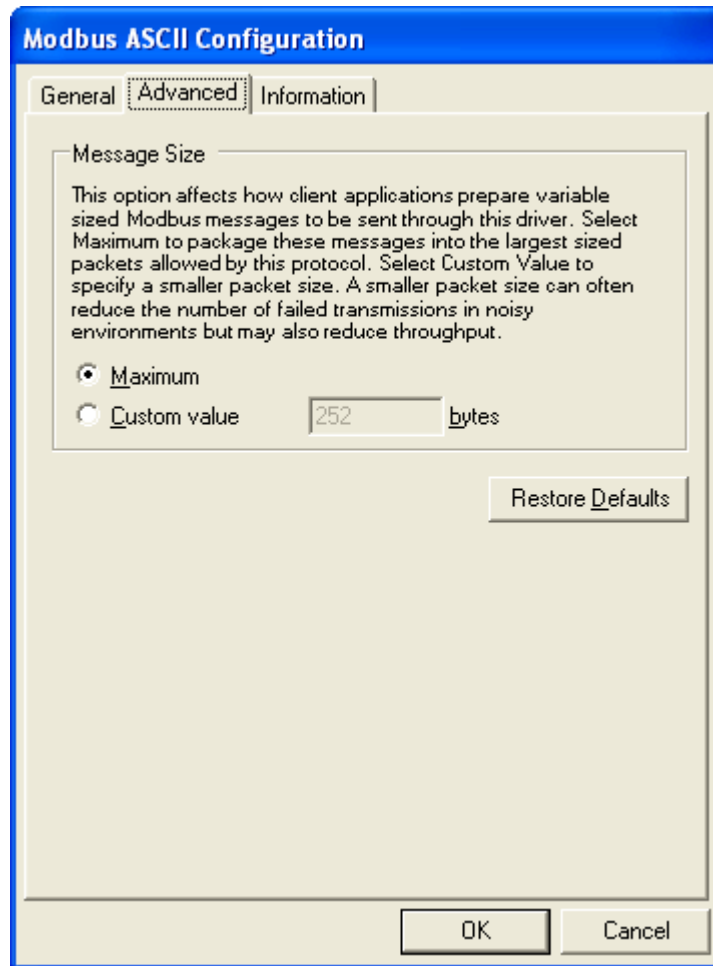


Figure 86: Modbus ASCII Configuration (Advanced) Dialog Box

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 250 when Addressing is set to Extended and Station is 255 or higher. When Addressing is set to Extended and Station is less than 255 valid values are 2 to 252. When Addressing is set to Standard valid values are 2 to 252.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.5.5 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

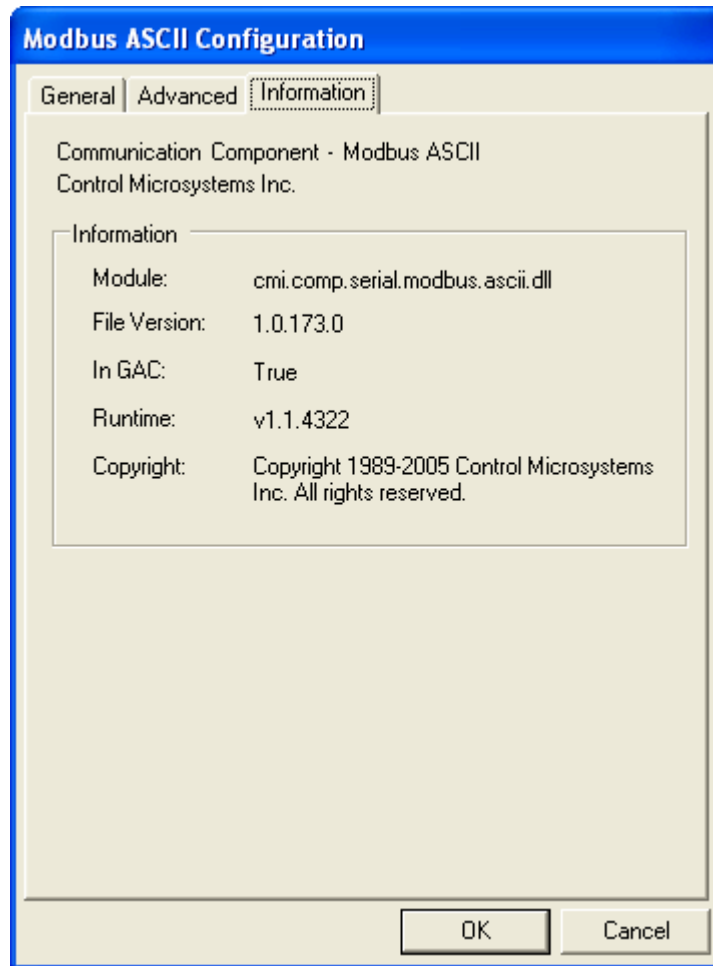


Figure 87: Modbus ASCII Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

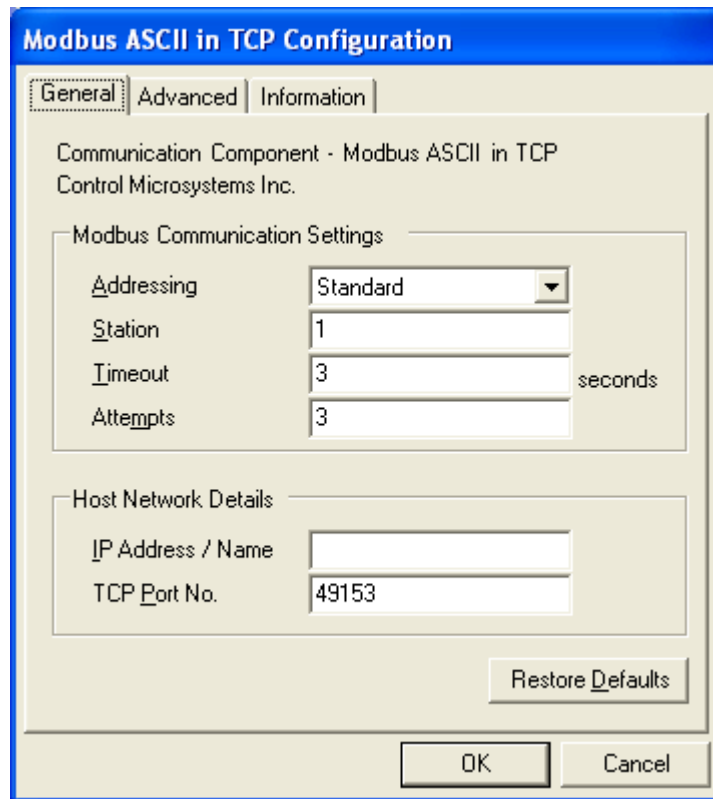
4.5.3.6 Modbus ASCII in TCP

Modbus ASCII in TCP message format is exactly same as that of the Modbus ASCII protocol. The main difference is that Modbus ASCII in TCP protocol communicates with a SCADAPack controller through the Internet and Modbus ASCII through the serial port. The Modbus ASCII in TCP protocol does not include a six-byte header prefix, as with the Modbus\TCP, but does include the Modbus 'CRC-16' or 'LRC' check fields.

- To configure a Modbus ASCII in TCP protocol connection, highlight **Modbus ASCII in TCP** in the Communication Protocols window and click the **Configure** button. The Modbus ASCII in TCP Configuration window is displayed.
- To select a configured Modbus ASCII in TCP protocol connection, highlight **Modbus ASCII in TCP** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.6.1 General Parameters

When Modbus ASCII in TCP is selected for configuration the Modbus ASCII in TCP Configuration dialog is opened with the General tab selected as shown below.



The image shows a dialog box titled "Modbus ASCII in TCP Configuration". It has three tabs: "General", "Advanced", and "Information". The "General" tab is selected. Inside the dialog, it says "Communication Component - Modbus ASCII in TCP" and "Control Microsystems Inc.". There are two main sections: "Modbus Communication Settings" and "Host Network Details".

Modbus Communication Settings:

- Addressing:** A dropdown menu set to "Standard".
- Station:** A text box containing the value "1".
- Timeout:** A text box containing the value "3", with the unit "seconds" to its right.
- Attempts:** A text box containing the value "3".

Host Network Details:

- IP Address / Name:** An empty text box.
- TCP Port No.:** A text box containing the value "49153".

At the bottom right of the dialog is a "Restore Defaults" button. At the very bottom are "OK" and "Cancel" buttons.

Figure 88: Modbus ASCII in TCP Configuration (General) Dialog Box

The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts), or ultimately failing. Valid entries are 1 to 255. The default is 3.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **TCP Port No.** field specifies the TCP port of the remote device. Valid values are 0 to 65535. The default value is 49153.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

4.5.3.6.2 Advanced Parameters

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

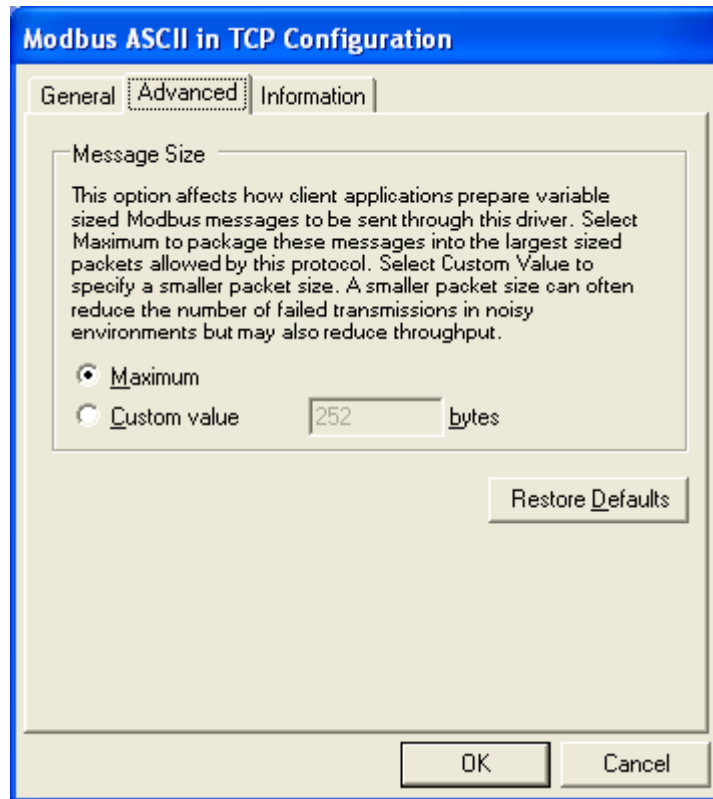


Figure 89: Modbus ASCII in TCP Configuration (Advanced) Dialog Box

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 250 when Addressing is set to Extended and Station is 255 or higher. When Addressing is set to Extended and Station is less than 255 valid values are 2 to 252. When Addressing is set to Standard valid values are 2 to 252.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.6.3 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

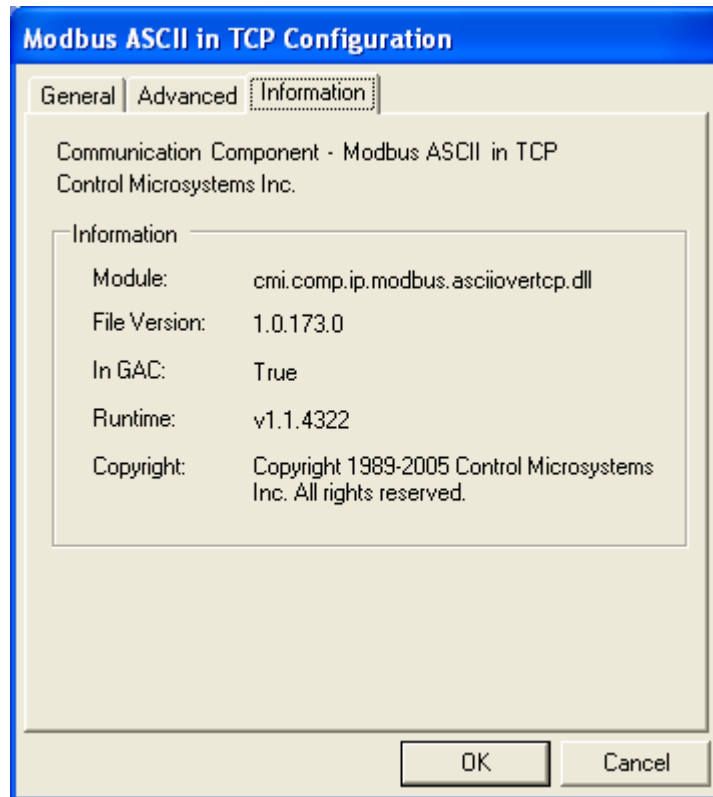


Figure 90: Modbus ASCII in TCP Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

4.5.3.7 Modbus ASCII in UDP

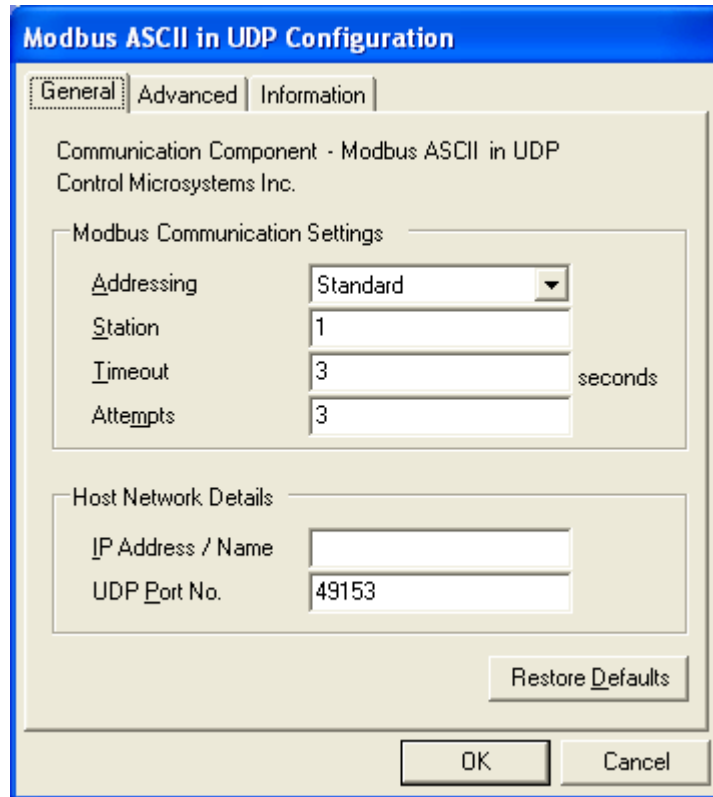
Modbus ASCII in UDP protocol is similar to Modbus ASCII in TCP protocol. It has the same message format as the Modbus ASCII in TCP. The only difference between them is one uses TCP protocol and another uses UDP protocol.

- To configure a Modbus ASCII in TCP protocol connection, highlight **Modbus ASCII in UDP** in the Communication Protocols window and click the **Configure** button. The Modbus ASCII in UDP Configuration window is displayed.
- To select a configured Modbus ASCII in TCP protocol connection, highlight **Modbus ASCII in UDP** in the Communication Protocols window and click the **OK** button.

- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.7.1 General Parameters

When Modbus ASCII in UDP is selected for configuration the Modbus ASCII in UDP Configuration dialog is opened with the General tab selected as shown below.



The image shows a Windows-style dialog box titled "Modbus ASCII in UDP Configuration". It has three tabs: "General" (selected), "Advanced", and "Information". Below the tabs, it says "Communication Component - Modbus ASCII in UDP" and "Control Microsystems Inc.". The dialog is divided into two main sections. The first section, "Modbus Communication Settings", contains four fields: "Addressing" (a dropdown menu set to "Standard"), "Station" (a text box with "1"), "Timeout" (a text box with "3" and "seconds" to its right), and "Attempts" (a text box with "3"). The second section, "Host Network Details", contains two fields: "IP Address / Name" (an empty text box) and "UDP Port No." (a text box with "49153"). At the bottom right of the dialog is a "Restore Defaults" button. At the very bottom are "OK" and "Cancel" buttons.

Figure 91: Modbus ASCII in UDP Configuration (General) Dialog Box

The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts), or ultimately failing. Valid entries are 1 to 255. The default is 3.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **UDP Port No.** field specifies the UDP port of the remote device. Valid values are 0 to 65535. The default value is 49153.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

4.5.3.7.2 Advanced Parameters

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

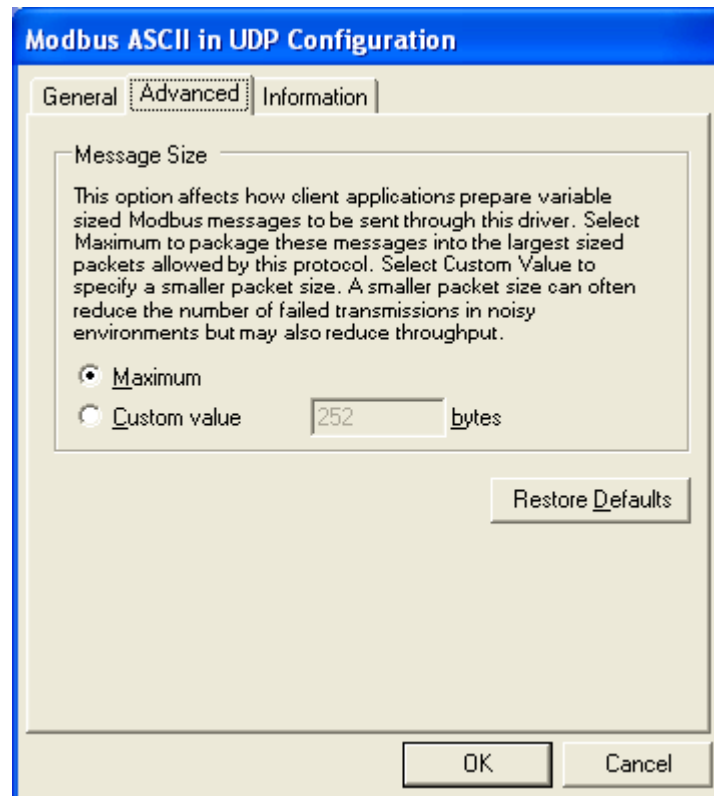


Figure 92: Modbus ASCII in UDP Configuration (Advanced) Dialog Box

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed

transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 250 when Addressing is set to Extended and Station is 255 or higher. When Addressing is set to Extended and Station is less than 255 valid values are 2 to 252. When Addressing is set to Standard valid values are 2 to 252.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.7.3 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

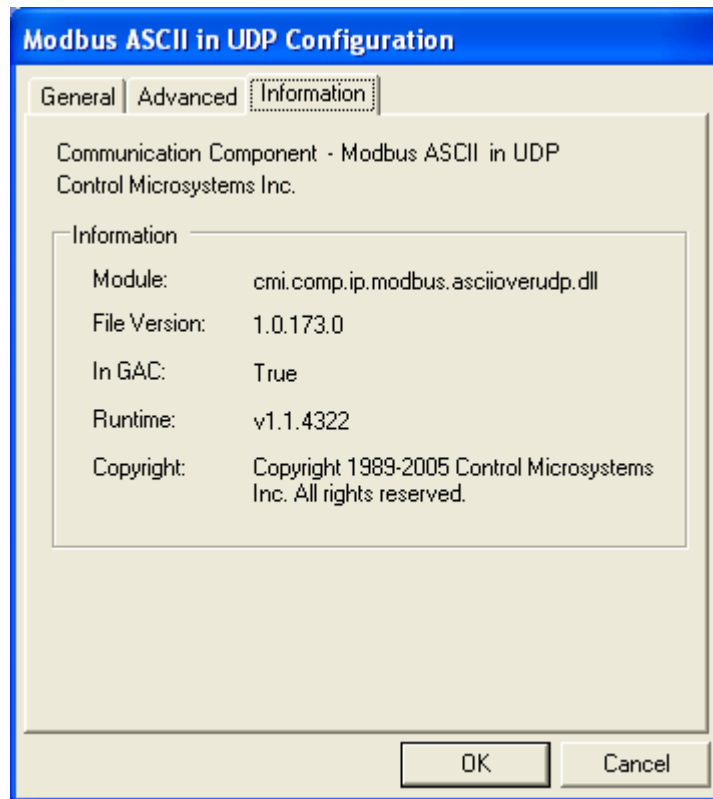


Figure 93: Modbus ASCII in UDP Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

4.5.3.8 Modbus RTU

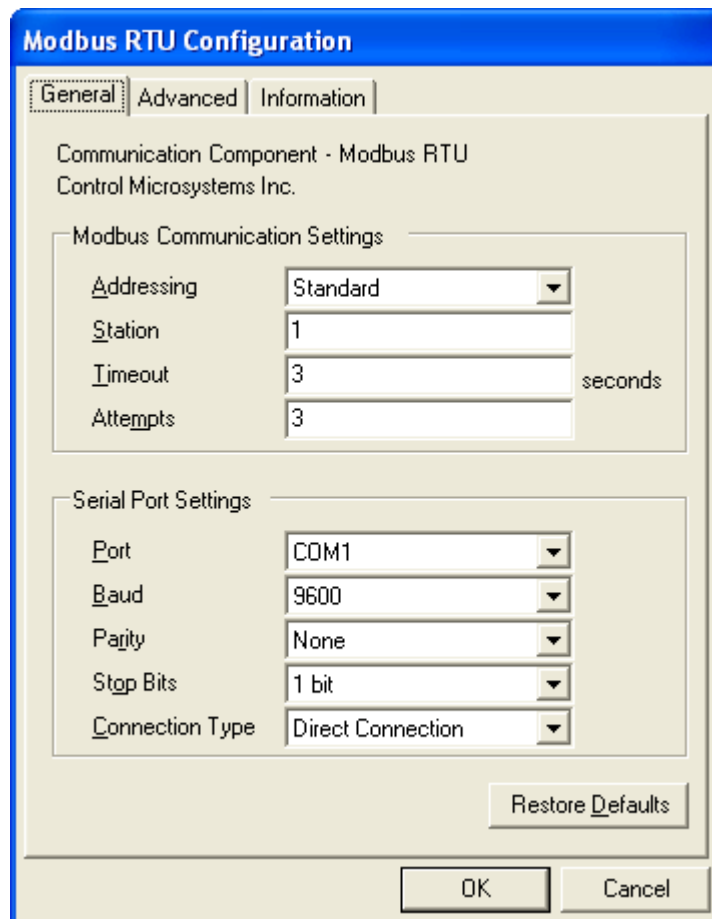
4.5.3.8.1 Introduction

The Modbus RTU protocol driver is used to communicate over a serial network, using Modbus RTU framing, to SCADAPack controllers configured for Modbus RTU protocol.

- To configure a Modbus RTU protocol connection, highlight **Modbus RTU** in the Communication Protocols window and click the **Configure** button. The Modbus RTU Configuration window is displayed.
- To select a configured Modbus RTU protocol connection, highlight **Modbus RTU** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.8.2 General Parameters

When Modbus RTU is selected for configuration the Modbus RTU Configuration dialog is opened with the General tab selected as shown below.



The image shows a Windows-style dialog box titled "Modbus RTU Configuration". It has three tabs: "General" (selected), "Advanced", and "Information". The "General" tab contains the following settings:

- Communication Component - Modbus RTU
Control Microsystems Inc.
- Modbus Communication Settings
 - Addressing: Standard (dropdown)
 - Station: 1 (text box)
 - Timeout: 3 (text box) seconds
 - Attempts: 3 (text box)
- Serial Port Settings
 - Port: COM1 (dropdown)
 - Baud: 9600 (dropdown)
 - Parity: None (dropdown)
 - Stop Bits: 1 bit (dropdown)
 - Connection Type: Direct Connection (dropdown)
- Restore Defaults (button)
- OK (button)
- Cancel (button)

Figure 94: Modbus RTU Configuration (General) Dialog Box

The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts), or ultimately failing. Valid entries are 1 to 255. The default is 3.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid entries are 1 to 20. The default is 3.

This **Serial Port Settings** grouping contains details directly related to the PC's communication port including the port number, the baud rate, parity and stop bit settings.

The **Port** parameter specifies the PC serial port to use. The DNP driver determines what serial ports are available on the PC and presents these in the drop-down menu list. The available serial ports list will include any USB to serial converters used on the PC. The default value is the first existing port found by the driver.

The **Baud** parameter specifies the baud rate to use for communication. The menu list displays selections for 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, and 57600. The default value is 9600.

The **Parity** parameter specifies the type of parity to use for communication. The menu list displays selections for none, odd and even parity. The default value is None.

The **Stop Bits** parameter specifies the number of stop bits to use for communication. The menu list displays selections for 1 and 2 stop bits. The default value is 1 bit.

The **Connection Type** parameter specifies the serial connection type. The Modbus RTU driver supports direct serial connection with no flow control, Request-to-send (RTS) and clear-to-send (CTS) flow control and PSTN dial-up connections. The menu list displays selections for Direct Connection, RTS/CTS Flow Control and Dial Up Connection. The default selection is Direct Connection.

- Select **Direct Connection** for RS-232 for RS-485 connections that do not require the hardware control lines on the serial ports.
- Select **RTS/CTS Flow Control** to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking. Selecting RTS/CTS Flow Control adds a new tab, Flow Control, to the Modbus RTU Configuration dialog. Refer to the Flow Control Parameters section below for configuration details.
- Select **Dial Up Connection** to communication over dial up modems. Selecting Dial Up Connection adds a new tab, Dial Up, to the Modbus RTU Configuration dialog. Refer to the Dial Up Parameters section below for configuration details.
- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.8.3 Modbus RTU Configuration (Flow Control)

Flow Control parameters are used to configure how RTS and CTS control is used. When RTS/CTS Flow Control is selected for Connection Type the Flow Control tab is added to the Modbus RTU Configuration dialog. When the Flow Control tab heading is clicked the Flow Control dialog is opened as shown below.

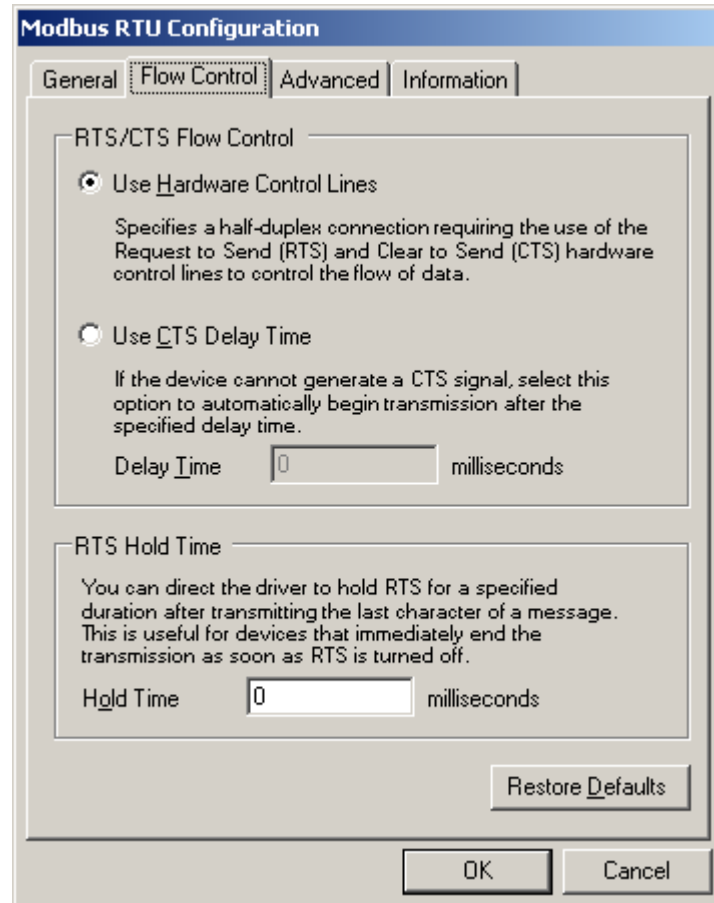


Figure 95: Modbus RTU Configuration (Flow Control)

The **RTS/CTS Flow Control** grouping contains two mutually exclusive options, Use Hardware Control Lines and Use CTS Delay Time. These options enable the driver to communicate over radio or leased-line networks using modems that require RTS/CTS handshaking.

The **Use Hardware Control Lines** option specifies a half-duplex connection requiring the use of the Request to Send (RTS) and Clear to Send (CTS) hardware control lines to control the flow of data. This selection is used with radios and dedicated telephone line modems. The driver turns on the RTS signal when it wants to transmit data. The modem or other device then turns on CTS when it is ready to transmit. The driver transmits the data, and then turns off the RTS signal. This selection is mutually exclusive of the Use CTS Delay Time selection described below. This is the default selection.

The **Use CTS Delay Time** option is selected if the device cannot generate a CTS signal. The driver will assert RTS then wait the specified Delay Time, in milliseconds, before proceeding. This option is mutually exclusive with the Use Hardware Control Lines selection described above.

The **Delay Time** parameter sets the time in milliseconds that the driver will wait after asserting RTS before proceeding. The value of this field must be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

The **Hold Time** parameter specifies the time, in milliseconds, that the driver will hold RTS after the last character is transmitted. This is useful for devices that immediately end transmission when RTS is turned off. The value of this field must be smaller than the Time Out value set in the General parameters dialog. For example, if the Timeout value is set to 3 seconds, the CTS Delay Time can be set to 2999 milliseconds or less. The minimum value for this field is 0 milliseconds. The value is initially set to 0 by default.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.8.4 Modbus RTU Configuration (Dial Up)

Dial Up parameters are used to configure a dial up connection. When Dial Up is selected for Connection Type the Dial Up tab is added to the Modbus RTU Configuration dialog. When the Dial Up tab heading is clicked the Dial Up dialog is opened as shown below.

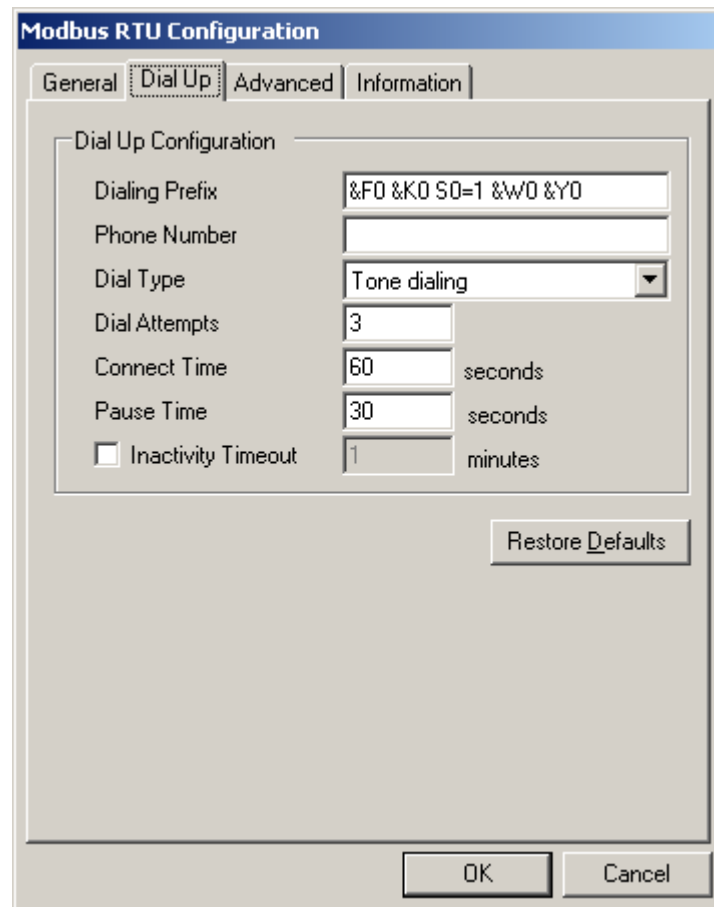
The image shows a Windows-style dialog box titled "Modbus RTU Configuration". It has four tabs: "General", "Dial Up", "Advanced", and "Information". The "Dial Up" tab is currently selected. Inside the dialog, there is a section titled "Dial Up Configuration" which contains several fields: "Dialing Prefix" with the text "&F0 &K0 S0=1 &W0 &Y0", "Phone Number" (empty), "Dial Type" (a dropdown menu showing "Tone dialing"), "Dial Attempts" (a text box with "3"), "Connect Time" (a text box with "60" followed by "seconds"), "Pause Time" (a text box with "30" followed by "seconds"), and "Inactivity Timeout" (a checkbox that is unchecked, followed by a text box with "1" and "minutes"). At the bottom right of the "Dial Up Configuration" section is a button labeled "Restore Defaults". At the very bottom of the dialog are "OK" and "Cancel" buttons.

Figure 96: Modbus RTU Configuration (Dial Up)

The **Dialing Prefix** parameter specifies the commands sent to the modem before dialing. A maximum of 32 characters can be entered. All characters are valid. The default value is "&F0 &K0 S0=1 &W0 &Y0".

The **Phone Number** parameter specifies the telephone number of the remote controller. A maximum of 32 characters can be entered. All characters are valid. This field's default value is blank.

The **Dial Type** parameter specifies the dialing type. Valid values are Pulse and Tone. The default value is Tone.

The **Dial Attempts** parameter specifies how many dialing attempts will be made. Valid values are 1 to 10. The default value is 1.

The **Connect Time** parameter specifies the amount of time in seconds the modem will wait for a connection. Valid values are 6 to 300. The default value is 60.

The **Pause Time** parameter specifies the time in seconds between dialing attempts. Valid values are 6 to 600. The default value is 30.

Check the **Inactivity Timeout** check box to automatically terminate the dialup connection after a period of inactivity. The Inactivity Time edit box is enabled only if this option is checked. The default state is checked.

Enter the inactivity period, in minutes, in the **Inactivity Timeout** box. The dialup connection will be terminated automatically after the specified number of minutes of inactivity has lapsed. This option is only active if the Inactivity Timeout box is checked. Valid values are from 1 to 30 minutes. The default value is 1.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the Phone Number field. The content of this field will remain unchanged.

4.5.3.8.5 Advanced Parameters

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

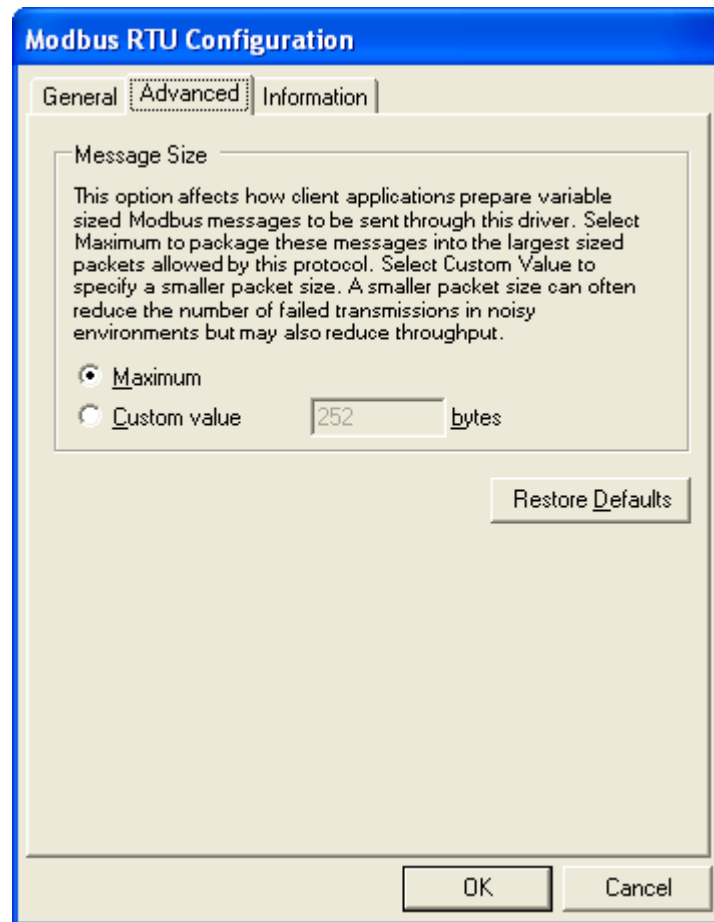


Figure 97: Modbus RTU Configuration (Advanced) Dialog Box

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 250 when Addressing is set to Extended and Station is 255 or higher. When Addressing is set to Extended and Station is less than 255 valid values are 2 to 252. When Addressing is set to Standard valid values are 2 to 252.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.8.6 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

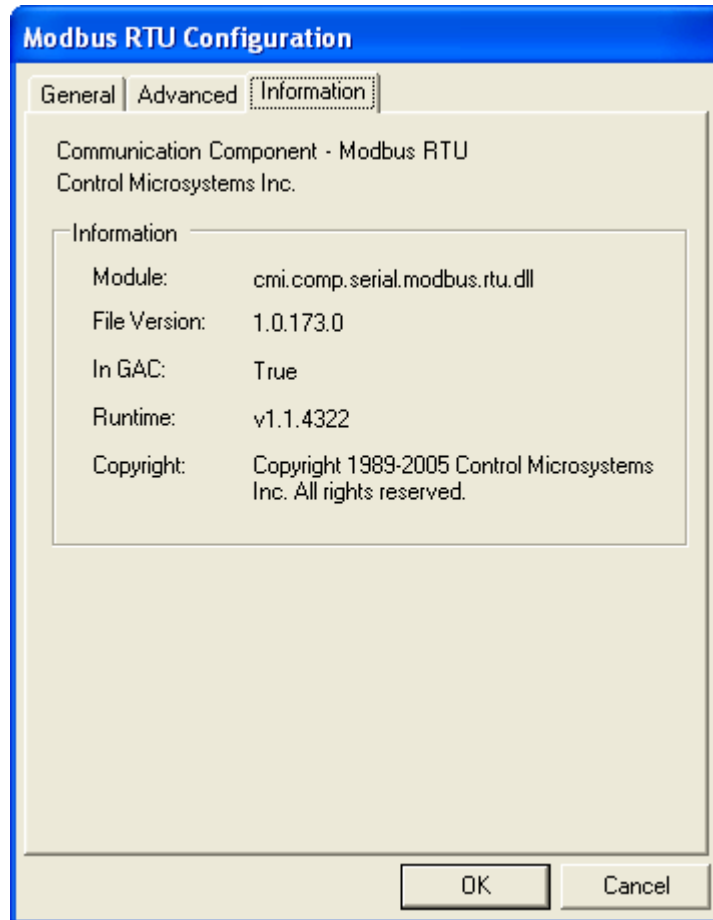


Figure 98: Modbus RTU Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

4.5.3.9 Modbus RTU in TCP

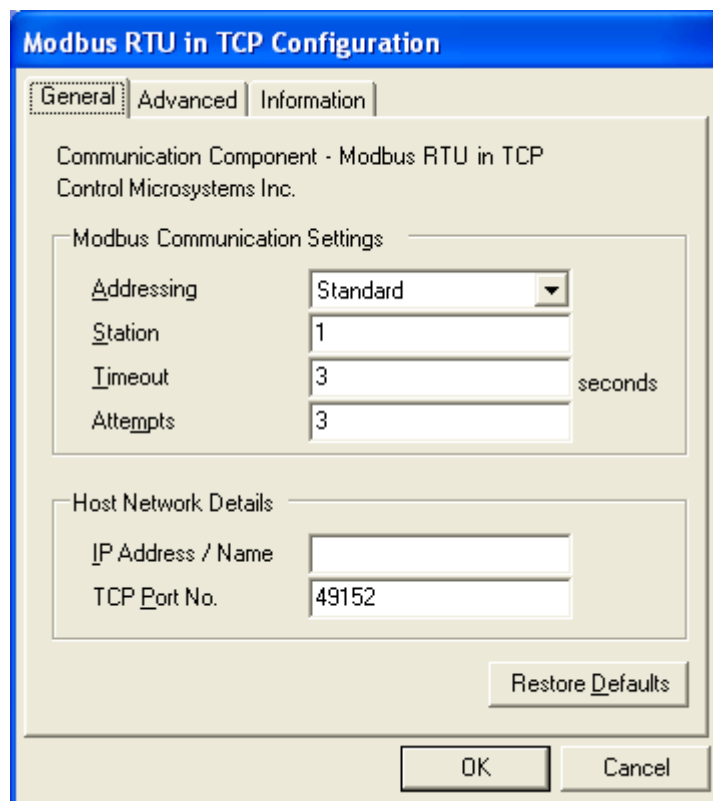
Modbus RTU in TCP message format is exactly same as that of the Modbus RTU protocol. The main difference is that Modbus RTU in TCP protocol communicates with a controller through the Internet and Modbus RTU protocol through the serial port. The Modbus RTU in TCP protocol does not include a six-byte header prefix, as with the Modbus/TCP, but does include the Modbus 'CRC-

16' or 'LRC' check fields. The Modbus RTU in TCP message format supports Modbus RTU message format.

- To configure a Modbus RTU in TCP protocol connection, highlight **Modbus RTU in TCP** in the Communication Protocols window and click the **Configure** button. The Modbus RTU in TCP Configuration window is displayed.
- To select a configured Modbus RTU in TCP protocol connection, highlight **Modbus RTU in TCP** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.9.1 General Parameters

When Modbus RTU in TCP is selected for configuration the Modbus RTU in TCP Configuration dialog is opened with the General tab selected as shown below.



The image shows a Windows-style dialog box titled "Modbus RTU in TCP Configuration". It has three tabs: "General" (selected), "Advanced", and "Information". The "General" tab contains the following elements:

- Text: "Communication Component - Modbus RTU in TCP" and "Control Microsystems Inc."
- Section: "Modbus Communication Settings" (indicated by a minus sign icon)
 - "Addressing": A dropdown menu currently set to "Standard".
 - "Station": A text input field containing the value "1".
 - "Timeout": A text input field containing the value "3", followed by the text "seconds".
 - "Attempts": A text input field containing the value "3".
- Section: "Host Network Details" (indicated by a minus sign icon)
 - "IP Address / Name": A text input field.
 - "TCP Port No.": A text input field containing the value "49152".
- Buttons: "Restore Defaults", "OK", and "Cancel".

Figure 99: Modbus RTU in TCP Configuration (General) Dialog Box

The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts), or ultimately failing. Valid entries are 1 to 255. The default is 3.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **TCP Port No.** field specifies the TCP port of the remote device. Valid values are 0 to 65535. The default value is 49152.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

4.5.3.9.2 Advanced Parameters

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

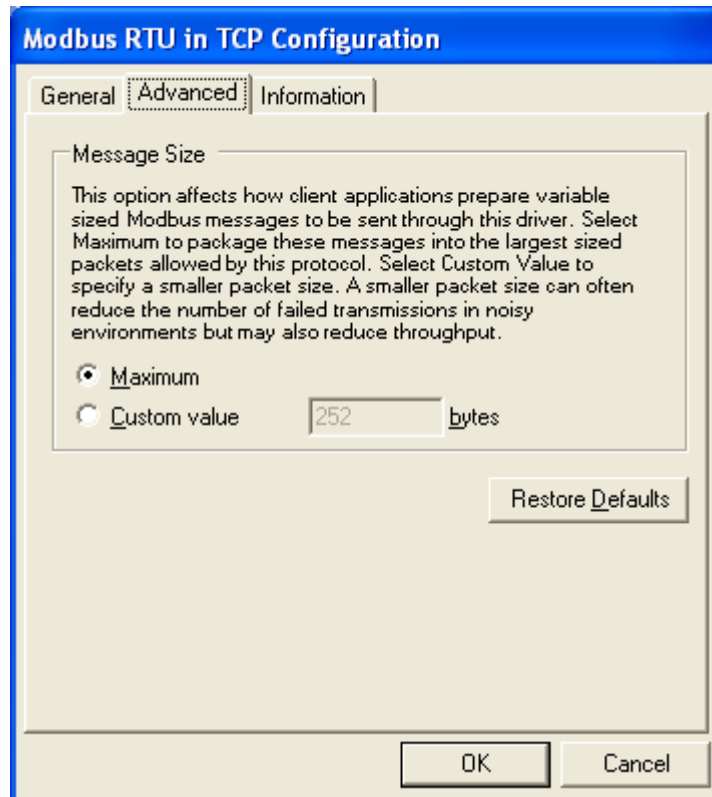


Figure 100: Modbus RTU in TCP Configuration (Advanced) Dialog Box

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 250 when Addressing is set to Extended and Station is 255 or higher. When Addressing is set to Extended and Station is less than 255 valid values are 2 to 252. When Addressing is set to Standard valid values are 2 to 252.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.9.3 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

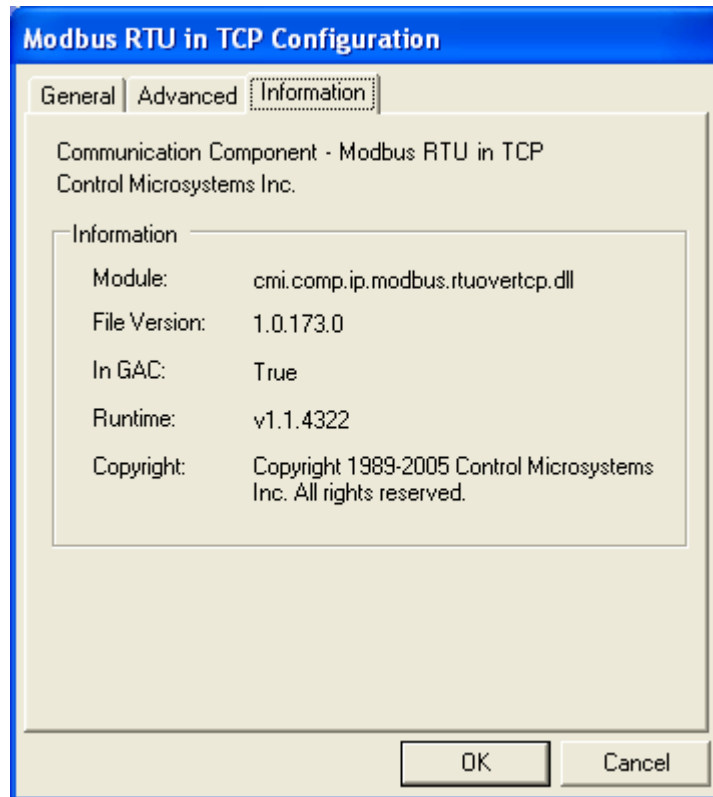


Figure 101: Modbus RTU in TCP Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

4.5.3.10 Modbus RTU in UDP

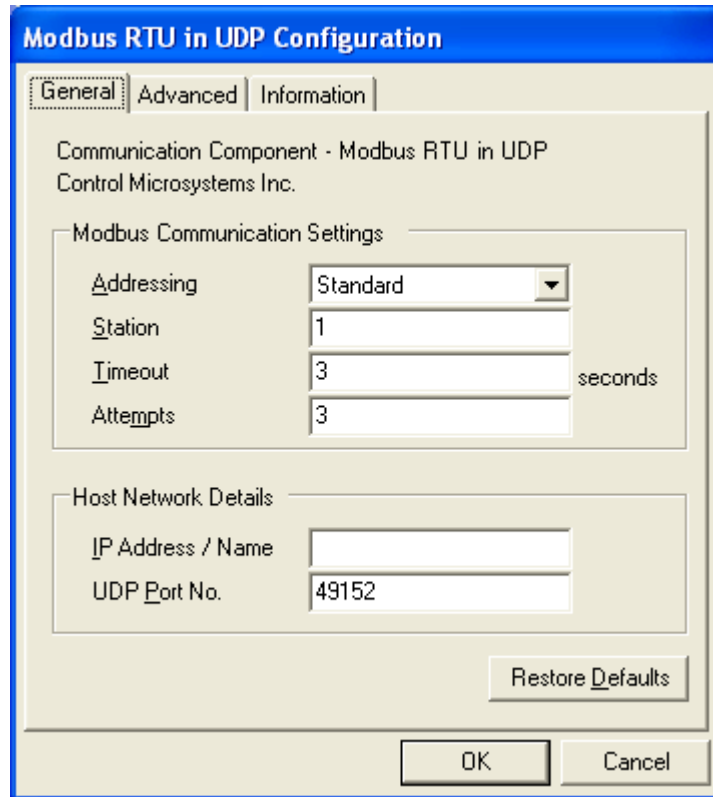
Modbus RTU in UDP protocol is similar to Modbus RTU in TCP protocol. It has the same message format as the RTU in TCP message. The only difference between them is one uses TCP protocol and another uses UDP protocol.

- To configure a Modbus RTU in UDP protocol connection, highlight **Modbus RTU in UDP** in the Communication Protocols window and click the **Configure** button. The Modbus RTU in UDP Configuration window is displayed.
- To select a configured Modbus RTU in UDP protocol connection, highlight **Modbus RTU in UDP** in the Communication Protocols window and click the **OK** button.

- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.10.1 General Parameters

When Modbus RTU in UDP is selected for configuration the Modbus RTU in UDP Configuration dialog is opened with the General tab selected as shown below.



The image shows a Windows-style dialog box titled "Modbus RTU in UDP Configuration". It has three tabs: "General" (selected), "Advanced", and "Information". The dialog is from "Control Microsystems Inc." and is for the "Communication Component - Modbus RTU in UDP".

Under the "Modbus Communication Settings" group, there are four fields:

- Addressing:** A dropdown menu set to "Standard".
- Station:** A text box containing the value "1".
- Timeout:** A text box containing the value "3", with the unit "seconds" to its right.
- Attempts:** A text box containing the value "3".

Under the "Host Network Details" group, there are two fields:

- IP Address / Name:** An empty text box.
- UDP Port No.:** A text box containing the value "49152".

At the bottom right of the dialog is a "Restore Defaults" button. At the very bottom are "OK" and "Cancel" buttons.

Figure 102: Modbus RTU in UDP Configuration (General) Dialog Box

The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts), or ultimately failing. Valid entries are 1 to 255. The default is 3.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **UDP Port No.** field specifies the UDP port of the remote device. Valid values are 0 to 65535. The default value is 49152.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

4.5.3.10.2 Advanced Parameters

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

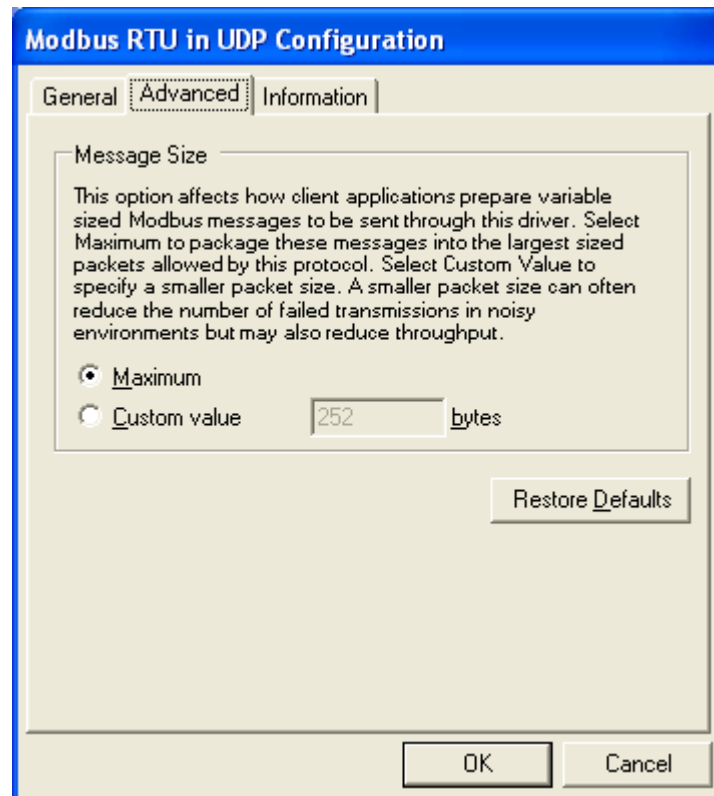


Figure 103: Modbus RTU in UDP Configuration (Advanced) Dialog Box

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed

transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 250 when Addressing is set to Extended and Station is 255 or higher. When Addressing is set to Extended and Station is less than 255 valid values are 2 to 252. When Addressing is set to Standard valid values are 2 to 252.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.10.3 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

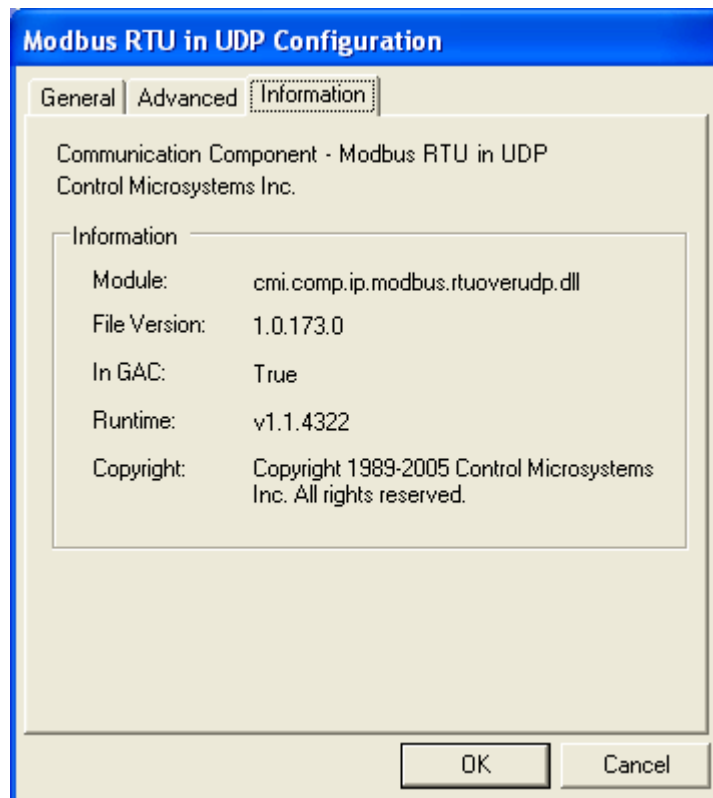


Figure 104: Modbus RTU in UDP Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

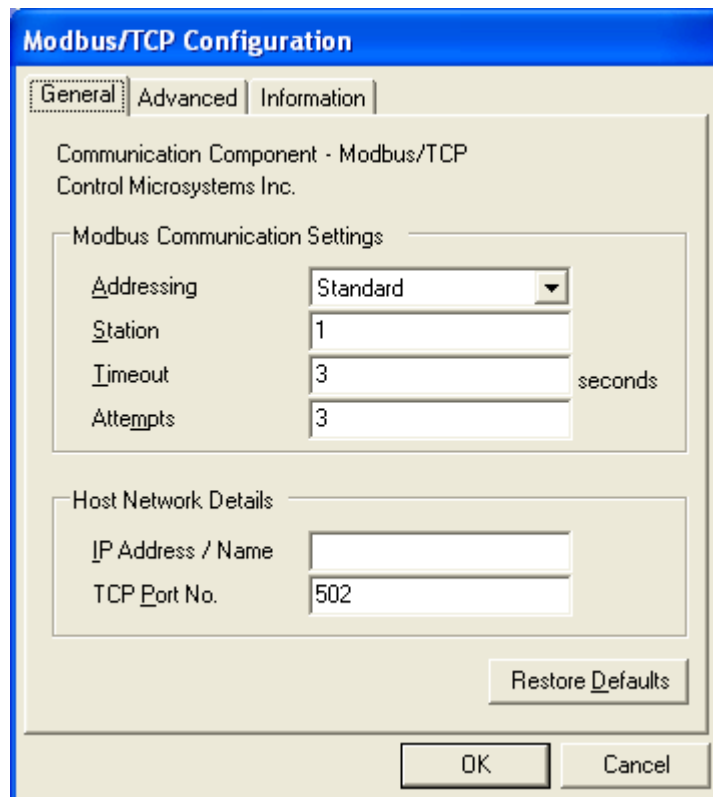
4.5.3.11 Modbus/TCP

Modbus/TCP is an extension of serial Modbus, which defines how Modbus messages are encoded within and transported over TCP/IP-based networks. The Modbus/TCP protocol uses a custom Modbus protocol layer on top of the TCP protocol. Its request and response messages are prefixed by six bytes. These six bytes consist of three fields: transaction ID field, protocol ID field and length field. The encapsulated Modbus message has exactly the same layout and meaning, from the function code to the end of the data portion, as other Modbus messages. The Modbus 'CRC-16' or 'LRC' check fields are not used in Modbus/TCP. The TCP/IP and link layer (e.g. Ethernet) checksum mechanisms instead are used to verify accurate delivery of the packet.

- To configure a Modbus/TCP protocol connection, highlight **Modbus/TCP** in the Communication Protocols window and click the **Configure** button. The Modbus/TCP Configuration window is displayed.
- To select a configured Modbus/TCP protocol connection, highlight **Modbus/TCP** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.11.1 General Parameters

When Modbus/TCP is selected for configuration the Modbus/TCP Configuration dialog is opened with the General tab selected as shown below.



The image shows a screenshot of the 'Modbus/TCP Configuration' dialog box. The 'General' tab is selected, and the 'Communication Component' is set to 'Modbus/TCP' by 'Control Microsystems Inc.'. The 'Modbus Communication Settings' section includes a dropdown for 'Addressing' set to 'Standard', a text field for 'Station' set to '1', a text field for 'Timeout' set to '3' with a unit of 'seconds', and a text field for 'Attempts' set to '3'. The 'Host Network Details' section includes a text field for 'IP Address / Name' and a text field for 'TCP Port No.' set to '502'. There is a 'Restore Defaults' button at the bottom right of the settings area, and 'OK' and 'Cancel' buttons at the very bottom.

Figure 105: Modbus/TCP Configuration (General) Dialog Box

The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts), or ultimately failing. Valid entries are 1 to 255. The default is 3.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **TCP Port No.** field specifies the UDP port of the remote device. Valid values are 0 to 65535. The default value is 502.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

4.5.3.11.2 Advanced Parameters

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

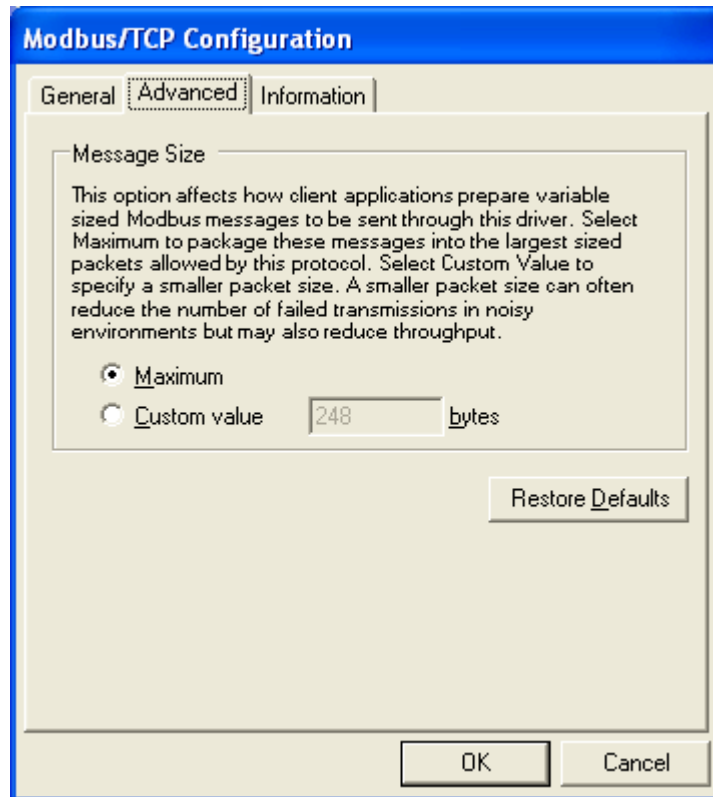


Figure 106: Modbus/TCP Configuration (Advanced) Dialog Box

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 246 when Addressing is set to Extended and Station is 255 or higher. When Addressing is set to Extended and Station is less than 255 valid values are 2 to 248. When Addressing is set to Standard valid values are 2 to 248.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.11.3 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

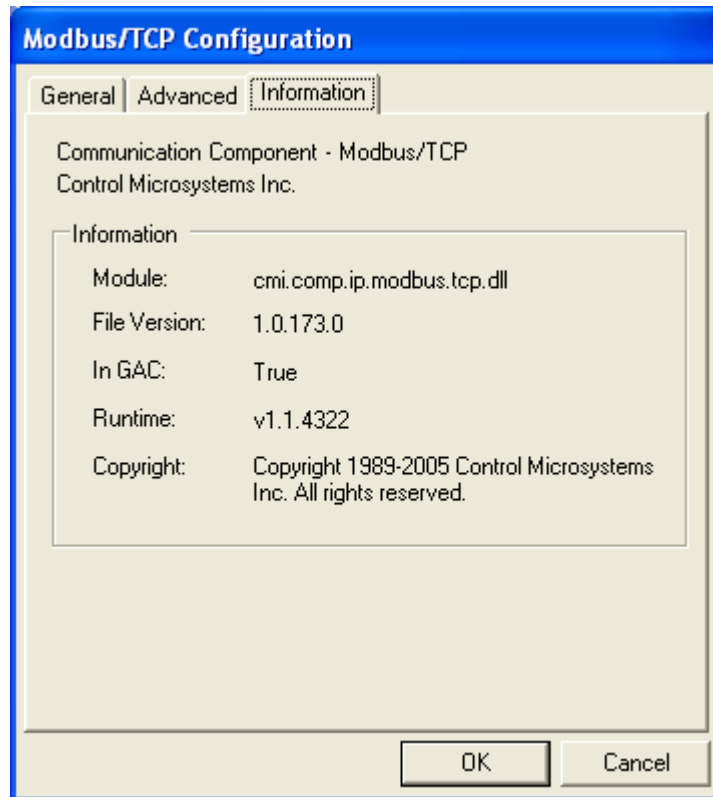


Figure 107: Modbus/TCP Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

4.5.3.12 Modbus/UDP

Modbus/UDP communication mode is similar to Modbus/TCP communication mode. It has the same message format with the Modbus/TCP. The only difference between them is one uses TCP protocol and another uses UDP protocol.

- To configure a Modbus/UDP protocol connection, highlight **Modbus/UDP** in the Communication Protocols window and click the **Configure** button. The Modbus/UDP Configuration window is displayed.
- To select a configured Modbus/UDP protocol connection, highlight **Modbus/UDP** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.12.1 General Parameters

When Modbus/UDP is selected for configuration the Modbus/ UDP Configuration dialog is opened with the General tab selected as shown below.

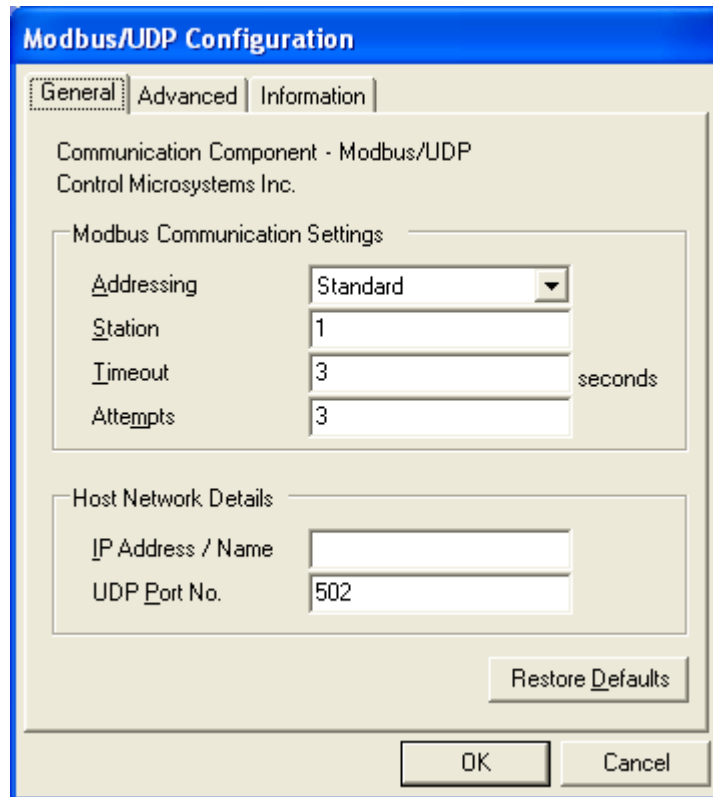
The image shows a Windows-style dialog box titled "Modbus/UDP Configuration". It has three tabs: "General", "Advanced", and "Information", with "General" being the active tab. Below the tabs, it says "Communication Component - Modbus/UDP" and "Control Microsystems Inc.". The dialog is divided into two main sections. The first section, "Modbus Communication Settings", contains four fields: "Addressing" is a dropdown menu set to "Standard"; "Station" is a text box containing "1"; "Timeout" is a text box containing "3" with the unit "seconds" to its right; and "Attempts" is a text box containing "3". The second section, "Host Network Details", contains two fields: "IP Address / Name" is an empty text box, and "UDP Port No." is a text box containing "502". At the bottom right of the dialog is a "Restore Defaults" button. At the very bottom are "OK" and "Cancel" buttons.

Figure 108: Modbus/UDP Configuration (General) Dialog Box

The **Modbus Communication Settings** grouping contains Modbus specific communication settings including the addressing mode, the station address, the timeout interval as well as the number of attempts.

The **Addressing** parameter selects standard or extended Modbus addressing. Standard addressing allows 255 stations and is compatible with standard Modbus devices. Extended addressing allows 65534 stations, with stations 1 to 254 compatible with standard Modbus devices. The default is Standard.

The **Station** parameter sets the target station number. The valid range is 1 to 255 if standard addressing is used, and 1 to 65534 if extended addressing is used. The default is 1.

The **Timeout** parameter sets the length of time, in seconds, to wait for a response from the controller before retrying (see Attempts), or ultimately failing. Valid entries are 1 to 255. The default is 3.

The **Attempts** parameter sets number of times to send a command to the controller before giving up and reporting this failure to the host application. Valid entries are 1 to 20. The default is 3.

The **Host Network Details** grouping contains entries for the host's IP address or name and the TCP port on which it is listening.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255
- 255.0.0.0 through 255.255.255.255.

The **UDP Port No.** field specifies the UDP port of the remote device. Valid values are 0 to 65535. The default value is 502.

- Click **Restore Defaults** to restore default values to all fields on this page, except for the IP Address / Name field. The content of this field will remain unchanged.

4.5.3.12.2 Advanced Parameters

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

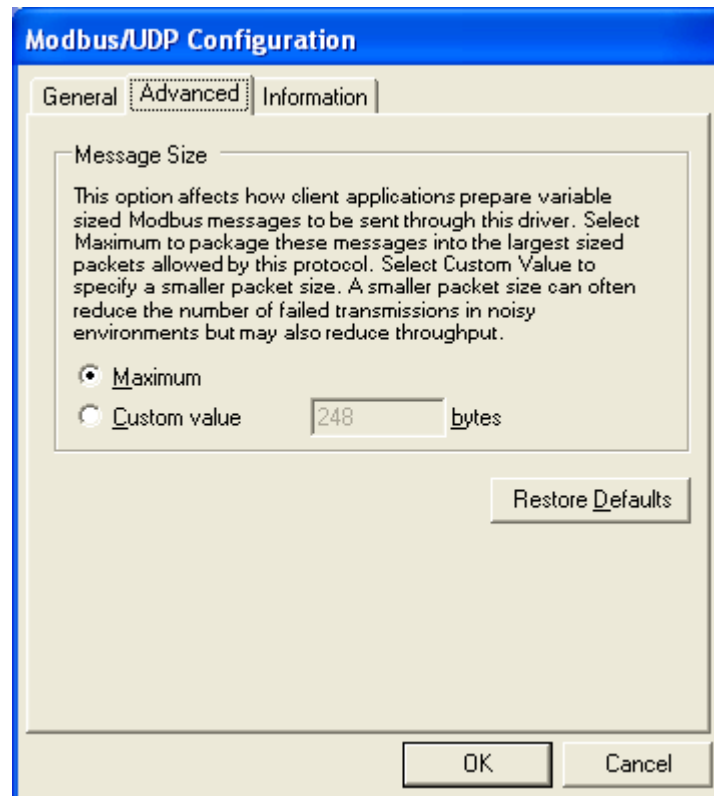


Figure 109: Modbus/UDP Configuration (Advanced) Dialog Box

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed

transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 246 when Addressing is set to Extended and Station is 255 or higher. When Addressing is set to Extended and Station is less than 255 valid values are 2 to 248. When Addressing is set to Standard valid values are 2 to 248.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.12.3 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

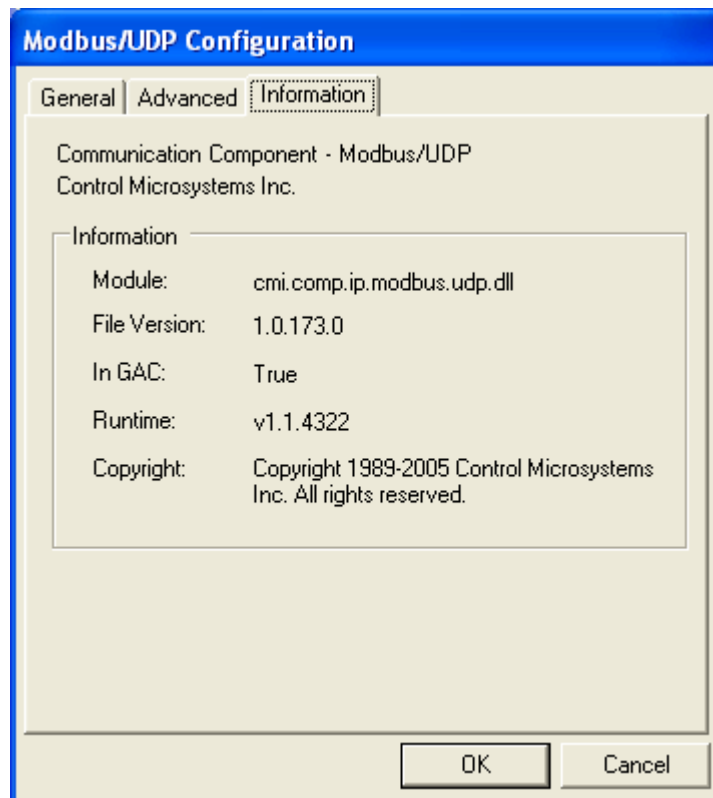


Figure 110: Modbus/UDP Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

4.5.3.13 Modbus/USB

This driver provides the means to communicate with SCADAPack controllers equipped with a Universal Serial Bus (USB) peripheral port using Modbus/USB messaging. The driver does not require configuration making it possible to connect and communicate with a SCADAPack controller almost instantaneously.

- To configure a Modbus/USB protocol connection, highlight **Modbus/USB** in the Communication Protocols window and click the **Configure** button. The Modbus/ USB Configuration window is displayed. The pages in this configuration window are described in the sections below.
- To select a configured Modbus/USB protocol connection, highlight **Modbus/ USB** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

The following sections describe the information presented and user input required to configure the **Modbus/USB** driver.

4.5.3.13.1 General Page

The general page identifies the type of driver and its author. This page also allows a user to specify how the application searches and connects to a USB equipped SCADAPack controller. User input depends on the number of USB equipped controllers connected on the bus.

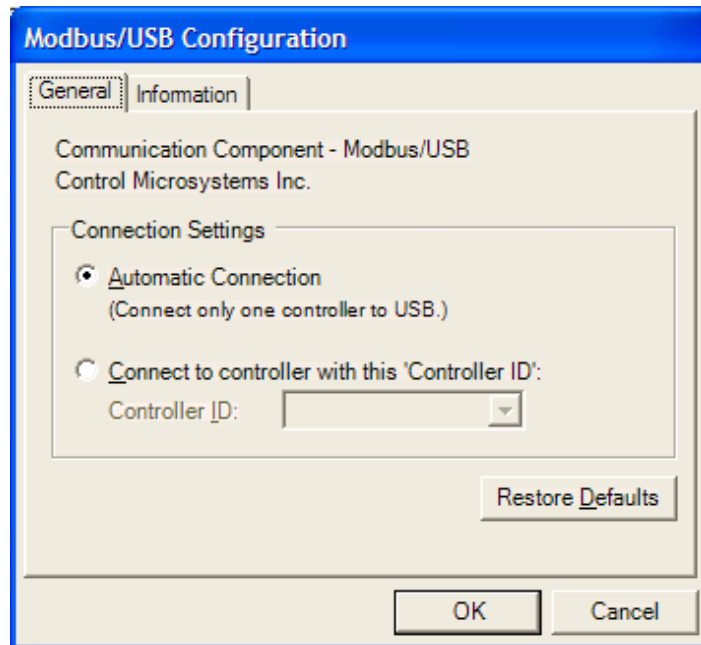


Figure 111: Modbus/USB Configuration (General) Dialog Box

The **Connection Settings** grouping presents two connection options:

- Select **Automatic Connection** when a single controller is present on the bus.

A typical scenario involves a single controller connected directly to a USB port on the host PC.

The driver will reject connection requests by the application if this mode is selected with multiple controllers detected on the bus. In this case, the following error message is displayed:

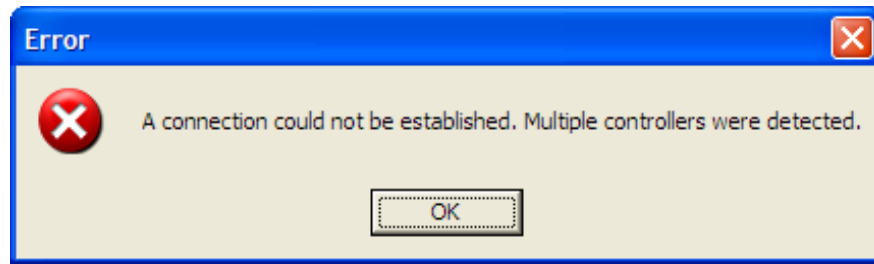


Figure 112: Multiple Controller USB Error Dialog

Note that this option is selected by default.

- Select **Connect to controller with this 'Controller ID** when multiple controllers are present on the bus.

A typical scenario involves more than one USB equipped SCADAPack controller connected via a USB hub to the host PC.

If multiple controllers exist on the bus, the controller ID drop down box will display a list of all identifiable devices. The user connects to the controller in question by selecting its Controller ID from the list. The Controller ID takes on the format 'A123456' and can be found printed on the controller casing or the circuit board.

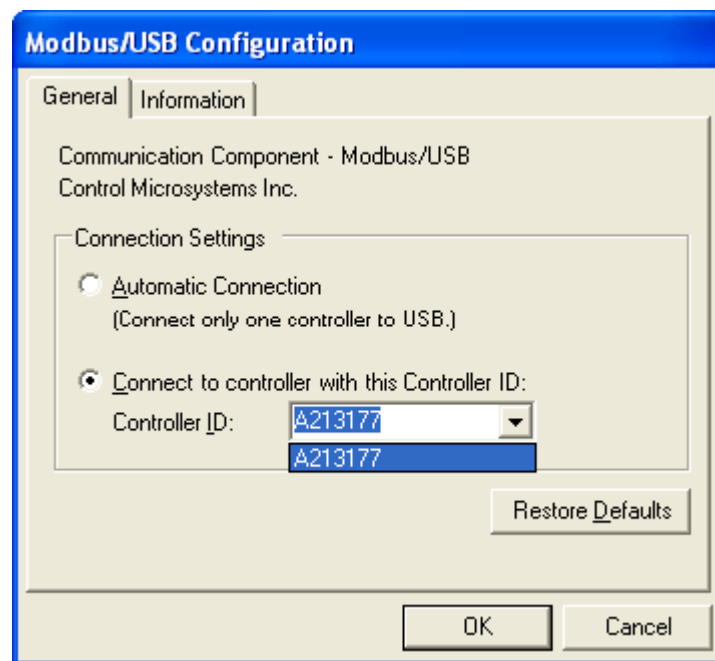


Figure 113: Selecting a USB controller from list

Note that this option can also be used when there is a single controller present on the bus. However, the Controller ID must be known and entered in the Controller ID dialog.

The chosen controller does need to be present on the bus at configuration time.

- Click on the **Restore Defaults** button to reset the page contents to its default state.

In the default state, the **Automatic Connection** option is selected and the **Controller ID** text box is disabled. Any text in the Controller ID text box remains but is displayed in grey.

4.5.3.13.2 Information Page

The Information page identifies the driver type and author. This page further provides detailed driver information which can be useful in identification and troubleshooting scenarios.

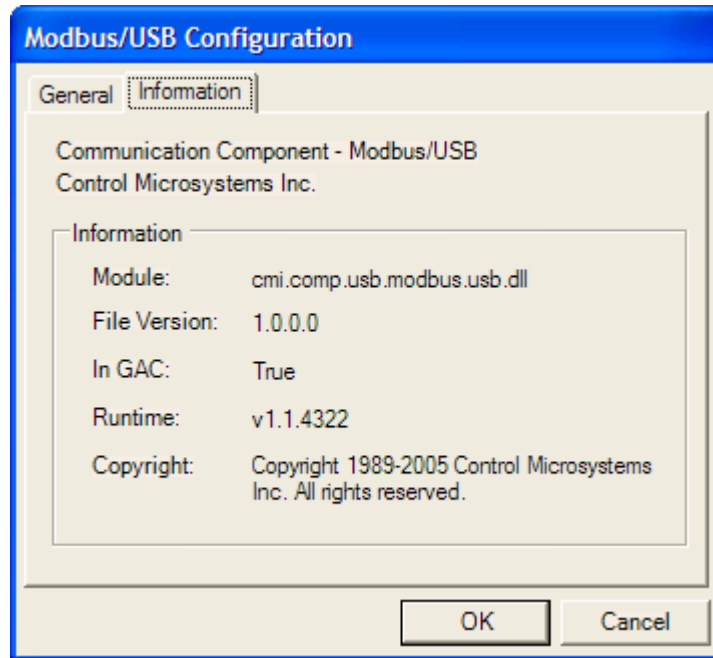


Figure 114: Modbus/USB Configuration (Information) Dialog Box

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

4.5.3.14 SCADA Server

The SCADA Server protocol specifies a SCADA Server Host connection. Applications will act as an OPC client and route all programming commands through the SCADA Server Host to the SCADAPack controller. The type of connection to the field device: no flow control, hardware flow control or dial-up modem is configured in the SCADA Server Host itself.

- To configure a SCADA Server protocol connection, highlight **SCADA Server** in the Communication Protocols window and click the **Configure** button. The SCADA Server Configuration window is displayed.
- To select a configured SCADA Server protocol connection, highlight **SCADA Server** in the Communication Protocols window and click the **OK** button.
- To close the dialog, without making a selection click the **Cancel** button.

4.5.3.14.1 General Parameters

When SCADAServer is selected for configuration the SCADAServer Configuration dialog is opened with the General tab selected as shown below.

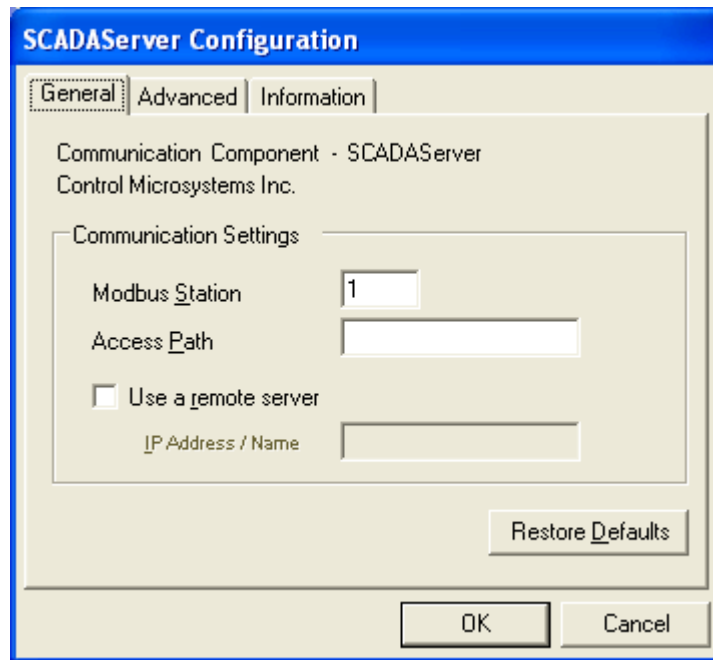


Figure 115: SCADAServer Configuration (General) Dialog Box

The **Communication Settings** grouping contains all essential details necessary to establish communication to a device through a local or remote SCADAServer installation.

The **Modbus Station** parameter specifies the station address of the target device. The valid range is 1 to 65534. The default is station 1.

The **Access Path** parameter specifies the access path to a SCADAServer connection. This parameter is entered as a string with a maximum size of 16 characters. This access path was named when a connection was defined within the SCADAServer installation. If the access path is left blank, the default SCADAServer connection will be used, as defined within the SCADAServer installation. The default for this entry is blank.

The **Use a remote server** check box defines whether the SCADAServer connection uses a SCADAServer installation installed on the same physical PC as the client application or on a remote PC. If the SCADAServer installation is located on a separate machine, check this option and enter the host name or IP address of the remote PC into the "IP Address / Name" edit box. If the SCADAServer installation is located on the same PC as the client application leave this box unchecked. The default state for this check box is unchecked.

The **IP Address / Name** entry specifies the Ethernet IP address in dotted quad notation, or a DNS host name that can be resolved to an IP address, of the PC where the ClearSCADA server is installed. The following IP addresses are not supported and will be rejected:

- 0.0.0.0 through 0.255.255.255
- 127.0.0.0 through 127.255.255.255 (except 127.0.0.1)
- 224.0.0.0 through 224.255.255.255

- 255.0.0.0 through 255.255.255.255.
- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.14.2 Advanced Parameters

Advanced parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput. When the Advanced tab heading is clicked the Advanced dialog is opened as shown below.

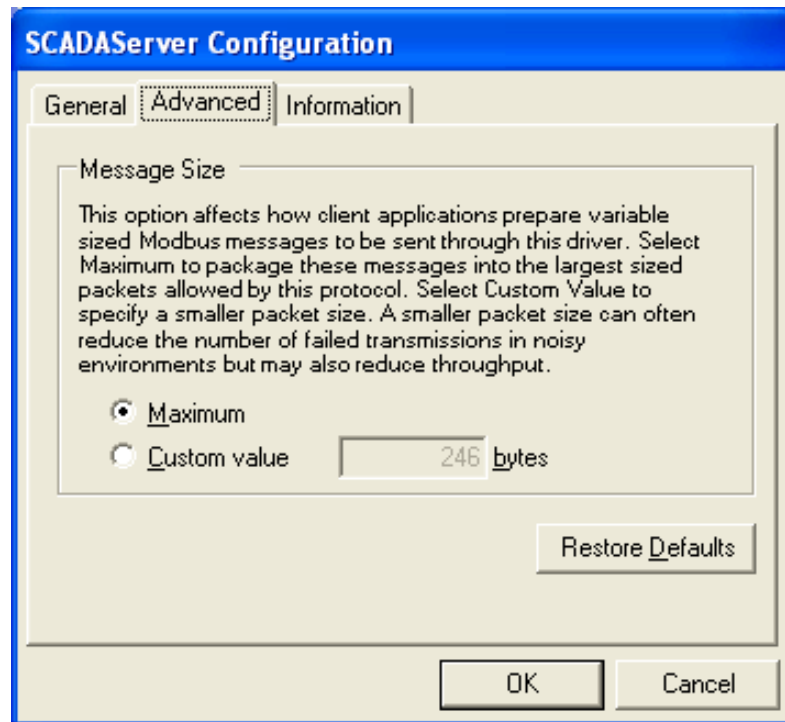


Figure 116: SCADAServer Configuration (Advanced) Dialog Box

The **Message Size** grouping parameters are used to control the message size for the protocol. Control over message length is needed when writing large amounts of data over certain communication networks. A larger value can improve communication speed but can increase the number of failed transmissions. A smaller value can reduce the number of failed transmissions but may reduce throughput.

The **Maximum** selection indicates that the host application is to package messages using the maximum size allowable by the protocol.

The **Custom Value** selection specifies a custom value for the message size. This value indicates to the host application to package messages to be no larger than what is specified, if it is possible. Valid values are 2 to 246.

- Click **Restore Defaults** to restore default values to all fields on this page.

4.5.3.14.3 Information

Information displays detailed driver information. When the Information tab heading is clicked the Information dialog is opened as shown below.

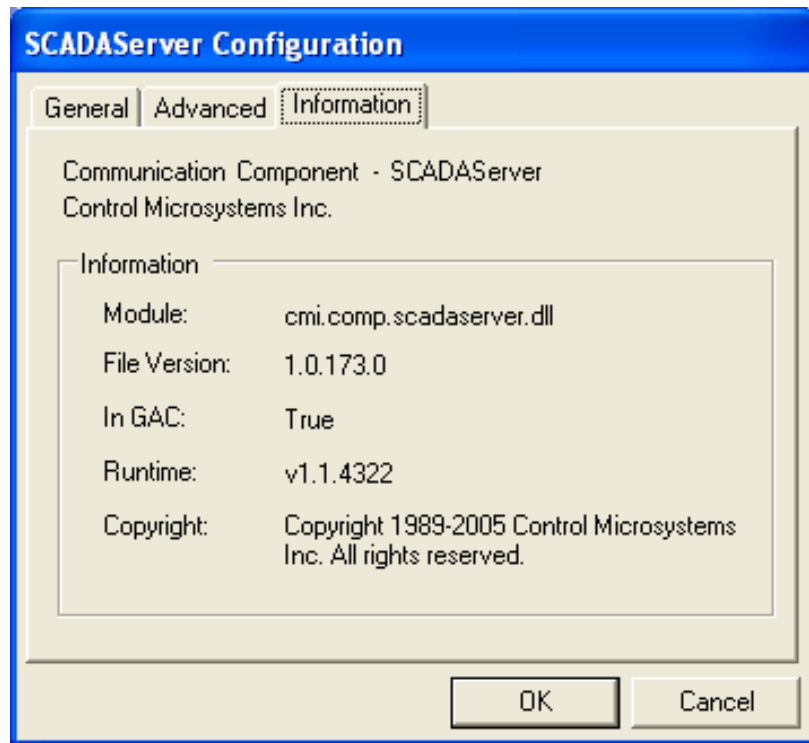


Figure 117: SCADAServer Configuration (Information) Dialog Box

The Information grouping presents informative details concerning the executing protocol driver.

Module is the physical name of the driver.

File Version is the version number of the driver.

In GAC indicates whether the module (assembly) was loaded from the Global Assembly Cache (GAC).

Runtime is the version of the Common Language Runtime (CLR) the driver was built against.

Copyright indicates the copyright information of the protocol driver.

4.6 Activity Menu

The Activity commands select the mode of operation for the TelePACE program. The TelePACE program can operate in three different programming modes, Edit Off Line, Edit On Line and Monitor On Line. A check mark on the Activity drop down menu selection indicates the current mode.

4.6.1 Edit Off Line

The Edit Off Line selection enables editing of a ladder logic program without a connection to a controller. The editing commands affect only the program file in use in the TelePACE Network Ladder Editor. Forcing commands cannot be used in this mode. All other TelePACE commands can be used in this mode.

4.6.2 Edit On Line

The Edit On Line selection is used to edit a ladder logic program loaded into a controller. All editing commands affect the program in the controller and the file in the TelePACE Ladder Network Editor. The multiple element selection and the cut, copy and paste functions of the Editor are not available in this mode. Forcing commands can be used in this mode.

Note: Switch to Edit Off Line mode before performing a Cold Boot on the target controller. The ladder logic application in the TelePACE Network Ladder Editor may be altered by the Cold Boot process.

4.6.3 Monitor On Line

The Monitor On Line selection enables the monitoring of a program executing in a controller. The editor shows the power flow through the ladder network on the screen. The Monitor List window is displayed. No changes can be made to the program in monitor mode. Editing commands are disabled on the menus. Forcing commands can be used in this mode. The C program loader is disabled in this mode.

Note: Switch to Edit Off Line mode before performing a Cold Boot on the target controller. The ladder logic application in the TelePACE Network Ladder Editor may be altered by the Cold Boot process.

4.7 Operation Menu

The Operation menu commands control the execution of the ladder logic program in the controller. (Control of execution of a C Program in the controller is controlled from the C Program Loader dialog.) The status of ladder logic program execution in the controller is displayed on the Status Bar.

4.7.1 Stop

The Stop command stops execution of the ladder logic program in the controller.

4.7.2 Debug

The Debug command start's the execution of the ladder logic program in debug mode. In debug mode the monitor on line feature of the editor can be used to monitor the operation of the program. The program runs slower in debug mode than in run mode.

4.7.3 Run

The Run command starts execution of the ladder logic program in run mode. The monitor on line feature of the editor cannot be used to monitor execution in this mode.

4.8 Options Menu

The Options menu commands are used to modify the Ladder Network Editor display, the use of multiple coils and warning message options. The Option drop-down menu allows selection of the option to modify.

4.8.1 Screen Font

The Screen Font selection selects the font used to display text on the Ladder Editor pane. This selection does not affect the font in the Comment Editor pane, this pane uses the system font. A dialog box appears that allows selection of the Font, Font Style and the Size of the font to be used in the Ladder Editor pane.

4.8.2 Colors

The Colors dialog sets the colors used to display networks on the ladder editor and monitor on line screens. The sample text and elements show how the colors look for the Edit Modes, Monitor Mode, Forced Elements and the Contact Monitor.

The **Item** drop-down list box selects which item to modify. The follow items can be changed:

- Element
- Power Flow
- Cursor
- Forced (Foreground)
- Forced (Background)
- Contact Power Flow
- The **Color** drop-down list box selects the color for the item.
- Click **Default** to return the color selections to default values.
- Click **OK** to accept the changes or **Cancel** to cancel the changes.

4.8.3 Floating-Point Settings

The Floating Point dialog controls the display precision and format of floating point numbers used in TelePACE.

The Display Precision selection sets the number of digits after the decimal place that will be displayed. Valid entries are 1 through 7. The default value is 2.

The Format drop down box selects the format of floating point numbers. The valid selections are:

- Fixed displays numbers in the format #####.##.
- Scientific displays numbers in the format #.### E###.

Click **OK** to close the dialog and save the settings.

Click **Cancel** to close the dialog and discards the changes.

4.8.4 Tool Bar

Selecting the Tool Bar option enables the Tool Bar to be displayed in the Network Editor. A check mark to the left of the selection indicates the Tool Bar is active.

4.8.5 Title Bar

Selecting the Title Bar option enables the Title Bar to be displayed in the Network Editor. A check mark to the left of the selection indicates the Title Bar is active.

4.8.6 Status Bar

Selecting the Status Bar option enables the Status Bar to be displayed in the Network Editor. A check mark to the left of the selection indicates the Status Bar is active.

4.8.7 Single Tag Names

The Single Tag Names selection enables the Editor to use one line to display tag names and addresses on the Editor screen and printed network listings. The first 8 characters of the tag name are

displayed. A numeric value will be displayed if no tag name is defined for the address or constant. A check beside the selection on the pull down menu indicates the mode selected.

4.8.8 Double Tag Names

The Double Tag Names selection enables the Editor to use two lines to display tag names and addresses on the Editor screen and printed network listings. The first 16 characters of the tag name will be displayed. A numeric value will be displayed if no tag name is defined for the address or constant, or if the tag name is 8 characters or less in length. A check beside the selection on the pull down menu indicates the mode selected.

4.8.9 Tag and Address

The Tag and Address selection enables the Editor to use three lines to display tag names and addresses on the Editor screen and printed network listings. The first 16 characters of the tag name are displayed, followed by the numeric address. A numeric value alone will be displayed if no tag name is defined for the address or constant. A check beside this selection on the pull down menu indicates the mode selected. A check beside the selection on the pull down menu indicates the mode selected.

4.8.10 Numeric Address

The Numeric Address selection enables the Editor to use one line for numeric addresses on the editor screen and printed network listings. No tag name is displayed, even if one is defined. A check beside the selection on the pull down menu indicates the mode selected.

4.8.11 Allow Multiple Coils

The Allow Multiple Coils selection enables the use of multiple coils. Multiple coils are coils with the same address.

When the Allow Multiple Coils is not selected the Editor will not allow the insertion of a coil that has an address already in use in the program. An error message is displayed when the insertion is attempted.

When the Allow Multiple Coils is enabled a Dialog box is displayed asking for confirmation when a coil is inserted that has an address already in use in the program. A check beside this selection in the pull down menu indicates multiple coils are allowed.

A check mark is displayed beside this command when it is enabled.

4.8.12 Warning Messages

The Warning Messages option enables additional warning messages. A check mark is displayed beside this command when it is enabled.

When this option is enabled warning messages are displayed on all errors.

When this option is disabled, some warning messages are replaced by a beep. This speeds up editing for experienced users as they do not have to clear a message box when they occasionally make a mistake.

4.9 Help Menu

The Help selection allows access to the TelePACE program extensive on line help. A drop down menu displays the selections for Contents, How to Use Help and About Program.

4.9.1 Contents

The Contents selection displays the TelePACE Program Help Window. Help is available on the following topics.

The **TelePACE Program Reference** describes how each TelePACE command works.

The **Ladder Logic Function Reference** describes the logic function blocks.

The **Controller Register Assignment** describes the I/O modules used in the Register Assignment.

4.9.2 About Program

The About Program selection displays program version information and technical support phone and fax numbers.

5 TelePACE Ladder Logic Functions

The function blocks are described in detail in the pages that follow.

5.1 Register Types

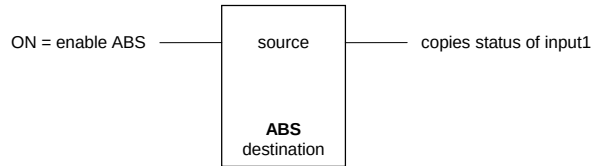
TelePACE function elements support a number of different register formats. The type of register format used will depend on the type function. The following table defines the register formats used.

Unsigned Integer	Unsigned integer data are in the range 0 to 65535. This data requires a single 16-bit register, 3xxxx or 4xxxx.
Signed Integer	Signed integer data are in the range –32768 to 32767. This data requires a single 16-bit register, 3xxxx or 4xxxx.
Unsigned Double	Unsigned long integer data are in the range 0 to 4,294,967,295. This data requires two 16 bit registers, 3xxxx or 4xxxx. The lower numbered register contains the lower order word.
Signed Double	Signed long integer data are in the range -2,147,483,648 to 2,147,483,647. This data requires two 16 bit registers, 3xxxx or 4xxxx. The lower numbered register contains the lower order word.
Floating Point	Floating-point data are in the IEEE single precision floating-point format. This data requires two 16 bit registers, 3xxxx or 4xxxx. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

5.2 ABS – Absolute Value

Description

The **ABS** function stores the absolute value of a signed constant or register into a holding register. When the **enable ABS** is ON the absolute value of the **source** register or constant is stored in the **destination** holding register.



Function Variables

Variable	Valid Types	Description
Source	Constant (-32768...32767) input register (3xxxx) holding register (4xxxx)	a signed value or register
Destination	holding register (4xxxx)	destination = abs(source)

Related Functions

ABSF – Floating Point Absolute Value

Example

network 1:



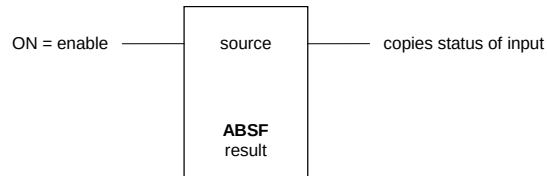
In this example the **absolute value** of the **source**, holding register 44001, is stored in the **destination**, holding register 45211. Assuming holding register 44001 has a value of -4455, the value stored in holding register 45211 would be +4455.

5.3 ABSF - Floating-Point Absolute Value

Description

The ABSF element stores the absolute value of a floating-point register or constant in a floating-point holding register.

When the **enable** input is ON the absolute value of the source is stored in the **result** floating-point holding register. The element output is ON when the input is.



Function Variables

The element has two parameters.

Variable	Valid Types	Description
Source	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	a floating-point register or constant value
Result	2 holding registers (4xxxx)	floating-point absolute value of source register

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38} .

Related Functions

ABS – Absolute Value

Example

network 1:



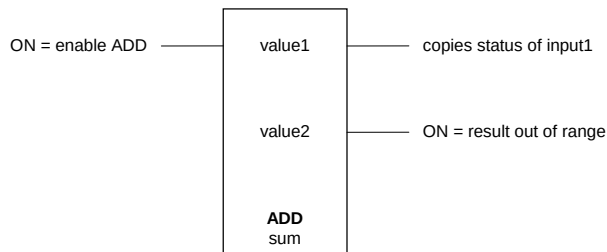
In this example the **absolute value** of the **source**, holding register 40001, is stored in the **result**, floating point register 45555. Assuming floating point register 40001 contains a floating-point value of – 77988.99, the value stored in floating point register 45555 would be +77988.99.

5.4 ADD – Add Signed Values

Description

The **ADD** function block adds two registers or constants and stores the result in a holding register. Signed addition is used. If the result is out of range, the sum rolls over and the out of range output is enabled.

When the **enable ADD** input is ON the sum of **value 1** and **value 2** is stored in **sum** holding register. If the sum is out of range (–32768...32767) the **result out of range** output is ON.



Function Variables

Variable	Valid Types	Description
Value 1	Constant (-32768...32767) input register (3xxxx) holding register (4xxxx)	first signed value to add
Value 2	constant (-32768...32767) input register (3xxxx) holding register (4xxxx)	second signed value to add
Sum	holding register (4xxxx)	sum = value 1 + value 2

Related Functions

Example

network 1:



The ADD function in network 1 adds holding register 42399 to holding register 43567 and stores the result in holding register 44981. When the addition is out of range – less than –32768 or greater than 32767–output coil 00017 is energized. The table below shows some examples.

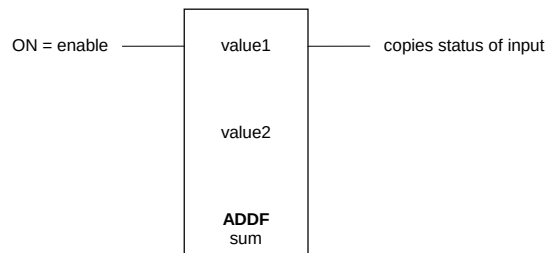
value 1	Value 2	sum	out of range
Holding Register 42399	Holding Register 43567	Holding Register 44981	Output Coil 00017
23411	1098	24509	OFF
-2047	819	-1228	OFF
12767	23000	-29769	ON

5.5 ADDF - Add Floating-Point Values

Description

The **ADDF** element adds two floating-point registers or constants and stores the result in a floating-point holding register.

When the **enable** input is ON the sum **value 1** + **value 2** is stored in the **sum** floating point register. The element output is ON when the input is.



Function Variables

The element has three parameters.

Variable	Valid Types	Description
Value1	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	first floating-point register or constant value to add
Value2	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	second floating-point register or constant value to add
Sum	2 holding registers (4xxxx)	floating-point sum = value1+value2

Notes

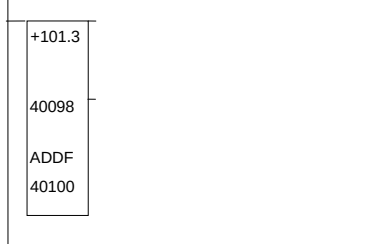
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38} .

Related Functions

Example

network 1:



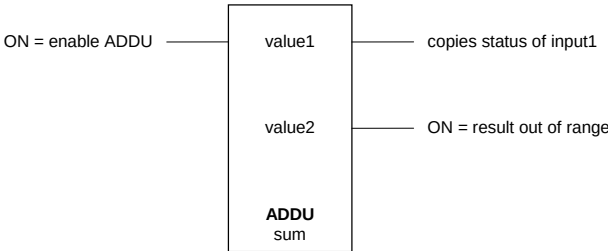
The ADDF function in network 1 adds floating-point constant +101.3 to floating-point register 40098 (registers 40098 and 40099). Assuming floating point register 40098 contains a value of +1000.00 then the result is +1101.30. This value is stored in floating-point register 40100 (registers 40100 and 40101).

5.6 ADDU – Add Unsigned Values

Description

The **ADDU** function block adds two registers or constants and stores the result in a holding register. Unsigned addition is used. If the result is out of range, the sum rolls over and the out of range output is enabled.

When the **enable ADDU** input is ON the sum of **value 1** and **value 2** is stored in **sum** holding register. If the sum is out of range (> 65535) the **result out of range** output is ON.



Function Variables

Variable	Valid Types	Description
Value 1	constant (0...65535) input register (3xxxx) holding register (4xxxx)	first unsigned value to add
Value 2	constant (0...65535) input register (3xxxx) holding register (4xxxx)	Second unsigned value to add
Sum	holding register (4xxxx)	Sum = value 1 + value 2

Related Functions

Example

network 1:



The ADDU function in network 1 adds holding register 46549 to holding register 44723 and stores the result in holding register 44483. When the addition is out of range, greater than 65535, output coil 00021 is energized. The table below shows some examples.

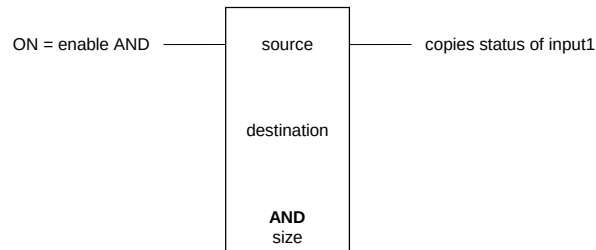
value 1	Value 2	Sum	out of range
Holding Register 46549	Holding Register 44723	Holding Register 44483	Output Coil 00021

value 1	Value 2	Sum	out of range
2048	819	2867	OFF
23000	42530	65530	OFF
35000	35000	4464	ON

5.7 AND – And Block

Description

The **AND** function block logically ANDs the **source** block of registers with the **destination** block of registers and stores the result in the **destination** block of registers. The number of registers in the source and destination blocks is determined by the **size**.



Function Variables

Variable	Valid Types	Description
Source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The first register in the first source block. The address for a coil or status register block is the first register in a group of 16 registers that will be ANDed.
Destination	coil block (0xxxx) holding register (4xxxx)	The first register in the second source block and destination block. The address for a coil register block is the first register in a group of 16 registers that will be ANDed.
Size	constant (1...100)	The number of 16 bit words in the block.

Notes

AND accesses 16 bit words. Coil and status register blocks are groups of 16 registers that start with the register specified as the block address. A block size of 2 corresponds to 32 coils, or two holding registers.

Coil and status register blocks must begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033, etc.

Related Functions

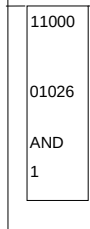
OR – Or Block

XOR – Exclusive Or Block

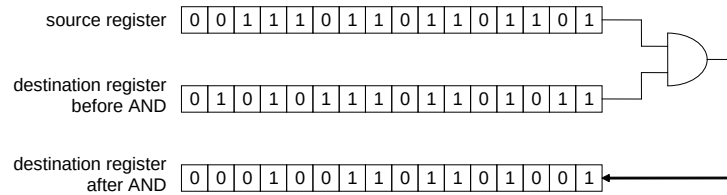
NOT – Not Block

Example

network 1:



In this example the source register has a value of 1521310 (0011101101101101). The destination register has a value of 2237910 (0101011101101011) before the AND function and a value of 496910 (0001001101101001) after the AND function.



5.8 CALL - Execute Subroutine

Description

The **CALL** element transfers execution from the **main** program to a **subroutine**. Execution of the main program is suspended until the subroutine returns.

Subroutine calls can be used to execute parts of a program only when needed. The subroutine is called only if the input to the **CALL** element is ON.

The function variable *number* indicates which subroutine to call. The subroutine number must correspond to the number of a SUBR element.

ON = call subroutine — **CALL** — copy of input
number

Function Variables

The **CALL** element has one parameter:

Variable	Valid Types	Description
Number	Constant	the number of the subroutine. Any number in the range 1 to 500 is valid.

Notes

A subroutine can call other subroutines. This is called nesting. The maximum level of nesting is 20 calls.

Subroutine calls cannot be recursive. For example, subroutine 1 cannot call itself or call another subroutine that calls subroutine 1. This prevents potential infinite loops in the ladder logic program.

The normal order of network evaluation is followed when executing each network. Execution proceeds down the columns, then across the rows. When the **CALL** function is executed, the execution of the current network is suspended at the point of the **CALL** until the subroutine returns.

A subroutine can be called by one or more **CALL** elements.

Related Functions

SUBR - Start of Subroutine

Example

This example calls subroutine 1 when input 10001 is on. Subroutine 1, in turn, calls Subroutine 2. The sequence of execution is as follows.

The networks in the start of the main program are executed.

If input 10001 is on, Subroutine 1 is called.

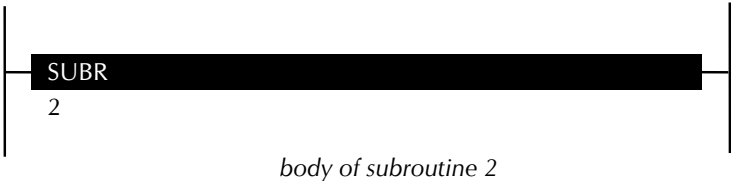
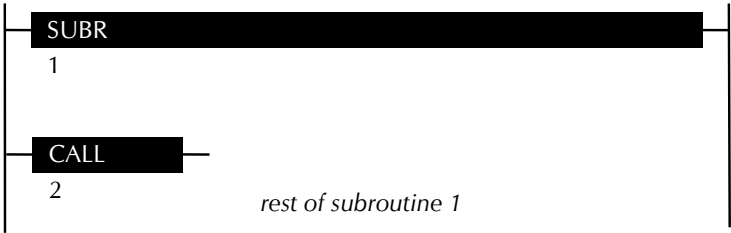
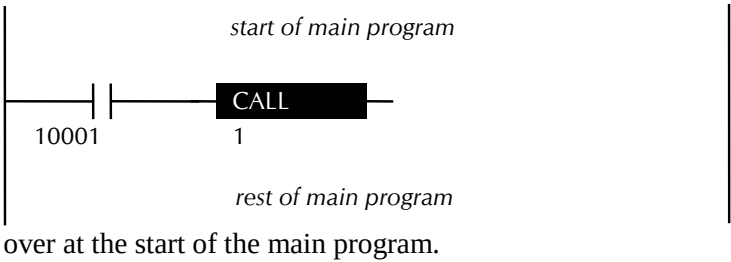
At the beginning of Subroutine 1, Subroutine 2 is called.

The body of Subroutine 2 is executed.

The rest of Subroutine 1 is executed.

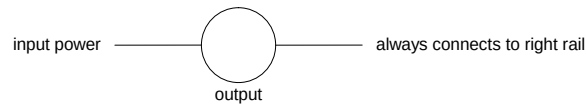
The networks in the rest of the main program are executed.

When execution reaches the start of Subroutine 1, the main program is complete. Execution starts



5.9 Coil

Description



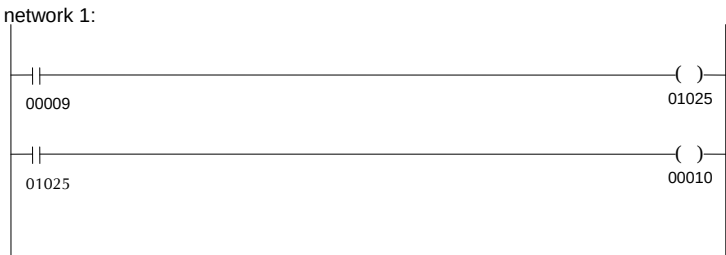
The **COIL** function block enables an **output** coil when its **input power** is ON.

Function Variables

Variable	Valid Types	Description
Output	coil register (0xxxx)	output register address

Related Functions

Example

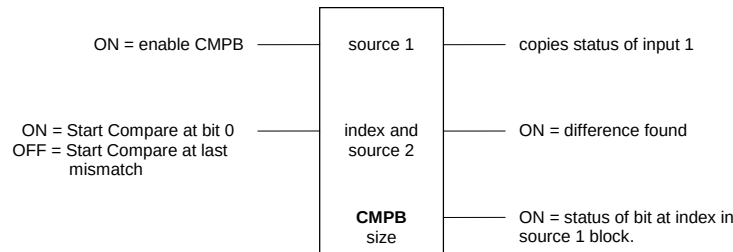


When discrete input contact 00009 is closed power is applied to the **input power** input of coil 01025. This will enable the coil and the associated contact of 01025 is energized. The NO contacts 01025 are closed and this causes power to be applied to output coil 00010.

5.10 CMPB – Compare Bit

Description

The **CMPB** function block compares two blocks of registers, bit by bit and identifies the first bit that does not match. The bit index will be set to the bit offset of the first mismatched bit.



Function Variables

Variable	Valid Types	Description
Source 1	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The first register in the first source block. The address for a coil or status register block is the first register in a group of 16 registers that will be compared.
Index and source 2	holding register (4xxxx)	The bit index register and the second block of source registers. The bit index is in register [4xxxx]. A bit index of 0 is the most significant bit of the first register of the source blocks. The second source register block is at register [4xxxx+1] to register [4xxxx+ size].
Size	constant (1..100)	The number of 16 bit words in the blocks.

Notes

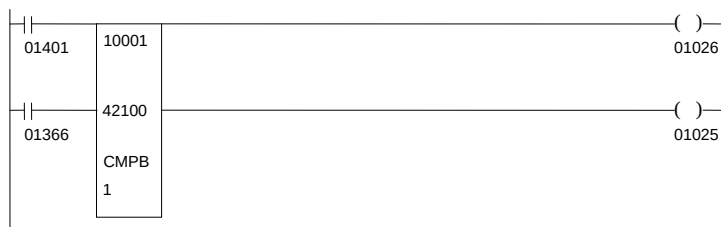
Compare accesses 16 bit words. Coil and status register blocks are groups of 16 registers that start with the register specified as the block address. A block size of 2 corresponds to 32 coils, or two holding registers.

Coil and status register blocks must begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033, etc.

Related Functions

Example

network 1:



The CMPB function in network 1 is used to check the status inputs 10001 to 10016. If Holding Register 42100 is set to zero, with a PUTU function, any status inputs in the range 10001 to 10016 that are on will be detected by the CMPB function.

Using the information in the chart below, status input registers 10004, 10013 and 10014 are on. Closing contacts 01366 will start the index at bit 0 and the index will be set to the bit offset of the first mismatched bit, bit 2. Discrete output coil 01026 will be on and discrete output coil 01025 will be on.

If contacts 01366 are left closed the index will start at bit 0 and be set to the bit offset of the first mismatched bit, bit 2, on each scan of network 1. Discrete output coil 01026 will be on and discrete output coil 01025 will be on.

Opening contacts 01366 will allow the index to start at bit 2 on the next scan of network 1. The index will be set to the bit offset of the next mismatched bit, bit 3. Discrete output coil 01026 will be on and coil register 01025 will be on.

On the next scan of network 1 the index will start at bit 3. The index will be set to the bit offset of the next mismatched bit, bit 12. Coil 01026 will be on and coil register 01025 will be on.

The next scan of network 1 will have the index starting at bit 12. The remaining bits are compared and no further mismatches occur. The index will be set to bit 15 and wait for the contacts 01366 to close to start the compare at bit index 0.

Bit Index:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Discrete Input State:	0	0	1	1	0	0	0	0	0	0	0	0	1	0	0	0
Discrete Input Address: (read top to bottom)	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1

5.11 CMP – Compare Signed Values

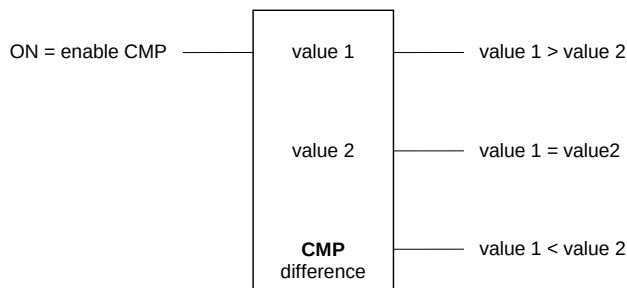
Description

The **CMP** function compares two signed registers or constants and stores the difference in a holding register. When the **enable CMP** input is energized **value 1** is compared to **value 2**.

If value 1 is greater than value 2 the **value 1 > value 2** output is energized and the difference of value 1 minus value 2 is stored in **difference** register.

If value 1 is less than value 2 the **value 1 < value 2** output is energized and the difference of value 2 minus value 1 is stored in the **difference** register.

If value 1 equals value 2 the **value 1 = value 2** output is energized and zero is stored in the **difference** register.



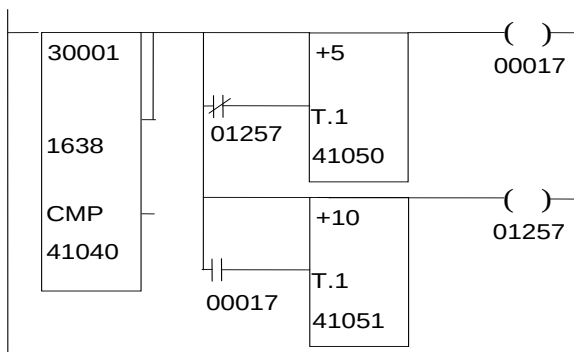
Function Variables

Variable	Valid Types	Description
value 1	Constant (-32768..32767) input register (3xxxx) holding register (4xxxx)	first signed value to compare
Value 2	Constant (-32768..32767) input register (3xxxx) holding register (4xxxx)	Second signed value to compare
Difference	holding register (4xxxx)	if value 1 > value 2 then difference = value 1 - value 2 if value 2 > value 1 then difference = value 2 - value 1

Related Functions

Example

network 1:



The CMP function in network 1 above compares the signed value of Analog Input Register 30001 with the value 1638. The output coil 00017 could be tied to a light to indicate when the Input Register value is over 1638.

The **value 1 > value 2** and **value 1 = value 2** outputs are tied together. When value of register 30001 is greater than or equal to 1638 these outputs are energized. This will enable the flasher circuit of Timers 41050 and 41051 to energize and output coil 00017 will cycle at a rate of 1 second on and 0.5 seconds off.

5.12 CMPU – Compare Unsigned Values

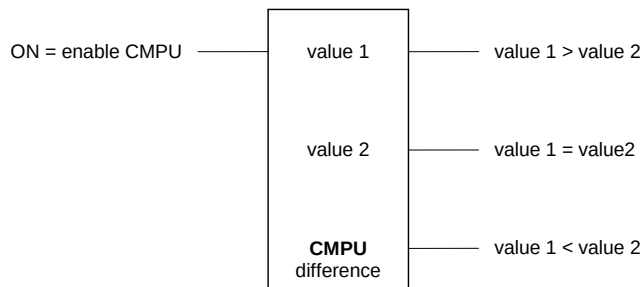
Description

The **CMPU** function compares two unsigned registers or constants and stores the difference in a holding register. When **enable CMPU** input is energized **value 1** is compared to **value 2**.

If value 1 is greater than value 2 the **value 1 > value 2** output is energized and the difference of value 1 minus value 2 is stored in **difference** register.

If value 1 is less than value 2 the **value 1 < value 2** output is energized and the difference of value 2 minus value 1 is stored in the **difference** register.

If value 1 equals value 2 the **value 1 = value 2** output is energized.



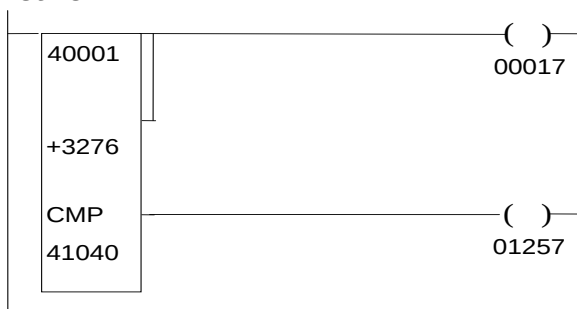
Function Variables

Variable	Valid Types	Description
value 1	Constant (0..65535) input register (3xxxx) holding register (4xxxx)	first unsigned value to compare
Value 2	Constant (0..65535) input register (3xxxx) holding register (4xxxx)	Second unsigned value to compare
Difference	holding register (4xxxx)	If value 1 > value 2 then Difference = value 1 - value 2. If value 2 > value 1 then difference = value 2 - value 1.

Related Functions

Example

network 1:



The CMPU function in network 1 above compares the unsigned value of holding register 40001 with the value 3276. Coil 00017 is energized when holding register 40001 is **greater than or equal to** the constant 3276.

The **value 1 > value 2** and **value 1 = value 2** outputs are tied together.

5.13 DCTR – Down Counter

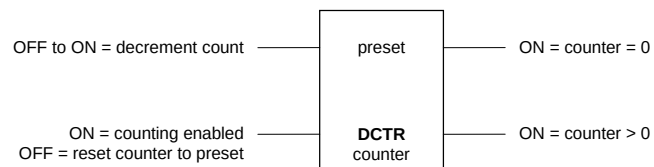
Description

The **DCTR** function block decrements (subtracts 1 from) the value in the **preset** register or constant and stores it in the **counter** holding register when the **decrement count input** changes from OFF to ON. The counter stops counting when it reaches 0.

When the **counting enabled** input is ON and the **decrement count** goes from OFF to ON the counter decrements by one. When the **counting enabled** input is OFF the counter value is reset to preset.

When the **counting enabled** input is ON and the counter has decremented to zero then the **counter = 0** output is ON.

When the **counting enabled** input is ON and the counter is greater than the preset then the **counter > 0** output is ON.



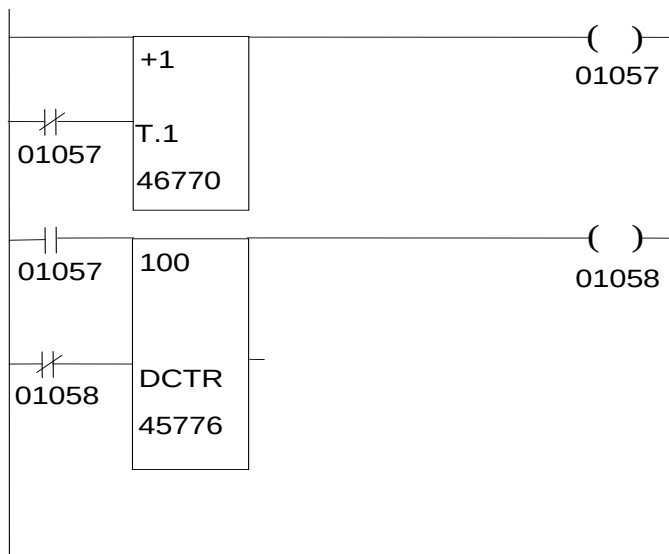
Function Variables

Variable	Valid Types	Description
Preset	constant (1..65535) input register (3xxxx) holding register (4xxxx)	Preset value for counter
Counter	holding register (4xxxx)	Current counter value

Related Functions

Example

network 1:



The DCTR function in network 1 decrements each time the timer output is enabled. The timer limit is 0.1 seconds and it will take 10 seconds for the DCTR to decrement 100 times.

When the DCTR decrements to 0, output coil 01058 is energized. NC contacts 01058 will open and the DCTR will reset counter to preset. This will happen every 10 seconds.

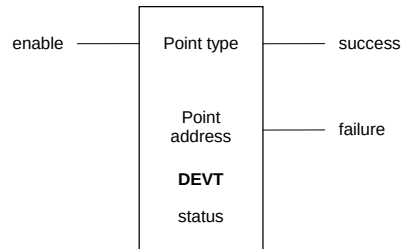
5.14 DEVT – Generate DNP Event

Description

This function generates a DNP change event for the specified DNP point.

A change event is generated for the **Point Address** specified when the **enable** input changes from OFF to ON. The **status** register is set to 0 and the **success** output is energized if the function is successful and a change event was generated.

The status register is set to 1 and the **failure** output is energized if the function failed and a change event was not generated. The failure output indicates that the specified DNP point is invalid, or the DNP configuration has not been created.



Function Variables

Variable	Valid Types	Description
Point Type	Constant or holding register (4xxxx)	DNP Point Type. Valid values are: 0 = Binary Input, 1 = 16-bit Analog Input, 2 = 32-bit Analog Input, 3 = Float Analog Input, 5 = 16-bit Counter Input, 6 = 32-bit Counter Input.
Point Address	Constant or holding register (4xxxx)	DNP Point Address
Status	Holding register (4xxxx)	Status of the operation 0 = success 1 = failure

Related Functions

DUNS – Trigger a DNP unsolicited response message

DPOL – Trigger a DNP class poll

5.15 DIAL – Control Dial-Up Modem

Description

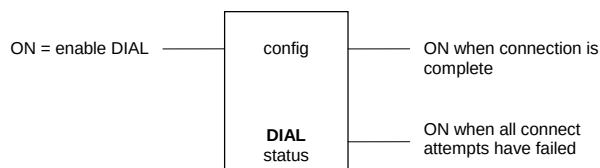
The DIAL function connects an internal modem or an external modem to a remote modem. Only one DIAL block may be active on each serial communication port at any one time. The DIAL function handles all port sharing and multiple dialing attempts.

Note: The SCADAPack 100 does not support dial up connections on com port 1. SCADASense Series controllers do not support dialup connections.

When the **enable DIAL** input is ON the DIAL element sends the modem initialization string to the modem. The modem will attempt to connect to the remote modem. If a connection is made the **ON when connection complete** output is turned ON. If a connection is not made in the specified number of attempts the **ON when all connect attempts have failed** output is turned ON.

The **enable DIAL** input must be energized for the entire DIAL time, including multiple dial attempts.

Note: A pause is required between hanging up the phone line after a DIAL and initiating a new DIAL attempt. This is required for the external modem to hang-up.



Function Variables

Variable	Valid Types	Description
Config	Holding register (4xxxx)	Address of the first register in the configuration block. There are 39 registers in the block at addresses config+0 to config+38. +0 = Communication port +1 = Dialing attempts +2 = Connect time +3 = Pause time +4 = Dialing type +5 = Length of modem string +6 = Modem initialization string +22 = Length of phone number +23 = Phone number NOTE: Registers should be programmed using the Element Configuration command.

Variable	Valid Types	Description
Config	Holding register (4xxxx)	Address of the first register in the configuration block. There are 39 registers in the block at addresses config+0 to config+38. +0 = Communication port +1 = Dialing attempts +2 = Connect time +3 = Pause time +4 = Dialing type +5 = Length of modem string +6 = Modem initialization string +22 = Length of phone number +23 = Phone number NOTE: Registers should be programmed using the Element Configuration command.
Status	Holding register (4xxxx)	address of the first register in the status block. There are 2 registers in the block at addresses status+0 and status+1. +0 = error code +1 = reservation identifier

Element Configuration

This element is configured using the DIAL Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.

WARNING: If the controller is initialized, using the Initialize command in the Controller menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.

DIAL Element Configuration

Addresses 43000 to 43038

Port com2

Attempts 2

Connect Time 60 seconds

Pause Time 15 seconds

Dial Type Tone dialing

Modem Initialization
X4 V1 E0 &C1 &D2 S0=1 S7=50

Phone Number
1-555-555-1212

OK Default Cancel

The modem initialization string and the phone number are packed. Two ASCII characters are stored in each register.

The **error** codes are:

Error Code	Description
0	No Error
1	Bad configuration error occurs when an incorrect initialization string is sent to the modem. This usually means the modem does not understand a specific command in the initialization string.
2	No modem connected to the controller serial port, or the controller serial port is not set to RS232 Modem.
3	Initialization error occurs when the modem does not respond to the initialization string and may be turned off.
4	No dial tone error indicates modem is not connected to a telephone line or the S6 setting in the modem is too short.
5	Busy line indicates another device is using the phone line.
6	Call aborted by the program. This will occur if the enable DIAL input goes OFF before a modem connection occurs.
7	Failed to connect error occurs when there are no other errors but the modem failed to make a connection with the remote modem.
8	Carrier lost error occurs when carrier is lost for a time exceeding the S10 setting in the modem. Phone lines with call waiting are very susceptible to this condition.
9	"Serial port is not available" error occurs when the DIAL function attempts to use the serial port when another ladder communication element, C program or an incoming call has control of the port.

Notes

The reservation identifier is required for the operation of the dialer but can be ignored by the ladder program.

The DIAL function outputs are powered only when the **enable DIAL** input is powered ON.

The dial-up connection handler prevents outgoing calls from using the serial port when an incoming call is in progress and communication is active. If communication stops for more than five minutes, then outgoing call requests are allowed to end the incoming call. This prevents problems with the modem or the calling application from permanently disabling outgoing calls.

Note: The SCADAPack 100 does not support dial up connections on com port 1.
SCADASense Series controllers do not support dial up connections.

To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator display terminals, and for programming and diagnostics using the TelePACE program.

Related Functions

INIM – Initialize Dial-Up Modem

Example

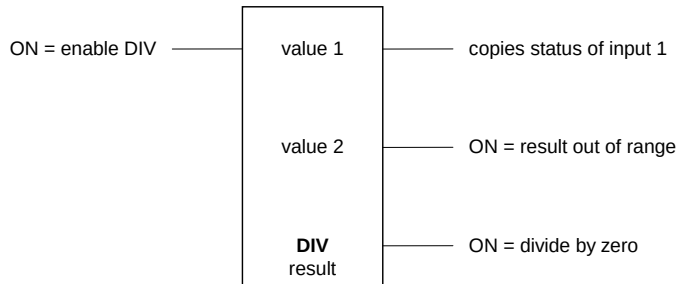
An example of dialing using the DIAL block is the file **DIAL.LAD**. This file is automatically installed on the computer when the TelePACE program is installed. It is located in the **TELEPACE\EXAMPLES** directory.

5.16 DIV – Divide Signed Values

Description

The **DIV** function block divides two registers or a constant, **value 1**, by a register or constant, **value 2**, and stores the quotient in a Holding Register, **result**. Signed division is used. The values and result are signed numbers.

The **result out of range** output is enabled if the result is greater than 32767 or less than –32768. The **divide by zero output** is enabled if **value 2** equals 0.



Function Variables

Variable	Valid Types	Description
value 1	constant (-32768..32767) 2 Input registers (3xxxx) 2 holding registers (4xxxx)	Signed value to divide The low order word is stored in the first register if registers are used.
Value 2	constant (-32768..32767) input register (3xxxx) holding register (4xxxx)	Signed value to divide by
Result	holding registers (4xxxx)	Result = value 1 / value 2

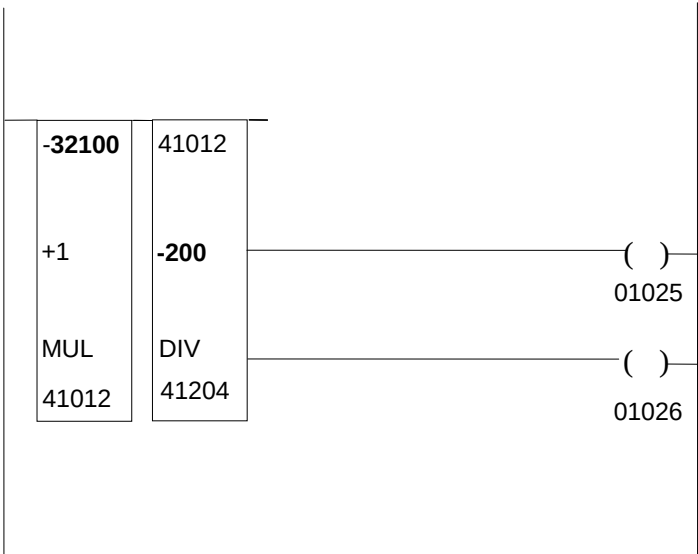
Related Functions

DIVF - Divide Floating-Point Values

DIVU – Divide Unsigned Values

Example

network 1:



The **MUL** function is used to create the double precision values required for **value 1**. A constant is multiplied by 1 with the result stored in Holding Register 41012. The low order word, 41012, and the high order word, 41013, are returned from the MUL function.

The following table illustrates various **value 1** and **value 2** values and the **result** of the division.

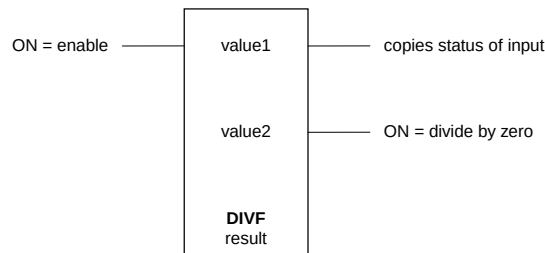
	high order	low order	Constant Value	Register 41204
	Register 41013	Register 41012		
	value 1		value 2	result
Example 1	-1	-32070	-200	160
Example 2	0	32070	-200	-160
Example 3	-1	-2048	16	-128
Example 4	0	3276	30	109

5.17 DIVF - Divide Floating-Point Values

Description

The DIVF element divides one floating-point register or constant by another and stores the result in a floating-point holding register.

When the **enable** input is ON the result **value 1** ÷ **value 2** is stored in the **result** floating-point register. The top output is ON when the input is. The middle output is ON if value2 equals 0 and the input is ON.



Function Variables

The element has three parameters.

Variable	Valid Types	Description
value1	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	Floating-point register or constant value to divide
Value2	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	Floating-point register or constant value to divide by
Result	2 holding registers (4xxxx)	Floating-point result = value1 ÷ value2

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38} .

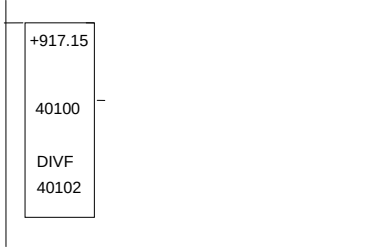
Related Functions

DIV – Divide Signed Values

DIVU – Divide Unsigned Values

Example

network 1:



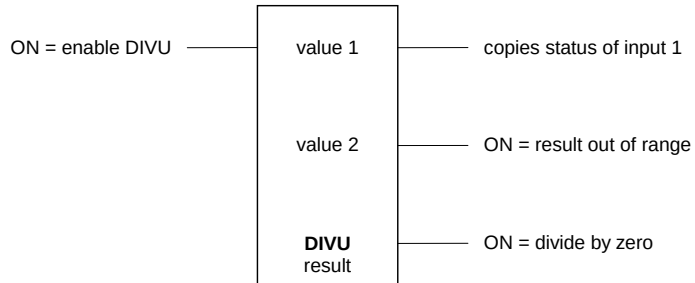
The DIVF function in network 1 divides floating-point constant +917.15 by the contents of floating-point register 40100 (registers 40100 and 40101). Assuming floating-point register 40100 has a value of 5.00 then the result is +183.43. This value is stored in floating-point register 40102 (registers 40102 and 40103).

5.18 DIVU – Divide Unsigned Values

Description

The **DIVU** function block divides two registers or a constant, **value 1**, by a register or constant, **value 2**. The quotient is stored in a Holding Register, **result**. Unsigned division is used. The values and result are unsigned numbers.

The **result out of range output** is enabled if the result is greater than 65535 or less than 0. The **divide by zero output** is enabled if **value 2** equals 0.



Function Variables

Variable	Valid Types	
value 1	constant (0..65535) 2 Input registers (3xxxx) 2 holding registers (4xxxx)	Unsigned value to divide The low order word is stored in the first register if registers are used.
Value 2	constant (0..65535) input register (3xxxx) holding register (4xxxx)	Unsigned value to divide by
Result	holding registers (4xxxx)	Result = value 1 / value 2

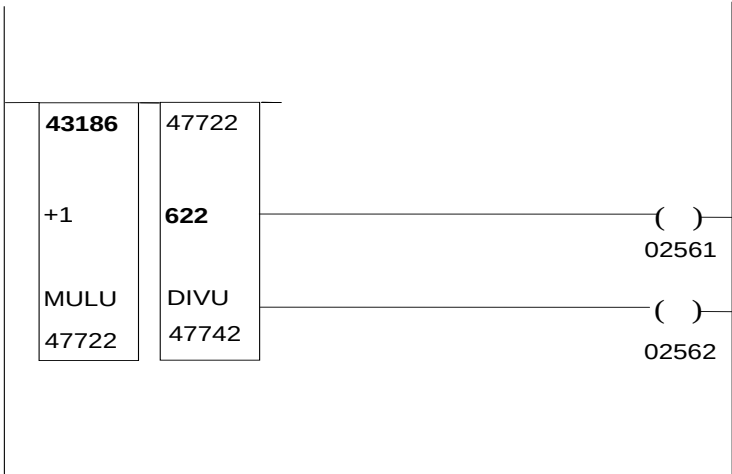
Related Functions

DIV – Divide Signed Values

DIVF - Divide Floating-Point Values

Examples

network 1:



The Mulu function is used to create the double precision values required for **value 1**. A constant is multiplied by 1 with the result stored in Holding Register 47722. The low order word, 47722, and the high order word, 47723, are returned from the Mulu function. The Mulu function ensures a value of 0 in the high order word.

The following table illustrates various **value 1** and **value 2** values and the **result** of the **DIVU** function.

	high order	low order		
	Register 47723	Register 47722	Constant Value	Register 47742
	value 1		value 2	result
Example 1	0	53186	622	85
Example 2	0	7000	70	100
Example 3	0	8192	819	10
Example 4	0	42998	226	190

5.19 DLGF – Data Log to File

Data Log to File Operation Overview

The SCADAPack 330, SCADAPack 334 and SCADAPack 350 controllers support data logging to the internal file system and data logging to a mass storage device connected via the USB host port. Each data log can be independently configured to write data to either the internal file system or a mass storage drive connected through the USB host interface.

Each data log creates a configurable number of data log files. Data Log records are buffered to optimize file-writing dynamics. The log specific buffer is flushed to the active log file once per minute.

Each log can be suspended and resumed. In a suspended state data log records can still be written to the log, but the buffer won't be flushed to file. The state is useful when external medium has to be exchanged.

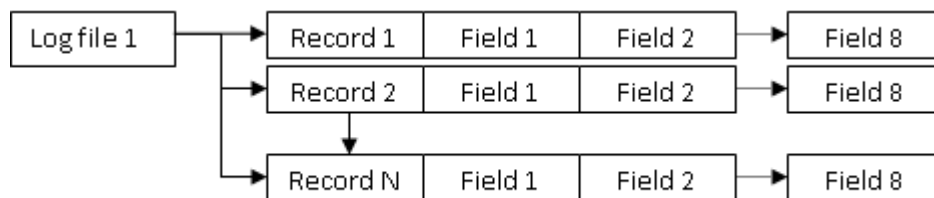
Data Log Storage Medium

Data log files are stored either on the internal file system or an external drive in form of an USB memory stick or an USB hard drive. If more than one USB drive is connected through a hub, only the first drive connected is the active USB log drive.

Logging to File Mechanism

The data is buffered in RAM before being written to file. The buffer holds a maximum of 600 records. This is done to optimize data writes to file and reduce the total CPU load. The data is buffered in non-volatile memory so that a power cycle does not result in the loss of data.

The structure for a single log file is shown below.



A single log file is made up of a configured number of records with each record containing a maximum of 8 data fields. Records are added to the buffered data in RAM each time the log data input for the DLGF function transitions from OFF to ON.

Once per minute the records that have been added to the buffered data in RAM are added to the log file in the internal drive or the mass storage drive. Based on the number of records configured and the logging interval eventually the log file will be full.

There are two methods for handling the full file:

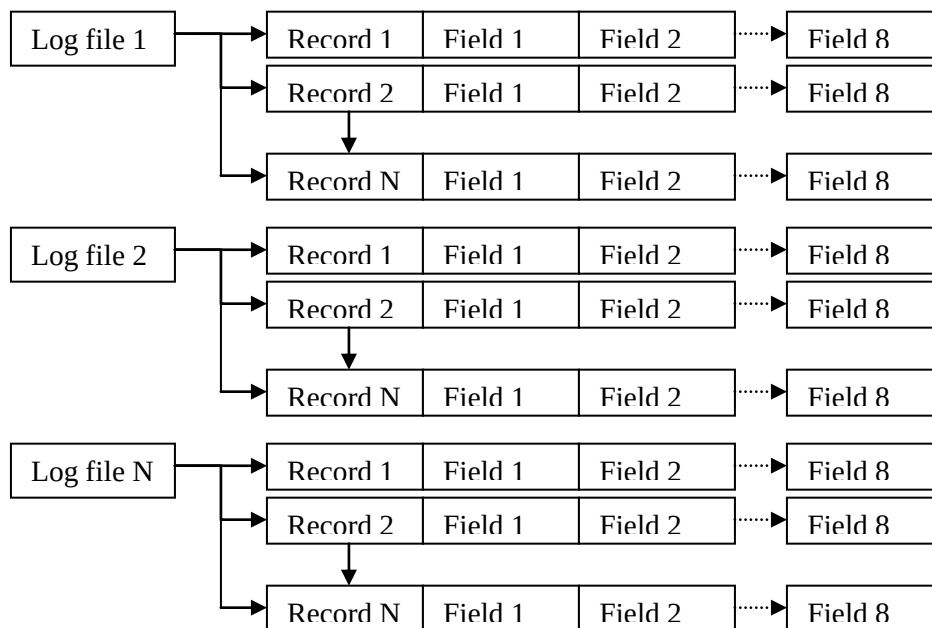
- Logging can be configured to stop when the maximum number of records and maximum number of log files has been reached. In this case there is a single log file configured and logging will stop when all records are filled.

- Logging can be configured to overwrite the oldest log file, with a new file, when the maximum number of log files is reached. In this case there is a single log file configured. The log file will be deleted and a new log file created when all records are filled.

The way full files are handled limits the effectiveness of a single log file. The file system on the internal drive of the SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203 supports multiple data log files.

- The number of **log files** that can be configured for a each instance of the DLGF function is a maximum of 65535.
- Each log file contains a minimum of 600 **records** and is configurable for maximum of 65535 individual records. Each record contains a maximum of 8 configurable data **fields**.

The figure below shows the log file, record and field structure for data logging using multiple log files.



Records are added to the buffered data in RAM each time the log data input for the DLGF function transitions from OFF to ON.

After 1 minute Log file 1 is created and the records that have been added to the buffered data in RAM are added to Log file 1 in the internal drive or the mass storage drive.

This process repeats until the configured number of records per file, Record N, for Log file 1 is reached.

Again records are added to the buffered data in RAM each time the log data input for the DLGF function transitions from OFF to ON.

After 1 minute Log file 2 is created and the records that have been added to the buffered data in RAM are added to Log file 2 in the internal drive or the mass storage drive. Log file 2 is saved in the internal drive or the mass storage drive.

This process repeats until the configured number of records per file, Record N, for Log file 2 is reached and continues to repeat until the configured number of records, Record N, for Log file N is reached.

Now the entire data log is filled. After another 600 records the buffer will be full and no further buffering will occur. The log status register will have the number 17 indicating the **status failed to write data**.

If the full file handling is configured to **Stop logging** when the maximum number of records and maximum number of log files has been reached and the buffer is filled then data logging fails..

- If Logging mode is set for **Internal drive** then the log file is saved on the internal drive.
- If the logging mode is set for **Internal drive, auto copy** then once a mass storage drive is inserted in the USB host port the data files are copied to the mass storage device.
- If the logging mode is set for **Internal drive, auto move** then once a mass storage drive is inserted in the USB host port the data files are moved to the mass storage device.
- If the logging mode is set for **Mass storage drive** then the log file is saved on the mass storage drive.

If the full file handling is configured to **Overwrite oldest log file** when the maximum number of records and maximum number of log files has been reached and the buffer is filled then data logging continues by overwriting the oldest log file. In this case Log file 1 is over written and the process continues repeatedly.

File Formats

Data Log to file requires 2 types of files.

- The first is the directory file which is stored for each log in the specified path on the target drive.
- The second type of file is the actual data files. There may, and likely will be, multiple data files for each log that is created. These data files are in binary format. The SCADALog Data Converter application is used to convert the binary files to a CSV formatted file.

Directory File

The directory file contains the list of all datalog filenames, excluding path information, for the specified log. Each file name is on a new line separated by the log file tag. This file is used to determine the actual log file names so that they can be read by SCADALog, or another process.

The directory file is stored in XML format on the target drive in the same directory as the log files. Its name is *controllerID_logname.xml*. The path is excluded from *controllerID_logname*. The attribute name in the logDirectory tag contains the full log name.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<logDirectory logName="datalogName">
```

```
<!--List of data log files-->
```

```
    <logFile>A288389_ datalogName _20080324_082311.nlg</logFile>
```

```
<logFile>A288389_ datalogName _20080324_083011.nlg</logFile>  
<logFile>A288389_ datalogName _20080324_083611.nlg</logFile>  
<logFile>A288389_ datalogName _20080324_084311.nlg</logFile>  
<logFile>A288389_ datalogName _20080324_085011.nlg</logFile>  
</logDirectory>
```

Log File Names

Log file names are created using the **Log file name** entry in the DLGF Element Configuration dialog. When the files are saved to the internal drive or the mass storage device additional information is added to the file name. Data logs files are named according to the convention:

Annnnnnn_logfile_YYYYMMDD_HHMMSS.nlg

Where:

- **Annnnnnn** is the controller ID
- **logfile** is the path qualified log file name as created in the DLGF Element Configuration..
- **YYYY** is the 4 digit year the file was created
- **MM** is the 2 digit month the file was created
- **DD** is the 2 digit day of the month the file was created
- **HH** is the 2 digit hour (24 hour clock) the file was created
- **MM** is the 2 digit minute the file was created
- **SS** is the 2 digit second the file was created

Note that the file name is path qualified to permit similar data log names on the same controller. i.e. directory1\log and directory2\log are both unique permitted names. The drive is excluded from the file name.

DLGF Description

Note: The DLGF function is supported on the SCADAPack 350, SCADAPack 330 and SCADAPack 334 controllers and the SCADASense 4203 only.

The **DLGF** function records entries to a data log file. When a low to high transition occurs on the **enable DLGF** input, the data log identified by the **config structure** is created and initialized. If the data log currently exists and has a different configuration then an error will be generated. If an error is generated (see **Status Registers** section below) then a different data log name must be logged to, or the first data log must be deleted. When a high to low transition occurs on the **enable DLGF** input the datalog previously created has its configuration deleted.

While the **enable DLGF** is ON and a low to high transition occurs on the **log data** input, an entry is recorded in the data log.

When the **suspend logging** input is high data will be internally buffered, but not written to file until the input goes low. Any data that is in the buffer when the suspend logging goes high will be written to file. This is useful when logging to an external mass storage device. Logging can be suspended to allow the storage device to be exchanged for another.

The **DLGF** function outputs are powered only when the **create log config** input is powered (ON).

The **copies state of input 1** output is a copy of the **create/delete log config** input.

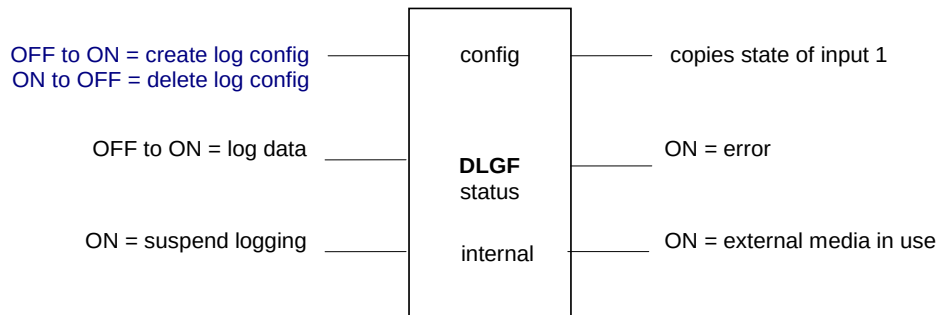
The **error** output is ON if there was a creation, configuration or data logging error. See the **Status Registers** section below for error codes.

The **external media in use** output is enabled (ON) when the data log performs write operations to external USB media. This happens under the following conditions:

- When the logging mode is selected for mass storage drive.
- When the logging mode is selected for auto transfer and the DLGF function block is in the process of transferring files to the mass storage drive.

When the **external media in use** output is OFF the removal of the external mass storage drive is safe from the perspective of this particular DLGF function block. As multiple DLGF functions can be active all DLGF function blocks and any running C/C++ applications need to be considered when removing the external media.

The **external media in use** output is always OFF for DLGF function blocks configured to use the internal drive without auto transfer. It is also OFF when the DLGF function block is configured to use mass storage drive and the **suspend logging** input is high. The output goes ON for DLGF function blocks configured to use auto transfer and a new mass storage device (USB memory stick) is connected. When the output turns OFF this indicates that the write process to that media is finished.



Function Variables

Variable	Valid Types	Description
config	holding register (4xxxx)	Address of the first register of the log configuration block. There are 39 registers in the block at config+0 to config+38 These values are setup by the element configuration and should not be changed while the block is enabled. +0 = Number of data fields +1 = Field #1 register address +2 = Field #1 data type (See Data Types below) +3 = Field #2 register address +4 = Field #2 data type +5 = Field #3 register address +6 = Field #3 data type +7 = Field #4 register address +8 = Field #4 data type +9 = Field #5 register address +10 = Field #5 data type +11 = Field #6 register address +12 = Field #6 data type +13 = Field #7 register address +14 = Field #7 data type +15 = Field #8 register address +16 = Field #8 data type +17 = Logging Mode 0 = internal drive 1 = internal drive, auto copy 2 = internal drive, auto move 3 = mass storage drive +18 = Full File Handling 0 = overwrite oldest log file 1 = stop logging +19 = Number of log files, minimum 1 +20 = Number of records per file. Minimum 600

Variable	Valid Types	Description
		+21, +22 = Security Token; note that the input through TelePACE only allows 8 hex characters, e.g. 1A2B3C4D; the security token is only considered for auto transfer logs which are not supported on controllers without USB host port. +23 - +38 Log Name, path qualified
status	holding register (4xxxx)	Address of the status register (see below for all status code descriptions) +0 = log status code +1 = auto transfer status code +2 = media status code
internal	holding register (4xxxx)	Identifies the internal log. This value is used internally and must not be modified. +0, +1 = Internal Data

The configuration **Field #n register address** is the address of a Modbus register, or the address of the first Modbus register of a small block of consecutive registers for floating point, signed double and unsigned double data. Valid values are any register in the range 40001 to 49999.

The configuration **Field #n data types** are described in the following table.

Data Type	Description
0	16 bit unsigned integer
1	16 bit signed integer
2	32 bit unsigned integer
3	32 bit signed integer
4	32 bit floating point
5	Days and hundredths of seconds in two 32 bit unsigned integers. The first two registers are an Unsigned Double containing the number of complete days since 01/01/97 The last two registers are an Unsigned Double containing the number of hundredths of a second since the start of the current day.

Status Registers

The log status register stores the result of a log attempt. If the log status register is not equal to 10 then the error output is turned on.

The **log status** codes are:

Status Code	Description
10	The configuration is valid and data can be logged.
11	A different configuration already exists for the log.
12	The log ID is invalid or has not been created.
13	One or more data types are invalid
14	There is not enough memory available to create a log of the requested size.
15	The number of data fields was invalid.

17	Failed to write data
18	Invalid configuration

The auto transfer status register indicates the status of an auto transfer operation. A status code unequal 0 signals the user that no more log files have to be transferred. This includes the case that no files can be transferred due to a missing or wrong security token. Note that the status could indicate that the auto transfer is in progress with the **external media in use** output set to OFF. This is the case if not all log files could be transferred due to full media. DLGF blocks not configured to use auto transfer always show the value 4. The same value is displayed when an auto transfer was never triggered for a configured log.

The **auto transfer** status codes are:

Status Code	Description
0	Auto transfer is not done or in progress.
1	Auto transfer is done; all files (at least one) were transferred.
2	Auto transfer is done; no log file had to be transferred .
3	Auto transfer is done because no valid security token could be found.
4	Auto transfer not configured or not started.

The media status register shows the condition of the target media. The value 0 indicates an error free status, while the value 1 indicates a warning as long as the log buffer isn't full. A full buffer is indicated by log status 17.

The **media transfer** status codes are:

Status Code	Description
0	Media is present.
1	No external media present.
2	External media is full.
3	Internal media is full.
4	External and internal media are full.

DLGF Function Configuration

This DLGF function is configured using the DLGF Element Configuration dialog. To open the DLGF Element Configuration highlight the DLGF function by moving the cursor over the function and clicking the mouse. Then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.

WARNING: If the controller is initialized, using the Initialize command in the Controller menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.

When Element Configuration is selected for the DLGF function the following dialog is displayed.

DLGF Element Configuration

Addresses 42000 to 42038

Field	Registers	Type	Tag Name
1	----	time and date	
2	30001	unsigned	Raw Flow Rate
3	41000 to 41001	floating point	Flow Volume
4	41002 to 41003	floating point	Injection Press
5	41004	unsigned	Bit Map Alarms

Logging mode Mass storage drive

Full file handling Overwrite oldest log file

Number of log files 7

Records per file 1440

Security token 0

Log file name A_Field\Site12

OK
Cancel
Add
Edit
Delete
Move Up
Move Down

Addresses

This is the register address range that was entered for the **Data log configuration block** entry in the DLGF Insert / Edit Network Element dialog. These registers are described in the **Function Variables** section above.

Records Area

The Records area of the DLGF Element Configuration describes the data that is going to be logged and the order of the fields in each record. A single record is comprised of a maximum of 8 fields. Each field is individually configured using the Add/Edit Log Field dialog.

- To open the Add/Edit Log Field dialog click the **Add** button for a new field entry or click the **Edit** button to modify an existing field entry. See the **Add/Edit Log Field Dialog** section below for a description of the dialog parameters.
- To sort the field entries use the **Move Up** and **Move Down** buttons.

Logging Mode

Data logs may be logged to the internal drive or to an external mass storage drive. When stored to the internal drive data logs can be selected for automatic copying or moving to the mass storage drive when the mass storage drive is inserted into the USB host port on the controller. To allow a degree of confidentiality the data log can be configured to only transfer data if a specific security token is present on the inserted mass storage device.

The following options for **logging mode** are available.

- Internal drive
- Internal drive, auto copy

- Internal drive, auto move
- Mass storage drive

The transfer maintains any folder structure that exists on the internal file system. In other words the files will remain in the same relative position to the root of the drive. Folders are created on the mass storage device if needed.

Logs that the token grants access to are copied or moved to the mass storage device. Each data log is suspended before its log files are copied or moved. The data log is resumed if no more log files can be copied or moved. This is the case if there are no more log files to be copied or moved or if the mass storage medium is full.

If the log is configured to move the files they are deleted after the move is successful. In the event of a failure part way through the process, the files will remain on the internal file system and the operation can be attempted again later.

Full File Handling

The data is buffered in RAM before being written to file. The data buffer will hold 600 records. This is done to optimize data writes to file and reduce the total CPU load. The data is buffered in non-volatile memory so that a power cycle does not result in the loss of data.

There are two methods for handling full files:

- Logging can be configured to **stop logging** when the maximum number of records and maximum number of log files has been reached.
- Logging can be configured to **overwrite the oldest log file**, with a new file, when the maximum number of log files is reached.

In the event that no disk space remains the data will be buffered in RAM memory until that buffer fills as well, at which point new records will be discarded.

The data is written to the active log file, from buffered RAM, at 1 minute intervals.

Number of Log Files

The **number of log files can** parameter can be any value between 1 and 65535.

Records per File

The number of **Records per file** is any value between 600 and 65535.

Security Token

The security token is a 32 bit value containing **8 hex characters**. This token provides for a method of authenticating the mass storage device as a trusted and authorized device. This method of authentication does meet the criteria for 2 factor security because it requires something you know (the security token), and something you have (physical access).

- If a data log is configured to have a security token value of **0** then the security is disabled on this log.

Security tokens are stored in an XML script file. The tokens are ASCII representations of a 32-bit value, i.e. if the security token value was 0x00579ACE then it would be represented as 00579ACE in the token file.

The file must be named **controller.xml** and reside in the root directory of the mass storage device.

Security Token File Format

```
<controller>
<securityTokens>
    <allControllers>
        <token>token1</token>
        <token>token2</token>
        ...
        <token>tokenN</token>
    </allControllers>
</securityTokens>
</controller>
```

There is no limit to the number of items that can be listed in the securityTokens section. The file may contain more information than that listed above, but any additional information will not be used. Any token listed must be represented as a hex value without any prefix or suffix, for example:

```
<token>A1B2C3D4</token>
```

The security tokens are not keyed to the mass storage device that they reside on. This improves the usability of the security token; however it does reduce the security offered by the token since the tokens could be freely copied from one device to another.

When a security token is read that matches that of a data log then the data log data will be transferred if that the data log is configured for auto transfer mode.

If a data log is configured to have a security token value of 0x00000000 then the security is disabled on this log.

Because security tokens are checked on external USB media for auto transfer logs, the definition of the token during log creation is meaningless for pure internal or mass storage logging modes as well as for any logs on controllers which do not support an USB host.

Log File Name

Log file names are created using the **Log file name** entry in the DLGF Element Configuration dialog. The log file name can be up to 32 characters long and can contain folder information as well as the log name.

5.20 DLOG - Data Logger

Description

The **DLOG** function records entries in a data log. When a low to high transition occurs on the **enable DLOG** input, the data log identified by `LogID` is created and initialized. If the data log exists and has a different configuration then an error will be generated. If an error is generated (see status codes below) then a different `LogID` must be used or the log must be deleted using the **delete log** input.

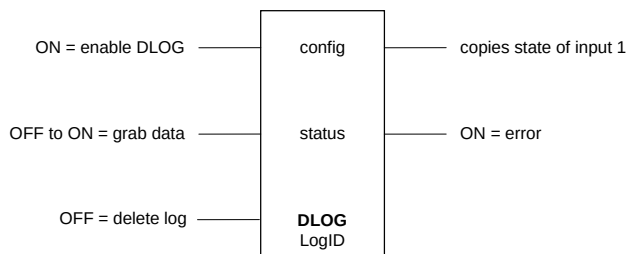
While the **enable DLOG** is ON and a low to high transition occurs on the **grab data** input, an entry is recorded in the data log identified by `LogID`.

When the **delete log** input is low, all records in the data log are deleted.

The top output is a copy of the top input. The **error** output is ON if there was a creation, configuration or data logging error. The outputs are powered only when the enable input is powered.

NOTE: In order for the changes to an existing DLOG configuration to take effect, the following steps must be followed:

- Read the data from the log, if desired.
- Disable the DLOG
- Delete the log
- Write the new configuration
- Enable the log



Function Variables

Variable	Valid Types	Description
Config	18 holding registers (4xxxx)	Address of the first register of the log configuration block. There are 18 registers in the block at config+0 to config+17. +0 = Maximum number of records in the log. +1 = number of data fields. +2 = Field #1 register address +3 = Field #1 data type (see data types below) +4 = Field #2 register address +5 = Field #2 data type +6 = Field #3 register address

Variable	Valid Types	Description
		+7 = Field #3 data type +8 = Field #4 register address +9 = Field #4 data type +10 = Field #5 register address +11 = Field #5 data type +12 = Field #6 register address +13 = Field #6 data type +14 = Field #7 register address +15 = Field #7 data type +16 = Field #8 register address +17 = Field #8 data type
Status	Holding register (4xxxx)	Address of the status register. +0 = status code (see status codes below)
Log ID	Constant (1 to 16)	Identifies the internal log.

Configuration Register

A Field Register Address is the address of a Modbus register, or the address of the first Modbus register of a small block of consecutive registers. Valid values are any register in the range 00001 to 49999.

The configuration data types are:

Data Type	Description
0	16 bit unsigned integer
1	16 bit signed integer
2	32 bit unsigned integer
3	32 bit signed integer
4	32 bit floating point
5	Days and hundredths of seconds in two 32 bit unsigned integers. The first two registers are an Unsigned Double containing the number of complete days since 01/01/97 The last two registers are an Unsigned Double containing the number of hundredths of a second since the start of the current day.

Status Register

The status register stores the result of a log attempt. It can have the following values. If the status register is not equal to 10 then the error output is turned on.

The status codes are:

Status Code	Description
10	The configuration is valid and data can be logged.
11	A different configuration already exists for the log.
12	The log ID is invalid.
13	The configuration was not valid and was rejected.
14	There is not enough memory available to create a log of the request size.

Status Code	Description
15	The number of data fields was invalid.
16	The log was successfully deleted. No log configuration exists.
17	Undefined status. This should never occur.

DLOG Element Configuration

This element is configured using the DLOG Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.

WARNING: If the controller is initialized, using the Initialize command in the Controller menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.

DLOG Element Configuration Dialog

Selecting Element Configuration from the Edit menu while the cursor is over a DLOG element can access this dialog.

Field	Registers	Type	Tag
1	40001	unsigned	
2	40002	signed	
3	40003 to 40004	unsigned double	
4	40005 to 40006	floating point	
5	----	time and date	
6	30001	unsigned	
7	30002	unsigned	
8	30003	unsigned	

The **addresses** are determined from the ladder element. They cannot be modified in this dialog.

The **Number of Records** selects the number of records in the log. Each record contains the data for all the fields in the log. The number of records available will depend on the type of controller used.

For **SCADAPack 100: 256K** controllers the maximum number of records is 190 records of 8 unsigned integer fields.

For **SCADAPack LP and SCADAPack 100: 1024K** controllers the maximum number of records is approximately 22970 records of 8 unsigned integer fields.

For **SCADAPack** and **Micro16** controller the number of records depends on the available memory in socket U10 in the controller. For example a 128K RAM in U10 will allow approximately 190 records of 8 unsigned integer fields and a 512K RAM in U10 will allow approximately 22970 records of 8 unsigned integer fields.

For **SCADAPack 32** controllers the maximum number of records is 58,352 records of 8 unsigned integer fields for a total of 466816 words. The maximum number of records will be reduced if DNP protocol, RealFLO or custom C++ applications are used in the controller.

For **SCADAPack 330, SCADAPack 334 and SCADAPack 350** controllers the maximum number of records is 56,743 records of 8 unsigned integer fields for a total of 453947 words. The maximum number of records will be reduced if DNP protocol, RealFLO or custom C++ applications are used in the controller.

The **Field** entries contain information about each item in the log. The fields are numbered, 1 through 8, as they are added to the record. The **Registers** column contains the Modbus registers used in for

the field. The **Type** column contains the Modbus register type and the **Tag** column contains the tag name for the Modbus Register.

Press the **Add** key to add an item to the end of the list. When selected the **Add/Edit DLOG Field** dialog is displayed, see description of this dialog below. The key is grayed if the list is full.

Press the **Edit** key to modify the currently selected item in the list. When selected the Add/Edit DLOG Field dialog is displayed. The key is grayed if the list is empty.

Press the **Delete** key to delete the currently selected item from the list. The key is grayed if the list is empty.

Press the **Move Up** key to move the currently selected item up one position in the list. The key is grayed if the list is empty or the selected item is at the top of the list.

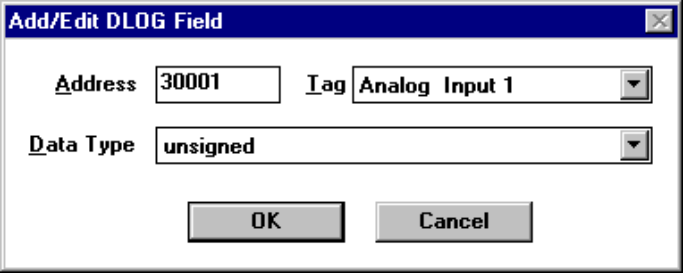
Press the **Move Down** key to move the currently selected item down one position in the list. The key is grayed if the list is empty or the selected item is at the bottom top of the list.

Press **OK** to accept the current list.

Press **Cancel** to discard the changes made to the list.

Data Log Field Dialog

This dialog can be accessed by pressing **Add** or **Edit** from the DLOG Element Configuration dialog.



The **Address** entry will contain the Modbus register address for the record if the Edit button was selected. This entry will be zero if the Add button was selected. Valid values are any register in the range 00001 to 49999.

The **Tag** entry contains the tagname for the selected register address if one exists. The drop down list will display all tag names for valid address. Selecting a tag name from the list will cause the associated register address to be displayed in the Address entry.

The **Data Type** selection allows the selection of one of the following data types:

16 bit unsigned integer

16 bit signed integer

32 bit unsigned integer

32 bit signed integer

32 bit floating point

Days and hundredths of seconds in two 32 bit unsigned integers

Press **OK** to validate the data and close the dialog.

Press **Cancel** to discard the changes made in the dialog.

The output bits are powered only when the enable input is powered.

Related Functions:

GETL – DLOG Extract

5.21 DPOL – Trigger a DNP class poll

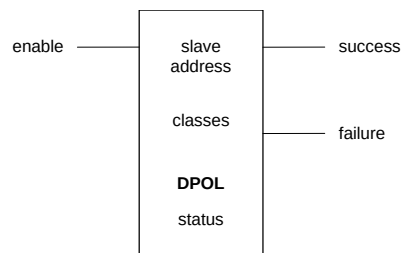
Description

This function sends a DNP Class Poll message to request the specified data classes from a DNP slave. The DNP class poll is requested when the **enable** input changes from OFF to ON.

The **status** register is set to 0 and the **success** output is energized if the function was successful.

The 'success' output is energized immediately if the poll was triggered successfully. It does not provide any information about the success or otherwise of the actual message transaction.

The **status** register is set to 1 and the **failure** output is energized if the function failed. Failure indicates the specified slave address has not been configured in the DNP Routing Table, or the DNP configuration has not been created.



Function Variables

Variable	Valid Types	Description
Slave Address	Constant or holding register (4xxxx)	DNP slave station address
Classes	Constant or holding register (4xxxx)	This is a bit-mapped register containing the classes of data to be included in the class poll message. If multiple classes are required the following values should be added together. 1 = class 0 data 2 = class 1 data 4 = class 2 data 8 = class 3 data
Status	Holding register (4xxxx)	Status of the operation 0 = success 1 = failure

Notes

This function is available on the SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADAPack 32 controllers only.

The function sets internal flags to trigger a DNP poll message, and returns immediately. The DNP message will be sent some time after the function call.

Related Functions

DEVT – Generate DNP Event

DUNS – Trigger a DNP Unsolicited Response Message

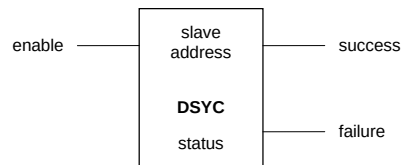
5.22 DSYC – Trigger a DNP clock synchronization

Description

This function sends a Clock Synchronization message to a DNP slave.

The Clock Synchronization message is sent to the **slave address** when the **enable** input changes from OFF to ON. The status register is set to 0 and the **success** output is energized if the function was successful.

The **status** register is set to 1 and the **failure** output is energized if the Time Synchronization message failed. The **failure** output indicates the specified slave address has not been configured in the DNP Routing Table, or the DNP configuration has not been created.



Function Variables

Variable	Valid Types	Description
Slave Address	Constant or holding register (4xxxx)	DNP slave station address
Status	Holding register (4xxxx)	Status of the operation 0 = success 1 = failure

Notes

This function is available on the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers only.

The function sets internal flags to trigger a DNP message and returns immediately. The DNP message will be sent some time after the function call.

Related Functions

DEVT – Generate DNP Event

DPOL – Trigger a DNP Class Poll

DUNS – Trigger a DNP Unsolicited Response Message

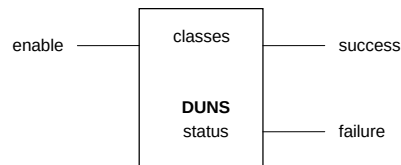
5.23 DUNS – Trigger a DNP unsolicited response message

Description

The DUNS function triggers a DNP unsolicited response message of the specified class or classes.

The unsolicited response for the specified **classes** is triggered when the **enable** input changes from OFF to ON. The **status** register is set to 0 and the success output is energized if the message was triggered successfully.

The **status** register is set to 1 and the **failure** output is energized if the unsolicited response message failed. The failure output indicates master addresses have not been configured in the DNP Routing Table, or the DNP configuration has not been created.



Function Variables

Variable	Valid Types	Description
Classes	Constant or holding register (4xxxx)	This is a bit-mapped register containing the classes of data to be included in the unsolicited message. If multiple classes are required the following values should be added together. 1 = class 0 data 2 = class 1 data 4 = class 2 data 8 = class 3 data
Status	Holding register (4xxxx)	Status of the operation 0 = success 1 = failure

Notes

The function sets internal flags to trigger a DNP unsolicited message, and returns immediately. The DNP message will be sent some time after the function call.

DNP unsolicited messages do not have to be enabled in the Application Layer configuration for events to be sent with this function. Unsolicited messages at Start Up must be enabled, or the master must enable unsolicited messages before the first message can be sent.

If no events are pending an empty unsolicited message will be sent.

The message will be sent to the configured DNP master station or stations.

The return status indicates whether the flags were set. The status will be FALSE if the DNP engine is not running.

Related Functions

DPOL – Trigger a DNP Class Poll

DUNS – Trigger a DNP Unsolicited Response Message

5.24 FIN – FIFO Queue Insert

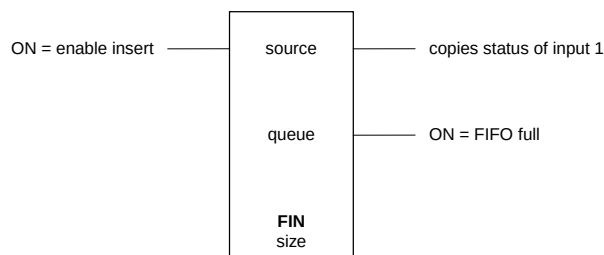
Description

The **FIN** function inserts the contents of the source register into the First-In-First-Out (FIFO) queue. Values in the FIFO queue are pushed down until the queue is full.

When the **enable insert** input is ON and the **FIFO full** is not ON, the source register is inserted into the queue.

The index is incremented by one after each insert until the index value equals the size. When the index equals the size and **enable insert** is ON then **FIFO full** turns ON and no further insertions can take place.

The FIFO Queue Insert function is used with the FIFO Queue Remove function to enable full control of the FIFO Queue.



Function Variables

Variable	Valid Types	Description
Source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The address of the source register. The address for a coil or status register block is the first register in a group of 16 registers that will be inserted.
Queue	holding register (4xxxx)	The address of the index register and the queue registers. The index is stored in register [4xxxx]. The FIFO queue is located in register [4xxxx+1] to register [4xxxx + size].
Size	constant (1..9999)	The number of registers in the queue.

Notes

FIN accesses 16 bit words. Coil and status register blocks are groups of 16 registers that start with the register specified as the block address. Coil and status register blocks must begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033, etc.

Related Functions

FOUT – FIFO Queue Remove

Example

See the example for the FOUT function.

5.25 FOUT – FIFO Queue Remove

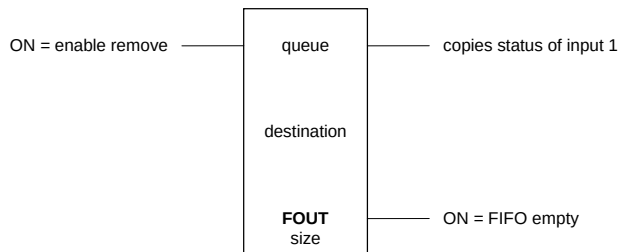
Description

The **FOUT** function removes a value from the first-in-first-out (FIFO) queue and transfers it into the destination register. Values in the queue are moved up, until the queue is empty.

When the **enable remove** input is ON and the **FIFO empty** is not ON, the first queue register is transferred into the destination register.

The index is decremented by one after each transfer until the index value equals zero. When the index equals zero and the **enable remove** input is ON the **FIFO empty** turns ON and no further removals can take place.

The FIFO Queue Remove function is used with the FIFO Queue Insert function to enable full control of the FIFO Queue. See FIFO Queue example.



Function Variables

Variable	Valid Types	Description
Queue	Holding register (4xxxx)	The address of the index register and the queue registers. The index is stored in register [4xxxx]. The FIFO queue is located in register [4xxxx+1] to Register [4xxxx + size].
Destination	coil block (0xxxx) holding register (4xxxx)	The address of the destination register. The address for a coil register block is the first register in a group of 16 registers.
Size	Constant (1..9999)	The number of 16 bit registers in the queue.

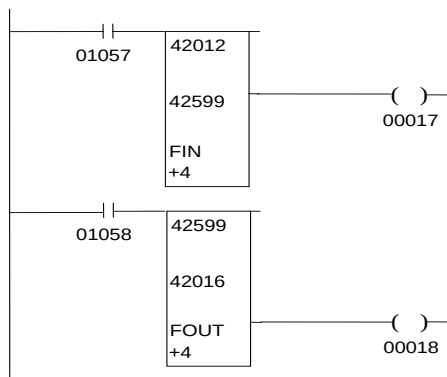
Notes

FOUT accesses 16 bit words. Coil blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, etc.

Related Functions

FIN – FIFO Queue Insert

Example



The following discussion describes the operation of a FIN function. The queue starts out empty with the FIFO index register having a value of zero.

Source Register	Source Register Data	FIFO Queue Registers
42012	Xxxx	FIFO Index--> 42599
		FIFO register 0--> 42600
		FIFO register 1--> 42601
		FIFO register 2--> 42602
		FIFO register 3--> 42603

When FIFO queue index register 42599 has a value of 0 and the FIN function is enabled, by closing contact 1057, the contents of source register 42012 is inserted into FIFO Queue register 42600 and the index is incremented to 1.

When FIFO queue index register 42599 has a value of 1 and the FIN function is enabled, by closing contact 1057, the contents of queue register 42600 is transferred to queue register 42601. The contents of source register 42012 are inserted into FIFO queue register 42600, and the index is incremented to 2.

When FIFO queue index register 42599 has a value of 2 and the FIN function is enabled, by closing contact 1057, the contents of queue register 42601 is transferred to queue register 42602. The contents of queue register 42600 is transferred to queue register 42601, the contents of source register 42012 is inserted into FIFO queue register 42600 and the index is incremented to 3.

When FIFO queue index register 42599 has a value of 3 and the FIN function is enabled, by closing contact 1057, the contents of queue register 42602 is transferred to queue register 42603. The contents of queue register 42601 is transferred to queue register 42602, the contents of queue register 42600 is transferred to queue register 42601, the contents of source register 42012 is inserted into FIFO queue register 42600 and the index is incremented to 4.

The queue is now full and no further insertions are allowed. Coil 00017 will turn ON. The FOUT Queue Remove function must be used to remove values from the queue.

The following discussion describes the operation of a FOUT function. The queue starts out full with the FIFO index register having a value of three.

	FIFO Queue Registers	FIFO Queue Register Data	Destination Register
FIFO Index-->	42599		
FIFO register 0- ->	42600		
FIFO register 1- ->	42601		
FIFO register 2- ->	42602		
FIFO register 3- ->	42603	xxxx	42016

When FIFO queue index register 42599 has a value of 4 and the FOUT function is enabled, by closing contact 1058, the contents of queue register 42603, pointed to by the index is transferred to the FIFO destination register 42016 and the index is decremented by one.

When FIFO queue index register 42599 has a value of 3 and the FOUT function is enabled, by closing contact 1058, the contents of queue register 42602 is transferred to the FIFO destination register 42016 and the index is decremented by one.

When FIFO queue index register 42599 has a value of 2 and the FOUT function is enabled the contents of queue register 42601 is transferred to the FIFO destination register 42016 and the index register is decremented by one.

When FIFO queue index register 42599 has a value of 1 and the FOUT function is enabled, by closing contact 1058, the contents of queue register 42600 is transferred to the FIFO destination register 42016 and the index is decremented to 0.

The queue is now empty and no further removals are allowed. Coil 00018 will be now be energized. The FIN Queue Insert function must be used to insert values into the queue.

5.26 FLOW – Flow Accumulator

Description

The FLOW function accumulates flow from pulse type devices such as turbine flow meters.

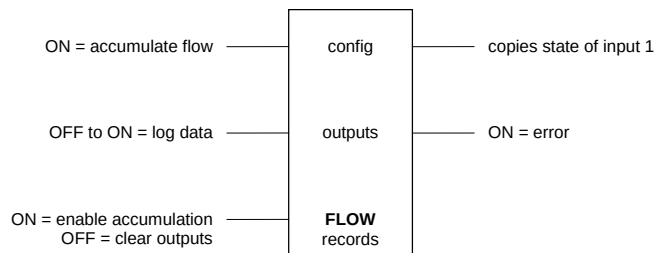
When the accumulate input is ON the function accumulates volume and calculates the flow rate.

When the Log Data input goes from OFF to ON, the accumulated volume, flow time and the time at the end of the period is saved in the history registers. Older history is pushed down and the oldest record is discarded. The Log Data input must be triggered at least once every 119 hours (at the maximum pulse rate). Otherwise the volume accumulator will overflow, and the accumulated volume will not be accurate.

Note: The accuracy of the FLOW block improves with longer periods between logging data. For greatest accuracy avoid logging small periods of time.

When the accumulator enabled input is ON, accumulation and rate calculations are enabled. When the input is OFF, all accumulators and the rate outputs are set to zero.

The function reads and accumulates the number of pulses, and divides by the K factor to calculate the total volume. This is done on each scan of the controller logic. The function calculates the flow rate in engineering units based on the K-factor provided. The rate updates once per second if there is sufficient flow. If the flow is insufficient, the update slows to as little as once every ten seconds to maintain resolution of the calculated rate.



Function Variables

Variable	Valid Types	Description
Config	holding register (4xxxx)	Address of the first register of the configuration block. There are five registers in the block at config+0 to config+4. +0, 1 = K factor (floating-point) +2 = input register (3xxxx or 4xxxx) +3 = input type (see table below) +4 = rate period (see table below)

Variable	Valid Types	Description
Outputs	holding register (4xxxx)	Address of the first register of the output block. There are 18 to 222 registers in the block at outputs+0 to outputs+221. The number of registers used depends on the records variable. +0 = status (see table below) +1,2 = flow rate (floating-point) +3,4 = internal: rate accumulator +5,6 = internal: time of last rate +7,8 = internal: input at last sample +9,10 = internal: pulse accumulator +11 = number of records to follow +12,13 = volume for period 1 (float) +14,15 = time at end of period 1 +16,17 = flow time for period 1 +18,19= volume for period 2 (float) +20,21 = time at end of period 2 +22,23 = flow time for period 2 +24,25= volume for period 3 (float) +26,27 = time at end of period 3 +28,29 = flow time for period 3 ... +216,217 = volume for period 35 (float) +218,219 = time at end of period 35 +220,221 = flow time for period 35
Records	Constant	The number of measurement records stored in the output array. The valid values are 1 to 35. This value determines the number of output registers used by the function.

Element Configuration

This element is configured using the FLOW Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.

WARNING: If the controller is initialized, using the Initialize command in the Controller menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.

The **K Factor** is a floating-point value. It may be any non-zero, positive value.

The **Input Register** is the address of the register that holds the input pulse count. Valid values are any input or holding register in the range 30001 to 49999.

The **Input Type** may be one of the following.

16 bit counter A free running 16-bit counter.

Holding register (4xxxx) = 0

32 bit counter A free running 32-bit counter with low word in the first register.

Holding register (4xxxx) = 1

16 bit difference The 16-bit difference between any two counters.

Holding Register (4xxxx) = 2

The **Rate Period** determines the units of time used to display the flow rate. It may be one of the following. Note that this value does not affect when the rate calculation is performed.

per second Holding register (4xxxx) = 0

per minute Holding register (4xxxx) = 1

per hour Holding register (4xxxx) = 2

per day Holding Register (4xxxx) = 3

Output Registers

The output registers store the results of the flow accumulation and act as internal workspace for the accumulation.

The **status** register indicates the status of the flow accumulation. It can have the following values. If the status register is non-zero, the error output is turned ON.

Status Register Value	Description
0	no error
1	invalid <i>K factor</i> configuration
2	invalid <i>Input Register</i> configuration
3	invalid <i>Input Type</i> configuration
4	invalid <i>Rate Period</i> configuration
5	pulse rate is too low for accurate flow rate calculation

The **flow rate** register contains the flow rate in engineering units based on the K-factor and rate period provided.

The eight **internal** registers are used for the accumulation of the flow and calculation of the flow rate.

The **Number of Records** register indicates how many sets of flow and time registers follow. This value is equal to the *Records* variable. There are four registers in each record.

The **Volume for Period n** is stored as a floating-point number in two consecutive registers.

The **End Time for Period n** is stored as a 32-bit integer in two consecutive registers. The registers hold the number of seconds since January 1, 1970. This is an unsigned number.

The **Flow Time for Period n** is stored as a 32-bit integer in two consecutive registers. The register holds the number of seconds flow was accumulated in the period. This measures the time the Accumulate input is ON, including times when the flow was zero.

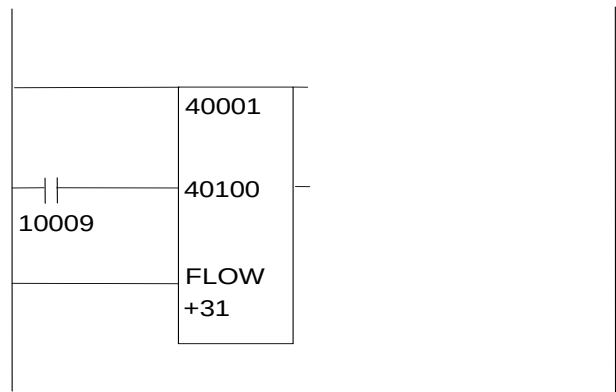
Notes

The maximum pulse rate is 10 kHz.

The accumulated flow is calculated by measuring the accumulated pulses and dividing by the K factor. The K factor cannot be changed while an accumulation is in progress. Only change the K factor while accumulation is stopped.

Example

network 1:



The FLOW function in network 1 has the *accumulate flow* and *enable accumulation* inputs connected to the left power rail. When these inputs are continuously powered the FLOW function accumulates volume and calculates flow rate.

Each time contact 10009 is closed, powering the log data input, the accumulated volume, flow time and the time at the end of the period is saved in the history registers.

The *records* value is set to 31. This means that 31 sets of history registers, volume, flow time and end of period time, will be logged. In this example the accumulation input is never turned off meaning that once 31 sets of history registers are saved the next time contact 10009 is closed the 31st record is removed and the newest record is added to the history.

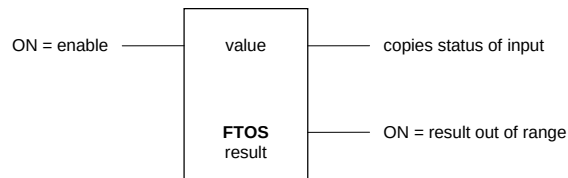
5.27 FTOS - Floating-Point to Signed Integer

Description

The FTOS element converts a floating-point register or constant into a signed integer and stores the result in a holding register.

When the **enable** input is ON, **value** is converted into a signed number and stored in the **result** register. The value is rounded to the nearest integer. If the value is less than -32768 , then -32768 is stored. If the value is greater than 32767 , then 32767 is stored.

The top output is ON when the input is. The bottom output is ON if the floating-point value is less than -32768 or greater than 32767 .



Function Variables

The element has two parameters.

Variable	Valid Types	Description
value	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	a floating-point register or constant
Result	holding register (4xxxx)	Converted signed value

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38} .

Related Functions

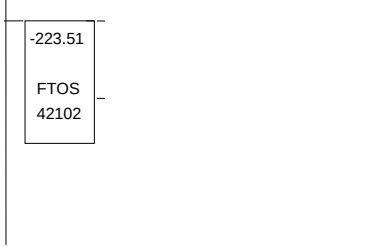
UTOF - Unsigned Integer to Floating-Point

STOF - Signed Integer to Floating-Point

FTOU - Floating-Point to Unsigned Integer

Example

network 1:



The FTOS function in network 1 converts floating-point constant -223.51 to a signed integer value and puts the value into register 42102. In this example the content of register 42102 is the signed integer -224 .

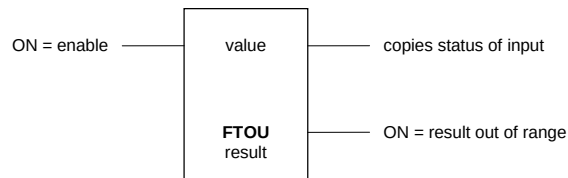
5.28 FTOU - Floating-Point to Unsigned Integer

Description

The FTOU element converts a floating-point register or constant into an unsigned integer and stores the result in a holding register.

When the **enable** input is ON, **value** is converted into an unsigned number and stored in the **result** register. The value is rounded to the nearest integer. If the value is less than zero, then zero is stored. If the value is greater than 65535, then 65535 is stored.

The top output is ON when the input is. The bottom output is ON if the floating-point value is less than 0 or greater than 65535.



Function Variables

The element has two parameters.

Variable	Valid Types	Description
value	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	a floating-point register or constant
Result	holding register (4xxxx)	Converted unsigned value

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38} .

Related Functions

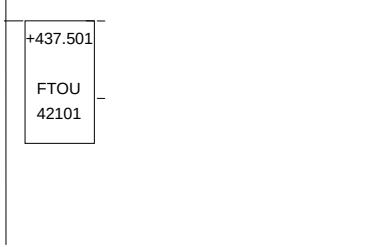
UTOF - Unsigned Integer to Floating-Point

STOF - Signed Integer to Floating-Point

FTOS - Floating-Point to Signed Integer

Example

network 1:



The FTOU function in network 1 converts floating-point constant +437.501 to an unsigned integer value and puts the value into register 42101. In this example the content of register 42101 is the unsigned integer +438.

5.29 GETB – Get Bit from Block

Description

The **GETB** function block reads the status of a bit, at the bit index, from the source block of registers.

When the **bit index** is defined as a holding register the **enable GETB** input must be ON to increment bit index or reset bit index to zero.

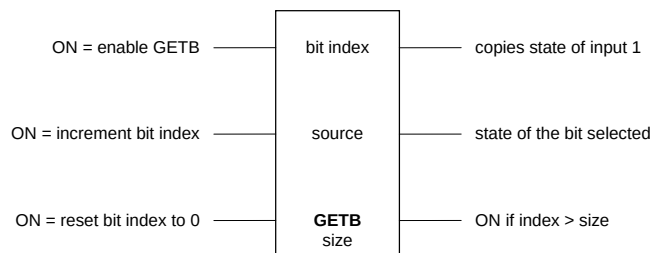
The **increment bit index** input increments the bit index once, each scan, when enabled.

The **reset bit index to 0** input resets the bit index to 0 when enabled.

When the bit index is defined as a constant, the GETB function checks the status of the bit at the bit index when **enable GETB** is ON. The **increment bit index** and **reset bit index to 0** inputs have no effect on the function block in this condition.

When the **enable GETB** is ON, the **state of bit selected** output is ON if the source bit is 1 and OFF if the source bit is 0.

The **index>size** output is enabled when the bit index is equal to **size** + 1 and the **enable GETB** input is ON. The **reset bit index to 0** input must be set to continue function operation.



Function Variables

Variable	Valid Types	Description
bit index	Constant (0..65535) holding register (4xxxx)	The index of the bit within the source block. A bit index of 0 is the most significant bit of the first register of the source block.
Source	coil block (0xxxx) holding register (4xxxx)	The address of the source register. The address for a coil register block is the first register in a group of 16 registers.
Size	Constant (1..100)	number of 16 bit words in the block.

Notes

The Get bit from block function accesses 16 bit words. Coil blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, etc.

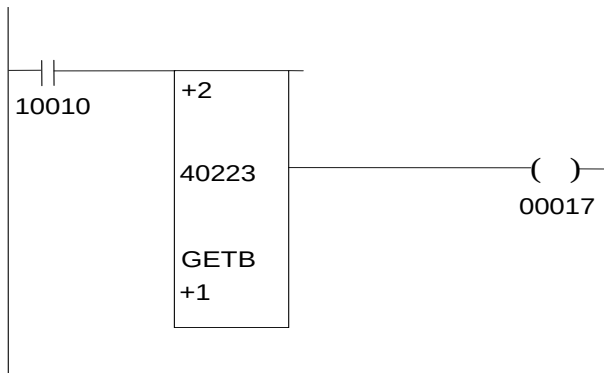
Related Functions

PUTB – Put Bit into Block

ROTB – Rotate Bits in Block

CMPB – Compare Bit

Example



In this example the GETB function is used to test bit 13 of the PID block 0 Status register to check if the Control Block is in manual mode.

The twenty-five holding registers used for PID block 0 are assigned to the **CNFG PID Control Block I/O Module** in the Register Assignment. In this case holding registers 40220 through 40245 are assigned to the I/O module.

The Control Block status register, 40223 is tested to see if bit 13 is ON. The GETB function uses bit index of 0 for the most significant bit. This means that a bit index of 2 will test for bit 13 of the Control Block register.

Closing contacts 10010 sets the **enable GETB** input on. If PID block 0 is in manual mode output 00017 will turn ON.

5.30 GETL - Data Logger Extract

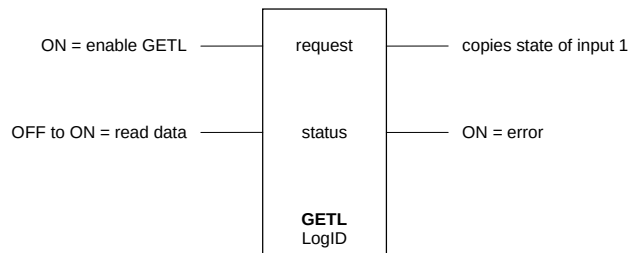
Description

The **GETL** function extracts one record from a data log and writes it into holding registers.

When a low to high transition occurs on the **enable GETL** input, the error code and sequence number status registers are updated and the data length is cleared.

When the **enable GETL** input is ON and a low to high transition occurs on the **read data** input, the status registers are updated and the data for the requested sequence number is copied to the data from log holding registers.

The **error** output is ON if the log is not configured or if there was a data reading error, see table below. The outputs are powered only when the enable input is powered.



Function Variables

Variable	Valid Types	Description
Request	2 holding registers (4xxxx)	Unsigned long sequence number requested from the log. +0 = least significant word +1 = most significant word
Status	Up to 36 holding registers (4xxxx)	Status and data registers +0 = error code +1,2 = sequence number +3 = length of data +4 to+35 = data from log
LogID	Constant (1 to 16)	Identifies the internal log.

Status Register

The status register stores the result of an enable log attempt and a log data attempt. It can have the following values. If the status register is does not contain the code 20 then the error output is turned on. The **status** error codes are:

Error Code	Description
20	The configuration is valid and data can be retrieved.
21	The log ID is invalid.
22	The log is not configured.
23	The requested sequence number was not in the valid range.

Notes:

To get the oldest sequence number in the log toggle the **enable GETL** input, the error code and sequence number status registers are updated and the data length is cleared. To clear the error 23 the **enable GETL** input must be turned off.

Related Functions:

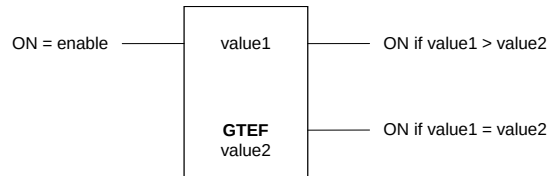
DLOG - Data Logger

5.31 GTEF – Floating-Point Greater Than or Equal

Description

The GTEF element tests if a floating-point register or constant is greater than or equal to another floating-point register or constant.

When the **enable** input is ON, value1 and value2 are compared. If value1 is greater than value2, the top output is ON. If value1 is equal to value2 the bottom output is ON.



Function Variables

The element has two parameters.

Variable	Valid Types	Description
Value1	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	a floating-point register or constant
Value2	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	a floating-point register or constant

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number. Floating point numbers can represent positive and negative values in the range – 3.402×10^{38} to 3.402×10^{38} .

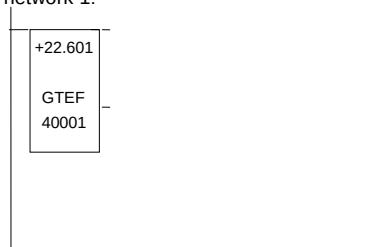
Related Functions

CMP – Compare Signed Values

CMPU – Compare Unsigned Values

Example

network 1:



The GTEF function in network 1 compares the floating-point constant +22.601 to the contents of floating-point register 40001. The top output of the GTEF function is ON if the value of register 40001 is less than +22.601. The bottom output of the GTEF is ON if the value of register 40001 is equal to +22.601.

5.32 HART – Send HART Command

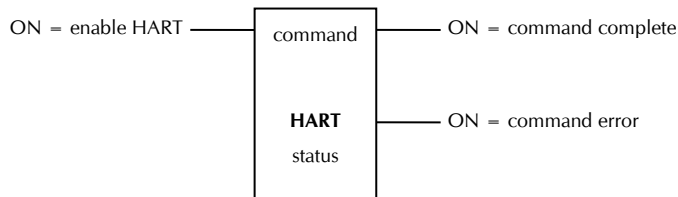
Description

The HART element sends a HART protocol command and processes the response. The SCADAPack 100 or SCADASense Series controllers do not support this command.

The HART element is a two-cell element, with one input and two outputs. When the **enable HART** input changes from OFF to ON, the element sends the HART command to the HART device. When the **enable HART** input changes from ON to OFF, the current command is aborted and no response is processed.

When a response is received, the data in the response is stored in the I/O database and the **command complete** output turns ON. If there is an error in the command, or if the HART device fails to respond to any of the attempts, the **command error** output turns ON.

The **enable HART** input must remain ON until the response has been received or an error has occurred. All outputs turn OFF when the input is OFF.



Function Variables

The HART element has two parameters:

Variable	Valid Types	Description
command	Holding register (4xxxx)	Address of the first register in the HART command block. There are 5 registers in the block at addresses command+0 to command+4. +0 = 5904 HART interface module number +1 = HART device address +2 = command number +3 = command data register address +4 = response register address
status	Holding register (4xxxx)	Address of the first register of the status block. There are two registers in the block at addresses status+0 and status+1. +0 = command status +1 = additional information code

The **5904 HART interface module number** must be set to 0, 1, 2 or 3. This must correspond with the module number of a *CNFG 5904 HART Interface* module in the Register Assignment.

The **HART device address** must be in the range 0 to 15.

Valid values for **command number** are shown below.

The **command data register address** specifies an I/O database register where the data for the command is stored. The program must set the values in these registers. The tables below show how this address is used.

The **response register address** specifies an I/O database register where the data in the response is stored. The tables below show how this address is used. Registers marked as (float) contain floating-point values.

Command	0
Purpose	Read the device identifier
Command Data Registers	Not used
Response Registers	+0 = manufacturer ID +1 = manufacturer Device Type +2 = preambles Requested +3 = command Revision +4 = transmitter Revision +5 = software Revision +6 = hardware Revision +7 = flags +8,9 = device ID (double)
Notes	This command is also the link initialization command. The HART element performs link initialization automatically. The user does not have to send this command unless the device identifier is needed.

Command	1
Purpose	Read primary variable (PV)
Command Data Registers	Not used
Response Registers	+0,1 = PV (float) +2 = PV units code

Command	2
Purpose	Read primary variable current and percent of span
Command Data Registers	Not used
Response Registers	+0,1 = PV current (float) +2 = PV current units code +3,4 = PV percent (float) +5 = PV percent units code

Command	3
Purpose	Read dynamic variables and primary variable current
Command Data Registers	Not used
Response Registers	+0,1 = primary variable current (float) +2 = primary variable current units code +3,4 = primary variable value (float) +5 = primary variable units code +6,7 = secondary variable value (float) +8 = secondary variable units code +9,10 = tertiary variable value (float) +11 = tertiary variable units code +12,13 = fourth variable value (float) +14 = fourth variable units code
Notes	Not all devices return primary, secondary, tertiary and fourth variables. If the device does not support them, zero is written into the value and units code for that variable.

Command	33
Purpose	Read specified transmitter variables
Command Data Registers	+0 = variable code 0 +1 = variable code 1 +2 = variable code 2 +3 = variable code 3
Response Registers	+0,1 = variable 0 value (float) +2 = variable 0 units code +3,4 = variable 1 value (float) +5 = variable 1 units code +6,7 = variable 2 value (float) +8 = variable 2 units code +9,10 = variable 3 value (float) +11 = variable 3 units code

The **status** registers contains the current command status and additional code returned from the HART interface. The table below shows the values.

status	Description	Code
0	HART interface module is not communicating	not used
1	Command ready to send to device	not used
2	Command sent to device	current attempt number
3	Response received	response code from HART device (see Notes)

status	Description	Code
4	No valid response received after all attempts made	0=no response from HART device. Other = error response code from HART device (see Notes)
5	HART interface is not ready to transmit	not used

Element Configuration

This element is configured using the HART Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **E**lement **C**onfiguration command on the **E**dit menu to modify the configuration block.

WARNING: If the controller is initialized (using the TelePACE *Initialize* command), all I/O database registers used for element configuration are set to zero. The Ladder Logic program must be re-loaded to the controller to restore the element configuration.

Addresses shows the register addresses that will be used by the element.

Module number selects the HART interface module to which the command will be sent. The valid range is 0 to 3.

Device Address selects the address of the HART device. The valid range is 0 to 15.

Command selects the HART command to be sent to the device. The valid values are:

Read Device Identifier = 0

Read PV = 1

Read PV Current and % = 2

Read Dynamic Variables = 3

Read Specified Variables = 33

Data Register Address selects the registers where data for the command is stored. This field is grayed for the *Read Device Identifier*, *Read PV*, *Read PV Current and %*, and *Read Dynamic Variables* commands. These commands do not have any data.

Response Register Address selects the registers where the data in the response from the command will be stored.

Notes

The HART element performs link initialization automatically. The user does not have to send the link initialization command. If the command fails after all attempts, the link is automatically reinitialized by the HART element, at the time of the next command.

Some variables read from HART devices are 32-bit floating-point values. Each floating-point value is stored in two consecutive database registers. The most significant 16 bits of the value is stored in the lower numbered register. The least significant 16 bit of the value is stored in the higher numbered register.

The variable codes specify which variables are read from the HART device. See the documentation for your HART device for valid values for the variable codes.

The unit codes are standardized for all HART devices. See the HART specification or the documentation for your HART device for valid values.

The response code from the HART device contains communication error and status information. The information varies by device, but there are some common values.

If bit 7 of the high byte is set, the high byte contains a communication error summary. This field is bit-mapped. The following table shows the meaning of each bit as defined by the HART protocol specifications. Consult the documentation for the HART device for more information.

Bit	Description
6	vertical parity error
5	overflow error
4	framing error
3	longitudinal parity error
2	reserved – always 0
1	buffer overflow
0	Undefined

If bit 7 of the high byte is cleared, the high byte contains a command response summary. The following table shows common values. Other values may be defined for specific commands. Consult the documentation for the HART device.

Code	Description
32	Busy – the device is performing a function that cannot be interrupted by this command
64	Command not Implemented – the command is not defined for this device.

The low byte contains the field device status. This field is bit-mapped. The following table shows the meaning of each bit as defined by the HART protocol specifications. Consult the documentation for the HART device for more information.

Bit	Description
7	field device malfunction
6	Configuration changed
5	cold start
4	more status available (use command 48 to read)

Bit	Description
3	primary variable analog output fixed
2	primary variable analog output saturated
1	non-primary variable out of limits
0	primary variable out of limits

Example

An example of polling using the HART block is in the file **HART.LAD**. This file is automatically installed on the computer when the TelePACE program is installed. It is located in the **TELEPACE\EXAMPLES** directory.

See Also

MSTR - Master Message

5.33 INIM – Initialize Dial-Up Modem

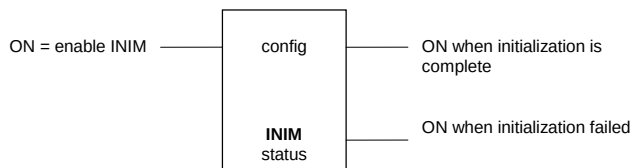
Description

The **INIM** function initializes an internal modem or an external modem, typically to set it to receive calls. Only one INIM block may be active on each serial communication port at any one time.

When **enable INIM** input is ON, the INIM element sends the modem initialization string to the modem. When the initialization is complete the **ON when initialization is complete** output is turned ON. If an error occurs, the **ON when initialization failed** output is turned ON.

The **enable INIM** input must be energized for the entire INIM time, including multiple initialization attempts.

Note: The SCADAPack 100 does not support dial up connections on com port 1.
SCADASense Series controllers do not support dial connections on any serial port.



Function Variables

Variable	Valid Types	Description
Config	holding register (4xxxx)	Address of the first register in the configuration block. There are 18 registers in the block at addresses configuration+0 to configuration+17. +0 = Communication port +1 = Length of modem string +2 = Modem initialization string Registers should be programmed using the Element Configuration command.
Status	Holding register (4xxxx)	Address of the first register in the status block. There are 2 registers in the block at addresses status+0 to status+1. +0 = error code +1 = reservation identifier

Notes

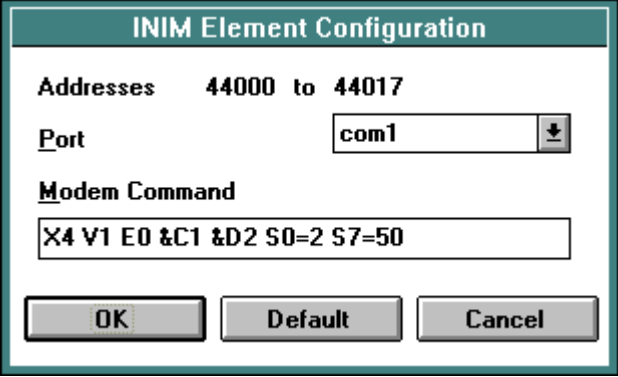
The modem initialization string is packed. Two ASCII characters are stored in each register.

The reservation identifier is required for the operation of the dialer but can be ignored by the ladder program.

Element Configuration

This element is configured using the INIM Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.

WARNING: If the controller is initialized, using the **Initialize** command in the **Controller** menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.



The dialog box is titled "INIM Element Configuration". It contains the following fields and buttons:

- Addresses:** 44000 to 44017
- Port:** A dropdown menu showing "com1" with a small arrow icon to its right.
- Modem Command:** A text box containing the string "X4 V1 E0 &C1 &D2 S0=2 S7=50".
- Buttons:** "OK", "Default", and "Cancel" at the bottom.

Error Code	Description
0	No Error
1	Bad configuration error occurs when an incorrect initialization string is sent to the modem. This usually means the modem does not understand a specific command in the initialization string.
2	No modem is connected to the controller serial port, or the controller serial port is not set to RS232 Modem.
3	Initialization error occurs when the modem does not respond to the initialization string and may be turned off.
6	Call aborted by the program. This will occur if the enable INIM input goes OFF before a modem connection occurs.
9	"Serial port is not available" error occurs when the INIM function attempts to use the serial port when another ladder communication element, C program or an incoming call has control of the port.

To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator display terminals, and for programming and diagnostics using the TelePACE program.

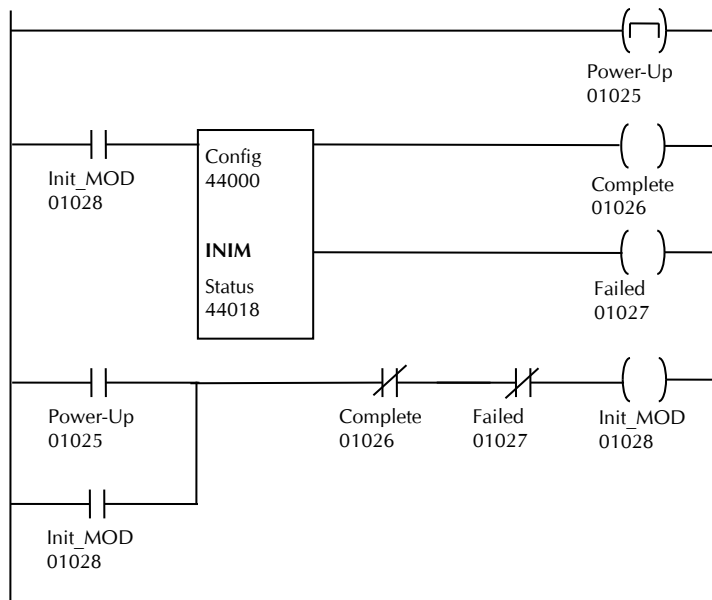
Related Functions

DIAL – Control Dial-Up Modem

Example

The following network configures a generic Hayes compatible modem to answer on the second ring. The modem is initialized when the controller is powered up.

Network 1: Initialize Modem on Power Up



5.34 L→L – List to List Transfer

Description

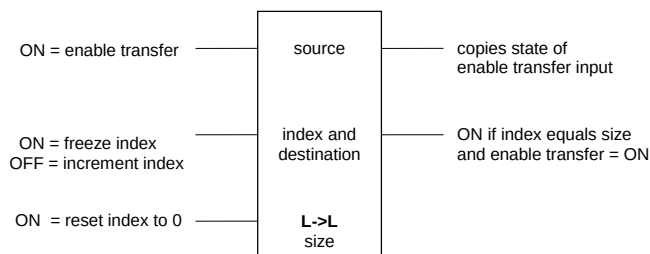
The **L→L** function transfers the contents of one register from the source list of registers into one register from the destination list of registers, at the position of the index. The index register points to the same position in both the source and the destination list of registers.

When the **enable transfer** input is ON, and the **index equals size** output is not ON, the source list register is transferred to the destination list register at the position of the index in both lists.

The **freeze/increment index** input allows control of the index. If this input is ON the index is not incremented. If the input is OFF the index is incremented by one after each transfer.

When the **reset index** input is ON the index is reset to zero.

The index is incremented by one on each transfer until the index equals size. When the **index equals size** output is ON no further increments in index value occur until the index is reset.



Function Variables

Variable	Valid Types	Description
source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The address of the source register list. The address for a coil or status register block is the first register in a group of 16 registers that will be copied.
Index and destination	holding register (4xxxx)	The address of the index register and the destination register list. The index is stored in register [4xxxx]. The data is stored in register [4xxxx+1] to register [4xxxx+size]
Size	constant (1..9999)	The number of registers in the source and destination lists.

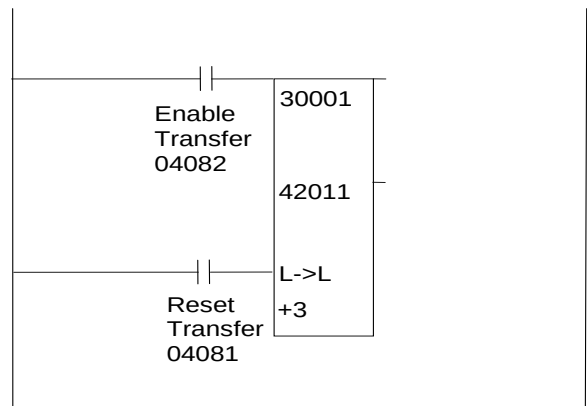
Related Functions

L→R – List to Register Transfer

R→L – Register to List Transfer

MOVE – Move Block

Example



The List to List function transfers the contents of one register in the **source** list of registers into one register of the **destination** list of registers, at the position of the index. The index register points to the same position in the source and destination lists.

	<u>Source List</u>		<u>Destination List</u>
		List Index---->	42011
list register 0-->	30001	list register 0-->	42012
>			
list register 1-->	30002	list register 1-->	42013
>			
list register 2-->	30003	list register 2-->	42014
>			
list register 3-->	30004	list register 3-->	42015
>			

In this example the **source** register list starts at input register 30001. The **index** is holding register 42011 and the **destination** register list starts at holding register 42012. The source register list contains the first four analog input registers. The index is a general purpose analog output register that will have a value of 0, 1, 2 or 3. The destination register list starts at the next sequential register after the index register. The destination list contains four general purpose analog output registers and has a size of four.

When list index register 42011 has a value of 0 and the L->L function is enabled the contents of source list register 30001 is transferred to destination list register 40012.

When list index register 42011 has a value of 1 and the L->L function is enabled the contents of source list register 30002 is transferred to destination list register 42013.

When list index register 42011 has a value of 2 and the L->L function is enabled the contents of source list register 30003 is transferred to destination list register 42014.

When list index register 42011 has a value of 3 and the L->L function is enabled the contents of source list register 30004 is transferred to destination register 42015.

5.35 L→R – List to Register Transfer

Description

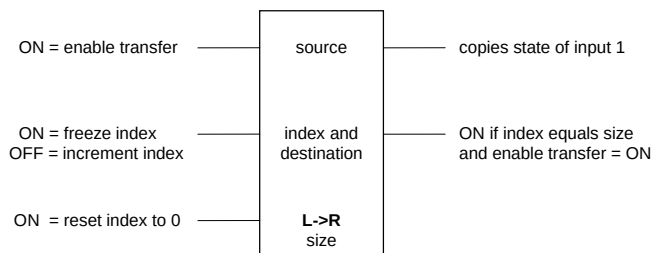
The **L→R** function transfers the contents of a register indicated by the index from the source list of registers into the destination register.

When the **enable transfer** input is ON, and the **index equals size** output is OFF, the source list register pointed to by the index is transferred to the destination register.

The **freeze/increment index** input allows control of the index. If this input is ON the index is not incremented. If the input is OFF the index is incremented by one after each transfer.

When the **reset index** input is ON the index is reset to zero.

The index is incremented by one on each transfer until the index equals size. When the **index equals size** output is ON no further increments in the index value occur until the index is reset.



Function Variables

Variable	Valid Types	Description
Source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The address of the source register list. The address for a coil or status register block is the first register in a group of 16 registers that will be read.
Index and destination	holding register (4xxxx)	The address of the index register and the destination register. The index to the source list is stored in holding register [4xxxx]. The destination register is holding register [4xxxx+1].
Size	constant (1..9999)	The number of registers in the source list.

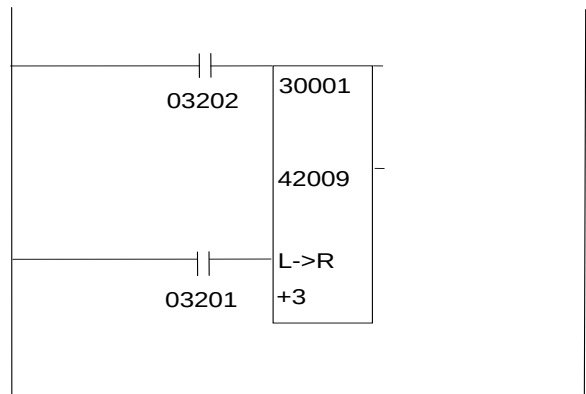
Related Functions

R→L – Register to List Transfer

L→L – List to List Transfer

MOVE – Move Block

Example



The List to Register function transfers the contents of a register indicated by the index from the source list of registers into the destination register.

Source Register List			
list register 0-->	30001	List Index---->	4200
list register 1-->	30002		9
list register 2-->	30003	Destination	4201
list register 3-->	30004	Register->	0

An example of the registers used in a List to Register (L->R) transfer is shown in the above diagram. The source register list contains the first four analog input registers and has a size of four. The index is a general purpose analog output register that will have a value of 0, 1, 2 or 3. The destination register is a general purpose analog output register that is the next sequential register after the index register.

When list index register 42009 has a value of 0 and the L->R function is enabled, the contents of source register 30001 is transferred to destination register 42010.

When list index register 42009 has a value of 1 and the L->R function is enabled, the contents of source register 30002 is transferred to destination register 42010.

When list index register 42009 has a value of 2 and the L->R function is enabled, the contents of source register 30003 is transferred to destination register 42010.

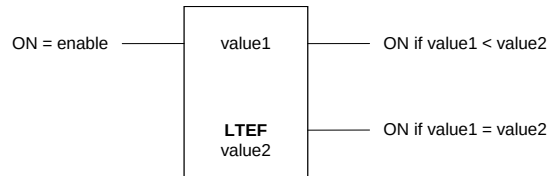
When list index register 42009 has a value of 3 and the L->R function is enabled, the contents of source register 30004 is transferred to destination register 42010.

5.36 LTEF - Floating-Point Less Than or Equal

Description

The LTEF element tests if a floating-point register or constant is less than or equal to another floating-point register or constant.

When the **enable** input is ON, value1 and value2 are compared. If value1 is less than value2, the top output is ON. If value1 is equal to value2 the bottom output is ON.



Function Variables

The element has two parameters.

Variable	Valid Types	Description
value1	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	a floating-point register or constant
Value2	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	a floating-point register or constant

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number. Floating point numbers can represent positive and negative values in the range – 3.402×10^{38} to 3.402×10^{38} .

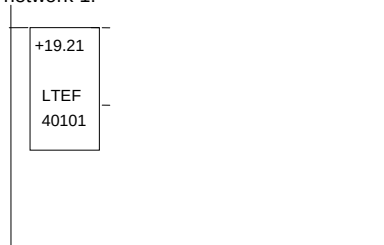
Related Functions

CMP – Compare Signed Values

CMPU – Compare Unsigned Values

Example

network 1:



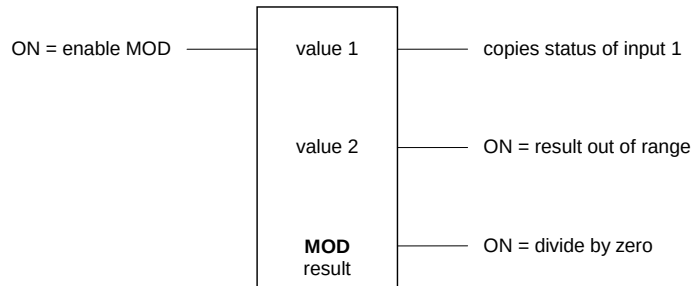
The LTEF function in network 1 compares the floating-point constant +19.21 to the contents of floating-point register 40101. The top output of the LTEF function is ON if the value of register 40101 is greater than +19.21. The bottom output of the LTEF is ON if the value of register 40001 is equal to +19.21.

5.37 MOD – Modulus of Signed Values

Description

The **MOD** function divides two registers or a constant, **value 1**, by a register or constant, **value 2**, and stores the modulus (remainder) in a holding register, **result**. Signed division is used. The value 1, value 2 and result are signed numbers.

The **out of range** output is enabled if the result is greater than 32767 or less than -32768. The **divide by zero** output is enabled if value 2 equals 0.



Function Variables

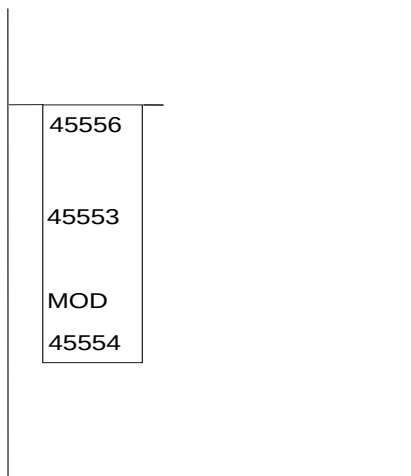
Variable	Valid Types	Description
Value 1	constant (-32768..32767) 2 Input registers (3xxxx) 2 holding registers (4xxxx)	Signed value to divide. The low order word is stored in the first register if registers are used.
Value 2	constant (-32768..32767) input register (3xxxx) holding register (4xxxx)	Signed value to divide by
Result	holding register (4xxxx)	Result = value 1 modulo value 2

Related Functions

MODU – Modulus of Unsigned Values

Example

network 1:



The MOD function in network 1 divides **value 1**, double word registers 45556 and 45557 by **value 2**, register 45553. The modulus of this division is stored in register 45554. The table below shows some examples of different values for the registers in the MOD function.

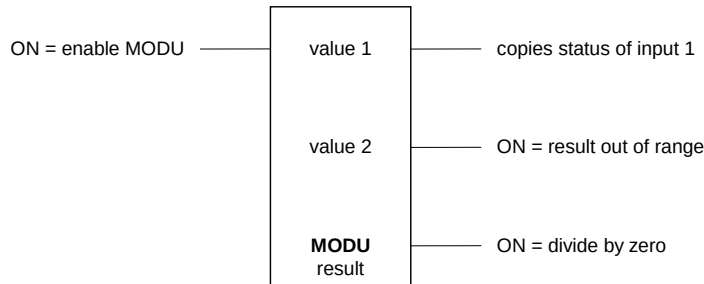
	Value of Register 45556	Value of Register 45553	Modulus Register 45554
example 1	5689	77	68
example 2	-32600	-234	-74
example 3	-22765	7659	-7447
Example 4	19823	-7692	4439

5.38 MODU – Modulus of Unsigned Values

Description

The **MODU** function divides two registers or a constant, **value 1**, by a register or constant, **value 2**, and stores the modulus (remainder) in a holding register, **result**. Unsigned division is used. The value 1, value 2 and result are unsigned numbers.

The **out of range output** is enabled if the result is greater than 65535 or less than 0. The **divide by zero output** is enabled if value 2 equals 0.



Function Variables

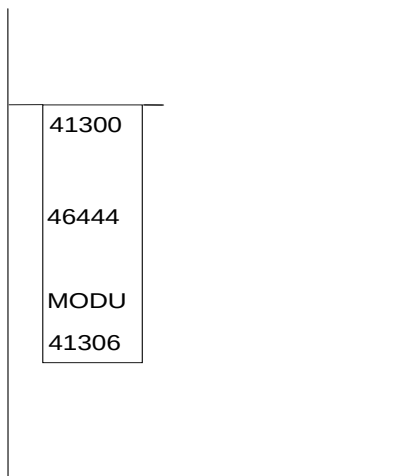
Variable	Valid Types	Description
Value 1	constant (0..65535) 2 Input registers (3xxxx) 2 holding registers (4xxxx)	Unsigned value to divide. The low order word is stored in the first register if registers are used.
Value 2	constant (0..65535) input register (3xxxx) holding register (4xxxx)	unsigned value to divide by
Result	holding register (4xxxx)	result = value 1 modulo value 2

Related Functions

MOD – Modulus of Signed Values

Example

network 1:



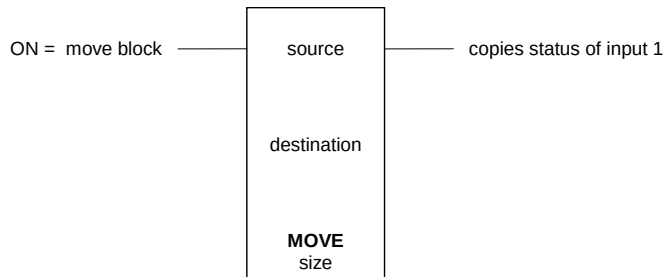
The MODU function in network 1 divides **value 1**, double word registers 41300 and 41301 by **value 2**, register 46444. The modulus of this division is stored in register 41306. The table below shows some examples of different values for the registers in the MOD function.

	Value of Register 45556	Value of Register 45553	Modulus Register 45554
Example 1	26744	320	184
Example 2	56398	34022	20878
Example 3	37697	2366	2207
Example 4	33005	1100	5

5.39 MOVE – Move Block

Description

The **MOVE** function block copies the **source** block of registers into the **destination** block of coil, or holding, registers.



Function Variables

Variable	Valid Types	Description
Source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The first register in the source block. The address for a coil or status register block is the first register in a group of 16 registers that will be copied.
Destination	coil block (0xxxx) holding register (4xxxx)	The first register in the destination block. The address for a coil register block is the first register in a group of 16 registers into which data will be copied.
Size	constant (1..100)	The number of 16 bit words in the block.

Block move copies 16 bit words. Coil and status register blocks are groups of 16 registers that start with the register specified as the block address. A block size of 2 corresponds to 32 coils, or two holding registers.

Coil and status register blocks must begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033, etc.

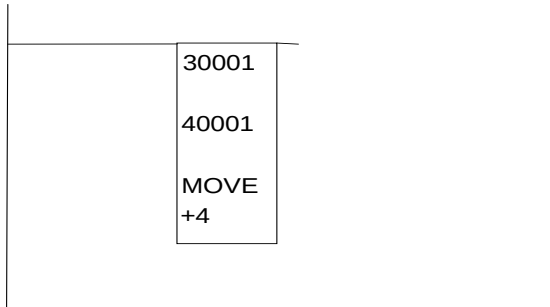
Related Functions

L→L – List to List Transfer

L→R – List to Register Transfer

Example

network 1:



In the above figure the source register entered is 30001 (3reg) and the destination register entered is 40001 (4reg) and the block size entered is 4.

When this MOVE block is enabled:

the contents of register 30001 are copied to register 40001

the contents of register 30002 are copied to register 40002

the contents of register 30003 are copied to register 40003

the contents of register 30004 are copied to register 40004

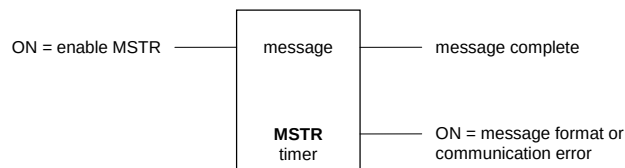
5.40 MSTR – Master Message

Description

The **MSTR** function exchanges data with another controller using either the Modbus or the DF1 communication protocols.

Note Only one **MSTR** block may be active on each serial communication port at any one time.

When the **enable MSTR** goes from OFF to ON the MSTR function will send one master message to the slave address. The **enable MSTR** must go from OFF to ON to send another MSTR message. The **message complete** output will be energized if the MSTR receives a valid response from the slave station. The **message format or communication** error is energized if an error occurs. See the status code table below for a list of possible errors.



Function Variables

Variable	Valid Types	Description
Message	holding register (4xxxx)	Address of the first register in the message control block. There are 7 registers in the block at addresses message+0 to message+6. +0 = port +1 = function code +2 = slave controller address +3 = slave register address +4 = master register address +5 = length +6 = time-out in 0.1s increments
Timer	holding register (4xxxx)	timer accumulator register

Notes

To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator display terminals, and for programming and diagnostics using the TelePACE program.

WARNING: If the controller is initialized, using the **Initialize** command in the **Controller** menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.

Using MSTR with Modbus Protocol

This element is configured using the MSTR Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.

MSTR Element Configuration

Addresses 42000 to 42006

Port

Protocol Modbus RTU

Function Code

Slave RTU Address

Slave Register Address

Master Register Address

Length

Time Out tenths of seconds

OK Cancel

Valid **Function Codes** are shown below. Functions 05, 06, 15, and 16 support broadcast messages. Functions 129, 130, 132, 133, 135, 136, 138, and 139 may be broadcast, but some Enron Modbus slave devices may not support broadcast messages. See the *TeleBUS Protocols Modbus Compatible Protocols User Manual* for details on each function code.

Function	Description	Purpose	Maximum Registers
01	Read Coil Status	Read digital output registers.	2000
02	Read Input Status	Read digital input registers.	2000
03	Read Holding Register	Read analog output registers.	125
04	Read Input Register	Read analog input registers.	125
05	Write Single Coil	Write digital output register.	1
06	Write Single Register	Write analog output registers.	1
15	Write Multiple Coils	Write digital output registers.	880
16	Write Multiple Registers	Write analog output registers.	60
128	Read Enron Boolean	Read Enron Boolean registers	2000
129	Write Enron Single Boolean	Write Enron Boolean register	1
130	Write Enron Multiple Booleans	Write Enron Boolean registers	880
131	Read Enron Short Integers	Read Enron short integer register	125

Function	Description	Purpose	Maximum Registers
132	Write Enron Single Short Integer	Write Enron short integer register	1
133	Write Enron Multiple Short Integers	Write Enron short integer registers	60
134	Read Enron Long Integers	Read Enron long integer register	62
135	Write Enron Single Long Integer	Write Enron long integer register	1
136	Write Enron Multiple Long Integers	Write Enron long integer registers	30
137	Read Enron Floats	Read Enron floating-point register	62
138	Write Enron Single Float	Write Enron floating-point register	1
139	Write Enron Multiple Floats	Write Enron floating-point registers	30

The **port** must be set to 1, 2, 3 or 4 corresponding to the com1, com2, com3 and com4 serial ports.

The **slave controller address** edit-box sets the station number of the controller. For the Modbus ASCII and Modbus RTU protocols, the valid range is 0 to 255 if standard addressing is used, and 0 to 65534 if extended addressing is used. Address 0 is reserved for broadcast messages.

The **slave register address** specifies the first register where data will be read from or written to in the slave controller. The following table shows the valid slave register addresses for each command.

Function Code	Slave Register Address	Length
Read Coil Status	00001 to 09999	2000
Read Input Status	10001 to 19999	2000
Read Holding Register	40001 to 49999	125
Read Input Register	30001 to 39999	125
Write Single Coil	00001 to 09999	1
Write Single Register	40001 to 49999	1
Write Multiple Coils	00001 to 09999	880
Write Multiple Registers	40001 to 49999	60
Read Enron Boolean	0 to 65535	2000
Write Enron Boolean	0 to 65535	1
Write Enron Multiple Boolean	0 to 65535	880
Read Enron Short Integer	0 to 65535	125
Write Enron Short Integer	0 to 65535	1
Write Enron Multiple Short Integer	0 to 65535	60
Read Enron Long Integer	0 to 65535	62
Write Enron Long Integer	0 to 65535	1
Write Enron Multiple Long Integer	0 to 65535	30
Read Enron Floating Point	0 to 65535	62
Write Enron Floating Point	0 to 65535	1
Write Enron Multiple Floating Point	0 to 65535	30

The **master register address** specifies the first register where data will be written to or read from in this controller. The register type does not have to match the type of the slave register address. It is possible to store input registers from a slave into output registers on the master and vice-versa.

The **length** parameter specifies how many registers are to be transferred. The maximum length for each function code is shown in the above table. Functions 05 and 06 always transfer 1 register, regardless of the value of **length**.

The **time-out** is in 0.1s increments. The error output is energized if the time-out period ends without a valid response to the message.

When an error occurs, an error status code is stored in the master command status register in the analog input database. There is one status register for each port.

The module **DIAG Serial port protocol status** must be added to the register assignment to monitor the master command status register for the desired serial port. The status codes are shown in the following table.

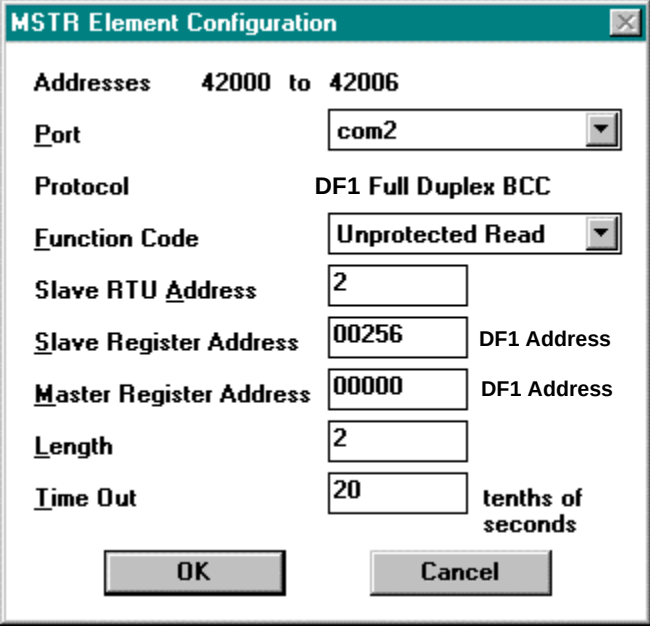
Status Code	Description
0	Message sent - waiting for response
1	Response received (no error occurred)
3	bad value in function code register
4	bad value in slave controller address register
5	bad value in slave register address register
6	bad value in length register
7	serial port or protocol is invalid
12	response timeout
24	exception response: invalid function code
25	exception response: invalid address
26	exception response: invalid value
27	protocol is invalid or serial port queue is full
28	slave and master stations are equal; they must be different
29	exception response: slave device failure
30	exception response: slave device busy

Example

An example of polling using the MSTR block is in the file **MSTR.LAD**. This file is automatically installed on the computer when the TelePACE program is installed. It is located in the **TELEPACE\EXAMPLES** directory.

Using MSTR with DF1 Protocol

This element is configured using the MSTR Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.



The image shows a dialog box titled "MSTR Element Configuration". It contains the following fields and values:

- Addresses:** 42000 to 42006
- Port:** com2 (selected from a dropdown)
- Protocol:** DF1 Full Duplex BCC
- Function Code:** Unprotected Read (selected from a dropdown)
- Slave RTU Address:** 2
- Slave Register Address:** 00256 (labeled as DF1 Address)
- Master Register Address:** 00000 (labeled as DF1 Address)
- Length:** 2
- Time Out:** 20 (labeled as tenths of seconds)

At the bottom are "OK" and "Cancel" buttons.

The **Port** must be set to 1, 2, 3 or 4 corresponding to the com1, com2, com3 and com4 serial ports.

Valid **Function Codes** are:

Function	Description	Purpose	Maximum Registers
00	Protected Write	Writes words of data to limited areas of the database.	121
01	Unprotected Read	Reads words of data from any area of the database.	122
02	Protected Bit Write	Sets or resets individual bits within limited areas of the database.	1
05	Unprotected Bit Write	Sets or resets individual bits in any area of the database.	1
08	Unprotected Write	Writes words of data to any area of the database.	121

The **Slave RTU Address** must be in the range 0 to 254.

The **Slave Register Address** specifies the first DF1 physical 16-bit address where data will be read from or written to in the slave controller.

The **Master Register Address** specifies the first physical 16-bit address where data will be written to or read from in this controller. The register type does not have to match the type of the slave register address. It is possible to store input registers from a slave into output registers on the master and vice-versa.

The **Length** parameter specifies how many 16-bit registers are to be transferred for functions 00, 01 and 08. For function codes 02 and 05, this field is labeled **Bit Mask**. **Bit Mask** selects the 16-bit bitmask specifying which bits in the **Master Register** to send. If a bit is set in **Bit Mask**, the corresponding bit in the register will be sent.

The **Time Out** is in 0.1s increments. The error output is energized if the time-out period ends without a valid response to the message. If the MSTR function is used to send a broadcast message,

the **Slave RTU Address** must be 255 and the **Time Out** can be set to 0 since there is no response to wait for.

The module **DIAG Serial port protocol status** must be added to the register assignment to monitor the master command status register for the desired serial port. The status codes are shown in the following table.

Status Code	Description
0	Message sent - waiting for response
1	Response received (no error occurred)
3	bad value in function code register
4	bad value in slave controller address register
5	bad value in slave register address register
6	bad value in length register
7	slave has no more responses to send.
8	Response received didn't match command sent.
9	Specified protocol is not supported by this controller.
16	slave error - illegal command or format.
80	slave error - addressing problem or memory protect rungs.
Other	error codes from other DF1 compatible controllers - Consult the documentation for specific controller.

Notes

Details about the DF1 driver implementation on the SCADAPack controllers can be found in the **TeleBUS DF1 Protocol** Booklet in this manual.

An example of polling using the MSTR block is in the file **ab_mstr.lad**. This file is automatically installed on the computer when the TelePACE program is installed. It is located in the **TELEPACE\EXAMPLES** directory.

5.41 MSIP – Master IP Message

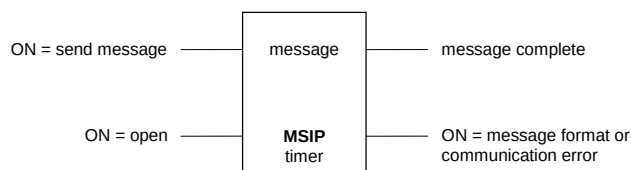
Description

The **MSIP** function exchanges data with another controller using the Modbus IP communication protocol. This function sends a master message to a remote IP address.

Note Unlike the **MSTR** function, multiple **MSIP** blocks may be active on the same communication interface at the same time. The number of active **MSIP** blocks is limited only by the number of available connections (typical capacity is 20 connections).

When the **open** input goes from OFF to ON the MSIP function opens an IP connection. Set the **open** input to OFF to close the connection.

When the **send message** input goes from OFF to ON, and the **open** input is ON, the MSIP function will send one Modbus IP master message to the slave address. The **send message** input must go from OFF to ON to send another MTCP message. Each master message is complete when either output is energized: The **message complete** output will be energized if the MSIP block receives a valid response from the slave station. The **message format or communication** error is energized if an error occurs. See the status code table below for a list of possible errors.



Function Variables

Variable	Valid Types	Description
message	holding register (4xxxx)	Address of the first register in the message control block. There are 11 registers in the block at addresses message+0 to message+11. +0 = byte1 of slave IP address MSB, using format byte1.byte2.byte3.byte4 +1 = byte2 of slave IP address +2 = byte3 of slave IP address +3 = byte4 of slave IP address LSB +4 = protocol type +5 = function code +6 = slave controller address +7 = slave register address +8 = master register address +9 = length +10 = time-out in 0.1s increments

Variable	Valid Types	Description
timer	holding register (4xxxx)	Address of the first register in the timer control block. There are 5 registers in the block at addresses timer+0 and timer+4. +0 = timer accumulator register +1 = command status register +2 = internal: connection ID +3 = internal: closing state +4 = internal: initialized

Sending a Message

Set the **open** input to ON to open an IP connection. This input may be set to ON at the same time as the **send message** input in the next step.

If the maximum number of opened connections has been reached, an error occurs: The **error** output is energized and the value of the **command status register** indicates the maximum number of connections has been reached. Close the connection used by another MSIP function to correct the error and repeat step 1.

Send one command message by toggling the **send message** input from OFF to ON.

The value of the **command status register** is 0 indicating that the message was sent.

Sending is complete when either the **message complete** output or **error** output is energized. The **message complete** output is energized if the response is received successfully. The **error** output is energized if there was a command error, timeout, or connection failure. The value of the **command status register** is 1 for success or another value for error conditions.

Repeat steps 2 to 4 to send additional command messages.

Notes

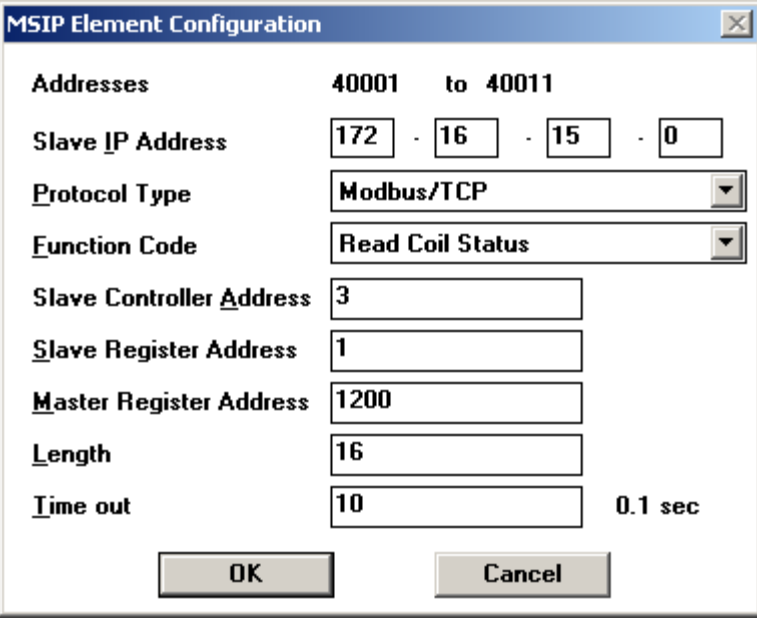
The **time-out** register determines how long to wait for a response before closing the connection and reporting an error. The connection is re-connected automatically when the next message is sent. The **open** input does not need to be toggled to do this.

Leave the **open** input ON to keep the connection open for this MTCP function.

WARNING: If the controller is initialized, using the **Initialize** command in the **Controller** menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.

Element Configuration

This element is configured using the MSIP Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.



The image shows a Windows-style dialog box titled "MSIP Element Configuration". It contains several fields for configuring a Modbus slave. The "Addresses" field shows a range from 40001 to 40011. The "Slave IP Address" is entered as 172.16.15.0. The "Protocol Type" is set to "Modbus/TCP" and the "Function Code" is "Read Coil Status". Other fields include "Slave Controller Address" (3), "Slave Register Address" (1), "Master Register Address" (1200), "Length" (16), and "Time out" (10 seconds). At the bottom are "OK" and "Cancel" buttons.

Addresses	40001 to 40011
Slave IP Address	172 - 16 - 15 - 0
Protocol Type	Modbus/TCP
Function Code	Read Coil Status
Slave Controller Address	3
Slave Register Address	1
Master Register Address	1200
Length	16
Time out	10 0.1 sec

The **slave IP address** has the format 255.255.255.255 where one register is used for each byte: byte1.byte2.byte3.byte4

The **protocol type** must be one of the values shown below:

Protocol Type	Description
1	Modbus/TCP
2	Modbus RTU over UDP
3	Modbus ASCII over UDP

Valid **Function Codes** are shown below. Functions 05, 06, 15, and 16 support broadcast messages. Functions 129, 130, 132, 133, 135, 136, 138, and 139 may be broadcast, but some Enron Modbus slave devices may not support broadcast messages. See the *TeleBUS Protocols Modbus Compatible Protocols User Manual* for details on each function code.

Function	Description	Purpose	Maximum Registers
01	Read Coil Status	Read digital output registers.	2000
02	Read Input Status	Read digital input registers.	2000
03	Read Holding Register	Read analog output registers.	125 ^{note 1}
04	Read Input Register	Read analog input registers.	125 ^{note 1}
05	Write Single Coil	Write digital output register.	1
06	Write Single Register	Write analog output registers.	1
15	Write Multiple Coils	Write digital output registers.	880
16	Write Multiple Registers	Write analog output registers.	60

Function	Description	Purpose	Maximum Registers
128	Read Enron Booleans	Read Enron Boolean registers	2000
129	Write Enron Single Boolean	Write Enron Boolean register	1
130	Write Enron Multiple Booleans	Write Enron Boolean registers	880
131	Read Enron Short Integers	Read Enron short integer register	125 ^{note 1}
132	Write Enron Single Short Integer	Write Enron short integer register	1
133	Write Enron Multiple Short Integers	Write Enron short integer registers	60
134	Read Enron Long Integers	Read Enron long integer register	62 ^{note 2}
135	Write Enron Single Long Integer	Write Enron long integer register	1
136	Write Enron Multiple Long Integers	Write Enron long integer registers	30
137	Read Enron Floats	Read Enron floating-point register	62
138	Write Enron Single Float	Write Enron floating-point register	1
139	Write Enron Multiple Floats	Write Enron floating-point registers	30

Notes

For SCADASense Series controllers, the MSIP function can read a maximum of 123 registers with a single MSIP function. Note that SCADASense Series controllers do not support Enron Modbus commands.

The slave controller address edit-box sets the station number of the controller. The valid range is 0 to 255 if standard addressing is used, and 0 to 65534 if extended addressing is used. Address 0 is reserved for broadcast messages, which are directed to the IP address specified in the slave IP address field only.

The slave register address specifies the first register where data will be read from or written to in the slave controller. The following table shows the valid slave register addresses for each command.

Function Code	Slave Register Address	Length
Read Coil Status	00001 to 09999	2000
Read Input Status	10001 to 19999	2000
Read Holding Register	40001 to 49999	125
Read Input Register	30001 to 39999	125
Write Single Coil	00001 to 09999	1
Write Single Register	40001 to 49999	1
Write Multiple Coils	00001 to 09999	880
Write Multiple Registers	40001 to 49999	60
Read Enron Boolean	0 to 65535	2000

Function Code	Slave Register Address	Length
Write Enron Boolean	0 to 65535	1
Write Enron Multiple Boolean	0 to 65535	880
Read Enron Short Integer	0 to 65535	125
Write Enron Short Integer	0 to 65535	1
Write Enron Multiple Short Integer	0 to 65535	60
Read Enron Long Integer	0 to 65535	62
Write Enron Long Integer	0 to 65535	1
Write Enron Multiple Long Integer	0 to 65535	30
Read Enron Floating Point	0 to 65535	62
Write Enron Floating Point	0 to 65535	1
Write Enron Multiple Floating Point	0 to 65535	30

The **master register address** specifies the first register where data will be written to or read from in this controller. The register type does not have to match the type of the slave register address. It is possible to store input registers from a slave into output registers on the master and vice-versa.

The **length** parameter specifies how many registers are to be transferred. The maximum length for each function code is shown in the above table. Functions 05 and 06 always transfer 1 register, regardless of the value of **length**.

The **time-out** is in 0.1s increments. The error output is energized if the time-out period ends without a valid response to the message.

When an error occurs, an error status code is stored in the **command status register** in the timer control block. The status codes are shown in the following table.

Status Code	Description
0	valid command has been sent
1	response was received
2	no message was sent
3	invalid function code
4	invalid slave station address
5	invalid database address
6	invalid message length
7	serial port or protocol is invalid
8	connecting to slave IP address
9	connected to slave IP address
10	timeout while connecting to slave IP address
11	TCP/IP error has occurred while sending message
12	timeout has occurred waiting for response
13	slave has closed connection; incorrect response; or, incorrect response length
14	disconnecting from slave IP address is in progress
15	connection to slave IP address is disconnected
16	invalid connection ID
17	invalid protocol type
18	invalid slave IP address
19	last message is still being processed

Status Code	Description
20	master connection has been released
21	error while connecting to slave IP address
22	no more connections are available
24	exception response: invalid function code
25	exception response: invalid address
26	exception response: invalid value
27	protocol is invalid or serial port queue is full
28	slave and master stations are equal; they must be different
29	exception response: slave device failure
30	exception response: slave device busy

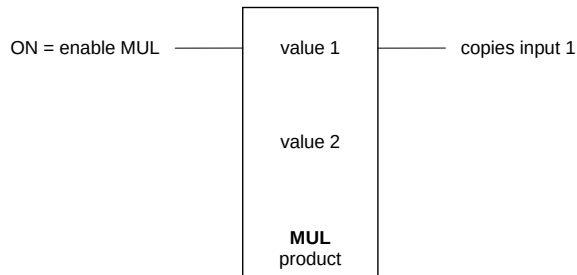
Example

An example of polling using the MSIP block is in the file **MTCP.LAD**. This file is automatically installed on the computer when the TelePACE program is installed. It is located in the **TELEPACE\EXAMPLES** directory.

5.42 MUL – Multiply Signed Values

Description

The **MUL** function block multiplies a signed register or constant, **value 1**, by a signed register or constant, **value 2** and stores the result in two consecutive holding registers, **product**. Signed multiplication is used.



Function Variables

Variable	Valid Types	Description
Value 1	Constant (-32768..32767) input register (3xxxx) holding register (4xxxx)	first signed value to multiply
Value 2	Constant (-32768..32767) input register (3xxxx) holding register (4xxxx)	Second signed value to multiply
Product	2 holding registers (4xxxx)	product = value 1 x value 2 The result is stored in two consecutive holding registers. The low order word is stored in the first register.

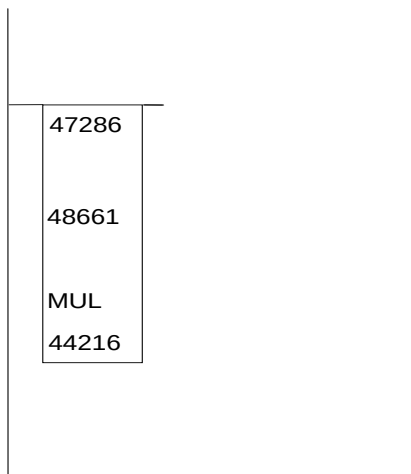
Related Functions

MULF - Multiply Floating-Point Values

MULU – Multiply Unsigned Values

Example

network 1:



In network 1 a **MUL** function is used to multiply the contents of Holding Register 47286, **value 1**, with the contents of Holding Register 48661, **value 2**. The product is stored in Holding Registers 44216, **low order word** and Holding Register 44217, **high order word**.

The examples in the table below show the product of the multiplication for various values in registers 47286 and 48661. The product is displayed as registers 44216, **low order word** and register 44217, **high order word**.

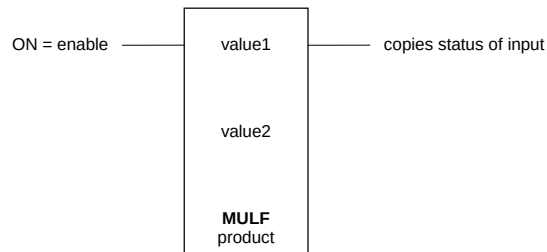
			high order	low order
	Register 47286	Register 48661	Register 44217	Register 44216
	Value 1	value 2	Product	product
example 1	-3057	-10	0	30570
example 2	-15	100	-1	-1500
example 3	1654	17	0	28118
example 4	164	-55	-1	-9020

5.43 MULF - Multiply Floating-Point Values

Description

The MULF element multiplies two floating-point registers or constants and stores the result in a floating-point holding register.

When the **enable** input is ON the product **value 1** × **value 2** is stored in the **product** floating-point register. The element output is ON when the input is.



Function Variables

The element has three parameters.

Variable	Valid Types	Description
Value1	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	first floating-point register or constant to multiply
Value2	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	second floating-point register or constant to multiply
Product	2 holding registers (4xxxx)	floating-point product = value1 × value2

Notes

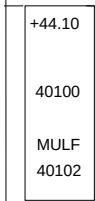
Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number. Floating point numbers can represent positive or negative values in the range – 3.402×10^{38} to 3.402×10^{38} .

Related Functions

MULU – Multiply Unsigned Values

Example

network 1:

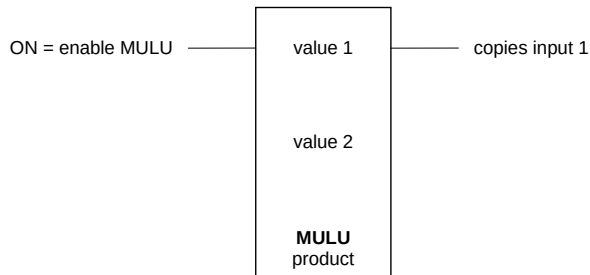


The MULF function in network 1 multiplies floating-point constant +44.10 by the contents of floating-point register 40100 (registers 40100 and 40101). Assuming floating-point register 40100 contains a value of +3.5 then the result is +154.35. This value is stored in floating-point register 40102 (registers 40102 and 40103).

5.44 MULU – Multiply Unsigned Values

Description

The **MULU** function block multiplies an unsigned register or constant, **value 1**, and an unsigned register or constant, **value 2**. The result is stored in two consecutive Holding Registers, **product**. Unsigned multiplication is used.



Function Variables

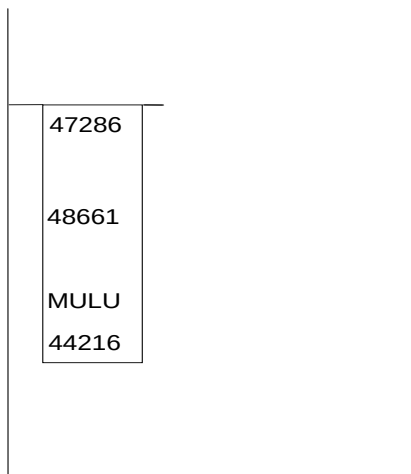
Variable	Valid Types	Description
value 1	Constant (0..65535) input register (3xxxx) holding register (4xxxx)	first unsigned value to multiply
value 2	Constant (0..65535) input register (3xxxx) holding register (4xxxx)	Second unsigned value to multiply
Product	2 holding registers (4xxxx)	Product = value 1 x value 2 The result is stored in two consecutive holding registers. The low order word is stored in the first register.

Related Functions

MULF - Multiply Floating-Point Values

Example

network 1:



In network 1 a **MULU** function is used to multiply the contents of Holding Register 47286, **value 1**, with the contents of Holding Register 48661, **value 2**. The product is stored in Holding Registers 44216, **low order word** and Holding Register 44217, **high order word**.

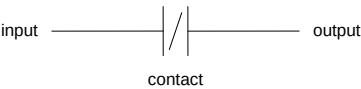
The examples in the table below show the product of the multiplication for various values in registers 47286 and 48661. The product is displayed as registers 44216, **low order word** and register 44217, **high order word**.

			high order	low order
	Register 47286	Register 48661	Register 44217	Register 44216
	Value 1	value 2	Product	product
Example 1	2048	30	0	61440
Example 2	251	251	0	63001
Example 3	77	77	0	5929
Example 4	650	735	7	18998

5.45 Normally Closed Contact

Description

The **NC** contact function block conducts power when the coil or status register is OFF.



Function Variables

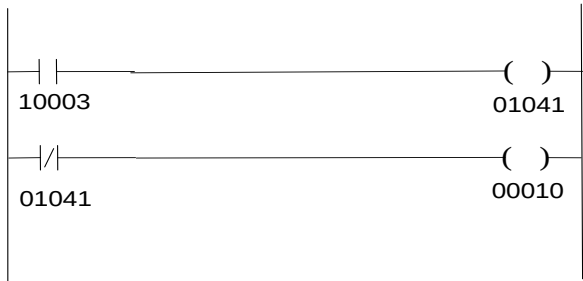
Variable	Valid Types	Description
contact	coil register (0xxxx) status register (1xxxx)	Output is ON if contact is OFF and input is ON

Related Functions

Normally Open Contact

Example

network 1:

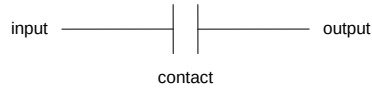


Network 1 shows NO contact 10003 and **NC** contact 01041. Power conducts through **NC** contact 01041 keeping coil 00010 ON. When **NO** contact 10003 is closed coil 01041 turns ON and **NC** contact 01041 will open, turning OFF coil 00010.

5.46 Normally Open Contact

Description

The **NO** contact function block conducts power when the coil or status register is ON.



Function Variables

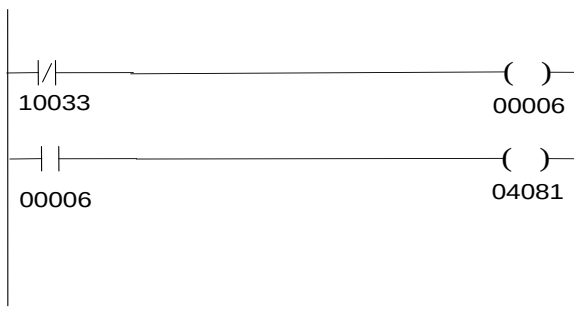
Variable	Valid Types	Description
contact	coil register (0xxxx) status register (1xxxx)	output is ON if contact is ON and input is ON

Related Functions

Normally Closed Contact

Example

network 1:

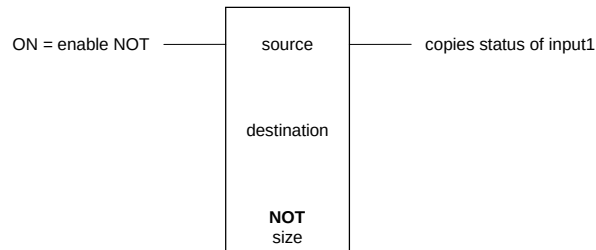


Network 1 shows NC contact 10033 and **NO** contact 00006. Power conducts through NC contact 10033 keeping coil 00006 ON. With coil 00006 ON, **NO** contact 00006 is closed and coil 04081 is ON.

5.47 NOT – Not Block

Description

The **NOT** function block inverts the **source** block of registers and stores the result in the **destination** block of coil or Holding registers. **NOT** changes bits that are ON to OFF, and changes bits that are OFF to ON.



Function Variables

Variable	Valid Types	Description
Source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The first register in the source block. The address for a coil or status register block is the first register in a group of 16 registers that will be complemented.
Destination	coil block (0xxxx) holding register (4xxxx)	The first register in the destination block. The address for a coil register block is the first register in a group of 16 registers where the result will be stored.
Size	Constant (1..100)	The number of 16 bit words in the block.

Notes

NOT accesses 16 bit words. Coil and status register blocks are groups of 16 registers that start with the register specified as the block address. A block size of 2 corresponds to 32 coils, or two holding registers.

Coil and status register blocks must begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033, etc.

Related Functions

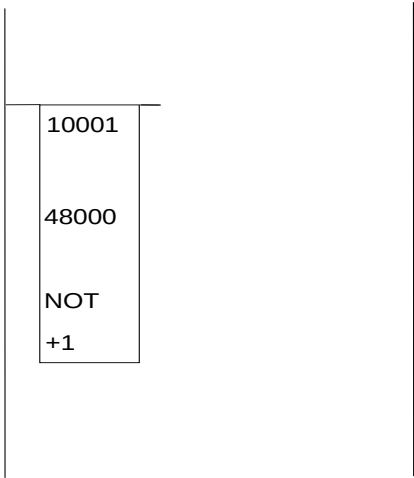
AND – And Block

OR – Or Block

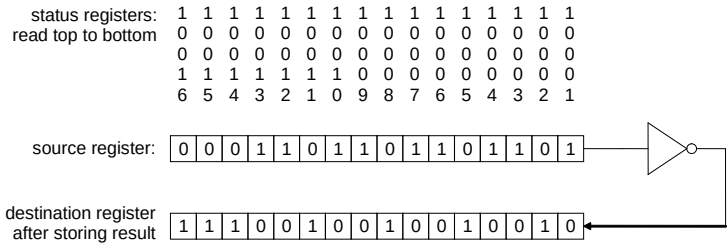
XOR – Exclusive Or Block

Example

network 1:



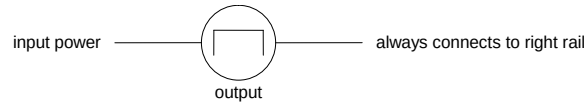
In this example the status of discrete inputs 10001 through 10016 are used as the **source** for the NOT function. These inputs are inverted and stored in the **destination** register 48000.



5.48 One Shot Coil

Description

The **One Shot Coil** function block enables an output coil for one scan when the input power changes from OFF to ON.



Function Variables

Variable	Valid Types	Description
Output	coil register (0xxxx)	output register address

Related Functions

Coil

Example

network 1:



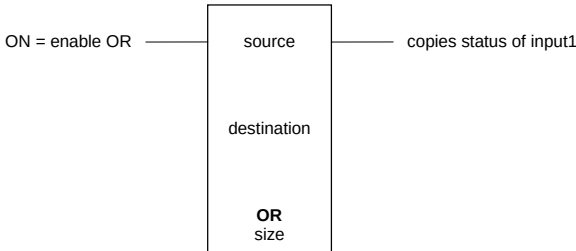
One shot coil 00004 is energized for one scan on power up of the ladder logic program.

Refer to the *I/O Database Registers* section of this manual for more information on coil outputs.

5.49 OR – Or Block

Description

The **OR** function block logically ORs the **source** block of registers with the **destination** block of registers and stores the result in the **destination** block of registers.



Function Variables

Variable	Valid Types	Description
Source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The first register in the source block. The address for a coil or status register block is the first register in a group of 16 registers that will be ORed.
Destination	coil block (0xxxx) holding register (4xxxx)	The first register in the destination block. The address for a coil register block is the first register in a group of 16 registers that will be ORed.
Size	Constant (1..100)	The number of 16 bit words in the block.

Notes

OR accesses 16 bit words. Coil and status register blocks are groups of 16 registers that start with the register specified as the block address. A block size of 2 corresponds to 32 coils, or two holding registers.

Coil and status register blocks must begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033, etc.

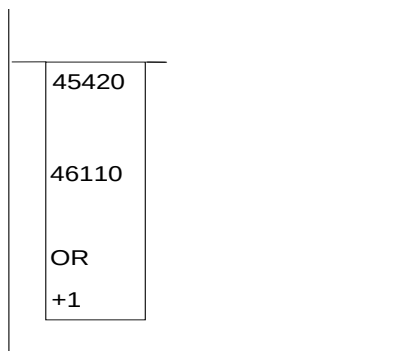
Related Functions

AND – And Block

NOT – Not Block

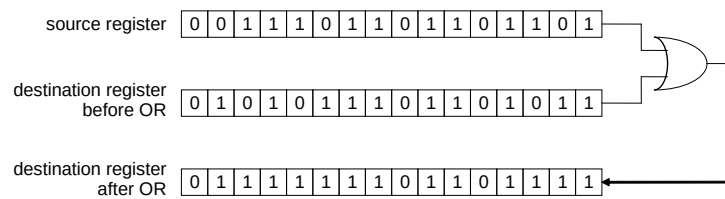
XOR – Exclusive Or Block

Example



In this example the contents of source register, 45420, are ORed with the destination register, 46110. The result is stored in the destination register. The content of register 45420 is 15213₁₀ (11101101101101). The content of the destination register before the OR function is 22379₁₀

(101011101101011). The content of the destination register after the OR function is 32623_{10}



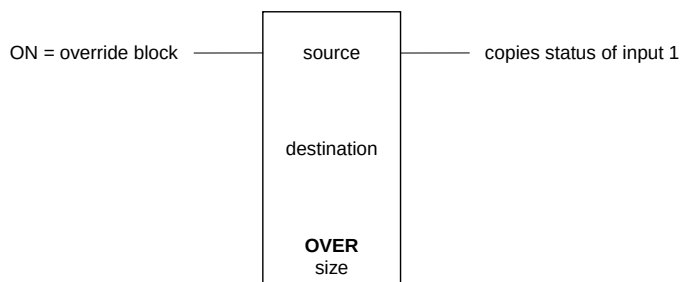
(11111101101111).

5.50 OVER – Override Block of Registers

Description

The **OVER** function copies, in one scan, the **source** block of registers into the destination block of registers and forces all registers in the **destination** block. When the **override block** input transitions from OFF to ON the destination registers are forced. The **destination** register values are overwritten with the **source** register values while the **override block** input is ON. Destination registers are unforced when the override block input transitions from ON to OFF. The function block output copies the input state.

The FORCE led on the controller is turned on when the OVER function is enabled. This function is useful for simulating inputs while debugging a control program.



Function Variables

Variable	Valid Types	Description
Source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The first register in the source block. The address for a coil or status register block is the first register in a group of 16 registers that will be copied.
Destination	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The first register in the destination block. The address for a coil or status register block is the first register in a group of 16 registers.
Size	constant (1 to 100)	The number of 16 bit words in the block.

Notes

The OVER function copies 16-bit words. Coil and status register blocks are groups of 16 registers that start with the register specified as the block address. A block size of 2 corresponds to 32 coils, or two holding registers.

Coil and status register blocks must begin at the start of a 16-bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033, etc.

An ON to OFF transition of the input must occur to remove the forcing on the registers. If you place an OVER function in a subroutine, be sure to execute the subroutine at least once with the input OFF to clear forcing (see the example below).

If forcing is cleared with a TelePACE command while the OVER function is enabled, the forcing will not be set again by the OVER function. You must disable and re-enable the function to force the registers. The function will continue to attempt to write to destination registers even though they are not forced.

Related Functions

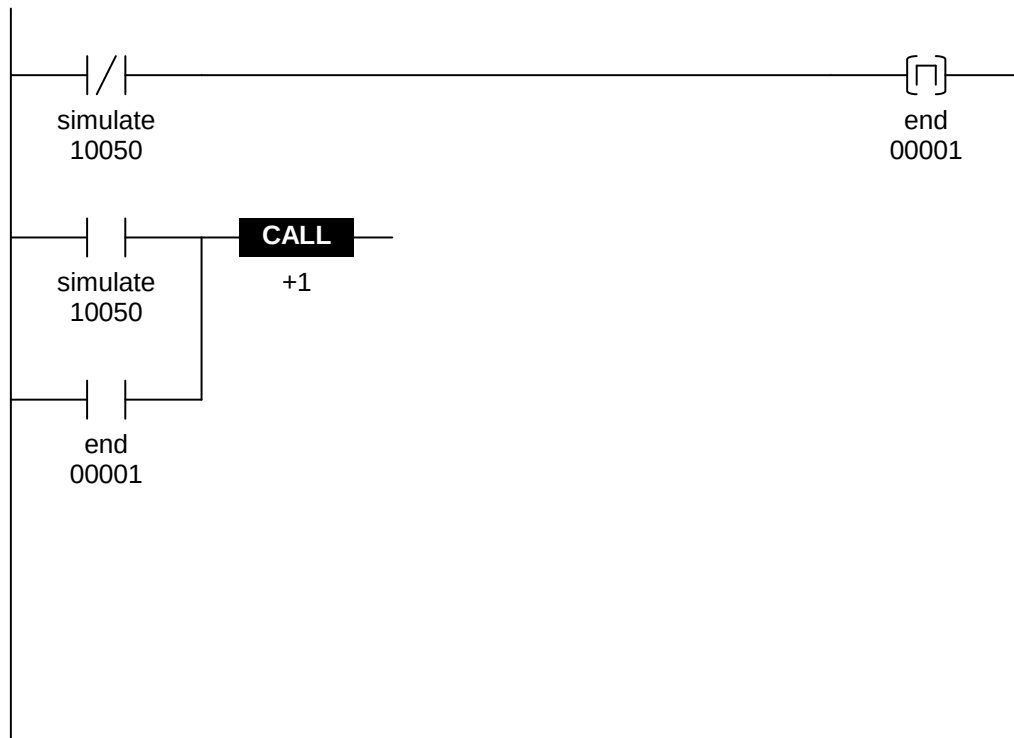
MOVE – Move Block

Example

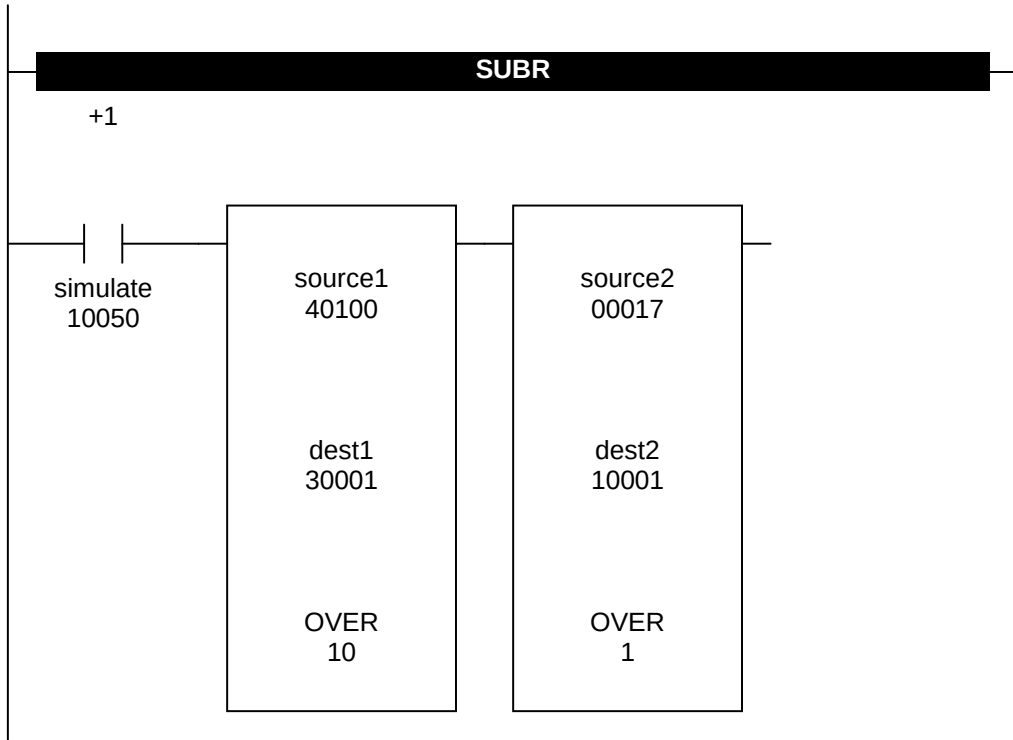
This example shows how simulated values can be written over input values in the I/O database. For this example, assume there is a register assignment that connects input hardware to registers 30001 to 30011, and to registers 10001 to 10016.

The simulate contact (10050) enables and disables the simulation. When the contact is ON simulated values are used. To simulate the values of the inputs, the program overrides the values in registers 30001 to 30011, and registers 10001 to 10016.

The first network calls the subroutine when the simulate contact is ON, and one additional time when the contact goes from ON to OFF.



The second network is a subroutine that overrides the destination registers while the simulate contact is closed. A subroutine is used so that the logic is not scanned when the simulated values are not needed. In most programs this is the normal operating state, so scanning the functions would add unnecessarily to the execution time of the program.



5.51 PID Controller

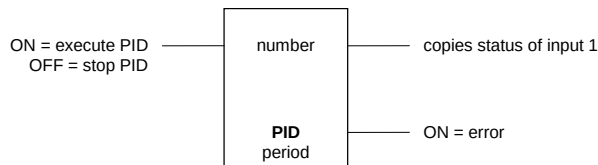
Description

The **PID** function controls the execution of a PID controller. PID controllers execute independently of the ladder logic scan. A PID controller is configured by writing data into the PID controller registers defined in the register assignment.

The PID block starts execution of the PID controller number on the rising edge of the **execute PID** input. Execution of the PID controller is stopped when the **execute PID** is OFF.

The **execute PID** input must be toggled from OFF to ON under program control to start PID execution. The **execute PID** input cannot be directly connected to the power rail or PID execution will not restart when the program restarts.

The **error** output is enabled if the PID number is not found in the Register Assignment. The error output is also enabled if an error code is present in the PID Status Register (SR).



Function Variables

Variable	Valid Types	Description
number	Constant (0..31)	PID controller number
period	Constant (0..65535)	PID controller execution period (tenths of seconds)

The I/O module, **CNFG PID control block**, must be configured in the Register Assignment when PID functions are used. The I/O module **CNFG PID control block** provides control over the configuration of a PID control block and provides access to the control block parameters. Control block parameters are assigned to 25 consecutive holding registers. The I/O module address refers to the PID control block number.

Notes

The SCADAPack 32 controller does not support the PID function. See the functions **PIDA – Analog Output PID** and the function **PIDD – Digital Output PID**.

5.52 PIDA – Analog Output PID

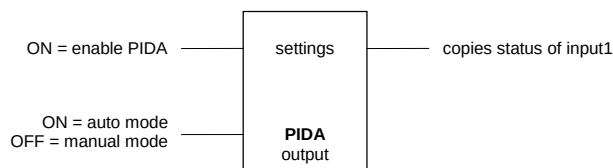
Description

The **PIDA** function performs a PID algorithm and calculates an analog output.

When the **enable PIDA** input is ON, the function executes the PID algorithm. Execution of the PID algorithm is stopped when the **enable PIDA** is OFF.

When the **auto/manual mode** input is ON, the PID is placed in automatic mode. In automatic mode the **output** is calculated using the PID algorithm. A new calculation is done at the rate specified by **cycle time**.

When the **auto/manual mode** input is OFF, the PID is placed in manual mode. In manual mode the **output** is set to the value specified by **manual mode output**. The **output** is limited to the range set by **full** and **zero**.



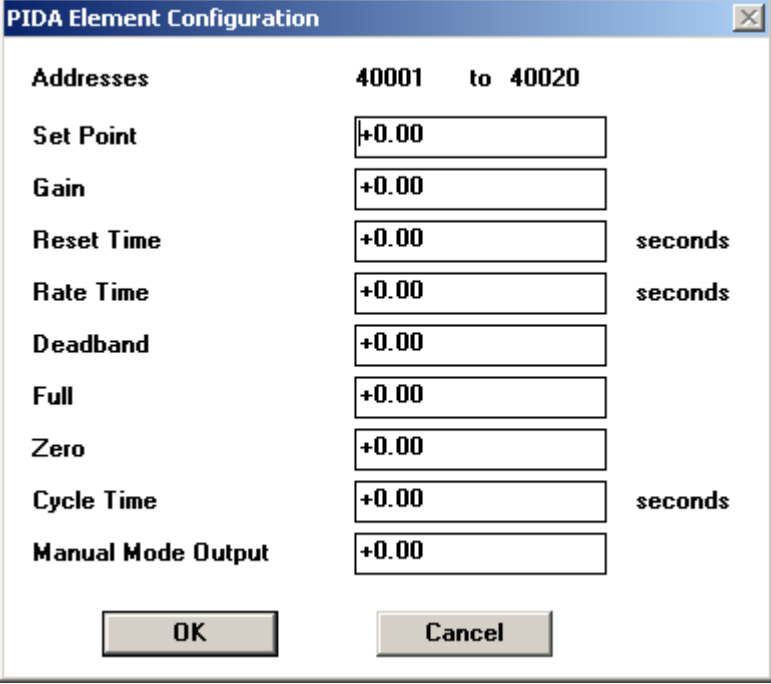
Function Variables

Variable	Valid Types	Description
Settings	holding register (4xxxx)	Address of the first register in the settings control block. There are 20 registers in the block at addresses settings+0 to settings+19. +0,1 = process value (float) +2,3 = setpoint (float) +4,5 = gain (float) +6,7 = reset time in seconds (float) +8,9 = rate time in seconds (float) +10,11 = deadband (float) +12,13 = full (float) +14,15 = zero (float) +16,17 = cycle time in seconds (float) +18,19 = manual mode output (float)
Output	holding register (4xxxx)	+0,1 = PID output (float) +2,3 = internal: process value N-1st (float) +4,5 = internal: process value N-2nd (float) +6,7 = internal: error N-1st (float) +8,9 = internal: time of last scan (UINT32)

The **process value** is a value that represents the actual state of the process being controlled. This value is typically input to the controller as an analog input signal or as a signal from a MVT. The process value is a floating-point number. If the process value is an analog input signal to the controller it must be converted to a floating-point number using the STOF function. If the analog input requires scaling use the SCAL function to scale the input and provide a floating-point number for the process value.

Element Configuration

This element is configured using the PIDA Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **E**lement **C**onfiguration command on the **E**dit menu to modify the configuration block.



The image shows a 'PIDA Element Configuration' dialog box. It has a title bar with the text 'PIDA Element Configuration' and a close button. The dialog contains a table of configuration parameters. The first row is 'Addresses' with the value '40001 to 40020'. The following rows are 'Set Point', 'Gain', 'Reset Time', 'Rate Time', 'Deadband', 'Full', 'Zero', 'Cycle Time', and 'Manual Mode Output', each with a text input field containing '+0.00'. To the right of the 'Reset Time', 'Rate Time', and 'Cycle Time' fields, the unit 'seconds' is displayed. At the bottom of the dialog are two buttons: 'OK' and 'Cancel'.

Addresses	40001 to 40020
Set Point	+0.00
Gain	+0.00
Reset Time	+0.00 seconds
Rate Time	+0.00 seconds
Deadband	+0.00
Full	+0.00
Zero	+0.00
Cycle Time	+0.00 seconds
Manual Mode Output	+0.00

PID Velocity Algorithm

The PIDD function uses the velocity form of the PID algorithm. The velocity form calculates the change in the output and adds it to the previous output.

$$e_n = s_n - p_n$$

$$m_n = m_{n-1} + K \left[e_n - e_{n-1} + \frac{T}{T_i} e_n + \frac{R}{T} (p_n - 2p_{n-1} + p_{n-2}) \right]$$

Where: e = error

s = setpoint

p = process value

K = gain

T = execution period

T_i = integral or reset time

R = rate gain

m = output

The Element Configuration dialog is used to input each parameter into the calculation. These parameters are described below.

Setpoint

The **setpoint** is a floating-point value representing the desired value of the process value. The error value is the difference between the process value and the setpoint.

error = process value – setpoint (+/- deadband).

Process Value

The **process value** is a value that represents the actual state of the process being controlled. See the Function Variables section above for the registers to use for the process value input to the PID.

Gain

The proportional (P) part of the PID algorithm is the **gain**. A positive value of gain configures a forward-acting PID controller and a negative value of gain configures a reverse acting controller.

Reset Time

The integral (I) part of the PID algorithm is the **reset time**. This value, in seconds, controls the reset gain (or magnitude of integral action) in a PI or PID controller. This is typically referred to as Seconds Per Repeat. From the equation above it is seen that the integral action of the PI or PID controller is a function of the reset time and the execution period (cycle time). A smaller reset time provides more integral action and a larger reset time provides less integral action. Valid range is any value greater than 0. A value of 0 disables the reset action.

Rate Gain

The derivative (D) part of the PID algorithm is the **rate time**. This value, in seconds, controls the rate gain (or magnitude of derivative action) in a PD or PID controller. From the equation above it is seen that the derivative action of the PD or PID controller is a function of the rate gain and the execution period (cycle time). A larger rate gain provides more derivative action and a smaller rate gain provides less derivative action. Valid range is any value greater than 0. A value of 0 disables the rate action.

Deadband

The **deadband** parameter is used by the PID algorithm to determine if the process requires the control outputs to be changed. If the absolute value of the error is less than the deadband, then the function block skips execution of the control algorithm. This prevents changes to the output when the process value is near the setpoint and can reduce wear on the control elements. Valid range is any value greater than 0. The **setpoint** is a floating-point value representing the desired value of the process value.

Full

The **full** setting is used in limiting the maximum output value of the PIDA function. If the PID algorithm calculates an **output** quantity that is greater than the value stored in **full**, the output quantity is set equal to the value stored in **full**. The **full** setting should always be greater than the **zero** setting.

Zero

The **zero** setting is used in limiting the minimum output value of the PIDA function. If the PID algorithm calculates an **output** quantity that is less than the value stored in **zero**, the **output** quantity is set equal to the value stored in **zero**. The **zero** setting should always be less than the **full** setting.

Cycle Time

The **cycle time** is the floating-point value of the PID algorithm execution period measured in seconds. Any value greater than or equal to 0.001 seconds (1 ms) may be specified. If the cycle time specified is less than the scan time of the TelePACE program, the program scan time becomes the PID cycle time.

Manual Mode

The **manual mode output** is the value that the **output** is set to when the PIDA function is in manual mode.

Related Functions

PIDD – Digital Output PID

5.53 PIDD – Digital Output PID

Description

The **PIDD** function performs a PID velocity algorithm (see description below) and operates two discrete outputs to maintain PID control.

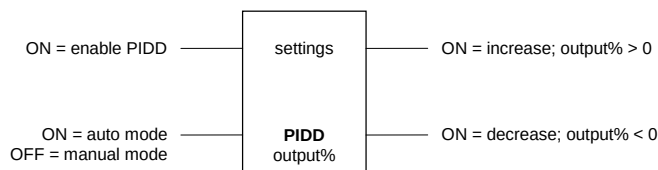
The output of the PIDD is the output percent value **output%**. The **output%** is a duty cycle of the **cycle time** as defined by the **zero%** and **full%** parameters.

The **increase** output is cycled when the **output%** is greater than zero. The **decrease** output is cycled when the **output%** is less than zero.

When the **enable PIDD** input is ON, the function executes the PID algorithm. Execution of the PID algorithm is stopped when the **enable PIDD** is OFF.

When the **auto/manual mode** input is ON, the PID is placed in automatic mode. In automatic mode the output is calculated using the PID algorithm. A new calculation is done at the rate specified by **cycle time**.

When the **auto/manual mode** input is OFF, the PID is placed in manual mode. In manual mode the **output%** is set to the value specified by **manual output%**. The output is limited to the range set by **full%** and **zero%**.



Function Variables

Variable	Valid Types	Description
settings	holding register (4xxxx)	Address of the first register in the settings control block. There are 21 registers in the block at addresses settings+0 to settings+20. +0,1 = process value (float) +2,3 = setpoint (float) +4,5 = gain (float) +6,7 = reset time in seconds (float) +8,9 = rate time in seconds (float) +10,11 = deadband (float) +12,13 = full% (float) +14,15 = zero% (float) +16,17 = cycle time in seconds (float) +18,19 = manual mode output% (float) +20 = motor output enabled
output%	holding register (4xxxx)	+0,1 = PID output % (float) +2,3 = internal: process value N-1 st (float)

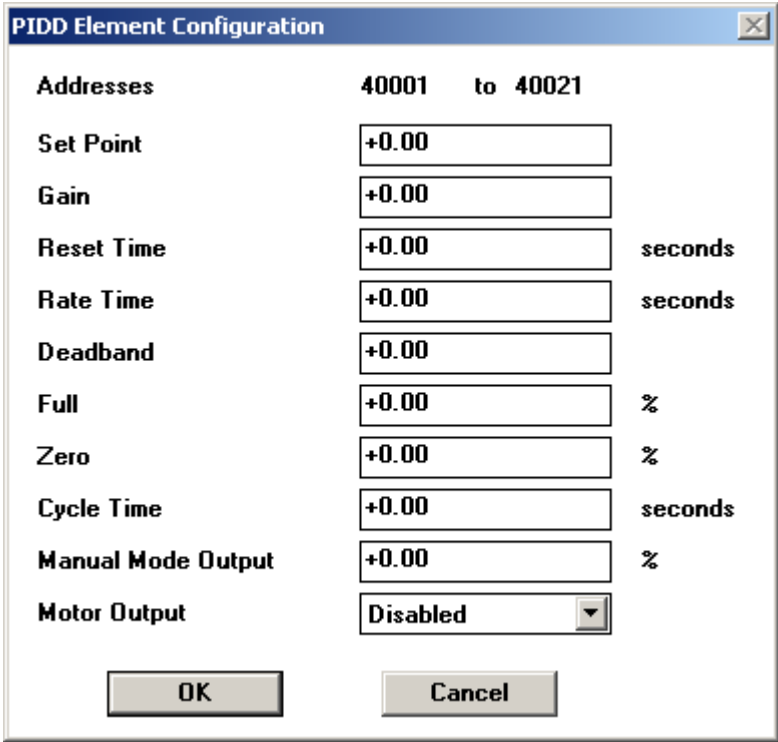
Variable	Valid Types	Description
		+4,5 = internal: process value N-2 nd (float) +6,7 = internal: error N-1 st (float) +8,9 = internal: time of last scan (UINT32) +10,11 = internal: on time for controller (UINT32)

The **process value** is a value that represents the actual state of the process being controlled. This value is typically input to the controller as an analog input signal or as a signal from a MVT. The process value is a floating-point number. If the process value is an analog input signal to the controller it must be converted to a floating-point number using the STOF function. If the analog input requires scaling use the SCAL function to scale the input and provide a floating-point number for the process value.

The remaining **settings** and **output%** registers are described in the Element Configuration section below.

Element Configuration

This element is configured using the PID Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.



The image shows a software dialog box titled "PID Element Configuration". It contains several input fields and a dropdown menu, each with a label to its left. The fields are for Set Point, Gain, Reset Time, Rate Time, Deadband, Full, Zero, Cycle Time, Manual Mode Output, and Motor Output. The Motor Output field is a dropdown menu currently set to "Disabled". At the bottom of the dialog are "OK" and "Cancel" buttons. The dialog also displays the address range "40001 to 40021" at the top.

Parameter	Value	Unit
Addresses	40001 to 40021	
Set Point	+0.00	
Gain	+0.00	
Reset Time	+0.00	seconds
Rate Time	+0.00	seconds
Deadband	+0.00	
Full	+0.00	%
Zero	+0.00	%
Cycle Time	+0.00	seconds
Manual Mode Output	+0.00	%
Motor Output	Disabled	

PID Velocity Algorithm

The PID function uses the velocity form of the PID algorithm. The velocity form calculates the change in the output and adds it to the previous output.

$$e_n = s_n - p_n$$

$$m_n = m_{n-1} + K \left[e_n - e_{n-1} + \frac{T}{T_i} e_n + \frac{R}{T} (p_n - 2p_{n-1} + p_{n-2}) \right]$$

Where: e = error

s = setpoint

p = process value

K = gain

T = execution period

T_i = integral or reset time

R = rate gain

m = output

The Element Configuration dialog is used to input each parameter into the calculation. These parameters are described below.

Setpoint

The **setpoint** is a floating-point value representing the desired value of the process value. The error value is the difference between the process value and the setpoint.

error = process value – setpoint (+/- deadband).

Process Value

The **process value** is a value that represents the actual state of the process being controlled. See the Function Variables section above for the registers to use for the process value input to the PID.

Gain

The proportional (P) part of the PID algorithm is the **gain**. A positive value of gain configures a forward-acting PID controller and a negative value of gain configures a reverse acting controller.

Reset Time

The integral (I) part of the PID algorithm is the **reset time**. This value, in seconds, controls the reset gain (or magnitude of integral action) in a PI or PID controller. This is typically referred to as Seconds Per Repeat. From the equation above it is seen that the integral action of the PI or PID controller is a function of the reset time and the execution period (cycle time). A smaller reset time provides more integral action and a larger reset time provides less integral action. Valid range is any value greater than 0. A value of 0 disables the reset action.

Rate Gain

The derivative (D) part of the PID algorithm is the **rate time**. This value, in seconds, controls the rate gain (or magnitude of derivative action) in a PD or PID controller. From the equation above it is seen that the derivative action of the PD or PID controller is a function of the rate gain and the execution period (cycle time). A larger rate gain provides more derivative action and a smaller rate gain provides less derivative action. Valid range is any value greater than 0. A value of 0 disables the rate action.

Deadband

The **deadband** parameter is used by the PID algorithm to determine if the process requires the control outputs to be changed. If the absolute value of the error is less than the deadband, then the function block skips execution of the control algorithm. This prevents changes to the output when the process value is near the setpoint and can reduce wear on the control elements. Valid range is any value greater than 0.

Full Scale Output

The **full%** setting is the full-scale output limit in percent of **cycle time**. For example if the cycle time is 10 seconds and the full% value is 100 then the maximum duty cycle of the output% is 100 percent or 10 seconds.

When the zero% value is 0 or greater then the increase output is turned on for the duty cycle of the output%. When the zero% value is less than zero the decrease output is turned on for the duty cycle of the output% when the output% is negative.

Zero Scale Output

The **zero%** setting is the zero scale output limit in percent of **cycle time**. When the zero% value is 0 or greater then the increase output is turned on for the duty cycle of the output%. When the zero% value is less than zero the decrease output is turned on for the duty cycle of the output% when the output% is negative.

Cycle Time

The **cycle time** is the floating-point value of the PID algorithm execution period measured in seconds. Any value greater than or equal to 0.001 seconds (1 ms) may be specified. If the cycle time specified is less than the scan time of the TelePACE program, the program scan time becomes the PID cycle time.

Manual Mode Output

The **manual mode output%** is the value that the **output%** is set to when the PID function is in manual mode.

Motor Output

When the **motor output** is set to **enabled** the **increase** and **decrease** outputs are de-energized when error is within the deadband. When the **motor output** is set to **disabled** the **increase** and **decrease** outputs operate continuously based on the **output%**.

Related Functions

PIDA – Analog Output PID

5.54 POWR – Floating-Point Raised to Power

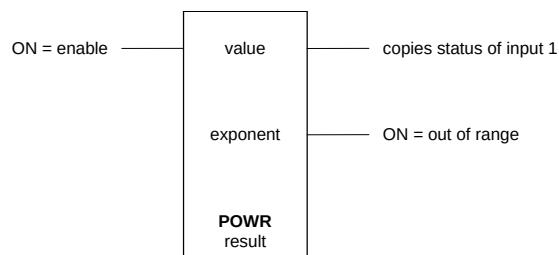
Description

The **POWR** function calculates the result of a value raised to an exponent and stores the result in a floating-point holding register.

When the **enable** input is ON, **value** is raised to the **exponent** and stored in the **result** floating-point register.

The top output is ON when the input is.

The **out-of-range** output is ON if the **value** is zero and **exponent** is less than zero; and if **value** is negative. The result is not calculated in these cases.



Function Variables

Variable	Valid Types	Description
Value	Floating Point Constant 2 input registers (3xxxx) 2 holding register (4xxxx)	Floating Point value to be raised. The high order word is stored in the first register.
Exponent	Floating Point Constant 2 input registers (3xxxx) 2 holding register (4xxxx)	Floating Point value to raise value to. The high order word is stored in the first register.
Result	2 holding register (4xxxx)	result = value ^{exponent} The high order word is stored in the first register.

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

The calculation is performed using a single-precision floating-point number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38} .

Related Functions

ABSF - Floating-Point Absolute Value

ADDF - Add Floating-Point Values

DIVE - Divide Floating-Point Values

MULF - Multiply Floating-Point Values

SQRF - Square Root of Floating-Point Value

SUBF – Subtract Floating-Point Values

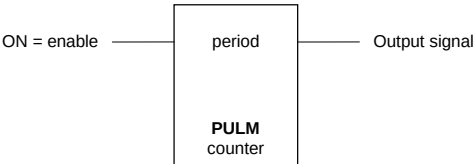
5.55 PULM - Pulse Minutes

Description

The **PULM** function controls a square wave digital signal.

While the **enable** is ON, the **output** signal will be active ON or OFF based on the period. The output is powered only when the enable input is ON.

For example when the enable input is on and a period of 10 minutes is used the result will be an output square wave that is ON for 5 minutes and then OFF for 5 minutes.



Function Variables

Variable	Valid Types	Description
Period	Constant (1..65535) input register (3xxxx) holding register (4xxxx)	The period of the waveform, in minutes.
Counter	2 holding registers (4xxxx)	Contains the accumulated time of the period in seconds.

Notes

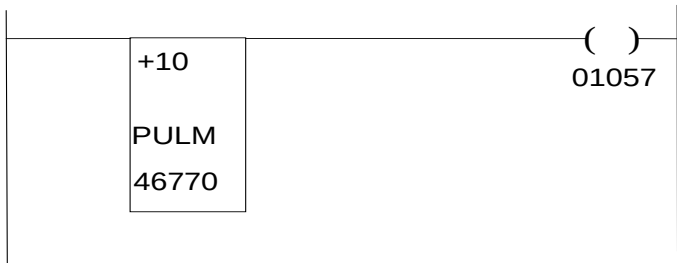
The Period is from 1 to 65535 minutes.

Related Functions

PULS - Pulse Seconds

Example

network 1:



In this example the PULM function has a period value of 10 minutes. Coil 01057 will be on for 5 minutes and then off for 5 minutes. When the program is first run coil 01057 will start in on state for 5 minutes then go off for 5 minutes, repeating this cycle as long as the enable input is true. Registers 46770 and 46771 contain the accumulated time for the period in seconds.

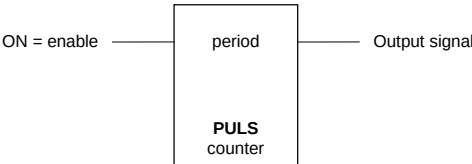
5.56 PULS - Pulse Seconds

Description

The **PULS** function controls a square wave digital signal.

While the **enable** is ON, the **output** signal will be active ON or OFF based on the period. The output is powered only when the enable input is ON.

For example when the enable input is on and a period of 10 seconds is used the result will be an output square wave that is ON for 5 seconds and then OFF for 5 seconds.



Function Variables

Variable	Valid Types	Description
Period	Constant (1..65535) input register (3xxxx) holding register (4xxxx)	The period of the waveform, in seconds.
Counter	2 holding registers (4xxxx)	Contains the accumulated time of the period in tenths of seconds.

Notes

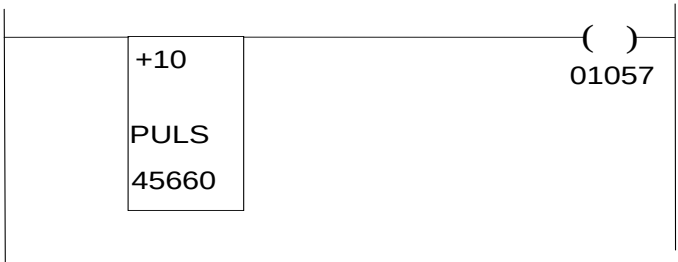
The Period is from 1 to 65535 seconds. The smallest actual period will depend on the ladder scan time. Smaller ladder programs will be able to handle smaller periods.

Related Functions

Timers

Example

network 1:



In this example the PULM function has a period value of 10 seconds. Coil 01057 will be on for 5 seconds and then off for 5 seconds. When the program is first run coil 01057 will start in on state for 5 seconds then go off for 5 minutes, repeating this cycle as long as the enable input is true. Registers 46770 and 46771 contain the accumulated time for the period in seconds.

5.57 PUTB – Put Bit into Block

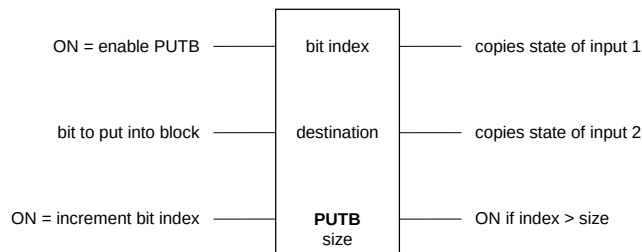
Description

The **PUTB** function block writes the status of a bit, at the **bit index**, into the **destination** block of registers.

The **enable PUTB** input must be energized for a PUTB function to execute.

The **bit to put into block** input is the bit value that will be written into the destination register at the location of the bit index when **enable PUTB** is energized.

The **increment bit index** input increments the index, once each scan, when **enable PUTB** is energized.



Function Variables

Variable	Valid Types	Description
bit index	Constant (0..65535) Holding register (4xxxx)	The index of the bit within the destination block. A bit index of 0 is the most significant bit of the first register of the destination block.
Destination	coil block (0xxxx) holding register (4xxxx)	The address of the destination register block. The address for a coil register block is the first register in a group of 16 registers.
Size	Constant (1..100)	number of 16 bit words in the block.

Notes

Put bit into block accesses 16 bit words. Coil blocks are groups of 16 registers that start with the register specified as the block address. Coil blocks must begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 00033, etc.

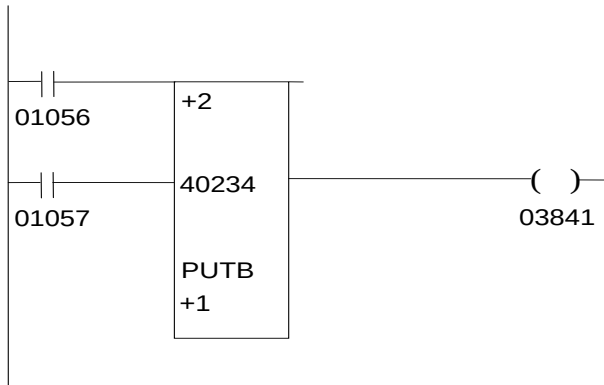
Related Functions

GETB – Get Bit from Block

ROTB – Rotate Bits in Block

CMPB – Compare Bit

Example



In network 1 the PUTB function is being used to set the manual control bit in the PID Block 0 Control Register, bit 13 of register 40234. The bit index of the manual control bit is 2, with 0 being the most significant bit.

Closing contacts 01056 allows power to be supplied to the **enable PUTB** input of the PUTB function block. With the function block enabled, closing contacts 01057 allows power to the **put bit into block** input of the function block. The bit at the index is then set in the destination register, the PID Block 0 Control Register. Output coil 03841 is energized.

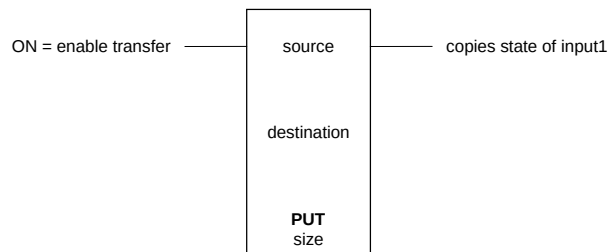
5.58 PUT – Put Signed Value into Registers

Description

The **PUT** function transfers, in one scan, the **source** holding register or signed constant into the **destination** holding registers. The number of destination registers is determined by **size**. The same value is stored in each destination register.

The source register, or signed constant, is transferred to the destination holding register when **enable transfer** is ON.

If the source is a register, this function is identical to the **PUTU** function.



Function Variables

Variable	Valid Types	Description
source	Constant (-32768..32767) input register (3xxxx) holding register (4xxxx)	The signed value or address of the register containing the value to be stored.
destination	Holding register (4xxxx)	The address of the first destination register.
Size	Constant (1..9999)	The number of destination registers.

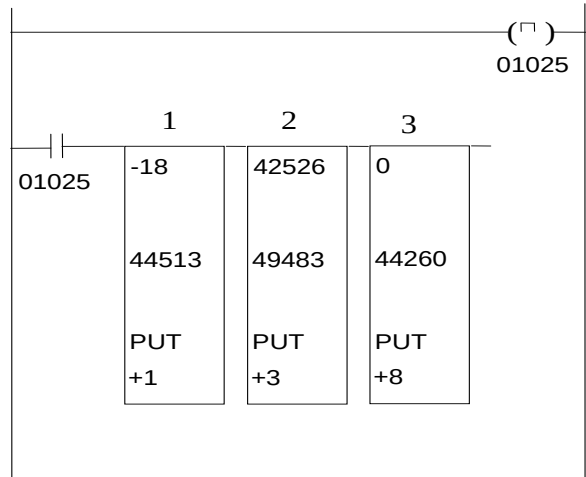
Related Functions

PUTU – Put Unsigned Value into Registers

PUTF - Put Floating-Point Value

Example

network 1:



In this example there are three PUT functions in network 1. On power up One Shot coil 01025 is energized for one scan. This will cause NO contacts 01025 to close, applying power to the enable transfer input on PUT functions 1,2 and 3.

In PUT 1 the constant value -18 is transferred to holding register 44513.

In PUT 2 the value of holding register 42526 (1000) is transferred to holding registers 49483 through 49485.

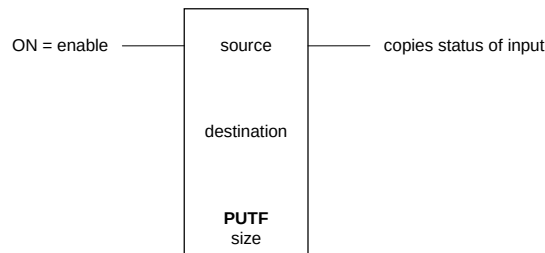
In PUT 3 the constant value 0 is transferred to holding registers 44260 through 44267.

5.59 PUTF - Put Floating-Point Value

Description

The PUTF element transfers, in one scan, the **source** floating-point holding register or constant into the **destination** floating-point holding registers. The number of floating-point destination registers is determined by **size**. The same value is stored in each floating-point register pair.

When the **enable** input is ON, the source register or constant is transferred to the destination registers. The element output is ON when the input is.



Function Variables

The element has three parameters.

Variable	Valid Types	Description
source	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	Floating-point register or constant to be stored
Destination	Holding register (4xxxx)	Address of the first pair of destination registers
Size	Constant (1...4999)	the number of floating-point destination registers pairs to store

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number. Floating point numbers can represent positive and negative values in the range -3.402×10^{38} to 3.402×10^{38} .

The size parameter determines the number of floating-point register pairs affected. The number of registers affected is twice the value of size, since each floating-point value uses two registers.

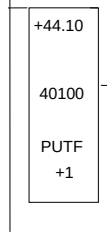
Related Functions

PUT – Put Signed Value into Registers

PUTU – Put Unsigned Value into Registers

Example

network 1:



The PUTF function in network 1 puts the floating-point constant +44.10 value into floating-point register 40100 (registers 40100 and 40101).

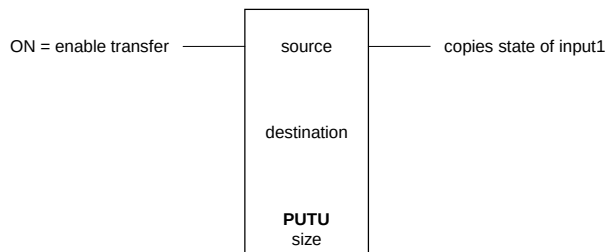
5.60 PUTU – Put Unsigned Value into Registers

Description

The **PUTU** function transfers, in one scan, the **source** holding register or unsigned constant into the **destination** holding registers. The number of destination registers is determined by **size**. The same value is stored in each destination register.

The source register, or unsigned constant, is transferred to the destination holding register when **enable transfer** is ON.

If the source is a register, this function is identical to the **PUT** function.



Function Variables

Variable	Valid Types	Description
source	Constant (0..65535) input register (3xxxx) holding register (4xxxx)	The unsigned value or address of the register containing the value to be stored.
Destination	Holding register (4xxxx)	The address of the first destination register.
Size	Constant (1..9999)	The number of destination registers.

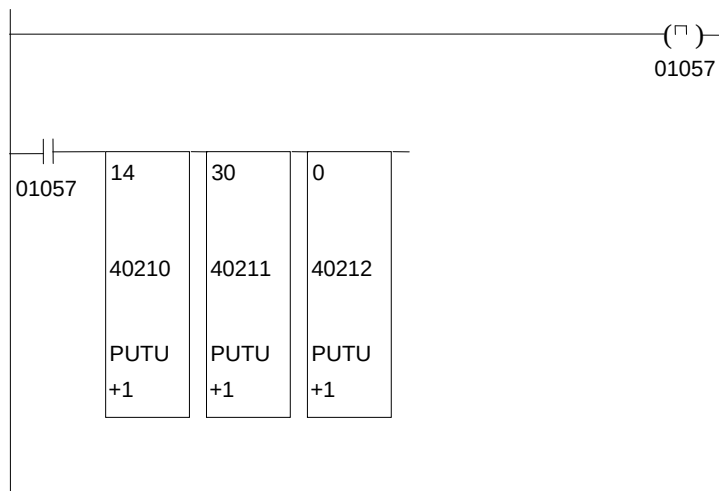
Related Functions

PUT – Put Signed Value into Registers

PUTF - Put Floating-Point Value

Example

network 1:



This example uses two PUTU functions to write the correct time to the Real Time Clock. The time for this example is 14:30:00.

The holding registers used to in this example must be assigned to the **CNFG Real Time Clock and Alarm I/O Module** in the Register Assignment. Holding registers 40210 through 40220 are assigned to this I/O module.

On power up One Shot coil 01057 is energized for one scan. This will cause NO contacts 01057 to close, applying power to the enable transfer inputs on the two PUTU functions.

The real time clock continues to run with the new time setting immediately after these PUTU functions are enabled. Note that all seven clock registers must be set to valid values for the clock to operate correctly.

Another method for setting the clock, is to use the *Edit/Force Register* dialog to write the current time to the appropriate module registers. Leave the force box in the dialog unchecked so that the data is only written, not forced.

5.61 R→L – Register to List Transfer

Description

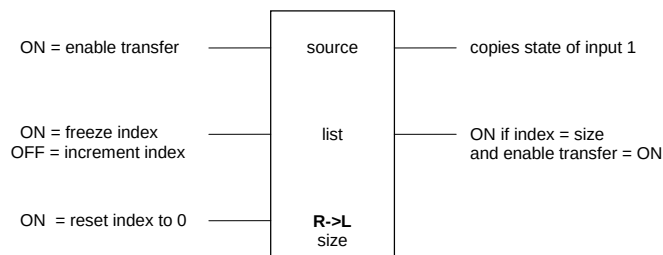
The **R→L** function transfers the contents of the source register into a register in the destination list of registers indicated by the index.

When the **enable transfer** input is ON and the **index equals size** output is OFF, the source register is transferred to the destination list register at the position of the index.

The **freeze/increment index** input allows control of the index. If this input is ON the index is not incremented. If the input is OFF the index is incremented by one after each transfer.

When the **reset index** input is ON the index is reset to zero.

The index is incremented by one on each transfer until the index equals size. When the **index equals size** output is ON no further increments of the index value occur until the index is reset.



Function Variables

Variable	Valid Types	Description
source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The address of the source register. The address for a coil or status register block is the first register in a group of 16 registers that will be read.
index and destination	Holding register (4xxxx)	The address of the index register and the data registers. The index is stored in register [4xxxx]. The data is stored in register [4xxxx+1] to register [4xxxx+size]
Size	Constant (1..9999)	The number of registers in the list.

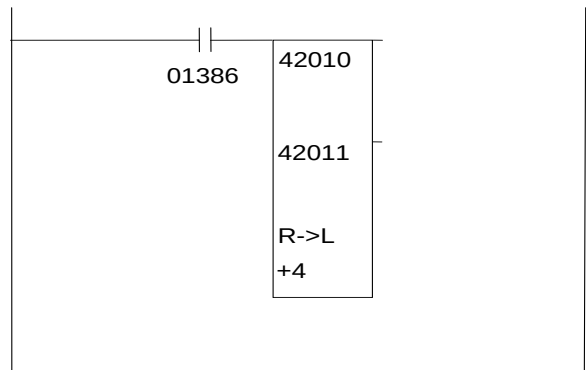
Related Functions

L→L – List to List Transfer

L→R – List to Register Transfer

MOVE – Move Block

Example



The Register to List function transfers the contents of the source register 42010 into a register in the destination list of registers, 42012 to 42015, indicated by the index register 42011.

Destination Register List		
Source Register->	List Index-->	42011
	list register 0-->	42012
	list register 1-->	42013
	list register 2-->	42014
	list register 3-->	42015

An example of the registers used in a Register to List (R->L) transfer is shown in the above diagram. The source register is a general purpose analog output register. The index is a general purpose analog output register that will have a value of 0, 1, 2 or 3. The destination register list starts at the next sequential register after the index register. The destination list contains four general purpose analog output registers and has a size of 4.

When list index register 42011 has a value of 0 and the R->L function is enabled the contents of source register 42010 is transferred to destination list register 40012.

When list index register 42011 has a value of 1 and the R->L function is enabled the contents of source register 42010 is transferred to destination list register 42013.

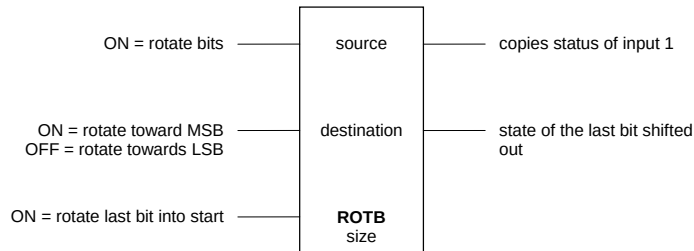
When list index register 42011 has a value of 2 and the R->L function is enabled the contents of source register 42010 is transferred to destination list register 42014.

When list index register 42011 has a value of 3 and the R->L function is enabled the contents of source register 42010 is transferred to destination register 42015.

5.62 ROTB – Rotate Bits in Block

Description

The **ROTB** function block rotates the bits in the **source** block by one bit and stores the result in the **destination block**. The block can be rotated towards either the most significant bit (MSB) or towards the least significant bit (LSB).



Function Variables

Variable	Valid Types	Description
Source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The address of the first source register. The address for a coil or status register block is the first register in a group of 16 registers that will be rotated.
Destination	coil block (0xxxx) holding register (4xxxx)	The address of the first destination register. The address for a coil or status register block is the first register in a group of 16 registers that will be rotated.
Size	Constant (1..100)	The number of 16 bit words in the block.

Related Functions

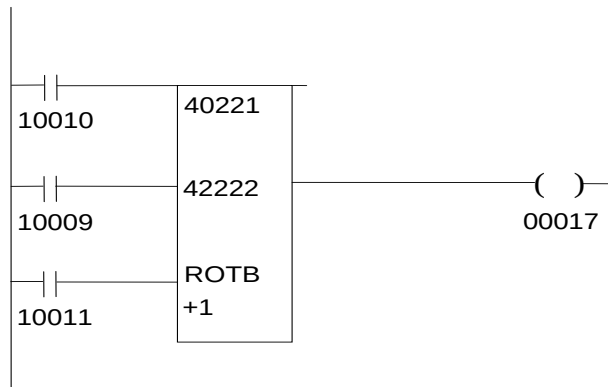
PUTB – Put Bit into Block

GETB – Get Bit from Block

CMPB – Compare Bit

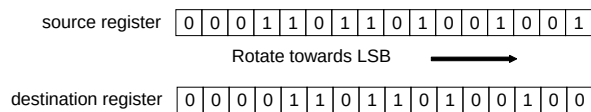
Example

network 1:

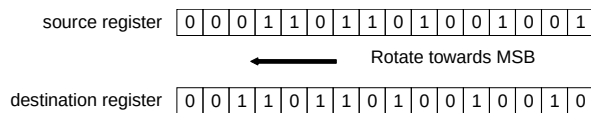


The ROTB function in network 1 is configured to demonstrate the operation of the function block. Register 40221, the source register, has a value of 6985. Register 40222 is the destination register.

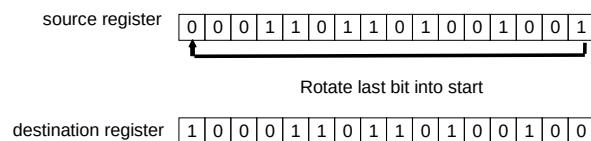
The first example is a rotate towards LSB. Closing the **rotate bits** input, contacts 10010, will cause the bits in the source register to rotate right by one bit and the result to be stored in the destination register.



The next example is a rotate towards MSB. Closing the **rotate towards MSB** input, contacts 10009 and then closing the **rotate bits** input, contacts 10010, will cause the bits in the source register to rotate once to the left. The result is stored in the destination register.



In the last example the **rotate bits** input is enabled by closing contact 10010. Closing contacts 10011 will enable the **rotate last bit to start** input. The LSB of the source register is rotated to the MSB and the remaining bits are rotated right.



5.63 SCAL – Scale Analog Value

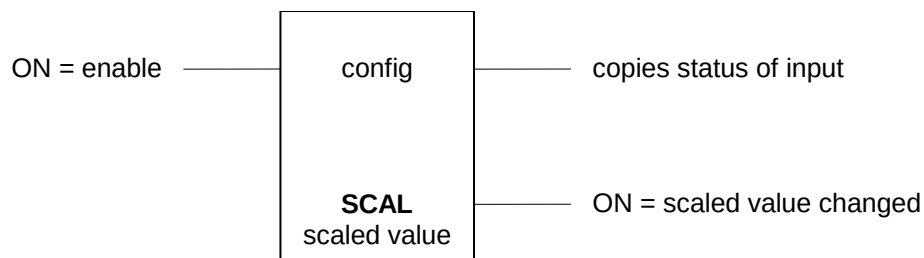
Description

The SCAL function scales an integer into a floating-point value and detects changes in the floating-point value.

When the **Enable** input is ON the SCAL function calculates the scaled value.

When the **Enable** input is ON the scaled value is compared to the previous scaled value. If the difference is greater than the deadband the scaled value changed output is turned ON and the scaled value is copied to the previous scaled value. The previous scaled value does not change if scaled value is within the deadband.

When the **Enable** input changes from OFF to ON, the scaled value is copied to previous scaled value.



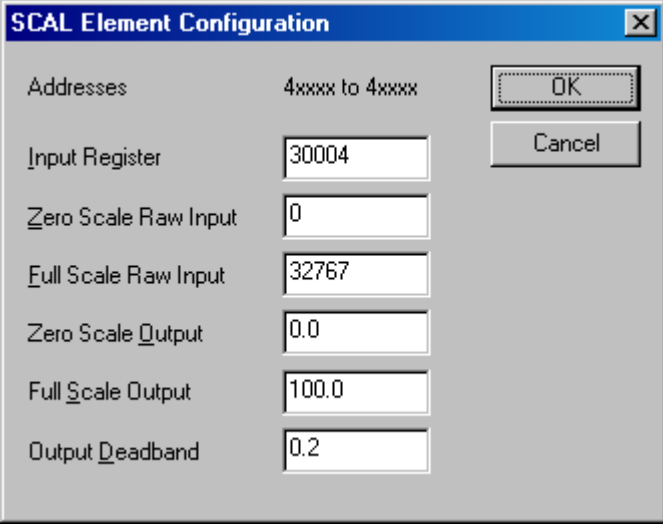
Function Variables

Variable	Valid Types	Description
config	holding register (4xxxx)	Address of the first register in the configuration block. There are 9 registers in the block at addresses message+0 to message+8. +0 = input register +1 = zero scale raw input +2 = full scale raw input +3-4 = zero scale output (floating point) +5-6 = full scale output (floating point) +7-8 = deadband (floating point)
scaled value	4 holding registers (4xxxx)	Address of the first register in the output block. There are 4 registers in the block at addresses output+0 to output+3 +0-1: scaled value (floating point) +2-3: previous scaled value (floating point)

Element Configuration

This element is configured using the SCAL Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.

WARNING: If the controller is initialized, using the **Initialize** command in the **Controller** menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.



The image shows a 'SCAL Element Configuration' dialog box. It has a title bar with a close button. The dialog contains several input fields and two buttons. The fields are: 'Addresses' with a value of '4xxxx to 4xxxx', 'Input Register' with '30004', 'Zero Scale Raw Input' with '0', 'Full Scale Raw Input' with '32767', 'Zero Scale Output' with '0.0', 'Full Scale Output' with '100.0', and 'Output Deadband' with '0.2'. The 'OK' button is highlighted with a dashed border.

Addresses	4xxxx to 4xxxx	OK
Input Register	30004	Cancel
Zero Scale Raw Input	0	
Full Scale Raw Input	32767	
Zero Scale Output	0.0	
Full Scale Output	100.0	
Output Deadband	0.2	

The **Input Register** specifies the register from which the input is read. Valid values are any input (3xxxx) or holding (4xxxx) register. The range depends upon the registers supported in the selected controller type.

The **Zero Scale Raw Input** specifies the minimum raw input value. Valid values are -32768 to 32767. This value must be less than the full-scale raw input.

The **Full Scale Raw Input** specifies the maximum raw input value. Valid values are -32768 to 32767. This value must be greater than the zero scale raw input.

The **Zero Scale Output** is the floating-point value calculated when the input is equal to the zero scale raw input. This value must be less than the full-scale output. Valid values are any floating-point number.

The **Full Scale Output** is the floating-point value calculated when the input is equal to the full-scale raw input. This value must be greater than the zero scale output. Valid values are any floating-point number.

The **Output Deadband** is the amount by which the output must change for the **scaled value changed** output to be energized. Valid values are any floating-point number greater than or equal to 0.0.

Notes

The input is scaled using this formula.

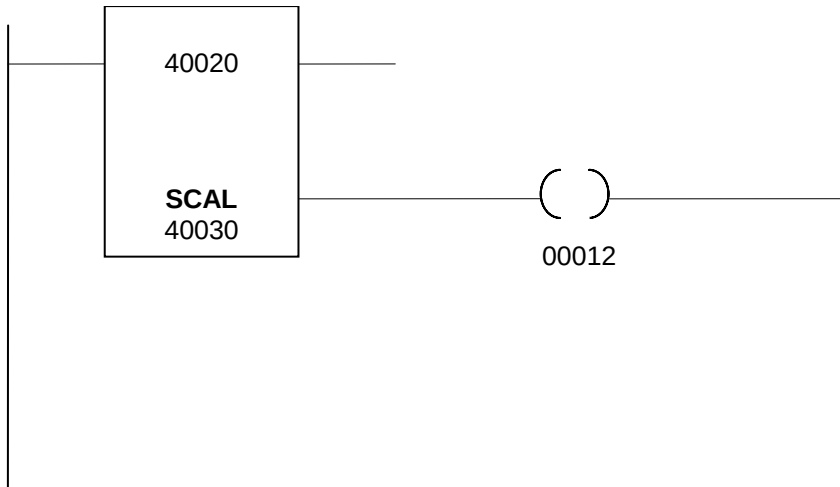
$$scaled = \frac{input - zero\_raw}{full\_raw - zero\_raw} \times (full\_output - zero\_output) + zero\_output$$

The previous scaled value changes only when difference between the current and previous values is greater than the deadband. The **scaled value changed** output is energized when this occurs.

The input register may contain values less than the zero scale raw input or greater than the full-scale raw input. The scaling calculation is still performed.

Example

In this example a SCAL block is used to convert an analog input into engineering units and close a contact each time a change of 10 units occurs.



The element configuration for the SCAL block is as follows

Parameter	Value	Notes
Input Register	30001	assign to analog input module in register assignment
Zero Scale Raw Input	0	minimum value for unipolar analog input
Full Scale Raw Input	32767	maximum value for analog input
Zero Scale Output	0	corresponds to the minimum value of analog input
Full Scale Output	1000	corresponds to the maximum value of analog input
Output Deadband	10	change in output to turn on change output

5.64 Shunts

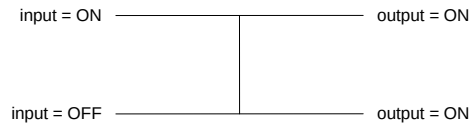
Horizontal

The horizontal shunt conducts power from left to right.



Vertical

The vertical shunt conducts power between rows. If any of the rows connected by vertical shunts are powered, all output rows will be powered, as shown in the example below.

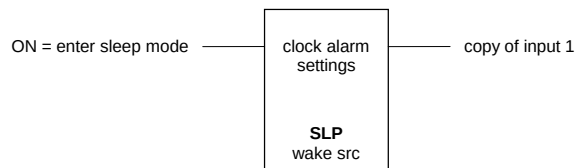


5.65 SLP – Put Controller into Sleep Mode

Description

The **SLP** function places the controller into sleep mode. Sleep mode reduces power consumption to a minimum by halting the microprocessor clock and shutting down the power supply. All programs halt until the controller resumes execution. All output points turn off while the controller is in sleep mode.

This function is not supported in the SCADAPack 32, SCADAPack 100 or SCADASense Series controllers.



Function Variables

Variable	Valid Types	Description
clock alarm settings	holding register (4xxxx)	Address of the first register in the clock alarm settings block. There are 4 registers in the block at addresses settings+0 to settings + 3 +0 = type +1 = hour +2 = minute +3 = second Valid values for type are 0 = no alarm 1 = absolute time alarm 2 = elapsed time alarm
wake src	holding register (4xxxx)	address of the wake up source register. This register indicates the reason the controller left sleep mode. Valid values are: 1 = real time clock alarm 2 = interrupt input 4 = led power switch 8 = counter 0 overflow 16 = counter 1 overflow 32 = counter 2 overflow 256 = Assertion of: - digital input 0 for SCADAPack 350 - Any digital input on SCADAPack LP - INT/Cntr digital Input on SCADAPack with 5203/5204 controller board

Notes

The SCADAPack 32 and SCADAPack 100 controllers do not support the SLP function.

Note that this function block triggers the same interrupt as the CNFG Real Time Clock and Alarm register assignment. Using both alarm settings, therefore, in the same program is not recommended.

The SCADAPack 330, SCADAPack 334 and SCADAPack 350 controllers will not go to sleep if the USB peripheral port is connected to a host device such as a PC.

Element Configuration

This element is configured using the SLP Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.

WARNING: If the controller is initialized, using the **Initialize** command in the **Controller** menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.

The screenshot shows the 'SLP Element Configuration' dialog box. It has a title bar with the text 'SLP Element Configuration' and a close button (X). The main area contains the following fields: 'Addresses' with the value '40198 to 40201', 'Alarm Type' with a dropdown menu showing 'Absolute Time', 'Hours' with a text box containing '20', 'Minutes' with a text box containing '30', and 'Seconds' with a text box containing '0'. At the bottom, there are two buttons: 'OK' and 'Cancel'.

Valid values for the time parameters are listed in the table below.

Alarm Type	Parameter	Range
Absolute	Hours	0 to 23
	Minutes	0 to 59
	Seconds	0 to 59
Elapsed	Hours	0 to 23
	Minutes	0 to 1439
	Seconds	0 to 65535 Total of all time must be less than 23 hours, 59 minutes, 59 seconds

The controller will wake up under the following conditions.

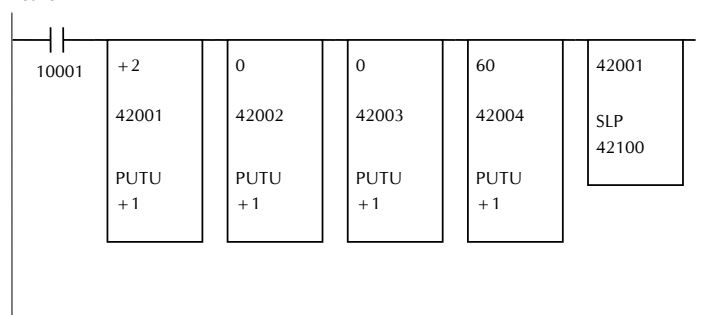
Condition	Wake Up Effects
Hardware Reset	Application programs execute from start of program.
External Interrupt	Program execution continues from point sleep function was executed.
Real Time Clock Alarm	Program execution continues from point sleep function was executed.
LED Power Button	Program execution continues from point sleep function was executed.

Condition	Wake Up Effects
Hardware Counter Rollover	Software portion of counter is incremented. Program execution continues from point sleep function was executed.

Example

The following network will put the controller into sleep mode. The clock alarm is set to wake the controller in 60 seconds. Other conditions may wake up the controller before the 60 seconds expires.

network 1:



The **Element Configuration** dialog may be used to edit the configuration of the SLP block, instead of PUTU functions. This reduces the memory used by the program. However, the alarm settings are written to the controller only when the program is loaded. The network using the PUTU elements configures the alarm settings on every execution of the SLP element.

network 1:

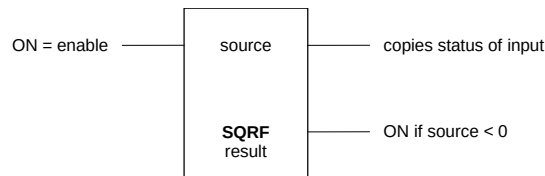


5.66 SQRF - Square Root of Floating-Point Value

Description

The SQRF element stores the square root of a floating-point register or constant in a floating-point holding register.

When the **enable** input is ON the square root of the **source** is stored in the **result** floating-point holding register. The top output is ON when the input is. The bottom output is ON if the source is negative, and the input is ON.



Function Variables

The element has two parameters.

Variable	Valid Types	Description
source	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	a floating-point register or constant
result	2 holding registers (4xxxx)	Floating-point square root of source

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number. Floating point numbers can represent positive or negative values in the range – 3.402×10^{38} to 3.402×10^{38} .

The square root of a negative number cannot be calculated. The source must be positive or zero to calculate a result..

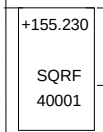
Related Functions

DIV – Divide Signed Values

DIVU – Divide Unsigned Values

Example

network 1:



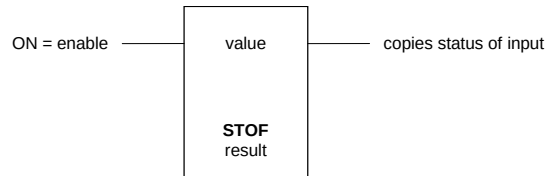
The SQRF function in network 1 takes the square root of the floating-point constant +155.23 and puts the result in floating-point register 40001 (registers 40001 and 40002). The value in 40001 is =12.459.

5.67 STOF - Signed Integer to Floating-Point

Description

The STOF element converts a signed register or constant into a floating-point number and stores the result in a floating-point holding register.

When the **enable** input is ON, **value** is converted into a floating-point number and stored in the **result** floating-point register. The element output is ON when the input is.



Function Variables

The element has two parameters.

Variable	Valid Types	Description
Value	Constant (-32768...32767) input register (3xxxx) holding register (4xxxx)	Signed value to convert
Result	2 holding registers (4xxxx)	Converted floating-point value

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38} .

Related Functions

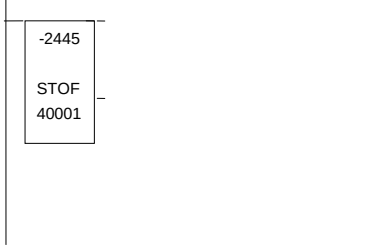
UTOF - Unsigned Integer to Floating-Point

FTOS - Floating-Point to Signed Integer

FTOU - Floating-Point to Unsigned Integer

Example

network 1:

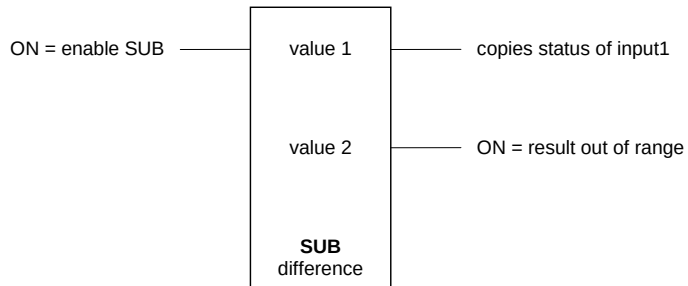


The STOF function in network 1 converts signed integer constant -2445 to a floating-point value and puts the value into floating point-register 40001 (registers 40001 and 40002). In this example the content of floating-point register 40001 is the floating-point value -2445.00.

5.68 SUB – Subtract Signed Values

Description

The **SUB** function block subtracts a signed register or constant, **value 2**, from a signed register or constant, **value 1**, and stores the result in a holding register, **result**. Signed subtraction is used. The **result out of range** output is enabled if the result is greater than 32767 or less than -32768.



Function Variables

Variable	Valid Types	Description
value 1	Constant (-32768..32767) input register (3xxxx) holding register (4xxxx)	Signed value to subtract from
value 2	Constant (-32768..32767) input register (3xxxx) holding register (4xxxx)	Signed value to subtract
Difference	holding register (4xxxx)	Difference = value 1 - value 2

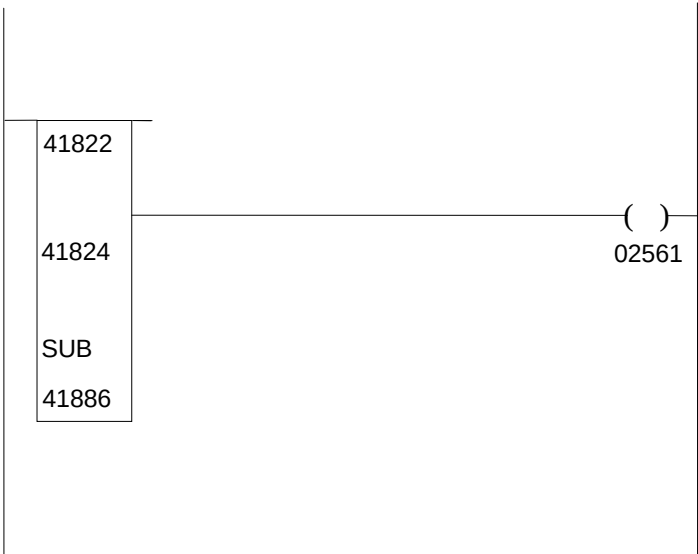
Related Functions

SUBF – Subtract Floating-Point Values

SUBU – Subtract Unsigned Values

Example

network 1:



The SUB function in network 1 subtracts the contents of register 41824, **value 2**, from the contents of register 41822, **value 1**. The **difference** is stored in register 41886. Some examples of different values for registers 41822 and 41824 are shown in the table below.

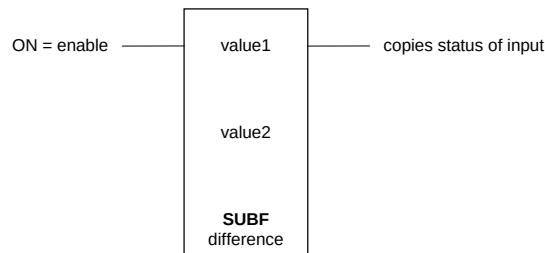
	Register 41822	Register 41824	Register 41886	Coil 02561
	value 1	value 2	difference	out of range
Example 1	-27765	-18755	-9010	OFF
Example 2	3276	-12269	15545	OFF
Example 3	-30000	27881	7655	ON

5.69 SUBF – Subtract Floating-Point Values

Description

The SUBF element subtracts one floating-point register or constant from another and stores the result in a floating-point holding register.

When the **enable** input is ON the difference **value 1 – value 2** is stored in the **difference** holding register. The element output is ON when the input is.



Function Variables

The element has three parameters.

Variable	Valid Types	Description
value1	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	Floating-point register or constant to subtract from
Value2	FP constant 2 input registers (3xxxx) 2 holding registers (4xxxx)	floating-point register or constant to subtract
Difference	2 holding registers (4xxxx)	floating-point difference = value1 – value2

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38} .

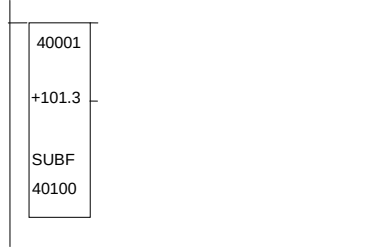
Related Functions

SUB – Subtract Signed Values

SUBU –Subtract Unsigned Values

Example

network 1:



The SUBF function in network 1 subtracts floating-point constant +101.3 from floating-point register 40001 (registers 40001 and 40002). Assuming floating-point register 40001 contains a value of 1000.00 the result is +898.7. The result is stored in floating-point register 40100 (registers 40100 and 40101).

5.70 SUBR - Start of Subroutine

Description

The **SUBR** element defines the start of a **subroutine**.

A subroutine is a group of logic networks that can be executed conditionally. The subroutine begins with the network containing the **SUBR** element. A subroutine ends when another subroutine element is encountered, or when the end of the ladder logic program is reached.

The element is a single cell element. Visually it covers the entire width of the first row of the network.



Function Variables

The **SUBR** element has one parameter:

Variable	Valid Types	Description
number	Constant	The number of the subroutine. Any number in the range 1 to 500 is valid.

Notes

The **SUBR** element must be located in row 1, column 1 of a network.

No other element can be located to the right of the **SUBR** element.

Outputs of a subroutine remain in their last state when a subroutine is not called. For example, a coil that is turned on by a subroutine remains on when the subroutine is *not* called. The output will only turn off when the subroutine is called and it turns the output off.

The subroutine number is a constant. Constant tag names are used with this value.

The subroutine number must be unique. No other subroutine can use the same number.

Subroutines do not have to be programmed in any particular numerical order. For example, subroutine 1 can follow subroutine 2, subroutine 200 can follow subroutine 400.

Related Functions

CALL - Execute Subroutine

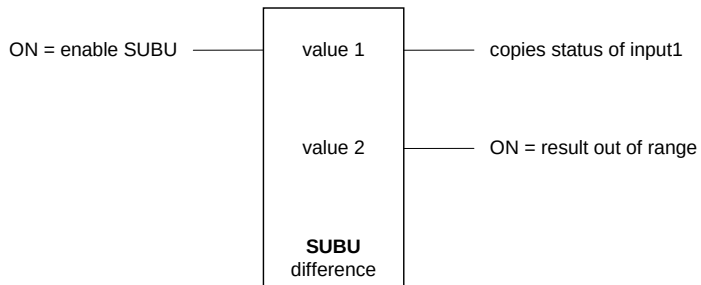
Example

See the example for the CALL function.

5.71 SUBU –Subtract Unsigned Values

Description

The **SUBU** function block subtracts an unsigned register or constant, **value 2**, from an unsigned register or constant, **value 1**, and stores the result in a holding register, difference. Unsigned subtraction is used. The **result out of range** output is enabled if the result is greater than 65535 or less than 0.



Variable	Valid Types	Description
value 1	Constant (0..65535) input register (3xxxx) holding register (4xxxx)	Unsigned value to subtract from
Value 2	Constant (0..65535) input register (3xxxx) holding register (4xxxx)	Unsigned value to subtract
Difference	holding register (4xxxx)	Difference = value 1 - value 2

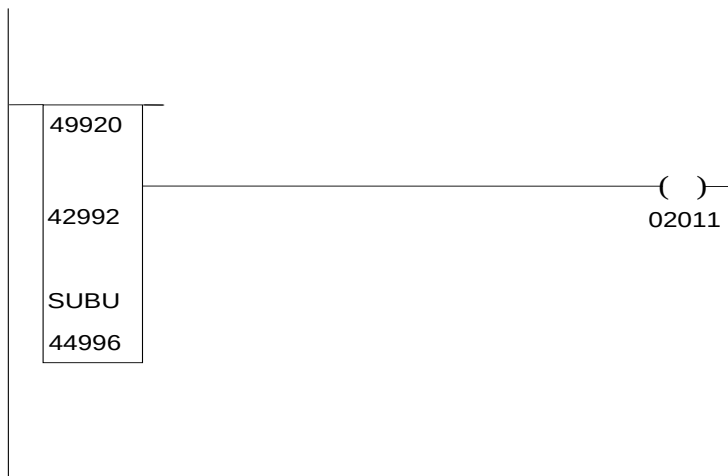
Related Functions

SUB – Subtract Signed Values

SUBF – Subtract Floating-Point Values

Example

network 1:



The SUBU function in network 1 subtracts the contents of register 42992, **value 2**, from the contents of register 49920, **value 1**. The **difference** is stored in register 44996. Some examples of different values for registers 49920 and 42992 are shown in the table below.

	Register 49920	Register 42992	Register 44996	Coil 02011
	value 1	value 2	Difference	out of range
Example 1	22675	1988	20687	OFF
Example 2	61223	2399	58824	OFF
Example 3	3877	54440	14973	ON

5.72 Timers

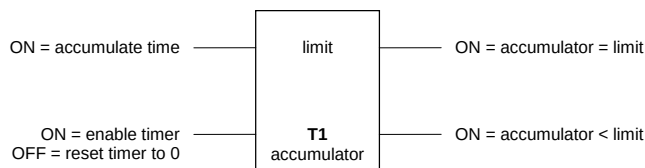
Description

The three timer functions measure elapsed time and save it in a holding register, **accumulator**. The **T1** timer measures time in 1.0 second intervals. The **T.1** timer measures time in 0.1 second intervals. The **T.01** timer measures time in 0.01 second intervals.

When the **enable timer** input is ON and the **accumulate time** is ON the accumulator increments by the Timer interval to the value of the **limit**. When the **enable timer** is off the accumulator is reset to zero.

When the **enable timer** is ON and the **accumulator** equals the **limit** then the **accumulator = limit** output is On.

When the **enable timer** is ON and the **accumulator** is less than the **limit** then the **accumulator < limit** output is On.



Variable	Valid Types	Description
Limit	Constant (1..65535) input register (3xxxx) holding register (4xxxx)	Upper limit for accumulator
Accumulator	Holding register (4xxxx)	Register holds time during which accumulate time input has been ON

Note

Timer functions are affected by the ladder logic scan time in that the Accumulator value is updated only when the Timer function is scanned.

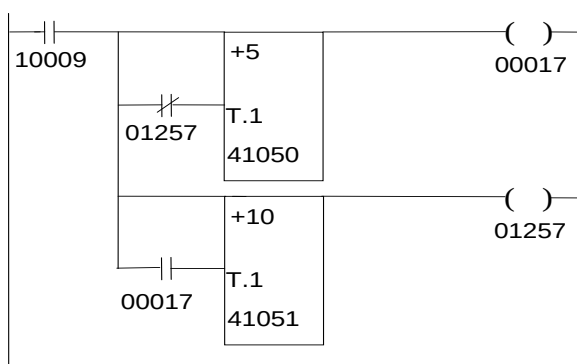
For the T1 timer the Accumulator value is updated when the function encounters a new second when it is scanned, and not necessarily after exactly 1.000 seconds has elapsed. As a result when N seconds have been set for the Accumulator Limit value the limit will be reached anywhere between (N-1) and (N+ scan time) seconds.

For the T.1 timer the Accumulator value is updated when the function encounters a new second when it is scanned, and not necessarily after exactly 0.1 seconds has elapsed. As a result when N tenths of seconds have been set for the Accumulator Limit value the limit will be reached anywhere between (N-1) and (N+ scan time) tenths of seconds.

For the T.01 timer the Accumulator value is updated when the function encounters a new second when it is scanned, and not necessarily after exactly 0.01 seconds has elapsed. As a result when N hundredths of seconds have been set for the Accumulator Limit value the limit will be reached anywhere between (N-1) and (N+ scan time) hundredths of seconds.

Example

network 1:



In network 1 two Timer functions are configured such that output coil 00017 is ON for 1.0 second ($10 * 0.1$) and OFF for 0.5 second ($5 * 0.1$).

When power is applied to the circuit by closing contacts 10009, output coil 00017 is OFF and Timer 41050 starts to accumulate time. When the accumulator reaches the limit of 5, output coil 00017 turns ON. Timer 41051 is enabled by the NO contacts of output coil 00017 and starts to accumulate time. When the accumulator reaches the limit of 10, output coil 01257 turns ON and resets Timer 41050 by the NC contacts of 01257. Output coil 00017 turns OFF. The process is then repeated until contacts 10009 are opened.

5.73 TOTL – Analog Totalizer

Description

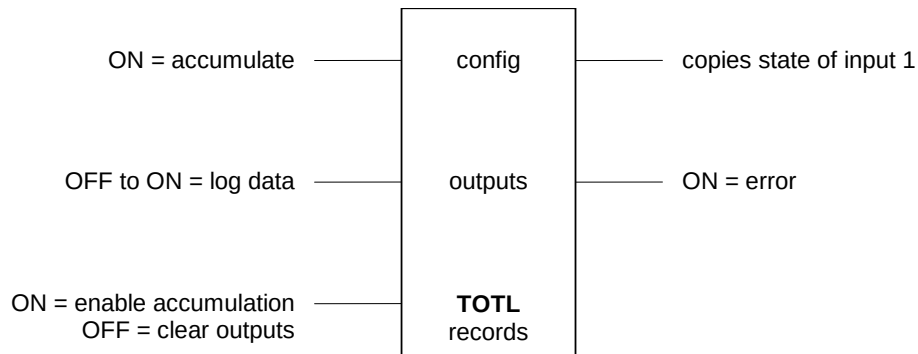
The TOTL function reads a rate input and accumulates (integrates) a total. It is used to measure a flow rate and accumulate a volume, or a similar calculation.

When the **Accumulate** input is ON and it has been longer than the sample interval since the last accumulation, the function reads the value of the input, scales it by time difference from the previous accumulation, and adds it into the total.

When the **Log Data** input goes from OFF to ON, the accumulated total, accumulation time, and the time at the end of the period is saved in the history registers. Older history is pushed down and the oldest record is discarded.

When the **Enable Accumulation** input is ON, accumulation is enabled. When the input is OFF, all accumulators and outputs are set to zero.

The **Error** output is ON if there is an error in the configuration registers.



Function Variables

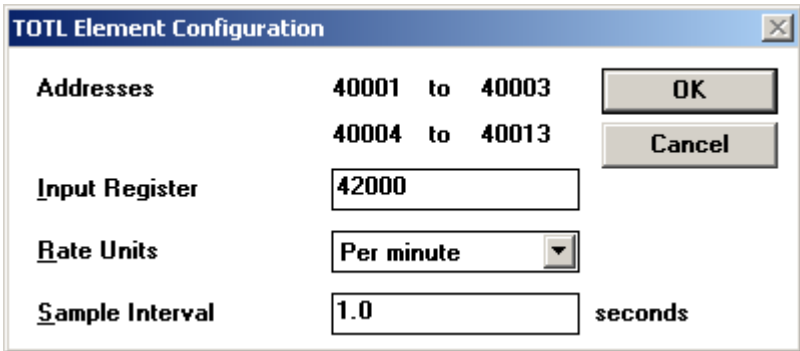
Variable	Valid Types	Description
Config	holding register (4xxxx)	Address of the first register in the configuration block. There are 3 registers in the block at addresses message+0 to message+2. +0 = rate input register +1 = input rate (see table below) +2 = sample interval (tenths of seconds)

Variable	Valid Types	Description
Outputs	holding register (4xxxx)	Address of the first register of the output block. There are 10 to 214 registers in the block at outputs+0 to outputs+213. The number of registers used depends on the records variable. +0 = number of records to follow +1 = status +2,3 = internal: time at last sample +4,5 = period 1: total (float) +6,7 = period 1: end time +8,9 = period 1: accumulation time +10,11= period 2: total (float) +12,13 = period 2: end time +14,15 = period 2: accumulation time +16,17= period 3: total (float) +18,19 = period 3: end time +20,21 = period 3: accumulation time ... +208,209 = period 35: total (float) +210,211 = period 35: end time +212,213 = period 35: accumulation time
Records	Constant	The number of measurement records stored in the output array. The valid values are 1 to 35. This value determines the number of output registers used by the function.

Element Configuration

This element is configured using the TOTL Element Configuration dialog. Highlight the element by moving the cursor over the element and then use the **Element Configuration** command on the **Edit** menu to modify the configuration block.

WARNING: If the controller is initialized, using the **Initialize** command in the **Controller** menu, all I/O database registers used for Element Configuration are set to zero. The application program must be re-loaded to the controller.



The dialog box titled "TOTL Element Configuration" contains the following fields and buttons:

- Addresses:** Two text boxes showing "40001 to 40003" and "40004 to 40013".
- Input Register:** A text box containing "42000".
- Rate Units:** A dropdown menu currently set to "Per minute".
- Sample Interval:** A text box containing "1.0" followed by the label "seconds".
- Buttons:** "OK" and "Cancel" buttons are located on the right side.

The **Input Register** specifies the register address from which the rate input is read. Valid value for this register is any floating point register. The function will use the specified register, and the next

sequential register, as a floating point rate value. The address range depends upon the registers supported in the selected controller type.

The **Rate Units** specifies the units of time for the rate input. It may be one of the following.

per second (holding register = 0)

per minute (holding register = 1)

per hour (holding register = 2)

per day (holding register = 3)

The **Sample Interval** specifies the interval at which the rate input will be sampled. A sample is taken and a calculation performed if the time since the last sample is greater than or equal to the sample interval. The exact time depends on the scan time of the logic program. The valid values are 0.1 to 6553.5 seconds (1 to 65535 tenths of a second). The dialog shows the value in seconds; the register contains an integer value in tenths of seconds.

Note: The period over which readings are taken will vary, but will stay in the range $-1 * (\text{Expected Interval} + \text{Configured Sample Interval} + \text{Maximum time between ladder logic scans}) \leq \text{Actual Sampling Interval} \leq +1 * (\text{Expected Interval} + \text{Configured Sample Interval} + \text{Maximum time between ladder logic scans})$. As a result the total for individual periods may fluctuate despite a constant input.

Output Registers

The output registers store the results of the accumulation and act as internal workspace for the accumulation.

The **Number of Records** register indicates how many sets of total and time registers follow. This value is equal to the *Records* variable. There are six registers in each record.

The **status** register indicates the status of the accumulation. It can have the following values. If the status register is non-zero, the error output is turned ON.

Status	Description
0	no error
1	invalid rate units configuration
2	invalid sample interval

The function block uses the two **internal** registers. The registers are not intended for use in a ladder logic program.

The **Period n: Total** is stored as a floating-point number in two consecutive registers.

The **Period n: End Time** is stored as a 32-bit integer in two consecutive registers. The registers hold the number of seconds since January 1, 1970. This is an unsigned number.

The **Period n: Accumulation** is stored as a 32-bit integer in two consecutive registers. The register holds the number of seconds accumulation occurred in the period. This measures the time the Accumulate input is ON, including times when the rate was zero.

Notes

The accumulated value is a floating-point number. All floating-point numbers are approximations. If the accumulated value grows large, then low rate inputs will have little or no effect on the

accumulated value and the accumulated value will not be accurate. Use the Log Data input to save the accumulated value and start a new accumulation when the accumulated value grows large.

The Log Data input must be triggered at a suitable rate or the accumulator will overflow, and the accumulated value will not be accurate.

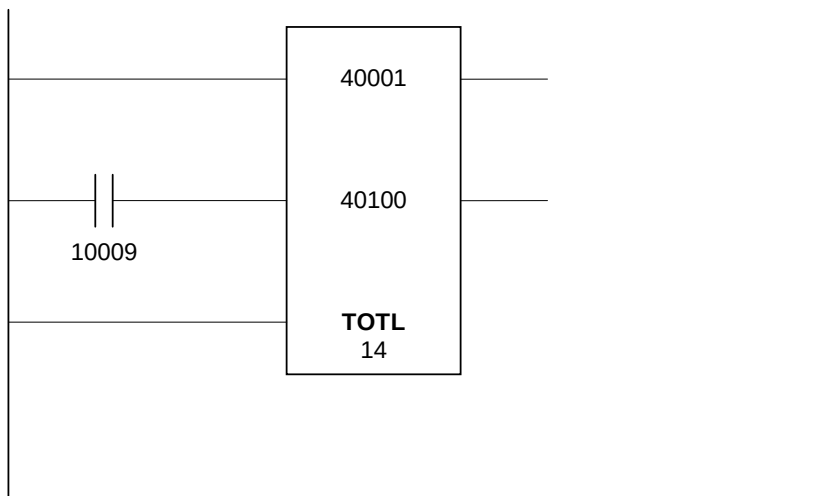
Related Functions

FLOW – Flow Accumulator

Example

In this example a TOTL function is used to read a flow rate from an analog input and accumulate 14 days of data. The analog input measures a flow rate in gallons per minute.

network 1



The TOTL function in network 1 has the **Accumulate** and **Enable Accumulation** inputs connected to the left power rail. When these inputs are continuously powered the TOTL function accumulates totals.

The **Records** variable is set to 14. This means 14 sets of history registers (total, flow time and end of period time) will be logged. In this example the accumulation input is never turned off meaning that once 14 sets of history registers are saved the next time contact 10009 is closed the 14th record is removed and the newest record is added to the history.

Each time contact 10009 is closed it powers the log data input. The accumulated total, accumulation time and the time at the end of the period is saved in the history registers. The contact should be closed by another network once per day for this example.

The element configuration for the TOTL block is as follows

Parameter	Value	Notes
Input Register	42000	The register that contains the rate input. This is a floating point value contained in register 42000 and 42001.
Rate Units	per minute	The value in floating point register 42000 represents the flow rate in units/minute
Sample Interval	10	sample, calculate, and accumulate approximately every 10.0 seconds

5.74 UCTR – Up Counter

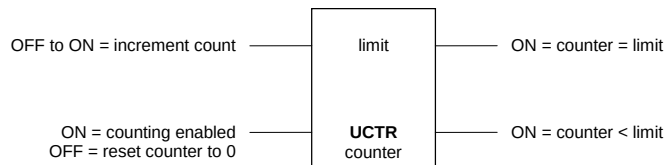
Description

The **UCTR** function block increments (adds one to) the value in the **counter** register when the **increment count** input changes from OFF to ON. The counter stops counting when the **limit** register or constant is reached.

When the **counting enabled** input is ON and the **increment count** changes from OFF to ON the counter value increments by one. When the **counting enabled** input is OFF the counter is reset to zero.

When the **counting enabled** input is ON and the counter equals the limit then the **counter = limit** output is ON.

When the **counting enabled** input is ON and the counter is less than the limit then the **counter < limit** output is ON.



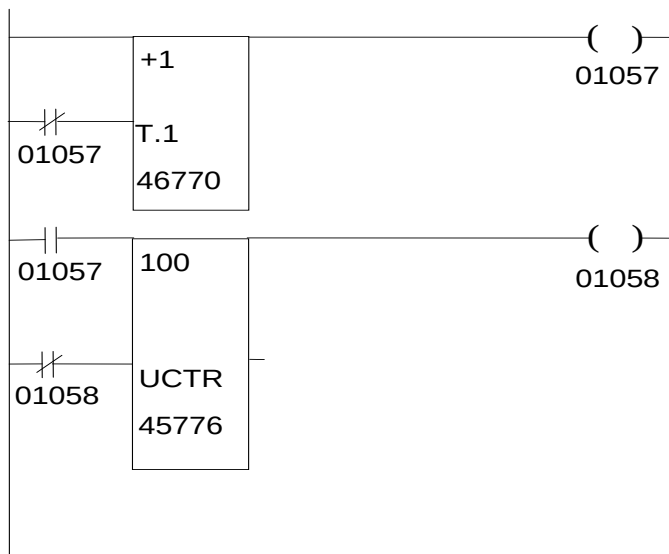
Variable	Valid Types	Description
Limit	constant (1..65535) input register (3xxxx) holding register (4xxxx)	Upper limit for counter
Counter	holding register (4xxxx)	Register holds current counter value

Related Functions

DCTR – Down Counter

Example

network 1:



The UCTR function in network 1 increments each time the timer output is enabled. The timer limit is 0.1 seconds and it will take 10 seconds for the UCTR to increment 100 times.

When the timer limit is reached after 0.1 seconds the output coil 01057 is energized. NC contact 01057 will open resetting the timer and NO contact 01057 will close incrementing the UCTR.

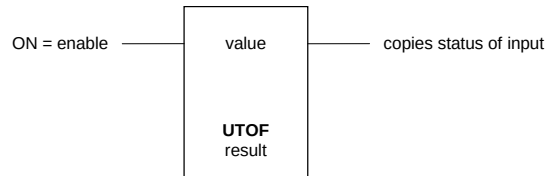
When the UCTR increments to 100 output coil 01058 is energized. NC contacts 01058 will open and the UCTR will reset counter to zero. This will happen every 10 seconds.

5.75 UTOF - Unsigned Integer to Floating-Point

Description

The UTOF element converts an unsigned register or constant into a floating-point number and stores the result in a floating-point holding register.

When the **enable** input is ON, **value** is converted into a floating-point number and stored in the **result** floating-point register. The element output is ON when the input is.



Function Variables

The element has two parameters.

Variable	Valid Types	Description
value	Constant (0...65535) input register (3xxxx) holding register (4xxxx)	Unsigned value to convert
result	2 holding registers (4xxxx)	Converted floating-point value

Notes

Floating-point values are stored in two consecutive I/O database registers. The lower numbered register contains the upper 16 bits of the number. The higher numbered register contains the lower 16 bits of the number.

Floating point numbers can represent positive or negative values in the range -3.402×10^{38} to 3.402×10^{38} .

Related Functions

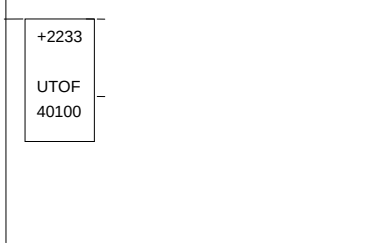
STOF - Signed Integer to Floating-Point

FTOS - Floating-Point to Signed Integer

FTOU - Floating-Point to Unsigned Integer

Example

network 1:

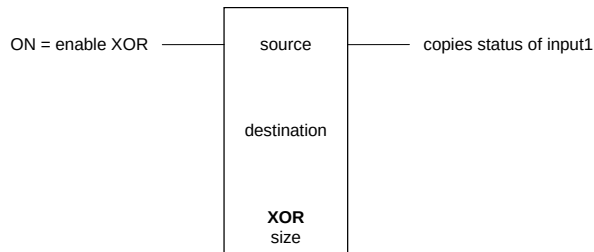


The UTOF function in network 1 converts unsigned constant +2233 to a floating-point number and puts the value into register 40100. In this example the content of floating point-register 40100 (registers 40100 and 40101) is 2233.00.

5.76 XOR – Exclusive Or Block

Description

The **XOR** function block logically EXCLUSIVE-ORs the **source** block of registers with the **destination** block of registers and stores the result in the destination block of registers.



Variable	Valid Types	Description
Source	coil block (0xxxx) status block (1xxxx) input register (3xxxx) holding register (4xxxx)	The first register in the first source block. The address for a coil or status register block is the first register in a group of 16 registers that will be EXCLUSIVE-ORed.
Destination	coil block (0xxxx) holding register (4xxxx)	The first register in the second source block and destination block. The address for a coil register block is the first register in a group of 16 registers that will be EXCLUSIVE-ORed.
Size	constant (1..100)	The number of 16 bit words in the block.

Notes

Exclusive-or accesses 16 bit words. Coil and status register blocks are groups of 16 registers that start with the register specified as the block address. A block size of 2 corresponds to 32 coils, or two holding registers.

Coil and status register blocks must begin at the start of a 16 bit word within the controller memory. Suitable addresses are 00001, 00017, 10001, 10033, etc.

Related Functions

AND – And Block

NOT – Not Block

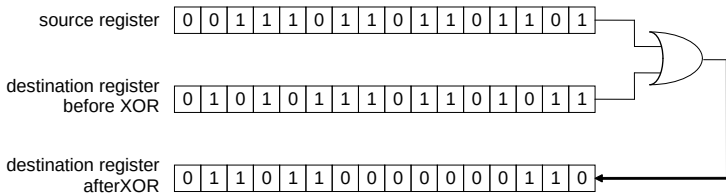
OR – Or Block

Example

network 1:



In this example the source block has a value of 15213₁₀ (11101101101101). The destination register has a value of 22379₁₀ (101011101101011) before the XOR function and a value of 27654₁₀ after the XOR function.



6 TelePACE Register Assignment Reference

A complete description of each I/O Module in the Register Assignment is presented in this reference. The I/O modules are divided into classes that group similar modules together for ease of reference.

All I/O hardware that is used by the controller must be assigned to I/O database registers in order for the I/O data to be used by the ladder program. Ladder logic programs may read data from, or write data to the I/O hardware through user-assigned registers in the I/O database.

Register assignments are stored in the user configured Register Assignment and are downloaded with the ladder logic application program.

The Register Assignment assigns I/O database registers to user-assigned registers using I/O Modules. An I/O Module can refer to an actual I/O hardware module (e.g. 5401 Digital Input Module) or it may refer to a set of controller parameters, such as serial port settings.

Analog Input (AIN)

These modules are used to assign data from physical analog inputs to input registers in the I/O Database. The physical analog inputs are specific 5000 Series I/O modules, generic I/O modules, and internal controller data such as RAM battery voltage and board temperature.

Note: To properly view analog input/output registers, the register type should be set to type signed. Setting the register type to anything else may result in inaccurate readings being displayed/written to these registers.

Analog Output (AOUT)

These modules are used to assign data from the I/O Database to physical analog outputs. The physical analog outputs are specific 5000 Series I/O modules or generic I/O modules.

Note: To properly view analog input/output registers, the register type should be set to type signed. Setting the register type to anything else may result in inaccurate readings being displayed/written to these registers.

Digital Input (DIN)

These modules are used to assign data from physical digital inputs to input registers in the I/O Database. The physical digital inputs are specific 5000 Series I/O modules, generic I/O modules, and controller digital inputs.

Digital Output (DOUT)

These modules are used to assign data from the I/O Database to physical digital outputs. The physical digital outputs are specific 5000 Series I/O modules or generic I/O modules.

Counter Input (CNTR)

These modules are used to assign data from physical counter inputs to input registers in the I/O Database. The physical counter inputs are specific 5000 Series I/O modules and controller counter inputs.

SCADAPack (SCADAPack 5601, SCADAPack 5602, SCADAPack 5604, and SCADAPack AOUT)

These modules are used to assign data to registers in the I/O Database, from physical SCADAPack digital and analog I/O. The physical inputs and outputs are specific SCADAPack I/O modules.

Controller Diagnostic (DIAG)

These modules are used to assign diagnostic data from the controller to input or status registers in the I/O Database. The diagnostic data is used to monitor internal controller data such as controller status code, the force LED, serial port communication status and serial port protocol status.

Controller Configuration (CNFG)

These modules are used to assign data from I/O Database coil and holding registers to controller configuration registers. The configuration data is used to configure controller settings such as clearing protocol and serial counters, real time clock settings, HART protocol interface and PID control blocks.

Default Register Assignment

This provides a default assignment of registers for the selected controller type.

6.1 Register Assignment Specification

The Register Assignment I/O modules described in this manual are formatted as follows.

Header

The module header displays the I/O module name.

Module Description

The module description section describes the function, data origin and data destination of the I/O module.

Register Assignment

The register assignment description is used to describe the information required in the **Add Register Assignment** dialog. The information required for each section of the **Add Register Assignment** dialog is explained in the table shown below.

The column at the left of the table indicates the information that is required for the I/O module in the **Add Register Assignment** dialog. The requirements vary depending on the I/O module.

Module	Module Name	
Address	Physical I/O modules are assigned a unique module address between 0 and 15. No other similar type modules may use this module address. Diagnostic and configuration modules and SCADAPack 5601, 5602 and 5604 I/O modules do not require a physical address.	
Type	Register type of 0xxxx, 1xxxx, 3xxxx or 4xxxx.	
Start	First register of any unused block of consecutive registers.	3xxxx
End	Last register of block	3xxxx + 7
Registers	Number of registers for module.	
Description	Description of module.	
Extended Parameters	Additional parameters, if applicable.	

Register Data

The register data table describes what data each assigned register will contain.

Notes Section

The notes section is used to provide any additional information that may be useful when using the module.

6.2 Register Assignment Example

In this example, the application program requires a 5501 Analog Input module. The analog input data from the 5000 Series 5501 module must be assigned to the I/O Database before the Ladder

Logic application program can access it. The register assignment is used to assign the input data to the I/O database.

Run the TelePACE program.

Select **F**ile from the menu bar. From the drop down **F**ile menu, select **N**ew.

Select **C**ontroller from the menu bar. From the drop down **C**ontroller menu, select **R**egister **A**ssignment.

The Register Assignment is displayed. The table is initially empty when a new file is started.

Selecting the **A**dd button opens the **Add Register Assignment** dialog.

The first I/O module in the I/O module list is displayed and highlighted in the **M**odule window.

Click the left mouse button on the down arrow at the left side of the **M**odule window.

The entire I/O module list is displayed in the drop down window. Use the up and down arrows at the right side of the list to scroll through the list of I/O modules.

Select the AIN 5501 module by clicking the left mouse button on the AIN 5501 module text. When selected the AIN 5501 module is highlighted and the module list is closed.

The AIN 5501 module is described in the **Register Assignment** section of this manual.

The module reference for the AIN module is described in detail below. Each section of the module reference provides information that is required to configure the register assignment. All modules in the *Register Assignment* follow a similar format.

Header

The module header displays the I/O module name. For example, AIN 5501 Module is displayed.

Module Description

The module description section describes the function, data origin and data destination of the I/O module. The description for the AIN 5501 module indicates that data is received from the 5501 Analog Input module. This data is then assigned to eight consecutive input (3xxxx) registers in the I/O database. The I/O database registers used by the AIN 5501 module are continuously updated with data from the 5000 Series 5501 analog input module.

Register Assignment Description

The register assignment description is used to describe the information required in the **Add Register Assignment** dialog. The information required for each section of the **Add Register Assignment** dialog is explained in the table. A sample is shown below.

Module	AIN 5501 Module	
Address	This module is assigned a unique module address between 0 and 15. No other AIN-type module may use this module address.	
Type	Input Register	
Start	First register of any unused block of 8 consecutive input registers.	3xxxx
End	Last register of block	3xxxx + 7
Registers	8 input registers.	

Description	None
Extended Parameters	None

The column at the left of the table indicates the information that is required for the I/O module in the **Add Register Assignment** dialog. The requirements vary depending on the I/O module.

Module The I/O module highlighted in the **Module** window.

Address The physical address of the 5000 Series 5501 analog input module. This address is set equal to the address of 5501 analog input module. See the *5501 Analog Input Module User Manual*.

Type The I/O database register type that the I/O module will assign data to. The I/O database register types are:

Coil Register **0xxxx**

Status Register **1xxxx**

Input Register **3xxxx**

Holding Register **4xxxx**

The AIN 5501 module will assign data to input registers (3xxxx).

Start The I/O database register address where the AIN 5501 module begins the register assignment. For the AIN 5501 module, this is the first of eight consecutive input registers.

End The I/O database register where the AIN 5501 module ends the register assignment. This value is automatically set to **Start + 8** by the AIN 5501 module.

Registers Number of I/O database registers that the AIN 5501 module assigns. This is automatically set to 8 for the AIN 5501 module.

Description The register type description for I/O modules that assign multiple types of registers. The AIN 5501 module does not have an entry for this field.

Extended Parameters None.

Register Data

The register data table describes what data each assigned register will contain. In the case of the AIN 5501 Module, the data will be the physical analog input channels from the 5000 Series 5501 module.

Notes Section

The notes section is used to provide any additional information that may be useful when using this module.

6.3 Analog Input I/O Modules

Analog input modules are used to assign data from physical analog inputs to input registers in the I/O Database. The physical analog inputs are specific 5000 Series I/O modules, generic I/O modules, and internal controller data such as RAM battery voltage and board temperature.

Analog input I/O modules may assign data to any input registers in the I/O database that are not being used by another Analog Input or Counter I/O module. There are 1024 I/O database input registers available. These input registers are numbered 30001 - 31024. Input registers are referred to as 3xxxx registers throughout this manual.

All I/O database input registers that are not assigned to any other I/O modules may be used as general purpose input registers in a ladder program. The I/O modules available are described in the following pages.

Note: To properly view analog input/output registers, the register type should be set to type signed. Setting the register type to anything else may result in inaccurate readings being displayed/written to these registers.

6.3.1 AIN Controller RAM Battery V

Description

The module, **AIN Controller RAM battery V**, reads the RAM backup battery voltage of the SCADAPack controllers. This module reads the input voltage for the SCADASense programmable controllers.(SCADASense 4202 DR, 4202 DS, 4203 DR, and 4203 DS) The voltage value is assigned to one input register. The input register is updated continuously with voltage data read from the RAM battery.

Register Assignment

Module	AIN Controller RAM battery V	
Address	No physical address is required.	
Type	Input Register	
Start	Any unused input register.	3xxxx
End	Same as start register.	
Registers	1 input register	
Description	None	
Extended Parameters	None	

Register Data

Register Assignment	Assignment to Module Hardware
1 input register	RAM battery voltage (0-5000 millivolts for SCADAPack controllers). Input power voltage (0-32767 millivolts for SCADASense Programmable Controllers).

Notes

One input register is always assigned to the I/O Database when this module is used.

The 3.6V lithium battery will return a typical value of 3600 or 3700. A reading less than 3000 (3.0V) indicates that the lithium battery requires replacement. The RAM battery voltage resolution is 100 millivolts.

6.3.2 AIN Controller temperature

Description

The module, **AIN Controller temperature**, reads the circuit board temperature of the 5203 or 5204 controller. The temperature data is assigned to two consecutive input registers. The input registers are updated continuously with data read from the controller.

Register Assignment

Module	AIN Controller temperature	
Address	No physical address is required.	
Type	Input Register	
Start	First register of any unused block of 2 consecutive input registers.	3xxxx
End	Last register of block	3xxxx + 1
Registers	2 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Register Assignment	Assignment to Module Hardware
Start Register	Board temperature (degrees C)
Start Register + 1	Board temperature (degrees F)

Notes

Two input registers are always assigned to the I/O Database when this module is used.

The temperature sensor returns a value in the range -40°C to 75°C or -40°F to 167°F. Temperatures outside this range cannot be measured.

Refer to the *System Manual* for further information on the circuit board temperature.

6.3.3 AIN 5501 Module

Description

The **AIN 5501 module** provides eight analog inputs, from the 5501 analog input module, for each module address. Data from each module is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Register Assignment

Module	AIN 5501 module	
Address	This module is assigned a unique module address between 0 and 15. No other AIN-type module may use this module address.	
Type	Input Register	
Start	First register of any unused block of 8 consecutive input registers.	3xxx
End	Last register of block	3xxx + 7
Registers	8 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Notes

Eight input registers are always assigned to the I/O Database when this module is used. Refer to the *5501 Analog Input Module User Manual* for further information.

6.3.4 AIN 5502 Module

Description

The **AIN 5502 module** provides eight analog inputs, from the 5502 analog input module, for each module address. Data from each module is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Register Assignment

Module	AIN 5502 module	
Address	This module is assigned a unique module address between 0 and 15. No other AIN-type module may use this module address.	
Type	Input Register	
Start	First register of any unused block of 8 consecutive input registers.	3xxx
End	Last register of block	3xxx + 7
Registers	8 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Notes

Eight input registers are always assigned to the I/O Database when this module is used. Refer to the *5502 Analog Input Module User Manual* for further information.

6.3.5 AIN 5503 Module

Description

The **AIN 5503 module** provides four analog inputs, from the 5503 RTD analog input module, for each module address. Data is assigned to four consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Register Assignment

Module	AIN 5503 module	
Address	This module is assigned a unique module address between 0 and 15. No other AIN-type module may use this module address.	
Type	Input Register	
Start	First register of any unused block of 4 consecutive input registers.	3xxxx
End	Last register of block	3xxxx + 3
Registers	4 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3

Notes

Four input registers are always assigned to the I/O Database when this module is used.

Refer to the *5503 RTD Analog Input Module User Manual* for further information.

6.3.6 AIN 5504 Module

Description

The **AIN 5504 module** provides eight analog inputs, from the Thermocouple analog input module, for each module address. Data is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Register Assignment

Module	AIN 5504 module	
Address	This module is assigned a unique module address between 0 and 15. No other AIN-type module may use this module address.	
Type	Input Register	
Start	First register of any unused block of 8 consecutive input registers.	3xxxx
End	Last register of block	3xxxx + 7
Registers	8 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Notes

Eight input registers are always assigned to the I/O Database when this module is used.

Refer to the *5504 Thermocouple Analog Input Module User Manual* for further information.

6.3.7 AIN 5505 Module

Description

The **AIN 5505 module** provides four RTD analog input points from a 5505 RTD module. A maximum of sixteen 5505 RTD modules may be installed on the I/O bus. The 5505 RTD module can operate in Native mode or in 5503 Emulation Mode. The operating mode is determined via a DIP switch on the 5505 RTD module. Refer to the ***Control Microsystems Hardware Manual*** for details on the DIP switch settings for the module.

When using the 5505 RTD module in 5503 Emulation mode the AIN 5503 module is used in the register assignment. See the

AIN 5503 Module section if the module is to be used in 5503 Emulation mode.

The 5505 RTD module native mode provides enhanced capability over the 5503 emulation mode. The module operates in native mode if the 5503 Emulation DIP switch is open. This mode is recommended for all new installations.

- Each input is individually configurable for resistance measurement or RTD temperature measurement
- Each RTD input is configurable to return the measured temperature in degrees Celsius, Kelvin, or Fahrenheit.
- All inputs have a common configurable filter rate that can be used to dampen process variations or noise.
- The module returns diagnostic and status information for each input, such as RTD type (3 or 4 wire), open RTD annunciation, and RTD out of range annunciation.
- Data is returned as an IEEE 32 bit, single precision floating point number that requires no additional scaling.

RTD data is assigned to eight consecutive input (3xxxx) registers, two for each floating point RTD value. The input registers are updated continuously with data read from the analog inputs.

The 5505 RTD module provides 16 internal digital input (1xxxx) registers that return status and diagnostic information about the RTDs. The digital input registers are updated continuously with data read from the digital inputs.

Register Assignment

The register assignment for the AIN 5505 module is used to configure the registers used for the module data and the Extended Parameters available for the module. To open the register assignment dialog for the AIN 5505 module:

- Select **Register Assignment** from the **Controller** menu in TelePACE.
- In the **Register Assignment** window select the **Add** button.
- In the **Add Register Assignment** dialog select the **AIN 5505 module** from the list presented in the **Module** window.

The register assignment dialog for the **AIN 5505 module** is shown below.

The dialog box titled "Add Register Assignment" contains the following fields and sections:

- Module:** A dropdown menu showing "AIN 5505 module".
- Address:** A dropdown menu showing "0".
- Buttons:** "OK" and "Cancel" buttons are located to the right of the Module and Address fields.
- Register Table:** A table with columns: Type, Start, End, Registers, and Description.

Type	Start	End	Registers	Description
1xxxx	10001	10016	16	digital inputs
3xxxx	30001	30008	8	analog inputs
- Extended Parameters:** A section with five dropdown menus:
 - AIN 0 Type: deg C
 - AIN 1 Type: deg C
 - AIN 2 Type: deg C
 - AIN 3 Type: deg C
 - AIN Filter: 0.5 s

The **Module** selection displays the name of the 5505 RTD module used for the register assignment.

The **Address** selection displays the module address of the physical 5505 RTD module. The address on the module is selected via dip switches on the module. A maximum of sixteen AIN type modules may be added to a system.

The register type **1xxxx** defines the register range for the digital input status data. Sixteen sequential registers are needed for the AIN 5505 module status data. Any unused block of sixteen registers may be used. See the **Status Register Data** section below for an explanation of the digital input status data.

The register type **3xxxx** defines the register range for the analog input data. Eight sequential registers are needed for the AIN 5505 module analog data. Any unused block of eight registers may be used. See the **Input Register Data** section below for an explanation of the analog input.

The **Extended Parameters** grouping defines the type of data returned from the 5505 RTD module, for each RTD input, and the filtering used by all RTD inputs.

The **Type** selection for each input (AIN 0, 1, 2 and 3) defines the measurement type. The data in the input registers assigned to the input is set to the type selected.

- The **deg C** selection sets the input point measurement type to degrees Celsius.
- The **deg F** selection sets the input point data measurement to degrees Fahrenheit.
- The **deg K** selection sets the input point data measurement to degrees Kelvin.
- The **ohms** selection sets the input point data measurement as resistance measurement in ohms.

The **AIN Filter** selection set the input filter rate for all RTD inputs. The filter rate is used to dampen process variations or noise.

- The **0.5s** selection sets the filter rate to 0.5 seconds.
- The **1s** selection sets the filter rate to 1 second.
- The **2s** selection sets the filter rate to 2 seconds.
- The **4s** selection sets the filter rate to 4 seconds.

Status Register Data

Status register data is assigned to sixteen consecutive status (1xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment and continues for sixteen registers.

Registers	Assignment to Module Hardware
Start Register	OFF = channel 0 RTD is good ON = channel 0 RTD is open or PWR input is off
Start Register + 1	OFF = channel 0 data in range ON = channel 0 data is out of range
Start Register + 2	OFF = channel 0 RTD is using 3-wire measurement ON = channel 0 RTD is using 4-wire measurement
Start Register + 3	Reserved for future use
Start Register + 4	OFF = channel 1 RTD is good ON = channel 1 RTD is open or PWR input is off
Start Register + 5	OFF = channel 1 data in range ON = channel 1 data is out of range
Start Register + 6	OFF = channel 1 RTD is using 3-wire measurement ON = channel 1 RTD is using 4-wire measurement
Start Register + 7	Reserved for future use
Start Register + 8	OFF = channel 2 RTD is good ON = channel 2 RTD is open or PWR input is off
Start Register + 9	OFF = channel 2 data in range ON = channel 2 data is out of range
Start Register + 10	OFF = channel 2 RTD is using 3-wire measurement ON = channel 2 RTD is using 4-wire measurement
Start Register + 11	Reserved for future use
Start Register + 12	OFF = channel 3 RTD is good

Registers	Assignment to Module Hardware
	ON = channel 3 RTD is open or PWR input is off
Start Register + 13	OFF = channel 3 data in range ON = channel 3 data is out of range
Start Register + 14	OFF = channel 3 RTD is using 3-wire measurement ON = channel 3 RTD is using 4-wire measurement
Start Register + 15	Reserved for future use

Input Register Data

Input register data is assigned to eight consecutive input (3xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment and continues for eight registers. Note that each input point uses two input registers formatted in IEEE 754 floating point format.

Registers	Assignment to Module Hardware
Start Register and Start Register + 1 (in IEEE 754 floating point format)	Analog input 0
Start Register + 2 and Start Register + 3 (in IEEE 754 floating point format)	Analog input 1
Start Register + 4 and Start Register + 5 (in IEEE 754 floating point format)	Analog input 2
Start Register + 6 and Start Register + 7 (in IEEE 754 floating point format)	Analog input 3

Notes

Refer to the *5505 RTD Analog Input Module User Manual* for further information.

6.3.8 AIN 5506 Module

Description

The **AIN 5506 module** provides eight analog inputs from a 5506 Analog Input module. A maximum of sixteen 5506 analog input modules may be installed on the I/O bus. The 5506 Analog Input module can operate in Native mode or in 5501 Emulation Mode. The operating mode is determined via a DIP switch on the 5506 Analog Input. Refer to the ***Control Microsystems Hardware Manual*** for details on the DIP switch settings for the module.

When using the 5506 Analog Input module in 5501 Emulation mode the AIN 5501 module is used in the register assignment. See the

AIN 5501 Module section if the module is to be used in 5501 Emulation mode.

5506 native mode provides enhanced capability over the 5501 emulation mode. The module operates in native mode if the 5501 Emulation DIP switch is open.

- Each input is individually configurable for 0-5V, 1-5V, 0-20mA, or 4-20mA operation.
- Converted analog input data is returned as a signed 15-bit value, providing 8 times more resolution than in 5501 mode. Negative values are possible, for example if a 4-20mA is open loop.
- The module returns status information for each analog input indicating if the analog input is in or out of range for the defined signal type.
- All inputs have a configurable filter rate.
- The input-scanning rate is software configurable to 50 or 60 hertz.

Analog input data is assigned to eight consecutive input (3xxxx) registers. The input registers are updated continuously with data read from the analog inputs.

The 5506 Analog Input module provides 8 internal digital input (1xxxx) registers that return status information about the analog inputs. The digital input registers are updated continuously with data read from the digital inputs.

Register Assignment

The register assignment for the AIN 5506 module is used to configure the registers used for the module data and the Extended Parameters available for the module. To open the register assignment dialog for the AIN 5505 module:

- Select **Register Assignment** from the **Controller** menu in TelePACE.
- In the **Register Assignment** window select the **Add** button.
- In the **Add Register Assignment** dialog select the **AIN 5506 module** from the list presented in the **Module** window.

The register assignment dialog for the **AIN 5506 module** is shown below.

Type	Start	End	Registers	Description
1xxxx	10001	10008	8	digital inputs
3xxxx	30001	30008	8	analog inputs

Extended Parameters	
AIN 0 Type	0-5 V
AIN 1 Type	1-5 V
AIN 2 Type	0-20 mA
AIN 3 Type	4-20 mA
AIN 4 Type	0-5 V
AIN 5 Type	0-5 V
AIN 6 Type	0-5 V
AIN 7 Type	0-5 V
AIN Filter	3 Hz
Scan Frequency	60 Hz

The **Module** selection displays the name of the 5506 Analog Input module used for the register assignment.

The **Address** selection displays the module address of the physical 5506 Analog Input module. The address on the module is selected via dip switches on the module. A maximum of sixteen AIN type modules may be added to a system.

The register type **1xxxx** defines the register range for the digital input status data. Eight sequential registers are needed for the AIN 5506 module status data. Any unused block of eight registers may be used. See the **Status Register Data** section below for an explanation of the digital input status data.

The register type **3xxxx** defines the register range for the analog input data. Eight sequential registers are needed for the AIN 5506 module analog data. Any unused block of eight registers may be used. See the **Input Register Data** section below for an explanation of the analog input.

The **Extended Parameters** grouping defines the type of data returned from the 5506 AIN module, filtering used by all inputs and the scan frequency.

The **Type** selection for each input (AIN 0, 1, 2, 3, 4, 5, 6, and 7) defines the input measurement type for the analog input.

- The **0-5V** selection sets the input type to measure 0 to 5V input signals.
- The **1-5V** selection sets the input type to measure 1 to 5V input signals.
- The **0-20 mA** selection sets the input type to measure 0 to 20mA input signals.
- The **4-20 mA** selection sets the input type to measure 4 to 20mA input signals.

The **AIN Filter** selection set the input filter for all analog inputs. Filtering is used to dampen process variations or noise.

- The **3 Hz** filter selection sets the response time to 155ms at 60Hz and 185ms at 50Hz.
- The **6 Hz** filter selection sets response time to 85ms at 60Hz and 85ms at 50Hz.
- The **11 Hz** filter selection sets response time to 45ms at 60Hz and 55ms at 50Hz.
- The **30 Hz** filter selection sets response time to 30ms at 60Hz and 30ms at 50Hz.

The **Scan Frequency** selection set the input scan rate for all analog inputs. The scan rate selection is not critical but AC noise rejection is improved at the correct frequency. If the module is used in a DC environment, the 60 Hz setting will yield slightly faster response time.

- The **60 Hz** selection synchronizes the input scanning to 60Hz.
- The **50 Hz** selection synchronizes the input scanning to 50Hz.

Status Register Data

Status register data is assigned to sixteen consecutive status (1xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment and continues for sixteen registers.

Registers	Assignment to Module Hardware
Start Register	OFF = channel 0 is OK ON = channel 0 is over or under range
Start Register + 1	OFF = channel 1 is OK ON = channel 1 is over or under range
Start Register + 2	OFF = channel 2 is OK ON = channel 2 is over or under range
Start Register + 3	OFF = channel 3 is OK ON = channel 3 is over or under range
Start Register + 4	OFF = channel 4 is OK ON = channel 4 is over or under range
Start Register + 5	OFF = channel 5 is OK ON = channel 5 is over or under range
Start Register + 6	OFF = channel 6 is OK ON = channel 6 is over or under range
Start Register + 7	OFF = channel 7 is OK ON = channel 7 is over or under range

Input Register Data

Input register data is assigned to eight consecutive input (3xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment and continues for eight registers.

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Notes

Refer to the *5506 Analog Input Module User Manual* for further information.

6.3.9 AIN 5521 Module

Description

The **AIN 5521 module** provides eight analog inputs, from the 5521 Potentiometer analog input module, for each module used. Data is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Register Assignment

Module	AIN 5521 module	
Address	This module is assigned a unique module address between 0 and 7. No other AIN-type module may use this module address.	
Type	Input Register	
Start	First register of any unused block of 8 consecutive input registers.	3xxx
End	Last register of block	3xxx + 7
Registers	8 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Notes

Eight input registers are always assigned to the I/O Database when this module is used.

Refer to the *5521 Potentiometer Analog Input Module User Manual* for further information.

6.3.10 AIN Generic 8 Point Module

Description

The **AIN Generic 8 point module** provides eight analog inputs. Data is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs. The **AIN Generic 8 point module** may be used in place of any other 8 point AIN-type module.

The **AIN Generic 8 point module** type is a useful selection early on in the system design stage before the final selection of the specific AIN module type is known.

Register Assignment

Module	AIN Generic 8 Point Module	
Address	This module is assigned a unique module address between 0 and 15. No other AIN-type module may use this module address.	
Type	Input Register	
Start	First register of any unused block of 8 consecutive input registers.	3xxxx
End	Last register of block	3xxxx + 7
Registers	8 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Notes

Eight input registers are always assigned to the I/O Database when this module is used.

6.4 Analog Output I/O Modules

Analog output modules are used to assign data from the I/O Database to physical analog outputs. The physical analog outputs are specific 5000 Series I/O modules or generic I/O modules.

Analog output I/O modules may assign data to any holding registers in the I/O database that are not being used by another Analog output I/O module. There are 9999 holding registers available. These holding registers are numbered 40001 - 49999. Holding registers are referred to as 4xxxx registers throughout this manual.

All holding registers that are not assigned to any other I/O modules may be used as general purpose holding registers in a ladder program. The I/O modules available are described in the following pages.

Note: To properly view analog input/output registers, the register type should be set to type signed. Setting the register type to anything else may result in inaccurate readings being displayed/written to these registers.

6.4.1 AOOUT 5301 Module

Description

The **AOOUT 5301 module** provides two analog outputs, to the 5000 Series 5301 Analog output module, for each module used. Data is assigned from two consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers.

Register Assignment

Module	AOOUT 5301 module	
Address	This module is assigned a unique module address between 0 and 15. No other AOOUT-type module may use this module address.	
Type	Holding Register	
Start	First register of any unused block of 2 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 1
Registers	2 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog output 0
Start Register + 1	Analog output 1

Notes

Two holding registers are always assigned to the I/O Database when this module is used.

Refer to the *5301 Analog Output Module User Manual* for further information.

6.4.2 AOOUT 5302 Module

Description

The **AOOUT 5302 module** provides four analog outputs, to the 5000 Series 5302 Analog output module, for each module used. Data is assigned from four consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers.

Register Assignment

Module	AOOUT 5302 module	
Address	This module is assigned a unique module address between 0 and 15. No other AOOUT-type module may use this module address.	
Type	Holding Register	

Start	First register of any unused block of 4 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 3
Registers	4 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog output 0
Start Register + 1	Analog output 1
Start Register + 2	Analog output 2
Start Register + 3	Analog output 3

Notes

Four holding registers are always assigned to the I/O Database when this module is used.

Refer to the *5302 Analog Output Module User Manual* for further information.

6.4.3 AOUT 5304 Module

Description

The **AOUT 5304 module** provides four analog outputs, to the 5000 Series 5304 Analog output module, for each module used. Data is assigned from four consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers.

Register Assignment

Module	AOUT 5304 module	
Address	This module is assigned a unique module address between 0 and 15. No other AOUT-type module may use this module address.	
Type	Holding Register	
Start	First register of any unused block of 4 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 3
Registers	4 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog output 0
Start Register + 1	Analog output 1
Start Register + 2	Analog output 2
Start Register + 3	Analog output 3

Notes

Four holding registers are always assigned to the I/O Database when this module is used.

Refer to the *5304 Analog Output Module User Manual* for further information.

6.4.4 AOUT Generic 2 Point Module

Description

The **AOUT Generic 2 point module** provides two analog outputs. Data is assigned from two consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers. The **AOUT Generic 2 point module** may be used in place of any other 2 point AOUT-type module.

The **AOUT Generic 2 point module** type is a useful selection early on in the system design stage before the final selection of the specific AOUT module type is known.

Register Assignment

Module	AOUT Generic 2 point module	
Address	This module is assigned a unique module address between 0 and 15. No other AOUT-type module may use this module address.	
Type	Holding Register	
Start	First register of any unused block of 2 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 1
Registers	2 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog output 0
Start Register + 1	Analog output 1

Notes

Two holding registers are always assigned to the I/O Database when this module is used.

6.4.5 AOUP Generic 4 Point Module

Description

The **AOUP Generic 4 point module** provides four analog outputs. Data is assigned from four consecutive holding registers. The analog outputs are updated continuously with data read from the holding registers. The **AOUP Generic 4 point module** may be used in place of any other 4 point AOUP-type module.

The **AOUP Generic 4 point module** type is a useful selection early on in the system design stage before the final selection of the specific AOUP module type is known.

Register Assignment

Module	AOUP Generic 4 point module	
Address	This module is assigned a unique module address between 0 and 15. No other AOUP-type module may use this module address.	
Type	Holding Register	
Start	First register of any unused block of 4 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 3
Registers	4 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Analog output 0
Start Register + 1	Analog output 1
Start Register + 2	Analog output 2
Start Register + 3	Analog output 3

Notes

Four holding registers are always assigned to the I/O Database when this module is used.

6.5 Configuration I/O Modules

Configuration modules are used to assign data from the I/O Database coil and holding registers to controller configuration settings. The configuration data is used to configure controller settings such as clearing protocol and serial counters, real time clock settings, HART protocol interface and PID control blocks.

Configuration I/O modules assign data to I/O database coil or holding registers, depending on the type of data. Configuration I/O modules may assign data to any coil or holding registers in the I/O database that are not being used by another I/O module.

All I/O database coil or holding registers that are not assigned to any other I/O modules may be used as general purpose coil or holding registers in a ladder program. The configuration modules available are described in the following pages.

6.5.1 CNFG 5904 HART Interface Module

Description

The module, **CNFG 5904 HART Interface** provides control of 5904 HART Interface modules. When configured the module maintains communication with a 5904 interface. Up to four 5904 interfaces may be attached to a controller.

The interface settings are assigned from four consecutive holding registers. The registers are initialized with the current value of the module parameters when the controller is reset, and when the Register Assignment is downloaded to the controller. Thereafter the data from all four holding registers are written to the module parameters, whenever any of the registers change.

Register Assignment

Module	CNFG 5904 HART Interface	
Address	0 to 3	
Type	Holding Register	
Start	The first register of any unused block of 4 consecutive holding registers.	4xxxx
End	The last register of the block	4xxxx + 3
Registers	4 holding registers	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Attempts: the number of times each HART command will be sent. Valid values are 1 to 4. Default : = 3
Start Register + 1	Preambles: number of preambles to send if fixed preambles selected. Valid values are 2 to 15. This value must be set as it is used by link initialization. Default : = 15
Start Register + 2	Auto-preamble control: 0 = use fixed number of preambles 1 = use number of preambles requested by device except for link initialization Default : = 1
Start Register + 3	Device type 0 = secondary master device 1 = primary master device (recommended) Default : = 1

Notes

This module provides the basis for communication with HART devices connected to the 5904 interface module. It does not send commands to the HART devices. Use the Ladder Logic HART element or the C Tools HART API functions to send commands to the HART devices.

The number of preambles must be set even if the auto-preamble control is selected. The preambles value will be used for the link initialization command which determines the number of preambles requested by the device.

In most sensor networks, the controller is the primary master device. A hand-held programmer is typically a secondary master device. Set the device type to primary unless another HART primary master device is connected to the network.

The settings can be saved to EEPROM with the *CNFG Save Settings to EEPROM* module.

6.5.2 CNFG Clear Protocol Counters

Description

The module, **CNFG Clear protocol counters**, provides a way of clearing the protocol event counters for a specific serial port. The protocol event counters for the serial port are cleared when the assigned coil register is set to 1. Once the counters are cleared, the assigned coil register is automatically reset to zero.

Register Assignment

Module	CNFG Clear protocol counters	
Address	com1, com2, com3, or com4.	
Type	Coil Register	
Start	Any unused coil register.	0xxxx
End	Same as start register.	
Registers	1 coil register.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Clear protocol event counters.

Notes

Refer to the module, **DIAG Serial port protocol status**, for a description of the individual counters.

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

6.5.3 CNFG Clear Serial Port Counters

Description

The module, **CNFG Clear serial port counters**, provides a way of clearing the communication event counters for a specific serial port. The serial port counters are cleared when the assigned coil register is set to 1. Once the counters are cleared the assigned coil register is automatically reset to 0.

Register Assignment

Module	CNFG Clear serial port counters	
Address	Com1, com2, com3, or com4.	
Type	Coil Register	
Start	Any unused coil register.	0xxxx
End	Same as start register.	
Registers	1 coil register.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Clear communication event counters.

Notes

Refer to the module, **DIAG Serial port comm. status**, for a description of the individual counters.

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

6.5.4 CNFG Device Configuration

Description

The module **CNFG Device Configuration** provides a method for users to assign an ID number to the applications running in a SCADAPack or SCADASense controller. This module is supported on all SCADAPack controllers with 2.44 (or later) firmware; SCADAPack 32 controllers with 1.92 (or later) firmware and SCADAPack 330, SCADAPack 334, SCADAPack 350 or SCADASense 4203 controllers with 1.25 (or later) firmware.

The CNFG Device Configuration register assignment sets the Company ID, Logic Application number and logic Application version. This information is then saved in a group of reserved registers the I/O database. Refer to the **Device Configuration Read Only Registers** section below for details on the Logic Application ID. These registers may be polled by SCADA Host software or other Modbus compatible devices.

Users must add a rung to the logic application that specifies the appropriate information for the Logic Application ID. See the **Example** section below.

Note: For SCADAPack and SCADAPack 32 controllers it is necessary to create a logic application consisting of at least one element. If there is no logic program present then the logic application ID is cleared.

The number of applications and version control are the responsibility of the user.

Register Assignment

Module	CNFG Device Configuration	
Address	Fixed	
Type	Coil Register	
Start	First register of any unused block of 4 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 3
Registers	1 coil register.	
Description	4 output registers	
Extended Parameters	Device information and control	

Register Data

Registers	Assignment to Module Hardware
Start Register	0 = configuration registers operate as normal input registers 1 = configuration registers report device configuration
Start Register + 1	Company ID – contact Control Microsystems to obtain your company ID
Start Register + 2	Logic Application number
Start Register + 3	Logic Application version (major * 100 + minor)

Device Configuration Read Only Registers

The Device configuration is stored in Modbus input (3xxxx series) registers as shown below. The registers are read with standard Modbus commands. These registers cannot be written to. Device configuration registers used fixed addresses. This facilitates identifying the applications in a standard manner.

Note: These registers must not be used for other purposes in a logic or C application. This includes the following uses:

- MSTR block uses registers 39800 to 39999 (in element configuration) as a destination.
- OVER block uses registers 39800 to 39999 as a destination.
- Any register assignment module uses registers 39800 to 39999.

The Device configuration registers can be enabled or disabled by entering a 0 or 1 in the Start Register. They are disabled until enabled by a logic application. This provides compatibility with controllers that have already used these registers for other purposes.

The application IDs are cleared on every controller reset. Applications must run and set the application ID for it to be valid.

The following information is stored in the device configuration. 2 logic application identifiers are provided for compatibility with SCADAPack ES/ER controllers that provide 2 ISaGRAF applications. The second logic application identifier is not used with other controllers. 32 application identifiers are provided to accommodate C applications in SCADAPack 33x/350 controllers.

Register	Description
39800	Controller ID (ASCII value), first byte.
39801	Controller ID (ASCII value), second byte.
39802	Controller ID (ASCII value), third byte.
39803	Controller ID (ASCII value), fourth byte.
39804	Controller ID (ASCII value), fifth byte.
39805	Controller ID (ASCII value), sixth byte.
39806	Controller ID (ASCII value), seventh byte.
39807	Controller ID (ASCII value), eighth byte.
39808	Firmware version (major*100 + minor)
39809	Firmware version build number (if applicable)
39810	Logic Application 1 - Company ID (see below)
39811	Logic Application 1 - Application number (0 to 65535)
39812	Logic Application 1 - Application version (major*100 + minor)
39813	Logic Application 2 - Company ID (see below)
39814	Logic Application 2 - Application number (0 to 65535)
39815	Logic Application 2 - Application version (major*100 + minor)
39816	Number of applications identifiers used (0 to 32) Identifiers are listed sequentially starting with identifier 1. Unused identifiers will return 0.
39817	Application identifier 1 (see format below)
39820	Application identifier 2 (see format below)
39823	Application identifier 3 (see format below)
39826	Application identifier 4 (see format below)
39829	Application identifier 5 (see format below)

Register	Description
39832	Application identifier 6 (see format below)
39835	Application identifier 7 (see format below)
39838	Application identifier 8 (see format below)
39841	Application identifier 9 (see format below)
39844	Application identifier 10 (see format below)
39847	Application identifier 11 (see format below)
39850	Application identifier 12 (see format below)
39853	Application identifier 13 (see format below)
39856	Application identifier 14 (see format below)
39859	Application identifier 15 (see format below)
39862	Application identifier 16 (see format below)
39865	Application identifier 17 (see format below)
39868	Application identifier 18 (see format below)
39871	Application identifier 19 (see format below)
39874	Application identifier 20 (see format below)
39877	Application identifier 21 (see format below)
39880	Application identifier 22 (see format below)
39883	Application identifier 23 (see format below)
39886	Application identifier 24 (see format below)
39889	Application identifier 25 (see format below)
39892	Application identifier 26 (see format below)
39895	Application identifier 27 (see format below)
39898	Application identifier 28 (see format below)
39901	Application identifier 29 (see format below)
39904	Application identifier 30 (see format below)
39907	Application identifier 31 (see format below)
39910	Application identifier 32 (see format below)
39913 to 39999	Reserved for future expansion

Application Identifier

The application identifier is formatted as follows.

Register	Description
Start	Company ID (see below)
Start +1	Application number (0 to 65535)
Start +2	Application version (major*100 + minor)

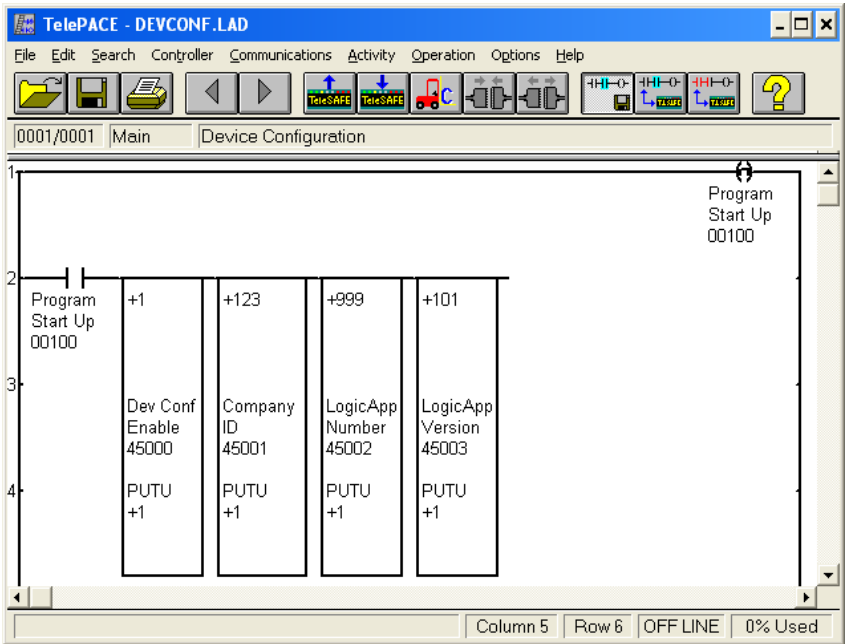
Company Identifier

Control Microsystems maintains a list of company identifiers to ensure the company ID is unique. Contact Control Microsystems for a Company ID. Company ID 0 indicates an identifier is unused.

Company IDs 1 to 100 are reserved for Control Microsystems use.

Example

The following logic enables the Device Configuration registers and sets the Company ID, Logic Application Number and Logic Application Version. The CNFG Device Configuration uses registers 45000 through 45003.



The following is a view of the registers associated with the Device Configuration. Registers 39800 through 39807 contain ASCII values for the controller ID.

Register	Tag	Value	Format
39800		65	Unsigned
39801		48	Unsigned
39802		56	Unsigned
39803		51	Unsigned
39804		57	Unsigned
39805		56	Unsigned
39806		57	Unsigned
39807		0	Unsigned
39808		244	Unsigned
39809		0	Unsigned
39810		123	Unsigned
39811		999	Unsigned
39812		101	Unsigned
45000	Dev ConfEnable	1	Unsigned
45001	Company ID	123	Unsigned
45002	LogicAppNumber	999	Unsigned
45003	LogicAppVersion	101	Unsigned

6.5.5 CNFG DTR Off

Description

The module, **CNFG DTR Off**, provides a way of controlling the DTR signal for a specific serial port. When the assigned coil is energized DTR for the serial port is de-asserted. DTR is constantly turned off when the coil is energized.

When the assigned coil is de-energized DTR for the serial port is asserted. The DTR signal is not constantly asserted. It is asserted on the transition from an energized coil to a de-energized coil. Since DTR is not constantly re-asserted other programs will be able to control the DTR setting. This way other ladder blocks and programs will be able to control the DTR settings.

Register Assignment

Module	CNFG DTR Off	
Address	com1, com2, com3, or com4.	
Type	Coil Register	
Start	Any unused coil register.	0xxxx
End	Same as start register.	
Registers	1 coil register.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Controls DTR signal 0 = DTR asserted 1 = DTR not asserted Default: 0 = DTR asserted

Notes

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

6.5.6 CNFG IP Settings

Description

The module, **CNFG IP settings**, provides control of the IP settings for a specific communication interface. Settings are assigned from thirteen consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all thirteen holding registers is written to the module parameters, whenever any of the registers change.

This module is used by the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers only.

Use this module only when IP settings need to be changed within a Ladders program. It is more convenient to use the *Controller TCP/IP Settings* dialog to download IP settings with the Ladders program.

Note: Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users must ensure the ladder function is only energized for one program scan. If the ladder function is always energized problems with the TCP/IP communication will result.

Register Assignment

Module	CNFG IP settings	
Address	Ethernet1	
Type	Holding Register	
Start	First register of any unused block of 13 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 12
Registers	13 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Byte1 of the IP Address using format: Byte1.Byte2.Byte3.Byte4 Default: 0
Start Register + 1	Byte2 of the IP Address Default: 0
Start Register + 2	Byte3 of the IP Address Default: 0
Start Register + 3	Byte4 of the IP Address Default: 0
Start Register + 4	Byte1 of the Subnet Mask using format: Byte1.Byte2.Byte3.Byte4 Default: 255
Start Register + 5	Byte2 of the Subnet Mask Default: 255

Registers	Assignment to Module Hardware
Start Register + 6	Byte3 of the Subnet Mask Default: 255
Start Register + 7	Byte4 of the Subnet Mask Default: 255
Start Register + 8	Byte1 of the Default Gateway using format Byte1.Byte2.Byte3.Byte4 Default: 0
Start Register + 9	Byte2 of the Default Gateway Default: 0
Start Register + 10	Byte3 of the Default Gateway Default: 0
Start Register + 11	Byte4 of the Default Gateway Default: 0
Start Register + 12	IP Configuration Mode 0 = Default gateway is on the LAN subnet 1 = Default gateway is on the com1 PPP subnet 2 = Default gateway is on the com2 PPP subnet 3 = Default gateway is on the com3 PPP subnet 4 = Default gateway is on the com4 PPP subnet

Notes

Thirteen holding registers are always assigned to the I/O Database when this module is used.

6.5.7 CNFG LED Power Settings

Description

The module, **CNFG LED power settings**, provides control of the power settings for the controller LEDs. The state of the assigned coil register is the default state for the LED power. The time to return to the Default State is contained within the assigned holding register.

The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the module parameters are updated continuously with data read from the assigned registers.

Register Assignment

Module	CNFG LED power settings	
Address	No physical address is required.	
Type	Coil Register	
Start	Any unused coil register.	0xxxx
End	Same as start register.	
Registers	1 coil register.	
Description	default state	
Type	Holding Register	
Start	Any unused holding register	4xxxx
End	Same as start register	
Registers	1 holding register	
Description	time to return to default state	
Extended Parameters	None	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Default state for power to the controller LEDs 0 = off 1 = on Default Value: 1 = on

Holding Register Data

Registers	Assignment to Module Hardware
Start Register	Time in minutes to return to the default state. 1 to 65535 minutes Default Value: 5 minutes

Notes

One holding register and one coil register are always assigned to the I/O Database when this module is used.

6.5.8 CNFG Modbus IP Interface

Description

The module, **CNFG Modbus IP Interface**, provides control of the protocol interface settings used by all Modbus IP protocols on the specified communication interface. Settings are assigned from five consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all five holding registers is written to the module parameters, whenever any of the registers change.

This module is used by the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers only.

Use this module only when Modbus IP protocol interface settings need to be changed within a Ladders program. It is more convenient to use the *Controller Modbus IP Protocols* dialog to download protocol interface settings with the Ladders program.

Note: Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users must ensure the ladder function is only energized for one program scan. If the ladder function is always energized problems with the Modbus IP communication will result.

Register Assignment

Module	CNFG Modbus IP Interface	
Address	Ethernet1	
Type	Holding Register	
Start	First register of any unused block of 5 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 4
Registers	5 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Modbus station number 1 to 255 in standard Modbus 1 to 65534 in extended Modbus Default: 1
Start Register + 1	Modbus store and forward enable 0 = disabled 1 = enabled Default: 0 = disabled

Registers	Assignment to Module Hardware
Start Register + 2	Modbus addressing mode 0 = standard 1 = extended Default: 0 = standard
Start Register + 3	Enron Modbus enable 0 = disabled 1 = enabled Default: 0 = disabled
Start Register + 4	Enron Modbus station number 1 to 255 in standard Modbus 1 to 65534 in extended Modbus Default: 1

Notes

Five holding registers are always assigned to the I/O Database when this module is used.

6.5.9 CNFG Modbus IP Protocols

Description

The module, **CNFG Modbus IP Protocols**, provides control of the settings for a Modbus IP protocol. Settings are assigned from ten consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all ten holding registers is written to the module parameters, whenever any of the registers change.

This module is used by the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers only.

Use this module only when Modbus IP protocol settings need to be changed within a Ladders program. It is more convenient to use the *Controller Modbus IP Protocols* dialog to download protocol settings with the Ladders program.

Note: Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users must ensure the ladder function is only energized for one program scan. If the ladder function is always energized problems with the Modbus IP communication will result.

Register Assignment

Module	CNFG Modbus IP Protocols	
Address	1 = Modbus/TCP, 2 = Modbus RTU over UDP, or 3 = Modbus ASCII over UDP	
Type	Holding Register	
Start	First register of any unused block of 10 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 9
Registers	10 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Protocol Port Number: 1 to 65534
	Modbus/TCP Default: 502
	Modbus RTU over UDP Default: 49152
	Modbus ASCII over UDP Default: 49153

Registers	Assignment to Module Hardware
Start Register + 1 and Start Register + 2	Master Idle Timeout: The length of time, in seconds, that a master connection will wait for the user to send the next command before ending the connection. This allows the slave device to free unused connections while the master application may retain the connection allocation. (32 bit register) 0 = disable timeout and let application close the connection. Default: 10 TCP protocols only. Not used by UDP protocols.
Start Register + 3 and Start Register + 4	Server Idle Timeout: The length of time, in seconds, that a server connection will wait for a message before ending the connection. (32 bit register) 0 = disable timeout and let client close connection. Default: 250 TCP protocols only. Not used by UDP protocols.
Start Register + 5	Protocol Server Enable: 0 = disable server. 1 = enable server. Default: 1
Start Register + 6	Reserved
Start Register + 7	Reserved
Start Register + 8	Reserved
Start Register + 9	Reserved

6.5.10 CNFG Modbus/TCP Settings

Description

The module, **CNFG Modbus/TCP settings**, provides control of the Modbus/TCP protocol settings. Settings are assigned from six consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all six holding registers is written to the module parameters, whenever any of the registers change.

This module is used by the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers only.

Use this module only when Modbus/TCP settings need to be changed within a Ladders program. It is more convenient to use the *Controller Modbus/TCP Settings* dialog to download Modbus/TCP settings with the Ladders program.

Note: Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users must ensure the ladder function is only energized for one program scan. If the ladder function is always energized problems with the Modbus/TCP communication will result.

Register Assignment

Module	CNFG Modbus/TCP settings	
Address	No physical address is required.	
Type	Holding Register	
Start	First register of any unused block of 6 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 5
Registers	6 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Port for Modbus/TCP protocol Default: 502
Start Register + 1 and Start Register + 2	Master Idle Timeout: The length of time, in seconds, that a master connection will wait for the user to send the next command before ending the connection. (32 bit register) 0 = disable timeout and let user close the connection. Default: 10

Registers	Assignment to Module Hardware
Start Register + 3 and Start Register + 4	Server Idle Timeout: The length of time, in seconds, that a server connection will wait for a message before ending the connection. (32 bit register) 0 = disable timeout and let client close connection. Default: 250
Start Register + 5	Maximum number of server connections up to the capacity available (typical capacity is 20 connections) Default: 20

6.5.11 CNFG PID Control Block

Description

The module, **CNFG PID control block**, provides control over the configuration of a PID control block and provides access to the control block parameters. Control block parameters are assigned to 25 consecutive holding registers. The module address refers to the PID control block number.

The holding registers are updated continuously with data read from the module parameters. Whenever a Ladder Logic program or C Application changes any of the registers, only data from the changed register is written to the corresponding module parameter.

Register Assignment

Module	CNFG PID control block	
Address	This module is assigned a unique module address between 0 and 31. No other PID-type module may use this module address.	
Type	Holding Register	
Start	First register of any unused block of 25 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx +24
Registers	25 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware	
Start Register	Process Value	(PV)
Start Register + 1	Error	(ER)
Start Register + 2	Output Quantity	(OP)
Start Register + 3	Status Register	SR)
Start Register + 4	Integrated Error	(IN)
Start Register + 5	Setpoint	(SP)
Start Register + 6	Deadband	(DB)
Start Register + 7	Gain	(GA)
Start Register + 8	Reset Time	(RE)
Start Register + 9	Rate Time	(RA)
Start Register + 10	Full Scale Output Limit	(FS)
Start Register + 11	Zero Scale Output Limit	(ZE)
Start Register + 12	High Alarm Level	(HI)
Start Register + 13	Low Alarm Level	(LO)
Start Register + 14	Control Register	(CR)
Start Register + 15	Input Source	(IP)

Registers	Assignment to Module Hardware
Start Register + 16	Increase Output Address (IO)
Start Register + 17	Decrease Output Address (DO)
Start Register + 18	Input Bias (IB)
Start Register + 19	Output Bias (OB)
Start Register + 20	Inhibit Execution Address (IH)
Start Register + 21	Alarm Output Address (AO)
Start Register + 22	Cascade Setpoint Source (CA)
Start Register + 23	Execution Period
Start Register + 24	Execution Period at Power Up

Notes

Twenty-five holding registers are always assigned to the I/O Database when this module is used.

Refer to the *TelePACE PID Controller Reference Manual* for further information.

6.5.12 CNFG Power Mode

Description

The module, **CNFG Power Mode**, provides control over the power consumption of the controller. The LAN port and USB host port can be individually disabled to conserve power. The controller can also run at a reduced speed to conserve power.

The status registers are updated continuously with the current LAN, controller, and USB host power states. The current settings can be changed by writing to the appropriate holding register.

Register Assignment

Module	CNFG Power Mode	
Address	fixed	
Type	Holding Register	
Start	First register of any unused block of 3 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 2
Registers	3 holding registers	
Description	None	

Holding Register Data

Registers	Assignment to Module Hardware
Start Register	LAN Operation State (0 = LAN Enabled, 1 = LAN Disabled)
Start Register + 1	Controller Power State (0 = Normal Power Mode, 1 = Reduced Power Mode)
Start Register + 2	USB Host Port Power State (0 = USB host port enabled, 1 = USB host port disabled)

Notes

Three holding registers are always assigned to the I/O Database when this module is used.

6.5.13 CNFG Protocol Settings Method 1

Description

The module, **CNFG Protocol settings method 1**, provides control of the protocol settings for a specific serial port. Settings are assigned from three consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all three holding registers is written to the module parameters, whenever any of the registers change.

Use this module only when protocol settings need to be changed within a Ladders program. It is more convenient to use the *Controller Serial Ports Settings* dialog to download protocol settings with the program. Refer to the *Serial Ports* section in this manual for further information on serial port settings.

Note: Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users must ensure the ladder function is only energized for one program scan. If the ladder function is always energized problems with the serial communication will result.

Register Assignment

Module	CNFG Protocol settings	
Address	com1, com2, com3, or com4.	
Type	Holding Register	
Start	First register of any unused block of 3 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 2
Registers	3 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Protocol type 0 = none 1 = Modbus RTU 2 = Modbus ASCII 3 = DF1 Full Duplex BCC 4 = DF1 Full Duplex CRC 5 = DF1 Half Duplex BCC 6 = DF1 Half Duplex CRC 7 = DNP 8 = PPP Default : 1 = Modbus RTU

Registers	Assignment to Module Hardware
Start Register + 1	Protocol station number 1 to 255 in Modbus 0 to 254 in DF1 Default : 1
Start Register + 2	Store and forward enable 0 = disabled 1 = enabled Default : 0 = disabled

Notes

This register assignment module replaces the CNFG Protocol Settings module used in earlier versions of TelePACE.

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 or 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

Refer to the *TeleBUS Protocols User Manual* for further information on protocols.

Three holding registers are always assigned to the I/O Database when this module is used.

6.5.14 CNFG Protocol Settings Method 2

The module, **CNFG Protocol settings method 2**, provides control of the protocol settings for a specific serial port. This module supports the use of extended addressing.

Settings are assigned from four consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all four holding registers is written to the module parameters, whenever any of the registers change.

Use this module only when protocol settings need to be changed within a Ladders program. It is more convenient to use the *Controller Serial Ports Settings* dialog to download protocol settings with the program. Refer to the *Serial Ports* section in this manual for further information on serial port settings.

Note: Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users must ensure the ladder function is only energized for one program scan. If the ladder function is always energized problems with the serial communication will result.

Register Assignment

Module	CNFG Protocol Extended Settings	
Address	com1, com2, com3, or com4.	
Type	Holding Register	
Start	First register of any unused block of 4 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 3
Registers	4 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Protocol type 0 = none 1 = Modbus RTU 2 = Modbus ASCII 3 = DF1 Full Duplex BCC 4 = DF1 Full Duplex CRC 5 = DF1 Half Duplex BCC 6 = DF1 Half Duplex CRC 7 = DNP 8 = PPP Default : 1 = Modbus RTU

Registers	Assignment to Module Hardware
Start Register + 1	Protocol station number 1 to 255 in standard Modbus 1 to 65534 in extended Modbus 0 to 254 in DF1 Default : 1
Start Register + 2	Store and forward enable 0 = disabled 1 = enabled Default : 0 = disabled
Start Register + 3	Protocol addressing mode 0 = standard 1 = extended Default: 0 = standard

Notes

This register assignment module replaces the CNFG Protocol Extended Settings module used in earlier versions of TelePACE.

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

Extended addressing is supported only by the Modbus RTU and Modbus ASCII protocols.

To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

Refer to the *TeleBUS Protocols User Manual* for further information on protocols.

Four holding registers are always assigned to the I/O Database when this module is used.

6.5.15 CNFG Protocol Settings Method 3

Description

The module, **CNFG Protocol Settings Method 3**, provides control of the protocol settings for a specific serial port. This module supports extended addressing and Enron Modbus parameters.

Settings are assigned from six consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all six holding registers is written to the module parameters, whenever any of the registers change.

Use this module only when protocol settings need to be changed within a Ladders program. It is more convenient to use the *Controller Serial Ports Settings* dialog to download protocol settings with the program. Refer to the *Serial Ports* section in this manual for further information on serial port settings.

Note: Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users must ensure the ladder function is only energized for one program scan. If the ladder function is always energized problems with the serial communication will result.

Register Assignment

Module	CNFG Protocol Settings Method3	
Address	com1, com2, com3, or com4.	
Type	Holding Register	
Start	First register of any unused block of 6 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 5
Registers	6 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Protocol type 0 = none 1 = Modbus RTU 2 = Modbus ASCII 3 = DF1 Full Duplex BCC 4 = DF1 Full Duplex CRC 5 = DF1 Half Duplex BCC 6 = DF1 Half Duplex CRC 7 = DNP 8 = PPP Default : 1 = Modbus RTU
Start Register + 1	Protocol station number 1 to 255 in standard Modbus 1 to 65534 in extended Modbus 0 to 254 in DF1 Default : 1
Start Register + 2	Store and forward enable 0 = disabled 1 = enabled Default : 0 = disabled
Start Register + 3	Protocol addressing mode 0 = standard 1 = extended Default: 0 = standard
Start Register + 4	Enron Modbus enable 0 = disabled 1 = enabled Default : 0 = disabled
Start Register + 5	Enron Modbus station number 1 to 255 in standard Modbus 1 to 65534 in extended Modbus Default : 1

Notes

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program. Extended addressing is supported only by the Modbus RTU and Modbus ASCII protocols.

To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

Refer to the *TeleBUS Protocols User Manual* for further information on protocols.

Six holding registers are always assigned to the I/O Database when this module is used.

6.5.16 CNFG Real Time Clock and Alarm

Description

The module, **CNFG Real time clock and alarm**, provides access to the controller real time clock data and alarm settings. Real Time Clock and Alarm settings are assigned from eleven consecutive holding registers. The registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller.

The clock data from the first 7 holding registers is written to the module parameters, whenever any of these registers are changed by a Ladders program or C Application. Otherwise, these holding registers are updated continuously with data read from the real time clock.

Note: Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users must ensure the ladder function is only energized for one program scan. If the ladder function is continuously energized the controller real time clock will not change.

The alarm settings are updated continuously with data read from the last 4 holding registers.

Register Assignment

Module	CNFG Real time clock and alarm	
Address	No physical address is required.	
Type	Holding Register	
Start	First register of any unused block of 11 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx +10
Registers	11 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Real time clock hour 0 to 23
Start Register + 1	Real time clock minute 0 to 59
Start Register + 2	Real time clock second 0 to 59
Start Register + 3	Real time clock year 00 to 99
Start Register + 4	Real time clock month 1 to 12
Start Register + 5	Real time clock day 1 to 31
Start Register + 6	Real time clock day of week 1 to 7, 1=Sunday, 2=Monday...7=Saturday
Start Register + 7	Alarm hour 0 to 23
Start Register + 8	Alarm minute 0 to 59
Start Register + 9	Alarm second 0 to 59

Registers	Assignment to Module Hardware
Start Register + 10	Alarm control 0 = no alarm 1 = absolute time 2 = elapsed time alarm (relative from now) ** See note below.

** The Alarm Control register (Start Register + 10) enables or disables the alarm. A value of 0 in this register disables the alarm. A value of 1 sets an alarm to be triggered in absolute time, based on the values in the preceding three registers. A value of 2 sets the alarm to be triggered at a latter time, starting now.

Note that the alarm settings above and those in the Sleep (SLP) function block will trigger the same interrupt. See the ***Ladder Logic Function Reference*** in this book for details on the SLP function. Using both alarm settings, therefore, in the same program is not recommended.

Additional Notes

The *5203/4 Controllers* have a hardware based real-time clock that independently maintains the time and date for the operating system. The time and date remain accurate during power-off. The calendar automatically handles leap years.

One method for setting the clock is to use the *Edit/Force Register* dialog to write the current time to the appropriate module registers. Leave the force box in the dialog unchecked so that the data is only written, not forced.

All seven registers for the real time clock must be set to valid values for the clock to operate properly. Refer to real time clock specifications in the *Controller System Manual* for further information.

Eleven holding registers are always assigned to the I/O Database when this module is used.

6.5.17 CNFG Save Settings to EEPROM

Description

The module, **CNFG Save settings to EEPROM**, provides a way to save controller settings to EEPROM. Saving to EEPROM is controlled by one assigned coil register. The controller settings are saved when the coil register is set to 1. Once saved the coil register is automatically reset to 0.

Register Assignment

Module	CNFG Save settings to EEPROM	
Address	No physical address is required.	
Type	Coil Register	
Start	Any unused coil register.	0xxxx
End	Same as start register.	
Registers	1 coil register.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Save the following controller settings: Serial port settings Protocol settings Enable store and forward settings LED power settings Mask for wake-up sources HART interface configuration Execution period on power-up for each PID

6.5.18 CNFG Serial Port Settings

Description

The module, **CNFG Serial port settings**, controls the configuration settings for a serial port. Settings are assigned from nine consecutive holding registers. The holding registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the data from all 9 holding registers is written to the module parameters, whenever any of the registers change.

Use this module only when serial port settings need to be changed within a Ladders program. It is more convenient to use the *Controller Serial Ports Settings* dialog to download settings with the program. Refer to the *Serial Ports* section under *Controller Menu*, in this manual, for further information on serial port settings.

Note: Any block transfer Ladder Logic function may be used to write data to the holding registers used for this module. The data in the holding registers is written to the module parameters only when the ladder function is energized. Users must ensure the ladder function is only energized for one program scan. If the ladder function is always energized problems with the serial communication will result.

Register Assignment

Module	CNFG Serial port settings	
Address	Com1, com2, com3, or com4.	
Type	Holding Register	
Start	First register of any unused block of 9 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx +8
Registers	9 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Baud rate 0 = 75 baud 1 = 110 baud 2 = 150 baud 3 = 300 baud 4 = 600 baud 5 = 1200 baud 6 = 2400 baud 7 = 4800 baud 8 = 9600 baud 9 = 19200 baud 10 = 38400 11 = 115200 com3, com4 only 12 = 57600 com3, com4 only Default : 8 = 9600 baud

Start Register + 1	Duplex com1, com2 com3, com4	0 = half duplex 1 = full duplex Default : 1 = full Default : 0 = half
Start Register + 2	Parity	0 = none 1 = even 2 = odd Default : 0 = none
Start Register + 3	Data bits	0 = 7 bits 1 = 8 bits Default : 1 = 8 bits
Start Register + 4	Stop bits	0 = 1 bit 1 = 2 bits Default : 0 = 1 bit
Start Register + 5	Receiver flow control	0 = none 1 = Modbus RTU (default)
Start Register + 6	Transmitter flow control Com1, Com2 Com3, Com4	0 = none (default) 1 = XON/XOFF 0 = none (default) 1 = Ignore CTS
Start Register + 7	Port type	0 = automatic 1 = RS232 3 = RS485 6 = RS232 MODEM 7 = RS232 Collision Avoidance Default : 1 = RS232
Start Register + 8	Serial time-out delay	0 to 65535 (x 0.1 seconds) Default : 600 = 60 seconds

Notes

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

6.5.19 CNFG Store and Forward

Description

The module, **CNFG Store and forward**, controls the store and forward table. Store and Forward table data is assigned from 512 consecutive holding registers. One coil register is assigned to clear the translation table.

This module is used by the SCADAPack controller series only. For SCADAPack 32 controllers use the Store and Forward command in TelePACE.

The holding registers are initialized with the current value of the module parameters on power-up, and when the Register Assignment is downloaded to the controller. Thereafter the table is updated continuously with data read from the holding registers. The table is cleared when the coil register is set to 1. Once cleared the coil register is automatically reset to 0.

Register Assignment

Module	CNFG Store and Forward	
Address	No physical address is required.	
Type	Coil Register	
Start	Any unused coil register.	0xxxx
End	Same as start register.	
Registers	1 coil register.	
Description	clear Store and Forward table	
Type	Holding Register	
Start	First register of any unused block of 512 consecutive holding registers.	4xxxx
End	Last register in block.	4xxxx +511
Registers	512 holding registers	
Description	Translation table	
Extended Parameters	None	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Clear Translation table

Holding Register Data

Registers	Assignment to Module Hardware
Start Register	Translation table entry 0 Port A 0 = com1 1 = com2 2 = com3 3 = com4 Default : 0

Start Register + 1	Translation table entry 0 Station A 0 to 255 standard addressing 0 to 65534 extended addressing 65535 = disable Default : 65535
Start Register + 2	Translation table entry 0 Port B 0 = com1 1 = com2 2 = com3 3 = com4 Default : 0
Start Register + 3	Translation table entry 0 Station B 0 to 255 standard addressing 0 to 65534 extended addressing 65535 = disable Default : 65535
Start Register + 4 to Start Register + 7	Translation table entry 1
Start Register + 8 to Start Register + 11	Translation table entry 2
Start Register + 12 to Start Register + 511	Translation table entries 3 to 127

Notes

Refer to the *TeleBUS Protocols User Manual* for further information on Store and Forward messaging.

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

Store and forward messaging is enabled on the required serial ports using the module **CNFG Protocol settings**.

6.6 Counter I/O Modules

Counter input modules are used to assign data from physical counter inputs to input registers in the I/O Database. The physical counter inputs are specific 5000 Series I/O modules and controller counter inputs.

Counter input modules may assign data to any input registers that are not used by another Analog Input module or Counter input module. There are 1024 I/O database input registers available. These input registers are numbered 30001 - 31024. Input registers are referred to as 3xxxx registers throughout this manual.

All I/O database input registers that are not assigned to any other I/O modules may be used as general purpose input registers in a ladder program. The I/O modules available are described in the following pages.

6.6.1 CNTR Controller Counter Inputs

Description

The module, **CNTR controller counter inputs**, reads the three counter inputs on the Micro16, SCADAPack, SCADAPack Light, SCADAPack, SCADAPack Plus, SCADAPack 32 and SCADAPack 32P controllers. Data is assigned to six consecutive input registers. The input registers are updated continuously with data read from the counters.

Each counter is a 32-bit number, stored in two 16-bit registers. The first register holds the least significant 16 bits of the counter. The second register holds the most significant 16 bits of the counter. To obtain the correct count, both registers must be read with the same protocol read command. Ladder Logic programs can read the registers in any order, provided both registers are examined on the same pass through the program.

The maximum count is 4,294,967,295. Counters roll over to 0 when the maximum count is exceeded.

Register Assignment

Module	CNTR controller counter inputs	
Address	No physical address is required.	
Type	Input Register	
Start	First register of any unused block of 6 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +5
Registers	6 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register and Start Register + 1	Controller counter input 0 (32 bit register)
Start Register + 2 and Start Register + 3	Controller counter input 1 (32 bit register)
Start Register + 4 and Start Register + 5	Controller counter input 2 (32 bit register)

Notes

Refer to the *Hardware User Manual* for further information on controller board counter inputs.

6.6.2 CNTR Controller Interrupt Input

Description

The module, **CNTR controller interrupt input**, reads the interrupt input on the Micro16, SCADAPack, SCADAPack Light, SCADAPack, SCADAPack Plus, SCADAPack 32 and SCADAPack 32P controllers as a counter input. Data is assigned to two consecutive input registers. The input registers are updated continuously with data read from the counter.

The counter is a 32-bit number, stored in two 16-bit registers. The first register holds the least significant 16 bits of the counter. The second register holds the most significant 16 bits of the counter. To obtain the correct count, both registers must be read with the same protocol read command. Ladder Logic programs can read the registers in any order, provided both registers are examined on the same pass through the program.

The maximum count is 4,294,967,295. The counter rolls over to 0 when the maximum count is exceeded.

Register Assignment

Module	CNTR controller interrupt input	
Address	No physical address is required.	
Type	Input Register	
Start	First register of any unused block of 2 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +1
Registers	2 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register and Start Register + 1	Interrupt input counter (32 bit register)

Notes

Refer to the *SCADAPack and Micro16 System Manual* for further information on controller board counter inputs.

6.6.3 CNTR 5410 input module

Description

The module, **CNTR 5410 input module**, provides four counter inputs. Data is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the counters.

Each counter is a 32-bit number, stored in two 16-bit registers. The first register holds the least significant 16 bits of the counter. The second register holds the most significant 16 bits of the counter. To obtain the correct count, both registers must be read with the same Modbus protocol read command. Ladder Logic programs can read the registers in any order, provided both registers are examined on the same pass through the program.

The maximum count is 4,294,967,295. Counters roll back to 0 when the maximum count is exceeded.

Register Assignment

Module	CNTR 5203/4 counter inputs	
Address	This module is assigned a unique module address between 0 and 15. No other CNTR-type module may use this module address.	
Type	Input Register	
Start	First register of any unused block of 8 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +7
Registers	8 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register and Start Register + 1	Counter input 0 (32 bit register)
Start Register + 2 and Start Register + 3	Counter input 1 (32 bit register)
Start Register + 4 and Start Register + 5	Counter input 2 (32 bit register)
Start Register + 6 and Start Register + 7	Counter input 3 (32 bit register)

Notes

Refer to the *5410 High Speed Counter Input Module User Manual* for further information.

6.7 Diagnostic I/O Modules

Diagnostic modules are used to assign diagnostic data from the controller to input or status registers in the I/O Database. The diagnostic data is used to monitor internal controller data such as controller status code, the force LED, serial communication status and protocol status.

Diagnostic I/O modules assign data to I/O database input or status registers, depending on the type of data. Diagnostic I/O modules may assign data to any input or status registers in the I/O database that are not being used by another I/O module.

All I/O database input and status registers that are not assigned to any other I/O modules may be used as general purpose input or status registers in a ladder program. The I/O modules available are described in the following pages.

6.7.1 DIAG Controller Status Code

Description

The module, **DIAG Controller Status Code**, reads the controller status code and assigns it in one input register. The input register is updated continuously with the status code.

The Status LED blinks a binary sequence equal to the controller status code. Note that if a “*Register assignment table checksum error*” occurs, only the Status LED will indicate this status code. The I/O scan is disabled if this error occurs, so that the module **DIAG Controller status code** is no longer updated.

Register Assignment

Module	DIAG Controller Status Code	
Address	No physical address is required.	
Type	Input Register	
Start	Any unused input register.	3xxxx
End	Same as start register.	
Registers	1 input register.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Controller status code: 0 = Normal 1 = I/O module communication failure 2 = Register Assignment Checksum error

Status LED - I/O Error Indication

When the Status LED flashes the error code 1 (i.e. a short flash, once every second), there is a communication failure with one or more I/O module. To correct the problem, do one of the following:

1. Ensure that every module contained in the Register Assignment is connected to the controller. Check that the module address selected for each module agrees with the selection made in the Register Assignment.
2. If a module is currently not connected to the controller, delete it from the Register Assignment. Download the new Register Assignment to the controller.
3. If a module is still suspect of having failed, confirm the failure by removing the module from the Register Assignment. Download the new Register Assignment to the controller. The Status LED should stop flashing.
4. If unused modules must be intentionally left in the Register Assignment, the I/O error indication may be disabled from a selection box on the Register Assignment dialog.

Notes

Refer to the *5203/4 System Manual* or the *SCADAPack System Manual* for further information on the Status LED and Status Output.

6.7.2 DIAG DNP Connection Status

Description

This module determines the connection status of a remote DNP station, by repeatedly sending a short message to the selected remote station and monitoring the response.

The connection status information is assigned to an input register, which is updated continuously.

One holding register is assigned to specify the repeat time period for polling the remote station. A second holding register specifies the remote station address.

This module is used by the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers only.

Register Assignment

Module	DIAG DNP connection status	
Address	No address is required	
Type	Input Register	
Start	Any unused input register	3xxxx
End	Same as start register	
Registers	1 input register	
Description	Connection status: 0 = connection OK 1 = connection failed	
Type	Holding Register	
Start	First register of any unused block of 2 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx +1
Registers	2 holding registers.	
Description	Configuration data	
Extended Parameters	None	

Holding Register Data

Registers	Assignment to Module Hardware
Start Register	DNP station address
Start Register + 1	Repeat rate (in seconds) for connection request message

6.7.3 DIAG DNP Event Status Module

Description

This module provides diagnostic data about DNP change events. The number of change events stored in the event buffers is totaled for all DNP classes, and for all DNP point types.

The status information is assigned to nine consecutive input registers. The input registers are updated continuously with data concerning the DNP change events.

One coil register is assigned to clear the change event buffers. The data is cleared when the coil register is set to 1. Once the data is cleared the coil register is automatically reset to 0.

Register Assignment

Module	DIAG DNP event status	
Address	No address is required	
Type	Input Register	
Start	First register of any unused block of 9 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +8
Registers	12 input registers	
Description	diagnostic data	
Type	Coil Register	
Start	Any unused coil register	0xxxx
End	Same as start register	
Registers	1 coil register	
Description	clear event buffers	
Extended Parameters	None	

Input Register Data

Registers	Assignment to Module Hardware
Start Register	DNP event count for Binary Inputs
Start Register + 1	DNP event count for 16-bit Counter Inputs
Start Register + 2	DNP event count for 32-bit Counter Inputs
Start Register + 3	DNP event count for 16-bit Analog Inputs
Start Register + 4	DNP event count for 32-bit Analog Inputs
Start Register + 5	DNP event count for Short Float Analog Inputs
Start Register + 6	DNP event count for Class 1 Data
Start Register + 7	DNP event count for Class 2 Data
Start Register + 8	DNP event count for Class 3 Data
Start Register + 9	reserved for future use
Start Register +10	reserved for future use
Start Register +11	reserved for future use

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Clear change event buffers

6.7.4 DIAG DNP Port Status Module

Description

This module provides DNP diagnostic data for a specific communication port. Diagnostic information is totaled for all DNP connections on the selected communication port.

The status information is assigned to nine consecutive input registers. The input registers are updated continuously with the DNP diagnostic data.

One coil register is assigned to clear the diagnostic data. The data is cleared when the coil register is set to 1. Once the data is cleared the coil register is automatically reset to 0.

Register Assignment

Module	DIAG DNP port status	
Address	com1, com2, com3, com4, or LAN1	
Type	Input Register	
Start	First register of any unused block of 7 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +6
Registers	10 input registers	
Description	diagnostic data	
Type	Coil Register	
Start	Any unused coil register	0xxxx
End	Same as start register	
Registers	1 coil register	
Description	clear diagnostic data	
Extended Parameters	None	

Input Register Data

Registers	Assignment to Module Hardware
Start Register	DNP message successes Number of successful DNP message transactions initiated by this station
Start Register + 1	DNP message failures Number of failed DNP message transactions initiated by this station
Start Register + 2	DNP fails since last success Number of consecutive failed DNP message transactions initiated by this station since the last success
Start Register + 3	DNP format errors Number of received DNP message frames with formatting errors (including checksum errors and unrecognized commands)
Start Register + 4	DNP message frames received Number of DNP message frames received
Start Register + 5	DNP message frames sent Number of DNP message frames sent. This includes any message retries at application or data link layer.

Registers	Assignment to Module Hardware
Start Register + 6	DNP messages received Number of complete DNP messages received
Start Register + 7	DNP messages sent Number of complete DNP messages sent. This includes any message retries at the application layer.
Start Register + 8	reserved for future use
Start Register + 9	reserved for future use

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Clear diagnostic data

6.7.5 DIAG DNP Station Status

Description

This module provides DNP diagnostic data for a specific DNP remote station address. Diagnostic information is totaled for all DNP communication with the station.

The status information is assigned to nine consecutive input registers. The input registers are updated continuously with data concerning the DNP status of the specific interface.

One coil register is assigned to clear the diagnostic data. The data is cleared when the coil register is set to 1. Once the data is cleared the coil register is automatically reset to 0.

One holding register specifies the remote station address.

Register Assignment

Module	DIAG DNP station status	
Address	No address is required	
Type	Input Register	
Start	First register of any unused block of 10 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +9
Registers	10 input registers	
Description	diagnostic data	
Type	Coil Register	
Start	Any unused coil register	0xxxx
End	Same as start register	
Registers	1 coil register	
Description	clear diagnostic data	
Type	Holding Register	
Start	Any unused holding register.	4xxxx
End	Same as start register	
Registers	1 holding register.	
Description	DNP station address	
Extended Parameters	None	

Input Register Data

Registers	Assignment to Module Hardware
Start Register	DNP message successes Number of successful DNP message transactions initiated by this station
Start Register + 1	DNP message failures Number of failed DNP message transactions initiated by this station
Start Register + 2	DNP fails since last success Number of consecutive failed DNP message transactions initiated by this station since the last success

Registers	Assignment to Module Hardware
Start Register + 3	DNP format errors Number of received DNP message frames with formatting errors (including checksum errors and unrecognized commands)
Start Register + 4	DNP message frames received Number of DNP message frames received
Start Register + 5	DNP message frames sent Number of DNP message frames sent. This includes any message retries at application or data link layer.
Start Register + 6	DNP messages received Number of complete DNP messages received
Start Register + 7	DNP messages sent Number of complete DNP messages sent. This includes any message retries at the application layer.
Start Register + 8	reserved for future use
Start Register + 9	reserved for future use

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Clear diagnostic data

Holding Register Data

Registers	Assignment to Module Hardware
Start Register	DNP station address

6.7.6 DIAG Force

Description

All user accessible Modbus registers in a SCADAPack controller can be forced to a desired state or value. The **DIAG Force** module is used to determine if a Modbus register in a SCADAPack controller is forced to some state or value. This module reads the ‘forced’ state of a SCADAPack controller, and assigns it to one status register. The status register is updated continuously with the current ‘forced’ state of the controller.

Note that the Force LED on SCADAPack controllers, excluding the SCADASense programmable controllers, also indicates if a Modbus register is forced.

Register Assignment

Module	DIAG Force	
Address	No physical address is required.	
Type	Status Register	
Start	Any unused status register.	1xxxx
End	Same as start register.	
Registers	1 status register.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	0 = Modbus register is not forced 1 = Modbus register is forced Force LED: 0 = Force LED is OFF 1 = Force LED is ON

6.7.7 DIAG IP Connections

Description

The module, **DIAG IP connections**, provides a summary of the IP connections. The connection information is assigned to three consecutive input registers. The input registers are updated continuously with connection data.

This module is used by the SCADAPack 32, SCADAPack 330/334 and SCADAPack 350 controllers only. The connection statistics include all IP connection types.

Register Assignment

Module	DIAG IP connections	
Address	No physical address is required.	
Type	Input Register	
Start	First register of any unused block of 3 consecutive input registers.	3xxxx
End	Last register of block	3xxxx + 2
Registers	3 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Number of current slave IP connections.
Start Register + 1	Number of current master IP connections.
Start Register + 2	Number of free IP connections.

6.7.8 DIAG Logic Status

Description

The module, **DIAG Logic Status**, reads the Ladder Logic Application status in the RAM and FLASH memory and assigns it to two input registers. The input registers are updated continuously with the current application status.

Register Assignment

Module	DIAG Logic Status	
Address	No physical address is required.	
Type	Input Register	
Start	Any unused input register.	3xxxx
End	Last register in block.	3xxxx+1
Registers	2 input registers	
Description	None	

Input Register Data

Registers	Assignment to Module Hardware
Start Register	Ladder Logic Program Status 0 = No Program 1 = Program Stopped 2 = Program Running in Debug Mode 3 = Program Running
Start Register + 1	Type of memory where Ladder Logic program is loaded: 0 = No program in RAM or FLASH; 1 = Program loaded in RAM 2 = Program loaded in FLASH 3 = Program loaded in RAM and FLASH

6.8 DIAG Modbus Protocol Status

Description

The module, **DIAG Modbus protocol status**, provides diagnostic data about Modbus protocols for a specific communication interface. Diagnostic information is totaled for all Modbus protocols on the selected communication interface. When multiple Modbus IP connections exist on the same interface, diagnostic information is totaled for all connections.

This module is used by the SCADAPack 32 controller series only.

The status information is assigned to nine consecutive input registers. The input registers are updated continuously with data concerning the Modbus protocol status of the specific interface.

One coil register is assigned to clear the diagnostic data. The data is cleared when the coil register is set to 1. Once the data is cleared the coil register is automatically reset to 0.

Register Assignment

Module	DIAG Modbus protocol status	
Address	com1, com2, com3, com4, or Ethernet1	
Type	Input Register	
Start	First register of any unused block of 9 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +8
Registers	9 input registers	
Description	diagnostic data	
Type	Coil Register	
Start	Any unused coil register.	0xxxx
End	Same as start register.	
Registers	1 coil register.	
Description	clear diagnostic data	
Extended Parameters	None	

Input Register Data

Registers	Assignment to Module Hardware
Start Register	Modbus command errors
Start Register + 1	Modbus format errors
Start Register + 2	Modbus checksum errors
Start Register + 3	Modbus slave commands received
Start Register + 4	Modbus master commands sent
Start Register + 5	Modbus master responses received
Start Register + 6	Modbus slave responses sent
Start Register + 7	Modbus stored messages
Start Register + 8	Modbus forwarded messages

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Clear diagnostic data

6.8.1 DIAG Serial Port Comm. Status

Description

The module, **DIAG Serial port comm. status**, provides diagnostic data about a specific serial port. The status information is assigned to five consecutive input registers. The input registers are updated continuously with data concerning the communication status of the serial port.

Register Assignment

Module	DIAG Serial port comm. Status	
Address	com1, com2, com3, or com4.	
Type	Input Register	
Start	First register of any unused block of 5 consecutive input registers.	3xxx
End	Last register of block	3xxx + 4
Registers	5 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Framing errors
Start Register + 1	Parity errors
Start Register + 2	Character overrun errors
Start Register + 3	Buffer overrun errors
Start Register + 4	Integer value indicating the status of Port I/O lines CTS and DCD: The LSB = state of the CTS line. Second to LSB = state of DCD input Example: Value of 3 implies CTS and DCD is ON. Value of 1 implies CTS is ON and DCD is OFF.

Notes

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

6.8.2 DIAG Serial Port Protocol Status

Description

The module, **DIAG Serial port protocol status**, provides diagnostic data about a specific serial port. The status information is assigned to ten consecutive input registers. The input registers are updated continuously with data concerning the protocol status of the serial port.

Register Assignment

Module	DIAG Serial port comm. Status	
Address	com1, com2, com3, or com4.	
Type	Input Register	
Start	First register of any unused block of 10 consecutive input registers.	3xxxx
End	Last register of block	3xxxx + 9
Registers	10 input registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Protocol command errors
Start Register + 1	Protocol format errors
Start Register + 2	Protocol checksum errors
Start Register + 3	Protocol slave commands received
Start Register + 4	Protocol master commands sent
Start Register + 5	Protocol master responses received
Start Register + 6	Protocol slave responses sent

Registers	Assignment to Module Hardware
Start Register + 7	Protocol master command status 0 = message sent, waiting for response 1 = response received, no error occurred 2 = not used 3 = bad value in function code register 4 = bad value in slave controller address register 5 = bad value in slave register address 6 = bad value in length register 7 = serial port or protocol is invalid 8 = not used 9 = specified protocol is not supported by this controller 11 = an unexpected length response was received This means corrupt messages for regular Modbus. Check the format and checksum counters. For Enron Modbus, this is normal. 12 = response timeout 24 = exception response: invalid function code 25 = exception response: invalid address 26 = exception response: invalid value 27 = protocol is invalid or serial port queue is full 28 = slave and master stations are equal; they must be different 29 = exception response: slave device failure 30 = exception response: slave device busy
Start Register + 8	Protocol stored messages
Start Register + 9	Protocol forwarded messages

Notes

For SCADAPack, SCADAPack Light and SCADAPack Plus controllers Com3 is supported only when the **SCADAPack 5601 and 5604 I/O module** is installed. Com4 is supported only when the **SCADAPack 5602 I/O module** is installed. To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator terminals, and for programming and diagnostics using the TelePACE program.

6.9 Digital Input I/O Modules

Digital input modules are used to assign data from physical digital inputs to status registers in the I/O Database. The physical digital inputs are specific 5000 Series I/O modules, generic I/O modules, and controller digital inputs.

Digital input modules may assign data to any status registers in the I/O database that are not being used by another Digital input module. There are 4096 I/O database status registers available. These status registers are numbered 10001 - 14096. Status registers are referred to as 1xxxx registers throughout this manual.

All I/O database status registers that are not assigned to any other I/O modules may be used as general-purpose status registers in a ladder program. The I/O modules available are described in the following pages.

6.9.1 DIN Controller Digital Inputs

Description

The module, **DIN controller digital inputs**, reads the three digital inputs on the Micro16, SCADAPack, SCADAPack Light, SCADAPack, SCADAPack Plus, SCADAPack 32 and SCADAPack 32P controllers. Data is assigned to three consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Register Assignment

Module	DIN controller digital inputs	
Address	No physical address is required.	
Type	Status Register	
Start	First register of any unused block of 3 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 2
Registers	3 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Controller board digital input 0
Start Register + 1	Controller board digital input 1
Start Register + 2	Controller board digital input 2

Notes

Refer to the *Hardware User Manual* for further information.

6.9.2 DIN Controller Interrupt Input

Description

The module, **DIN controller interrupt input**, reads the state of the interrupt input on the Micro16, SCADAPack, SCADAPack Light, SCADAPack, SCADAPack Plus, SCADAPack 32 and SCADAPack 32P controllers. Data is assigned to one status register. The status register is updated continuously with the state of the interrupt input.

Register Assignment

Module	DIN controller interrupt input	
Address	No physical address is required.	
Type	Status Register	
Start	Any unused status register.	1xxxx
End	Same as start register.	
Registers	1 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Controller board Interrupt Input

Notes

Refer to the Hardware User Manual for further information on the Controller interrupt input.

6.9.3 DIN Controller Option Switches

Description

The module, **DIN Controller option switches**, reads the state of the three option switches on the Micro16, SCADAPack, SCADAPack Light, SCADAPack, SCADAPack Plus, SCADAPack 32 and SCADAPack 32P controllers. Data is assigned to three status registers. The status registers are updated continuously with data read from the option switches.

Register Assignment

Module	DIN 5203/4 option switches	
Address	No physical address is required.	
Type	Status Register	
Start	First register of any unused block of 3 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 2
Registers	3 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Controller board option switch 1
Start Register + 1	Controller board option switch 2
Start Register + 2	Controller board option switch 3

Notes

The option switches are also used as configuration switches for the SCADAPack I/O modules. Refer to these I/O modules for details.

Refer to the *5203/4 System Manual* or the *SCADAPack System Manual* for further information on the controller option switches.

6.9.4 DIN SCADAPack 32 Option Switches

Description

The module, **DIN SCADAPack 32 option switches**, reads the state of the four option switches on the SCADAPack 32 controller board. Data is assigned to four status registers. The status registers are updated continuously with data read from the option switches.

Register Assignment

Module	DIN SCADAPack 32 option switches	
Address	No physical address is required.	
Type	Status Register	
Start	First register of any unused block of 4 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 3
Registers	4 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Controller board option switch 1
Start Register + 1	Controller board option switch 2
Start Register + 2	Controller board option switch 3
Start Register + 3	Controller board option switch 4

Notes

The option switches are also used as configuration switches for the SCADAPack I/O modules. Refer to these I/O modules for details.

Refer to the *SCADAPack 32 System Manual* for further information on the controller option switches.

6.9.5 DIN 5401 Module

Description

The **DIN 5401 module** provides eight digital inputs. Data is assigned to eight consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

When there are digital outputs being used in combination with digital inputs on the 5401, a full 8 status registers must still be assigned to the **DIN 5401 module**. When this is the case, a **DOUT 5401 module** must also be added to the Register Assignment with the same module address.

Register Assignment

Module			DIN 5401 module
Address	This module is assigned a unique module address between 0 and 7. No other DIN-type module may use this module address.		
Type	Status Register		
Start	First register of any unused block of 8 consecutive status registers.	1xxxx	
End	Last register of block	1xxxx + 7	
Registers	8 status registers.		
Description	None		
Extended Parameters	None		

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7

Notes

Refer to the *5401 Digital I/O Module User Manual* for further information.

6.9.6 DIN 5402 Module

Description

The **DIN 5402 module** provides sixteen digital inputs. Data is assigned to sixteen consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Register Assignment

Module	DIN 5402 module	
Address	This module is assigned a unique module address between 0 and 15. No other DIN-type module may use this module address.	
Type	Status Register	
Start	First register of any unused block of 16 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 15
Registers	16 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7
Start Register + 8	Digital input 8
Start Register + 9	Digital input 9
Start Register + 10	Digital input 10
Start Register + 11	Digital input 11
Start Register + 12	Digital input 12
Start Register + 13	Digital input 13
Start Register + 14	Digital input 14
Start Register + 15	Digital input 15

Notes

Refer to the *5402 Digital I/O Module User Manual* for further information.

6.9.7 DIN 5403 Module

Description

The **DIN 5403 module** provides eight digital inputs. Data is assigned to eight consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Register Assignment

Module	DIN 5403 module	
Address	This module is assigned a unique module address between 0 and 7. No other DIN-type module may use this module address.	
Type	Status Register	
Start	First register of any unused block of 8 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 7
Registers	8 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7

Notes

Refer to the *5403 Digital Input Module User Manual* for further information.

6.9.8 DIN 5404 Module

Description

The **DIN 5404 module** provides sixteen digital inputs. Data is assigned to sixteen consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Register Assignment

Module	DIN 5404 module	
Address	This module is assigned a unique module address between 0 and 15. No other DIN-type module may use this module address.	
Type	Status Register	
Start	First register of any unused block of 16 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 15
Registers	16 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7
Start Register + 8	Digital input 8
Start Register + 9	Digital input 9
Start Register + 10	Digital input 10
Start Register + 11	Digital input 11
Start Register + 12	Digital input 12
Start Register + 13	Digital input 13
Start Register + 14	Digital input 14
Start Register + 15	Digital input 15

Notes

Refer to the *5404 Digital Input Module User Manual* for further information.

6.9.9 DIN 5405 Module

Description

The **DIN 5405 module** provides thirty-two digital inputs. Data is assigned to thirty-two consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Register Assignment

Module	DIN 5404 module	
Address	This module is assigned a unique module address between 0 and 15. No other DIN-type module may use this module address.	
Type	Status Register	
Start	First register of any unused block of 32 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 31
Registers	32 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7
Start Register + 8	Digital input 8
Start Register + 9	Digital input 9
Start Register + 10	Digital input 10
Start Register + 11	Digital input 11
Start Register + 12	Digital input 12
Start Register + 13	Digital input 13
Start Register + 14	Digital input 14
Start Register + 15	Digital input 15
Start Register + 16	Digital input 16
Start Register + 17	Digital input 17
Start Register + 18	Digital input 18
Start Register + 19	Digital input 19
Start Register + 20	Digital input 20
Start Register + 21	Digital input 21

Registers	Assignment to Module Hardware
Start Register + 22	Digital input 22
Start Register + 23	Digital input 23
Start Register + 24	Digital input 24
Start Register + 25	Digital input 25
Start Register + 26	Digital input 26
Start Register + 27	Digital input 27
Start Register + 28	Digital input 28
Start Register + 29	Digital input 29
Start Register + 30	Digital input 30
Start Register + 31	Digital input 31

Notes

Refer to the *5405 Digital Input Module User Manual* for further information.

6.9.10 DIN 5414 Module

Description

The **DIN 5414 module** provides sixteen digital inputs from a 5414 Digital Input module. A maximum of sixteen 5414 digital input modules may be installed on the I/O bus.

Register Assignment

The register assignment for the DIN 5414 module is used to configure the registers used for the module data and the Extended Parameters available for the module. To open the register assignment dialog for the DIN 5414 module:

Select **Register Assignment** from the **Controller** menu in TelePACE.

In the **Register Assignment** window select the **Add** button.

In the **Add Register Assignment** dialog select the **DIN 5414 module** from the list presented in the **Module** window.

The register assignment dialog for the **DIN 5414 module** is shown below.

The dialog box titled "Add Register Assignment" has a blue header bar with a close button (X) in the top right corner. It contains the following fields and controls:

- Module:** A dropdown menu showing "DIN 5414 module".
- Address:** A dropdown menu showing "0".
- OK** and **Cancel** buttons are located to the right of the Module and Address fields.
- Table:** A table with columns: Type, Start, End, Registers, and Description.

Type	Start	End	Registers	Description
1xxxx	10016	10031	16	
- Extended Parameters:** A section with two dropdown menus:
 - AC/DC:** Set to "DC".
 - Scan Frequency:** Set to "60 Hz".

The **Module** selection displays the name of the 5414 Digital Analog Input module used for the register assignment.

The **Address** selection displays the module address of the physical 5414 Digital Input module. The address on the module is selected via dip switches on the module. A maximum of sixteen DIN type modules may be added to a system.

The register type **1xxxx** defines the register range for the digital input status data. Sixteen sequential registers are needed for the DIN 5414 module. Any unused block of sixteen registers may be used.

The **Extended Parameters** grouping defines the type of data returned from the DIN 5414 module, filtering used by all inputs and the scan frequency. The **AC/DC** selection for all inputs defines the input measurement type for the digital inputs.

- The **DC** selection sets the input to measure DC input signals.
- The **AC** selection sets the input to measure AC input signals.

The **Scan Frequency** selection set the scan rate for all digital inputs. . The scan rate selection is not critical but AC noise rejection is improved at the correct frequency. If the module is used in a DC environment, the 60 Hz setting will yield slightly faster response time.

- The **60 Hz** selection synchronizes the input scanning to 60 Hz.
- The **50 Hz** selection synchronizes the input scanning to 50 Hz.

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7
Start Register + 8	Digital input 8
Start Register + 9	Digital input 9
Start Register + 10	Digital input 10
Start Register + 11	Digital input 11
Start Register + 12	Digital input 12
Start Register + 13	Digital input 13
Start Register + 14	Digital input 14
Start Register + 15	Digital input 15

Notes

Refer to the *5414 Digital Input Module User Manual* for further information.

6.9.11 DIN 5421 Module

Description

The **DIN 5421 module** provides eight digital inputs. Data is assigned to eight consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Register Assignment

Module	DIN 5421 module	
Address	This module is assigned a unique module address between 0 and 7. No other DIN-type module may use this module address.	
Type	Status Register	
Start	First register of any unused block of 8 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 7
Registers	8 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7

Notes

Refer to the *5421 Switch Input Module Manual* for further information.

6.9.12 DIN Generic 16 Point Module

Description

The **DIN Generic 16 point module** provides sixteen digital inputs. Data is assigned to sixteen consecutive status registers. The status registers are updated continuously with data read from the digital inputs. The **DIN Generic 16 point module** may be used in place of any other 16 point DIN-type module.

The **DIN Generic 16 point module** type is a useful selection early on in the system design stage before the final selection of the specific DIN module type is known.

Register Assignment

Module	DIN Generic 16 point module	
Address	This module is assigned a unique module address between 0 and 15. No other DIN-type module may use this module address.	
Type	Status Register	
Start	First register of any unused block of 16 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 15
Registers	16 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7
Start Register + 8	Digital input 8
Start Register + 9	Digital input 9
Start Register + 10	Digital input 10
Start Register + 11	Digital input 11
Start Register + 12	Digital input 12
Start Register + 13	Digital input 13
Start Register + 14	Digital input 14
Start Register + 15	Digital input 15

Notes

For further information, refer to the *User Manual* for the specific 16 point DIN module used.

6.9.13 DIN Generic 8 Point Module

Description

The **DIN Generic 8 point module** provides eight digital inputs. Data is stored in eight consecutive status registers. The status registers are updated continuously with data read from the digital inputs. The **DIN Generic 8 point module** may be used in place of any other 8 point DIN-type module.

The **DIN Generic 8 point module** type is a useful selection early on in the system design stage before the final selection of the specific DIN module type is known.

Register Assignment

Module	DIN Generic 8 point module	
Address	This module is assigned a unique module address between 0 and 7. No other DIN-type module may use this module address.	
Type	Status Register	
Start	First register of any unused block of 8 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 7
Registers	8 status registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7

Notes

For further information, refer to the *User Manual* for the specific 8 point DIN module used.

6.10 Digital Output I/O Modules

Digital output modules are used to assign data from the I/O Database to physical digital outputs. The physical digital outputs are specific 5000 Series I/O modules or generic I/O modules.

Digital output modules may assign data to any coil registers in the I/O database that are not being used by another Digital output module. There are 4096 I/O database coil registers available. These coil registers are numbered 00001 - 04096. Coil registers are referred to as 0xxxx registers throughout this manual.

All I/O database coil registers that are not assigned to any other I/O modules may be used as general-purpose coil registers in a ladder program. The I/O modules available are described in the following pages.

6.10.1 DOUT 5401 Module

Description

The **DOUT 5401 module** provides eight digital outputs. Data is assigned from eight consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

When there are digital inputs being used in combination with digital outputs on the 5401, a full 8 coil registers must still be assigned to the **DOUT 5401 module**. When this is the case, a **DIN 5401 module** must also be added to the Register Assignment with the same module address.

Register Assignment

Module	DOUT 5401 module	
Address	This module is assigned a unique module address between 0 and 7. No other DOUT-type module may use this module address.	
Type	Coil Register	
Start	First register of any unused block of 8 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 7
Registers	8 coil registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7

Notes

Refer to the *5401 Digital I/O Module User Manual* for further information.

6.10.2 DOUT 5402 Module

Description

The **DOUT 5402 module** provides sixteen digital outputs. Data is assigned from sixteen consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Register Assignment

Module	DOUT 5402 module	
Address	This module is assigned a unique module address between 0 and 15. No other DOUT-type module may use this module address.	
Type	Coil Register	
Start	First register of any unused block of 16 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 15
Registers	16 coil registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7
Start Register + 8	Digital output 8
Start Register + 9	Digital output 9
Start Register + 10	Digital output 10
Start Register + 11	Digital output 11
Start Register + 12	Digital output 12
Start Register + 13	Digital output 13
Start Register + 14	Digital output 14
Start Register + 15	Digital output 15

Notes

Refer to the *5402 Digital I/O Module User Manual* for further information.

6.10.3 DOUT 5406 Module

Description

The **DOUT 5406 module** provides sixteen digital outputs. Data is assigned from sixteen consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Register Assignment

Module	DOUT 5406 module	
Address	This module is assigned a unique module address between 0 and 15. No other DOUT-type module may use this module address.	
Type	Coil Register	
Start	First register of any unused block of 16 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 15
Registers	16 coil registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7
Start Register + 8	Digital output 8
Start Register + 9	Digital output 9
Start Register + 10	Digital output 10
Start Register + 11	Digital output 11
Start Register + 12	Digital output 12
Start Register + 13	Digital output 13
Start Register + 14	Digital output 14
Start Register + 15	Digital output 15

Notes

Refer to the *5406 Digital Output Module User Manual* for further information.

6.10.4 DOUT 5407 Module

Description

The **DOUT 5407 module** provides eight digital outputs. Data is assigned from eight consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Register Assignment

Module	DOUT 5407 module	
Address	This module is assigned a unique module address between 0 and 7. No other DOUT-type module may use this module address.	
Type	Coil Register	
Start	First register of any unused block of 8 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 7
Registers	8 coil registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7

Notes

Refer to the *5407 Relay Output Module Manual* for further information.

6.10.5 DOUT 5408 Module

Description

The **DOUT 5408 module** provides eight digital outputs. Data is assigned from eight consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Register Assignment

Module	DOUT 5408 module	
Address	This module is assigned a unique module address between 0 and 7. No other DOUT-type module may use this module address.	
Type	Coil Register	
Start	First register of any unused block of 8 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 7
Registers	8 coil registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7

Notes

Refer to the *5408 Digital Output Module Manual* for further information.

6.10.6 DOUT 5409 Module

Description

The **DOUT 5409 module** provides eight digital outputs. Data is assigned from eight consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Register Assignment

Module	DOUT 5409 module

Address	This module is assigned a unique module address between 0 and 7. No other DOUT-type module may use this module address.	
Type	Coil Register	
Start	First register of any unused block of 8 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 7
Registers	8 coil registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7

Notes

Refer to the *5409 Digital Output Module Manual* for further information.

6.10.7 DOUT 5411 Module

Description

The **DOUT 5411 module** provides thirty-two digital outputs. Data is assigned from thirty-two consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Register Assignment

Module	DOUT 5411 module	
Address	This module is assigned a unique module address between 0 and 15. No other DOUT-type module may use this module address.	
Type	Coil Register	
Start	First register of any unused block of 32 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 31
Registers	32 coil registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7
Start Register + 8	Digital output 8
Start Register + 9	Digital output 9
Start Register + 10	Digital output 10
Start Register + 11	Digital output 11
Start Register + 12	Digital output 12
Start Register + 13	Digital output 13
Start Register + 14	Digital output 14
Start Register + 15	Digital output 15
Start Register + 16	Digital output 16
Start Register + 17	Digital output 17
Start Register + 18	Digital output 18
Start Register + 19	Digital output 19
Start Register + 20	Digital output 20
Start Register + 21	Digital output 21

Registers	Assignment to Module Hardware
Start Register + 22	Digital output 22
Start Register + 23	Digital output 23
Start Register + 24	Digital output 24
Start Register + 25	Digital output 25
Start Register + 26	Digital output 26
Start Register + 27	Digital output 27
Start Register + 28	Digital output 28
Start Register + 29	Digital output 29
Start Register + 30	Digital output 30
Start Register + 31	Digital output 31

Notes

Refer to the *5411 Digital Output Module User Manual* for further information.

6.10.8 DOUT 5415 Module

Description

The **DOUT 5415 module** provides twelve relay digital outputs. Data is assigned from twelve consecutive coil registers. The 5415 digital outputs are updated continuously with data read from the coil registers.

Register Assignment

Module	DOUT 5415 module	
Address	This module is assigned a unique module address between 0 and 15. No other DOUT-type module may use this module address.	
Type	Coil Register	
Start	First register of any unused block of 12 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 11
Registers	11 coil registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7
Start Register + 8	Digital output 8
Start Register + 9	Digital output 9
Start Register + 10	Digital output 10
Start Register + 11	Digital output 11

Notes

Refer to the *5415 Digital Output Module User Manual* for further information.

6.10.9 DOUT Generic 16 Point Module

Description

The **DOUT Generic 16 point module** provides sixteen digital outputs. Data is assigned from sixteen consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers. The **DOUT Generic 16 point module** may be used in place of any other 16 point DOUT-type module.

The **DOUT Generic 16 point module** type is a useful selection early on in the system design stage before the final selection of the specific DOUT module type is known.

Register Assignment

Module	Generic 16 point module	
Address	This module is assigned a unique module address between 0 and 15. No other DOUT-type module may use this module address.	
Type	Coil Register	
Start	First register of any unused block of 16 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 15
Registers	16 coil registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7
Start Register + 8	Digital output 8
Start Register + 9	Digital output 9
Start Register + 10	Digital output 10
Start Register + 11	Digital output 11
Start Register + 12	Digital output 12
Start Register + 13	Digital output 13
Start Register + 14	Digital output 14
Start Register + 15	Digital output 15

Notes

For further information, refer to the *Digital Output Module User Manual* for the specific 16-point DOUT module used.

6.10.10 DOUT Generic 8 Point Module

Description

The **DOUT Generic 8 point module** provides eight digital outputs. Data is assigned from eight consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers. The **DOUT Generic 8 point module** may be used in place of any other 8 point DOUT-type module.

The **DOUT Generic 8 point module** type is a useful selection early on in the system design stage before the final selection of the specific DOUT module type is known.

Register Assignment

Module	DOUT Generic 8 point module	
Address	This module is assigned a unique module address between 0 and 7. No other DOUT-type module may use this module address.	
Type	Coil Register	
Start	First register of any unused block of 8 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 7
Registers	8 coil registers.	
Description	None	
Extended Parameters	None	

Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7

Notes

For further information, refer to the *Digital Output Module Manual* for the specific 8 point DOUT module used.

6.11 SCADAPack and SCADASense Series I/O Modules

SCADAPack and SCADASense Series I/O modules assign data from registers to physical outputs, and assign data from physical inputs to registers. These I/O modules provide analog I/O points, digital I/O points and may assign data to registers that are not used by any other I/O module.

The following modules interface with SCADAPack and SCADASense Series I/O hardware.

6.11.1 4202 DR Extended/4203 DR IO

Description

The SCADASense 4203 DR, and a SCADASense 4202 DR manufactured after July 13, 2004, may use the 4202 DR Extended/4203 DR I/O register assignment. The 4202 DR must have controller board version 5 and terminal board version 6.

A SCADASense 4203 DR, or 4202 DR with Extended I/O, provides one digital point, which may operate as a digital input/counter or as a digital output. The digital output shares the same physical connections as the digital input. When the output is turned OFF, the point may be used as input. When the output is turned on it functions only as a digital output.

Digital point data is assigned to a single coil register and a single status register. The coil register is updated continuously with data read from the coil register. The status register is updated continuously with data read from the digital input.

Analog and counter input data is assigned to five consecutive input registers. One input register is used to monitor the controller input power and two counter input double registers (32 bit). These input registers are updated continuously with data read from the analog inputs and counter input.

Analog output data is assigned to a single analog output register. The analog output is updated continuously with the data read from the holding register.

Register Assignment

Module	4202 DR Extended/4203 DR I/O Module	
Address	Fixed	
Type	Coil Register	
Start	Any unused coil register.	0xxxx
End	Same as first register.	0xxxx
Registers	1 coil registers	
Description	digital output	
Type	Status Register	
Start	Any unused status register	1xxxx
End	Same as first register.	1xxxx
Registers	1 status register	
Description	Digital input	
Type	Input Register	
Start	First register of any unused block of 5 consecutive status registers.	3xxxx
End	Last register of block	3xxxx + 4
Registers	5 input registers	
Description	Analog and counter inputs	
Type	Analog Output Register	
Start	Any unused analog output register	4xxxx
End	Same as first register	4xxxx
Registers	1 analog output register	
Description	Analog outputs	

Extended Parameters	None

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0

Analog and Counter Input Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input Supply Voltage (mV)
Start Register + 1 and +2	Counter input, 32 bit register
Start Register + 3 and +4	Counter input, 32 bit register

Analog Output Register Data

Registers	Assignment to Module Hardware
Start Register	Analog output – Input Voltage (mV)

Notes

Only one **4202 DR Extended/4203 DR I/O** may be assigned.

Refer to the **SCADASense 4202 DR Hardware Manual** for information on analog and digital I/O usage.

6.11.2 4202 DR I/O

Description

Digital input data is assigned to a single status register. The status register is updated continuously with data read from the digital input.

Analog and counter input data is assigned to five consecutive input registers. One input register is used to monitor the controller input power and two counter input double registers (32 bit). These input registers are updated continuously with data read from the analog inputs and counter input.

Analog output data is assigned to a single analog output register. The analog output is updated continuously with the data read from the holding register.

Register Assignment

Module	4202 DR I/O Module	
Address	Fixed	
Type	Status Register	
Start	Any unused status register	1xxxx
End	Same as first register.	1xxxx
Registers	1 status register	
Description	Digital input	
Type	Input Register	
Start	First register of any unused block of 5 consecutive status registers.	3xxxx
End	Last register of block	3xxxx + 4
Registers	5 input registers	
Description	Analog and counter inputs	
Type	Analog Output Register	
Start	Any unused analog output register	4xxxx
End	Same as first register	4xxxx
Registers	1 analog output register	
Description	Analog outputs	
Extended Parameters	None	

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0

Analog and Counter Input Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input Supply Voltage (mV)
Start Register + 1 and +2	Counter input, 32 bit register
Start Register + 3 and +4	Counter input, 32 bit register

Analog Output Register Data

Registers	Assignment to Module Hardware
Start Register	Analog output – Input Voltage (mV)

Notes

Only one **4202 DR I/O** may be assigned.

Refer to the **SCADASense 4202 DR Hardware Manual** for information on analog and digital I/O usage.

6.11.3 4202/4203 DS I/O

Description

The 4202/4203 DS I/O register assignment is used for the SCADASense DS Series of controllers (4202 DS and 4203 DS). This module provides two digital input/output (I/O) points. The first DI/O point may be used as a digital input/counter or as a digital output. The digital output shares the same physical connections as the digital input/counter. When the output is turned OFF, the point may be used as input/counter. When the output is turned on it functions only as a digital output. Digital point data is assigned to a single coil register and a single status register. The coil register is updated continuously with data read from the coil register. The status register is updated continuously with data read from the digital input.

The second DI/O point may be used as a turbine meter counter input or as a digital output. The digital output shares the same physical connections as the turbine meter counter input. When the output is turned OFF, the point may be used as a turbine meter counter input. When the output is turned on it functions only as a digital output. Digital point data is assigned to a single coil register. The coil register is updated continuously with data read from the coil register.

Analog and counter input data is assigned to seven consecutive input registers. Two input registers are used for the two analog inputs; one input register is used to monitor the controller input power; two input registers are used for the counter input (32 bit double word uses two registers) and two input registers are used for the turbine meter counter input (32 bit double word uses two registers). These input registers are updated continuously with data read from the analog inputs and counter inputs.

Register Assignment

Module	4202/4203 DS I/O Module	
Address	Fixed	
Type	Coil Register	
Start	Any unused coil register.	0xxxx
End	Last register of block.	0xxxx + 1
Registers	2 coil registers	
Description	Digital output	
Type	Status Register	
Start	Any unused status register	1xxxx
End	Same as first register.	1xxxx
Registers	1 status register	
Description	Digital input	
Type	Input Register	
Start	First register of any unused block of 7 consecutive status registers.	3xxxx
End	Last register of block	3xxxx + 6
Registers	7 input registers	
Description	Analog and counter inputs	
Extended Parameters	None	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register +1	Digital output 1

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0

Analog and Counter Input Register Data

Registers	Assignment to Module Hardware
Start Register	Analog Input 0
Start Register +1	Analog Input 1
Start Register +2	Analog input Supply Voltage (mV)
Start Register +3 and +4	Counter 0 input, 32 bit register
Start Register +5 and +6	Counter 1 input, 32 bit register

Notes

Only one **4202/4203 DS I/O** may be assigned.

Refer to the **SCADASense 4202 DS Hardware Manual** for information on analog and digital I/O usage.

6.11.4 SCADAPack AOUT Module

Description

The **SCADAPack AOUT module** provides two analog outputs. Data is read from two consecutive holding (4xxxx) registers. The analog outputs are updated continuously with the data read from the holding registers.

Register Assignment

Module	SCADAPack AOUT module	
Address	This module has a fixed module address of 0. No other AOUT-type module may use this module address when this module is used.	
Type	Holding Register	
Start	First register of any unused block of 2 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx + 1
Registers	2 holding registers.	
Description	None	
Extended Parameters	None	

Register Data

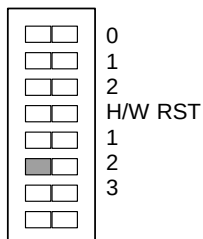
Registers	Assignment to Module Hardware
Start Register	Analog output 0
Start Register + 1	Analog output 1

Output Range Selection

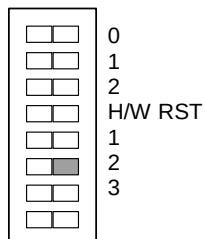
The SCADAPack Option Switch 2 selects the signal range. Both analog outputs are set to the same range.

The figure below shows the switch settings for selecting the output range.

0-20 mA Outputs



4-20 mA Outputs



How to Set the Range Switch

- Determine the desired range.
- Press the side of the switch shown in gray.

Press this side
for 0-20 mA

Press this side
for 4-20 mA



Notes

Refer to the **SCADAPack Hardware Manual** for further information on the option switches.

Option Switch 2 functions as the Output Range Switch regardless of whether the module, **5203/4 option switches**, is added to the Register Assignment or not.

6.11.5 SCADAPack 5601 I/O Module

Description

Digital output data is assigned from twelve consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Digital input data is assigned to sixteen consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Analog input data is assigned to eight consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

Register Assignment

Module	SCADAPack 5601 I/O Module	
Address	fixed at 0	
Type	Coil Register	
Start	First register of any unused block of 12 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 11
Registers	12 coil registers	
Description	digital outputs	
Type	Status Register	
Start	First register of any unused block of 16 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 15
Registers	16 status registers	
Description	digital inputs	
Type	Input Register	
Start	First register of any unused block of 8 consecutive input registers.	3xxxx
End	Last register of block	3xxxx + 7
Registers	8 input registers	
Description	analog inputs	
Extended Parameters	None	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7
Start Register + 8	Digital output 8

Registers	Assignment to Module Hardware
Start Register + 9	Digital output 9
Start Register + 10	Digital output 10
Start Register + 11	Digital output 12

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7
Start Register + 8	Digital input 8
Start Register + 9	Digital input 9
Start Register + 10	Digital input 10
Start Register + 11	Digital input 11
Start Register + 12	Digital input 12
Start Register + 13	Digital input 13
Start Register + 14	Digital input 14
Start Register + 15	Digital input 15

Input Register Data

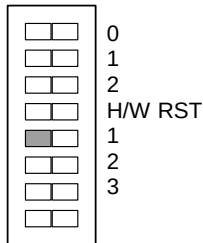
Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Analog Input Measurement Range Selection

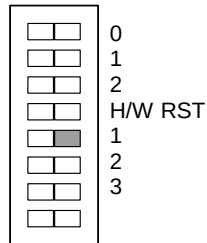
The SCADAPack Option Switch 1 selects the measurement range. All 8 analog inputs are set to the same range.

The figure below shows the switch settings for selecting the measurement range.

0-20 mA (0-5 V) AINs

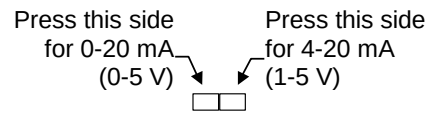


4-20 mA (1-5 V) AINs



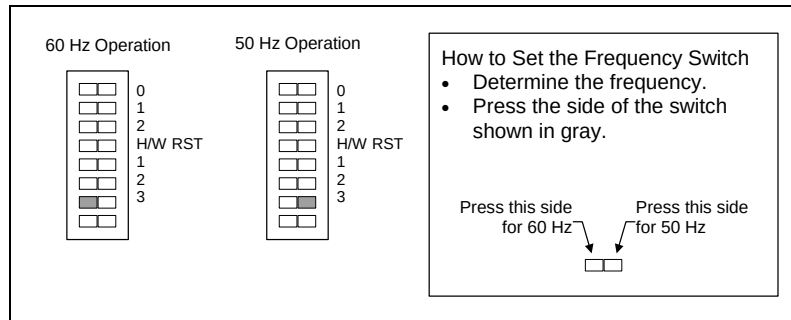
How to Set the Range Switch

- Determine the desired range.
- Press the side of the switch shown in gray.



Line Frequency Selection

The SCADAPack I/O board may select 50 or 60 Hz line frequency for digital and analog input processing. The SCADAPack Option Switch 3 selects this option. The figure below shows the switch settings for selecting the line frequency.



Notes

Only one **SCADAPack 5601 or 5604 I/O module** may be assigned to the same controller.

Option switches 1 and 3 function as described above regardless of whether the module, **DIN 5203/4 option switches**, is added to the Register Assignment or not.

Refer to the **SCADAPack Hardware Manual** for information on the option switches.

Two analog outputs are available for the **SCADAPack 5601 I/O module**. Refer to the **SCADAPack AOUT module** for more information.

6.11.6 SCADAPack 5602 I/O Module

Description

Digital output data is assigned from two consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Analog input data is assigned to five consecutive input registers. The input registers are updated continuously with data read from the analog inputs.

The same five analog inputs are also read as digital inputs. The digital input data is assigned to five consecutive status registers. The status registers are updated continuously with data read from the analog inputs.

A digital input is ON if the corresponding filtered analog input value is greater than or equal to 20% of its full-scale value, otherwise it is OFF.

Register Assignment

Module	SCADAPack 5602 I/O Module	
Address	fixed at 0	
Type	Coil Register	
Start	First register of any unused block of 2 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 1
Registers	2 coil registers	
Description	digital outputs	
Type	Status Register	
Start	First register of any unused block of 5 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 4
Registers	5 status registers	
Description	analog inputs read as digital inputs	
Type	Input Register	
Start	First register of any unused block of 5 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +4
Registers	5 input registers	
Description	analog inputs	
Extended Parameters	None	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0 *
Start Register + 1	Analog input 1 *
Start Register + 2	Analog input 2 *
Start Register + 3	Analog input 3 *
Start Register + 4	Analog input 4 *

* Analog inputs are read again as digital inputs.

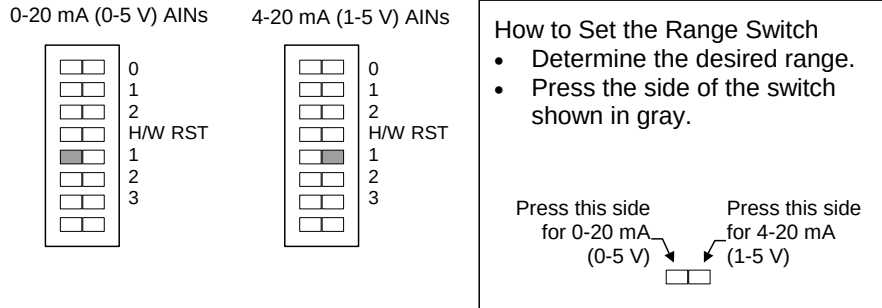
Input Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4

Analog Input Measurement Range Selection

The SCADAPack Option Switch 1 selects the measurement range. All 5 analog inputs are set to the same range.

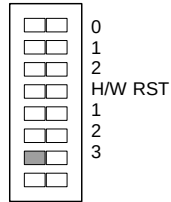
The figure below shows the switch settings for selecting the measurement range.



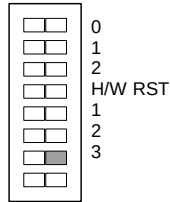
Line Frequency Selection

The SCADAPack I/O board may select 50 or 60 Hz line frequency for analog input processing. The SCADAPack Option Switch 3 selects this option. The figure below shows the switch settings for selecting the line frequency.

60 Hz Operation

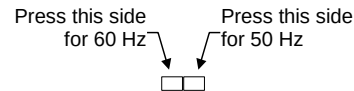


50 Hz Operation



How to Set the Frequency Switch

- Determine the frequency.
- Press the side of the switch shown in gray.



Notes

Only one **SCADAPack 5602 I/O module** may be assigned to the same controller. Option switches 1 and 3 function as described above regardless of whether the module, **DIN 5203/4 option switches**, is added to the Register Assignment or not. Refer to the **SCADAPack Hardware Manual** for information on the option switches.

6.11.7 SCADAPack 5604 I/O Module

Description

The **SCADAPack 5604 I/O Module** is comprised of four different types of I/O channels. The SCADAPack 5604 I/O hardware contains ten analog inputs, two analog outputs, thirty-five digital inputs and thirty-six digital outputs.

The **analog input** assignment provides ten I/O channels for the analog input data from the SCADAPack 5604 I/O Module hardware.

- Analog input channels 0 to 7 provide eight external analog inputs (0-10V or 0-40mA)
- Analog input channel 8 provides an external analog input for battery monitoring (0 to 32.768V)
- Analog input channel 9 provides an internal analog input for DC/DC converter monitoring.

The **analog output** assignment provides two I/O channels for the analog output data for the SCADAPack 5604 I/O Module hardware.

The **digital input** assignment provides thirty-five I/O channels for the digital input data from the SCADAPack 5604 I/O Module hardware.

The SCADAPack 5604 I/O Module provides thirty-two universal digital inputs or outputs. The inputs are for use with dry contacts such as switches and relay contacts.

- These are defined as digital input channels 0 to 31.
0 = contact open (LED off)
1 = contact closed (LED on)

The SCADAPack 5604 I/O Module also provides three internal digital inputs.

- Digital input channel 32 returns the DC/DC converter status.
0 = DC/DC converter off
1 = DC/DC converter on
- Digital input channel 33 returns the DC/DC converter over current status.
0 = Over current not detected
1 = Over current detected
- Digital output channel 34 returns the digital output mismatch status.
0 = No mismatch
1 = One or more digital outputs mismatch

The **digital output** assignment provides thirty-six I/O channels for the digital outputs of the SCADAPack 5604 I/O Module I/O hardware.

The SCADAPack 5604 I/O Module provides thirty-two universal digital inputs or outputs. Outputs are open-collector/open drain type.

- These are defined as digital output channels 0 to 31.
0 = output transistor off

1 = output transistor on

The SCADAPack 5604 I/O Module also provides two internal digital outputs.

- Digital output channel 32 is used to control the DC/DC converter.
0 = DC/DC converter off
1 = DC/DC converter on
- Digital output channel 33 is used to control the VLoop power supply.
0 = VLoop output off
1 = VLoop output on
- Digital output channels 34 and 35 control the SCADAPack 5604 I/O Module Analog Input filters.

Filter Setting	Digital Output 34	Digital Output 35
< 3 Hz	OFF	OFF
6 Hz	OFF	ON
11 Hz	ON	OFF
30 Hz	ON	ON

Register Assignment

Module	SCADAPack 5604 I/O Module	
Address	Fixed at 0	
Type	Coil Register	
Start	First register of any unused block of 36 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 35
Registers	36 coil registers	
Description	Digital outputs	
Type	Status Register	
Start	First register of any unused block of 35 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 34
Registers	35 status registers	
Description	Digital inputs	
Type	Input Register	
Start	First register of any unused block of 10 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +9
Registers	10 input registers	
Description	analog inputs	
Type	Holding Register	
Start	First register of any unused block of 2 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx +1
Registers	2 input registers	
Description	analog outputs	
Extended Parameters	None	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7
Start Register + 8	Digital output 8
Start Register + 9	Digital output 9
Start Register + 10	Digital output 10
Start Register + 11	Digital output 11
Start Register + 12	Digital output 12
Start Register + 13	Digital output 13
Start Register + 14	Digital output 14
Start Register + 15	Digital output 15
Start Register + 16	Digital output 16
Start Register + 17	Digital output 17
Start Register + 18	Digital output 18
Start Register + 19	Digital output 19
Start Register + 20	Digital output 20
Start Register + 21	Digital output 21
Start Register + 22	Digital output 22
Start Register + 23	Digital output 23
Start Register + 24	Digital output 24
Start Register + 25	Digital output 25
Start Register + 26	Digital output 26
Start Register + 27	Digital output 27
Start Register + 28	Digital output 28
Start Register + 29	Digital output 29
Start Register + 30	Digital output 30
Start Register + 31	Digital output 31
Start Register + 32	Digital output 32
Start Register + 33	Digital output 33
Start Register + 34	Digital output 34
Start Register + 35	Digital output 35

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3

Registers	Assignment to Module Hardware
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7
Start Register + 8	Digital input 8
Start Register + 9	Digital input 9
Start Register + 10	Digital input 10
Start Register + 11	Digital input 11
Start Register + 12	Digital input 12
Start Register + 13	Digital input 13
Start Register + 14	Digital input 14
Start Register + 15	Digital input 15
Start Register + 16	Digital input 16
Start Register + 17	Digital input 17
Start Register + 18	Digital input 18
Start Register + 19	Digital input 19
Start Register + 20	Digital input 20
Start Register + 21	Digital input 21
Start Register + 22	Digital input 22
Start Register + 23	Digital input 23
Start Register + 24	Digital input 24
Start Register + 25	Digital input 25
Start Register + 26	Digital input 26
Start Register + 27	Digital input 27
Start Register + 28	Digital input 28
Start Register + 29	Digital input 29
Start Register + 30	Digital input 30
Start Register + 31	Digital input 31
Start Register + 32	Digital input 32
Start Register + 33	Digital input 33
Start Register + 34	Digital input 34

Input Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

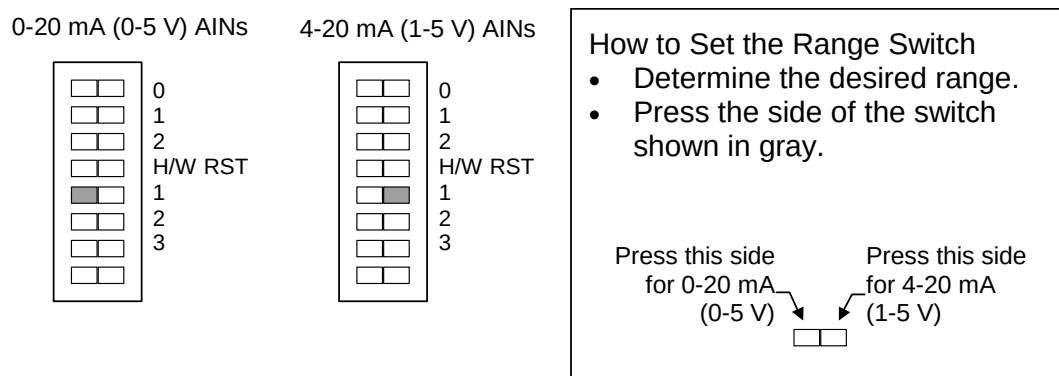
Holding Register Data

Registers	Assignment to Module Hardware
Start Register	Analog output 0
Start Register + 1	Analog output 1

Analog Input Measurement Range Selection

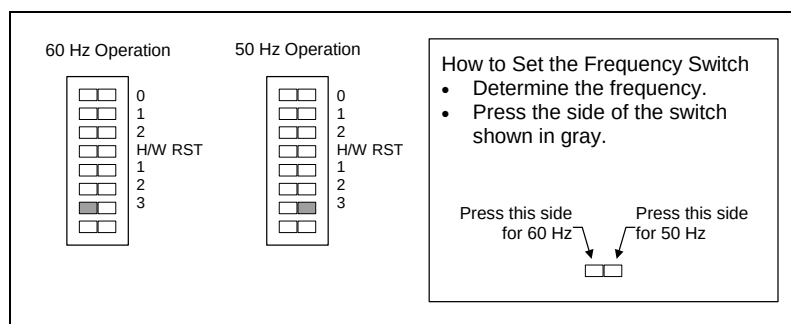
The SCADAPack Option Switch 1 selects the measurement range. All 8 analog inputs are set to the same range.

The figure below shows the switch settings for selecting the measurement range.



Line Frequency Selection

The SCADAPack I/O board may select 50 or 60 Hz line frequency for digital and analog input processing. The SCADAPack Option Switch 3 selects this option. The figure below shows the switch settings for selecting the line frequency.



Notes

Only one **SCADAPack 5601 or 5604 I/O module** may be assigned to the same controller.

Option switches 1 and 3 function as described above regardless of whether the module, **DIN 5203/4 option switches**, is added to the Register Assignment or not.

Refer to the **SCADAPack Hardware Manual** for information on the option switches.

Two analog outputs are available for the **SCADAPack 5601 I/O module**. Refer to the **SCADAPack AOOUT module** for more information.

6.11.8 SCADAPack 5606 I/O Module

Description

The **SCADAPack 5606 I/O Module** is comprised of four types of I/O channels, digital inputs, digital outputs, analog inputs, and analog outputs.

The **digital input** assignment provides forty I/O channels for the digital input data from the SCADAPack 5606 I/O Module hardware.

- The 5606 I/O Module provides thirty-two external digital inputs. These are externally wetted inputs.
- The module provides 8 internal digital inputs, which indicate if the corresponding analog input is in or out of range.

The **digital output** assignment provides sixteen digital outputs. All are digital output relays.

The **analog input** assignment provides eight I/O channels for the analog input data from the SCADAPack 5606 I/O Module hardware.

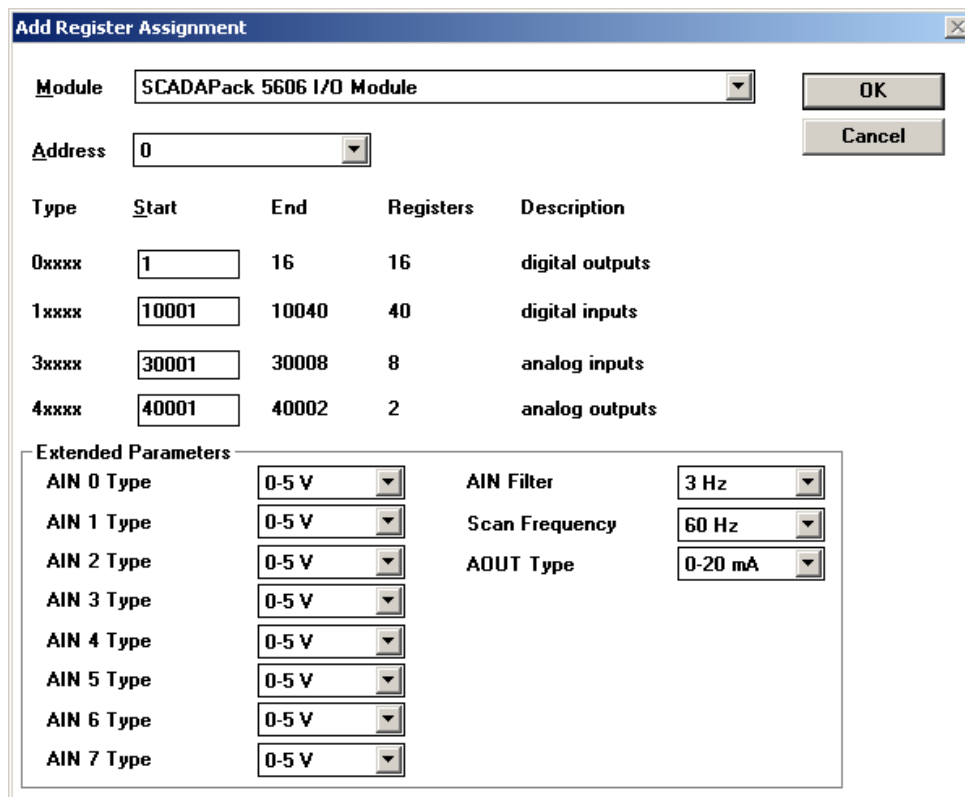
The **analog output** assignment provides two I/O channels for the analog output data for the SCADAPack 5606 I/O Module hardware when the optional analog output module is installed.

Register Assignment

The register assignment for the SCADAPack 5606 I/O Module is used to configure the registers used for the module data and the Extended Parameters available for the module. To open the register assignment dialog for the SCADAPack 5606 I/O module:

- Select **Register Assignment** from the **Controller** menu in TelePACE.
- In the **Register Assignment** window select the **Add** button.
- In the **Add Register Assignment** dialog select the **SCADAPack 5606 I/O Module** from the list presented in the **Module** window.

The register assignment dialog for the **SCADAPack 5606 I/O Module** is shown below.



Add Register Assignment

Module SCADAPack 5606 I/O Module

Address 0

OK **Cancel**

Type	Start	End	Registers	Description
0xxxx	1	16	16	digital outputs
1xxxx	10001	10040	40	digital inputs
3xxxx	30001	30008	8	analog inputs
4xxxx	40001	40002	2	analog outputs

Extended Parameters

AIN 0 Type	0-5 V	AIN Filter	3 Hz
AIN 1 Type	0-5 V	Scan Frequency	60 Hz
AIN 2 Type	0-5 V	AOUT Type	0-20 mA
AIN 3 Type	0-5 V		
AIN 4 Type	0-5 V		
AIN 5 Type	0-5 V		
AIN 6 Type	0-5 V		
AIN 7 Type	0-5 V		

The **Module** selection displays the name of the SCADAPack 5606 I/O module used for the register assignment.

The **Address** selection displays the module address of the physical SCADAPack 5606 I/O module. The address on the module is selected via dip switches on the module. A maximum of eight SCADAPack 5606 I/O type modules may be added to a system.

The register type **0xxxx** defines the register range for the digital output data. Sixteen sequential registers are needed for the SCADAPack 5606 I/O module digital output data. Any unused block of sixteen registers may be used. See the **Digital Output Register Data** section below for an explanation of the digital output data.

The register type **1xxxx** defines the register range for the digital input status data. Forty sequential registers are needed for the SCADAPack 5606 I/O module digital Input data. Any unused block of forty registers may be used. See the **Digital Input Register Data** section below for an explanation of the digital input status data.

The register type **3xxxx** defines the register range for the analog input data. Eight sequential registers are needed for the SCADAPack 5606 I/O module analog input data. Any unused block of eight registers may be used. See the **Input Register Data** section below for an explanation of the analog input data.

The register type **4xxxx** defines the register range for the analog output data. Two sequential registers are needed for the SCADAPack 5606 I/O module analog output data. Any unused block of two registers may be used. See the **Analog Output Register Data** section below for an explanation of the analog output data.

The **Extended Parameters** grouping defines the type of data returned from the SCADAPack 5606 I/O module analog inputs, the type of analog output signal, filtering used by all analog inputs and the scan frequency.

The **AIN Type** selection for each input (AIN 0, 1, 2, 3, 4, 5, 6, and 7) defines the input measurement type for the analog input.

- The **0-5V** selection sets the input type to measure 0 to 5V input signals.
- The **1-5V** selection sets the input type to measure 1 to 5V input signals.
- The **0-20 mA** selection sets the input type to measure 0 to 20mA input signals.
- The **4-20 mA** selection sets the input type to measure 4 to 20mA input signals.

The **AIN Filter** selection set the input filter for all analog inputs. Filtering is used to dampen process variations or noise.

- The **3 Hz** filter selection sets the response time to 155ms at 60Hz and 185ms at 50Hz.
- The **6 Hz** filter selection sets response time to 85ms at 60Hz and 85ms at 50Hz.
- The **11 Hz** filter selection sets response time to 45ms at 60Hz and 55ms at 50Hz.
- The **30 Hz** filter selection sets response time to 30ms at 60Hz and 30ms at 50Hz.

The **Scan Frequency** selection set the input scan rate for all analog inputs. The scan rate selection is not critical but AC noise rejection is improved at the correct frequency. If the module is used in a DC environment, the 60 Hz setting will yield slightly faster response time.

- The **60 Hz** selection synchronies the input scanning to 60Hz.
- The **50 Hz** selection synchronies the input scanning to 50Hz.

The **AOUT Type** selection for each input (AOUT 0 and 1) defines the output signal type for the analog output.

- The **0-20 mA** selection sets the output type for 0 to 20mA signals.
- The **4-20 mA** selection sets the output type for 4 to 20mA signals.

Digital Output Register Data

Digital output data is assigned to sixteen consecutive digital output (0xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment and continues for sixteen registers.

Registers	Assignment to Module Hardware
Start Register	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 1	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 2	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 3	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 4	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 5	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 6	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 7	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 8	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 9	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 10	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 11	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 12	0 = OFF (relay open); 1= ON (relay closed)

Registers	Assignment to Module Hardware
Start Register + 13	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 14	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 15	0 = OFF (relay open); 1= ON (relay closed)

Digital Input Register Data

Status register data is assigned to forty consecutive status (1xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment and continues for forty registers.

Registers	Assignment to Module Hardware
Start Register	0 = OFF (led off); 1= ON (led on)
Start Register + 1	0 = OFF (led off); 1= ON (led on)
Start Register + 2	0 = OFF (led off); 1= ON (led on)
Start Register + 3	0 = OFF (led off); 1= ON (led on)
Start Register + 4	0 = OFF (led off); 1= ON (led on)
Start Register + 5	0 = OFF (led off); 1= ON (led on)
Start Register + 6	0 = OFF (led off); 1= ON (led on)
Start Register + 7	0 = OFF (led off); 1= ON (led on)
Start Register + 8	0 = OFF (led off); 1= ON (led on)
Start Register + 9	0 = OFF (led off); 1= ON (led on)
Start Register + 10	0 = OFF (led off); 1= ON (led on)
Start Register + 11	0 = OFF (led off); 1= ON (led on)
Start Register + 12	0 = OFF (led off); 1= ON (led on)
Start Register + 13	0 = OFF (led off); 1= ON (led on)
Start Register + 14	0 = OFF (led off); 1= ON (led on)
Start Register + 15	0 = OFF (led off); 1= ON (led on)
Start Register + 16	0 = OFF (led off); 1= ON (led on)
Start Register + 17	0 = OFF (led off); 1= ON (led on)
Start Register + 18	0 = OFF (led off); 1= ON (led on)
Start Register + 19	0 = OFF (led off); 1= ON (led on)
Start Register + 20	0 = OFF (led off); 1= ON (led on)
Start Register + 21	0 = OFF (led off); 1= ON (led on)
Start Register + 22	0 = OFF (led off); 1= ON (led on)
Start Register + 23	0 = OFF (led off); 1= ON (led on)
Start Register + 24	0 = OFF (led off); 1= ON (led on)
Start Register + 25	0 = OFF (led off); 1= ON (led on)
Start Register + 26	0 = OFF (led off); 1= ON (led on)
Start Register + 27	0 = OFF (led off); 1= ON (led on)
Start Register + 28	0 = OFF (led off); 1= ON (led on)
Start Register + 29	0 = OFF (led off); 1= ON (led on)
Start Register + 30	0 = OFF (led off); 1= ON (led on)
Start Register + 31	0 = OFF (led off); 1= ON (led on)
Start Register + 32	OFF = AIN channel 0 is OK ON = AIN channel 0 is over or under range
Start Register + 33	OFF = AIN channel 1 is OK ON = AIN channel 1 is over or under range

Registers	Assignment to Module Hardware
Start Register + 34	OFF = AIN channel 2 is OK ON = AIN channel 2 is over or under range
Start Register + 35	OFF = AIN channel 3 is OK ON = AIN channel 3 is over or under range
Start Register + 36	OFF = AIN channel 4 is OK ON = AIN channel 4 is over or under range
Start Register + 37	OFF = AIN channel 5 is OK ON = AIN channel 5 is over or under range
Start Register + 38	OFF = AIN channel 6 is OK ON = AIN channel 6 is over or under range
Start Register + 39	OFF = AIN channel 7 is OK ON = AIN channel 7 is over or under range

Input Register Data

Input register data is assigned to eight consecutive analog input (3xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment and continues for eight registers.

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Analog Output Register Data

Analog output register data is assigned to two consecutive analog output (4xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment.

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1

Notes

Refer to the **SCADAPack Hardware Manual** for information on the SCADAPack 5606 I/O module.

6.11.9 SCADAPack 5607 I/O Module

Description

The **SCADAPack 5607 I/O Module** is comprised of four types of I/O channels, digital inputs, digital outputs, analog inputs, and analog outputs.

The **digital input** assignment provides forty I/O channels for the digital input data from the SCADAPack 5607 I/O Module hardware.

- The 5607 I/O Module provides sixteen external digital inputs. These are externally wetted inputs.
- The module provides 8 internal digital inputs, which indicate if the corresponding analog input is in or out of range.

The **digital output** assignment provides ten relay digital outputs.

The **analog input** assignment provides eight I/O channels for the analog input data from the SCADAPack 5607 I/O Module hardware.

The **analog output** assignment provides two I/O channels for the analog output data for the SCADAPack 5607 I/O Module hardware when the optional analog output module is installed.

Register Assignment

The register assignment for the SCADAPack 5607 I/O Module is used to configure the registers used for the module data and the Extended Parameters available for the module. To open the register assignment dialog for the SCADAPack 5607 I/O module:

- Select **Register Assignment** from the **Controller** menu in TelePACE.
- In the **Register Assignment** window select the **Add** button.
- In the **Add Register Assignment** dialog select the **SCADAPack 5606 I/O Module** from the list presented in the **Module** window.

The register assignment dialog for the **SCADAPack 5607 I/O Module** is shown below.

Add Register Assignment

Module: SCADAPack 5607 I/O module

Address: 0

OK Cancel

Type	Start	End	Registers	Description
0xxxx	1	10	10	digital outputs
1xxxx	10001	10024	24	digital inputs
3xxxx	30001	30008	8	analog inputs
4xxxx	40001	40002	2	analog outputs

Extended Parameters

AIN 0 Type	0-5 V	AIN Filter	3 Hz
AIN 1 Type	0-10 V	Scan Frequency	60 Hz
AIN 2 Type	4-20 mA	AOUT Type	0-20 mA
AIN 3 Type	4-20 mA		
AIN 4 Type	0-20 mA		
AIN 5 Type	0-10 V		
AIN 6 Type	0-5 V		
AIN 7 Type	0-5 V		

The **Module** selection displays the name of the SCADAPack 5607 I/O module used for the register assignment.

The **Address** selection displays the module address of the physical SCADAPack 5607 I/O module. The address on the module is selected via dip switches on the module. A maximum of eight SCADAPack 5607 I/O type modules may be added to a system.

The register type **0xxxx** defines the register range for the digital output data. Ten sequential registers are needed for the SCADAPack 5607 I/O module digital output data. Any unused block of ten registers may be used. See the **Digital Output Register Data** section below for an explanation of the digital output data.

The register type **1xxxx** defines the register range for the digital input status data. Twenty four sequential registers are needed for the SCADAPack 5607 I/O module digital Input data. Any unused block of twenty four registers may be used. See the **Digital Input Register Data** section below for an explanation of the digital input status data.

The register type **3xxxx** defines the register range for the analog input data. Eight sequential registers are needed for the SCADAPack 5607 I/O module analog input data. Any unused block of eight registers may be used. See the **Input Register Data** section below for an explanation of the analog input data.

The register type **4xxxx** defines the register range for the analog output data. Two sequential registers are needed for the SCADAPack 5607 I/O module analog output data. Any unused block of two registers may be used. See the **Analog Output Register Data** section below for an explanation of the analog output data.

The **Extended Parameters** grouping defines the type of data returned from the SCADAPack 5607 I/O module analog inputs, the type of analog output signal, filtering used by all analog inputs and the scan frequency.

The **AIN Type** selection for each input (AIN 0, 1, 2, 3, 4, 5, 6, and 7) defines the input measurement type for the analog input.

- The **0-5V** selection sets the input type to measure 0 to 5V input signals.
- The **1-5V** selection sets the input type to measure 1 to 5V input signals.
- The **0-20 mA** selection sets the input type to measure 0 to 20mA input signals.
- The **4-20 mA** selection sets the input type to measure 4 to 20mA input signals.

The **AIN Filter** selection set the input filter for all analog inputs. Filtering is used to dampen process variations or noise.

- The **3 Hz** filter selection sets the response time to 155ms at 60Hz and 185ms at 50Hz.
- The **6 Hz** filter selection sets response time to 85ms at 60Hz and 85ms at 50Hz.
- The **11 Hz** filter selection sets response time to 45ms at 60Hz and 55ms at 50Hz.
- The **30 Hz** filter selection sets response time to 30ms at 60Hz and 30ms at 50Hz.

The **Scan Frequency** selection set the input scan rate for all analog inputs. The scan rate selection is not critical but AC noise rejection is improved at the correct frequency. If the module is used in a DC environment, the 60 Hz setting will yield slightly faster response time.

- The **60 Hz** selection synchronies the input scanning to 60Hz.
- The **50 Hz** selection synchronies the input scanning to 50Hz.

The **AOUT Type** selection for each input (AOUT 0 and 1) defines the output signal type for the analog output.

- The **0-20 mA** selection sets the output type for 0 to 20mA signals.
- The **4-20 mA** selection sets the output type for 4 to 20mA signals.

Digital Output Register Data

Digital output data is assigned to sixteen consecutive digital output (0xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment and continues for sixteen registers.

Registers	Assignment to Module Hardware
Start Register	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 1	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 2	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 3	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 4	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 5	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 6	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 7	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 8	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 9	0 = OFF (relay open); 1= ON (relay closed)

Digital Input Register Data

Status register data is assigned to forty consecutive status (1xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment and continues for forty registers.

Registers	Assignment to Module Hardware
Start Register	0 = OFF (led off); 1= ON (led on)
Start Register + 1	0 = OFF (led off); 1= ON (led on)
Start Register + 2	0 = OFF (led off); 1= ON (led on)
Start Register + 3	0 = OFF (led off); 1= ON (led on)
Start Register + 4	0 = OFF (led off); 1= ON (led on)
Start Register + 5	0 = OFF (led off); 1= ON (led on)
Start Register + 6	0 = OFF (led off); 1= ON (led on)
Start Register + 7	0 = OFF (led off); 1= ON (led on)
Start Register + 8	0 = OFF (led off); 1= ON (led on)
Start Register + 9	0 = OFF (led off); 1= ON (led on)
Start Register + 10	0 = OFF (led off); 1= ON (led on)
Start Register + 11	0 = OFF (led off); 1= ON (led on)
Start Register + 12	0 = OFF (led off); 1= ON (led on)
Start Register + 13	0 = OFF (led off); 1= ON (led on)
Start Register + 14	0 = OFF (led off); 1= ON (led on)
Start Register + 15	0 = OFF (led off); 1= ON (led on)
Start Register + 16	OFF = AIN channel 0 is OK ON = AIN channel 0 is over or under range
Start Register + 17	OFF = AIN channel 1 is OK ON = AIN channel 1 is over or under range
Start Register + 18	OFF = AIN channel 2 is OK ON = AIN channel 2 is over or under range
Start Register + 19	OFF = AIN channel 3 is OK ON = AIN channel 3 is over or under range
Start Register + 20	OFF = AIN channel 4 is OK ON = AIN channel 4 is over or under range
Start Register + 21	OFF = AIN channel 5 is OK ON = AIN channel 5 is over or under range
Start Register + 22	OFF = AIN channel 6 is OK ON = AIN channel 6 is over or under range
Start Register + 23	OFF = AIN channel 7 is OK ON = AIN channel 7 is over or under range

Input Register Data

Input register data is assigned to eight consecutive analog input (3xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment and continues for eight registers.

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2

Registers	Assignment to Module Hardware
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Analog Output Register Data

Analog output register data is assigned to two consecutive analog output (4xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment.

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1

Notes

Refer to the **SCADAPack Hardware Manual** for information on the SCADAPack 5607 I/O module.

6.11.10 SCADAPack LP I/O

Description

The **SCADAPack LP I/O Module** is comprised of four different types of I/O channels. The SCADAPack LP I/O hardware contains eight analog inputs, two analog outputs, sixteen digital inputs and twelve digital outputs.

The **analog input** assignment provides seven I/O channels for the analog input data. Analog input data is a signed value in the range –32768 to 32767.

- Analog input channels 0 to 4 provide eight external analog inputs (0-10V or 0-40mA)
- Analog input channel 5 provides an external analog input for battery monitoring (0 to 32.768V)
- Analog input channel 6 provides an internal analog input for DC/DC converter monitoring.

The **analog output** assignment provides two I/O channels for the analog output data. Analog output data is a signed value in the range –32768 to 32767.

The **digital input** assignment provides sixteen I/O channels for the digital input data.

The SCADAPack LP provides eight universal digital inputs or outputs. The inputs are for use with dry contacts such as switches and relay contacts.

- These are defined as digital input channels 0 to 7.
 - 0 = contact open (LED off)
 - 1 = contact closed (LED on)

The SCADAPack LP also provides eight internal digital inputs.

- Digital input channel 8 returns the Com1 (RS-485) power status.
 - 0 = off
 - 1 = on
- Digital input channel 9 returns the Com3 (HMI) power output status.
 - 0 = off
 - 1 = on
- Digital input channel 10 returns the VLOOP output status.
 - 0 = off
 - 1 = on
- Digital input channel 11 returns the DC/DC converter status. This bit reports the true status of the DC/DC converter. If over-current causes the converter to be turned off, this bit will clear.
 - 0 = off
 - 1 = on
- Digital input channel 12 returns the VLOOP over-current status. This indicates VLOOP over-current has been detected. This input clears when VLOOP output is off, or the over-current condition clears.
 - 0 = off

1 = on

- Digital input channel 13 returns the digital output mismatch status.

0= No mismatch

1 = One or more digital outputs mismatch

- Digital input channel 14 returns the SCADAPack Vision power.

0 = No power to SCADAPack Vision

1 = Power to SCADAPack Vision

The **digital output** assignment provides twelve I/O channels for the digital outputs of the SCADAPack LP.

The SCADAPack LP provides eight universal digital inputs or outputs. Outputs are open-collector/open drain type.

- These are defined as digital output channels 0 to 7.

0 = output transistor off

1 = output transistor on

The SCADAPack LP also provides four internal digital outputs.

- Digital output channel 8 is not used. This is for internal use only.

- Digital output channel 9 is used to control com3 (HMI) power.

0= off

1 = on

- Digital output channel 10 is used to control the VLoop power supply.

0= VLoop output off

1= VLoop output on

- Digital output channel 11 is used to control the DC/DC converter.

0= off

1 = on

Register Assignment

Module	SCADAPack LP I/O Module	
Address	fixed	
Type	Coil Register	
Start	First register of any unused block of 12 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 11
Registers	12 coil registers	
Description	digital outputs	
Type	Status Register	
Start	First register of any unused block of 16 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 15

Module	SCADAPack LP I/O Module	
Address	fixed	
Registers	16 status registers	
Description	digital inputs	
Type	Input Register	
Start	First register of any unused block of 8 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +7
Registers	8 input registers	
Description	analog inputs	
Type	Holding Register	
Start	First register of any unused block of 2 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx +1
Registers	2 output registers	
Description	analog outputs	
Extended Parameters	None	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7
Start Register + 8	Digital output 8
Start Register + 9	Digital output 9
Start Register + 10	Digital output 10
Start Register + 11	Digital output 11

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7
Start Register + 8	Digital input 8
Start Register + 9	Digital input 9

Registers	Assignment to Module Hardware
Start Register + 10	Digital input 10
Start Register + 11	Digital input 11
Start Register + 12	Digital input 12
Start Register + 13	Digital input 13
Start Register + 14	Digital input 14
Start Register + 15	Digital input 15

Input Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Analog Output Register Data

Registers	Assignment to Module Hardware
Start Register	Analog output 0
Start Register + 1	Analog output 1

Notes

Only one SCADAPack LP I/O may be assigned to the same controller.

Refer to the SCADAPack LP Hardware Manual for information on analog and digital I/O usage.

6.11.11 SCADAPack 100 I/O

Description

Digital output data is assigned from six consecutive coil registers. The digital outputs are updated continuously with data read from the coil registers.

Digital input data is assigned to six consecutive status registers. The status registers are updated continuously with data read from the digital inputs.

Analog and counter input data is assigned to eight consecutive input registers. There are six analog input registers, and a counter input double register (32 bit). These input registers are updated continuously with data read from the analog inputs and counter input.

Register Assignment

Module	SCADAPack 100 I/O Module	
Address	Fixed	
Type	Coil Register	
Start	First register of any unused block of 6 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 5
Registers	6 coil registers	
Description	digital outputs	
Type	Status Register	
Start	First register of any unused block of 6 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 5
Registers	8 status registers	
Description	digital inputs	
Type	Input Register	
Start	First register of any unused block of 4 consecutive input registers.	3xxxx
End	Last register of block	3xxxx + 8
Registers	8 input registers	
Description	analog inputs	
Extended Parameters	None	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6

Analog and Counter Input Register Data

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4 Board Temperature (°C)
Start Register + 5	Analog input 5 Ram Battery Voltage (mV)
Start Register + 6 and + 7	Counter input, 32 bit register

Notes

Only one **SCADAPack 100 I/O** may be assigned to the same controller.

Refer to the **SCADAPack 100 Hardware Manual** for information on analog and digital I/O usage.

6.11.12 SCADAPack 330 I/O

Description

The **SCADAPack 330 I/O** is comprised of three types of I/O channels, three analog inputs (counters), one digital input and two digital outputs.

The **counter input** assignment provides three I/O channels for the analog input data.

- Counter input channel 0 is for counter input 0. This is a 32 bit value with automatic rollover.
- Counter input channel 1 is for counter input 1. This is a 32 bit value with automatic rollover.
- Counter input channel 2 is for counter input 2. This is a 32 bit value with automatic rollover.

The SCADAPack 330 provides one digital input for digital input data from the SCADAPack 330 I/O hardware.

- Digital input channel 0 is the status of COM3 HMI power.
0 = 5V at pin 1 of connector P7 (COM3) is **off**.
1 = 5V at pin 1 of connector P7 (COM3) is **on**.

The **digital output** assignment provides two I/O channels for the digital outputs of the SCADAPack 330 I/O hardware.

- Digital output channel 0 controls the USB STAT led.
0 = USB STAT led **off**.
1 = USB STAT led **on**.
- Digital output channel 1 controls the COM3 HMI power.
0 = 5V at pin 1 of connector P7 (COM3) is **off**.
1 = 5V at pin 1 of connector P7 (COM3) is **on**.

Register Assignment

Module	SCADAPack 330 I/O	
Address	fixed	
Type	Coil Register	
Start	First register of any unused block of 2 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 1
Registers	2 coil registers	
Description	digital outputs	
Type	Status Register	
Start	First register of any unused block of 1 consecutive status registers.	1xxxx
End	Last register of block	1xxxx
Registers	1status registers	
Description	digital inputs	
Type	Input Register	
Start	First register of any unused block of 6 consecutive input registers.	3xxxx

End	Last register of block	3xxxx +5
Registers	3 counter input registers	
Description	counter inputs	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	USB LED output
Start Register + 1	Com3 (HMI) power control

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Com3 (HMI) power status

Input Register Data

Registers	Assignment to Module Hardware
Start Register Start Register + 1	Counter input 0 (32 bit-register)
Start Register + 2 Start Register + 3	Counter input 1 (32 bit-register)
Start Register + 4 Start Register + 5	Counter input 2 (32 bit-register)

Notes

Only one **SCADAPack 330 I/O** may be assigned to the same controller.

Refer to the **SCADAPack 330 Hardware Manual** for information on counter and digital I/O usage.

6.11.13 SCADAPack 334 I/O

Description

The SCADAPack 334 I/O is comprised of a SCADAPack 330 I/O module and a 5607 I/O module. The SCADAPack 334 I/O provides for five types of I/O channels, 8 analog inputs, 3 counter inputs, 25 digital inputs, 12 digital outputs and 2 analog outputs.

- See the SCADAPack 330 I/O register assignment for a complete description of the SCADAPack 330 I/O.
- See the 5607 I/O module register assignment for a complete description of the 5607 I/O.

Register Assignment

Module	SCADAPack 33x I/O	
Address	fixed	
Type	Coil Register	
Start	First register of any unused block of 12 consecutive coil registers.	0xxxx
End	Second register of block	0xxxx + 1
Registers	2 coil registers	
Description	digital outputs	
Type	Status Register	
Start	First register of any unused block of 25 consecutive status registers.	1xxxx
End	First register of block	1xxxx
Registers	1status registers	
Description	digital inputs	
Type	Input Register	
Start	First register of any unused block of 14 consecutive input registers.	3xxxx
End	Fifth register of block	3xxxx +5
Registers	3 counter input registers	
Description	counter inputs	

Module	5607 I/O	
Address	0 to 7	
Type	Coil Register	
Start	Third register of any unused block of 12 consecutive coil registers.	0xxxx + 2
End	Last register of block	0xxxx + 12
Registers	10 coil registers	
Description	digital outputs	
Type	Status Register	
Start	Second register of any unused block of 25 consecutive status registers.	1xxxx + 1
End	Last register of block	1xxxx + 24
Registers	24 status registers	

Description	digital inputs	
Type	Input Register	
Start	Sixth register of any unused block of 14 consecutive input registers.	3xxxx + 6
End	Last register of block	3xxxx + 13
Registers	8 analog input registers	
Description	analog inputs	
Type	Holding Register	
Start	First register of any unused block of 6 consecutive input registers.	3xxxx + 6
End	Last register of block	3xxxx + 13
Registers	2 analog output	
Description	analog outputs	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	USB LED output
Start Register + 1	Com3 (HMI) power control
Start Register + 2	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 3	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 4	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 5	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 6	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 7	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 8	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 9	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 10	0 = OFF (relay open); 1= ON (relay closed)
Start Register + 11	0 = OFF (relay open); 1= ON (relay closed)

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Com3 (HMI) power status
Start Register + 1	0 = OFF (led off); 1= ON (led on)
Start Register + 2	0 = OFF (led off); 1= ON (led on)
Start Register + 3	0 = OFF (led off); 1= ON (led on)
Start Register + 4	0 = OFF (led off); 1= ON (led on)
Start Register + 5	0 = OFF (led off); 1= ON (led on)
Start Register + 6	0 = OFF (led off); 1= ON (led on)
Start Register + 7	0 = OFF (led off); 1= ON (led on)
Start Register + 8	0 = OFF (led off); 1= ON (led on)
Start Register + 9	0 = OFF (led off); 1= ON (led on)
Start Register + 10	0 = OFF (led off); 1= ON (led on)
Start Register + 11	0 = OFF (led off); 1= ON (led on)
Start Register + 12	0 = OFF (led off); 1= ON (led on)

Registers	Assignment to Module Hardware
Start Register + 13	0 = OFF (led off); 1= ON (led on)
Start Register + 14	0 = OFF (led off); 1= ON (led on)
Start Register + 15	0 = OFF (led off); 1= ON (led on)
Start Register + 16	0 = OFF (led off); 1= ON (led on)
Start Register + 17	OFF = AIN channel 0 is OK ON = AIN channel 0 is over or under range
Start Register + 18	OFF = AIN channel 1 is OK ON = AIN channel 1 is over or under range
Start Register + 19	OFF = AIN channel 2 is OK ON = AIN channel 2 is over or under range
Start Register + 20	OFF = AIN channel 3 is OK ON = AIN channel 3 is over or under range
Start Register + 21	OFF = AIN channel 4 is OK ON = AIN channel 4 is over or under range
Start Register + 22	OFF = AIN channel 5 is OK ON = AIN channel 5 is over or under range
Start Register + 23	OFF = AIN channel 6 is OK ON = AIN channel 6 is over or under range
Start Register + 24	OFF = AIN channel 7 is OK ON = AIN channel 7 is over or under range

Input Register Data

Registers	Assignment to Module Hardware
Start Register Start Register + 1	Counter input 0 (32 bit-register)
Start Register + 2 Start Register + 3	Counter input 1 (32 bit-register)
Start Register + 4 Start Register + 5	Counter input 2 (32 bit-register)
Start Register + 6	Analog input 0
Start Register + 7	Analog input 1
Start Register + 8	Analog input 2
Start Register + 9	Analog input 3
Start Register + 10	Analog input 4
Start Register + 11	Analog input 5
Start Register + 12	Analog input 6
Start Register + 13	Analog input 7

Analog Output Register Data

Analog output register data is assigned to two consecutive analog output (4xxxx) registers. The following table begins at the **Start** address defined in the Register Assignment.

Registers	Assignment to Module Hardware
Start Register	Analog output 0

Registers	Assignment to Module Hardware
Start Register + 1	Analog output 1

6.11.14 SCADAPack 350 I/O

Description

The **SCADAPack 350 I/O** is comprised of four different types of I/O channels: eight analog inputs, two analog outputs, thirteen digital inputs and eleven digital outputs.

The **analog input** assignment provides eight I/O channels for the analog input data.

- Analog input channels 0 to 4 provide five external analog inputs (0-10V or 0-40mA)
- Analog input channel 5 provides an external analog input for battery monitoring (0 to 32.768V)
- Analog input channel 6 provides an internal analog input for DC/DC converter monitoring.
- Analog input channel 7 is used internally by the SCADAPack 350.

The **analog output** assignment provides two I/O channels for the analog output data.

The **digital input** assignment provides thirteen I/O channels for the digital input data.

The SCADAPack 350 provides eight universal digital inputs or outputs. The inputs are for use with dry contacts such as switches and relay contacts.

- These are defined as digital input channels 0 to 7.

0 = contact open (LED off)

1 = contact closed (LED on)

The SCADAPack 350 I/O hardware also provides five internal digital inputs.

- Digital input channel 8 returns the VLOOP output status.
0 = off
1 = on
- Digital input channel 9 returns the DC/DC converter status. This bit reports the true status of the DC/DC converter. If over-current causes the converter to be turned off, this bit will clear.
0 = off
1 = on
- Digital input channel 10 returns the VLOOP over-current status. This indicates VLOOP over-current has been detected. This input clears when VLOOP output is off, or the over-current condition clears.
0 = off
1 = on
- Digital input channel 11 returns the digital output mismatch status.

Known outputs are compared to the corresponding inputs to detect incorrect outputs. A point is compared if it has been turned on at any time since controller reset. This input indicates if one or more outputs mismatch. The source of the mismatch can be determined by comparing each digital input against the corresponding digital output.

The SCADAPack 350 on board digital I/O can be inputs or outputs. There is no configuration for the type, the SCADAPack 350 accepts digital inputs on a point and will write a digital output if programmed.

For example if D I/O point 0 is assigned registers 10001 as an input it is also assigned as digital output 00001. If the application turns on 00001 then 10001 will be seen as ON. While this

flexibility is useful it can cause problems in an application if users inadvertently turn ON a DIO that is used as an input in an application.

The internal DI 11 is turned ON if the SCADAPack 350 detects that a point has been turned on as an input and then turned on as a digital output in the application. Each time the controller is reset it begins the internal test again.

0 = No mismatch

1 = One or more digital outputs mismatch

- Digital input channel 12 returns the Com3 (HMI) power control status.
0 = off
1 = on

The **digital output** assignment provides eleven I/O channels for the digital outputs of the SCADAPack 350.

The SCADAPack 350 provides eight universal digital inputs or outputs. Outputs are open-collector/open drain type.

- These are defined as digital output channels 0 to 7.
0 = output transistor off
1 = output transistor on

The SCADAPack 350 I/O hardware also provides three internal digital outputs.

- Digital output channel 8 is used to control the VLoop power supply.
0 = VLoop output off
1 = VLoop output on
- Digital output channel 9 is used to control the DC/DC converter.
0 = off
1 = on
- Digital output channel 10 is used to control the Com3 (HMI) power.
0 = off
1 = on

Register Assignment

Module	SCADAPack 350 I/O	
Address	fixed	
Type	Coil Register	
Start	First register of any unused block of 11 consecutive coil registers.	0xxxx
End	Last register of block	0xxxx + 9
Registers	10 coil registers	
Description	digital outputs	
Type	Status Register	
Start	First register of any unused block of 13 consecutive status registers.	1xxxx
End	Last register of block	1xxxx + 11
Registers	12 status registers	
Description	digital inputs	
Type	Input Register	

Start	First register of any unused block of 8 consecutive input registers.	3xxxx
End	Last register of block	3xxxx +7
Registers	8 input registers	
Description	analog inputs	
Type	Holding Register	
Start	First register of any unused block of 2 consecutive holding registers.	4xxxx
End	Last register of block	4xxxx +1
Registers	2 output registers	
Description	analog outputs	

Coil Register Data

Registers	Assignment to Module Hardware
Start Register	Digital output 0
Start Register + 1	Digital output 1
Start Register + 2	Digital output 2
Start Register + 3	Digital output 3
Start Register + 4	Digital output 4
Start Register + 5	Digital output 5
Start Register + 6	Digital output 6
Start Register + 7	Digital output 7
Start Register + 8	Digital output 8
Start Register + 9	Digital output 9
Start Register + 10	Digital output 10

Status Register Data

Registers	Assignment to Module Hardware
Start Register	Digital input 0
Start Register + 1	Digital input 1
Start Register + 2	Digital input 2
Start Register + 3	Digital input 3
Start Register + 4	Digital input 4
Start Register + 5	Digital input 5
Start Register + 6	Digital input 6
Start Register + 7	Digital input 7
Start Register + 8	Digital input 8
Start Register + 9	Digital input 9
Start Register + 10	Digital input 10
Start Register + 11	Digital input 11
Start Register + 12	Digital input 12

Input Register Data

Registers	Assignment to Module Hardware

Registers	Assignment to Module Hardware
Start Register	Analog input 0
Start Register + 1	Analog input 1
Start Register + 2	Analog input 2
Start Register + 3	Analog input 3
Start Register + 4	Analog input 4
Start Register + 5	Analog input 5
Start Register + 6	Analog input 6
Start Register + 7	Analog input 7

Analog Output Register Data

Registers	Assignment to Module Hardware
Start Register	Analog output 0
Start Register + 1	Analog output 1

Notes

Only one **SCADAPack 350 I/O** may be assigned to the same controller.

Refer to the **SCADAPack 350 Hardware Manual** for information on analog and digital I/O usage.

6.12 Controller Default Register Assignments

6.12.1 4202 DR Extended/4203 DR I/O Register Assignment

A default Register Assignment is provided for the SCADASense series of controllers. The table below provides the default register assignment for these controllers.

To enable the default register assignment for a SCADASense controller:

- Select **Type** from the **Controller** menu.
- Select the appropriate SCADASense controller.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment* dialog.
- If the controller type selected is a SCADASense 4202 DR, the user is presented with the default register assignment for a 4202 DR and 4202 DR Extended/4203 DR. Select the appropriate module depending on your controller type.

Module	Address	Default Register Assignment
SCADASense 4202 DR I/O	Fixed	
digital inputs		10001 to 10001
analog inputs		30001 to 30005
analog outputs		40500 to 40500
SCADASense 4202 DR Extended/4203 DR I/O	Fixed	
Digital output		00001 to 00001
digital inputs		10001 to 10001
analog inputs		30001 to 30005
analog outputs		40500 to 40500

Note: Note that the SCADASense 4202 DR Extended/4203 DR I/O module, while the default for the SCADASense 4203 DR, applies to the SCADASense 4202 DR series controllers manufactured after July 13, 2004.

6.12.2 SCADASense 4202/4203 DS I/O Default Register Assignment

A default Register Assignment is provided for the SCADASense DS series of controllers (4202 or 4203 DR). The table below contains a convenient assignment for the 4202 DR and 4203 DR controllers.

To enable the default register assignment:

- Select **Type** from the **Controller** menu.
- Select the appropriate SCADASense DS series controller.
- Select **Register Assignment** from the **Controller** menu.

Select the **Default** button in the *Register Assignment* dialog.

The following table summarizes the Default Register Assignment for the SCADAPack.

Module	Address	Default Register Assignment
SCADASense 4202/4203 DS I/O	Fixed	
Digital output		00001 to 00002
digital inputs		10001 to 10001
analog inputs		30001 to 30007

6.12.3 Micro16 Default Register Assignment (Backwards Compatible Modules)

The purpose of the Default Register Assignment is to provide backward compatibility to programs written for older TeleSAFE Micro16 controller firmware (versions 1.22 or earlier).

Instead of configurable Register Assignment older versions of the firmware use fixed mapping of the I/O hardware to the I/O database. The Default Register Assignment assigns all possible I/O modules according the previously fixed mapping used by the older firmware.

Note 1: It is recommended that all unused modules in the default table be deleted in order to optimize performance of the I/O scan.

Note 2: Only those I/O modules that were available to the older firmware are assigned in the default table.

To enable the default register assignment:

- Select **Type** from the **Controller** menu.
- Select **Micro16**.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment dialog*.
- Select Backwards Compatible Modules in the *Select Register Assignment Option dialog*.
- Use the **Delete** button to delete from the default table all modules that are not needed in your program.

The Default Register Assignment results in the following unassigned I/O database registers:

Coil Registers	01025 to 04096
Status Registers	11025 to 14096
Input Registers	30513 to 31024
Holding Register	41712 to 49999

These registers may be used as general purpose registers in ladder logic programs.

6.12.3.1 Default Analog Input Module Assignment

Module Type	Module Address	Default Register Assignment
AIN Generic 8 point	0	30001 to 30008
AIN Generic 8 point	1	30009 to 30016
AIN Generic 8 point	2	30017 to 30024
AIN Generic 8 point	3	30025 to 30032
AIN Generic 8 point	4	30033 to 30040
AIN Generic 8 point	5	30041 to 30048
AIN Generic 8 point	6	30049 to 30056
AIN Generic 8 point	7	30057 to 30064
AIN 5203/4 temperature	N/A	30193 to 30194
AIN 5203/4 RAM battery voltage	N/A	30195

6.12.3.2 Default Analog Output Module Assignment

Module Type	Module Address	Default Register Assignment
AOUT Generic 2 point	0	40001 to 40002

Module Type	Module Address	Default Register Assignment
AOUT Generic 2 point	1	40003 to 40004
AOUT Generic 2 point	2	40005 to 40006
AOUT Generic 2 point	3	40007 to 40008
AOUT Generic 2 point	4	40009 to 40010
AOUT Generic 2 point	5	40011 to 40012
AOUT Generic 2 point	6	40013 to 40014
AOUT Generic 2 point	7	40015 to 40016
AOUT Generic 2 point	8	40017 to 40018
AOUT Generic 2 point	9	40019 to 40020
AOUT Generic 2 point	10	40021 to 40022
AOUT Generic 2 point	11	40023 to 40024
AOUT Generic 2 point	12	40025 to 40026
AOUT Generic 2 point	13	40027 to 40028
AOUT Generic 2 point	14	40029 to 40030
AOUT Generic 2 point	15	40031 to 40032

6.12.3.3 Default Configuration Module Assignment

Module Type	Module Address	Default Register Assignment
CNFG Clear serial port counters	0	00209
CNFG Clear serial port counters	1	00210
CNFG Clear protocol counters	0	00217
CNFG Clear protocol counters	1	00218
CNFG Save settings to EEPROM	N/A	00225
CNFG LED power settings	N/A	
• default state		00242
• time to return to default state		40211
CNFG Serial port settings	0	40131 to 40139
CNFG Protocol settings	0	40140 to 40142
CNFG Serial port settings	1	40146 to 40154
CNFG Protocol settings	1	40155 to 40157
CNFG Real time clock and alarm	N/A	40191 to 40201
CNFG PID control block	0	40220 to 40244
CNFG PID control block	1	40250 to 40274
CNFG PID control block	2	40280 to 40304
CNFG PID control block	3	40310 to 40334
CNFG PID control block	4	40340 to 40364
CNFG PID control block	5	40370 to 40394
CNFG PID control block	6	40400 to 40424
CNFG PID control block	7	40430 to 40454
CNFG PID control block	8	40460 to 40484
CNFG PID control block	9	40490 to 40514
CNFG PID control block	10	40520 to 40544
CNFG PID control block	11	40550 to 40574
CNFG PID control block	12	40580 to 40604
CNFG PID control block	13	40610 to 40634

Module Type	Module Address	Default Register Assignment
CNFG PID control block	14	40640 to 40664
CNFG PID control block	15	40670 to 40694
CNFG PID control block	16	40700 to 40724
CNFG PID control block	17	40730 to 40754
CNFG PID control block	18	40760 to 40784
CNFG PID control block	19	40790 to 40814
CNFG PID control block	20	40820 to 40844
CNFG PID control block	21	40850 to 40874
CNFG PID control block	22	40880 to 40904
CNFG PID control block	23	40910 to 40934
CNFG PID control block	24	40940 to 40964
CNFG PID control block	25	40970 to 40994
CNFG PID control block	26	41000 to 41024
CNFG PID control block	27	41030 to 41054
CNFG PID control block	28	41060 to 41084
CNFG PID control block	29	41090 to 41114
CNFG PID control block	30	41120 to 41144
CNFG PID control block	31	41150 to 41174
CNFG Store and forward Translation table Clear store and forward table	N/A	41200 to 41711 00226

6.12.3.4 Default Counter Input Module Assignment

Module Type	Module Address	Default Register Assignment
CNTR 5410 input	0	30201 to 30208
CNTR 5410 input	1	30209 to 30216
CNTR 5410 input	2	30217 to 30224
CNTR 5410 input	3	30225 to 30232
CNTR 5410 input	4	30233 to 30240
CNTR 5410 input	5	30241 to 30248
CNTR 5410 input	6	30249 to 30256
CNTR 5410 input	7	30257 to 30264
CNTR 5410 input	8	30265 to 30272
CNTR 5410 input	9	30273 to 30280
CNTR 5410 input	10	30281 to 30288
CNTR 5410 input	11	30289 to 30296
CNTR 5410 input	12	30297 to 30304
CNTR 5410 input	13	30305 to 30312
CNTR 5410 input	14	30313 to 30320
CNTR 5410 input	15	30321 to 30328
CNTR 5203/4 counter inputs	N/A	30329 to 30334

6.12.3.5 Default Diagnostic Module Assignment

Module Type	Module Address	Default Register Assignment
DIAG Controller status code	N/A	30196
DIAG Serial port comm. Status	0	30401 to 30405
DIAG Serial port protocol status	0	30406 to 30415
DIAG Serial port comm. Status	1	30421 to 30425
DIAG Serial port protocol status	1	30426 to 30435

6.12.3.6 Default Digital Input Module Assignment

Module Type	Module Address	Default Register Assignment
DIN Generic 8 point	0	10001 to 10008
DIN Generic 8 point	1	10009 to 10016
DIN Generic 8 point	2	10017 to 10024
DIN Generic 8 point	3	10025 to 10032
DIN Generic 8 point	4	10033 to 10040
DIN Generic 8 point	5	10041 to 10048
DIN Generic 8 point	6	10049 to 10056
DIN Generic 8 point	7	10057 to 10064
DIN Generic 16 point	8	10065 to 10080
DIN Generic 16 point	9	10081 to 10096
DIN Generic 16 point	10	10097 to 10112
DIN Generic 16 point	11	10113 to 10128
DIN Generic 16 point	12	10129 to 10144
DIN Generic 16 point	13	10145 to 10160
DIN Generic 16 point	14	10161 to 10176
DIN Generic 16 point	15	10177 to 10192
DIN 5203/4 digital inputs	N/A	10193 to 10195
DIN 5203/4 option switches	N/A	10196 to 10198
DIN 5203/4 interrupt input	N/A	10199

6.12.3.7 Default Digital Output Module Assignment

Module Type	Module Address	Default Register Assignment
DOUT Generic 8 point	0	00001 to 00008
DOUT Generic 8 point	1	00009 to 00016
DOUT Generic 8 point	2	00017 to 00024
DOUT Generic 8 point	3	00025 to 00032
DOUT Generic 8 point	4	00033 to 00040
DOUT Generic 8 point	5	00041 to 00048
DOUT Generic 8 point	6	00049 to 00056
DOUT Generic 8 point	7	00057 to 00064
DOUT Generic 16 point	8	00065 to 00080
DOUT Generic 16 point	9	00081 to 00096
DOUT Generic 16 point	10	00097 to 00112
DOUT Generic 16 point	11	00113 to 00128
DOUT Generic 16 point	12	00129 to 00144

Module Type	Module Address	Default Register Assignment
DOUT Generic 16 point	13	00145 to 00160
DOUT Generic 16 point	14	00161 to 00176
DOUT Generic 16 point	15	00177 to 00192

6.12.4 Micro16 Default Register Assignment (Controller I/O Only)

A default Register Assignment is provided for the Micro16 controller. The table contains a convenient assignment for the Micro16 controller board only.

To enable the SCADAPack LIGHT default register assignment:

- Select **Type** from the **Controller** menu.
- Select **Micro16**.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment dialog*.
- Select Controller I/O Only in the *Select Register Assignment Option dialog*.

The following table summarizes the Default Register Assignment for the Micro16.

Module	Address	Default Register Assignment
DIN Controller digital inputs		10001 to 10003
DIN Controller interrupt input		10004 to 10004

6.12.5 SCADAPack (5601 I/O Module) Default Register Assignment

A default Register Assignment is provided for the SCADAPack controller. The table contains a convenient assignment for the SCADAPack controller and 5601 I/O module.

To enable the SCADAPack default register assignment:

- Select **Type** from the **C**ontroller menu.
- Select **SCADAPack**.
- Select **Register Assignment** from the **C**ontroller menu.
- Select the **D**efault button in the *Register Assignment* dialog.
- Select 5601 I/O Module in the *Select Register Assignment Option* dialog.

The following table summarizes the Default Register Assignment for the SCADAPack.

See the

SCADAPack 5601 I/O Module section for information on the registers used.

Module	Address	Default Register Assignment
DIN Controller digital inputs		10017 to 10019
DIN Controller interrupt input		10020 to 10020
SCADAPack 5601 I/O module digital outputs digital inputs analog inputs	Fixed	00001 to 00012 10001 to 10016 30001 to 30008

6.12.6 SCADAPack (5604 I/O Module) Default Register Assignment

A default Register Assignment is provided for the SCADAPack controller. The table contains a convenient assignment for the SCADAPack controller and 5604 I/O module.

To enable the SCADAPack default register assignment:

- Select **Type** from the **C**ontroller menu.
- Select **SCADAPack**.
- Select **Register Assignment** from the **C**ontroller menu.
- Select the **D**efault button in the *Register Assignment* dialog.
- Select 5604 I/O Module in the *Select Register Assignment Option* dialog.

The following table summarizes the Default Register Assignment for the SCADAPack.

See the

SCADAPack 5604 I/O Module section for information on the registers used.

Module	Address	Default Register Assignment
DIN Controller digital inputs		10017 to 10019
DIN Controller interrupt input		10020 to 10020
SCADAPack 5604 I/O module digital outputs digital inputs analog inputs analog outputs	Fixed	00001 to 00036 10001 to 10035 30001 to 30010 40001 to 40002

6.12.7 SCADAPack LIGHT Default Register Assignment

A default Register Assignment is provided for the SCADAPack LIGHT controller. The table contains a convenient assignment for the SCADAPack controller and 5602 I/O module.

To enable the SCADAPack LIGHT default register assignment:

- Select **Type** from the **C**ontroller menu.
- Select **SCADAPack LIGHT**.
- Select **Register Assignment** from the **C**ontroller menu.
- Select the **D**efault button in the *Register Assignment dialog*.

The following table summarizes the Default Register Assignment for the SCADAPack LIGHT.

See the

SCADAPack 5602 I/O Module section for information on the registers used.

Module	Address	Default Register Assignment
DIN Controller digital inputs		10006 to 10008
DIN Controller interrupt input		10009 to 10009
SCADAPack 5602 I/O module digital outputs digital inputs analog inputs	Fixed	00001 to 00002 10001 to 10005 30001 to 30005

6.12.8 SCADAPack PLUS (5601 I/O Module) Default Register Assignment

A default Register Assignment is provided for the SCADAPack PLUS controller. The table contains a convenient assignment for the SCADAPack controller the 5601I/O module and 5602 I/O module.

To enable the SCADAPack PLUS default register assignment:

- Select **Type** from the **Controller** menu.
- Select **SCADAPack PLUS**.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment* dialog.
- Select 5601 I/O Module in the *Select Register Assignment Option* dialog.

The following table summarizes the Default Register Assignment for the SCADAPack PLUS.

See the

SCADAPack 5601 I/O Module and

SCADAPack 5602 I/O Module sections for information on the registers used.

Module	Address	Default Register Assignment
DIN Controller digital inputs		10022 to 10024
DIN Controller interrupt input		10025 to 10025
SCADAPack 5601 I/O module digital outputs digital inputs analog inputs	fixed	00001 to 00012 10001 to 10016 30001 to 30008
SCADAPack 5602 I/O module digital outputs digital inputs analog inputs	Fixed	00013 to 00014 10017 to 10021 30009 to 30013

6.12.9 SCADAPack Plus (5604 I/O Module) Default Register Assignment

A default Register Assignment is provided for the SCADAPack controller. The table contains a convenient assignment for the SCADAPack controller and 5604 I/O module.

To enable the SCADAPack default register assignment:

- Select **Type** from the **C**ontroller menu.
- Select **SCADAPack Plus**.
- Select **Register Assignment** from the **C**ontroller menu.
- Select the **D**efault button in the *Register Assignment* dialog.
- Select 5604 I/O Module in the *Select Register Assignment Option* dialog.

The following table summarizes the Default Register Assignment for the SCADAPack. See the

SCADAPack 5604 I/O Module section for information on the registers used.

Module	Address	Default Register Assignment
DIN Controller digital inputs		10041 to 10043
DIN Controller interrupt input		10044 to 10044
SCADAPack 5604 I/O module digital outputs digital inputs analog inputs analog outputs	Fixed	00001 to 00036 10001 to 10035 30001 to 30010 40001 to 40002
SCADAPack 5602 I/O module digital outputs digital inputs analog inputs	Fixed	00037 to 00038 10036 to 10040 30011 to 30015

6.12.10 SCADAPack LP Default Register Assignment

A default Register Assignment is provided for the SCADAPack LP controller. The table contains a convenient assignment for the SCADAPack LP controller.

To enable the SCADAPack LP default register assignment:

- Select **Type** from the **Controller** menu.
- Select **SCADAPack LP**.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment dialog*.

The following table summarizes the Default Register Assignment for the SCADAPack LP.

See the

SCADAPack LP I/O section for information on the registers used.

Module	Address	Default Register Assignment
CNTR Controller counter inputs		30009 to 30014
SCADAPack LP I/O	fixed	
digital outputs		00001 to 00012
digital inputs		10001 to 10016
analog inputs		30001 to 30008
analog outputs		40001 to 40002

6.12.11 SCADAPack 330 Default Register Assignment

A default Register Assignment is provided for the SCADAPack 330 controller. The table contains a convenient assignment for the SCADAPack 330 controller.

To enable the SCADAPack 330 default register assignment:

- Select **Type** from the **Controller** menu.
- Select **SCADAPack 330**.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment dialog*.

The following table summarizes the Default Register Assignment for the SCADAPack 330.

Module	Address	Default Register Assignment
SCADAPack 330 I/O	fixed	
digital outputs		00001 to 00002
digital inputs		10001 to 10001
counter inputs		30001 to 30006

6.12.12 SCADAPack 334 Default Register Assignment

Description

A default Register Assignment is provided for the SCADAPack 330 controller. The table contains a convenient assignment for the SCADAPack 330 controller.

To enable the SCADAPack 330 default register assignment:

- Select **Type** from the **Controller** menu.
- Select **SCADAPack 334**.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment dialog*.

The following table summarizes the Default Register Assignment for the SCADAPack 330.

Module	Address	Default Register Assignment
SCADAPack 33x I/O digital outputs digital inputs counter inputs	fixed	00001 to 00002 10001 to 10001 30001 to 30006
SCADAPack 5607 I/O module digital outputs digital inputs analog inputs analog outputs	0 to 7	00003 to 00012 10002 to 10025 30007 to 30014 40001 to 40002

6.12.13 SCADAPack 350 Default Register Assignment

A default Register Assignment is provided for the SCADAPack 350 controller. The table contains a convenient assignment for the SCADAPack 350 controller.

To enable the SCADAPack 350 default register assignment:

- Select **Type** from the **Controller** menu.
- Select **SCADAPack 350**.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment dialog*.

The following table summarizes the Default Register Assignment for the SCADAPack 350.

Module	Address	Default Register Assignment
CNTR Controller counter inputs		30009 to 30014
SCADAPack 350 I/O	fixed	
digital outputs		00001 to 00012
digital inputs		10001 to 10016
analog inputs		30001 to 30008
analog outputs		40001 to 40002

6.12.14 SCADAPack 32 (5601 I/O Module) Default Register Assignment

A default Register Assignment is provided for the SCADAPack 32 controller. The table contains a convenient assignment for the SCADAPack 32 controller.

To enable the SCADAPack 32 default register assignment:

- Select **Type** from the **C**ontroller menu.
- Select **SCADAPack 32**.
- Select **Register Assignment** from the **C**ontroller menu.
- Select the **D**efault button in the *Register Assignment* dialog.
- Select 5601 I/O Module in the *Select Register Assignment Option* dialog.

The following table summarizes the Default Register Assignment for the SCADAPack 32.

See the

SCADAPack 5601 I/O Module section for information on the registers used.

Module	Address	Default Register Assignment
DIN Controller digital inputs		10017 to 10019
DIN Controller interrupt input		10020 to 10020
SCADAPack 5601 I/O module digital outputs digital inputs analog inputs	Fixed	00001 to 00012 10001 to 10016 30001 to 30008

6.12.15 SCADAPack 32 (5604 I/O Module) Default Register Assignment

A default Register Assignment is provided for the SCADAPack 32 controller. The table contains a convenient assignment for the SCADAPack 32 controller.

To enable the SCADAPack 32 default register assignment:

- Select **Type** from the **C**ontroller menu.
- Select **SCADAPack 32**.
- Select **Register Assignment** from the **C**ontroller menu.
- Select the **Default** button in the *Register Assignment* dialog.
- Select 5601 I/O Module in the *Select Register Assignment Option* dialog.

The following table summarizes the Default Register Assignment for the SCADAPack 32.

See the

SCADAPack 5604 I/O Module section for information on the registers used.

Module	Address	Default Register Assignment
DIN Controller digital inputs		10017 to 10019
DIN Controller interrupt input		10020 to 10020
SCADAPack LP I/O	fixed	
digital outputs		00001 to 00012
digital inputs		10001 to 10016
analog inputs		30001 to 30008
analog outputs		40001 to 40002

6.12.16 SCADAPack 32P Default Register Assignment

A default Register Assignment is provided for the SCADAPack 32P controller. The table contains a convenient assignment for the SCADAPack 32P controller.

To enable the SCADAPack 32P default register assignment:

- Select **Type** from the **Controller** menu.
- Select **SCADAPack 32P**.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment dialog*.

The following table summarizes the Default Register Assignment for the SCADAPack 32P.

Module	Address	Default Register Assignment
DIN Controller digital inputs		10001 to 10003
DIN Controller interrupt input		10004 to 10004

6.12.17 SCADAPack 100 Default Register Assignment

A default Register Assignment is provided for the SCADAPack 100 controller. The table contains a convenient assignment for the SCADAPack 100 controller.

To enable the SCADAPack 100 default register assignment:

- Select **Type** from the **Controller** menu.
- Select **SCADAPack 100**.
- Select **Register Assignment** from the **Controller** menu.
- Select the **Default** button in the *Register Assignment dialog*.

The following table summarizes the Default Register Assignment for the SCADAPack 100.

See the

SCADAPack 100 I/O section for information on the registers used.

Module	Address	Default Register Assignment
SCADAPack 100 I/O	fixed	
digital outputs		00001 to 00006
digital inputs		10001 to 10006
analog inputs and counters		30001 to 30008

7 TeleBUS Protocols Overview

The TeleBUS communication protocols provide a standard communication interface to SCADAPack and TeleSAFE controllers. The TeleBUS protocols are compatible with the widely used Modbus RTU and ASCII protocols. Additional TeleBUS commands provide remote programming and diagnostics capability.

The TeleBUS protocols operate on a wide variety of serial data links. These include RS-232 serial ports, RS-485 serial ports, radios, leased line modems, and dial up modems. The protocols are generally independent of the communication parameters of the link, with a few exceptions.

TeleBUS protocol commands may be directed to a specific device, identified by its station number, or broadcast to all devices. Using extended addressing up to 65534 devices may connect to one communication network.

The TeleBUS protocols provide full access to the I/O database in the controller. The I/O database contains user-assigned registers and general purpose registers. Assigned registers map directly to the I/O hardware or system parameter in the controller. General purpose registers can be used by ladder logic and C application programs to store processed information, and to receive information from a remote device.

Application programs can initiate communication with remote devices. A multiple port controller can be a data concentrator for remote devices, by polling remote devices on one port and responding as a slave on another port.

The protocol type, communication parameters and station address are configured separately for each serial port on a controller. One controller can appear as different stations on different communication networks. The port configuration can be set from an application program, from the TelePACE programming software, or from another Modbus compatible device.

7.1 Compatibility

There are two TeleBUS protocols. The TeleBUS RTU protocol is compatible with the Modbus RTU protocol. The TeleBUS ASCII protocol is compatible with the Modbus ASCII protocol.

Compatibility refers to communication only. The protocol defines communication aspects such as commands, syntax, message framing, error handling and addressing. The controllers do not mimic the internal functioning of any programmable controller. Device specific functions – those that relate to the hardware or programming of a specific programmable controller – are not implemented.

7.2 Serial Port Configuration

7.2.1 Communication Parameters

The TeleBUS protocols are, in general, independent of the serial communication parameters. The baud rate, word length and parity may be chosen to suit the host computer and the characteristics of the data link.

The port configuration can be set in four ways:

- using the TelePACE program;
- using the **set_port** function from a C application program;
- writing to the I/O database from a C or ladder logic application program; or
- writing to the I/O database remotely from a Modbus compatible device.

To configure a serial port through the I/O database, add the module, CNFG Serial port settings, to the Register Assignment Table.

7.2.1.1 RTU Protocol Parameters

The TeleBUS RTU protocol is an eight bit binary protocol. The table below shows possible and recommended communication parameters.

<i>Parameter</i>	<i>Possible Settings</i>	<i>Recommended Setting</i>
Baud Rate	see Baud Rate section below	see Baud Rate section below
Data Bits	8 data bits	8 data bits
Parity	None Even Odd	none
Stop bits	1 stop bit 2 stop bits ¹	1 stop bit
Flow control	Disabled	disabled
Duplex	see Duplex section below	see Duplex section below

¹ Not applicable to SCADAPack 330, SCADAPack 350 or SCADASense 4203.

7.2.1.2 ASCII Protocol Parameters

The TeleBUS ASCII protocol is an seven bit character based protocol. The table below shows possible and recommended communication parameters.

<i>Parameter</i>	<i>Possible Settings</i>	<i>Recommended Setting</i>
Baud Rate	see Baud Rate section below	see Baud Rate section below
Data Bits	7 data bits 8 data bits	7 data bits
Parity	None Even Odd	none
Stop Bits	1 stop bit 2 stop bits ¹	1 stop bit
Flow Control	Enabled Disabled	disabled
Duplex	see Duplex section below	see Duplex section below

¹ Not applicable to SCADAPack 33x, SCADAPack 350 or SCADASense 4203.

NOTE: Flow control should never be enabled with modems or in noisy environments. Noise can result in the accidental detection of an XOFF character, which shuts down communication. Flow control is not recommended for any environment, but can be used on high quality, full duplex, direct wiring where speeds greater than 4800 baud are required.

7.2.1.3 Baud Rate

The baud rate sets the communication speed. The type of serial data link used determines the possible settings. The table below shows the possible settings for SCADAPack and TeleSAFE controllers. Note that not all port types and baud rates are available on all controller ports.

<i>Port Type</i>	<i>Possible Settings</i>	<i>Recommended Setting</i>
RS-232 or RS-232 Dial-up modem	75 baud 110 baud 150 baud 300 baud 600 baud 1200 baud 2400 baud 4800 baud 9600 baud 19200 baud 38400 baud 57600 baud 115200 baud	Use the highest rate supported by all devices on the network.
RS-485	75 baud 110 baud 150 baud 300 baud 600 baud 1200 baud 2400 baud 4800 baud 9600 baud 19200 baud 38400 baud 57600 baud 115200 baud	Use the highest rate supported by all devices on the network.

7.2.1.4 Duplex

The TeleBUS protocols communicate in one direction at a time. However the type of serial data link used determines the duplex setting. The table below shows the possible settings for SCADAPack and TeleSAFE controllers. Note that not all port types are available on all controllers.

<i>Port Type</i>	<i>Possible Settings</i>	<i>Recommended Setting</i>
RS-232 or RS-232 Dial-up modem	half duplex full duplex	Use full duplex wherever possible. Use half duplex for most external modems.
RS-485	half duplex full duplex	Slave stations always use half duplex. Master stations can use full duplex only on 4 wire systems.

7.2.2 Protocol Parameters

The TeleBUS protocols operate independently on each serial port. Each port may set the protocol type, station number, protocol task priority and store-and-forward messaging options.

The port configuration can be set in four ways:

- using the TelePACE or ISaGRAF programs;
- using the **set_protocol** function from a C or C++ application program;
- writing to the I/O database from a C, C++, ISaGRAF or ladder logic application program;
- writing to the I/O database remotely from a Modbus compatible device.

To configure protocol settings through the I/O database, add the module, CNFG Protocol settings, to the Register Assignment for TelePACE applications or use the setprot function in ISaGRAF applications.

7.2.2.1 Protocol Type

The protocol type may be set to emulate the Modbus ASCII and Modbus RTU protocols, or it may be disabled. When the protocol is disabled, the port functions as a normal serial port.

7.2.2.2 Station Number

The TeleBUS protocol allows up to 254 devices on a network using standard addressing and up to 65534 devices using extended addressing. Station numbers identify each device. A device responds to commands addressed to it, or to commands broadcast to all stations.

The station number is in the range 1 to 254 for standard addressing and 1 to 65534 for extended addressing. Address 0 indicates a command broadcast to all stations, and cannot be used as a station number. Each serial port may have a unique station number.

7.2.2.3 Task Priority

A task is responsible for monitoring each serial port for messages. The real time operating system (RTOS) schedules the tasks with the application program tasks according to the task priority. The priority can be changed only with the **set_protocol** function from an application program.

The default task priority is 3. Changing the priority is not recommended.

7.2.2.4 Store and Forward Messaging

Store and forward messaging re-transmits messages received by a controller. Messages may be re-transmitted on any serial port, with or without station address translation. A user-defined translation table determines actions performed for each message. The **Store and Forward Messaging** section below describes this feature in detail.

Store and forward messaging may be enabled or disabled on each port. It is disabled by default.

7.3 I/O Database

The TeleBUS protocols read and write information from the I/O database. The I/O database contains user-assigned registers and general purpose registers.

User-assigned registers map directly to the I/O hardware or system parameter in the controller. Assigned registers are initialized to the default hardware state or system parameter when the controller is reset. Assigned output registers do not maintain their values during power failures. However, output registers do retain their values during application program loading.

General purpose registers are used by ladder logic and C application programs to store processed information, and to receive information from remote devices. General purpose registers retain their values during power failures and application program loading. The values change only when written by an application program or a communication protocol.

The I/O database is divided into four sections.

- Coil registers are single bits which the protocols can read and write. Coil registers are located in the digital output section of the I/O database. The number of registers depends on the controller. Coil registers are numbered from 1 to the maximum for the controller.
- Status registers are single bits which the protocol can read. Status registers are located in the digital input section of the I/O database. The number of registers depends on the controller. Status registers are numbered from 10001 to the maximum for the controller.
- Input registers are 16 bit registers which the protocol can read. Input registers are located in the analog input section of the I/O database. The number of registers depends on the controller. Input registers are numbered from 30001 to the maximum for the controller.

- Holding registers are 16 bit registers that the protocol can read and write. Holding registers are located in the analog output section of the I/O database. The number of registers depends on the controller. Holding registers are numbered from 40001 to the maximum for the controller.

7.3.1 Accessing the I/O Database

TelePACE ladder logic programs access the I/O database through function blocks. All function blocks can access the I/O database. Refer to the ***TelePACE Ladder Logic Reference and User Manual*** for details.

ISaGRAF applications access the I/O database through dictionary variables with assigned network addresses or using Permanent Non-Volatile Modbus registers. See the ***ISaGRAF User and Reference Manual*** for details.

C language programs access the I/O database with two functions. The **dbase** function reads a value from the I/O database. The **setdbase** function writes a value to the I/O database. Refer to the ***TelePACE C Tools Reference and User Manual*** for full details on these functions.

7.3.1.1 Coil and Status Registers

Coil and status registers contain one bit of information, that is, whether a signal is off or on.

Writing any non-zero value to the register turns the bit on. Writing zero to the register turns the bit off. If the register is assigned to an I/O module, the bit status is written to the module output hardware or parameter.

Reading a coil or status register returns –1 if the bit is on, or 0 if the bit is off. The stored value is returned from general purpose registers. The I/O module point status is returned from assigned registers.

7.3.1.2 Input and Holding Registers

Input and holding registers contain 16 bit values.

Writing any value to a general purpose register stores the value in the register. Writing a value to an assigned register, writes the value to the assigned I/O module.

Reading a general purpose register returns the value stored in the register. Reading an assigned register returns the value read from the I/O module.

7.3.1.3 Exception Status

The exception status is a single byte containing controller specific status information. It is returned in response to the Read Exception Status function (see the ***Slave Mode*** section).

A C language application program can define the status information. The **modbusExceptionStatus** function sets the status information. Ladder logic programs cannot set this information.

7.3.1.4 Slave ID

The slave ID is a variable length message containing controller specific information. It is returned in response to the Report Slave ID function (see the **Slave Mode** section).

A C language application program can define the information and the length of the message. The **modbusSlaveID** function sets the information. Ladder logic programs cannot set this information.

7.4 Extended Station Addressing

The TeleBUS RTU and ASCII protocols support two type of Modbus station addressing. Standard Modbus addressing allows a maximum of 255 stations and is compatible with standard Modbus devices.

Extended Modbus addressing allows a maximum of 65534 stations. Extended Modbus addressing is fully compatible with standard Modbus addressing for addresses between 0 and 254.

7.4.1 Theory of Operation

The address field of a Modbus message is a single byte. Address 0 is a broadcast address; messages sent to this address are sent to all stations. Addresses 1 to 255 are station addresses. Figure 1 shows the format of a standard Modbus message.

Field	Address	Function	...
Size	1	1	N

Figure 118: Standard Modbus Message

The address field extension adds a two-byte extended address field to the message. Figure 2 shows the format of an extended address Modbus message.

Field	Address = 255	Extended Address (high)	Extended Address (low)	Function	...
Size	1	1	1	1	n

Figure 119: Extended Address Modbus Message

Messages for addresses 0 to 254 use the standard format message. The station address is stored in the address byte.

Messages for stations 255 to 65534 use the extended address format message. The address byte is set to 255. This indicates the extended address format is used. The actual address is stored in the two extended address bytes.

Station address 65535 is reserved and cannot be used as a station number. This station address is used in store-and-forward tables to indicate a disabled station.

Slave, master and store-and-forward stations treat the addresses in the same manner. The application program controls the use of the extended addressing format. It may enable or disable the extended addressing.

7.5 Slave Mode

The TeleBUS protocols operate in slave and master modes simultaneously. In slave mode the controller responds to commands sent by another device. Commands may be sent to a specific device or broadcast to all devices.

The TeleBUS protocols emulate the Modbus protocol functions required for communication with a host device. These functions are described below. It also implements functions for programming and remote diagnostics. These functions are not required for host communication, so are not described here.

A technical specification for the TeleBUS protocol is available from Control Microsystems. It describes all the functions in detail. In most cases knowledge of the actual commands is not required.

7.5.1 Broadcast Messages

A broadcast message is sent to all devices on a network. Each device executes the command. No device responds to a broadcast command. The device sending the command must query each device to determine if the command was received and processed. Broadcast messages are supported for some function codes that write information.

A broadcast message is sent to station number 0.

7.5.2 Function Codes

The table summarizes the implemented function codes. The maximum number of registers that can be read or written with one message is shown in the maximum column.

<i>Function</i>	<i>Name</i>	<i>Description</i>	<i>Maximum</i>
01	Read Coil Status	Reads digital output registers.	2000
02	Read Input Status	Reads digital input registers.	2000
03	Read Holding Register	Reads analog output registers.	125
04	Read Input Register	Reads analog input registers.	125
05	Force Single Coil	Writes digital output register.	1
06	Preset Single Register	Writes analog output registers.	1
07	Read Exception Status	Reads special information.	N/A
15	Force Multiple Coils	Writes digital output registers.	880
16	Preset Multiple Registers	Writes analog output registers.	60
17	Report Slave ID	Reads controller type information	N/A

Functions 5, 6, 15, and 16 support broadcast messages. The functions are described in detail below.

7.5.2.1 Read Coil Status

The Read Coil Status function reads data from the digital output section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is one bit.

7.5.2.2 Read Input Status

The Read Input Status function reads data from the digital input section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is one bit.

7.5.2.3 Read Holding Register

The Read Holding Register function reads data from the analog output section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

7.5.2.4 Read Input Register

The Read Input Register function reads data from the analog input section of the I/O database. Any number of registers may be read up to the maximum number. The read may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

7.5.2.5 Force Single Coil

The Force Single Coil function writes one bit into the digital output section of the I/O database. The write may specify any valid register.

7.5.2.6 Preset Single Register

The Preset Single Register function writes one 16 bit value into the analog output section of the I/O database. The write may specify any valid register.

7.5.2.7 Read Exception Status

The Read Exception Status function reads a single byte containing controller specific status information. The information is defined by the application program. This function is included for compatibility with devices expecting to communicate with a Modicon PLC.

7.5.2.8 Force Multiple Coils

The Force Multiple Coils function writes single bit values into the digital output section of the I/O database. Any number of registers may be written up to the maximum number. The write may start at any address, provided the entire block is within the valid register range. Each register is 1 bit.

7.5.2.9 Preset Multiple Registers

The Preset Multiple Register function writes 16 bit values into the analog output section of the I/O database. Any number of registers may be written up to the maximum number. The write may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

7.5.2.10 Report Slave ID

The Report Slave ID function reads a variable length message containing controller specific information. The information and the length of the message is defined by the application program. This function is included for compatibility with devices expecting to communicate with a Modicon PLC.

7.6 Modbus Master Mode

The TeleBUS protocol may act as a communication master on any serial port. In master mode, the controller sends commands to other devices on the network. Simultaneous master messages may be active on all ports.

The protocol cannot support master mode and store-and-forward mode simultaneously on a serial port. Enabling store and forward messaging disables processing of responses to master mode commands. Master mode may be used on one port and store-and-forward mode on another port.

7.6.1 Modbus Function Codes

The table shows the implemented function codes. The maximum number of registers that can be read or written with one message is shown in the maximum column. The slave device may support fewer registers than shown; consult the manual for the device for details.

<i>Function</i>	<i>Name</i>	<i>Description</i>	<i>Maximum</i>
01	Read Coil Status	Reads digital output registers.	2000
02	Read Input Status	Reads digital input registers.	2000
03	Read Holding Register	Reads analog output registers.	125
04	Read Input Register	Reads analog input registers.	125
05	Force Single Coil	Writes digital output register.	1
06	Preset Single Register	Writes analog output registers.	1
15	Force Multiple Coils	Writes digital output registers.	880
16	Preset Multiple Registers	Writes analog output registers.	60

7.6.1.1 Read Coil Status

The Read Coil Status function reads data from coil registers in the remote device. Data can be written into the digital input or the digital output sections of the I/O database.

Any number of registers may be read up to the maximum number supported by the slave device or the maximum number above, whichever is less. The read may start at any address, provided the entire block is within the valid register range. Each register is one bit.

7.6.1.2 Read Input Status

The Read Input Status function reads data from input registers in the remote device. Data can be written into the digital input or the digital output sections of the I/O database.

Any number of registers may be read up to the maximum number supported by the slave device or the maximum number above, whichever is less. The read may start at any address, provided the entire block is within the valid register range. Each register is one bit.

7.6.1.3 Read Holding Register

The Read Holding Register function reads data from holding registers in the remote device. Data can be written into the analog input or the analog output sections of the I/O database.

Any number of registers may be read up to the maximum number supported by the slave device or the maximum number above, whichever is less. The read may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

7.6.1.4 Read Input Register

The Read Input Register function reads data from input registers in the remote device. Data can be written into the analog input or the analog output sections of the I/O database.

Any number of registers may be read up to the maximum number supported by the slave device or the maximum number above, whichever is less. The read may start at any address, provided the entire block is within the valid register range. Each register is 16 bits.

7.6.1.5 Force Single Coil

The Force Single Coil function writes one bit into a coil register in the remote device. The data may come from the digital input or digital output sections of the I/O database.

The write may specify any valid coil register in the remote device.

7.6.1.6 Preset Single Register

The Preset Single Register function writes one 16 bit value into a holding register in the remote device. The data may come from the analog input or output sections of the I/O database.

The write may specify any valid holding register in the remote device.

7.6.1.7 Force Multiple Coils

The Force Multiple Coils function writes single bit values coil registers in the remote device. The data may come from the digital input or digital output sections of the I/O database.

Any number of registers may be written up to the maximum number supported by the slave device or the maximum number above, which ever is less. The write may start at any address, provided the entire block is within the valid register range of the remote device. Each register is 1 bit.

7.6.1.8 Preset Multiple Registers

The Preset Multiple Register function writes 16 bit values into holding registers of the remote device. The data may come from the analog input or output sections of the I/O database.

Any number of registers may be written up to the maximum number supported by the slave device or the maximum number above, which ever is less. The write may start at any address, provided the entire block is within the valid register range of the remote device. Each register is 16 bits.

7.6.2 Enron Modbus Master Mode

The Enron Modbus protocol is based on the Modbus ASCII and RTU protocols. Message framing is identical to the Modbus protocols. However, there are many differences in message formatting and register numbering, at both the logical and protocol levels.

The document *Specifications and Requirements for an Electronic Flow Measurement Remote Terminal Unit* describes the Enron Modbus protocol.

7.6.2.1 Variable Types

There are ranges of Enron registers to hold short integers, long integers and single precision floats. The ranges are as follows.

Range	Data Type
1001 - 1999	Boolean
3001 - 3999	Short integer
5001 - 5999	Long integer
7000 - 9999	Float

In general, both Numeric and Boolean function codes can be used to read and write all types of registers. Consult the Enron Modbus specification for details.

7.6.2.1.1 Boolean Registers

Enron Modbus Boolean registers are usually numbered 1001 to 1999.

Boolean registers are read using Modbus command 1. Boolean registers are written using Modbus command 5 for single registers and 15 for multiple registers.

The address offset in the message is equal to the register number.

The number of Modbus registers is equal to the number of Enron registers.

The response format is identical to the Modbus response format.

7.6.2.1.2 Short Integer Registers

Enron Modbus Short Integer registers are usually numbered 3001 to 3999.

Short Integer registers are read using Modbus command 3. Short Integer registers are written using Modbus command 6 for single registers and 16 for multiple registers.

The address offset in the message is equal to the register number.

The number of Modbus registers is equal to the number of Enron registers.

The response format is identical to the Modbus response format.

7.6.2.1.3 Long Integer Registers

Enron Modbus Long Integer registers are usually numbered 5001 to 5999.

Long Integer registers are read using Modbus command 3. Long Integer registers are written using Modbus command 6 for single registers and 16 for multiple registers.

The address offset in the message is equal to the register number.

The number of Modbus registers requested is equal to the number of Enron registers.

The number of Modbus registers expected in the response is equal to two times the number of Enron registers.

7.6.2.1.4 Floating Point Registers

Enron Modbus Floating-point registers are usually numbered 7001 to 7999.

Floating-point registers are read using Modbus command 3. Floating-point registers are written using Modbus command 6 for single registers and 16 for multiple registers.

The address offset in the message is equal to the register number.

The number of Modbus registers requested is equal to the number of Enron registers.

The number of Modbus registers expected in the response is equal to two times the number of Enron registers.

7.6.2.2 Enron Modbus Function Codes

The following table shows the implemented function codes for Enron Modbus. The maximum number of registers that can be read or written with one message is shown in the maximum column. The slave device may support fewer registers than shown; consult the manual for the device for details.

Functions 129, 130, 132, 133, 135, 136, 138, and 139 may be broadcast, but some Enron Modbus slave devices may not support broadcast messages. Consult the manual for the device for details.

Function	Name	Description	Maximum
128	Read Enron Boolean	Read Enron Boolean registers	2000
129	Write Enron Boolean	Write Enron Boolean register	1
130	Write Enron Multiple Boolean	Write Enron Boolean registers	880
131	Read Enron Short Integer	Read Enron short integer register	125
132	Write Enron Short Integer	Write Enron short integer register	1
133	Write Enron Multiple Short Integer	Write Enron short integer registers	60
134	Read Enron Long Integer	Read Enron long integer register	62
135	Write Enron Long Integer	Write Enron long integer register	1
136	Write Enron Multiple Long Integer	Write Enron long integer registers	30
137	Read Enron Floating Point	Read Enron floating-point register	62
138	Write Enron Floating Point	Write Enron floating-point register	1
139	Write Enron Multiple Floating Point	Write Enron floating-point registers	30

7.6.3 Sending Messages

A master message is initiated in one of five ways:

- using the **master_message** function from a C or C++ application program; or
- using the **MSTR** function block from a TelePACE ladder logic program; or
- using the **MSIP** function block from a TelePACE ladder logic program; or
- using the **master** function in an ISaGRAF program; or
- using the **masterip** function in an ISaGRAF program.

These functions specify the port on which to issue the command, the function code, the type of station addressing, the slave station number, and the location and size of the data in the slave and master devices. The protocol driver, independent of the application program receives the response to the command.

The application program detects the completion of the transaction by:

- calling the **get_protocol_status** function in a C application program; or
- using the output of the **MSTR** function block in a TelePACE ladder logic program; or
- using the output of the **master** function in an ISaGRAF program.

A communication error has occurred if the slave does not respond within the expected maximum time for the complete command and response. The application program is responsible for detecting this condition. When errors occur, it is recommended that the application program retry several times before indicating a communication failure.

The completion time depends on the length of the message, the length of the response, the number of transmitted bits per character, the transmission baud rate, and the maximum message turn-around time. One to three seconds is usually sufficient. Radio systems may require longer delays.

7.7 Store and Forward Messaging

Store and forward messaging is required on systems where there is no direct link between a host computer and all the remote sites. This occurs on radio systems where the host computer transmission cannot be heard by all remote sites. It occurs on systems where one controller is used as a data concentrator for several remote units. With store and forward messaging, a request to a controller that cannot be directly accessed by a host is routed through an intermediate controller, which can communicate with both the host and the remote controller.

The TeleBUS protocol provides store and forward messaging through address translation. A controller configured for store and forward operation receives messages destined for a remote station, re-addresses them according to translation table, and forwards the message to the remote station. Responses from the remote station are processed in the same manner.

The TeleBUS protocol allows messages to be re-transmitted on the same port with address translation. This is used with radio systems. The radio at the intermediate site is used as a type of repeater. The protocol allows messages to be re-transmitted on a different port, with or without address translation. This is used where the intermediate controller is a bridge between two networks.

The TeleBUS protocol driver maintains diagnostics counters at the store and forward site on the number of messages received and transmitted to aid in the diagnosing of communication problems.

The protocol cannot support master mode and store-and-forward mode simultaneously on a serial port. Enabling store and forward messaging disables processing of responses to master mode commands. Master mode may be used on one port and store-and-forward mode on another port. Applications requiring both modes on a single port must switch the modes under control of the application program.

7.7.1 Translation Table

The translation table specifies address and communication port translation. The translation table differs for SCADAPack and SCADAPack 32 controllers. Each entry in the translation table for SCADAPack controllers has four components, as shown in the table entry below.

Port	Station Address	Port	Station Address B

A	A	B	

The entry defines a bi-directional transfer. A message (poll or reply) received for station A on port A is re-transmitted to station B on port B. A message received for station B on port B is re-transmitted to station A on port A.

Each entry in the translation table for SCADAPack 32 controllers has five components, as shown in the table entry below.

Slave Interface	Slave Station	Forward Interface	Forward Station	Forward IP Address

The Slave Interface entry contains the receiving slave interface the message is received from for each translation.

The Slave Station entry contains the Modbus station address of the slave message.

The Forward Interface entry contains the interface the message is forwarded from. When forwarding to a TCP or UDP network, the protocol type is selected for the Forward Interface. The IP Stack automatically determines the exact interface (e.g. Ethernet1) to use when it searches the network for the Forward IP Address.

The Forward Station entry contains the Modbus station address of the forwarded message.

The Forward IP Address entry contains the IP address of the Forward Station. This field is blank unless a TCP or UDP network is selected for Forward Interface.

7.7.1.1 Table Size

The translation table holds 128 translation entries. This is sufficient to re-transmit one-half of 256 possible addresses. On a single port controller only 128 translations are required since each address must translate to a different address for re-transmission on the same port see Invalid Translations.

7.7.2 Invalid Translations

The following translations are not valid. The described action is taken when these translations are encountered.

- Re-transmission on the same port with the same address is not valid, except for broadcast messages. This restriction is required because many message responses are identical to the command. It is impossible for the master station to distinguish between the re-transmitted message and the response from the slave. The re-transmitted message would appear to be the response.
- The protocol re-transmits broadcast messages on the same port. Some stations will receive the broadcast message twice. The master station will also receive the message and may execute it if it is able to operate as a slave. The user must bear these consequences in mind when forwarding broadcast messages.
- The store and forward controller also processes broadcast messages.
- Translations where either of the station addresses are the same as the controller station address for the port, are not valid. The protocol processes these messages

as if they were directed to the controller. It does not look up the address in the translation table.

- Translations with non-existent port numbers or invalid addresses are not valid.
- Multiple translations for a port and station address combination are not valid.
- Translations where one station is DISABLED and the other station is not, are not valid. A DISABLED translation is a valid translation.

7.7.3 Store and Forward Configuration

The Store and Forward configuration varies depending on the controller you are configuring. The configuration for each type of controller is described in the following sections.

7.7.3.1 SCADAPack Controller

An application program, written in TelePACE Ladder Logic or TelePACE C Tools and ISaGRAF IEC61131 or ISaGRAF IEC61131 C Tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

7.7.3.1.1 TelePACE Ladder Logic

1. To enable the use of store and forward messaging on one or more serial ports the Configuration I/O Module **CNFG Protocol Settings Method 1, 2 or 3** must be added to the register assignment. The store and forward enable register must be set to enable.
2. Add the Configuration I/O Module **CNFG Store and Forward** to the register assignment to configure the translation table.
3. Configure the translation table by writing the necessary translation table entries to the registers defined in the CNFG Store and Forward I/O module.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when TelePACE programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.1.2 TelePACE C Tools

The TelePACE C language application program interface provides the following functions. Refer to the **TelePACE C Tools Reference and User Manual** for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED (station 256).

- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.1.3 ISaGRAF IEC61131

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The SandFEnabled input must be set to TRUE.
2. Configure the translation table by using the **setsf** function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when ISaGRAF IEC61131 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.1.4 ISaGRAF IEC61131 C Tools

The ISaGRAF C language application program interface provides the following functions. Refer to the *ISaGRAF C Tools Reference and User Manual* for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.2 SCADAPack Light Controller

An application program, written in TelePACE Ladder Logic or TelePACE C Tools and ISaGRAF IEC61131 or ISaGRAF IEC61131 C Tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

7.7.3.2.1 *TelePACE Ladder Logic*

1. To enable the use of store and forward messaging on one or more serial ports the Configuration I/O Module **CNFG Protocol Settings Method 1, 2 or 3** must be added to the register assignment. The store and forward enable register must be set to enable.
2. Add the Configuration I/O Module **CNFG Store and Forward** to the register assignment to configure the translation table.
3. Configure the translation table by writing the necessary translation table entries to the registers defined in the CNFG Store and Forward I/O module.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when TelePACE programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.2.2 *TelePACE C Tools*

The TelePACE C language application program interface provides the following functions. Refer to the **TelePACE C Tools Reference and User Manual** for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.2.3 *ISaGRAF IEC61131*

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The SandFEnabled input must be set to TRUE.
2. Configure the translation table by using the **setsf** function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when ISaGRAF IEC61131 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.2.4 ISaGRAF IEC61131 C Tools

The ISaGRAF C language application program interface provides the following functions. Refer to the *ISaGRAF C Tools Reference and User Manual* for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.3 SCADAPack Plus Controller

An application program, written in TelePACE Ladder Logic or TelePACE C Tools and ISaGRAF IEC61131 or ISaGRAF IEC61131 C Tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

7.7.3.3.1 TelePACE Ladder Logic

1. To enable the use of store and forward messaging on one or more serial ports the Configuration I/O Module **CNFG Protocol Settings Method 1, 2 or 3** must be added to the register assignment. The store and forward enable register must be set to enable.
2. Add the Configuration I/O Module **CNFG Store and Forward** to the register assignment to configure the translation table.
3. Configure the translation table by writing the necessary translation table entries to the registers defined in the CNFG Store and Forward I/O module.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when TelePACE programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.3.2 TelePACE C Tools

The TelePACE C language application program interface provides the following functions. Refer to the *TelePACE C Tools Reference and User Manual* for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.

- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.3.3 **ISaGRAF IEC61131**

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The SandFEnabled input must be set to TRUE.
2. Configure the translation table by using the **setsf** function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when ISaGRAF IEC61131 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.3.4 **ISaGRAF IEC61131 C Tools**

The ISaGRAF C language application program interface provides the following functions. Refer to the ***ISaGRAF C Tools Reference and User Manual*** for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.4 SCADAPack LP Controller

An application program, written in TelePACE Ladder Logic or TelePACE C Tools and ISaGRAF IEC61131 or ISaGRAF IEC61131 C Tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

7.7.3.4.1 TelePACE Ladder Logic

1. To enable the use of store and forward messaging on one or more serial ports the Configuration I/O Module **CNFG Protocol Settings Method 1, 2 or 3** must be added to the register assignment. The store and forward enable register must be set to enable.
2. Add the Configuration I/O Module **CNFG Store and Forward** to the register assignment to configure the translation table.
3. Configure the translation table by writing the necessary translation table entries to the registers defined in the CNFG Store and Forward I/O module.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when TelePACE programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.4.2 TelePACE C Tools

The TelePACE C language application program interface provides the following functions. Refer to the **TelePACE C Tools Reference and User Manual** for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.4.3 ISaGRAF IEC61131

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The SandFEnabled input must be set to TRUE.

2. Configure the translation table by using the **setsf** function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when ISaGRAF IEC61131 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.4.4 ISaGRAF IEC61131 C Tools

The ISaGRAF C language application program interface provides the following functions. Refer to the ***ISaGRAF C Tools Reference and User Manual*** for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.5 SCADAPack 100 Controller

An application program, written in TelePACE Ladder Logic or TelePACE C Tools and ISaGRAF IEC61131 or ISaGRAF IEC61131 C Tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

7.7.3.5.1 TelePACE Ladder Logic

1. To enable the use of store and forward messaging on one or more serial ports the Configuration I/O Module ***CNFG Protocol Settings Method 1, 2 or 3*** must be added to the register assignment. The store and forward enable register must be set to enable.
2. Add the Configuration I/O Module ***CNFG Store and Forward*** to the register assignment to configure the translation table.
3. Configure the translation table by writing the necessary translation table entries to the registers defined in the CNFG Store and Forward I/O module.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when TelePACE programming software or a SERVICE

boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.5.2 TelePACE C Tools

The TelePACE C language application program interface provides the following functions. Refer to the ***TelePACE C Tools Reference and User Manual*** for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.5.3 ISaGRAF IEC61131

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The SandFEnabled input must be set to TRUE.
2. Configure the translation table by using the **setsf** function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when ISaGRAF IEC61131 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.5.4 ISaGRAF IEC61131 C Tools

The ISaGRAF C language application program interface provides the following functions. Refer to the ***ISaGRAF C Tools Reference and User Manual*** for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).

- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.6 SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 and 32P Controller

An application program, written in TelePACE Ladder Logic or TelePACE C++ Tools and ISaGRAF IEC61131 or ISaGRAF IEC61131 C++ Tools programming, enables and configures store and forward messaging. A HMI host may enable and configure store and forward messaging through the controller I/O database.

7.7.3.6.1 TelePACE Ladder Logic

When a SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P controllers are used the store and forward translation table is configured using an **Element Configuration** dialog. From the **Controller** menu select the **Store and Forward** command to access the element configuration. Refer to the TelePACE Ladder Logic Program Reference Manual for complete information on using the Store and Forward element configuration.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when TelePACE programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.6.2 TelePACE C++ Tools

The SCADAPack 32 C++ language application program interface provides the following functions. Refer to the **SCADAPack 32 C++ Tools Reference and User Manual** for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.3.6.3 ISaGRAF IEC61131

1. To enable the use of store and forward messaging on one or more serial ports the Custom Function **setprot** or **setprot2** must be added to the project. The SandFEnabled input must be set to TRUE.
2. Configure the translation table by using the **setsfip2** function to write the necessary translation table entries.

The translation table must be initialized before store and forward messaging is enabled. Forwarding of messages is disabled when ISaGRAF IEC61131 programming software or a SERVICE boot initializes the controller. This prevents inadvertent forwarding of messages when new controllers are installed on networks.

7.7.3.6.4 ISaGRAF IEC61131 C++ Tools

The SCADAPack 32 C++ language application program interface provides the following functions. Refer to the *SCADAPack 32 C++ Tools Reference and User Manual* for details.

- The **getSFTranslation** function returns an entry from the store and forward translation table. The entry consists of two port and station address pairs.
- The **setSFTranslation** function writes an entry into the store and forward translation table. The entry consists of two port and station address pairs. The function checks for invalid translations; if the translation is not valid it is not stored. The function returns a status code indicating success or an error if the translation is not valid. A translation is cleared from the table by writing a translation with both stations set to DISABLED_STATION (65535).
- The **clearSFTranslationTable** function clears all entries in the translation table. A cleared entry has the port set to 0 (com1) and the station set to DISABLED_STATION (65535).
- The **checkSFTranslationTable** function checks the translation table for invalid entries. It returns a status structure indicating if the table is valid and the location and type of the first error if it is not valid.

7.7.4 Diagnostics Counters

The TeleBUS protocol provides diagnostics counters for each serial port. The counters aid in determining the source of communication errors. Store and forward messaging provides the following counters for each communication port. All counters have a maximum count of 65535. Counters roll back to zero on the next event.

- **Stored Message Counter:** the number of messages received, which qualified for forwarding. A message qualifies for forwarding if a valid translation is found for the port and station in the translation table.
- **Forwarded Message Counter:** the number of messages forwarded (transmitted) on this port.

Refer to the user manual for the controller and programming environment you are using for information on the diagnostics counters.

7.8 Point-To-Point Protocol (PPP)

SCADAPack 32 and SCADAPack 32P controllers support Point-to-Point Protocol (PPP) on the serial ports. Any serial port may be configured for the PPP protocol. Once a PPP connection is established the serial port has access to all IP protocol servers enabled on the controller.

A serial port configured for PPP supports an auto answer mode when dialed up through a modem. After answering the modem the serial port performs the login steps according to the authentication option selected for the port.

PPP provides two authentication protocols, which automates logins - PAP (Password Authentication Protocol) and CHAP (Challenge-Handshake Authentication Protocol).

PPP settings are configurable for each serial port on the SCADAPack 32 or SCADAPack 32P controller.

An inactivity timeout closes the PPP connection and hangs up the modem when the connection becomes idle. The timeout may also be disabled. Timeout range is 1 to 65535 minutes (~1092 hours maximum).

When the PPP protocol is selected for a serial port, the serial port must be assigned a unique IP address, different from the IP address assigned to Ethernet or any other active PPP connection.

The remote end of a PPP connection may request an IP address from the controller PPP Server. The PPP Server will provide this IP address if requested.

Only one default gateway may be assigned to the controller. A PPP connection may be configured as the gateway.

7.8.1 PPP Client Setup in Windows 2000

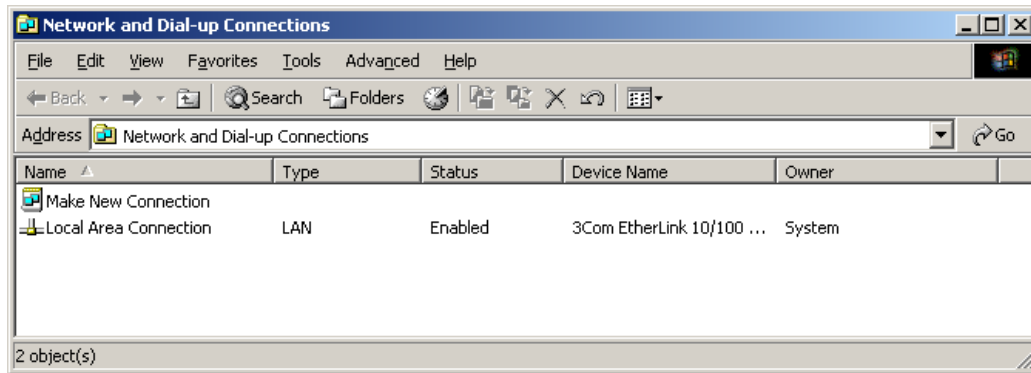
This section describes the procedure for setting up a PPP client from a Windows 2000 PC. Client setup for a dialup PPP connection and a direct serial PPP connection are presented.

7.8.1.1 Direct Serial PPP Connection using Windows 2000

7.8.1.1.1 Connection Setup

Use this connection when an only serial cable is used to establish a PPP connection between a Windows 2000 PC and a SCADAPack 32, without a dialup modem.

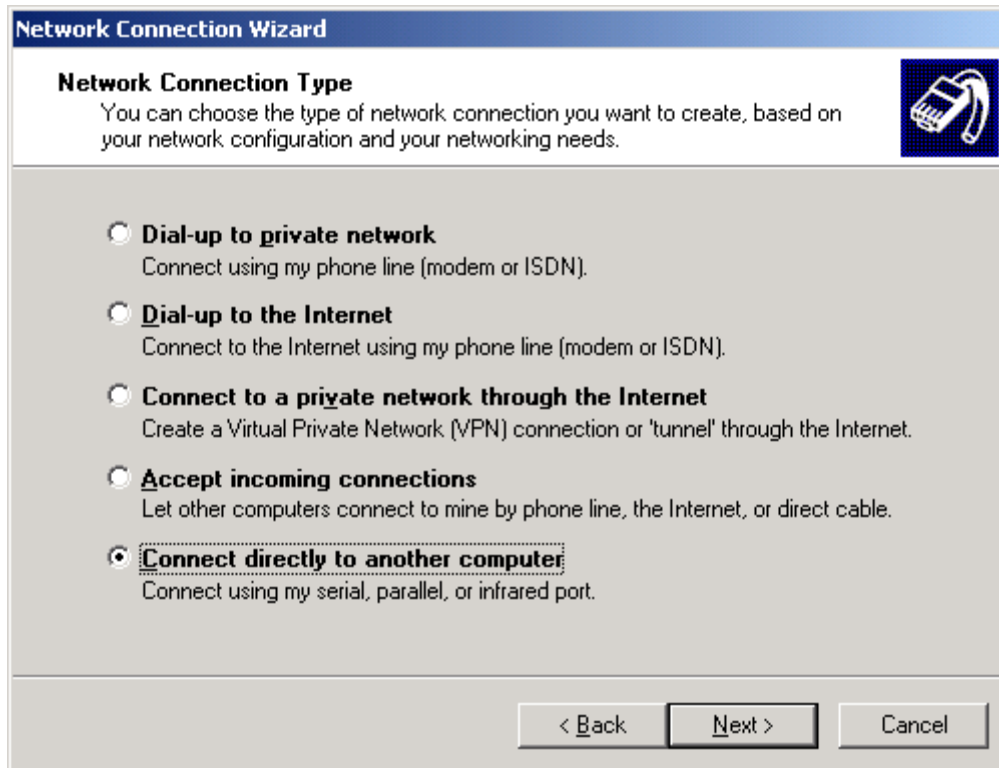
1. From the **Start** menu, right click **Network and Dial-up Connections** from the **Settings** group, and select **Open**. The *Network and Dial-up Connections* dialog is displayed.



2. Double click the item **Make New Connection** from the *Network and Dial-up Connections* dialog. The connection wizard dialog is displayed.



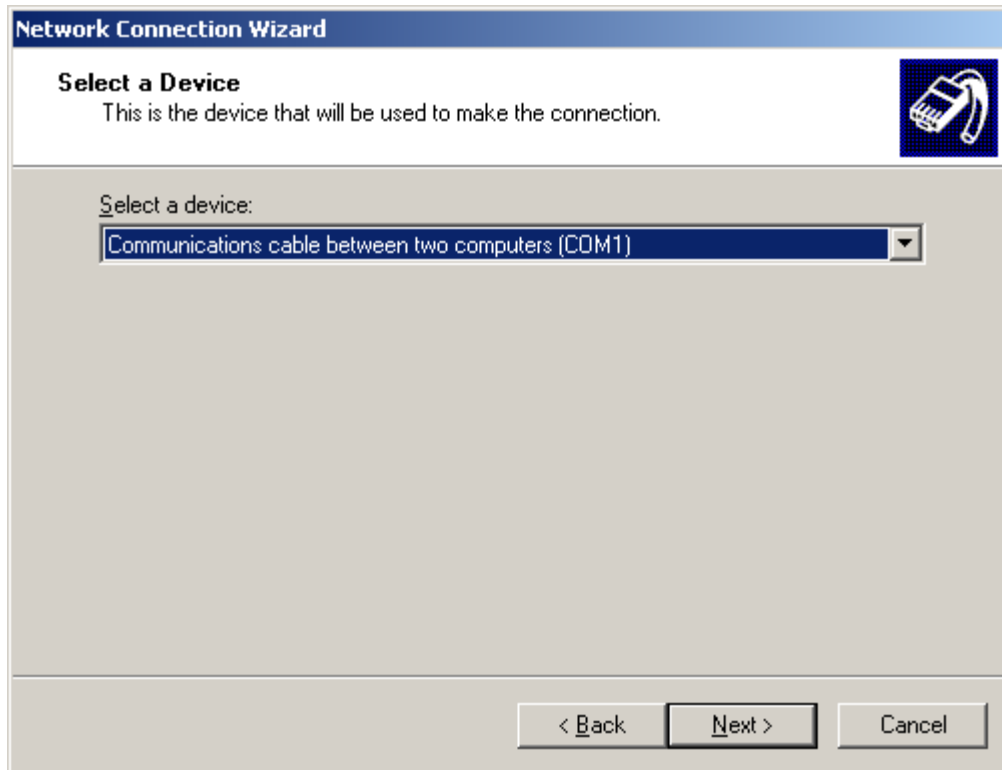
3. Select the **Next** button to display the connection type options dialog.



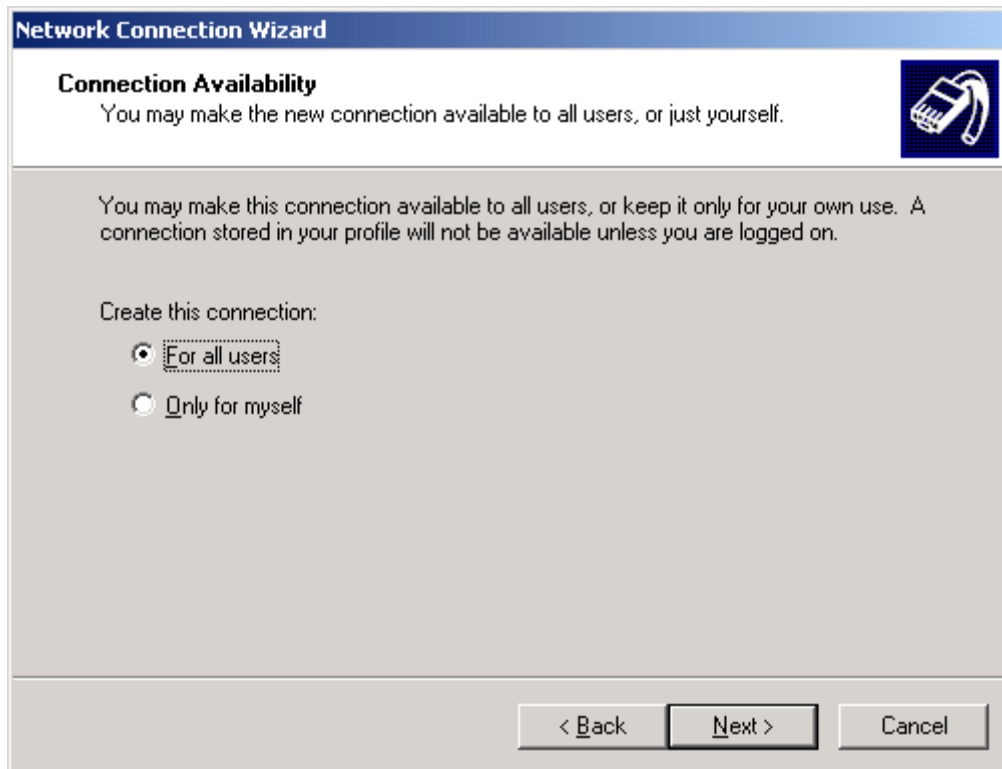
4. For Network Connection Type select the type **Connect directly to another computer** and select the **Next** button. The Host or Guest options dialog is displayed.



5. Select the **Guest** option and the **Next** button. The *Select a Device* dialog is displayed.



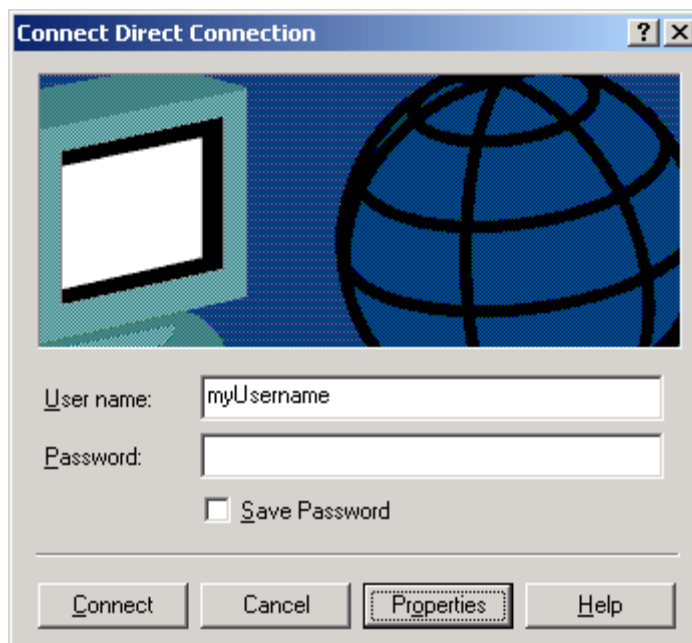
6. From the menu select the serial port on your PC that will be used to connect to the SCADAPack 32. Select the **Next** button. The *Connection Availability* dialog is displayed.



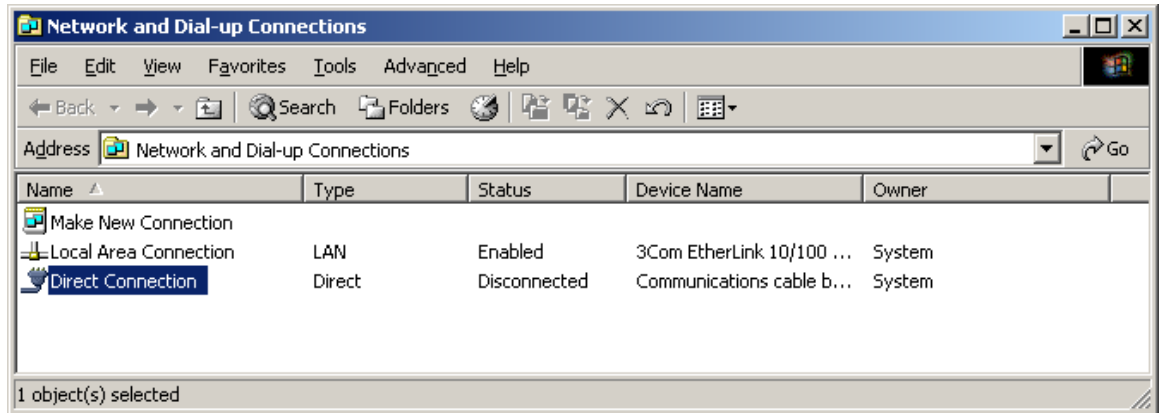
7. Select either option and then select the **Next** button. The *Connection Name* dialog is displayed.



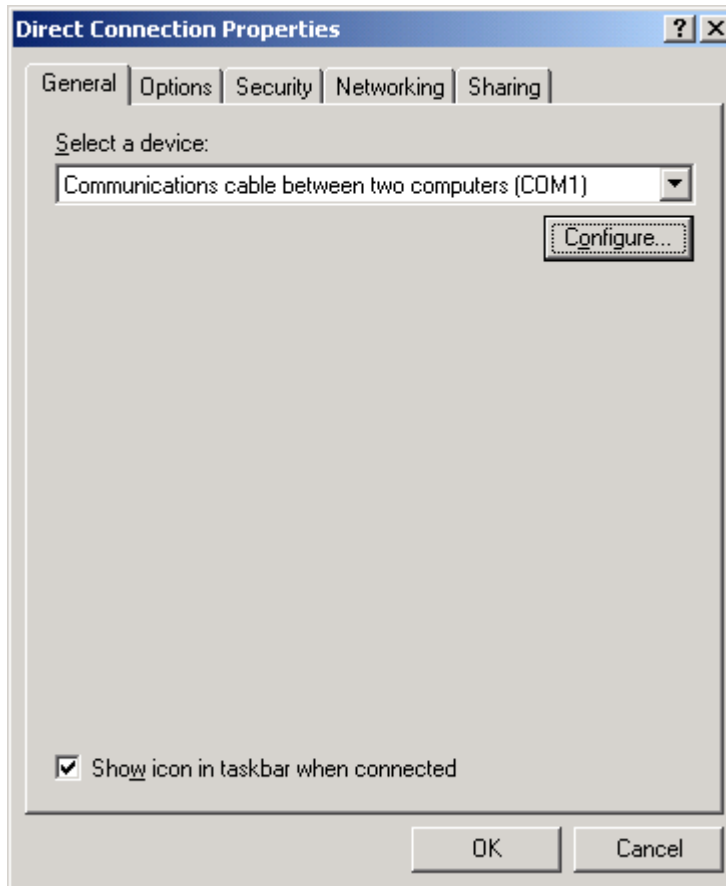
8. Enter a name for the connection and select the **Finish** button. The username and password prompt is displayed.



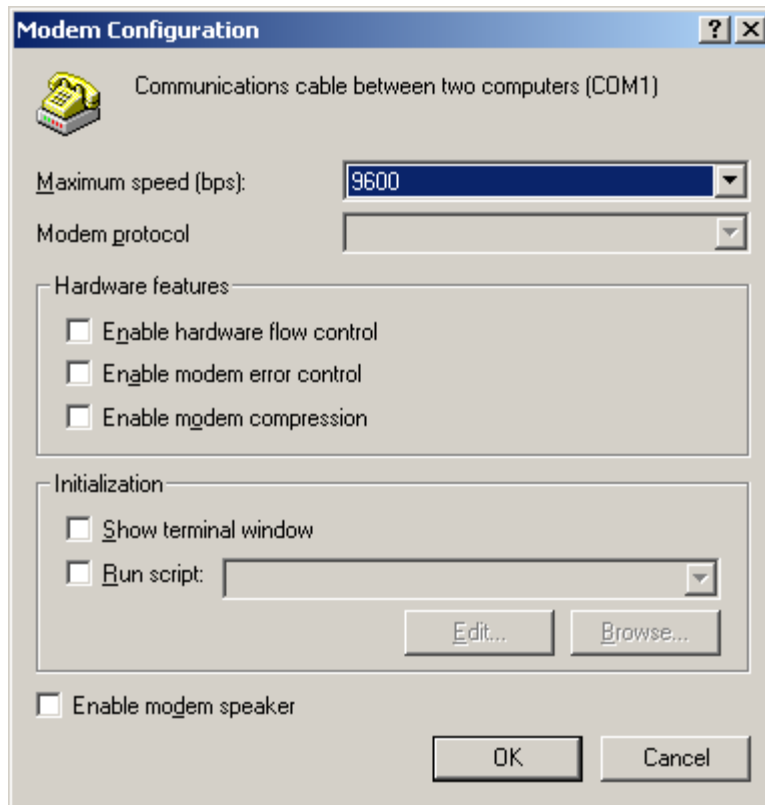
9. Select the **Cancel** button. The *Network and Dial-up Connections* dialog should be visible again.



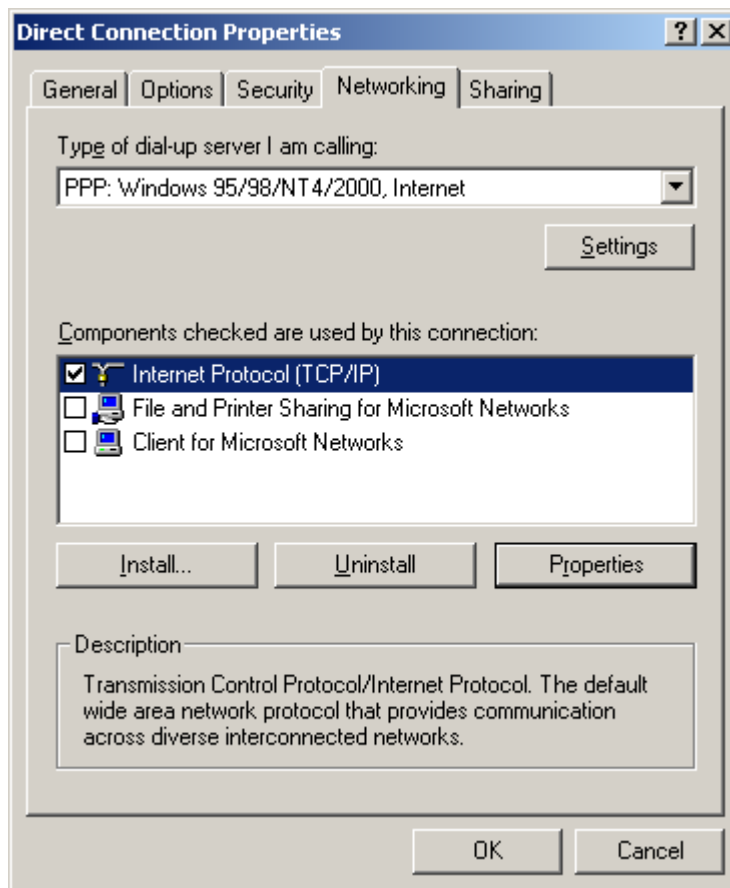
- Right click your new *Direct Connection* icon from the *Network and Dial-up Connections* dialog and select **Properties** from the list. The *Properties* dialog is displayed.



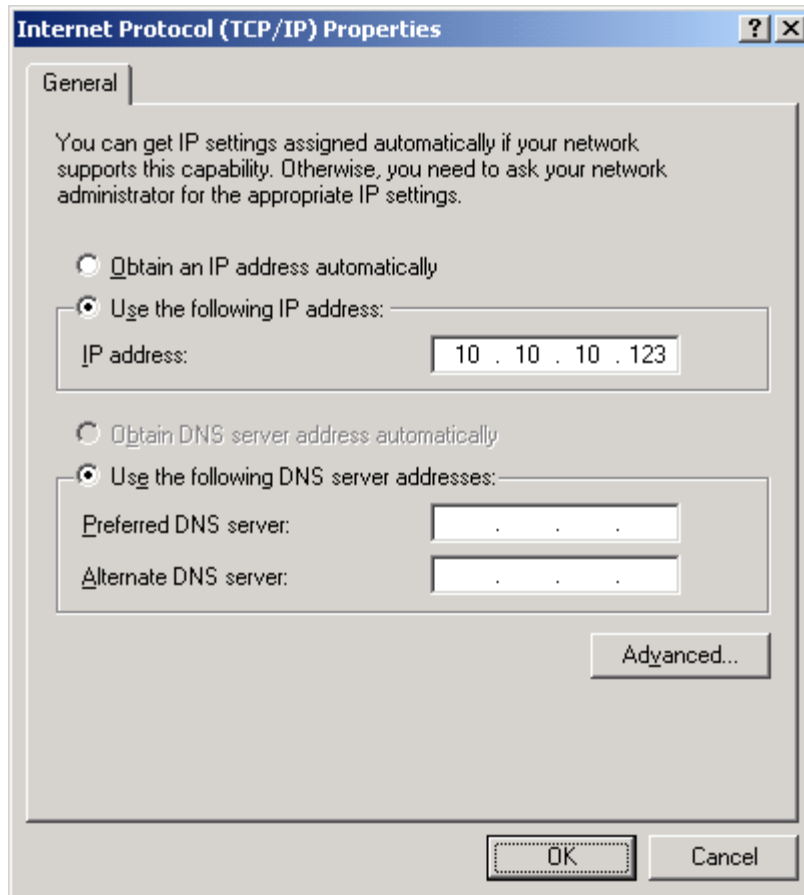
- Select the **Configure** button from the *General* page. The *Modem Configuration* dialog is displayed.



12. There is no modem in this direct serial connection so uncheck all items including *hardware flow control*. Select the baud rate you intend to use (e.g. 9600 bps). Select **OK** to return to the *Properties* dialog.
13. From the *Properties* dialog select the **Networking** page.



Uncheck all components except the component **Internet Protocol (TCP/IP)**. Select the component **Internet Protocol (TCP/IP)** and select the **Properties** button. The *Internet Protocol (TCP/IP) Properties* dialog is displayed.

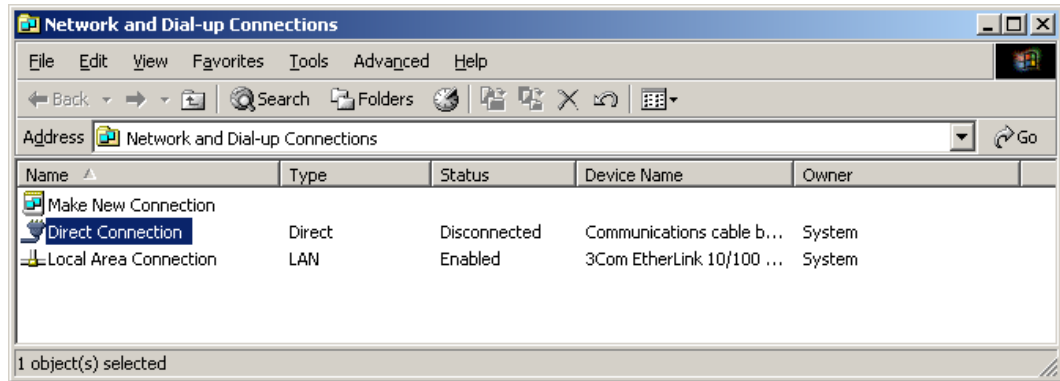


1. The SCADAPack 32 does not have a DHCP server to automatically provide an IP address. Instead the PC's serial port must be given a fixed IP address to use for PPP connections. Select the option **Use the following IP address**. Enter an IP address to assign to your PC's serial port. Obtain this IP address from your Network Administrator. Then select **OK** to return to the *Properties* dialog.
2. Select **OK** again to close the dialog.

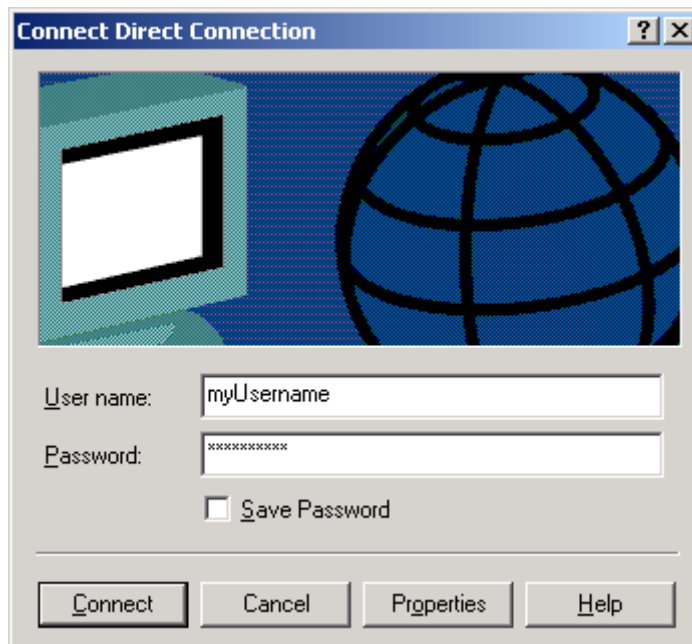
7.8.1.1.2 Making a PPP Connection to the SCADAPack 32

A connection can only be made after successfully setting up a Direct Connection icon as described in the section *Connection Setup* above. Also, a serial port on the SCADAPack 32 must already be configured for the PPP protocol using the *Controller IP Configuration* dialog and must be downloaded to the SCADAPack 32.

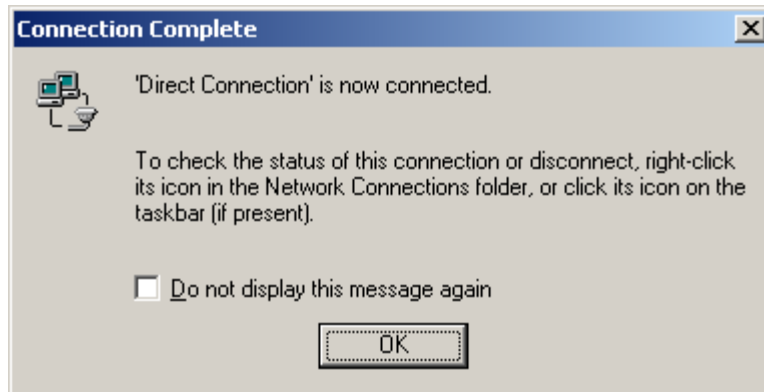
1. From the **Start** menu, double click **Network and Dial-up Connections** from the **Settings** group. The *Network and Dial-up Connections* dialog is displayed.



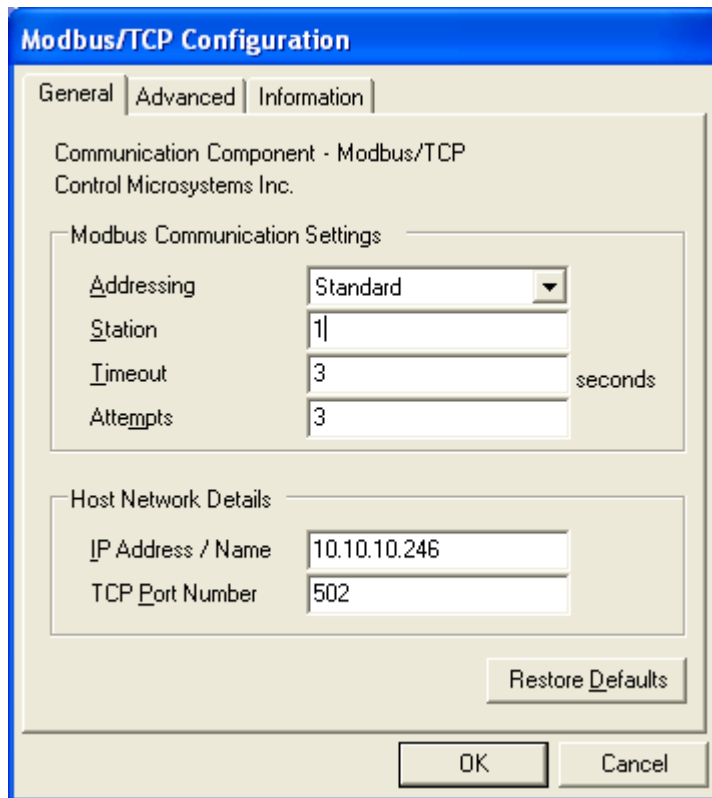
2. Right click your *Direct Connection* icon that was setup in the previous section and select **Connect** from the list. A prompt for username and password is displayed.



3. Enter a valid PAP or CHAP username and password. Valid usernames and passwords are configured on the *PPP Login* page of the *Controller IP Configuration* dialog and must be downloaded to the SCADAPack 32. Then select the **Connect** button. If neither PAP nor CHAP is being used, ignore the prompt and just select the **Connect** button.
4. A progress message is displayed. If the connection is successful the following message is displayed.



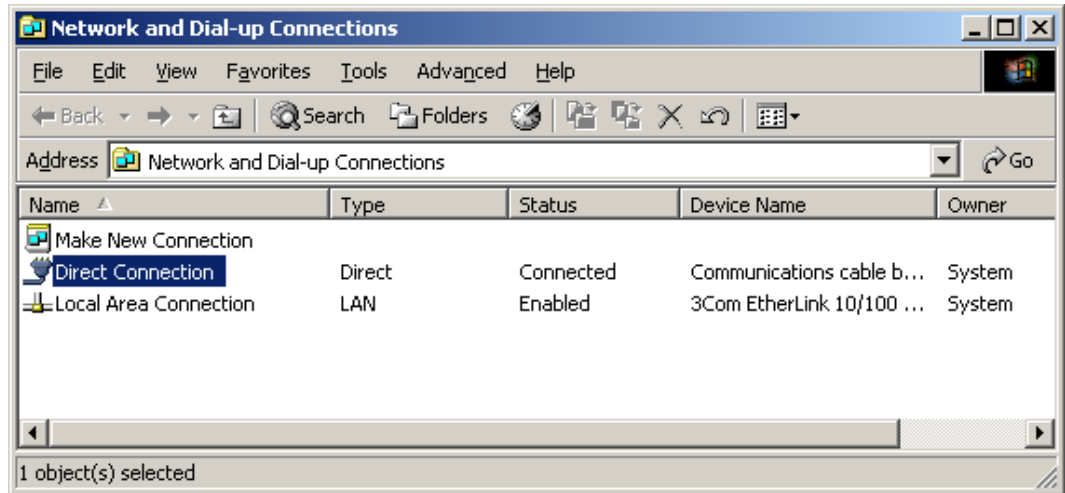
5. You may now connect to the IP address assigned to SCADAPack 32 PPP serial port using an appropriate application and a supported protocol (e.g. Modbus/TCP). In the example below, **Firmware Loader** is used to connect over PPP to the SCADAPack 32. From the *PC Communication Settings* dialog, the IP address assigned to the SCADAPack 32 PPP serial port is selected as the **Connect to Host**.



7.8.1.1.3 Disconnecting a PPP Connection

To disconnect a PPP connection made using the Windows PPP Client, do the following:

1. From the **Start** menu, double click **Network and Dial-up Connections** from the **Settings** group. The *Network and Dial-up Connections* dialog is displayed.



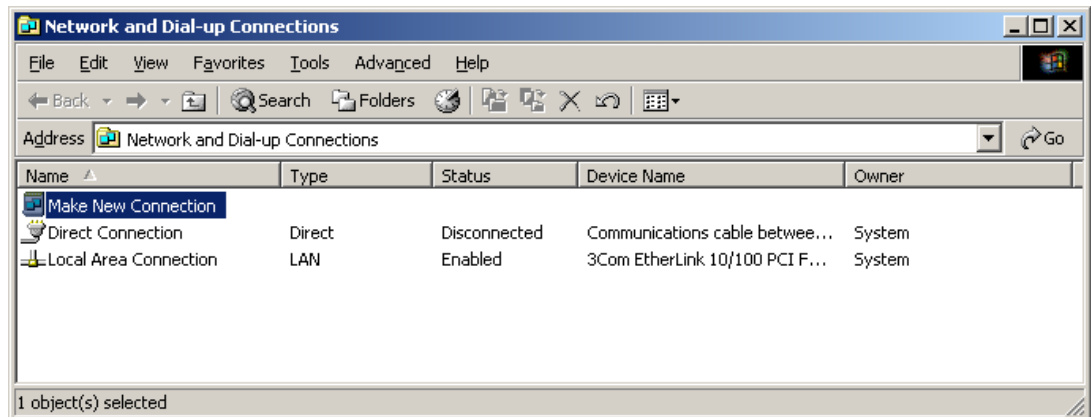
2. Your *Direct Connection* icon should display the word *Connected* in the Status column. To disconnect, right click your *Direct Connection* icon and select **Disconnect** from the list.

7.8.1.2 Dial-up PPP Connection using Windows 2000

7.8.1.2.1 Connection Setup using Windows 2000

Use this connection when a dial-up modem is used to establish a PPP connection between a Windows 2000 PC and a SCADAPack 32.

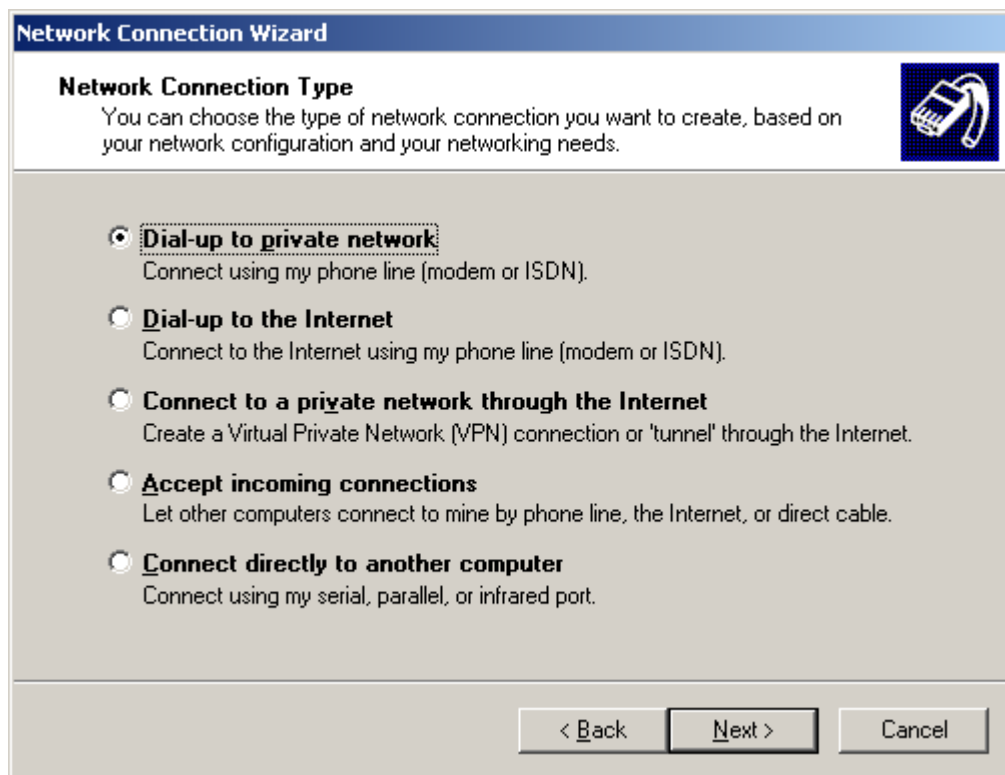
1. From the **Start** menu, right click **Network and Dial-up Connections** from the **Settings** group, and select **Open**. The *Network and Dial-up Connections* dialog is displayed.



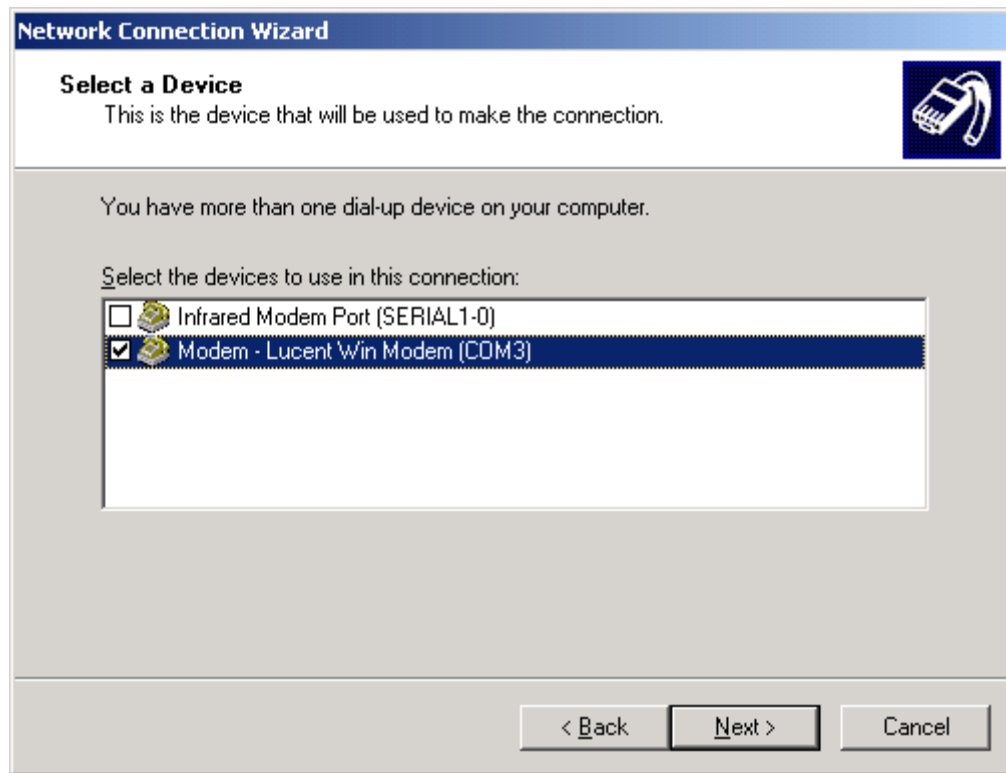
2. Double click the item **Make New Connection** from the *Network and Dial-up Connections* dialog. The connection wizard dialog is displayed.



3. Select the *Next* button to display the connection type options dialog.



4. For Network Connection Type select the type **Dial-up to private network** and select the **Next** button. If there is more than one modem installed on the PC, the *Select a Device* dialog is displayed. If not, proceed to the next step.



5. From the menu select the modem installed on your PC that will be used to connect to the SCADAPack 32. Select the **Next** button. The *Phone Number to Dial* dialog is displayed.

Network Connection Wizard

Phone Number to Dial

You must specify the phone number of the computer or network you want to connect to.

Type the phone number of the computer or network you are connecting to. If you want your computer to determine automatically how to dial from different locations, check Use dialing rules.

Area code: Phone number:

Country/region code:

☐ Use dialing rules

< Back Next > Cancel

6. Enter the phone number to dial (this can be changed later) and select the **Next** button. The **Connection Availability** dialog is displayed.

Network Connection Wizard

Connection Availability

You may make the new connection available to all users, or just yourself.

You may make this connection available to all users, or keep it only for your own use. A connection stored in your profile will not be available unless you are logged on.

Create this connection:

☒ For all users

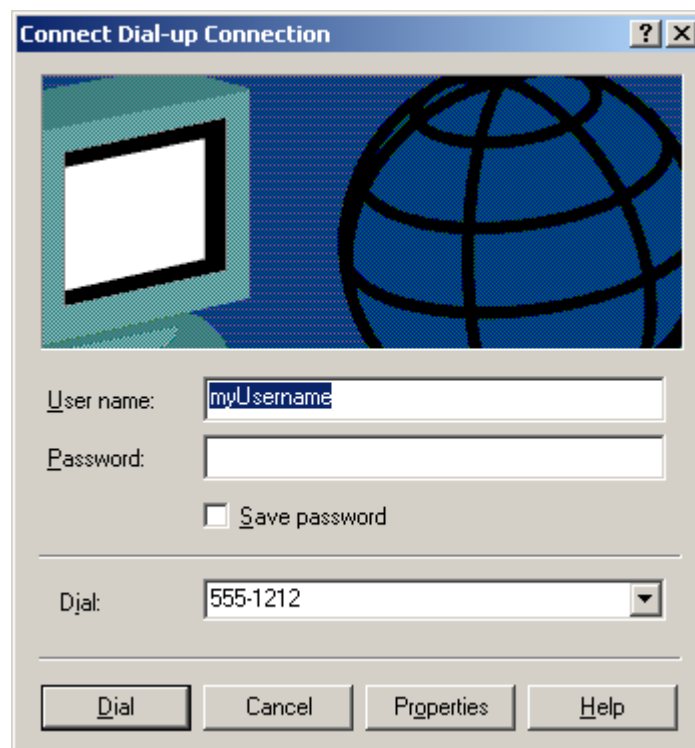
☐ Only for myself

< Back Next > Cancel

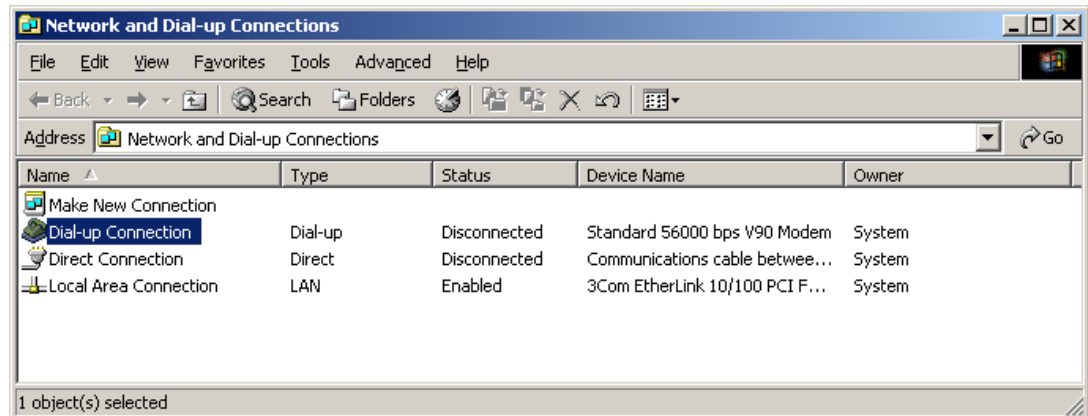
7. Select either option and then select the **Next** button. The *Connection Name* dialog is displayed.



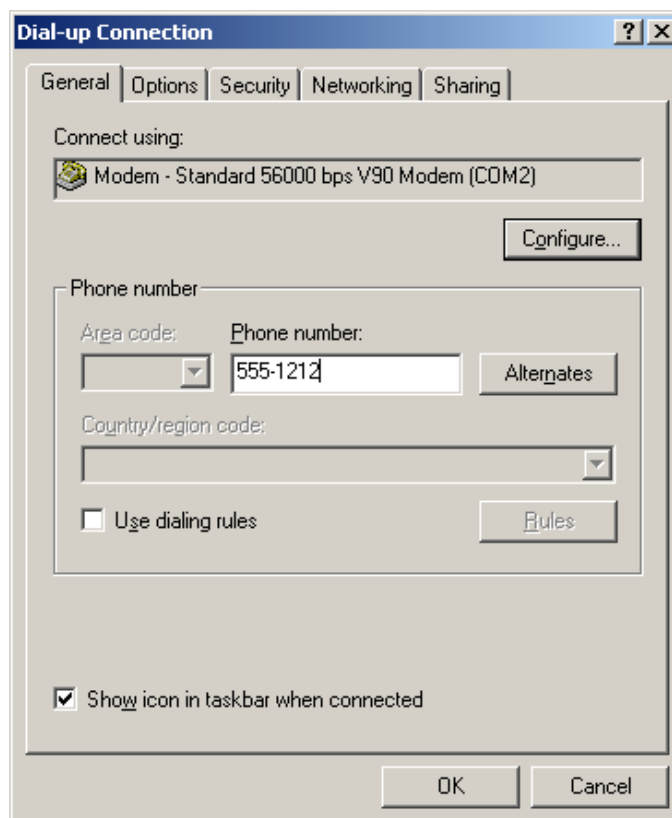
8. Enter a name for the connection and select the **Finish** button. The username and password prompt is displayed.



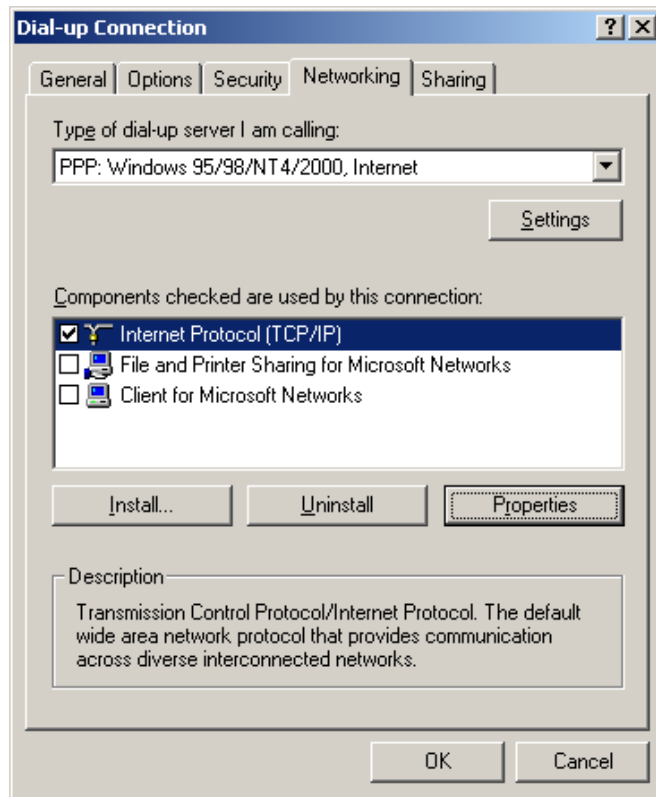
9. Select the **Cancel** button. The *Network and Dial-up Connections* dialog should be visible again.



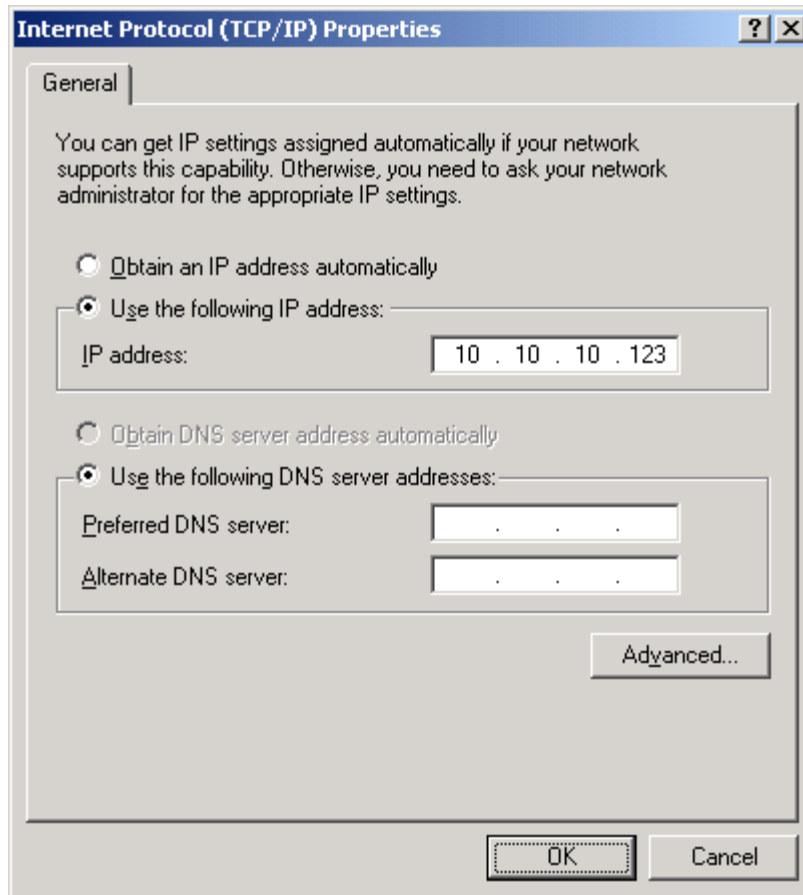
10. Right click your new *Dial-up Connection* icon from the *Network and Dial-up Connections* dialog and select **Properties** from the list. The *Properties* dialog is displayed.



11. From the *Properties* dialog select the **Networking** page.



12. Uncheck all components except the component **Internet Protocol (TCP/IP)**. Select the component **Internet Protocol (TCP/IP)** and select the **Properties** button. The *Internet Protocol (TCP/IP) Properties* dialog is displayed.

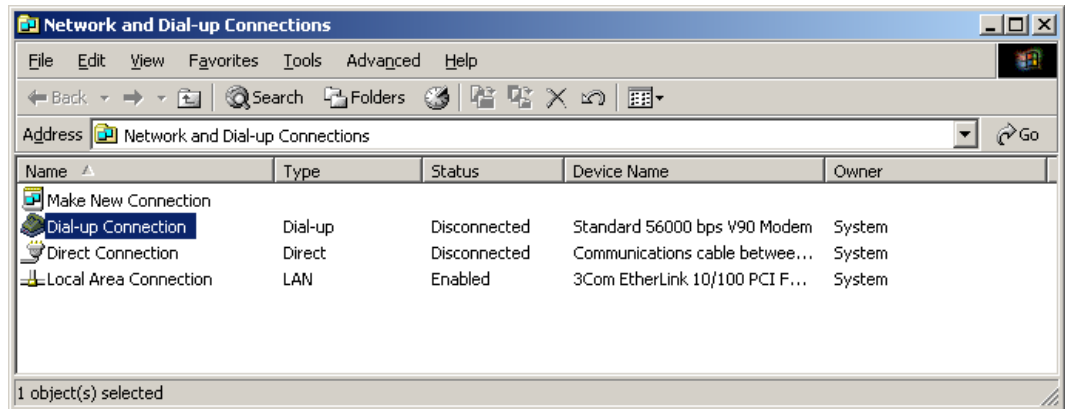


13. The SCADAPack 32 does not have a DHCP server to automatically provide an IP address. Instead the PC's serial port must be given a fixed IP address to use for PPP connections. Select the option **Use the following IP address**. Enter an IP address to assign to your PC's serial port. Obtain this IP address from your Network Administrator. Then select **OK** to return to the *Properties* dialog.
14. Select **OK** again to close the dialog.

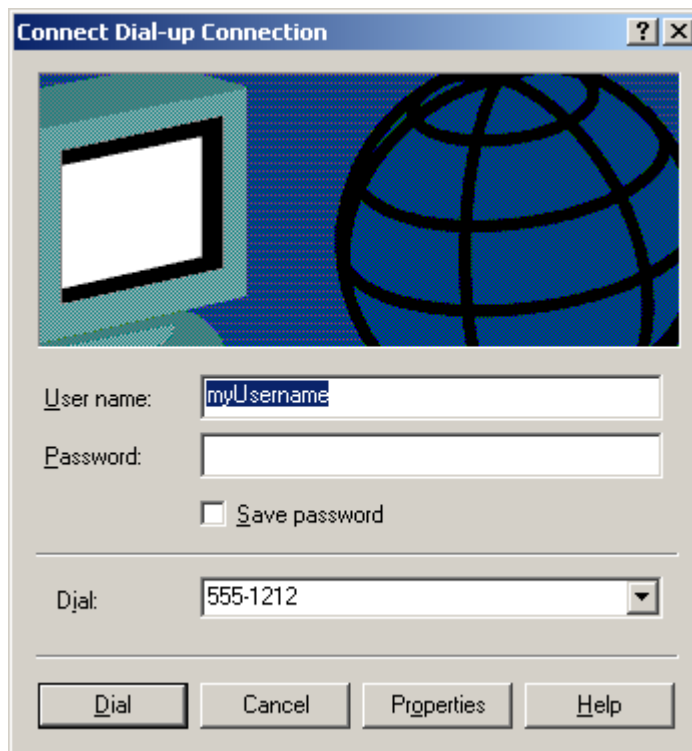
7.8.1.2.2 Making a PPP Dial-up Connection to the SCADAPack 32 using Windows 2000

A connection can only be made after successfully setting up a Dial-up Connection icon as described in the section *Connection Setup* above. Also, a serial port on the SCADAPack 32 must already be configured for the PPP protocol using the *Controller IP Configuration* dialog and must be downloaded to the SCADAPack 32.

1. From the **Start** menu, right click **Network and Dial-up Connections** from the **Settings** group, and select **Open**. The *Network and Dial-up Connections* dialog is displayed.

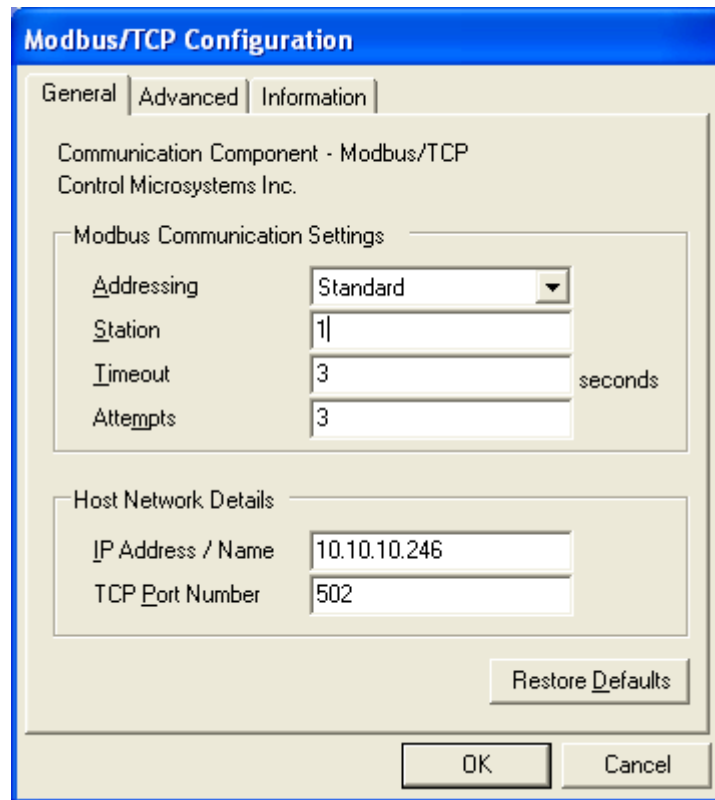


2. Right click your *Dial-up Connection* icon that was setup in the previous section and select **Connect** from the list. A prompt for username and password is displayed.



3. Enter a valid PAP or CHAP username and password. Valid usernames and passwords are configured on the *PPP Login* page of the *Controller IP Configuration* dialog and must be downloaded to the SCADAPack 32. Then select the **Dial** button. If neither PAP nor CHAP is being used, ignore the prompt and just select the **Dial** button.
4. A progress message is displayed. If the connection is successful your *Dial-up Connection* icon should display the word *Connected* in the Status column.
5. You may now connect to the IP address assigned to SCADAPack 32 PPP serial port using an appropriate application and a supported protocol (e.g.

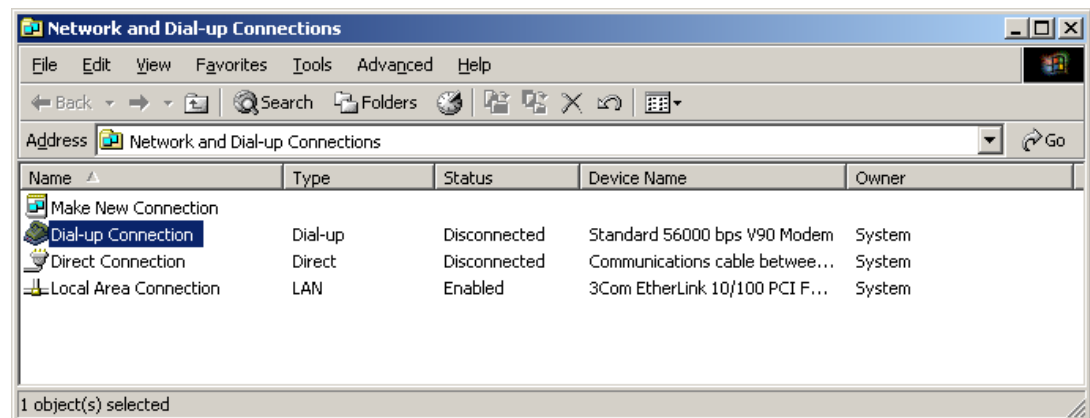
Modbus/TCP). In the example below, **Firmware Loader** is used to connect over PPP to the SCADAPack 32. From the *PC Communication Settings* dialog, the IP address assigned to the SCADAPack 32 PPP serial port is selected as the **Connect to Host**.



7.8.1.2.3 Disconnecting a PPP Connection using Windows 2000

To disconnect a PPP connection made using the Windows PPP Client, do the following:

1. From the **Start** menu, right click **Network and Dial-up Connections** from the **Settings** group, and select **Open**. The *Network and Dial-up Connections* dialog is displayed.



2. Your *Dial-up Connection* icon should display the word *Connected* in the Status column. To disconnect, right click your *Dial-up Connection* icon and select ***Disconnect*** from the list.

8 DNP3 User and Reference Manual

The manual details implementation of the Distributed Network Protocol (DNP3) driver on SCADAPack controllers. While we continuously improve upon the contents of this manual to simplify the driver configuration tasks, we also assume that users attempting to configure the DNP protocol on a SCADAPack controller have some preliminary understanding of the DNP3 communication protocol.

8.1 DNP3 Protocol Overview

DNP, the Distributed Network Protocol, is a standards-based communications protocol developed to achieve interoperability among systems in the electric utility, oil & gas, water/waste water and security industries. This robust, flexible non-proprietary protocol is based on existing open standards to work within a variety of networks.

DNP offers flexibility and functionality that go far beyond conventional communications protocols. Among its robust and flexible features DNP 3.0 includes:

- Multiple data types (Data Objects) may be included in both request and response messages.
- Multiple master stations are supported for outstations.
- Unsolicited responses¹ may be initiated from outstations to master stations.
- Data types (Objects) may be assigned priorities (Class) and be requested based on the priority.
- Addressing for over 65,000 devices on a single link.
- Time synchronization and time-stamped events.
- Broadcast messages.
- Data link and application layer confirmation
- Internal indications that report the health of a device and results of last request.
- Select-Before-Operate which is the ability to choose extra reliability when operating outputs.

8.1.1 DNP Architecture

DNP is a layered protocol that is based on the Open System Connection (OSI) 7-layer protocol. DNP supports the physical, data link and application layers only and terms this the Enhanced Performance Architecture (EPA). In addition to these three layers an additional layer, the pseudo-transport layer, is added to allow for larger application layer messages to be broken down into smaller frames for the data link layer to transmit.

Object Library	The data objects (Binary Inputs, Binary Outputs, and Analog Inputs etc.) that reside in the master or outstation.
Application Layer	Application tasks for sending of solicited requests (master messages) to outstations or sending of unsolicited responses from outstations. These request and response messages are referred to as fragments in DNP.
Pseudo-Transport Layer	Breaks the application layer messages into smaller packets that can be handled by the data link layer. These packets are referred to as frames in DNP.

¹ Unsolicited responses are also known as unsolicited messages

Data Link Layer	Handles the transmission and reception of data frames across the physical layer.
Physical Layer	This is the physical media, such as serial or Ethernet, which DNP communicates.

These layers are described in the following sections of this manual.

8.1.1.1 Object Library

The data types that are used in DNP are broadly grouped together into Object Groups such as Binary Input Objects and Analog Input Objects etc. Individual data points, or objects within each group, are further defined using Object Variations such as Binary Input Change with Time and 16-Bit Analog Inputs for example.

In general there are two categories of data within each data type, static objects and event objects. Static objects contain the current value of the field point or software point. Event objects are generated as a result of the data changing.

In addition to the object group and variation data objects can be assigned to classes. In DNP there are four object classes, Class 0, Class 1, Class 2 and Class 3. Class 0 contains all static data. Classes 1, 2 and 3 provide a method to assign priority to event objects. While there is no fixed rule for assigning classes to data objects typically class 1 is assigned to the highest priority data and class 3 is assigned to the lowest priority data.

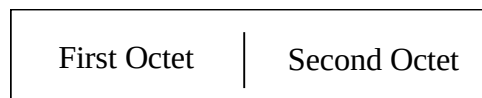
This object library structure enables the efficient transfer of data between master stations and outstations. The master station can poll for high priority data (class 1) more often than it polls for low priority data (class 3). As the data objects assigned to classes is event data when the master polls for a class only the changed, or event data, is returned by the outstation. For data in an outstation that is not assigned a class the master uses a class 0 poll to retrieve all static data from the outstation.

DNP allows outstations to report data to one or more master stations using unsolicited responses (report by exception) for event data objects. The outstation reports data based on the assigned class of the data. For example the outstation can be configured to only report high priority class 1 data.

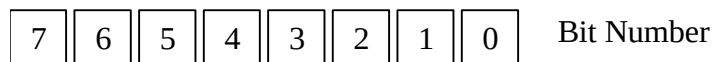
8.1.1.1.1 Internal Indication (IIN) Flags

An important data object is the Internal Indications (IIN) object. The Internal Indication (IIN) flags are set by a slave station to indicate internal states and diagnostic results. The following tables show the IIN flags supported by SCADAPack controllers. All bits except *Device Restarted* and *Time Synchronization required* are cleared when the slave station receives any poll or read data command.

The IIN is set as a 16 bit word divided into two octets of 8 bits. The order of the two octets is:



8.1.1.1.1.1 IIN First Octet



First Octet Bit	Description
0	last received message was a broadcast message
1	Class 1 data available

First Octet Bit	Description
2	Class 2 data available
3	Class 3 data available
4	Time Synchronization required
5	not used (returns 0)
6	Device trouble - Indicates memory allocation error in the slave, or - For master in mimic mode indicates communication failure with the slave device.
7	Device restarted (set on a power cycle)

8.1.1.1.1.2 IIN Second Octet

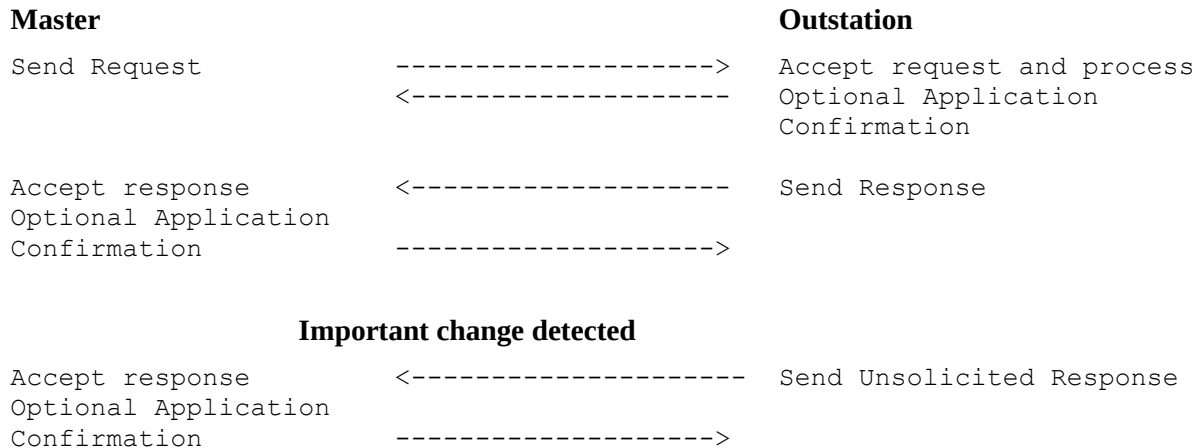
7	6	5	4	3	2	1	0	Bit Number

Second Octet Bit	Description
0	Function Code not implemented
1	Requested object unknown or there were errors in the application data
2	Parameters out of range
3	Event buffer overflowed Indicates event buffer overflow in the slave or master. The slave will set this bit if the event buffer in the slave is overflowed. The master will set this bit if the event buffer in the master has overflowed with events read from the slave. Ensure the event buffer size, in the master and slave, is set to a value that will ensure the buffer does not overflow and events are lost.
4	not used (returns 0)
5	not used (returns 0)
6	not used (returns 0)
7	not used (returns 0)

8.1.1.2 Application Layer

The application layer in DNP is responsible for the processing of complete messages for requesting, or responding to requests, for data.

The following shows the sequence of Application Layer messages between one master and one outstation.



The complete messages are received from and passed to the pseudo-transport layer. Application layer messages are broken into fragments with each fragment size usually a maximum of 2048 bytes. An application layer message may be one or more fragments in size and it is the responsibility of the application layer to ensure the fragments are properly sequenced.

Application layer fragments are sent with or without a confirmation request. When a confirmation is requested the receiving device replies with a confirmation indicating the message was received and parsed without any errors.

8.1.1.3 Pseudo-Transport Layer

The pseudo-transport layer formats the larger application layer messages into smaller packets that can be handled by the data link layer. These packets are referred to as frames in DNP. The pseudo-transport layer inserts a single byte of information in the message header of each frame. This byte contains information such as whether the frame is the first or last frame of a message as well as a sequence number for the frame.

8.1.1.4 Data Link Layer

The data link layer handles the transmission and reception of data frames across the physical layer. Each data link frame contains a source and destination address to ensure the receiving device knows where to send the response. To ensure data integrity data link layer frames contain two CRC bytes every 16 bytes.

Data link layer frames are sent with or without a confirmation request. When a confirmation is requested the receiving device replies with a confirmation indicating the message was received and the CRC checks passed.

8.1.1.5 Physical Layer

The physical layer handles the physical media, such as serial or Ethernet, which DNP communicates.

8.1.2 Modbus Database Mapping

In SCADAPack controllers static DNP objects such as binary input, analog input, binary counter and analog output are associated with Modbus registers. Whenever a DNP object is created an associated Modbus register(s) is also assigned. Application programs executing in the SCADAPack controller, C or logic, are able to assign physical I/O to Modbus registers using the TelePACE Register Assignment or the ISaGRAF I/O Connection and these physical I/O points can then be assigned to DNP objects. User application data such as runtimes, flow totals etc. may be also be assigned to DNP objects.

This architecture enables DNP master stations and outstations to pass not only physical data points between them but also to monitor and control user applications executing in the SCADAPack controller. For example a master station can monitor a level in an outstation and then, based on the application program, send a setpoint value to another outstation to control the level.

8.1.3 SCADAPack DNP Operation Modes

Within a DNP network, a SCADAPack controller can operate as a:

- DNP Outstation (Slave)
- DNP Master or Mimic Master or
- DNP Router

DNP Master Mimic and DNP Router are incompatible and mutually-exclusive modes of operation.

A DNP outstation forms the basic class of any DNP node in a network. All other operational modes derive from a DNP Outstation. A DNP outstation responds to requests from one or more DNP master stations on a network. Also, a DNP Outstation is able to initiate unsolicited responses (messages) based on event data to a master station.

A DNP Master is capable of polling for data, accepting and processing unsolicited messages, and sending control commands to an outstation. Note that a DNP Master can also act perform all the duties of a DNP Outstation.

A SCADAPack controller acting as a DNP Router is simply acting a pass through, basically redirecting messages from one DNP node to another. Similarly to a DNP Master, a DNP Router can also perform all the duties of a DNP Outstation.

DNP Network topologies comprise several combinations of DNP Masters, DNP Routers, and DNP Outstations. Typical configurations possible with SCADAPack controllers are:

- DNP Master and single DNP Outstation
- DNP Master and multi-dropped DNP Outstations
- DNP SCADA Host, Data Concentrator (Mimic Master) and multi-dropped DNP Outstations
- DNP SCADA Host, DNP Router and multi-dropped DNP Outstations

Major SCADAPack DNP operation modes are covered in the next chapters.

8.2 SCADAPack DNP Outstation

A DNP3 Outstation can be considered the base class of all terminal nodes on a DNP network. All other DNP3 configuration modes, such as Master, Mimic Master or Router, as implemented by the Control Microsystems DNP driver, inherit their properties from the outstation base class. In other words, a SCADAPack controller can simultaneously take on any other operation mode, in addition to being a DNP outstation.

When configured as a DNP outstation a SCADAPack controller is able to:

- Map physical I/O data to DNP points.
- Define DNP points as Class 0 (Static or None), Class 1, Class 2 or Class 3 data types.
- Respond to requests from one or more master stations such as a SCADA hosts or other SCADAPack controllers capable of operating as DNP Masters.
- Initiate unsolicited responses to one or more master stations.

Note: ‘Unsolicited responses’ are also known as ‘unsolicited messages’. ‘Unsolicited messages’ will be used predominantly in this document.

One distinguishing feature of a DNP outstation is this ability to trigger unsolicited messages to a master, upon event accumulation. Events are accumulated when the state of a DNP point changes or an analog values exceeds a threshold. Dead bands can be used to filter out noise from being reported as event data.

After accumulating a certain number of DNP events, or if a certain time period has expired, a DNP outstation will trigger an unsolicited message all its configured master DNP stations, reporting event data. As defined by the DNP specification, an outstation that triggers an unsolicited message expects a confirmation from all the targeted masters (or peers). If an acknowledgement is not received with a configured **Application Layer** timeout, the outstation will retransmit the initial unsolicited message. If no response is received within the Application Layer timeout, the outstation will retransmit again. This process continues until the outstation has retransmitted the message a number of times as configured by its **Application Layer Retries** parameter.

If all retry attempts fail, this message is discarded from the transmit buffer. As of this writing, re-transmission of the failed message will only resume after a new event occurs within the appropriate buffer. Future releases of the SCADAPack DNP driver will re-attempt a failed DNP transaction after a random period of time has expired. Retransmissions will be attempted until the messages are eventually received by the master.

Application Layer messages that are larger than **249 bytes** are broken down into Data Link frames. The DNP protocol allows one to configure acknowledgements of individual Data Link frames, this enhancing network robustness, especially under noisy environments. When the underlying network structure is noise free (wired or networks for instance), enabling Application and Data Link confirmations are not necessary.

8.2.1 How to Configure SCADAPack DNP Outstation

In this exercise, we will configure a DNP outstation with address 10. We will also configure the station with digital input points associated with Class 1 and Class 2 events. The station will be configured to trigger unsolicited messages to Master station 200, when Class 1 and Class 2 events occur on these digital inputs.

After this exercise, you should be able to:

- Enable the DNP protocol on a serial port.
- Configure the DNP Application and Data Link Layers
- Configure Class Events Generation and Transmission
- Configure a DNP Routing table
- Configure DNP points.

We will map two digital inputs mapped to Modbus registers 10001 and 10002 to DNP Addresses 1 and 2.

8.2.1.1 Tasks to Complete

Enable DNP Protocol on communication interface.

Configure a DNP Outstation with station address.

Configure DNP points and assign them to Class objects.

Configure outstation to be able to trigger unsolicited messages.

8.2.1.2 Enable DNP on Communication Interface

The first step recommended in configuration the DNP driver on a SCADAPack controller is to enable DNP on the communication interface. To enable the DNP protocol on com2,

1. From the **Controller** menu in either TelePACE or ISaGRAF, select **Serial Ports**.
2. Select **COM2** from the **Port** drop down list.
3. Set the Protocol type to **DNP**.
4. Click on **OK**.
5. If using an Ethernet equipped controller, enable **DNP in TCP** or **DNP in UDP** from the Controller IP configuration dialog.

8.2.1.3 Configure DNP Outstation

6. From the **Controller** Menu in either TelePACE or ISaGRAF, select **DNP Configuration** to launch the DNP Configuration dialog.
7. The **Application Layer** configuration panel is displayed by default.
 1. Under the Communication group box, change the **Retries** parameter to 2.
 2. Leave all other parameters under the Communication group box at default values.

TIP: It is not necessary to enable the Application Layer confirmation as unsolicited events, by their nature, request for an Application Layer confirmation.

3. Set **Time Synchronization** to None.

TIP: It is recommended that a DNP3 master initiate time synchronization.

4. Enable Unsolicited Class 1 events.
5. For Class 1 Events, set a **Hold Time** of 5 seconds and a **Hold Count** of 100.

TIP: On systems with multiple outstations that could potentially transmit unsolicited messages to a master at the same time, it is recommended to use a combination of the Hold Time and Hold Count parameters to avoid multiple stations from transmitting at the same time.

6. Enable Unsolicited Class 2 events.
7. For Class 2 Events, set a Hold Time of **3600** seconds and a Hold Count of **10**.

TIP: Class 2 events are typically of less importance than Class 1 events and may not need to be reported immediately to the master

8. All other parameters can be left at their default values. The completed Application Layer Configuration panel should look like this:

The screenshot shows the 'DNP Settings' dialog box with the 'Application Layer' tab selected. The left sidebar lists various configuration categories, with 'Application Layer' highlighted. The main panel contains the following settings:

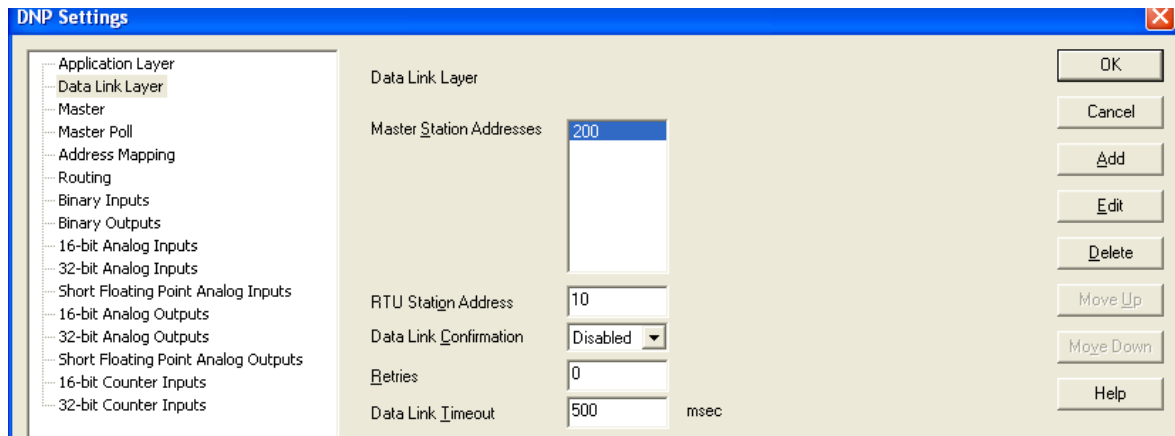
- Application Layer**
 - Communication**
 - Application Confirmation: Disabled
 - Maximum Fragment Length: 2048
 - Retries: 2
 - Application Timeout: 5000 msec
 - Time Synchronization**
 - None (selected)
 - At Start Up Only
 - Interval: 0 minutes
 - Unsolicited Responses**

	Class 1	Class 2	Class 3
Enable Unsolicited	Enabled	Enabled	Disabled
Hold Time	1	3600	60
Hold Count	100	10	10
Enable Unsolicited Responses on Start Up	Yes		
Send Initial Unsolicited Response on Start Up	No		
 - Dial Up**
 - Modem Initialization: &F0 S0=1 &W0 &Y0
 - Connect Timeout: 45 seconds
 - Attempts: 2
 - Inactivity Timeout: 45 seconds
 - Dial Type: Tone
 - Pause Time: 10 seconds
 - Operate Timeout**: 15 seconds
 - Report only Level 2 Objects in Class Polls**: No
 - ☐ **Limit Maximum Events in Read Response**: 0

At the bottom left, there is a checkbox labeled 'Allow Duplicate Modbus Addresses' which is currently unchecked. On the right side of the dialog, there are buttons for OK, Cancel, Add, Copy, Delete, Move Up, Move Down, and Help.

Note: Clicking on OK closes the DNP Configuration dialog. Click on OK only after you have completed the DNP configuration.

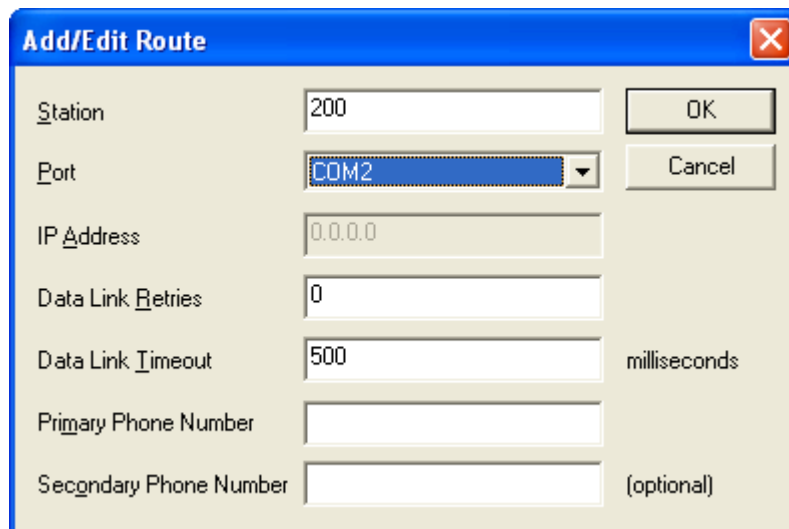
8. From the DNP Configuration panel, select the **Data Link Layer** tree node.
 - a. Click on the **Edit** button and change the Master **Station Address** to 200.
 - b. Change the **RTU Station Address** to 10.
 - c. Leave all other parameters at their default values. The completed dialog should look like this:



TIP: It may be necessary to enable the Data Link confirmation on noisy networks. However, if the Maximum Application Fragment Length is reduced to 249 bytes, it is not necessary to enable the Data Link confirmation, as each data link packet is in essence an Application Layer fragment.

9. From the DNP Configuration panel, select the Routing tree node.

- a. Click on the **Add** button to begin a new routing table entry.
- b. From the Add/Edit Route dialog,
 - i. Enter 200 for the destination **Station**.
 - ii. Set the **Port** to COM2.
 - iii. Leave default values for all other parameters.
 - iv. The completed dialog should look like this:

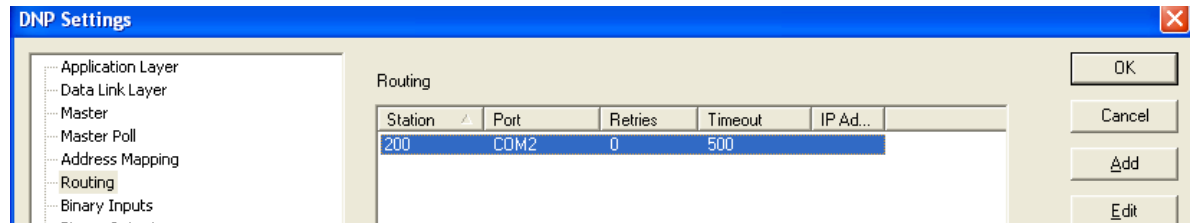


Note: The Data Link Timeout in this dialog takes precedence over the Data Link Timeout in the Data Link Layer configuration panel.

TIP: Even though a SCADAPack outstation will respond successfully to master request, without is routing entry to the master, it is a good practice to always define such a routing entry from an outstation to its master. Moreover, without a routing entry defined to the master, the outstation will not know which port to send out unsolicited messages, if configured, to the master.

- v. Click on **OK** to add this entry to the routing table and return to the Routing dialog.

The completed routing table should look like this:



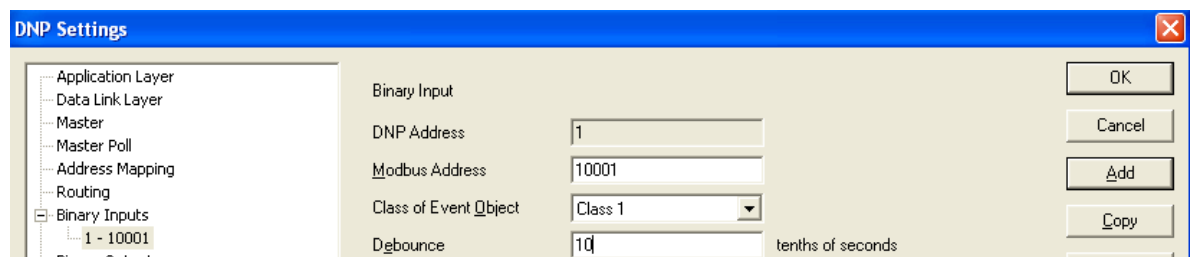
This next step assumes you have digital inputs mapped to Modbus registers 10001 and 10002.

10. From the DNP Configuration Panel, click on the Binary Inputs tree node.

- a. Set the **Starting DNP Address** to 1.
- b. Set the **Event Report Method** to **Log All Events**.

Note: If you want to log all events and not only the most recent, you must set the Event Reporting Method to Log all Events.

- c. Set the **Event Buffer Size** to 100. The completed panel should look like this:



- d. Click on **Add** to create a new DNP3 binary input point. Observe that a new binary input point is now visible under the Binary Input tree node with DNP Address 1 (Starting Address)
- e. Leave the default associating **Modbus Address** as 10001.
- f. Leave the default **Event Object** as Class 1.
- g. Set the **Debounce** property to 10.

TIP: It is a good idea to set a non- zero Debounce on unfiltered inputs, to avoid noise being collected as Class events. The same applies for analog inputs. A non-zero Deadband will prevent noise from being collected as Class events.

9. Set the **Debounce** property to 10.
10. Click on **Add** to submit this point to the database and start configuration for the next point. Note that a new point has been added under the Binary Inputs tree node in the DNP Configuration panel.
11. Change the associating **Modbus Address** to 10002.
12. Change the **Event Object** to Class 2.
13. Set the **Debounce** appropriately.
14. Click on **Add** to submit this point to the database and start configuration for the next point.
15. Repeat the previous two steps to add more points if desired.
16. Follow a similar procedure to configure other types of DNP3 objects.

8.2.1.4 Confirm Successful Configuration

To confirm that the DNP driver has been properly configured,

11. From the Controller menu, select **DNP Status**. You will be presented with the following dialog.

DNP Status

Overview | Binary In | Binary Out | AIN-16 | AIN-32 | AIN-Float | AOUT-16 | AOUT-32 | AOUT-Float | Counter-16 | Counter-32

DNP Status: 07: enabled, configured, running

Internal Indications: 0000

Communication Statistics

Direction	com1	com2	com3	com4	IP
Transmit	0	0	0	0	0
Receive	0	0	0	0	0
Successes	0	0	0	0	0
Fails	0	0	0	0	0
FailsNew	0	0	0	0	0

Reset

Last Message

Direction	Time	Port	Source	Dest	Length	Link Func	Appl Func	IIN
Transmit	29Jun07 13:14:21.92	com2	10	200	10	Send/No Reply	Response	0000
Receive	29Jun07 13:14:21.91	com2	200	10	11	Send/No Reply	Read	

Event Buffers

Binary In	AIN-16	AIN-32	AIN-Float	Counter-16	Counter-32	Class 1	Class 2	Class 3
0/3000	0/16	0/16	0/3000	0/16	0/16	0	0	0

Close

12. Ensure that the **DNP Status** field within this dialog displays **07: enabled, configured, running**.

13. You can also monitor the current state of the defined DNP binary input points from the **Binary-In** tab.
14. Toggle the state of digital input 1 configured earlier in this exercise and observe the event buffer for Binary Inputs increment on each change of state. After 5 seconds has elapsed, notice that an unsolicited DNP message is triggered to master station 200. Given that DNP master station 200 is not yet configured and connected, a response to the unsolicited message will not be received and the 5000ms Application layer timeout period will expire. The unsolicited message transmission will subsequently retransmitted and will be aborted after 3 retry attempts have been made. This confirms that your outstation is properly setup and unsolicited messages are being generated and sent. At the time of this implementation, the events will be re-attempted only after a new event occurs.
15. Also observe the Internal Indications show that Class 1 events are available as indicated in the figure below.

DNP Status

Overview | **Binary In** | Binary Out | AIN-16 | AIN-32 | AIN-Float | AOUT-16 | AOUT-32 | AOUT-Float | Counter-16 | Counter-32

DNP Status: 07: enabled, configured, running

Internal Indications: 0200: Class 1,

Communication Statistics

Direction	com1	com2	com3	com4	IP
Transmit	0	3	0	0	0
Receive	0	0	0	0	0
Successes	0	0	0	0	0
Fails	0	1	0	0	0
FailsNew	0	1	0	0	0

Reset

Last Message

Direction	Time	Port	Source	Dest	Length	Link Func	Appl Func	IIN
Transmit	04Jul07 21:09:12.07	com2	10	200	62	Send/No Reply	Unsolicited	0000
Receive	04Jul07 21:03:06.37	com2	200	10	14	Send/No Reply	Write	

Event Buffers

Binary In	AIN-16	AIN-32	AIN-Float	Counter-16	Counter-32	Class 1	Class 2	Class 3
7/3000	0/16	0/16	0/3000	0/16	0/16	7	0	0

For additional information on the any of the dialogs referenced in the above exercise, refer to the **DNP Configuration Menu Reference**.

8.3 SCADAPack DNP Master

DNP master modes currently apply only to the SCADAPack 32, SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203 controllers.

As a master, a SCADAPack controller can be a regular Master or Mimic master.

8.3.1 SCADAPack DNP Master Concepts

A DNP Master station inherits all the characteristics of a DNP Outstation. In addition, a DNP Master station is able to:

- Poll DNP outstations for static (Class 0) data and Class 1, 2 and 3 event data.
- Accept and process unsolicited response messages from polled outstations.

This configuration of a DNP Master (Client) and DNP Outstation (Server) forms the basis of a DNP3 Network. The SCADAPack DNP Master may be configured to periodically poll a SCADAPack DNP Outstation for Class 0, 1, 2, and 3 data objects and receive unsolicited responses from the outstation. The outstation may be configured to report change event data to the master station using unsolicited responses.

The arrowed line between the master and outstation in the diagram below represents a communication path connecting the two stations. This communication medium may be any type that is supported by both controllers, such as direct serial, leased line modem, dial-up modem and radio for example.

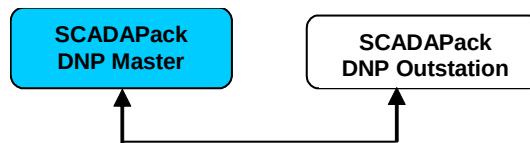


Figure 8-1: Simple SCADAPack Master-Outstation DNP Network

An extension of a simple DNP Master and single outstation network, involves a SCADAPack DNP Master connected to a number of outstations over a multi-drop communication channel. The DNP Master may be configured to periodically poll each SCADAPack DNP Outstation for Class 0, 1, 2, and 3 data objects and receive unsolicited responses from the outstations. The outstations may be configured to report change event data to the master station using unsolicited responses.

The arrowed line between the master and outstations, in the diagram below, represents the communication path connecting the stations. This communication path may be any multi-dropped type that is supported by the controllers, such as leased line modem, dial-up modem and radio for example.

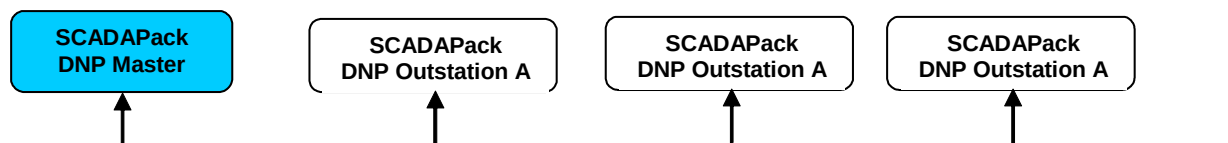


Figure 8-2: SCADAPack DNP Master and multi-dropped DNP Outstations

Note: The DNP Master feature is limited to a SCADAPack32, SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADASense 4203

8.3.1.1 SCADAPack DNP Mimic Master

In a typical DNP network a SCADA Host master communicates with a number of outstations. The SCADA Host will poll each outstation for data and may receive change event data in the form of unsolicited responses from the outstations. This type of DNP network is shown in the following diagram.

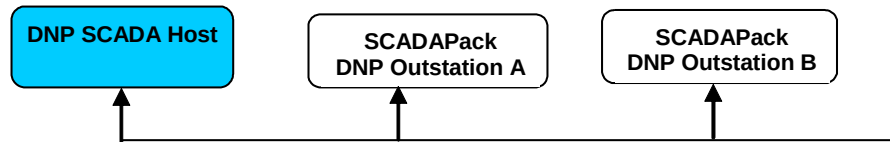


Figure 8-3: DNP SCADA Host and multi-dropped DNP Outstations

In the above configuration the SCADA Host manages the communication path with each outstation. When the communication path is slow, such as with dial-up communication, or subject to high error rates, such as with some radio communication, the data update rate at the SCADA host can become very slow.

Adding a SCADAPack controller configured for Master Mimic Mode, allows for the SCADA Host to poll the SCADAPack (Mimic Master) for all outstation data instead. In essence, the SCADAPack Mimic Master is acting as a Data Concentrator, reporting on behalf of all the outstations currently configured in its routing table. The following diagram shows the addition of the SCADAPack Mimic Master.

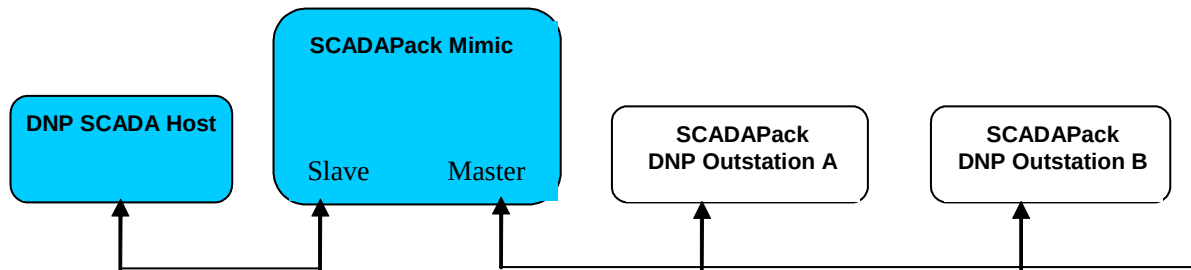


Figure 8-4: SCADAPack Mimic Master and multi-dropped DNP Outstations

In this configuration the outstation side of the network has been decoupled from the host side of the network, as the SCADAPack mimic master now manages all the communication with the outstations.

The SCADA Host and all outstations will typically be connected to different communication ports of the SCADAPack Mimic Master. The mimic will respond to the following DNP messages on behalf of the targeted station:

- Read messages (this includes class polls as well as individual point reads) from SCADA Host
- Write messages from SCADA Host
- Unsolicited messages from an outstation
- Direct operate messages from SCADA Host

The following DNP messages cannot be mimicked (Mimic does not respond on behalf of target DNP station), and are routed directly to the target outstation by the Mimic:

- Select and Operate messages
- Data Link Layer messages (e.g. get link status, reset link status, etc)
- Enable/Disable Unsolicited Message commands (FC 20 and 21)
- Other control messages

Routing for those messages that cannot be mimicked is subjected to the following rule:

```
if (a message is received which needs to be retransmitted to someone else)
    if (the message target is configured in our routing table)
        if (the destination port is different from the incoming port)
            or (routing is enabled on the incoming port)
                then retransmit the message
```

In order to provide current outstation data to the SCADA Host, the SCADAPack mimicking master independently communicates with each outstation to update a local copy of its database with data from the outstations. This communication may be initiated by the SCADAPack mimicking master, either by polling each outstation in turn using solicited messages; or the outstations could initiate unsolicited messages back to the mimicking master. There could also be a combination of solicited and unsolicited messages between the mimicking master and the outstations.

In the Mimic mode diagram above the SCADAPack mimic master polls each outstation, A and B, for data and holds images of this data in its memory. When the SCADA Host poll outstations A and B for data, the mimic master replies from its own images of the outstations. The SCADA Host can also poll the SCADAPack master for its own local data.

Typically the messaging strategy chosen will depend on the relative importance of the data, and the required maximum end-to-end delays for data being transferred through the network. If the requirement is for a reasonably short end-to-end delay for all data points, a round-robin polling scheme is best, without any unsolicited messages. If there are some data points, which are higher priority and must be transferred as fast as possible, unsolicited messages should be used.

The advantage of having the SCADA system communicating with the SCADAPack 32 mimic, instead of direct communication to the outstations is that communication delays and high error rates are effectively removed. The physical connection between the SCADA system and mimic master SCADAPack is typically a direct high-speed reliable connection and all message transactions are fast. Outstations may often be connected via slow PSTN or radio links, and therefore message transactions are subject to substantial delays. They may also be unreliable communication links subject to high error rates.

By having a multiple-level network the communication between the SCADAPack master and outstations is separated from communication between SCADA system and the SCADAPack master. The delays and error rates, which may be inherent in the outstation communication paths, can be isolated from communications with the SCADA system, thereby increasing overall system performance.

One particular advantage of Mimic Mode is that the master SCADAPack does not need to know, or be configured with, any details of the DNP points configured in the outstations. This makes it relatively simple to insert such a SCADAPack master into any existing DNP network. The SCADAPack master in Mimic Mode behaves transparently to the higher-level SCADA system, and can easily be configured with communication paths and polling instructions for each connected outstation.

Note: This feature is limited to the SCADAPack 32, SCADAPack 330/334, SCADAPack 350 and SCADASense 4203 controllers.

8.3.1.2 SCADAPack DNP Address Mapping

Address mapping provides a direct link between an outstation's DNP points and local Modbus registers within the SCADAPack DNP master. These remote DNP points are now mapped into specific regions of the DNP master's Modbus database.

DNP Master Address Mapping		DNP Outstation
Local Modbus Register		DNP Point
11001	→	1
11002	→	2

When DNP data points are received from an outstation, a cross reference to the address mapping table is made, and if a match is found, the DNP data will be written to the corresponding local Modbus register. 'Input' DNP object types from the outstation are mapped to the master's local input Modbus register space 1xxxx or 3xxxx. These local Modbus registers are updated after the corresponding DNP point gets updated; usually by a class poll to the outstation, or if the outstation issues an unsolicited response based on a change of value or state on these points.

'Output' DNP object types from the outstation are mapped to the master's local output Modbus register space 0xxxx or 4xxxx. Changes made to the local Modbus register will trigger a DNP Write message, with the current point value, to the outstation. DNP Write implemented in SCADAPack controllers requires an Application Layer confirmation from the target outstation.

By configuring the Address Mapping table, outstation DNP points are therefore mapped to local Modbus registers. As mapped local Modbus points, the data is available for use in application programs such as TelePACE and ISaGRAF. In addition a Modbus SCADA Host can poll the SCADAPack master for these points.

The following diagram shows a simple DNP Address Mapping network.

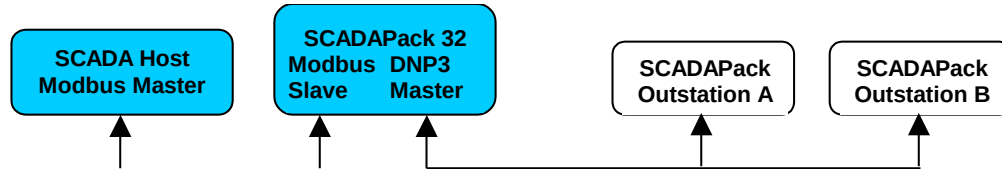


Figure 8-5: SCADAPack Address Mapping

In this network the SCADAPack master updates its local database with mapped outstation data. The manner and frequency with which the SCADAPack master updates the local Modbus registers, depends on the number and type of I/O object types the registers are mapped to.

This feature is limited to the SCADAPack 32, SCADAPack 330/334, SCADAPack 350 and SCADASense 4203 controllers.

Note: Mapping numerous local Modbus output registers (0xxxx and 4xxxx), to a remote DNP device may cause frequent communications between the master and the slave, if the associated registers are being changed frequently in the master. On limited bandwidth or radio networks, care must be taken to ensure that your network capacity can handle all the traffic that will be generated from these local changes.

8.3.2 How to Configure SCADAPack DNP Master

In this exercise, we will configure a SCADAPack DNP Master to poll a DNP outstation with address 10. The DNP master will be communicating to the outstation by requesting for Class event data and acknowledging receipt of unsolicited responses through com1.

After this exercise, you should be able to:

- Configure a DNP Master to poll for Static (Class 0) and Class 1 event data.
- Configure a DNP Master to accept and respond to unsolicited messages

8.3.2.1 Tasks to Complete

1. Enable DNP communication on com1 of the SCADAPack controller.
2. Configure a DNP Master with station address of 200, for example.
3. Configure the DNP master to issue class polls to the outstation created in the previous exercise.
4. Map outstation DNP points to local DNP points.

8.3.2.2 Configuration Steps

1. Use the same procedure of the previous exercise to enable the DNP protocol on com1.
2. From the **Controller** menu, launch the DNP Configuration panel.
3. From the Application Layer configuration panel,
 - a. Ensure that the **Application Layer Confirmation** is Disabled.

TIP: A master should not have to request for an Application Layer Confirmation, as an Application Layer response is implied in all master requests.

- b. Set the **Application Timeout** to 3000 seconds.
- c. Set **Time Synchronization** to none.

TIP: Master time synchronization to an outstation is configured in the Add/Edit Master Poll dialog.

- d. All other parameters can be left at their default values. The completed Application Layer Configuration panel should look like this:

DNP Settings

Application Layer

Communication

Application Confirmation: Disabled

Maximum Fragment Length: 2048

Retries: 0

Application Timeout: 3000 msec

Time Synchronization

☒ None

☐ At Start Up Only

☐ Interval: 60 minutes

Unsolicited Responses

	Class 1	Class 2	Class 3
Enable Unsolicited	Disabled	Disabled	Disabled
Hold Time	60	60	60
Hold Count	10	10	10
Enable Unsolicited Responses on Start Up	Yes		
Send Initial Unsolicited Response on Start Up	No		

Dial Up

Modem Initialization: &F0 S0=1 &W0 &Y0

Connect Timeout: 45 seconds

Attempts: 2

Inactivity Timeout: 45 seconds

Dial Type: Tone

Pause Time: 10 seconds

Operate Timeout: 15 seconds

Report only Level 2 Objects in Class Polls: No

☐ Limit Maximum Events in Read Response: 0

☐ Allow Duplicate Modbus Addresses

OK, Cancel, Add, Copy, Delete, Move Up, Move Down, Help

4. From the DNP Configuration dialog, click on the **Data Link Layer** tree node.
 - a. Leave the **Master Station Address** at the default value of 100.
 - b. Change the **RTU Station Address** to 200.

DNP Settings

Data Link Layer

Master Station Addresses: 100

OK, Cancel, Add, Edit, Delete

5. Click on the **Master** tree node from the DNP Settings dialog.
 - a. Set the **Base Poll Interval** to 1s.
 - b. Ensure **Mimic Mode** is Disabled.

TIP: A small Base Poll interval provides better granularity.

6. Click on the **Master Poll** tree node from the DNP Configuration panel.
 - a. Set the **Base Poll Interval** to 1s.
 - b. Ensure Mimic Mode is Disabled.
 - c. Click on the Add button within the Master Poll panel to create a new master poll schedule.
 - d. In the Add/Edit Master Poll dialog, do the following:
 - i. Set **Station** to 10
 - ii. Under **Class 0 Polling** group box, set the **Interval** to 3600 base poll intervals (1 hour).
 - iii. Leave the **Poll Offset** at the default of 0 base poll intervals.

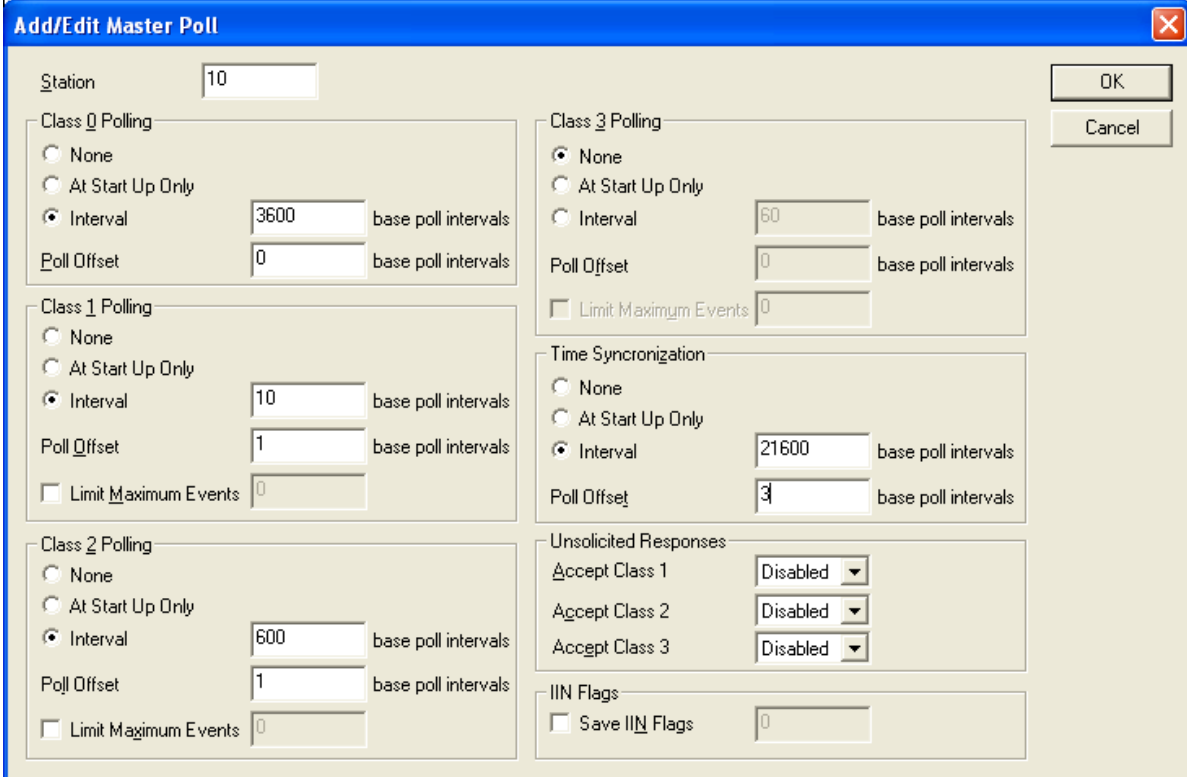
TIP: Static (Class 0) comprise current values of all DNP3 points in the I/O database. Due to the sheer size of this data, it is recommended to reduce the frequency of static polls. Urgent data will be updated at the master via Class polls or unsolicited messages.

- iv. Under **Class 1 Polling** group box, set the **Interval** to 10 base poll intervals (10 seconds).
- v. Set the **Poll Offset** to 1 base poll intervals.
- vi. Leave the **Limit Maximum Events** checkbox unchecked.
- vii. Under **Class 2 Polling** group box, set the **Interval** to 600 base poll intervals (10 minutes).
- viii. Set the **Poll Offset** to 2 base poll intervals.
- ix. Leave the **Limit Maximum Events** checkbox unchecked.
- x. Under the **Time Synchronization** group box, set the **Interval** to 21600 base poll intervals (6 hours).
- xi. Set the **Poll Offset** to 3 base poll intervals.

TIP: Polling intervals on Master request for time synchronization are configured in this dialog. If possible, set this to a daily frequency.

A small base poll interval limits that maximum poll interval to 32767 seconds. Daily polls (every 86400 seconds) are, therefore, not possible when the base poll interval is set for 1 second. .

- xii. Under **Unsolicited Response** group box, leave all fields at default. The completed dialog should look like this:



Add/Edit Master Poll

Station: 10

Class 0 Polling

☐ None

☐ At Start Up Only

☒ Interval: 3600 base poll intervals

Poll Offset: 0 base poll intervals

Class 1 Polling

☐ None

☐ At Start Up Only

☒ Interval: 10 base poll intervals

Poll Offset: 1 base poll intervals

☐ Limit Maximum Events: 0

Class 2 Polling

☐ None

☐ At Start Up Only

☒ Interval: 600 base poll intervals

Poll Offset: 1 base poll intervals

☐ Limit Maximum Events: 0

Class 3 Polling

☒ None

☐ At Start Up Only

☐ Interval: 60 base poll intervals

Poll Offset: 0 base poll intervals

☐ Limit Maximum Events: 0

Time Synchronization

☐ None

☐ At Start Up Only

☒ Interval: 21600 base poll intervals

Poll Offset: 3 base poll intervals

Unsolicited Responses

Accept Class 1: Disabled

Accept Class 2: Disabled

Accept Class 3: Disabled

IIN Flags

☐ Save IIN Flags: 0

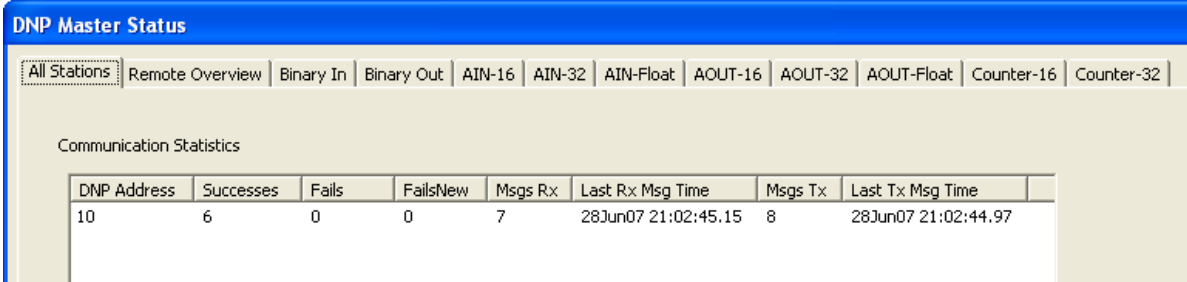
OK Cancel

7. From the DNP Configuration panel, select the **Routing** tree node.
 - c. Click on the **Add** button to begin a new routing table entry.
 - d. From the Add/Edit Route dialog,
 - i. Enter 10 for the destination **Station**.
 - ii. Set the **Port** to COM1.
 - iii. Leave default values for all other parameters.

8.3.2.3 Confirm Successful DNP Master Configuration

With this configuration and a valid communication link between com1 of the DNP Master and com2 of the DNP outstation, you can use the DNP Master Status dialog to see communication activity between the two devices.

Confirm that you have communication activity between the master and outstation as indicated in the screen capture below.



DNP Master Status

All Stations Remote Overview Binary In Binary Out AIN-16 AIN-32 AIN-Float AOUT-16 AOUT-32 AOUT-Float Counter-16 Counter-32

Communication Statistics

DNP Address	Successes	Fails	FailsNew	Msgs Rx	Last Rx Msg Time	Msgs Tx	Last Tx Msg Time
10	6	0	0	7	28Jun07 21:02:45.15	8	28Jun07 21:02:44.97

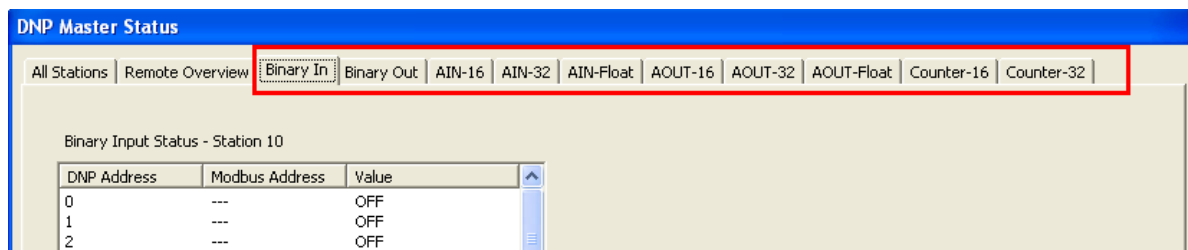
If the All Stations tab indicates successful message transmission between the Master and Outstation, congratulate yourself on completing the exercise.

For additional information on the aforementioned configuration parameters, referenced in the previous two exercises, refer to Chapter 8.6 in this manual.

8.3.3 How to Configure SCADAPack Address Mapping

At this stage in your configuration, the DNP master is able to poll for all outstation points. After a successful poll, you can verify the status of current value of outstation DNP points from the various data point type tabs available across the DNP Status Window.

The figure below shows the status of DNP digital input points 0, 1 and 2 on outstation 10.



The Modbus Address column is blank as these remote DNP points have not been mapped to any local Modbus registers.

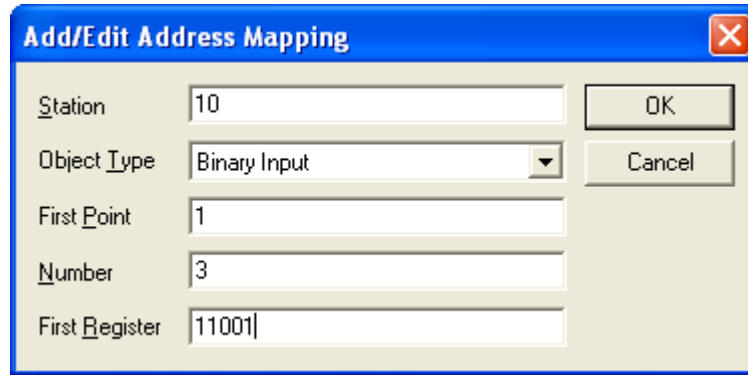
While this data is available in the DNP Address space of the master, it is not available for use within a local program. To render DNP data available to a local program, you would have to perform an Address Map. To map DNP binary input data from outstation 10 to this master's local DNP database, do the following:

From the **Controller** menu, select **DNP Configuration**.

1. Click on the **Address Map** tree node.
2. From the **Address Mapping** configuration panel
 - a. Click on the **Add** button to launch the **Add/Edit Address Mapping** dialog.
 - b. Enter 10 for Station.
 - c. Select **Binary Input** for Object Type.
 - d. Enter a value of 1 for First Point. This is the DNP Address of the first Binary Input point in Station 10.
 - e. Enter 3 for Number of points to map.
 - f. Enter 11001 for First Register (First Modbus Register) address. Note that Modbus address 11000 must exist in your controller database.

CAUTION: In a practical setting, a DNP Master may have local I/O mapped also mapped to points within its DNP database. Ensure that you are not mapping outstation DNP points to local address being used by local I/O.

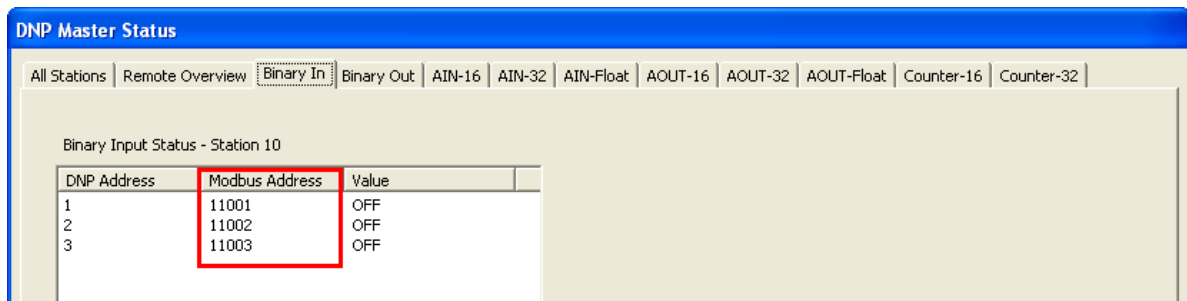
The completed dialog should look like this:



The dialog box 'Add/Edit Address Mapping' contains the following fields and buttons:

- Station:** 10
- Object Type:** Binary Input (dropdown menu)
- First Point:** 1
- Number:** 3
- First Register:** 11001
- Buttons:** OK, Cancel

- g. Click on **OK** to add this entry to the Address Mapping table.
- h. Use the DNP Master Status dialog, to confirm that remote points are being mapped to local Modbus registers as shown below:



The 'DNP Master Status' dialog box shows the 'Binary In' tab selected. The title is 'Binary Input Status - Station 10'. Below the title is a table with three columns: DNP Address, Modbus Address, and Value.

DNP Address	Modbus Address	Value
1	11001	OFF
2	11002	OFF
3	11003	OFF

The 'Modbus Address' column is highlighted with a red box.

You can also confirm that remote points are being mapped to Modbus registers by monitoring the status of Modbus registers 11001, 11002 and 11003.

8.3.4 How to Configure SCADAPack DNP Mimic Master

In addition to all the configuration procedures for a DNP Master, following the steps below to enable the DNP Mimic master.

From the **Controller** menu, select DNP Configuration

1. Follow all steps in the [Section 8.3.2.2](#) to configure the DNP master.
2. Click on the **Master** tree node.
3. Enable **Mimic Mode**.

8.4 SCADAPack DNP Router

All SCADAPack controllers can be configured as a DNP Router. A unique characteristic of a SCADAPack DNP router is the ability to:

- Route (or forward) DNP messages not destined to this station, using rules defined within a routing table.

Otherwise, a SCADAPack controller not configured for DNP routing will simply discard a message whose DNP destination address does not match that of the controller.

A DNP router is typically used when a direct communication link between the DNP master and outstation cannot be established, typically due to different physical layers on the two network segments. For instance, the physical network between the DNP SCADA Host and the router could be an Ethernet connection, while the physical layer between the router and all outstations could be a

multi-drop serial RS-485 or even an RS-232 radio connection. Given that messages are routed directly from the DNP SCADA Host to the outstations, bandwidth limitations are dictated by the speed of the serial multi-drop connection. On the contrary, there is no bandwidth limitation within a DNP Mimic architecture, as the Mimic Master immediately responds to the DNP SCADA Host on behalf of the targeted outstation. Of course, the side effect of the DNP Mimic architecture is that polled data obtained by the DNP SCADA Host may not be very current. In either case, careful design considerations based on these tradeoffs should be exercised.

As mentioned above, the SCADA Host has only one connection to a SCADAPack DNP Router. All target outstations of the SCADA Host are connected down stream of the DNP Router as illustrated in the figure below.

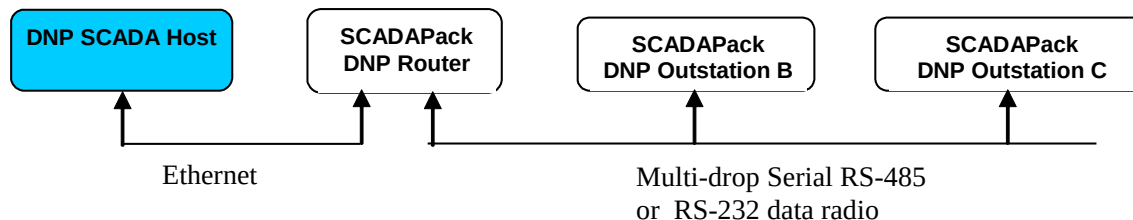


Figure 8-6: SCADAPack DNP Router and multi-dropped DNP Outstations

In the above configuration the SCADAPack DNP Router (Outstation A above) manages all the communication with the outstations. The SCADAPack DNP router receives messages from the SCADA Host for each outstation and **route** or forwards the messages to the outstations, based on routing rules established with the DNP Routing table.

DNP Messages are routed based on the following logic:

```

if (a message is received which needs to be retransmitted to someone else)
    if (the message target is configured in our routing table)
        if (the destination port is different from the incoming port)
            or (routing is enabled on the incoming port)
                then retransmit the message
  
```

Change event data in the form of unsolicited responses from the outstations are routed directly to the DNP SCADA Host, by the SCADAPack DNP router.

A DNP Router is different from a Mimic in that a router forwards all messages are directly to the outstations, whereas the mimic responds to some messages on behalf of the outstations. Therefore, both operation modes have the advantage of delegating the task of DNP Routing of multiple outstations to this intermediate unit. The SCADAPack DNP router handles all communications paths to outstations, including such tasks as dial-up radio communication. In contrast to Mimic mode, however, the SCADA Host system still has to handle the long delays and high error rates that may be present on the communications links to the outstations.

Note: Mimic Master and Routing are incompatible modes that should never be used together.

8.4.1 How to Configure a SCADAPack DNP Router

In this exercise, we will configure a SCADAPack 32 controller to route DNP messages received from DNP Master 32001 on its Ethernet port, out through com2. This message is destined for outstation 20. This exercise assumes a valid Ethernet connection between your PC or laptop and the SCADAPack 32.

After this exercise, you should be able to:

- Configure DNP/TCP on an Ethernet port
- Configure a SCADAPack DNP Router to route messages from a SCADA DNP Host to an outstation.

8.4.1.1 Tasks to Complete

1. Enable the DNP protocol communication on the communication interfaces involved in routing.
2. Enable routing on the communication interface.
3. Setup the forward and return entries in the DNP routing table.

8.4.1.2 Configuration Steps

1. From the Controller menu, click on **Serial Ports**.
2. In the Controller Serial Ports dialog, set the Protocol on COM2 to DNP.
3. In the Controller Serial Ports dialog, Enable Routing.
4. The completed dialog should look like this:

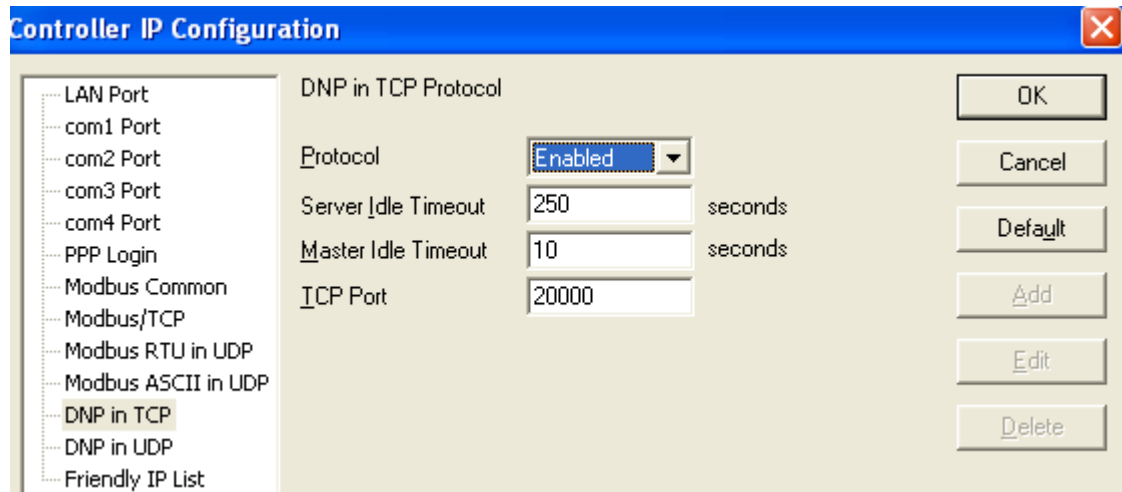
Controller Serial Ports - MASTER

Port	COM2	OK Cancel Default
Protocol	DNP	
Addressing	Standard	
Station	1	
Duplex	Full	
Baud Rate	9600	
Data Bits	8 Bits	
Parity	None	
Stop Bits	1 Bit	
Rx Flow	None	
Tx Flow	None	
Port Type	RS-232	
Routing	Enabled	
Enron Modbus	Disabled	
Enron Station	1	

5. Click on OK to close this dialog and save your settings.
6. From the Controller menu, click on IP Configuration.
7. Select the DNP/TCP tree node from the Controller IP Configuration dialog.
8. Enable the protocol and leave all other settings at default values.

Note: This exercise assumes that you have a valid IP Address, Subnet Mask and Default Gateway properly configured.

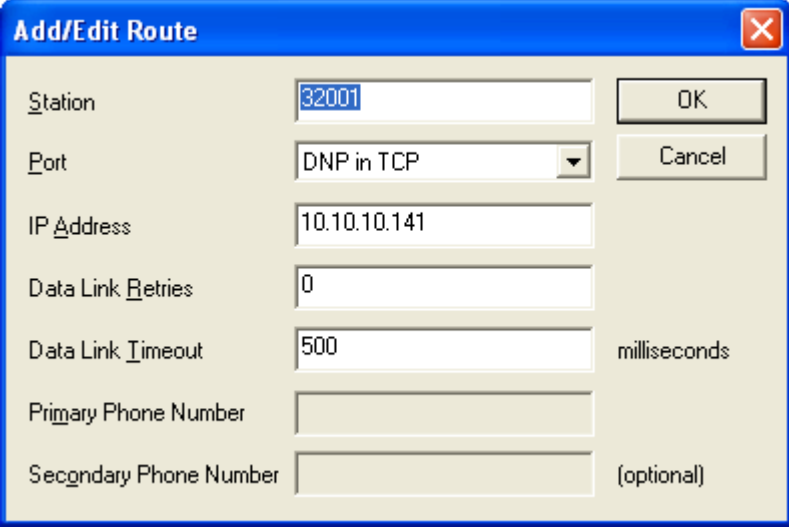
9. The completed dialog would look like this:



TIP: In this configuration, the SCADAPack DNP Router is acting as a DNP Server on the Ethernet port. The **Server Idle Timeout** parameter will be used to determine how long this connection will be kept open from time of last communication activity. For a **Server Idle Timeout** default value of 4 minutes, and an **Application Layer Timeout** default value of 5 minutes, there is the possibility that the IP port will be closed, if the router is experiencing communication problems with the outstations. In this case, it is a good idea to increase the Server Idle Timeout to at least 2x the DNP configuration Application Layer Timeout. Or, simply reduce the Application Layer timeout to a value less than 2x the Server Idle Timeout.

10. From the DNP Configuration panel, select the Routing tree node.
 - e. Click on the **Add** button to begin a new routing table entry.
 - f. From the Add/Edit Route dialog,
 - Add the route to Station 20:
 - i. Enter 20 for the **Station**.
 - ii. Set the **Port** to COM2.
 - iii. Leave default values for all other parameters.
 - iv. Click on **OK** to add this entry to the routing table and return to the Routing dialog.
 - Add the **Return** route from Station 20:
 - i. Enter 32001 for the **Station**.

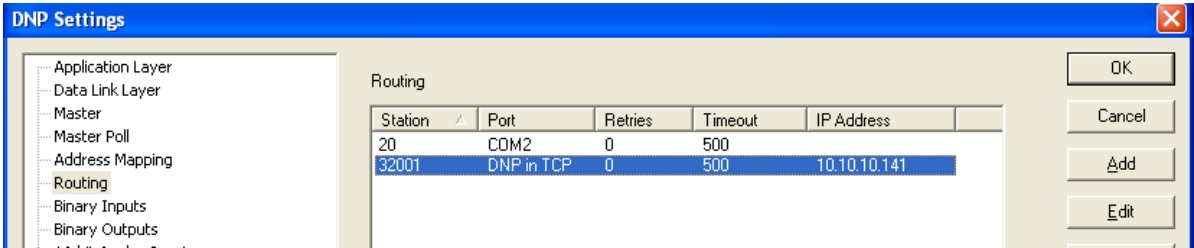
- ii. Set the **Port** to DNP in TCP.
- iii. Enter the IP Address of your DNP Master. In this case, the IP Address of my PC running a DNP SCADA Host software is 10.10.10.141.
- iv. Leave default values for all other parameters.
- v. The completed dialog should look like this:



The 'Add/Edit Route' dialog box contains the following fields and controls:

- Station:** Text box containing '32001'.
- Port:** Dropdown menu set to 'DNP in TCP'.
- IP Address:** Text box containing '10.10.10.141'.
- Data Link Retries:** Text box containing '0'.
- Data Link Timeout:** Text box containing '500', with 'milliseconds' indicated to the right.
- Primary Phone Number:** Empty text box.
- Secondary Phone Number:** Empty text box, with '(optional)' indicated to the right.
- Buttons:** 'OK' and 'Cancel' buttons on the right side.

- vi. Click on **OK** to add this entry to the routing table and return to the Routing dialog.
- vii. The completed routing table should look like this:



The 'DNP Settings' window shows the 'Routing' tab. On the left is a tree view with 'Routing' selected. The main area displays a routing table:

Station	Port	Retries	Timeout	IP Address
20	COM2	0	500	
32001	DNP in TCP	0	500	10.10.10.141

Buttons on the right include 'OK', 'Cancel', 'Add', and 'Edit'.

Note: For proper operation of the router, there must be two routing entries in the routing table for each outstation; An entry specifying how the communication path from this router to the outstation and another communication path from the router to the SCADA DNP Master.

8.5 Design Considerations

The strength of DNP lies in its ability to offer time-stamped data, scheduled polling of data from multiple outstations and time synchronization, event data buffering and reporting by exception.

DNP was originally design to be used over a serial point-to-point (RS-232) link. As such, the protocol implements certain measures against data corruption and data loss in its Application and Data Link layers. Such measures include timeouts, retries, and checksums.

These data recovery mechanisms provided by the protocol, can be counter productive when not properly configured over an underlying communication medium, such as Ethernet, that already provides robust measures. In almost all of such cases, the recovery mechanisms offered by DNP need to be turned off. Such considerations together with good engineering judgment, therefore, must be practiced before one embarks on the design of a large DNP network.

This chapter outlines special considerations of the DNP protocol and implications within the SCADAPack DNP driver that should be considered when designing large networks. We also list common malpractices and a list of Frequently Asked Questions (FAQs) that arise during the course of network design.

8.5.1 Considerations of DNP3 Protocol and SCADAPack DNP Driver

To ensure consistent network performance, even under worse case scenarios, the following DNP specification rules should be considered when designing a DNP network using SCADAPack as the main nodes.

8.5.1.1 Unsolicited Messages always request for a Confirmation

An outstation will always request for an Application Layer confirmation when it sends an unsolicited message, even if the Application Layer confirmation field is not enabled. If no response is received within an Application Layer timeout, the outstation will retry the message a number of times as determined by the Application Layer Retry parameter.

8.5.1.2 Master shall never request for Application Layer Confirmation

A Master request is always accompanied by a response message from an outstation. Hence, the Application Layer confirmation on the master RTU should never be enabled.

8.5.1.3 DNP Write Messages always request for a Confirmation

As implemented in the SCADAPack DNP driver, a DNP Write request (FC 02) requires an Application Layer response from the outstation. If an acknowledgement is not received within the configured Application Layer timeout interval, the message is retried a number of times as determined by the Application Layer retry parameter.

8.5.1.4 Only one DNP3 transaction can be pending at a time

A SCADAPack DNP station will not initiate or process another DNP transaction, as long as one is outstanding. Thus, once a SCADAPack has initiated a DNP transaction, all subsequent DNP3 messages received but not related to the original transaction are buffered.

8.5.1.5 SCADAPack controllers buffer 3 DNP messages

A SCADAPack serial port receive buffer can hold a maximum of 3 DNP messages or Data Link frames. If an additional DNP message is received when the buffer is full, the oldest message in the buffer is replaced with the newest one.

8.5.1.6 Output points in DNP Address Mapping issue DNP Write

Digital and analog output points contained within the DNP Address Mapping of a SCADAPack controller automatically issue DNP Write messages when their value or state changes.

8.5.2 Typical Configuration Malpractices and Recommendations

DNP is a capable protocol that effectively transfers some of the system engineering effort from designing a sophisticated logic program, to configuring and tuning the system using parameters. However, DNP does not eliminate the need to properly evaluate and engineer the communication media to support the performance expectations of the system, especially under worse case scenarios.

DNP networks can be designed around polling or report-by-exception. In a polling environment, each master request can be viewed as an invitation for an outstation device to transmit data on the shared communications medium. The master controls which device can transmit, thereby preventing collisions from occurring, as the timing of responses is predictable under all situations. In addition, masters can ask again if a response is not received, thus providing an opportunity for the outstation to re-send lost data. Using this strategy, a master effectively manages media access thereby preventing contention with those outstations unexpectedly transmitting on their own.

DNP networks can also be designed around unsolicited communications. In this case, the outstations transmit events to the master as they occur. When using this strategy, the communications media must be evaluated carefully in regards to the need for collision detection and prevention, if consistent network performance is to be expected.

Given that typical systems are designed using a combination of both strategies, is a good idea to start by configuring the network for poll mode, as it can be easily tuned to cater for unsolicited messaging, when system characteristics under worst case conditions become known. As with any communications system, the designer should pay careful attention to bandwidth allocation and management for a successful system implementation.

Below are several requirements of DNP system architecture that require careful engineering judgment.

1. Multiple high priority unsolicited messages configured in outstation.
2. System with multiple outstations, each containing numerous Class 1 events, configured with a Class 1 Hold Count of 1.
3. Relying on unsolicited messaging to get event data to master. System not designed around master polling for events.
4. Multiple masters with poor communication link.
5. Insufficient use of Deadband and debounce to curb event generation.
6. Master RTU has Application Layer confirmation enabled.
7. Enabling both the Application and Data Link Layer confirmations.
8. Setting very high Application Layer timeout values over high speed networks.
9. DNP Address mapping contains multiple analog and digital output points that change rapidly.

The aforementioned statements and recommendations are provided below. Note that these recommendations are to ensure, consistent performance under worse case situations, and are based on the special considerations provided in the previous section.

8.5.2.1 Multiple High Priority Unsolicited Messages

A common configuration malpractice is to enable numerous high priority events objects within an outstation, and configure the outstation to trigger an unsolicited message to the master each time a new event occurs. In a SCADAPack controller, this is accomplished by configuring numerous Class 1 event objects, and enabling Class 1 Unsolicited Responses (Messaging) with a Hold Count or Hold Time of 1.

A Hold Count of 1 and Hold Time of 60 seconds specified for Class 1 events, imply that the controller will immediately trigger an unsolicited event as one occurs. If this outstation and others have a multitude of Class 1 event objects, visualize worse case scenario as a burst of messages being transmitted to the master at the same time. Given that a SCADAPack serial port buffer can only handle three DNP Data Link frames at any given time, some messages might get lost, especially if the master is required to immediately retransmit this message to some other node in return.

Such a system is designed around unsolicited messaging and is, therefore, far more susceptible to network collisions if proper management of bandwidth is not exercised. Given that a SCADAPack controller can only process one DNP transaction at a time, there is also a good chance that the serial port receive buffer will overflow, adding to the cost of lost messages.

Recommendations:

In general, bandwidth is used more efficiently in a large DNP system if the master is designed to poll for event data more frequently and static data less regularly.

Recommended practice is also to reserve unsolicited messaging for a small number of critical data. If possible, it may be best to ensure that no more than 3 messages are sent to the master at exactly the same time, under worse case scenario, as some event data may be lost if the master is currently busy processing another transaction, unless random retry intervals are put in place.

If unsolicited messaging is the predominant data transfer method, an approach to manage network usage, could be to configure a group of three or less outstations with a Hold Time that is unique within the group.

The table below shows an example configuration for Hold Time and Hold Counts for Class 1 events across six outstations.

Table 8-1: Hold Time and Hold Count Setup in for Six DNP Outstations

DNP Outstation Address	Hold Time (seconds)	Hold Count
11	1	100
12	1	100
13	1	100
14	2	100
15	2	100
16	2	100

8.5.2.2 Master not polling frequently causing event buffer overflows

An outstation does not discard the events within its buffer until all its configured masters have acknowledged receipt of these events. This means that an outstation event buffer may eventually fill

up and overflow leading to loss of events. Buffer overflows typically indicate a poorly configured system.

When the system is designed around unsolicited messaging, there is a good likelihood of media contention causing buffer overflows. On the contrary, if the system is designed around frequent master poll for event data, there will be fewer chances of buffers overflowing causing loss of event data.

As stated earlier, immediate reporting of events using unsolicited messaging should be reserved for those critical, yet absolutely rare occurring events. This is because unsoliciting these messages back to the master will be reliable only if there is a substantial amount of unused bandwidth on the communication media. A good rule of thumb is to have 50% or more of unused bandwidth available, evenly distributed over a time frame.

Recommendation: Design the system around frequent master poll of class events and less regular integrity polls. Reserve unsolicited messaging for infrequent high priority events. If network traffic is predominated by unsolicited messaging, allocate 50% or more unused bandwidth as quiet time.

8.5.2.3 Outstation reports to Multiple Masters with Poor Comms

A poor communication link to one of an outstation's multiple masters will prevent the outstation's event buffer to be emptied, as events cannot be reported to the master. This could lead to buffer overflow situations and loss of event data.

Recommendation: Ensure that the communication path to all masters of an outstation is robust.

8.5.2.4 Insufficient Use of Input Deadband or Debounce

Event generation on a DNP analog input is controlled by a Deadband parameter. On a digital input, event generation is controlled by a debounce parameter. Default settings of zero for these parameters are typically overly aggressive and may lead to events being generated due to noise.

Recommendation: Set the analog Deadband and debounce parameters appropriately to non-zero values.

8.5.2.5 Master Confirmation and Retries

Application or Data Link Layer confirmations should never be enabled on a master as:

1. Master requests typically will fit within a single Application Layer fragment hence there is need for Data Link Layer confirmations.
2. Master request typically require a response, hence no need for Application Layer confirmations.

Thus, enabling the Application Layer Confirmation on a DNP master is obsolete practice and may instead degrade system performance.

Recommendation: Disable the Application Layer Confirmation in a master SCADAPack controller. Typical retry values for Application Layer retries lie between 1 and 3. Lengthy retries may instead burden the communication medium

8.5.2.6 Outstation Confirmations and Retries

Confirmations on an outstation serve two useful purposes:

1. Ensure that a master received unsolicited responses from the outstation.
2. To ensure that a master correctly received responses to its request

Unsolicited messages will always request for an Application Layer confirmation, whether or not the Application Layer Confirmation is enabled on the outstation. If network traffic is predominantly unsolicited messaging, the Application Layer confirmation does not need to be enabled.

When the master is configured, as recommended, to frequently poll the outstation for event data using **read request**, while imposing a limit on the number of events the outstation should include in its response, the outstation still needs to know if the master received its replies so that it can:

- Remove these events from its buffer
- Know what to transmit next.

To cater for confirmations to read responses, Application Layer Confirmation in the outstation typically needs to be enabled.

The Data Link Layer breaks down Application Layer fragments into smaller frames. Smaller packet sizes reduce bit error in noisy environments. While it is better to accept the overhead of confirming each Data Link Layer frame of a multi-frame message, and re-transmit corrupted frames, than to re-send an entire Application Layer fragment, a viable alternative is to reduce the Application Layer fragment size and use only Application Layer confirmations. When fragments are reduced to the size of a Data Link Layer frame, the overhead of Application Layer confirmations, and the probability of noise corrupting those confirmation messages, is nearly the same as for Data Link Layer confirmations.

Enabling the Data Link layer confirmation on the outstation, therefore, is not required when the communication medium is not reliable. For example, certain data radios, e.g. FreeWave 9000 MHz spread spectrum radios, implement a robust mechanism to ensure that a data packet make it to their desired destination; TCP/IP incorporates robust mechanisms to prevent data loss; a local serial link between stations is also very reliable. In these cases, it is not necessary to enable the Data Link Layer confirmations.

If, however, physical medium quality is below par, such as in the case of noisy radio networks, or a shaky PSTN connections, then one should enable the Data Link Layer confirmation only, or as mentioned earlier, reduce the Application Layer maximum fragment length below 249 bytes.

If either the Application or Data Link Layer Confirmation is enabled, retries should be configured to a low non-zero value. Typical retry values lie between 1 and 3. Lengthy retries may instead burden the communication medium

Recommendation:

Application and Data Link Layer confirmations in an outstation can be set according to the following table:		Communication Medium Reliability	
		High	Low
Data Acquisition Configuration	master polls outstation frequently for event data (also limits number of events in read response)	Enable Application Layer Confirmation Disable Data Link Layer Confirmation	Disable Application Layer Confirmation Enable Data Link Layer Confirmation
	master does not poll frequently enough and outstation generates lot of unsolicited messages	Enable Application Layer Confirmation Disable Data Link Layer Confirmation	Disable Application Layer Confirmation Enable Data Link Layer Confirmation
	Regardless of the data acquisition strategy, if the	Enable Application Layer	Enable Application

	Max Application Layer fragment is set to a values less than 249	Confirmation Disable Data Link Layer Confirmation	Layer Confirmation Disable Data Link Layer Confirmation

Note: It is never required to enable BOTH the Application and Data Link Layer Confirmations.

8.5.2.7 Setting relatively large Application Layer timeouts

On a high speed link, such as Ethernet, configuring a high Application Layer timeout does not increase network reliability. Instead this reduces system performance, as there will be a significant portion of time within the timeout period, after which the IP transaction may have been terminated.

Typically, an Ethernet transaction is completed in the order of a millisecond and a DNP master SCADAPack controller, by default, closes its DNP TCP port within 10 seconds of no activity. A DNP SCADAPack controller acting as an outstation closes its port by default in about 4 minutes.

Under these default conditions, if the application layer timeout on a SCADAPack DNP master is set for 15 seconds, for instance, the port may have closed 10 seconds after last activity, but the application may still be waiting for a timeout.

If a message is somehow lost, and the timeout is set for 5 seconds, for instance, the application will still be waiting for a response even though the IP transaction has terminated. This results to wasted bandwidth.

Recommendation: When operating over high speed links, make Application Layer timeouts as small as possible.

8.5.2.8 DNP Address mapping contains multiple output points

The DNP Address Mapping table allows local Modbus registers in the SCADAPack DNP master to be mapped to DNP points in an outstation. Each time an output register defined within the DNP Address Mapping table changes, a DNP Write message (FC 2) is immediately issued to update the corresponding DNP point in the outstation.

If numerous output registers that change frequently, are listed in the Address Mapping table, the network will be overburdened with a multitude of DNP Write messages.

Recommendation: Reserve Address Mapping only for mapping of outstation DNP data that needs to be used by the master Modbus database, or to segregate points from different outstations in the master. If numerous points are being mapped from the outstation to the master, the system is not designed properly. In this case, it may be worthwhile to consider transferring application logic from the master to the outstation.

8.5.3 Configuration FAQ

Complimentary commonly asked questions and answers are given below.

When configuring a routing entry in the DNP Routing table of a SCADAPack, one has to specify the Data Link Layer Timeout and Retries. Do these fields take precedence over the same fields found under the Data Link Layer configuration panel?

Yes.

In a master-outstation architecture, how do you recommend we setup time synchronization?

Recommended practice is to configure the master to initiate time synchronization to the outstations.

DNP3 provides 4 data classes; Class 0 (Static or None), Class 1, Class 2 and Class 3. How does I decide which class to assign any given I/O point?

In a SCADAPack controller, all configured DNP points by nature, are members of the Class 0 type. Class 0 data is the current value or state of a DNP point. So, when a master does a Class 0 poll to an outstation, the most current value or state of all DNP points within the database are returned.

Value or state changes on a point are captured as Class 1, Class 2 or Class 3 event data. Typically, highest priority events are assigned to Class 1 and the lowest priority event to Class 3.

What does Class of 'None' mean?

Class None is Class 0 or Static.

Why does this setting do: Enable Unsolicited Responses On Startup?

This setting enables unsolicited response (or unsolicited message) transmission, when power to an RTU is cycled or when its configuration is changed. In this case, the RTU does not have to wait for Function Code 20 or 0x14 (Enable Unsolicited Responses) from the master before it starts sending any collected events.

This field should be set to No, to allow a master control when an outstation is able to send unsolicited messages. Recommended practice is to allow a master to enable unsolicited message transmission on all outstations.

Why would I ever need to change the Application Layer Maximum Fragment Length?

The Application Layer Maximum Fragment Length determines the maximum amount of memory that is reserved for each application layer fragment. The default is 2048 bytes on SCADAPack controllers although outstations must be prepared to receive fragments sizes of at least 249 bytes. When communicating with those devices with insufficient memory it is necessary to limit the maximum application layer fragment length to what the outstation can handle. In addition, limiting the application layer fragment size beyond 249 also reduces the maximum Data Link layer frame length.

Certain data radios may give better efficiency when transmitting data packets less than the maximum data link fragment size of 249 bytes. With these radios, it is necessary to reduce the application layer's maximum fragment size below 249 bytes as required by the radio.

Other types of data radios, such as FreeWave's 900 MHz Spread Spectrum radios, provide configuration options to optimize efficiency by changing the maximum packet size. In this case, it is not necessary to reduce the application layer maximum fragment size.

In addition, when the communication medium is noisy, it is typically more efficient to transmit smaller packets than larger packets. In this case, setting small Application Layer fragments would force smaller data link frames, which is a better strategy in a noisy environment.

Caution: Limiting the application layer fragment size reduces the rate at which event data is retrieved from the buffers, thus increasing the possibility of event buffer overflows, if the event data is not being retrieved in a timely fashion. Reducing the maximum application layer fragment size, increases network traffic and also reduced data throughput as an Application Layer Confirmation is required for each fragment of a multi-fragment message.

Why would I ever want to 'Limit that maximum number of events in a read response'?

This is another strategy that can be used to limit the Application Layer fragment of an outstations' response message. This strategy could be used under noisy environments.

Also, this could be used to prevent an outstation with a large collection of event data to hold the communication media captive while transmitting all its events.

What behavior should we expect from a SCADAPack when the event logs are full?

When a new event is collected and the SCADAPack DNP event buffer is full, the oldest event is deleted and the newest event added into the buffer.

What is the main difference between SCADAPack DNP driver configuration modes?

	DNP Master	DNP Mimic Master	Address Mapping	Router	Outstation
Define DNP I/O points	Not necessary	Not necessary	Not necessary	Not necessary	Yes
Enable Application Layer Confirmation	No	No	No	No.	Not necessary.
Should I Initiate Time synchronization?	Yes	No.	No.	No.	No.
Poll for Class DNP static and Class data?	Yes	Yes	Yes	No.	No
Initiates Unsolicited messages?	No	No	No	No	Yes
Router Messages not destined to this station?	No	Some	No	Yes	No
When best to use	Master in a Point to Multipoint network	Data Concentrator with many outstations that will take a while to configure. When outstation data does not need to be available to logic in this node.	When remote DNP data is needed by local program	Strictly Repeater	Forms basic node in DNP network.

8.6 DNP Configuration Menu Reference

This section of the manual details the SCADAPack DNP3 driver configuration parameters. The DNP Configuration panel is accessed from the **Controller | DNP Configuration** menu from either TelePACE or ISaGRAF. Browse through this chapter to familiarize yourself with some key DNP3 concepts and their implementation in a SCADAPack controller.

When selected the DNP Settings window is opened, as shown below.

DNP Settings

Application Layer
Data Link Layer
Master
Master Poll
Address Mapping
Routing
Binary Inputs
Binary Outputs
16-bit Analog Inputs
32-bit Analog Inputs
Short Floating Point Analog Inputs
16-bit Analog Outputs
32-bit Analog Outputs
Short Floating Point Analog Outputs
16-bit Counter Inputs
32-bit Counter Inputs

Application Layer Communication

Application Confirmation: Disabled
Maximum Fragment Length: 2048
Retries: 0
Application Timeout: 5000 msec

Time Synchronization

☐ None
☐ At Start Up Only
☒ Interval: 60 minutes

Unsolicited Responses

	Class 1	Class 2	Class 3
Enable Unsolicited	Disabled	Disabled	Disabled
Hold Time	60	60	60
Hold Count	10	10	10
Enable Unsolicited Responses on Start Up	Yes		
Send Initial Unsolicited Response on Start Up	No		
☒ Resend Unreported Events after	60		

Dial Up

Modem Initialization: &F0 S0=1 &W0 &Y0
Attempts: 2
Dial Type: Tone
Connect Timeout: 45 seconds
Inactivity Timeout: 45 seconds
Pause Time: 10 seconds

Operate Timeout: 15 seconds
Report only Level 2 Objects in Class Polls: No
☐ Limit Maximum Events in Read Response: 0

☐ Allow Duplicate Modbus Addresses

OK, Cancel, Add, Copy, Delete, Move Up, Move Down, Help

The DNP Settings window has a tree control on the left side of the window. The tree control appears differently depending on the controller type selected. The SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 and SCADAPack 32P controllers support DNP master and include the bolded items in the following list. Other SCADAPack controllers do not support DNP master and do not include the bolded items. This tree control contains headings for:

- Application Layer
- Data Link Layer
- **Master**
- **Master Poll**
- **Address Mapping**
- Routing
- Binary Inputs
- Binary Outputs
- 16-Bit Analog Inputs
- 32-Bit Analog Inputs

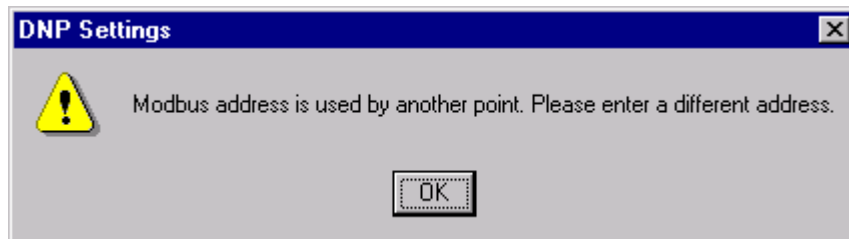
- Short Floating Point Analog Inputs
- 16-Bit Analog Outputs
- 32-Bit Analog Outputs
- Short Floating Point Analog Outputs
- 16-Bit Counter Inputs
- 32-Bit Counter Inputs

When a tree control is selected by clicking the mouse on a heading a property page is opened for the header selected. From the property page the DNP configuration parameters for the selected header is displayed.

As DNP objects are defined they are added as leaves to the object branch of the tree control. When an object is defined the object branch will display a collapse / expand control to the left of the branch.

The **Allow Duplicate Modbus Addresses** checkbox (in the bottom left corner) determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Uncheck the box if you want to be warned about duplicate addresses. If an attempt is made to use a Modbus address that has already been used for another DNP point the following warning is displayed.



8.6.1 Application Layer Configuration

The Application Layer property page is selected for editing by clicking Application Layer in the tree control section of the DNP Settings window. When selected the Application Link Layer property page is active.

DNP Settings

Application Layer

- Application Layer
- Data Link Layer
- Master
- Master Poll
- Address Mapping
- Routing
- Binary Inputs
- Binary Outputs
- 16-bit Analog Inputs
- 32-bit Analog Inputs
- Short Floating Point Analog Inputs
- 16-bit Analog Outputs
- 32-bit Analog Outputs
- Short Floating Point Analog Outputs
- 16-bit Counter Inputs
- 32-bit Counter Inputs

Communication

Application Confirmation: Disabled

Maximum Fragment Length: 2048

Retries: 0

Application Timeout: 5000 msec

Time Synchronization

None
At Start Up Only
☒ Interval: 60 minutes

Unsolicited Responses

	Class 1	Class 2	Class 3
Enable Unsolicited	Disabled	Disabled	Disabled
Hold Time	60	60	60
Hold Count	10	10	10
Enable Unsolicited Responses on Start Up	Yes		
Send Initial Unsolicited Response on Start Up	No		
☒ Resend Unreported Events after	60		

Dial Up

Modem Initialization: &F0 S0=1 &W0 &Y0

Attempts: 2

Dial Type: Tone

Connect Timeout: 45 seconds

Inactivity Timeout: 45 seconds

Pause Time: 10 seconds

Operate Timeout: 15 seconds

Report only Level 2 Objects in Class Polls: No

☐ Limit Maximum Events in Read Response: 0

☐ Allow Duplicate Modbus Addresses

Buttons: OK, Cancel, Add, Copy, Delete, Move Up, Move Down, Help

Application Layer parameters are set in this property page. Each parameter is described in the following paragraphs.

The **Communication** section of the dialog contains the configurable application layer communication parameters.

When the **Application Confirmation** feature is enabled, the SCADAPack controller requests a confirmation from the master station for any data transmitted. When it is disabled, the controller does not request a confirmation from the master station and assumes that the master receives the data it sends successfully. However if the data includes event data (including unsolicited messages), the controller requests a confirmation from the master regardless of whether this feature is enabled or disabled. Valid selections for this parameter are:

- Enabled
- Disabled

The **Maximum Fragment Length** is maximum size of a single response fragment that the RTU will send. If the complete response message is too large to fit within a single fragment, then the SCADAPack controller will send the response in multiple fragments. Valid values are between 100 and 2048 bytes.

This parameter is adjustable to allow for interoperability with simple DNP3 devices that require smaller application layer fragments. Devices with limited memory may restrict the application layer fragment size to as low as 249 bytes.

Note: The Maximum Fragment Length parameter applies to responses from read commands only. It does not affect unsolicited responses.

The **Retries** entry maximum number of times the application layer will retry sending a response or an unsolicited response to the master station. This does not include any retries performed by the data link layer. Valid values are between 0 and 255.

Note: Using application layer Confirmation and Retries is inherently more efficient than using data link layer Confirmation and Retries. Each fragment sent by the Application layer may require as many as 10 data link layer frames to be sent, each with its own confirmation message. The application layer is typically preferred for message confirmation for this reason.

The **Application Timeout** is the expected time duration (in milliseconds) that the master station's application layer requires to process and respond to a response from the SCADAPack controller. This SCADAPack controller uses this value in setting its time-out interval for master station responses. This value should be large enough to prevent response time-outs. The value must be kept small enough so as not to degrade system throughput. The value of this element is dependent on the master station. Valid values are between 100 and 60000 milliseconds.

The **Time Synchronization** section of the dialog defines when and how often the SCADAPack outstation prompts the master station to synchronize the SCADAPack controller time. Messages must be sent between the Master and Remote stations for Time Synchronization to work. Valid selections for this parameter are:

- The **None** selection will cause the SCADAPack controller to never request Time Synchronization.
- The **At Start Up Only** selection will cause the SCADAPack controller to request Time Synchronization at startup only.
- **The Interval** selection will cause the SCADAPack controller to request Time Synchronization at startup and then every **Interval** minutes after receiving a time synchronization from the master. Valid entries for Interval are between 1 and 32767 minutes. The default value is 60 minutes.

Note: Time Synchronization may instead be initiated by the Master for each Outstation. This may be selected in the Add/Edit Master Poll dialog. It is not required to enable Time Synchronization at both the Master and the Outstation.

The **Unsolicited Response** section of the dialog defines which **class** objects are enabled or disabled from generating report by exception responses. Unsolicited responses are individually configured for Class 1, Class 2, and Class 3 data.

The **Enable Unsolicited** controls enables or disables the generation of unsolicited events for Class 1, Class 2 or Class 3 data. If unsolicited responses are disabled for a Class the controller never generates unsolicited events for that Class. If unsolicited responses are enabled the controller generates unsolicited events for that Class if its value or state exceeds a defined threshold. Valid selections are:

- Enabled

- Disabled

Note that the controller does not transmit collected unsolicited messages (or responses) to a master, even after the Hold Time or Hold Count conditions have been met, unless its 'Enable Unsolicited Responses on Start Up' field is set to 'Yes' or the master triggers this transmission.

To configure a master to control unsolicited message transmission from a remote, see the **Master Poll** configuration panel.

The **Hold Time** parameter is used only when unsolicited responses are enabled for a Class. This parameter defines the maximum period (in seconds) the RTU will be allowed to hold its events before reporting them to the DNP master station. When the hold time has elapsed since the first event occurred, the RTU will report to the DNP master station all events accumulated up to then. This parameter is used in conjunction with the **Hold Count** parameter in customizing the unsolicited event reporting characteristics. The value used for the Hold Time depends on the frequency of event generation, topology and performance characteristics of the system. The valid values for this parameter are 0 - 65535. The default value is 60 seconds.

The **Hold Count** parameter is used only when unsolicited responses are enabled for a Class. This parameter defines the maximum number of events the RTU will be allowed to hold before reporting them to the DNP master station. When the hold count threshold is reached, the RTU will report to the master, all events accumulated up to that point. This parameter is used in conjunction with the **Hold Time** in customizing the unsolicited event reporting characteristics. To guarantee an unsolicited response is sent as soon as an event occurs, set the Hold Count parameter to 1. The valid values for this parameter are 1 - 65535. The default value is 10.

The **Enable Unsolicited Responses on Start Up** parameter enables or disables unsolicited responses on startup. This affects the default controller behaviour after a start-up or restart. Some hosts require devices to start up with unsolicited responses enabled. It should be noted this is non-conforming behaviour according to the DNP standard. Valid selections are:

- Yes
- No

The default selection is **Yes**.

The **Send Initial Unsolicited Response on Startup** parameter enables or disables Send Initial unsolicited responses on startup. This parameter controls whether an initial unsolicited response with null data is sent after a start-up or restart. Valid selections are:

- Yes
- No

The default selection is **No**.

The **Resend unreported events after** parameter enables or disables the retransmission of events after all attempts to report the events have failed.

Many communications networks experience occasional communications failures. In such networks, even when message retries are used, there is a chance that some messages will fail – meaning there is a chance some unsolicited messages will fail and change events will not be reported to the master station. The events remain in the outstation buffers until polled or additional events are generated.

To address this issue, and ensure guaranteed delivery of high priority events, the **Resend unreported events after** parameter is added to the DNP configuration. This parameter controls a timer for retrying the transmission of unsolicited messages.

Whenever a DNP unsolicited message fails, including all its retries, then instead of just retiring the message and reporting it as a failed message, an unsolicited resend timer is initiated. After the configured time delay has passed, the unsolicited message will be sent again, including the configured retries. This process will be repeated continuously until the unsolicited message is successfully sent and acknowledged. In the case of multiple masters the unsolicited resend timer is uninitiated after the retries are expired for the last master in the polling list.

Note: SCADAPack firmware 2.44 (and later), SCADAPack 32 Firmware 1.92 (and later), SCADAPack 330, SCADAPack 334, SCADAPack 350 or SCADASense firmware 1.25 (and later) and SOLARPack 410 firmware 1.32 (and Later) support this feature.

Valid values are 1 to 65535 seconds. The default value is 0 seconds. The control is unselected by default.

If **Resend unreported events** is not selected, the controller will not resend unreported events after all attempts fail, until polled or until additional events are generated and their reporting threshold is reached.

The **Resend unreported events** control can be selected even when no classes are enabled. This allows the feature to be used in a mimic controller that is being used to pass outstations events to a host.

The **Dial Up** section of the dialog defines modem parameters used when a dial up modem is used to communicate with stations that use dial up communication. The phone numbers for the stations are defined in the Routing table.

The **Modem Initialization** is the string that will be sent to the modem prior to each call. This is an ASCII null-terminated string. The maximum length of the string is 64 characters, including the null terminator.

The **Attempts** controls the maximum number of dial attempts that will be made to establish a Dial Up connection. The valid values for this parameter are 1 – 10. The default value is 2.

The **Dial Type** parameter controls whether tone or pulse dialing will be used for the call. Valid values are Tone dialing or Pulse dialing. The default value is Tone dialing.

The **Connect Timeout** controls the maximum time (in seconds) after initiating a dial sequence that the firmware will wait for a carrier signal before hanging up. The valid values for this parameter are 1 – 65535. The default value is 45.

The **Inactivity Timeout** controls the maximum time after message activity that a connection will be left open before hanging up. The valid values for this parameter are 1 – 65535 seconds. The default value is 45 seconds.

The **Pause Time** controls the delay time (in seconds) between dial events, to allow time for incoming calls. The valid values for this parameter are 1 – 65535. The default value is 10.

The **Operate Timeout** parameter specifies the timeout interval between a Select and Operate request from the Master. If after receiving a valid **Select** control output request from the master, the RTU does not receive the corresponding **Operate** request within this time-out interval, the control output request fails. The value of this parameter, expressed in seconds, is dependent on the master station, the data link and physical layer. Valid values are 1 to 6500 seconds. The default value is 15 seconds. The Master must have the Select/Operate functionally in order to use this feature.

The **Report only Level 2 Compliant Objects in Class Polls** parameter affects how Short Float Analog Input, Short Float Analog Output, and 32-bit Analog Output objects are reported. These

objects are converted to 32-bit Analog Input and 16-bit Analog Output objects when this parameter is selected. Valid selections are:

- Yes
- No

The default selection is No.

The **Limit Maximum Events in Read Response** parameter allows limiting the number of events in a read response. Select the checkbox to enable the limit. Valid values are 1 to 65535. The default value is disabled.

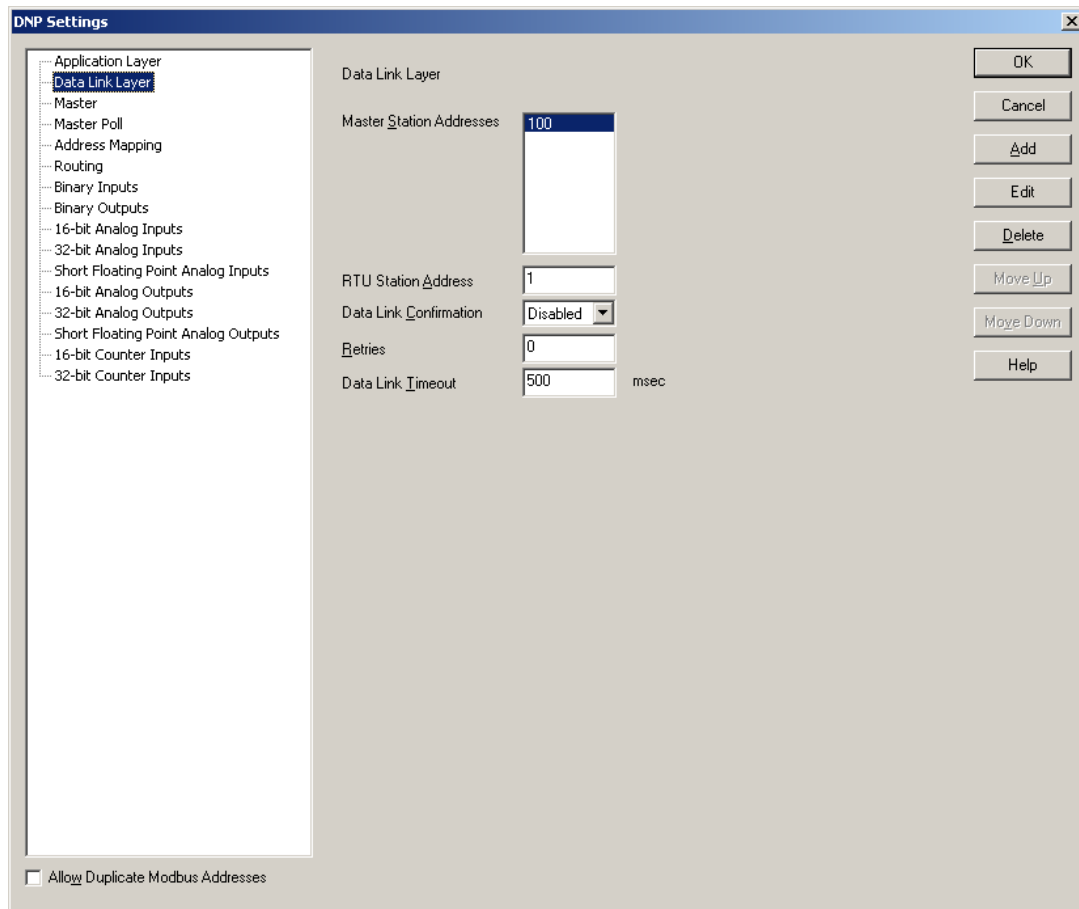
Note: The Maximum Events parameter applies to responses from read commands only. It does not affect unsolicited responses.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

- Click the **OK** button to accept the configuration changes and close the DNP Settings dialog.
- Click the **Cancel** button to close the dialog without saving any changes.

8.6.2 Data Link Layer Configuration

The Data Link Layer property page is selected for editing by clicking Data Link Layer in the tree control section of the DNP Settings window. When selected the Data Link Layer property page is active.



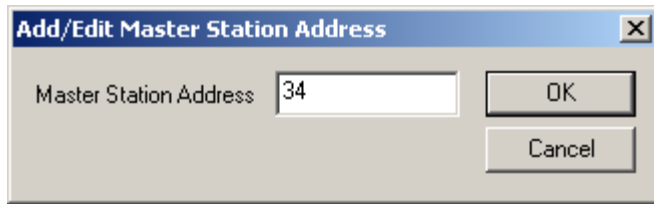
Data Link Layer parameters are set in this property page. Each parameter is described in the following paragraphs.

The **Master Station Addresses** list box contains a list of Master station addresses that the SCADAPack controller will respond to. The default list contains one master address of 100. This address may be edited, or changed, and up to 8 master stations may be added to the list. Valid entries for Master Station Addresses are 0 to 65519.

- When a master station polls for event data, the controller will respond with any events that have not yet been reported to that master station.
- When an unsolicited response becomes due, it will be sent to each configured master station in turn. A complete unsolicited response message transaction, including retries, will be sent to the first configured master station. When this transaction has finished, a complete unsolicited response message transaction including retries will be sent to the next configured master station, and so on for all the configured master stations.
- Change events will be retained in the event buffer until they have been successfully reported to all configured master stations.

Select the **Add** button to enter a new address to the Master Station Address list. Selecting the Add button opens the **Add Master Station Address** dialog. Up to 8 entries can be added to the table. An error message is displayed if the table is full.

Select the **Edit** button to edit address in the Master Station Address list. Selecting the Edit button opens the **Edit Master Station Address** dialog. The button is disabled if there are no entries in the list.



The **Master Station Address** edit box specifies the Master Station Address. Enter any valid Station address from 0 to 65519.

- The **OK** button adds the Master Station Address to the list and closes the dialog. An error is displayed if the Master Station Address is invalid, if the address is already in the list, or if the address conflicts with the RTU station address.
- The **Cancel** button closes the dialog without making any changes.

The **RTU Station Address** parameter specifies the address of this RTU. It is the source address used by this DNP driver when communicating with a master station. Each DNP station in a network must have a unique address, including the Master station. Valid entries for RTU Station Address are 0 to 65519.

The **Data Link Confirmation** parameter specifies whether or not the RTU requests the underlying data link transmitting its response to use a high quality service, which generally means that the data link requires the receiving data link to confirm receipt of all messages.

The **Retries** parameter specifies the maximum number of times the data link layer will retry sending a message to the master station. This parameter is only used when responding to a request from a Master station, when there is no corresponding entry in the Routing dialog for that station. This is independent of the application layer retries. The valid values for this parameter are 0 - 255. Setting the value to 0 disables sending retries.

Note: Using data link layer Confirmation and Retries is inherently less efficient than application layer Confirmation and Retries. Each fragment sent by the Application layer may require as many as 10 data link layer frames to be sent, each with its own confirmation message. The data link layer is typically not used for message confirmation for this reason.

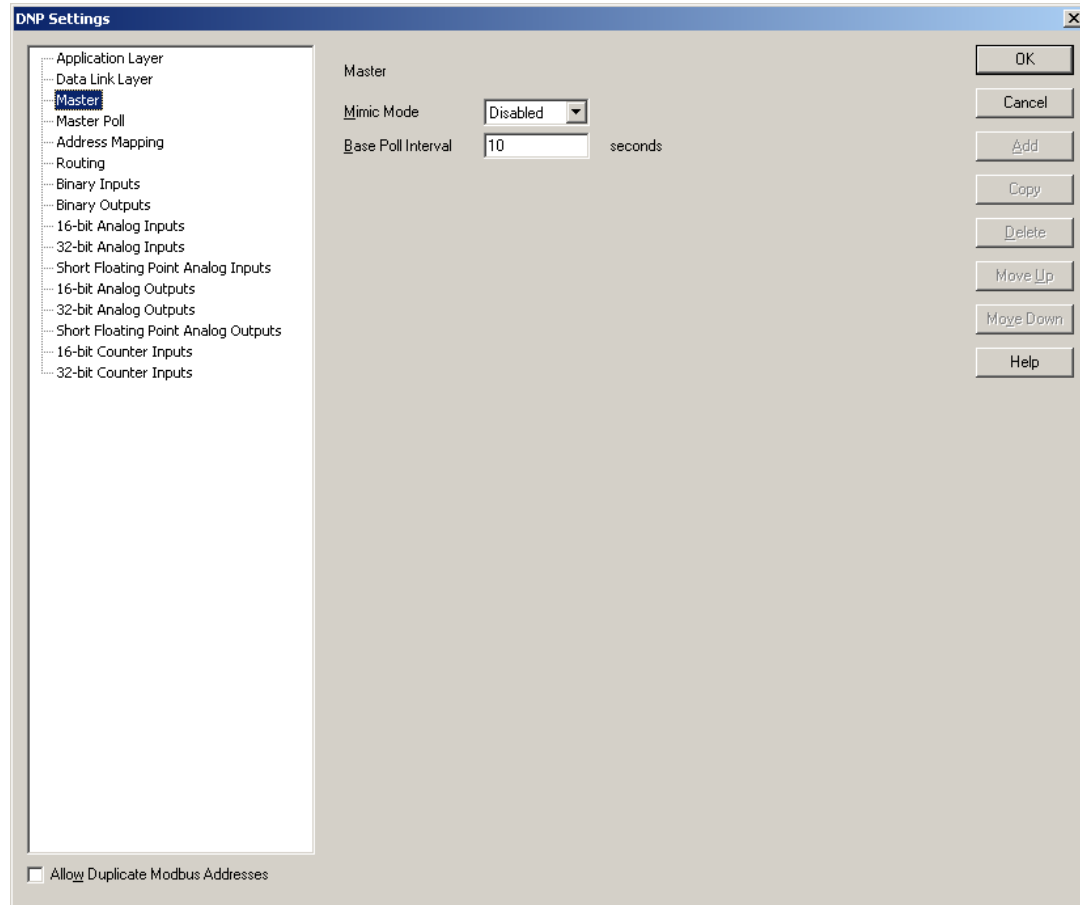
The **Data Link Timeout** parameter specifies the expected time duration that the master station's data link layer requires to process and respond to a message from the RTUs data link layer. It is used by the RTU in setting its time-out interval for master station responses. This value should be large enough to prevent response time-outs. The value must be kept small enough so as not to degrade system throughput. The value of this element is dependent on the master station. It is expressed in milliseconds. Valid values are 10 to 60000 milliseconds. The default value is 500 milliseconds.

- Click the **OK** button to accept the configuration changes and close the DNP Settings dialog.
- Click the **Cancel** button to close the dialog without saving any changes.
- Click the **Delete** button to remove the selected rows from the list. This button is disabled if there are no entries in the list.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

8.6.3 Master

The Master property page is selected for editing by clicking Master in the tree control section of the DNP Settings window. This selection is only visible if the controller type is SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P. These controllers support DNP Master. When selected the Master Application Link Layer property page is active.



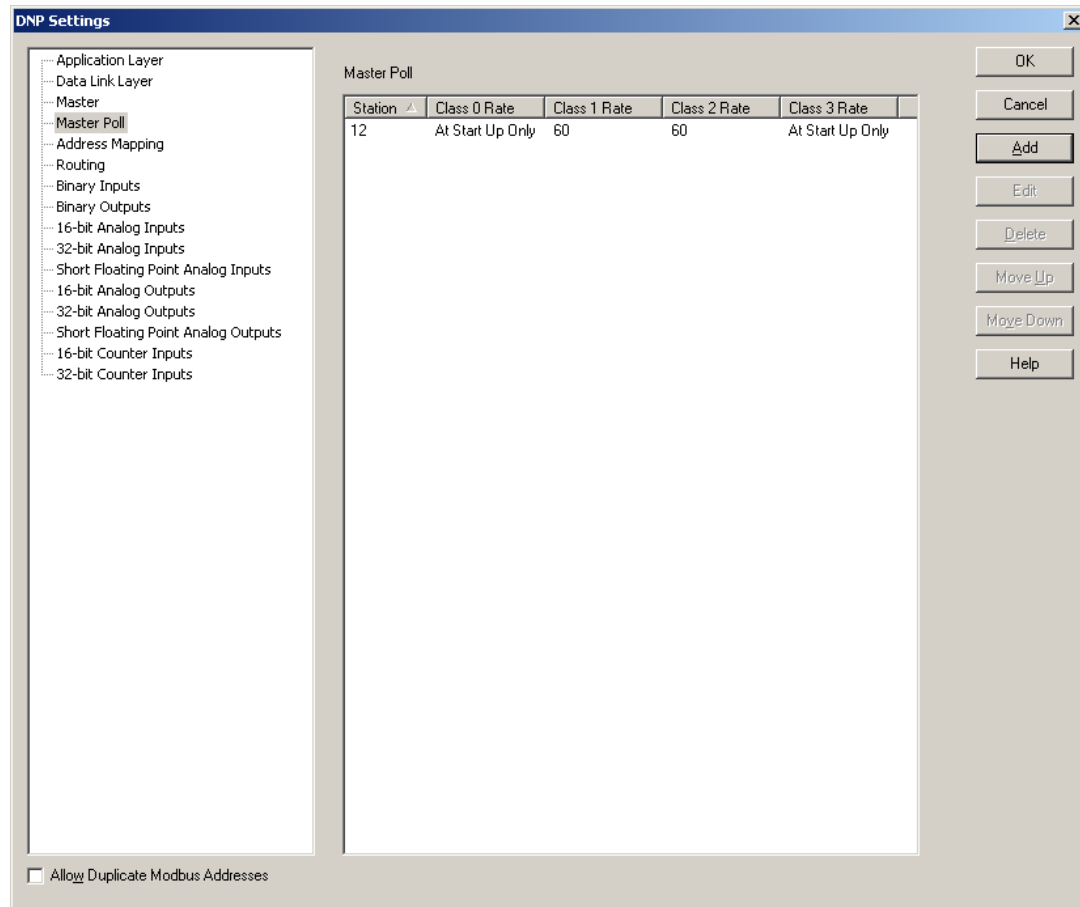
Master parameters are set in this property page. Each parameter is described in the following paragraphs.

The **Mimic Mode** parameter specifies the DNP Mimic Mode. The valid selections are Enable or Disable. When DNP Mimic Mode is enabled the controller will intercept DNP messages destined for a remote DNP station address, and will respond directly, as though the controller were the designated target. For read commands, the controller will respond with data from its Remote DNP Objects corresponding with the intended target address. For write commands, the controller will write data into its Remote DNP Objects, and issue a direct response to acknowledge the command. It will then issue a new command to write the data to the designated target. See Section [8.3.1.1-SCADAPack DNP Mimic Master](#) section for an explanation of the concept around Mimic Mode. The default selection is Disabled.

The **Base Poll Interval** parameter is the base interval (in seconds) for polling slave devices. The poll rates and issuing time synchronisation will be configured in multiples of the base poll interval. The slave devices with the same poll rates will be polled in the order they appear in the poll table. The valid values for this parameter are 1 to 65535. The default value is 10 seconds.

8.6.4 Master Poll

The Master Poll property page is selected for editing by clicking Master Poll in the tree control section of the DNP Settings window. This selection is only visible if the controller type is a SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P. These controllers support DNP Master. When selected the Master Poll property page is active and button Copy is renamed to Edit.



The Master Poll displays slave devices to be polled by this master station as a row, with column headings, in the table. The table may have up to 1000 entries. A vertical scroll bar is used if the list exceeds the window size.

Note: All slave devices in the Master Poll table need to be added to the Routing table.

The **Station** column displays the address of the DNP slave device to be polled. Each entry in the table should have unique DNP Station Address.

The **Class 0 Rate** column displays the rate of polling for Class 0 data, as a multiple of the base poll interval.

The **Class 1 Rate** column displays the rate of polling for Class 1 data, as a multiple of the base poll interval.

The **Class 2 Rate** column displays the rate of polling for Class 2 data, as a multiple of the base poll interval.

The **Class 3 Rate** column displays the rate of polling for Class 3 data, as a multiple of the base poll interval.

- The **OK** button saves the table data and closes the DNP Settings dialog.
- The **Cancel** button closes the dialog without saving changes.

Select the **Add** button to enter a new row in the Master Poll. Selecting the Add button opens the **Add/Edit Master Poll** dialog.

Select the **Edit** button to modify the selected row in the Master Poll. Selecting the Edit button opens the **Add/Edit Master Poll** dialog containing the data from the selected row. This button is disabled if more than one row is selected or if there are no entries in the table.

The **Delete** button removes the selected rows from the table. This button is disabled if there are no entries in the table.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click on the column headings to sort the data. Clicking once sorts the data in ascending order. Clicking again sorts the data in descending order.

8.6.4.1 Add/Edit Master Poll Dialog

This dialog is used to edit an entry or add a new entry in the Master Poll.

The **Station** edit control displays the address of the DNP slave device to be polled. Valid values are 0 to 65519.

The **Class 0 Polling** section of the dialog specifies the type and rate of polling for Class 0 data.

- The **None** selection disables class 0 polling for the slave station. This is the default selection.

- The **At Start Up Only** selection will cause the master to poll the slave station at startup only.
- The **Interval** selection will cause the master to poll the slave station at startup and then every **Interval** of the base poll interval. For example if the base poll interval is 60 seconds and the Interval parameter is set to 60 then the master will poll the slave station every hour. Valid values are 1 to 32767. The default value is 60.
- The **Poll Offset** parameter is used to distribute the load on the communication network. The Poll Offset is entered in multiples of the base poll interval. Valid values for this parameter are 0 to the Poll Interval value minus 1. Any non-zero value delays the start of polling for the specified objects by that amount. The default value is 0. This control is disabled when None is selected, and enabled otherwise. For an example of using the Poll Offset parameter see the **Poll Offset Example** at the end of this section.

The **Class 1 Polling** section of the dialog specifies the type and rate of polling for Class 1 data.

- The **None** selection disables class 1 polling for the slave station. This is the default selection.
- The **At Start Up Only** selection will cause the master to poll the slave station at startup only.
- The **Interval** selection will cause the master to poll the slave station at startup and then every **Interval** of the base poll interval. For example if the base poll interval is 60 seconds and the Interval parameter is set to 60 then the master will poll the slave station every hour. Valid values are 1 to 32767. The default value is 60.
- The **Poll Offset** parameter is used to distribute the load on the communication network. The Poll Offset is entered in multiples of the base poll interval. Valid values for this parameter are 0 to the Poll Interval value minus 1. Any non-zero value delays the start of polling for the specified objects by that amount. The default value is 0. This control is disabled when None is selected, and enabled otherwise. For an example of using the Poll Offset parameter see the **Poll Offset Example** at the end of this section.
- **Limit Maximum Events** allows limiting the number of events in poll responses for Class 1/2/3 data. The checkbox is not checked by default, meaning there is no limit on the number of events. Select the checkbox to specify a limit. The valid values for this parameter are 1 to 65535. The default value is 65535. This control is disabled when None is selected, and enabled otherwise. The Maximum Events parameter can be used to manage communication load on a system.

Consider the example of a master polling some data logging remotes, and the case where one of the remotes has been offline for a long time. The remote will have built up a large number of buffered events. If the master polled it for all events, the reply might take a long time, and cause an unwanted delay in the master's polling cycle. However if the master limits the number of events returned, the reply message duration will be more deterministic and the master can ensure its poll loop timing is maintained. In this case, the event retrieval from the data logger will be distributed over a number of poll cycles.

The **Class 2 Polling** section of the dialog specifies the type and rate of polling for Class 2 data.

- The **None** selection disables class 1 polling for the slave station. This is the default selection.
- The **At Start Up Only** selection will cause the master to poll the slave station at startup only.
- The **Interval** selection will cause the master to poll the slave station at startup and then every **Interval** of the base poll interval. For example if the base poll interval is 60 seconds and the Interval parameter is set to 60 then the master will poll the slave station every hour. Valid values are 1 to 32767. The default value is 60.

- The **Poll Offset** parameter is used to distribute the load on the communication network. The Poll Offset is entered in multiples of the base poll interval. Valid values for this parameter are 0 to the Poll Interval value minus 1. Any non-zero value delays the start of polling for the specified objects by that amount. The default value is 0. This control is disabled when None is selected, and enabled otherwise. For an example of using the Poll Offset parameter see the **Poll Offset Example** at the end of this section.
- **Limit Maximum Events** allows limiting the number of events in poll responses for Class 1/2/3 data. The checkbox is not checked by default, meaning there is no limit on the number of events. Select the checkbox to specify a limit. The valid values for this parameter are 1 to 65535. The default value is 65535. This control is disabled when None is selected, and enabled otherwise.

The Maximum Events parameter can be used to manage communication load on a system. Consider the example of a master polling some data logging remotes, and the case where one of the remotes has been offline for a long time. The remote will have built up a large number of buffered events. If the master polled it for all events, the reply might take a long time, and cause an unwanted delay in the master's polling cycle. However if the master limits the number of events returned, the reply message duration will be more deterministic and the master can ensure its poll loop timing is maintained. In this case, the event retrieval from the data logger will be distributed over a number of poll cycles.

The **Class 3 Polling** section of the dialog specifies the type and rate of polling for Class 3 data.

- The **None** selection disables class 1 polling for the slave station. This is the default selection.
- The **At Start Up Only** selection will cause the master to poll the slave station at startup only.
- The **Interval** selection will cause the master to poll the slave station at startup and then every **Interval** of the base poll interval. For example if the base poll interval is 60 seconds and the Interval parameter is set to 60 then the master will poll the slave station every hour. Valid values are 1 to 32767. The default value is 60.
- The **Poll Offset** parameter is used to distribute the load on the communication network. The Poll Offset is entered in multiples of the base poll interval. Valid values for this parameter are 0 to the Poll Interval value minus 1. Any non-zero value delays the start of polling for the specified objects by that amount. The default value is 0. This control is disabled when None is selected, and enabled otherwise. For an example of using the Poll Offset parameter see the **Poll Offset Example** at the end of this section.
- **Limit Maximum Events** allows limiting the number of events in poll responses for Class 1/2/3 data. The checkbox is not checked by default, meaning there is no limit on the number of events. Select the checkbox to specify a limit. The valid values for this parameter are 1 to 65535. The default value is 65535. This control is disabled when None is selected, and enabled otherwise.

The Maximum Events parameter can be used to manage communication load on a system. Consider the example of a master polling some data logging remotes, and the case where one of the remotes has been offline for a long time. The remote will have built up a large number of buffered events. If the master polled it for all events, the reply might take a long time, and cause an unwanted delay in the master's polling cycle. However if the master limits the number of events returned, the reply message duration will be more deterministic and the master can ensure its poll loop timing is maintained. In this case, the event retrieval from the data logger will be distributed over a number of poll cycles.

The **Time Synchronization Rate** section of the dialog specifies the rate of issuing a time synchronization to this device, as a multiple of the base poll interval. Valid selections for this parameter are:

- The **None** selection will disable issuing a time sync to this device. This is the default selection.
- The **At Start Up Only** selection will cause issuing a time synchronization at startup only.
- The **Interval** selection will cause the RTU to issue a time synchronization at startup and then every **Interval** of the base poll interval seconds. Valid entries for **Interval** are between 1 and 32767 the base poll interval seconds. The default value is 60.

The **Unsolicited Responses** section is used in conjunction with the Enable Unsolicited Responses on Start Up parameter on the Application Layer page. Certain non-SCADAPack slave devices are designed to start with their Enable Unsolicited Responses on Start Up parameter set to No. Selecting Enabled for any class causes the master to (after it detects the slave come online) send a command allowing the slave to begin sending Unsolicited Responses of that class.

With SCADAPack slaves the Enable Unsolicited Responses on Start Up parameter may be set to Yes, and the Accept Class parameters may be left at Disabled.

- The **Accept Class 1** selection displays the enable/disable status of unsolicited responses from the slave device for Class 1 events. The default selection is disabled.
- The **Accept Class 2** selection displays the enable/disable status of unsolicited responses from the slave device for Class 1 events. The default selection is disabled.
- The **Accept Class 3** selection displays the enable/disable status of unsolicited responses from the slave device for Class 1 events. The default selection is disabled.

The **Save IIN Flags** checkbox enables storing the IIN (Internal Indications) flags from the slave station in a Modbus database register. When this parameter is checked the IIN flags are saved to the entered Modbus register address. Valid entries are Modbus register addresses 30001 to 39999 and 40001 to 49999. The default value is 0.

The IIN flags are set by the slave to indicate the events in the following table. The events are bit mapped to the Modbus register. All bits except *Device Restarted* and *Time Synchronization required* are cleared when the slave station receives any poll or read data command. The master will write to bits 5 and 11 depending on the local conditions in the master.

Bit	Description
0	last received message was a broadcast message
1	Class 1 data available
2	Class 2 data available
3	Class 3 data available
4	Time Synchronization required
5	not used (returns 0)
6	Device trouble Indicates memory allocation error in the slave, or For master in mimic mode indicates communication failure with the slave device.
7	Device restarted (set on a power cycle)
8	Function Code not implemented
9	Requested object unknown or there were errors in the application data

Bit	Description
10	Parameters out of range
11	Event buffer overflowed Indicates event buffer overflow in the slave or master. The slave will set this bit if the event buffer in the slave is overflowed. The master will set this bit if the event buffer in the master has overflowed with events read from the slave. Ensure the event buffer size, in the master and slave, is set to a value that will ensure the buffer does not overflow and events are lost.
12	not used (returns 0)
13	not used (returns 0)
14	not used (returns 0)
15	not used (returns 0)

The **OK** button checks the data for this table entry. If the data is valid the dialog is closed. If the table data entered is invalid, an error message is displayed and the dialog remains open. The table entry is invalid if any of the fields is out of range. The data is also invalid if it conflicts with another entry in the table. Such conflict occurs when the station number is not unique. The ordering of items in this table is important.

The **Cancel** button closes the dialog without saving changes.

8.6.4.2 Poll Offset Example

The Poll Offset parameter enhances the control over timing of master poll messages, by allowing master poll messages to be staggered.

For example, a master station may have 10 slaves to poll, and must poll them every hour. If these are included in the poll table without any poll offset, they will all be polled in quick succession on the hour – resulting in a large burst of communication activity once per hour. On some types of communications networks (particularly radio) it is desirable to distribute communication load more evenly, to minimize the chance of collisions and to avoid the possibility of consuming bandwidth continuously for an extended period of time.

The poll offset parameter enables you to distribute the communication load evenly. In the above example, it is possible to stagger the master polls so slave stations are polled at 6-minute intervals. To do this, set the base poll interval to 10 seconds, and for each slave station set the poll rate and poll offset parameters as follows:

Base Poll (seconds)	Poll Rate (seconds)	Poll Offset (seconds)
10	360	0
10	360	36
10	360	72
10	360	108
10	360	144
10	360	180
10	360	216
10	360	252
10	360	288
10	360	324

8.6.5 Address Mapping

The Address Mapping property page is selected for editing by clicking Address Mapping in the tree control section of the DNP Settings window. This selection is only visible if the controller type is a

SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P.
These controllers support DNP Master.

DNP Settings

Application Layer
Data Link Layer
Master
Master Poll
Address Mapping
Routing
Binary Inputs
Binary Outputs
16-bit Analog Inputs
32-bit Analog Inputs
Short Floating Point Analog Inputs
16-bit Analog Outputs
32-bit Analog Outputs
Short Floating Point Analog Outputs
16-bit Counter Inputs
32-bit Counter Inputs

Address Mapping
Map Change Events: Disable

Station	Object Type	First Point	Number	First Register
1	Binary Input	0	1	10001

☐ Allow Duplicate Modbus Addresses

OK
Cancel
Add
Edit
Delete
Move Up
Move Down

The Address Mapping contains a set of mapping rules, which will allow the Remote DNP Objects to be mapped into local Modbus registers. This makes the data accessible locally, to be read and/or written locally in logic. It is also possible to perform data concentration – to map the remote DNP Objects into the local DNP address space – by defining local DNP objects and then mapping the remote DNP objects to the same Modbus registers. Change events can also be mapped in the same way - there is a configuration option to allow mapping of change events from a remote DNP slave into the local DNP change event buffer. The table may have up to 1000 entries. A vertical scroll bar is used if the list exceeds the window size.

The **Station** column displays the address of the remote DNP station.

The **Object Type** column displays the DNP data object type.

The **First Point** column displays the starting address of the remote DNP data points.

The **Number** column displays the number of remote points to be mapped.

The **First Register** column displays the starting address of local Modbus register where the remote data points are to be mapped.

The **Map Change Events** combo box enables or disables mapping of change events from a remote DNP slave into the local DNP change event buffer. Mapped change events may trigger an Unsolicited message to be sent, after the Hold Count or Hold Time is reached. It may be desired instead to map only static (live) values into local Modbus registers. The default selection is Disabled.

The default selection is Disabled.

The **OK** button saves the table data. No error checking is done on the table data.

The **Cancel** button closes the dialog without saving changes.

Select the **Add** button to enter a new row in the Address Mapping. Selecting the Add button opens the **Add/Edit Address Mapping** dialog.

Select the **Edit** button to modify the selected row in the Address Mapping. Selecting the Edit button opens the **Add/Edit Address Mapping** dialog containing the data from the selected row. This button is disabled if more than one row is selected. This button is disabled if there are no entries in the table.

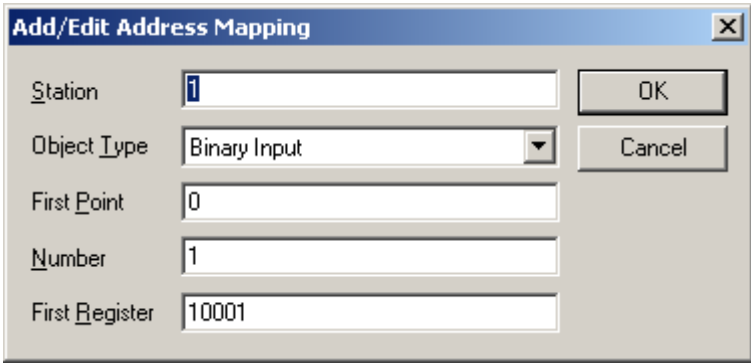
The **Delete** button removes the selected rows from the table. This button is disabled if there are no entries in the table.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click on the column headings to sort the data. Clicking once sorts the data in ascending order. Clicking again sorts the data in descending order.

8.6.5.1 Add/Edit Address Mapping Dialog

This dialog is used to edit an entry or add a new entry in the Address Mapping.

The image shows a Windows-style dialog box titled "Add/Edit Address Mapping". It contains five input fields: "Station" with the value "1", "Object Type" with a dropdown menu showing "Binary Input", "First Point" with the value "0", "Number" with the value "1", and "First Register" with the value "10001". To the right of the input fields are two buttons: "OK" and "Cancel".

The **Station** edit control displays the address of the remote DNP station. Valid values for this field are from 0 to 65519.

The **Object Type** combo box displays the DNP data Object Type. The list of available types includes: Binary Input, Binary Output, 16-bit Analog Input, 32-bit Analog Input, Short Floating Point Analog Input, 16-bit Analog Output, 32-bit Analog Output, Short Floating Point Analog Output, 16-bit Counter Input, 32-bit Counter Input. The Default selection is Binary Input.

The **First Point** edit control displays the starting address of the remote DNP data points. Valid values are from 0 to 65519.

The **Number** edit control displays the number of remote points to be mapped. Valid values for this field are from 1 to 9999.

The **First Register** edit control displays the starting address of local Modbus register where the remote data points are to be mapped. Valid values depend on the selection of DNP Object Type and are as follows:

For Binary Inputs valid range is from 10001 to 14096.

For Binary Outputs valid range is from 00001 to 04096.

For Analog Inputs and Counter Inputs valid range is from 30001 to 39999.

For Analog Outputs valid range is from 40001 to 49999.

The **OK** button checks the data for this table entry. If the data is valid the dialog is closed. If the table data entered is invalid, an error message is displayed and the dialog remains open. The table entry is invalid if any of the fields is out of range. The data is also invalid if it conflicts with another entry in the table. Such conflict occurs when the combination of station number, object type, and object address is not unique. The ordering of items in this table is not important.

The **Cancel** button closes the dialog without saving changes.

8.6.6 Routing

In a typical application the SCADAPack controller, configured for DNP, will act as a DNP slave station in a network. The SCADA system will communicate directly with all the DNP slave stations in the SCADA system.

DNP routing is a method for routing, or forwarding, of messages received from the SCADA system, through the SCADAPack controller, to a remote DNP slave station. The SCADAPack DNP slave station will respond to all messages sent to it from the SCADA system, as well as broadcast messages. When it receives a message that is not sent to it the message is sent on the serial port defined in the routing table. See Chapter 8.3 for an explanation of using and configuring DNP Routing.

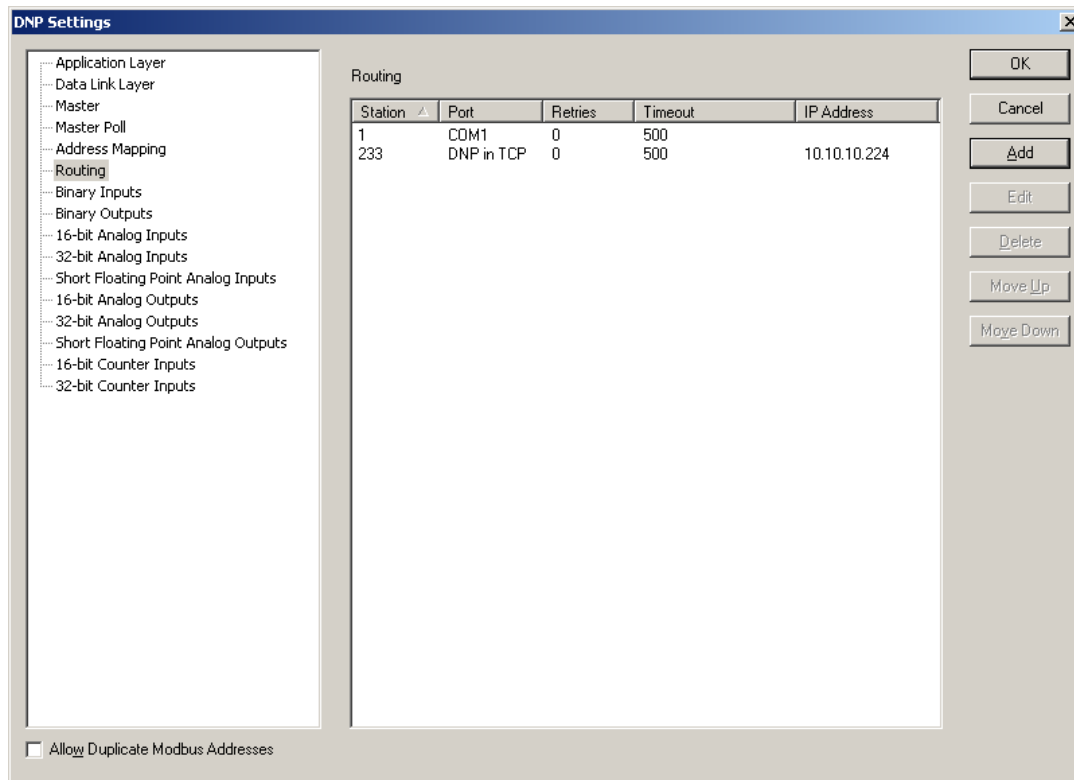
The advantage of this routing ability is that the SCADA system can communicate directly with the SCADAPack controller and the SCADAPack controller can handle the communication to remote DNP slave stations.

The DNP Routing table displays each routing translation as a row, with column headings, in the table. Entries may be added, edited or deleted using the button selections on the table. The table will hold a maximum of 128 entries.

The DNP Routing property page is selected for editing by clicking DNP Routing in the tree control section of the DNP Settings window. When selected the DNP Routing property page is displayed.

Notes:

- Routing must be enabled for the controller serial port in order to enable DNP routing.
- TelePACE version 2.63 cannot open files created with version 2.64, unless the Routing table is empty.
- TelePACE version 2.64 cannot open files created with version 2.65, unless the Routing table is empty.



The **Station** column displays the address of the remote DNP station.

The **Port** column displays the serial communications port, which should be used to communicate with this DNP station.

The **Retries** column displays the maximum number of Data Link retries, which should be used for this DNP station in the case of communication errors.

The **Timeout** column displays the maximum time (in milliseconds) to wait for a Data Link response before retrying or failing the message.

The **IP Address** column displays the IP address of the remote DNP station.

The **OK** button saves the table data. No error checking is done on the table data.

The **Cancel** button closes the dialog without saving changes.

Select the **Add** button to enter a new row in the DNP Routing table. Selecting the Add button opens the **Add/Edit DNP Route** dialog.

Select the **Edit** button to modify the selected row in the DNP Routing table. Selecting the Edit button opens the **Add/Edit DNP Route** dialog containing the data from the selected row. This button is disabled if more than one row is selected. This button is disabled if there are no entries in the table.

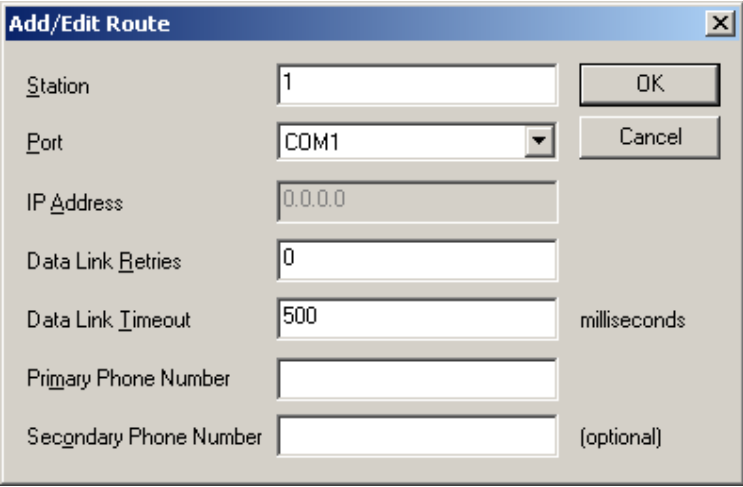
The **Delete** button removes the selected rows from the table. This button is disabled if there are no entries in the table.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click on the column headings to sort the data. Clicking once sorts the data in ascending order. Clicking again sorts the data in descending order.

8.6.6.1 Add/Edit DNP Route Dialog

This dialog is used to edit an entry or add a new entry in the DNP Routing table.

The image shows a Windows-style dialog box titled "Add/Edit Route". It contains several input fields and two buttons. The fields are: "Station" with a text box containing "1"; "Port" with a dropdown menu showing "COM1"; "IP Address" with a text box containing "0.0.0.0"; "Data Link Retries" with a text box containing "0"; "Data Link Timeout" with a text box containing "500" and the unit "milliseconds" to its right; "Primary Phone Number" with an empty text box; and "Secondary Phone Number" with an empty text box and the label "(optional)" to its right. The "OK" and "Cancel" buttons are located to the right of the "Station" and "Port" fields respectively.

The **Station** edit control displays the address of the remote DNP station. Valid values for this field are from 0 to 65519.

The **Port** combo box displays the communications port, which should be used to communicate with the remote DNP station. This combo box contains list of the valid communications ports, which will depend on the type of controller. For SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 and SCADAPack 32P controllers the list will contain DNP in TCP and DNP in UDP in addition to the serial port designations, COM1, COM2 etc.

The **IP Address** edit control is only enabled if the controller type is a SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32 or SCADAPack 32P. Enter the IP address of the remote DNP station.

The **Data Link Retries** edit control displays the maximum number of Data Link retries which should be used for this DNP station in the case of communication errors. This field overrides the Data Link Retries field in the global DNP parameters set in the Data Link Layer configuration. Valid values for this field are 0 to 255.

The **Data Link Timeout** edit control displays the maximum time (in milliseconds) to wait for a Data Link response before retrying or failing the message. This field overrides the Data Link Timeout field in the global DNP parameters in the Data Link Layer configuration. Valid values for this field are 100 to 60000, in multiples of 100.

The phone number parameters allow automatic dialing for stations that use dial-up ports. The Phone Number parameters are enabled only when the Port selected is a serial port.

The **Primary Phone Number** is the dialing string that will be used for the primary connection to the station. The controller will make 1 or more attempts, as configured in the Application layer, to connect using this number. If this connection fails then the Secondary Phone Number will be dialed, if it is entered.

Valid values are any ASCII string. The maximum length is 32 characters. Leave this blank if you are not using a dial-up connection. The default value is blank. The serial port type must be set to RS-232 Modem for dial-up operation.

The **Secondary Phone Number** is the dialing string that will be used for the secondary connection to the station. The controller will make 1 or more attempts, as configured in the Application layer, to connect using this number. This number is used after the primary connection fails on all attempts.

Valid values are any ASCII string. The maximum length is 32 characters. Leave this blank if you are not using a dial-up connection. The default value is blank. The serial port type must be set to RS-232 Modem for dial-up operation.

The **OK** button checks the data for this table entry. If the data is valid the dialog is closed. If the table data entered is invalid, an error message is displayed and the dialog remains open. The table entry is invalid if any of the fields is out of range. The data is also invalid if it conflicts with another entry in the table.

The **Cancel** button closes the dialog without saving changes.

8.6.6.2 Dynamic Routing

In addition to the configured routing table, there is an internal *dynamic* routing entry. This entry is not shown in the routing table. The dynamic routing entry listens to incoming messages and learns the address of the remote station and the communication port used for communicating with it.

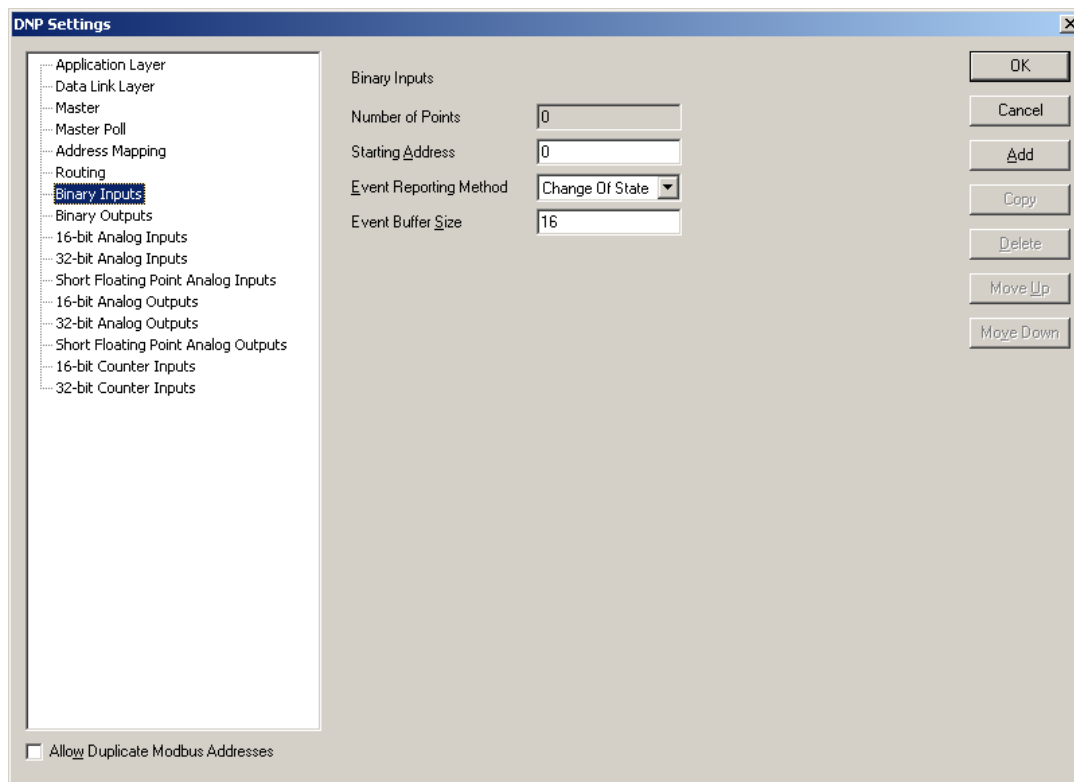
If there is no entry in the routing table, the RTU will use the dynamic routing entry to respond to a message on the same communication port as the incoming message.

The dynamic routing entry is not cleared on initialization. This is deliberate, and is important for controllers that need to be remotely reconfigured. In this case the host can initialize the controller without losing the communications link.

Note: Dynamic routing should not be used in a master station. Configure all slave stations in the routing table.

8.6.7 Binary Inputs Configuration

The Binary Inputs property page is selected for editing by clicking Binary Inputs in the tree control section of the DNP Settings window. When selected the Binary Inputs property page is active.



Binary Inputs parameters are set in this property page. Each parameter is described in the following paragraphs.

The **Number of Points** displays number of binary inputs reported by this RTU. This value will increment with the addition of each configured Binary Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Starting Address** parameter specifies the starting DNP address of the first Binary Input point.

The **Event Reporting Method** selection specifies how binary input events are reported. A **Change Of State** event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data. A **Log All Events** is event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. The selections are:

- Change of State
- Log All Events

The **Event Buffer Size** parameter specifies the maximum number of binary input change events buffered by the RTU. The buffer holds all binary input change events, regardless of the class to which they are assigned. If the buffer is completely full the RTU will lose the oldest events and retain the newest; the 'Event Buffer Overflowed' IIN flag will also be set to indicate that the buffer has overflowed. The Event Buffer size should be at least equivalent to the number of binary inputs defined as Change of State type. This will allow all binary inputs to change simultaneously without losing any events. The value of this parameter depends on how often binary input change events

occur and the rate at which the events are reported to the master station. The valid values for this parameter are 0 - 65535. Default value is 16.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

8.6.7.1 Adding Binary Inputs

Binary Inputs are added to the DNP configuration using the Binary Input property page. To add a Binary Input:

- Select **Binary Inputs** in the tree control section of the DNP Settings window.
- Click the **Add** button in the Binary Inputs property page.
- The **Binary Input** property page is now displayed.
- Edit the Binary Input parameters as required and then click the **Add** button.

As Binary Inputs are defined they are added as leaves to the Binary Inputs branch of the tree control. When Binary Inputs are defined the Binary Inputs branch will display a collapse / expand control to the left of the branch. Click this control to display all defined Binary Inputs.

The screenshot shows the 'DNP Settings' window. On the left is a tree control with the following structure: Application Layer, Data Link Layer, Master, Master Poll, Address Mapping, Routing, Binary Inputs (expanded), 0 - 10001 (selected), Binary Outputs, 16-bit Analog Inputs, 32-bit Analog Inputs, Short Floating Point Analog Inputs, 16-bit Analog Outputs, 32-bit Analog Outputs, Short Floating Point Analog Outputs, 16-bit Counter Inputs, and 32-bit Counter Inputs. The main area displays the 'Binary Input' property page with the following fields: DNP Address (0), Modbus Address (10001), Class of Event Object (Class 1), and Debounce (0 tenths of seconds). On the right side of the property page are buttons for OK, Cancel, Add, Copy, Delete, Move Up, Move Down, and Help.

The Binary Input parameters are described in the following paragraphs.

The **DNP Address** window displays the DNP Binary Input address of the point. Each Binary Input is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the Binary Inputs configuration dialog and increments by one with each defined Input.

The **Modbus Address** parameter specifies the Modbus address of the Binary Input assigned to the DNP Address. The SCADAPack and Micro16 controllers use Modbus addressing for all digital inputs. Refer to the *I/O Database Registers* section of the **TelePACE Ladder Logic Reference and User Manual** for complete information on digital input addressing in the SCADAPack and Micro16 controllers. Valid Modbus addresses are:

- 00001 through 09999
- 10001 through 19999

The **Class of Event Object** parameter specifies the event object class the Binary Input is assigned. The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Debounce** parameter limits the frequency of change events. The input must remain in the same state for the debounce time for a change of state to be detected. Note that the input is sampled every 0.1s. Changes shorter than the sample time cannot be detected. Valid values are 0 to 65535 tenths of seconds. The value 0 means no debounce. The default value is 0.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click the **OK** button to accept the Binary Input parameters and close the DNP Settings dialog.

Click the **Cancel** button to close the dialog without saving any changes.

Click the **Add** button to add the current Binary Input to the DNP configuration.

Click the **Copy** button to copy the current Binary Input parameters to the next DNP Address.

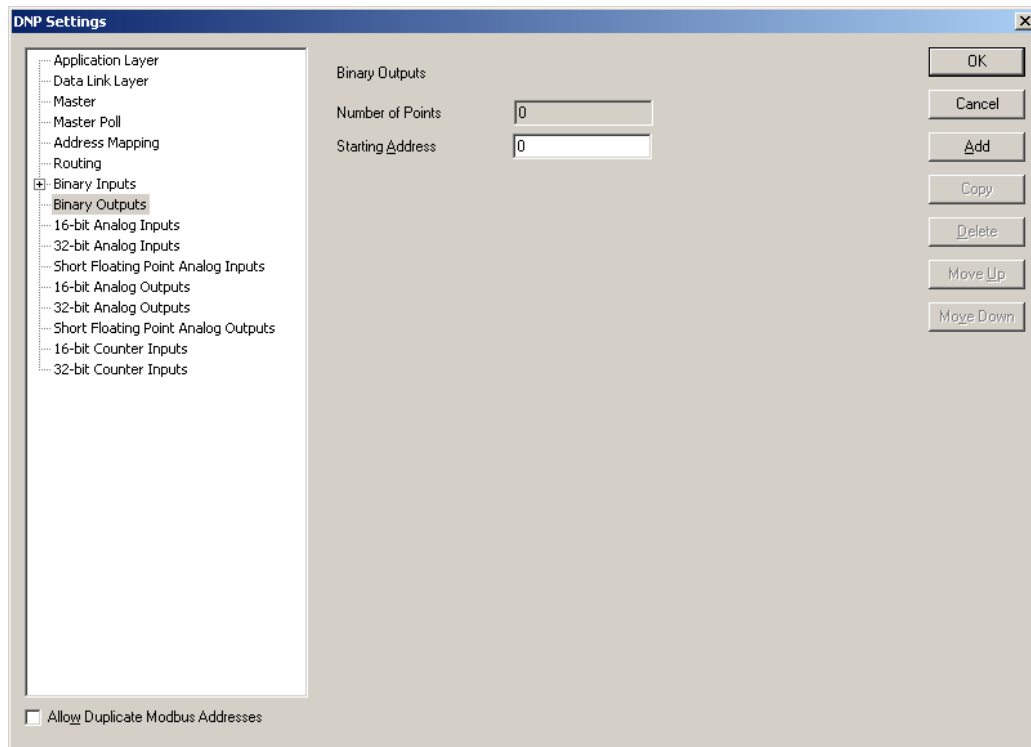
Click the **Delete** button to delete the current Binary Input.

Click the **Move Up** button to move the current Binary Input up one position in the tree control branch.

Click the **Move Down** button to move the current Binary Input down one position in the tree control branch.

8.6.8 Binary Outputs Configuration

The Binary Outputs property page is selected for editing by clicking Binary Outputs in the tree control section of the DNP Settings window. When selected the Binary Outputs property page is active.



Binary Outputs parameters are viewed in this property page.

The **Number of Points** displays the number of binary outputs reported by this RTU. This value will increment with the addition of each configured Binary Output point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Starting Address** parameter specifies the starting DNP address of the first Binary Output point.

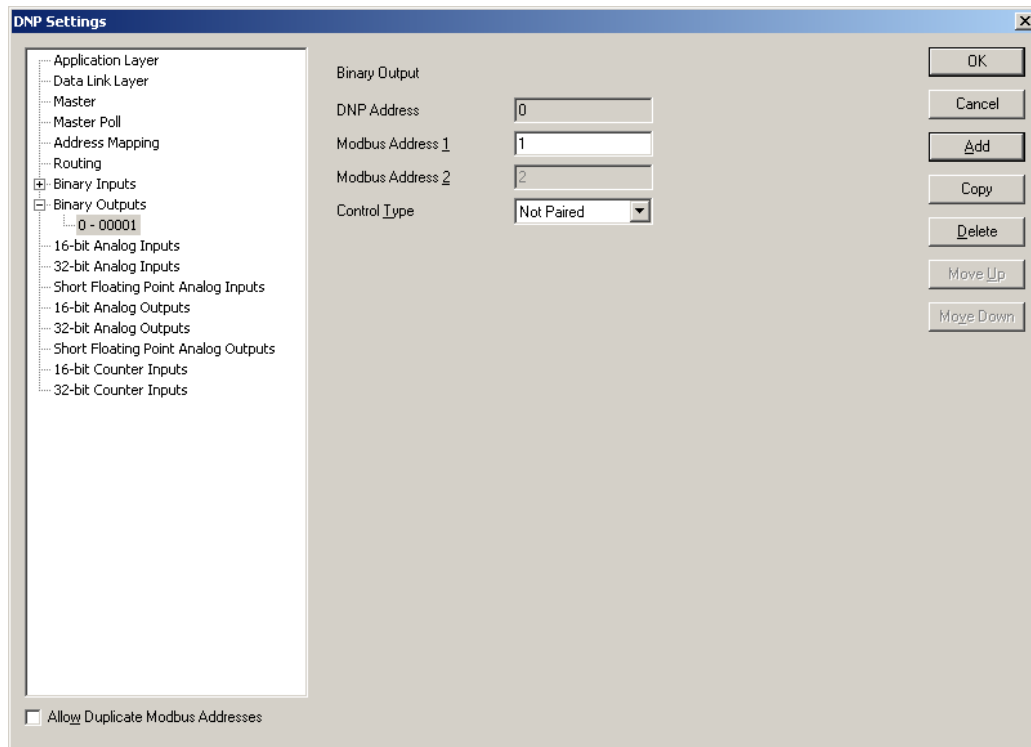
The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

8.6.8.1 Adding Binary Outputs

Binary Outputs are added to the DNP configuration using the Binary Output property page. To add a Binary Output:

- Select **Binary Outputs** in the tree control section of the DNP Settings window.
- Click the **Add** button in the Binary Outputs property page.
- The Binary Output property page is now displayed.
- Edit the Binary Output parameters as required and then click the **Add** button.

As Binary Outputs are defined they are added as leaves to the Binary Outputs branch of the tree control. When Binary Outputs are defined the Binary Outputs branch will display a collapse / expand control to the left of the branch. Click this control to display all defined Binary Outputs.



The Binary Output parameters are described in the following paragraphs.

The **DNP Address** window displays the DNP Binary Output address of the point. Each Binary Output is assigned a DNP address as they are defined. The DNP point address starts at the value defined in the Binary Outputs dialog and increments by one with each defined Output.

The **Modbus Address 1** parameter specifies the Modbus address of the Binary Output assigned to the DNP Address. The SCADAPack and Micro16 controllers use Modbus addressing for all digital outputs. Refer to the *I/O Database Registers* section of the *TelePACE Ladder Logic Reference Manual* for complete information on digital output addressing in the SCADAPack and Micro16 controllers. Valid Modbus addresses are:

- 00001 through 09999

The **Modbus Address 2** parameter specifies the second Modbus address of the second Binary Output assigned to the DNP Address when the Paired control type is selected. This selection is not active when the control type is Not Paired. Valid Modbus addresses are:

- 00001 through 09999

The **Control Type** parameter specifies whether the Binary Output is a paired control or not. If it is a paired control, i.e. trip/close output type, this means that the DNP address is associated to two physical control outputs and requires two Modbus addresses per DNP address. Control type selections are:

- Paired
- Not Paired

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click the **OK** button to accept the Binary Output parameters and close the DNP Settings dialog.

Click the **Cancel** button to close the dialog without saving any changes.

Click the **Add** button to add the current Binary Output to the DNP configuration.

Click the **Copy** button to copy the current Binary Output parameters to the next DNP Address.

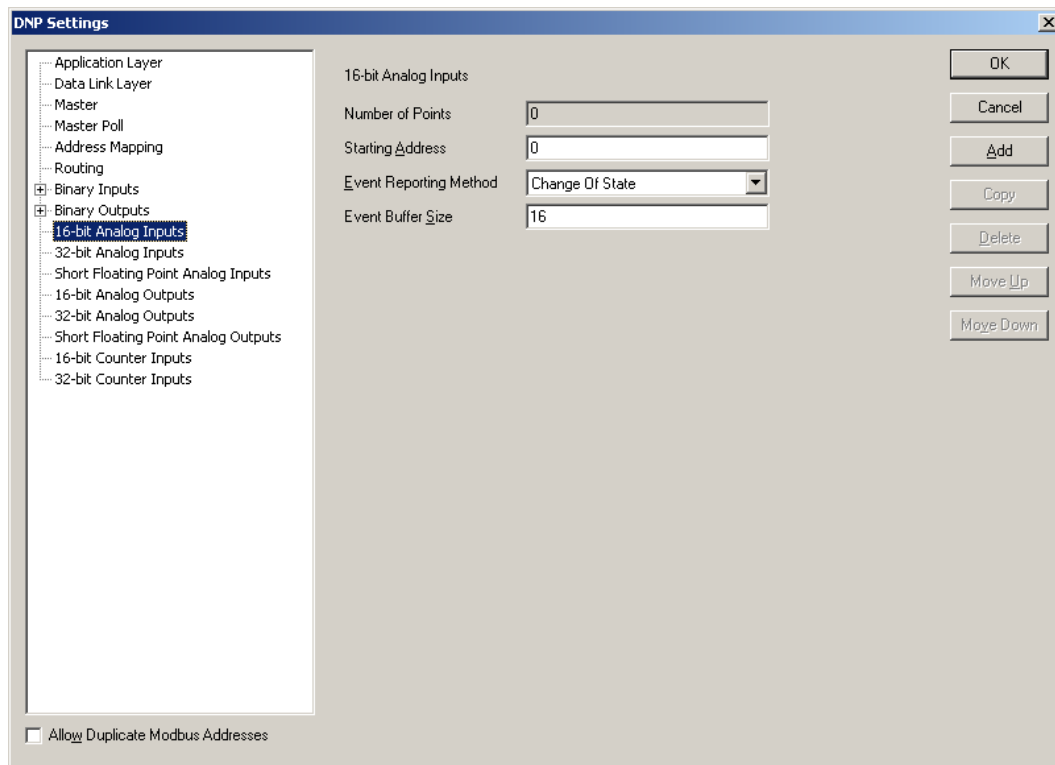
Click the **Delete** button to delete the current Binary Output.

Click the **Move Up** button to move the current Binary Output up one position in the tree control branch.

Click the **Move Down** button to move the current Binary Output down one position in the tree control branch.

8.6.9 16-Bit Analog Inputs Configuration

The 16-Bit Analog Inputs property page is selected for editing by clicking 16-Bit Analog Inputs in the tree control section of the DNP Settings window. When selected the 16-Bit Analog Inputs property page is active.



16-Bit Analog Inputs parameters are set in this property page. Each parameter is described in the following paragraphs.

The **Number of Points** displays the number of 16 bit analog inputs reported by the RTU. This value will increment with the addition of each configured 16-Bit Analog Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Starting Address** parameter specifies the DNP address of the first 16-bit Analog Input point.

The **Event Reporting Method** selection specifies how 16-bit Analog Input events are reported. A **Change Of State** event is an event object, without time, that is generated when the point changes

state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data. A **Log All Events** event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. The selections are:

- Change of State
- Log All Events

The **Event Buffer Size** parameter specifies the maximum number of 16-Bit Analog Input change events buffered by the RTU. The buffer holds all 16-Bit Analog Input events, regardless of the class to which they are assigned. If the buffer is completely full the RTU will lose the oldest events and retain the newest; the 'Event Buffer Overflowed' IIN flag will also be set to indicate that the buffer has overflowed. The Event Buffer size should be at least equivalent to the number of 16-Bit Analog Inputs defined as Change of State type. That will allow all 16-Bit Analog Inputs to exceed the deadband simultaneously without losing any events. The value of this parameter is dependent on how often 16-Bit Analog Input events occur and the rate at which the events are reported to the master station. The valid values for this parameter are 0 - 65535. Default value is 16.

For SCADAPack 32 and SCADAPack 32P controllers analog input events are processed by the DNP driver at a rate of 100 events every 100 ms. If more than 100 analog input events need to be processed they are processed sequentially in blocks of 100 until all events are processed. This allows the processing of 1000 analog input events per second.

For SCADASense Series of controllers, SCADAPack 100, SCADAPack LP, SCADAPack and Micro16 controllers analog input events are processed by the DNP driver at a rate of 20 events every 100 ms. If more than 20 analog input events need to be processed they are processed sequentially in blocks of 20 until all events are processed. This allows the processing of 200 analog input events per second.

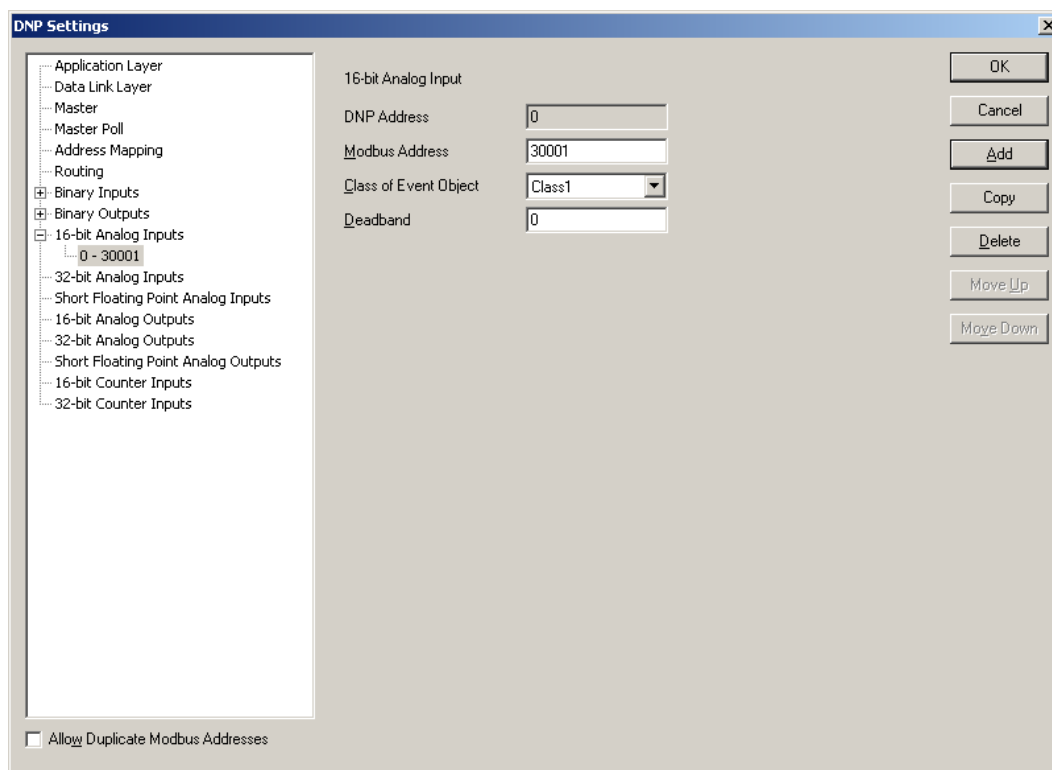
The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

8.6.9.1 Adding 16-Bit Analog Inputs

16-Bit Analog Inputs are added to the DNP configuration using the 16-Bit Analog Input property page. To add a 16-Bit Analog Input:

- Select **16-Bit Analog Inputs** in the tree control section of the DNP Settings window.
- Click the **Add** button in the 16-Bit Analog Inputs property page.
- The **16-Bit Analog Input** property page is now displayed.
- Edit the 16-Bit Analog Input parameters as required and then click the **Add** button.

As 16-Bit Analog Inputs are defined they are added as leaves to the 16-Bit Analog Inputs branch of the tree control. When 16-Bit Analog Inputs are defined the 16-Bit Analog Inputs branch will display a collapse / expand control to the left of the branch. Click this control to display all defined 16-Bit Analog Inputs.



The 16-Bit Analog Input parameters are described in the following paragraphs.

The **DNP Address** window displays the DNP 16-Bit Analog Input address of the point. Each 16-Bit Analog Input is assigned a DNP address as they are defined. The DNP point address starts at the value set in the 16-bit Analog Input configuration dialog and increments by one with each defined 16-Bit Analog Input.

The **Modbus Address** parameter specifies the Modbus address of the 16-Bit Analog Input assigned to the DNP Address. The SCADAPack and Micro16 controllers use Modbus addressing for all analog inputs. Refer to the *I/O Database Registers* section of the *TelePACE Ladder Logic Reference and User Manual* for complete information on analog input addressing in the SCADAPack and Micro16 controllers. Valid Modbus addresses are:

- 30001 through 39999
- 40001 through 49999

The **Class of Event Object** parameter specifies the event object class assigned to the 16-Bit Analog Input is assigned. If Unsolicited reporting is not required for a point, it is recommended to set its Class to **None**. All data points automatically become members of Class 0 or **None** (static data). The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Deadband** parameter specifies the minimum number of counts that the 16-Bit Analog Input must change since it was last reported in order to generate an event. Valid deadband values are 0 to 65535. A deadband of zero will cause any change to create an event.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click the **OK** button to accept the 16-Bit Analog Input parameters and close the DNP Settings dialog.

Click the **Cancel** button to close the dialog without saving any changes.

Click the **Add** button to add the current 16-Bit Analog Input to the DNP configuration.

Click the **Copy** button to copy the current 16-Bit Analog Input parameters to the next DNP Address.

Click the **Delete** button to delete the current 16-Bit Analog Input.

Click the **Move Up** button to move the current 16-Bit Analog Input up one position in the tree control branch.

Click the **Move Down** button to move the current 16-Bit Analog Input down one position in the tree control branch.

8.6.10 32-Bit Analog Inputs Configuration

The 32-Bit Analog Inputs property page is selected for editing by clicking 32-Bit Analog Inputs in the tree control section of the DNP Settings window. When selected the 32-Bit Analog Inputs property page is active.

The screenshot shows the 'DNP Settings' dialog box. On the left is a tree view with the following items: Application Layer, Data Link Layer, Master, Master Poll, Address Mapping, Routing, Binary Inputs, Binary Outputs, 16-bit Analog Inputs, **32-bit Analog Inputs** (selected), Short Floating Point Analog Inputs, 16-bit Analog Outputs, 32-bit Analog Outputs, Short Floating Point Analog Outputs, 16-bit Counter Inputs, and 32-bit Counter Inputs. The right pane is titled '32-bit Analog Inputs' and contains the following fields: 'Number of Points' (text box with '0'), 'Starting Address' (text box with '10000'), 'Event Reporting Method' (dropdown menu with 'Change Of State'), 'Event Buffer Size' (text box with '16'), and 'Word Order' (dropdown menu with 'TelePACE'). At the bottom left of the right pane is a checkbox labeled 'Allow Duplicate Modbus Addresses'. On the far right of the dialog are seven buttons: OK, Cancel, Add, Copy, Delete, Move Up, and Move Down.

32-Bit Analog Inputs parameters are set in this property page. Each parameter is described in the following paragraphs.

The **Number of Points** displays the number of 32-bit analog inputs reported by the RTU. This value will increment with the addition of each configured 32-Bit Analog Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Starting Address** parameter specifies the DNP address of the first 32-bit Analog Input point.

The **Event Reporting Method** selection specifies how 32-bit Analog Input events are reported. A **Change Of State** event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data. A **Log All Events** is event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. The selections are:

- Change of State
- Log All Events

The **Event Buffer Size** parameter specifies the maximum number of 32-Bit Analog Input change events buffered by the RTU. The buffer holds all 32-Bit Analog Input events, regardless of the class to which they are assigned. If the buffer is completely full the RTU will lose the oldest events and retain the newest; the 'Event Buffer Overflowed' IIN flag will also be set to indicate that the buffer has overflowed. The Event Buffer size should be at least equivalent to the number of 32-Bit Analog Inputs defined as Change of State type. That will allow all 32-Bit Analog Inputs to exceed the deadband simultaneously without losing any events. The value of this parameter is dependent on how often 32-Bit Analog Input events occur and the rate at which the events are reported to the master station. The valid values for this parameter are 0 - 65535. Default value is 16.

For SCADAPack 32 and SCADAPack 32P controllers analog input events are processed by the DNP driver at a rate of 100 events every 100 ms. If more than 100 analog input events need to be processed they are processed sequentially in blocks of 100 until all events are processed. This allows the processing of 1000 analog input events per second.

For SCADASense Series of controllers, SCADAPack 100, SCADAPack LP, SCADAPack and Micro16 controllers analog input events are processed by the DNP driver at a rate of 20 events every 100 ms. If more than 20 analog input events need to be processed they are processed sequentially in blocks of 20 until all events are processed. This allows the processing of 200 analog input events per second.

The **Word Order** selection specifies the word order of the 32-bit value. The selections are:

- **TelePACE** Least Significant Word in first register.
- **ISaGRAF** Most Significant Word in first register.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

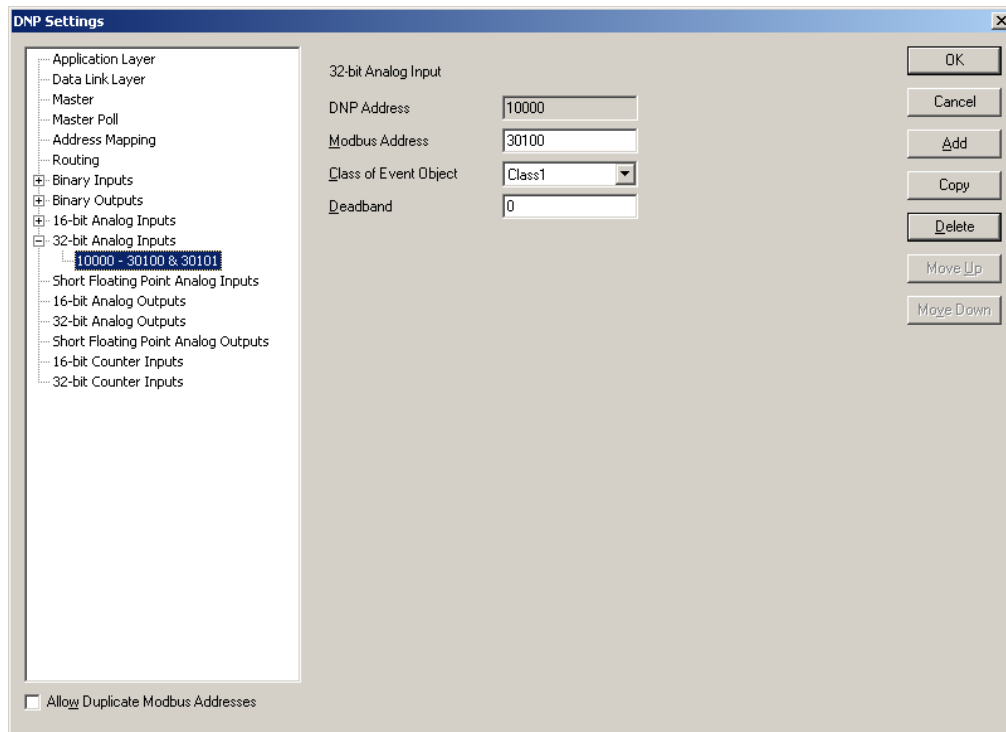
8.6.10.1 Adding 32-Bit Analog Inputs

32-Bit Analog Inputs are added to the DNP configuration using the 16-Bit Analog Input property page. To add a 32-Bit Analog Input:

- Select **32-Bit Analog Inputs** in the tree control section of the DNP Settings window.
- Click the **Add** button in the 32-Bit Analog Inputs property page.
- The **32-Bit Analog Input** property page is now displayed.
- Edit the 32-Bit Analog Input parameters as required and then click the **Add** button.

As 32-Bit Analog Inputs are defined they are added as leaves to the 32-Bit Analog Inputs branch of the tree control. When 32-Bit Analog Inputs are defined the 32-Bit Analog Inputs branch will display

a collapse / expand control to the left of the branch. Click this control to display all defined 32-Bit Analog Inputs.



The 32-Bit Analog Input parameters are described in the following paragraphs.

The **DNP Address** window displays the DNP 32-Bit Analog Input address of the point. Each 32-Bit Analog Input is assigned a DNP address as they are defined. The DNP point address starts at the value set in the 32-bit Analog Input configuration dialog and increments by one with each defined 32-Bit Analog Input.

The **Modbus Address** parameter specifies the Modbus addresses of the 32-Bit Analog Input assigned to the DNP Address. 32-Bit Analog Inputs use two consecutive Modbus registers for each assigned DNP Address, the address that is entered in this box and the next consecutive Modbus register. The SCADAPack and Micro16 controllers use Modbus addressing for all analog inputs. Refer to the *I/O Database Registers* section of the **TelePACE Ladder Logic Reference and User Manual** for complete information on analog input addressing in the SCADAPack and Micro16 controllers. Valid Modbus addresses are:

- 30001 through 39998
- 40001 through 49998

The **Class of Event Object** parameter specifies the event object class the 32-Bit Analog Input is assigned. If Unsolicited reporting is not required for a DNP point, it is recommended to set its Class 0 or **None**. All data points automatically become members of Class 0 or **None** (static data). The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Deadband** parameter specifies whether the RTU generates events. The value entered is the minimum number of counts that the 32-Bit Analog Input must change since it was last reported. Valid deadband values are 0 to 4,294,967,295. A deadband of zero will cause any change to create an event.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click the **OK** button to accept the 32-Bit Analog Input parameters and close the DNP Settings dialog.

Click the **Cancel** button to close the dialog without saving any changes.

Click the **Add** button to add the current 32-Bit Analog Input to the DNP configuration.

Click the **Copy** button to copy the current 32-Bit Analog Input parameters to the next DNP Address.

Click the **Delete** button to delete the current 32-Bit Analog Input.

Click the **Move Up** button to move the current 32-Bit Analog Input up one position in the tree control branch.

Click the **Move Down** button to move the current 32-Bit Analog Input down one position in the tree control branch.

8.6.11 Short Floating Point Analog Inputs

The Short Floating Point Analog Inputs property page is selected for editing by clicking Short Floating Point Analog Inputs in the tree control section of the DNP Settings window. When selected the Short Floating Point Analog Inputs property page is active.

The screenshot shows the 'DNP Settings' dialog box with the 'Short Floating Point Analog Inputs' property page selected. The left pane shows a tree view with 'Short Floating Point Analog Inputs' highlighted. The right pane contains the following fields and buttons:

- Number of Points:** 0
- Starting Address:** 20000
- Event Reporting Method:** Change Of State (dropdown)
- Event Buffer Size:** 16
- Word Order:** TelePACE/ISaGRAF (MSW First) (dropdown)
- Buttons:** OK, Cancel, Add, Copy, Delete, Move Up, Move Down
- Checkbox:** ☐ Allow Duplicate Modbus Addresses

Short Floating Point Analog Input parameters are set in this property page. Each parameter is described in the following paragraphs.

The **Number of Points** displays the number of Short Floating Point Analog Inputs reported by the RTU. This value will increment with the addition of each configured Short Floating Point Analog Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Starting Address** parameter specifies the DNP address of the first Short Floating Point Analog Input point.

The **Event Reporting Method** selection specifies how Short Floating Point Analog Input events are reported. A **Change Of State** event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data. A **Log All Events** is event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. The selections are:

- Change of State
- Log All Events

The **Event Buffer Size** parameter specifies the maximum number of Short Floating Point Analog Input change events buffered by the RTU. The buffer holds all Short Floating Point analog input events, regardless of the class to which they are assigned. If the buffer is completely full the RTU will lose the oldest events and retain the newest; the 'Event Buffer Overflowed' IIN flag will also be set to indicate that the buffer has overflowed. The Event Buffer size should be at least equivalent to the number of Short Floating point analog inputs defined as Change of State type. That will allow all Short Floating Analog Point Inputs to exceed the deadband simultaneously without losing any events. The value of this parameter is dependent on how often Short Floating Point Analog Input events occur and the rate at which the events are reported to the master station. The valid values for this parameter are 0 - 65535. Default value is 16.

For SCADAPack 32 and SCADAPack 32P controllers analog input events are processed by the DNP driver at a rate of 100 events every 100 ms. If more than 100 analog input events need to be processed they are processed sequentially in blocks of 100 until all events are processed. This allows the processing of 1000 analog input events per second.

For SCADASense Series of controllers, SCADAPack 100, SCADAPack LP, SCADAPack and Micro16 controllers analog input events are processed by the DNP driver at a rate of 20 events every 100 ms. If more than 20 analog input events need to be processed they are processed sequentially in blocks of 20 until all events are processed. This allows the processing of 200 analog input events per second.

The **Word Order** selection specifies the word order of the 32-bit value. The selections are:

- **TelePACE / ISaGRAF (MSW First)** Most Significant Word in first register.
- **Reverse (LSW First)** Least Significant Word in first register.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

8.6.11.1 Adding Short Floating Point Analog Inputs

Short Floating Point Analog Inputs are added to the DNP configuration using the 16-Bit Analog Input property page. To add a Short Floating Point Analog Input:

- Select **Short Floating Point Analog Input** in the tree control section of the DNP Settings window.
- Click the **Add** button in the Short Floating Point Analog Inputs property page.
- The **Short Floating Point Analog Input** property page is now displayed.
- Edit the Short Floating Point Analog Input parameters as required and then click the **Add** button.

As Short Floating Point Analog Inputs are defined they are added as leaves to the Short Floating Point Analog Inputs branch of the tree control. When Short Floating Point Analog Inputs are defined the Short Floating Point Analog Inputs branch will display a collapse / expand control to the left of the branch. Click this control to display all defined Short Floating Point Analog Inputs.

The screenshot shows the 'DNP Settings' window. On the left is a tree control with the following structure: Application Layer, Data Link Layer, Master, Master Poll, Address Mapping, Routing, Binary Inputs, Binary Outputs, 16-bit Analog Inputs, 32-bit Analog Inputs, Short Floating Point Analog Inputs (expanded), 16-bit Analog Outputs, 32-bit Analog Outputs, Short Floating Point Analog Outputs, 16-bit Counter Inputs, and 32-bit Counter Inputs. Under 'Short Floating Point Analog Inputs', there is a sub-entry '20000 - 30200 & 30201'. The main area of the window is titled 'Short Floating Point Analog Input' and contains the following fields: 'DNP Address' with value '20000', 'Modbus Address' with value '30200', 'Class of Event Object' with a dropdown menu showing 'Class1', and 'Deadband' with value '0'. On the right side of the main area are buttons: OK, Cancel, Add, Copy, Delete, Move Up, and Move Down. At the bottom left of the window is a checkbox labeled 'Allow Duplicate Modbus Addresses' which is currently unchecked.

The Short Floating Point Analog Input parameters are described in the following paragraphs.

The **DNP Address** window displays the DNP Short Floating Point Analog Input address of the point. Each Short Floating Point Analog Input is assigned a DNP address as they are defined. The DNP point address starts at the value set in the Short Floating Point Analog Input configuration dialog and increments by one with each defined Short Floating Point Analog Input.

The **Modbus Address** parameter specifies the Modbus addresses of the Short Floating Point Analog Input assigned to the DNP Address. Short Floating Point Analog Inputs use two consecutive Modbus registers for each assigned DNP Address, the address that is entered in this box and the next consecutive Modbus register. The SCADAPack and Micro16 controllers use Modbus addressing for all analog inputs. Refer to the *I/O Database Registers* section of the **TelePACE Ladder Logic Reference and User Manual** for complete information on analog input addressing in the SCADAPack and Micro16 controllers. Valid Modbus addresses are:

- 30001 through 39998
- 40001 through 49998

The Class of Event Object parameter specifies the event object class the Short Floating Point Analog Input is assigned. If Unsolicited reporting is not required for a DNP point, it is recommended to set its Class 0 or **None**. The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Deadband** parameter specifies whether the RTU generates events. The value entered is the minimum number of counts that the Short Floating Point Analog Input must change since it was last reported. Setting this value to zero disables generating events for the Short Floating Point Analog Inputs.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click the **OK** button to accept the Short Floating Point Analog Input parameters and close the DNP Settings dialog.

Click the **Cancel** button to close the dialog without saving any changes.

Click the **Add** button to add the current Short Floating Point Analog Input to the DNP configuration.

Click the **Copy** button to copy the current Short Floating Point Analog Input parameters to the next DNP Address.

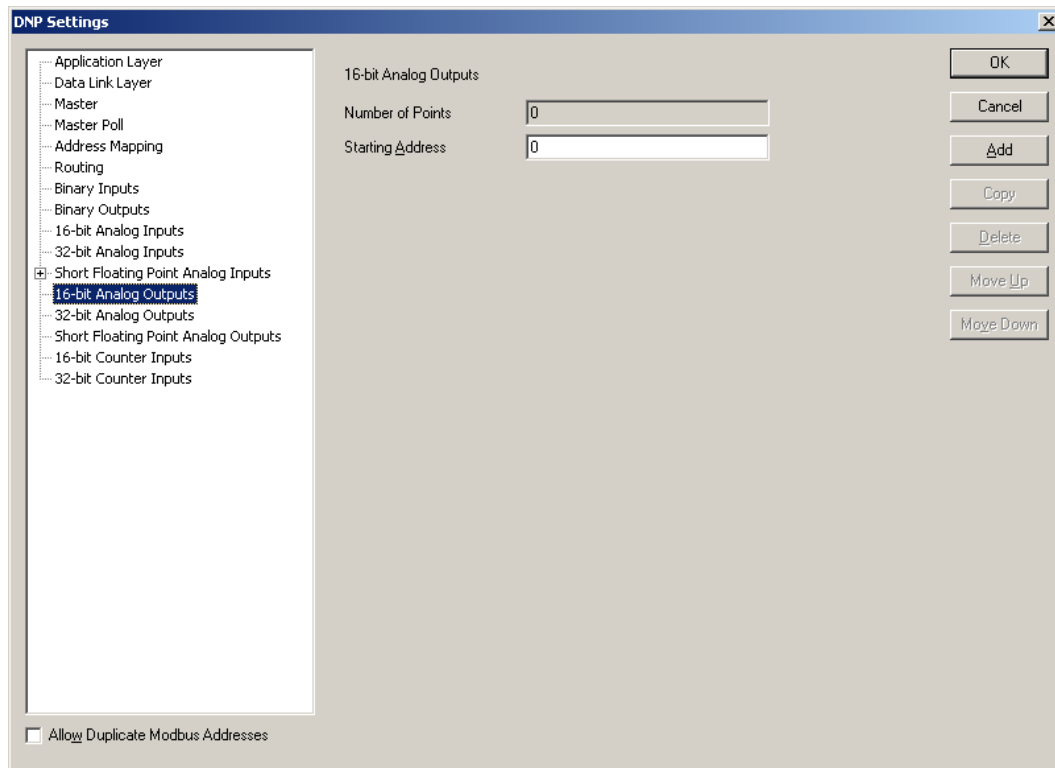
Click the **Delete** button to delete the current Short Floating Point Analog Input.

Click the **Move Up** button to move the current Short Floating Point Analog Input up one position in the tree control branch.

Click the **Move Down** button to move the current Short Floating Point Analog Input down one position in the tree control branch.

8.6.12 16-Bit Analog Outputs Configuration

The 16-Bit Analog Outputs property page is selected for editing by clicking 16-Bit Analog Outputs in the tree control section of the DNP Settings window. When selected the 16-Bit Analog Outputs property page is active.



16-Bit Analog Outputs parameters are viewed in this property page.

The **Number of Points** displays the number of 16-Bit Analog Outputs reported by this RTU. This value will increment with the addition of each configured 16-Bit Analog Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Starting Address** parameter specifies the DNP address of the first 16-bit Analog Output point.

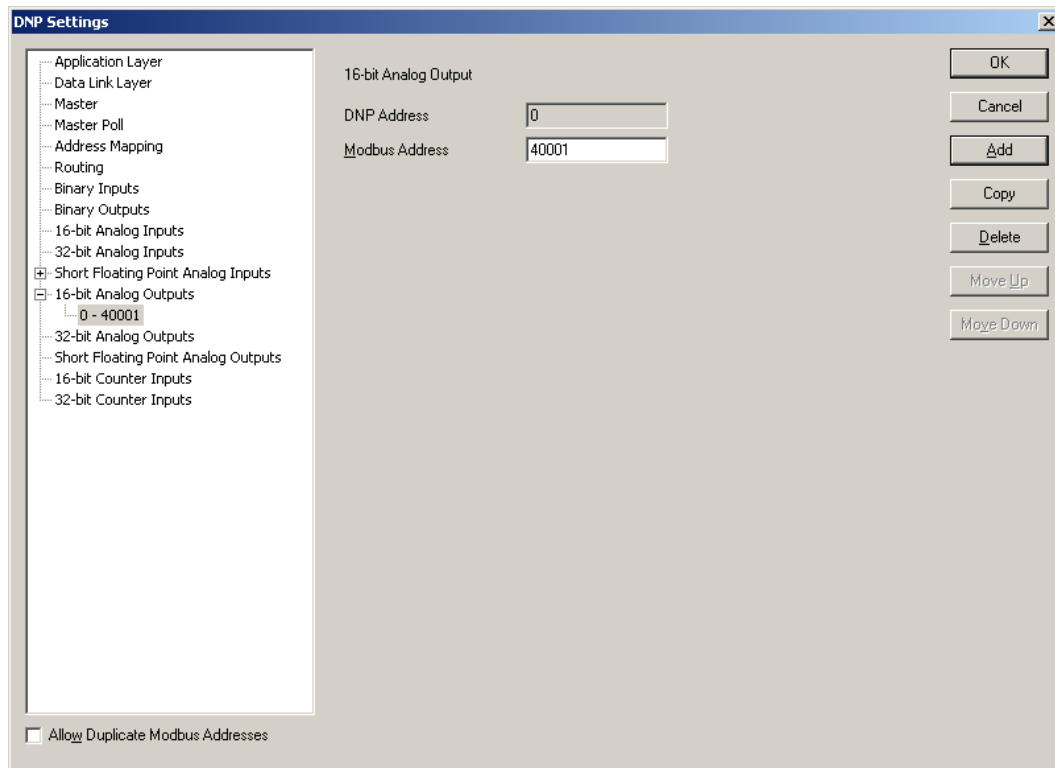
The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

8.6.12.1 Adding 16-Bit Analog Outputs

16-Bit Analog Outputs are added to the DNP configuration using the 16-Bit Analog Outputs property page. To add a 16-Bit Analog Output:

- Select **16-Bit Analog Outputs** in the tree control section of the DNP Settings window.
- Click the **Add** button in the 16-Bit Analog Outputs property page.
- The **16-Bit Analog Output** property page is now displayed.
- Edit the 16-Bit Analog Outputs parameters as required and then click the **Add** button.

As 16-Bit Analog Outputs are defined they are added as leaves to the 16-Bit Analog Output branch of the tree control. When 16-Bit Analog Outputs are defined the 16-Bit Analog Outputs branch will display a collapse / expand control to the left of the branch. Click this control to display all defined 16-Bit Analog Outputs.



The 16-Bit Analog Outputs parameters are described in the following paragraphs.

The **DNP Address** window displays the DNP 16-Bit Analog Output address of the point. Each 16-Bit Analog Output is assigned a DNP address as they are defined. The DNP point address starts at the value set in the 16-bit Analog Output configuration dialog and increments by one with each defined 16-Bit Analog Output.

The **Modbus Address** parameter specifies the Modbus address of the 16-Bit Analog Output assigned to the DNP Address. The SCADAPack and Micro16 controllers use Modbus addressing for all analog outputs. Refer to the ***I/O Database Registers*** section of the ***TelePACE Ladder Logic Reference and User Manual*** for complete information on analog output addressing in the SCADAPack and Micro16 controllers. Valid Modbus addresses are:

- 40001 through 49999

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click the **OK** button to accept the 16-Bit Analog Output parameters and close the DNP Settings dialog.

Click the **Cancel** button to close the dialog without saving any changes.

Click the **Add** button to add the current 16-Bit Analog Output to the DNP configuration.

Click the **Copy** button to copy the current 16-Bit Analog Output parameters to the next DNP Address.

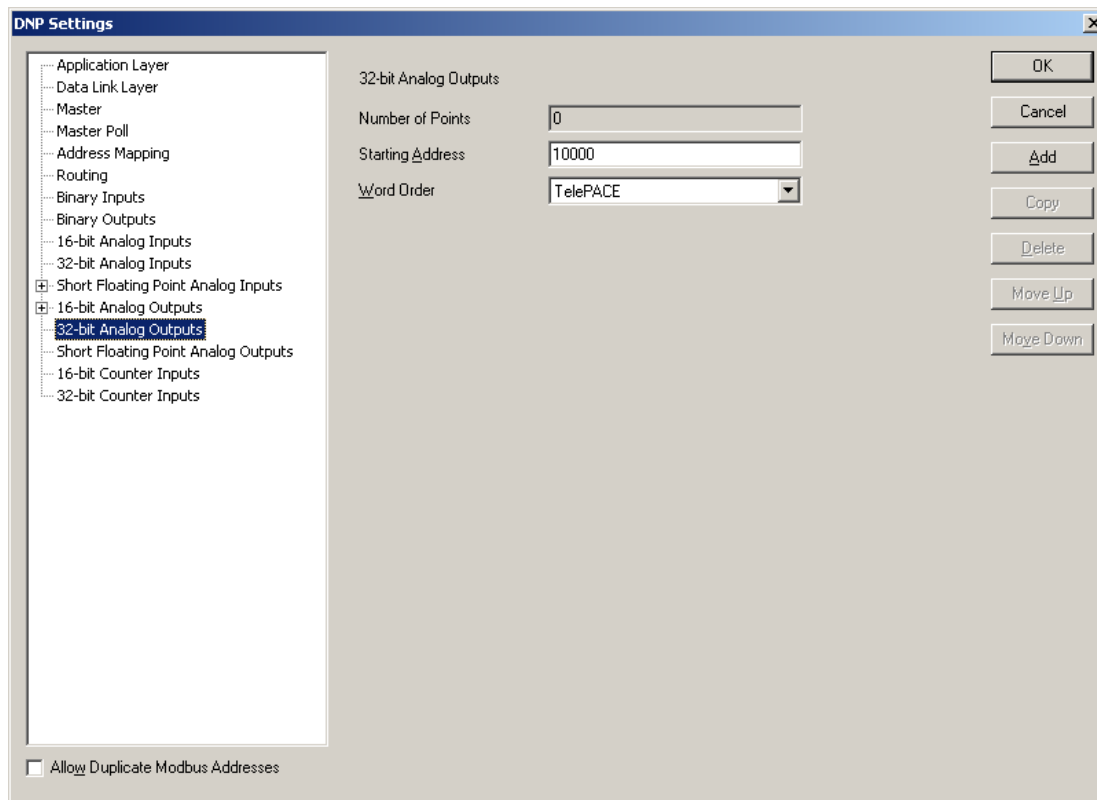
Click the **Delete** button to delete the current 16-Bit Analog Output.

Click the **Move Up** button to move the current 16-Bit Analog Output up one position in the tree control branch.

Click the **Move Down** button to move the current 16-Bit Analog Output down one position in the tree control branch.

8.6.13 32-Bit Analog Outputs Configuration

The 32-Bit Analog Outputs property page is selected for editing by clicking 32-Bit Analog Outputs in the tree control section of the DNP Settings window. When selected the 32-Bit Analog Outputs property page is active.



32-Bit Analog Outputs parameters are viewed in this property page.

The **Number of Points** displays the number of 32-Bit Analog Outputs reported by this RTU. This value will increment with the addition of each configured 32-Bit Analog Output point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Starting Address** parameter specifies the DNP address of the first 16-bit Analog Output point.

The **Word Order** selection specifies the word order of the 32-bit value. The selections are:

- **TelePACE** Least Significant Word in first register.
- **ISaGRAF** Most Significant Word in first register.

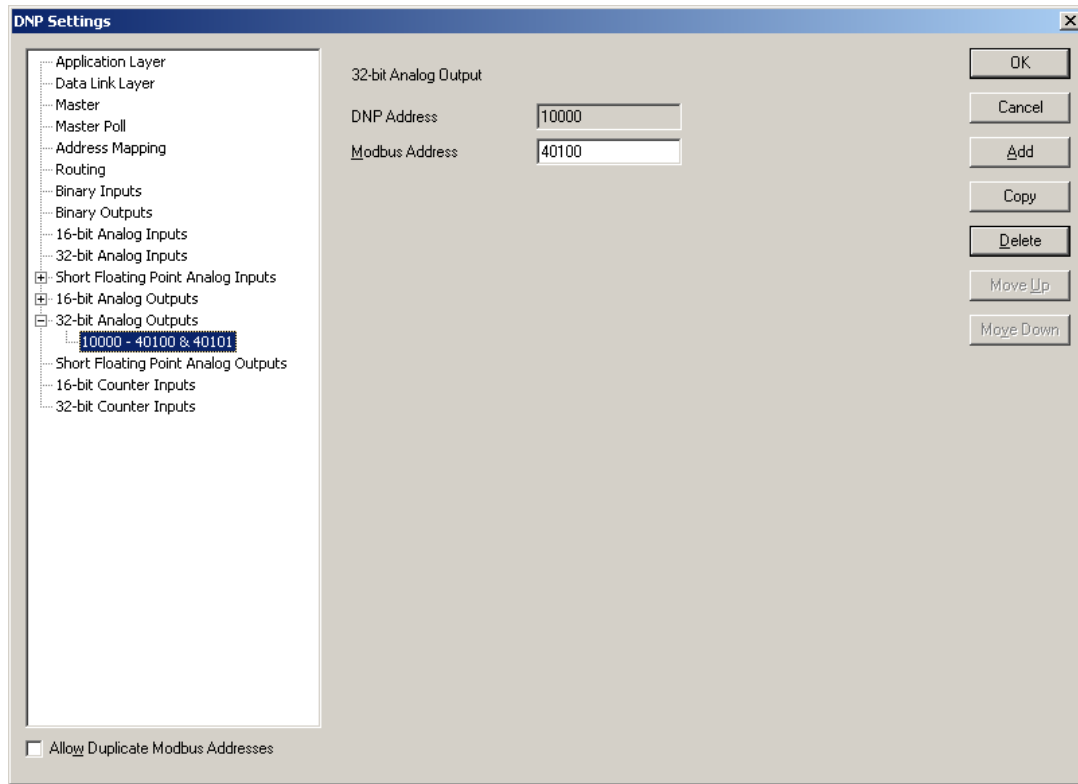
The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

8.6.13.1 Adding 32-Bit Analog Outputs

32-Bit Analog Outputs are added to the DNP configuration using the 32-Bit Analog Outputs property page. To add a 32-Bit Analog Output:

- Select **32-Bit Analog Outputs** in the tree control section of the DNP Settings window.
- Click the **Add** button in the 16-Bit Analog Outputs property page.
- The **32-Bit Analog Output** property page is now displayed.
- Edit the 32-Bit Analog Outputs parameters as required and then click the **Add** button.

As 32-Bit Analog Outputs are defined they are added as leaves to the Binary Inputs branch of the tree control. When 32-Bit Analog Outputs are defined the 32-Bit Analog Outputs branch will display a collapse / expand control to the left of the branch. Click this control to display all defined 32-Bit Analog Outputs.



The 32-Bit Analog Outputs parameters are described in the following paragraphs.

The **DNP Address** window displays the DNP 32-Bit Analog Output address of the point. Each 16-Bit Analog Output is assigned a DNP address as they are defined. The DNP point address starts at the value set in the 32-bit Analog Output configuration dialog and increments by one with each defined 32-Bit Analog Output.

The **Modbus Address** parameter specifies the Modbus address of the 32-Bit Analog Output assigned to the DNP Address. 32-Bit Analog Outputs use two consecutive Modbus registers for each assigned DNP Address, the address that is entered in this box and the next consecutive Modbus register. The SCADAPack and Micro16 controllers use Modbus addressing for all analog outputs. Refer to the **I/O Database Registers** section of the *TelePACE Ladder Logic Reference and User Manual* for complete information on analog output addressing in the SCADAPack and Micro16 controllers. Valid Modbus addresses are:

- 40001 through 49998

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click the **OK** button to accept the 16-Bit Analog Output parameters and close the DNP Settings dialog.

Click the **Cancel** button to close the dialog without saving any changes.

Click the **Add** button to add the current 32-Bit Analog Output to the DNP configuration.

Click the **Copy** button to copy the current 32-Bit Analog Output parameters to the next DNP Address.

Click the **Delete** button to delete the current 32-Bit Analog Output.

Click the **Move Up** button to move the current 32-Bit Analog Output up one position in the tree control branch.

Click the **Move Down** button to move the current 32-Bit Analog Output down one position in the tree control branch.

8.6.14 Short Floating Point Analog Outputs

The Short Floating Point Analog Outputs property page is selected for editing by clicking Short Floating Point Analog Outputs in the tree control section of the DNP Settings window. When selected the Short Floating Point Analog Outputs property page is active.

The screenshot shows the 'DNP Settings' dialog box. On the left is a tree control with the following items: Application Layer, Data Link Layer, Master, Master Poll, Address Mapping, Routing, Binary Inputs, Binary Outputs, 16-bit Analog Inputs, 32-bit Analog Inputs, Short Floating Point Analog Inputs, 16-bit Analog Outputs, 32-bit Analog Outputs, Short Floating Point Analog Outputs (highlighted), 16-bit Counter Inputs, and 32-bit Counter Inputs. The main area on the right is titled 'Short Floating Point Analog Outputs' and contains three input fields: 'Number of Points' with the value '0', 'Starting Address' with the value '20000', and 'Word Order' with a dropdown menu showing 'TelePACE/ISA GRAF (MSW First)'. To the right of these fields is a vertical stack of buttons: OK, Cancel, Add, Copy, Delete, Move Up, and Move Down. At the bottom left of the dialog is a checkbox labeled 'Allow Duplicate Modbus Addresses' which is currently unchecked.

Short Floating Point Analog Output parameters are set in this property page. Each parameter is described in the following paragraphs.

The **Number of Points** displays the number of Short Floating Point Analog Outputs reported by the RTU. This value will increment with the addition of each configured Short Floating Point Analog Input point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Starting Address** parameter specifies the DNP address of the first Short Floating Point Analog Output point.

The **Word Order** selection specifies the word order of the 32-bit value. The selections are:

- **TelePACE / ISaGRAF (MSW First)** Most Significant Word in first register.
- **Reverse (LSW First)** Least Significant Word in first register.

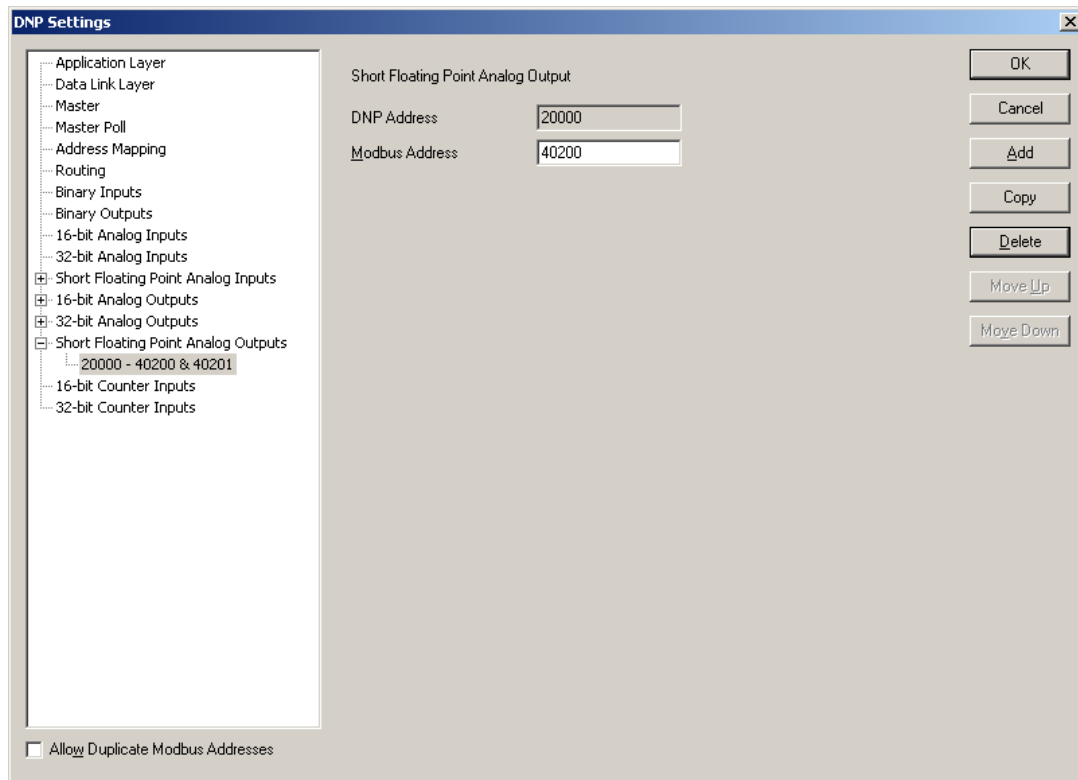
The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

8.6.14.1 Adding Short Floating Point Analog Outputs

Short Floating Point Analog Outputs are added to the DNP configuration using the Short Floating Point Analog Output property page. To add a Short Floating Point Analog Output:

- Select **Short Floating Point Analog Output** in the tree control section of the DNP Settings window.
- Click the **Add** button in the Short Floating Point Analog Inputs property page.
- The **Short Floating Point Analog Output** property page is now displayed.
- Edit the Short Floating Point Analog Output parameters as required and then click the **Add** button.

As Short Floating Point Analog Outputs are defined they are added as leaves to the Short Floating Point Analog Outputs branch of the tree control. When Short Floating Point Analog Outputs are defined the Short Floating Point Analog Outputs branch will display a collapse / expand control to the left of the branch. Click this control to display all defined Short Floating Point Analog Outputs.



The Short Floating Point Analog Output parameters are described in the following paragraphs.

The **DNP Address** window displays the DNP Short Floating Point Analog Output address of the point. Each Short Floating Point Analog Output is assigned a DNP address as they are defined. The DNP point address starts at the value set in the Short Floating Point Analog Output configuration dialog and increments by one with each defined Short Floating Point Analog Output.

The **Modbus Address** parameter specifies the Modbus addresses of the Short Floating Point Analog Output assigned to the DNP Address. Short Floating Point Analog Outputs use two consecutive Modbus registers for each assigned DNP Address, the address that is entered in this box and the next consecutive Modbus register. The SCADAPack and Micro16 controllers use Modbus addressing for all analog inputs. Refer to the ***I/O Database Registers*** section of the ***TelePACE Ladder Logic Reference and User Manual*** for complete information on analog input addressing in the SCADAPack and Micro16 controllers. Valid Modbus addresses are:

- 30001 through 39998
- 40001 through 49998

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click the **OK** button to accept the Short Floating Point Analog Input parameters and close the DNP Settings dialog.

Click the **Cancel** button to close the dialog without saving any changes.

Click the **Add** button to add the current Short Floating Point Analog Input to the DNP configuration.

Click the **Copy** button to copy the current Short Floating Point Analog Input parameters to the next DNP Address.

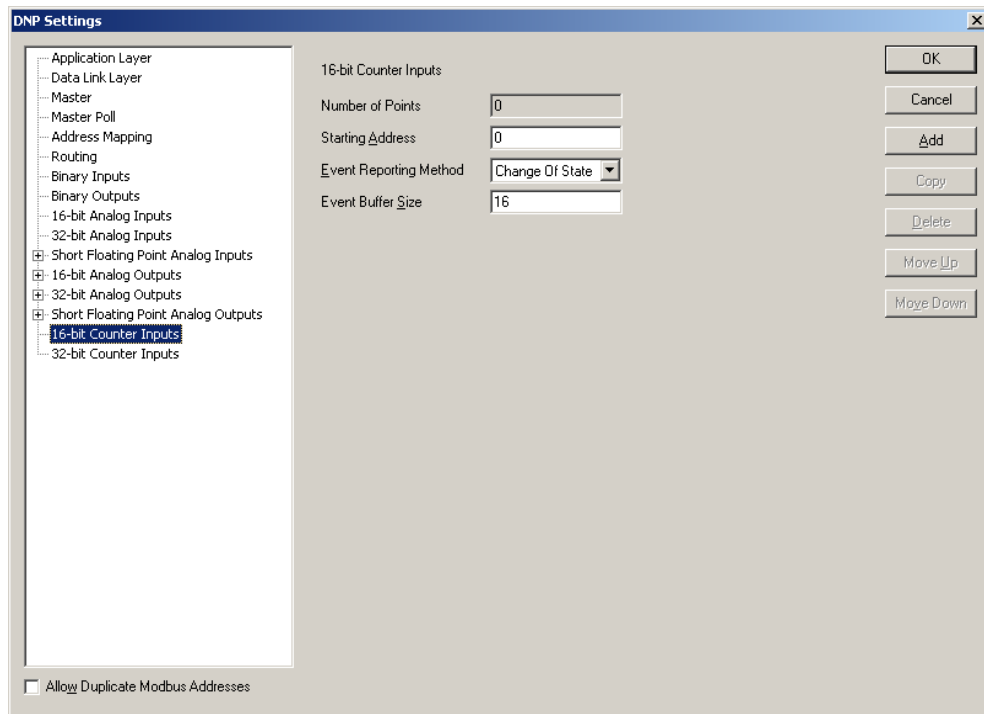
Click the **Delete** button to delete the current Short Floating Point Analog Input.

Click the **Move Up** button to move the current Short Floating Point Analog Input up one position in the tree control branch.

Click the **Move Down** button to move the current Short Floating Point Analog Input down one position in the tree control branch.

8.6.15 16-Bit Counter Inputs Configuration

The 16-Bit Counter Inputs property page is selected for editing by clicking 16-Bit Counter Inputs in the tree control section of the DNP Settings window. When selected the 16-Bit Counter Inputs property page is active.



16-Bit Counter Inputs parameters are set in this property page. Each parameter is described in the following paragraphs.

The **Number of Points** displays the number of 16-Bit Counter Inputs reported by the RTU. This value will increment with the addition of each configured 16-Bit Counter Inputs point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Starting Address** parameter specifies the DNP address of the first 16-Bit Counter Input point.

The **Event Reporting Method** selection specifies how 16-Bit Counter Input events are reported. A **Change Of State** event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data. A **Log All Events** is event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. The selections are:

- Change of State

- Log All Events

The **Event Buffer Size** parameter specifies the maximum number of 16-Bit Counter Input change without time events buffered by the RTU. The buffer holds all 16-Bit Counter Input events, regardless of the class to which they are assigned. If the buffer fills to 90 percent the RTU will send a buffer overflow event to the master station. If the buffer is completely full the RTU will lose the oldest events and retain the newest. The Event Buffer size should be at least equivalent to the number of 16-Bit Analog Inputs defined as Change of State type. That will allow all 16-Bit Counter Inputs to exceed the threshold simultaneously without losing any events. The value of this parameter is dependent on how often 16-Bit Counter Input events occur and the rate at which the events are reported to the master station. The valid values for this parameter are 0 - 65535. Default value is 16.

For SCADAPack 32 and SCADAPack 32P controllers counter input events are processed by the DNP driver at a rate of 100 events every 100 ms. If more than 100 counter input events need to be processed they are processed sequentially in blocks of 100 until all events are processed. This allows the processing of 1000 counter input events per second.

For SCADASense Series of controllers, SCADAPack 100, SCADAPack LP, SCADAPack and Micro16 controllers counter input events are processed by the DNP driver at a rate of 20 events every 100 ms. If more than 20 counter input events need to be processed they are processed sequentially in blocks of 20 until all events are processed. This allows the processing of 200 counter input events per second.

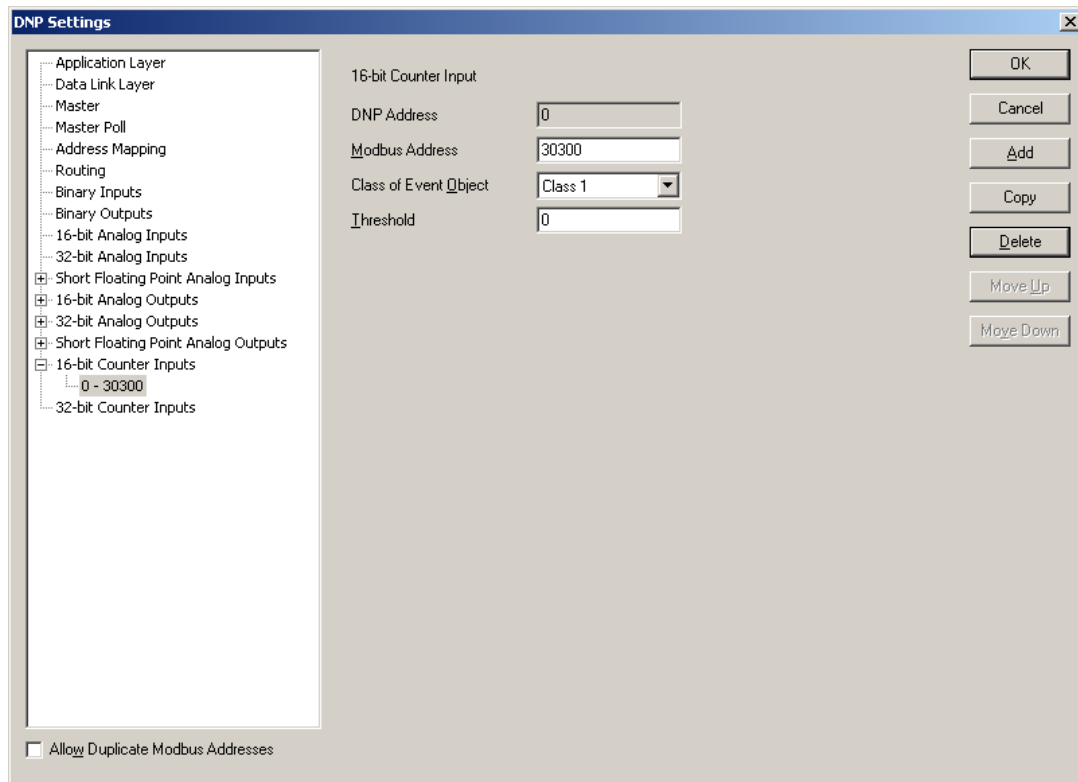
The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

8.6.15.1 Adding 16-Bit Counter Inputs

16-Bit Counter Inputs are added to the DNP configuration using the 16-Bit Counter Inputs property page. To add a 16-Bit Counter Input:

- Select **16-Bit Counter Inputs** in the tree control section of the DNP Settings window.
- Click the **Add** button in the 16-Bit Counter Inputs property page.
- The **16-Bit Counter Input** property page is now displayed.
- Edit the 16-Bit Counter Inputs parameters as required and then click the **Add** button.

As 16-Bit Counter Inputs are defined they are added as leaves to the 16-Bit Counter Inputs branch of the tree control. When 16-Bit Counter Inputs are defined the 16-Bit Counter Inputs branch will display a collapse / expand control to the left of the branch. Click this control to display all defined 16-Bit Counter Inputs.



The 16-Bit Counter Input parameters are described in the following paragraphs.

The **DNP Address** window displays the DNP 16-Bit Counter Input address of the point. Each 16-Bit Counter Input is assigned a DNP address as they are defined. The DNP point address starts at the value set in the 16-Bit Counter Input configuration dialog and increments by one with each defined 16-Bit Counter Input.

The **Modbus Address** parameter specifies the Modbus address of the 16-Bit Counter Input assigned to the DNP Address. The SCADAPack and Micro16 controllers use Modbus addressing for all counter inputs. Refer to the *I/O Database Registers* section of the **TelePACE Ladder Logic Reference and User Manual** for complete information on analog input addressing in the SCADAPack and Micro16 controllers. Valid Modbus addresses are:

- 30001 through 39999
- 40001 through 49999

The **Class of Event Object** parameter specifies the event object class the 16-Bit Counter Input is assigned. If Unsolicited reporting is not required for a DNP point, it is recommended to set its Class 0 or **None**. The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Threshold** parameter specifies whether the RTU generates events. The value entered is the minimum number of counts that the 16-Bit Counter Input must change since it was last reported.

Setting this value to zero disables generating events for the 16-Bit Counter Input point. Valid deadband values are 0 to 65535.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click the **OK** button to accept the 16-Bit Analog Counter parameters and close the DNP Settings dialog.

Click the **Cancel** button to close the dialog without saving any changes.

Click the **Add** button to add the current 16-Bit Analog Input to the DNP configuration.

Click the **Copy** button to copy the current 16-Bit Analog Input parameters to the next DNP Address.

Click the **Delete** button to delete the current 16-Bit Analog Input.

Click the **Move Up** button to move the current 16-Bit Analog Input up one position in the tree control branch.

Click the **Move Down** button to move the current 16-Bit Analog Input down one position in the tree control branch.

8.6.16 32-Bit Counter Inputs Configuration

The 32-Bit Counter Inputs property page is selected for editing by clicking 32-Bit Counter Inputs in the tree control section of the DNP Settings window. When selected the 32-Bit Counter Inputs property page is active.

The screenshot shows the 'DNP Settings' dialog box with the '32-bit Counter Inputs' configuration page selected. The left pane shows a tree view with '32-bit Counter Inputs' highlighted. The right pane contains the following fields and buttons:

- 32-bit Counter Inputs** (Section Header)
- Number of Points**: Text box containing '0'.
- Starting Address**: Text box containing '10000'.
- Event Reporting Method**: Dropdown menu set to 'Change Of State'.
- Event Buffer Size**: Text box containing '16'.
- Word Order**: Dropdown menu set to 'TelePACE'.
- Buttons**: OK, Cancel, Add, Copy, Delete, Move Up, Move Down.
- Checkbox**: ☐ Allow Duplicate Modbus Addresses.

32-Bit Counter Inputs parameters are set in this property page. Each parameter is described in the following paragraphs.

The **Number of Points** displays the number of 32-Bit Counter Inputs reported by the RTU. This value will increment with the addition of each configured 32-Bit Counter Inputs point. The maximum number of points is 9999. The maximum number of actual points will depend on the memory available in the controller.

The **Starting Address** parameter specifies the DNP address of the first 32-Bit Counter Input point.

The **Event Reporting Method** selection specifies how 32-Bit Counter Input events are reported. A **Change Of State** event is an event object, without time, that is generated when the point changes state. Only one event is retained in the buffer for each point. If a subsequent event occurs for a point, the previous event object will be overwritten. The main purpose of this mode is to allow a master station to efficiently poll for changed data. A **Log All Events** is event object with absolute time will be generated when the point changes state. All events will be retained. The main purpose of this mode is to allow a master station to obtain a complete historical data log. The selections are:

- Change of State
- Log All Events

The **Event Buffer Size** parameter specifies the maximum number of 32-Bit Counter Input change events buffered by the RTU. The buffer holds all 32-Bit Counter Input events, regardless of the class to which they are assigned. If the buffer is completely full the RTU will lose the oldest events and retain the newest; the 'Event Buffer Overflowed' IIN flag will also be set to indicate that the buffer has overflowed. The Event Buffer size should be at least equivalent to the number of 32-Bit Counter Inputs defined as Change of State type. That will allow all 32-Bit Counter Inputs to exceed the deadband simultaneously without losing any events. The value of this parameter is dependent on how often 32-Bit Counter Input events occur and the rate at which the events are reported to the master station. The valid values for this parameter are 0 - 65535. Default value is 16.

For SCADAPack 32 and SCADAPack 32P controllers counter input events are processed by the DNP driver at a rate of 100 events every 100 ms. If more than 100 counter input events need to be processed they are processed sequentially in blocks of 100 until all events are processed. This allows the processing of 1000 counter input events per second.

For SCADASense Series of controllers, SCADAPack 100, SCADAPack LP, SCADAPack and Micro16 controllers counter input events are processed by the DNP driver at a rate of 20 events every 100 ms. If more than 20 counter input events need to be processed they are processed sequentially in blocks of 20 until all events are processed. This allows the processing of 200 counter input events per second.

The **Word Order** selection specifies the word order of the 32-bit value. The selections are:

- **TelePACE** Least Significant Word in first register.
- **ISaGRAF** Most Significant Word in first register.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

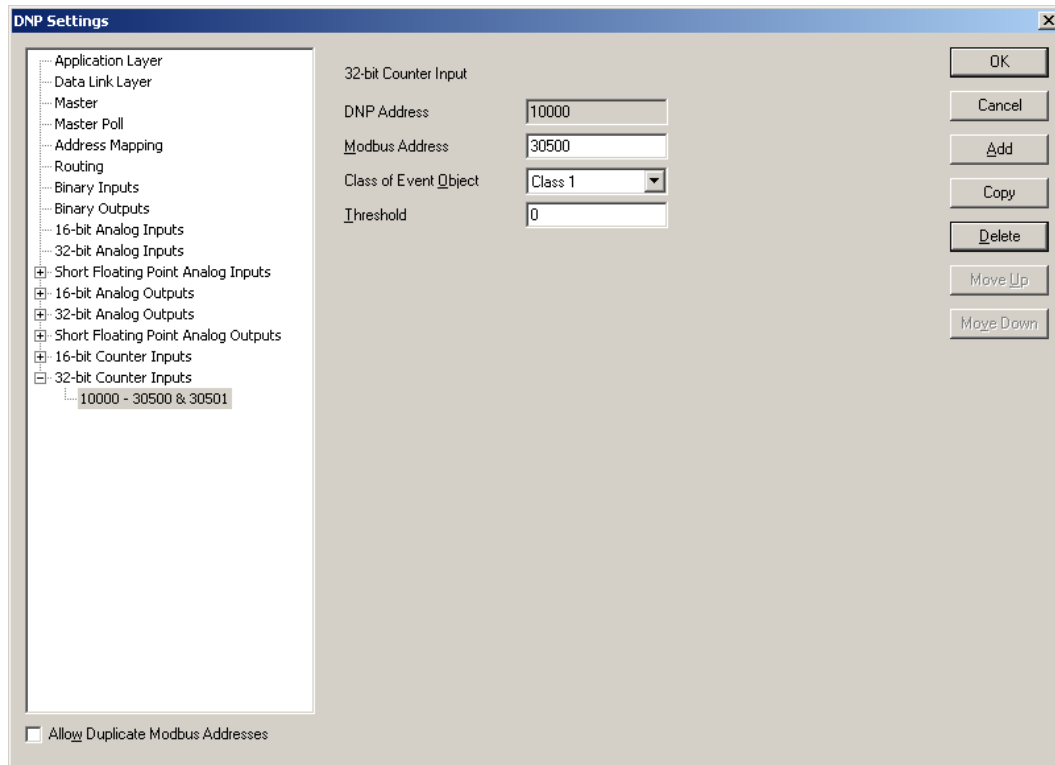
8.6.16.1 Adding 32-Bit Counter Inputs

32-Bit Counter Inputs are added to the DNP configuration using the 16-Bit Counter Input property page. To add a 32-Bit Analog Input:

- Select **32-Bit Counter Inputs** in the tree control section of the DNP Settings window.
- Click the **Add** button in the 32-Bit Counter Inputs property page.

- The **32-Bit Counter Input** property page is now displayed.
- Edit the 32-Bit Counter Input parameters as required and then click the **Add** button.

As 32-Bit Counter Inputs are defined they are added as leaves to the 32-Bit Counter Inputs branch of the tree control. When 32-Bit Counter Inputs are defined the 32-Bit Counter Inputs branch will display a collapse / expand control to the left of the branch. Click this control to display all defined 32-Bit Counter Inputs.



The 32-Bit Counter Input parameters are described in the following paragraphs.

The **DNP Address** window displays the DNP 32-Bit Counter Input address of the point. Each 32-Bit Counter Input is assigned a DNP address as they are defined. The DNP point address starts at the value set in the 32-Bit Counter Input configuration dialog and increments by one with each defined 32-Bit Counter Input.

The **Modbus Address** parameter specifies the Modbus addresses of the 32-Bit Counter Input assigned to the DNP Address. 32-Bit Counter Inputs use two consecutive Modbus registers for each assigned DNP Address, the address that is entered in this box and the next consecutive Modbus register. The SCADAPack and Micro16 controllers use Modbus addressing for all counter inputs. Refer to the *I/O Database Registers* section of the **TelePACE Ladder Logic Reference and User Manual** for complete information on analog input addressing in the SCADAPack and Micro16 controllers. Valid Modbus addresses are:

- 30001 through 39998
- 40001 through 49998

The **Class of Event Object** parameter specifies the event object class the 32-Bit Counter Input is assigned. If Unsolicited reporting is not required for a DNP point, it is recommended to set its Class 0 or **None**. The selections are:

- None
- Class 1
- Class 2
- Class 3

The **Threshold** parameter specifies whether the RTU generates events. The value entered is the minimum number of counts that the 32-Bit Counter Input must change since it was last reported. Setting this value to zero disables generating events for the 32-Bit Counter Input point. Valid threshold values are 0 to 4,294,967,295.

The **Allow Duplicate Modbus Addresses** checkbox determines if the Modbus I/O database addresses assigned to the DNP data points must be unique. Check this box if you want to allow more than one point to use the same Modbus address.

Click the **OK** button to accept the 32-Bit Counter Input parameters and close the DNP Settings dialog.

Click the **Cancel** button to close the dialog without saving any changes.

Click the **Add** button to add the current 32-Bit Counter Input to the DNP configuration.

Click the **Copy** button to copy the current 32-Bit Counter Input parameters to the next DNP Address.

Click the **Delete** button to delete the current 32-Bit Counter Input.

Click the **Move Up** button to move the current 32-Bit Counter Input up one position in the tree control branch.

Click the **Move Down** button to move the current 32-Bit Counter Input down one position in the tree control branch.

8.7 DNP Diagnostics

DNP Diagnostics provide Master station and Outstation DNP diagnostics. The diagnostics provide detailed information on the status of DNP communication and DNP data points. This information is useful when debugging DNP station and network problems that may arise.

DNP diagnostics are available for local DNP information using the **DNP Status** command.

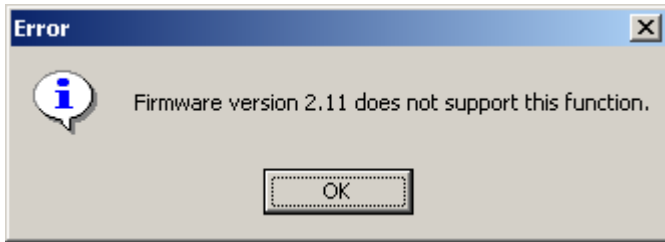
- For TelePACE applications select **Controller >> DNP Status** from the menu bar. See Chapter 8.7 for information on DNP Status diagnostics.
- For ISaGRAF applications select **Tools >> Controller >> DNP Status** from the program window menu bar.

SCADAPack 32 controllers support DNP master operations. DNP diagnostics are available for master stations using the **DNP Master Status** command.

- For TelePACE applications select **Controller >> DNP Master Status** from the menu bar. See section 8.7.2 for information on DNP Master Status diagnostics.
- For ISaGRAF applications select **Tools >> Controller >> DNP Master Status** from the program window menu bar.

DNP Diagnostics require firmware version 2.20 or newer for SCADAPack controllers and firmware version 1.50 or newer for SCADAPack 32 controllers. When an attempt is made to select the DNP

Status or DNP Master Status command for controllers with firmware that does not support the commands an error message is displayed. An example of the error message is shown below.



To enable the use of DNP diagnostics you will need to upgrade the firmware in the controller to the newer version.

8.7.1 DNP Status

When the DNP Status command is selected the DNP Status dialog is displayed. This dialog shows the run-time DNP diagnostics and current data values for the local DNP points.

The DNP Status dialog has a number of selectable tabs and opens with the Overview tab selected. The following tabs are displayed.

- Overview
- Binary In (binary inputs information)
- Binary Out (binary outputs information)
- AIN-16 (16-bit analog inputs information)
- AIN-32 (32-bit analog inputs information)
- AIN-Float (short float analog inputs information)
- AOUT-16 (16-bit analog outputs information)
- AOUT-32 (32-bit analog outputs information)
- AOUT-Float (short float analog outputs information)
- Counter-16 (16-bit counter inputs information)
- Counter-32 (32-bit counter inputs information)

Clicking on any tab opens the tab and displays the selected information.

8.7.1.1 Overview Tab

The Overview Tab displays the run-time diagnostics for the local DNP station. The Overview display is divided into five areas of diagnostic information: DNP Status, Internal Indications, Communication Statistics, Last Message and Event Buffer. Each of these is explained in the following paragraphs.

DNP Status

Overview | Binary In | Binary Out | AIN-16 | AIN-32 | AIN-Float | AOUT-16 | AOUT-32 | AOUT-Float | Counter-16 | Counter-32

DNP Status: 7: enabled, configured, running

Internal Indications: 8000: Restart,

Communication Statistics

Direction	com1	com2	com3	com4	IP
Transmit	692	0	0	0	361
Receive	0	0	0	0	269
Successes	0	0	0	0	265
Fails	839	0	0	0	92
FailsNew	839	0	0	0	0

Reset

Last Message

Direction	Time	Port	Source	Dest	Length	Link Func	Appl Func	IIN
Transmit	16Jan06 17:40:01.57	TCP	1	246	11	Send/No Reply	Read	
Receive	16Jan06 17:40:01.64	TCP	246	1	174	Send/No Reply	Response	0000

Event Buffers

Binary In	AIN-16	AIN-32	AIN-Float	Counter-16	Counter-32	Class 1	Class 2	Class 3
0/16	0/16	0/16	0/16	0/16	0/16	0	0	0

Close

The **DNP Status** window provides information on the status of the DNP protocol running in the controller. Depending on the status the window may contain the following text.

- **Enabled** or **Disabled** indicates whether the controller firmware supports DNP protocol.
- **Configured** or **Not Configured** indicates whether the controller has been configured with DNP protocol on at least one communications port.
- **Running** or **Not Running** indicates whether the DNP tasks are running in the controller.

The **Internal Indications** window displays the current state of the DNP internal indications (IIN) flags in the controller. For a detailed description of the IIN flags see the section [8.1.1.1.1](#) of this manual. Note that bits 0 – 7 (the first octet) are displayed on the left, then bits 8 - 15 (second octet) on the right.

The **Communication Statistics** window displays the message statistics for each DNP communication port. The statistics include the total number of messages transmitted and received and the total number of successes, failures, and failures since last success (which will only be updated for messages sent by this controller) for each communication port. The counters increment whenever a new DNP message is sent or received on the port, and roll over after 65535 messages.

- Click the **Reset** button to reset the counters to zero.

The **Last Message** window displays information about the most recent DNP message. The information is updated each time a new message is received or transmitted. The Last Message window contains the following information.

- **Direction** displays whether the message was received or transmitted.
- **Time** displays the time at which the message was received or sent.

- **Port** displays which communication port was used for the message.
- **Source** displays the source DNP station address for the message.
- **Dest** displays the destination DNP station address for the message.
- **Length** displays the message length in bytes.
- **Link Func** displays the Link Layer function code.
- **Appl Func** displays the Application Layer function code.
- **IIN** displays the Internal indications received with the last message

The **Event Buffers** window displays the number of events in each type of event buffer and the allocated buffer size. The event buffers displayed are:

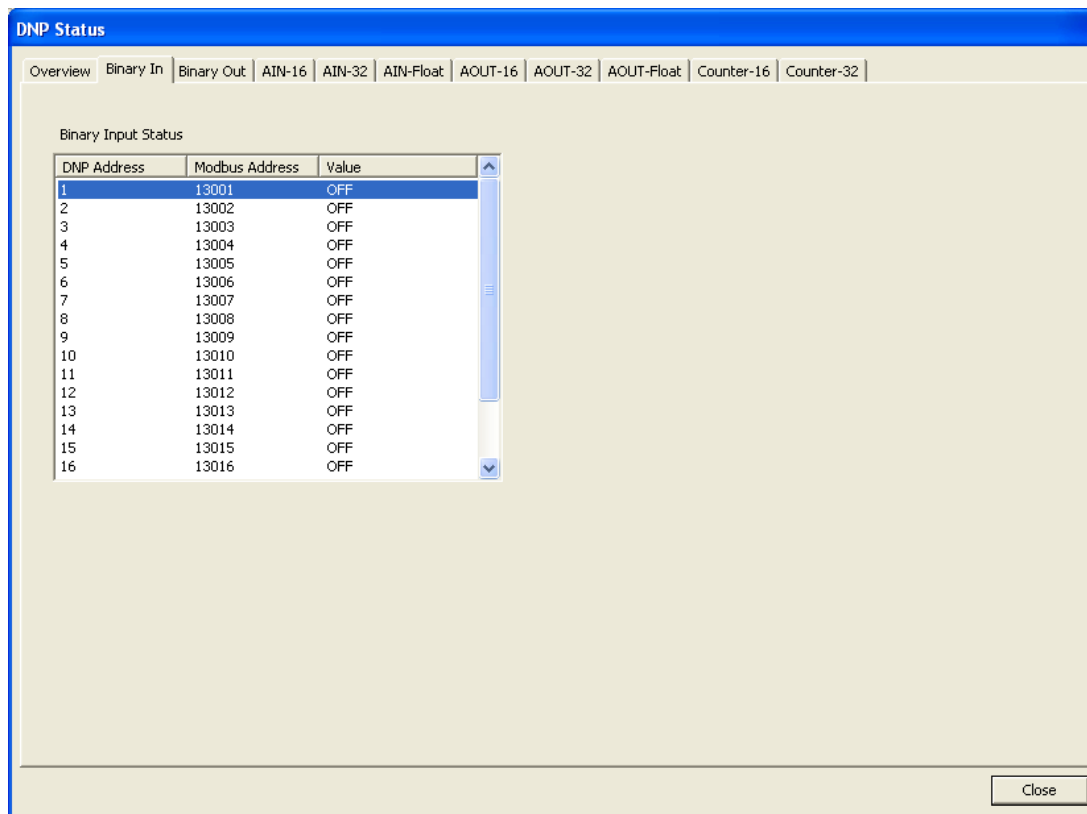
- Binary In (binary inputs)
- AIN-16 (16-bit analog inputs)
- AIN-32 (32-bit analog inputs)
- AIN-Float (floating point analog inputs)
- Counter-16 (16-bit counter inputs)
- Counter-32 (32-bit counter inputs)
- Class 1 (class 1 events)
- Class 2 (class 2 events)
- Class 3 (class 3 events)

8.7.1.2 Point Status Tabs

The point status tabs display the state of each point of the selected type in the controller. The following tabs are displayed.

- Binary In (binary inputs information)
- Binary Out (binary outputs information)
- AIN-16 (16-bit analog inputs information)
- AIN-32 (32-bit analog inputs information)
- AIN-Float (short float analog inputs information)
- AOUT-16 (16-bit analog outputs information)
- AOUT-32 (32-bit analog outputs information)
- AOUT-Float (short float analog outputs information)
- Counter-16 (16-bit counter inputs information)
- Counter-32 (32-bit counter inputs information)

Each of the tabs displays information in the same format. The example below shows the appearance of the binary input page.



The **DNP Address** column shows the DNP address of the point.

The **Modbus Address** column shows the Modbus register address of the point.

The **Value** column shows the value of the point. Binary points are shown as OFF or ON. Numeric points show the numeric value of the point.

8.7.2 DNP Master Status

When the DNP Master Status command is selected the **DNP Master Status** dialog is displayed. This dialog shows the run-time DNP diagnostics and status of the DNP outstations and current data values for the DNP points in these outstations.

The DNP Master Status dialog has a number of selectable tabs and opens with the All Stations tab selected. The following tabs are displayed.

- All Stations
- Remote Overview
- Binary In (binary inputs information)
- Binary Out (binary outputs information)
- AIN-16 (16-bit analog inputs information)
- AIN-32 (32-bit analog inputs information)
- AIN-Float (short float analog inputs information)
- AOUT-16 (16-bit analog outputs information)
- AOUT-32 (32-bit analog outputs information)

- AOUT-Float (short float analog outputs information)
- Counter-16 (16-bit counter inputs information)
- Counter-32 (32-bit counter inputs information)

8.7.2.1 All Stations Tab

The **All Stations** tab displays the run-time communications diagnostics for all outstations polled by the master or outstations reporting unsolicited data to the master.

DNP Address	Successes	Fails	FailsNew	Msgs Rx	Last Rx Msg Time	Msgs Tx	Last Tx Msg Time
2	0	27	27	0	---	20	16Jan06 16:49:55.40
3	0	25	25	0	---	23	16Jan06 16:49:56.90
6	0	26	26	0	---	22	16Jan06 16:49:58.40
246	29	17	0	29	16Jan06 16:50:00.02	42	16Jan06 16:50:00.10
247	7	13	0	7	16Jan06 16:49:54.53	11	16Jan06 16:49:54.50

The **Communication Statistics** window displays a list of all outstations and the communication statistics for each station in the list. The statistics counters increment whenever a new DNP message is sent or received, and roll over after 65535 messages. The following statistics are displayed.

- **DNP Address** displays the DNP address of the outstation.
- **Successes** display the number of successful message transactions between this master and the corresponding remote station. This number includes master polls to the remote station and unsolicited responses from the outstation.
- **Fails** displays the number of failed message transactions between this master and the corresponding remote station. This counter increments by 1 for a failed message transaction irrespective of the number of application layer retries.
- **FailsNew** displays the number failed message transactions between this master and the corresponding remote station since the last successful poll.
- **Msgs Rx** displays the number of DNP packets (frames) received from the outstation station. This number includes frames containing unsolicited responses from the outstation.
- **Last Rx Msg Time** displays the time the last DNP packet (frame) was received from the outstation.
- **Msgs Tx** displays the number of DNP packets (frames) sent to the outstation.
- **Last Tx Msg Time** displays the time the last DNP packet (frame) was sent to the outstation.

Note: The **Msgs Tx** and **Msgs Rx** counters could be greater than or equal to the Successes and Fails counters.

8.7.2.2 Remote Overview Tab

The Remote Overview tab displays the run-time diagnostics and current data values for a selected remote station. The data shown is from the image of the data in the master station.

DNP Master Status

All Stations | Remote Overview | Binary In | Binary Out | AIN-16 | AIN-32 | AIN-Float | AOUT-16 | AOUT-32 | AOUT-Float | Counter-16 | Counter-32

Remote Station: 247

Internal Indications: 0000

Communication Statistics

Successes	Fails	FailsNew	Msgs Rx	Last Rx Msg Time	Msgs Tx	Last Tx Msg Time
14	14	0	15	16Jan06 16:51:06.73	19	16Jan06 16:51:06.69

Reset

Event Buffers

Binary In	AIN-16	AIN-32	AIN-Float	Counter-16	Counter-32	Class 1	Class 2	Class 3
0/16	0/16	0/16	0/16	0/16	0/16	0	0	0

Close

The **Remote Station** window is where the DNP address of the remote station is entered. When the Remote station field is changed all data fields on this tab and the following I/O tabs are updated with the values for the newly selected Remote Station.

The **Internal Indications** window displays the current state of the DNP internal indications (IIN) flags for the selected remote station. For a detailed description of the IIN flags see the section [8.1.1.1.1](#) of this manual.

The **Communication Statistics** window displays communication statistics for the remote station selected. The statistics counters increment whenever a new DNP message is sent or received, and roll over after 65535 messages. The following statistics are displayed.

- **Successes** displays the number of successful messages received in response to master polls sent to the station. This number includes unsolicited responses from the outstation.
- **Fails** displays the number of failed or no responses to master polls sent to the outstation.
- **FailsNew** displays the number failed or no responses to master polls sent to the outstation since the last successful poll.
- **Msgs Rx** displays the number of messages received from the outstation station. This number includes unsolicited responses from the outstation.
- **Last Rx Msg Time** displays the time the last message was received from the outstation.
- **Msgs Tx** displays the number of messages sent to the outstation station.

- **Last Tx Msg Time** displays the time the last message was sent to the outstation.

Click **Reset** to reset the counters to zero.

Event Buffers shows the number of events in each type of event buffer and the allocated buffer size. The buffers shown are for binary inputs, 16-bit analog inputs, 32-bit analog inputs, Floating point analog inputs, 16-bit counter inputs, and 32-bit counter inputs, and Class 1, 2, and 3 events.

The **Event Buffers** window displays the number of events in each type of event buffer and the allocated buffer size for the selected remote station. The event buffers displayed are:

- Binary In (binary inputs)
- AIN-16 (16-bit analog inputs)
- AIN-32 (32-bit analog inputs)
- AIN-Float (floating point analog inputs)
- Counter-16 (16-bit counter inputs)
- Counter-32 (32-bit counter inputs)
- Class 1 (class 1 events)
- Class 2 (class 2 events)
- Class 3 (class 3 events)

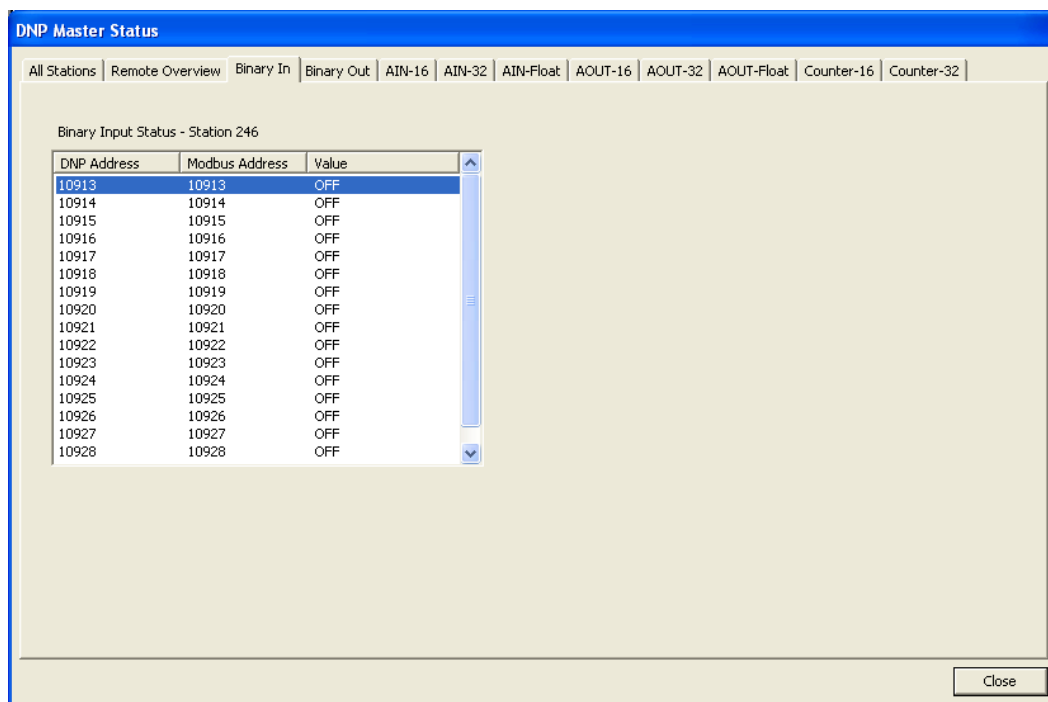
Note: Due to a limitation of the DNP3 protocol, an Unsolicited message from an outstation is not capable of including information stating which data class generated the message. As a result, all Unsolicited events when received by the master will be counted as Class 1 events. Events which are polled by the master, however, do contain class information and will be counted in the Event Buffer for the appropriate class.

8.7.2.3 Remote Point Status Tabs

The point status tabs show the state of each point of the selected type in the remote station selected on the Remote Overview tab. The values shown are from the image of the remote station in the master station.

Note: Class 0 polling of an outstation must be enabled in the master in order to allow that outstation's DNP points to be listed on these tabs. This is the only way for the master to retrieve a complete list of all points in an outstation.

The example below shows the appearance of the Binary In tab.



The **DNP Address** column shows the DNP address of the point.

The **Modbus Address** column shows the Modbus register address of the point. This is only relevant for points that have an address mapping in the master station. For points that have an address mapping, this will show the Modbus register address of the point. For points which do not have an address mapping, this will show '---'.

The **Value** column shows the value of the point. Binary points are shows as OFF or ON. Numeric points show the numeric value of the point.

8.8 DNP Master Device Profile Document

DNP v3.00 DEVICE PROFILE DOCUMENT	
Vendor Name: Control Microsystems Inc.	
Device Name: SCADAPack controllers	
Highest DNP Level Supported: For Requests 2 For Responses 2	Device Function: ☒ Master ☐ Slave

Notable objects, functions, and/or qualifiers supported in addition to the Highest DNP Levels Supported (the complete list is described in the attached table):

- Function code 14 (warm restart)
- Function code 20 (Enable Unsolicited Messages) for class 1, 2, 3 objects only.
- Function code 21 (Disable Unsolicited Messages) for class 1, 2, 3 objects only.
- Object 41, variation 1 (32-bit analog output block)

Maximum Data Link Frame Size (octets):

Transmitted 292
Received (must be 292)

Maximum Application Fragment Size (octets):

Transmitted 2048
Received 2048

Maximum Data Link Re-tries:

- ☐ None
☐ Fixed at
☒ Configurable, range 0 to 255

Maximum Application Layer Re-tries:

- ☐ None
☒ Configurable, range 0 to 255

Requires Data Link Layer Confirmation:

- ☐ Never
☐ Always
☐ Sometimes If 'Sometimes', when?

☒ Configurable for Always or Never

Requires Application Layer Confirmation:

- ☐ Never
☐ Always (not recommended)
☐ When reporting Event Data (Slave devices only)

☐ When sending multi-fragment responses (Slave devices only)

☐ Sometimes If 'Sometimes', when?

☒ Configurable for always or only when Reporting Event Data and Unsolicited Messages

Timeouts while waiting for:

Data Link Confirm	☐ None	☐ Fixed at _____	☐ Variable	☒ Configurable
Complete Appl. Fragment	☐ None	☐ Fixed at _____	☐ Variable	☒ Configurable
Application Confirm	☐ None	☐ Fixed at _____	☐ Variable	☒ Configurable
Complete Appl. Response	☐ None	☐ Fixed at _____	☐ Variable	☒ Configurable

Others _____

Sends/Executes Control Operations:

WRITE Binary Outputs	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
SELECT/OPERATE	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
DIRECT OPERATE	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
DIRECT OPERATE - NO ACK	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Count > 1	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Pulse On	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Pulse Off	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Latch On	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Latch Off	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Queue	☒ Never	☐ Always	☐ Sometimes	☐ Configurable
Clear Queue	☒ Never	☐ Always	☐ Sometimes	☐ Configurable

FILL OUT THE FOLLOWING ITEM FOR MASTER DEVICES ONLY:

Expects Binary Input Change Events:

- ☒ Either time-tagged or non-time-tagged for a single event
- ☐ Both time-tagged and non-time-tagged for a single event
- ☐ Configurable (attach explanation)

FILL OUT THE FOLLOWING ITEMS FOR SLAVE DEVICES ONLY:	
Reports Binary Input Change Events when no specific variation requested: ☐ Never ☐ Only time-tagged ☐ Only non-time-tagged ☐ Configurable to send both, one or the other (attach explanation)	Reports time-tagged Binary Input Change Events when no specific variation requested: ☐ Never ☐ Binary Input Change With Time ☐ Binary Input Change With Relative Time ☐ Configurable (attach explanation)
Sends Unsolicited Responses: ☐ Never ☐ Configurable by class ☐ Only certain objects ☐ Sometimes (attach explanation) ☒ ENABLE/DISABLE UNSOLICITED	Sends Static Data in Unsolicited Responses: ☐ Never ☐ When Device Restarts ☐ When Status Flags Change No other options are permitted.
Default Counter Object/Variation: ☐ No Counters Reported ☐ Configurable (attach explanation) ☐ Default Object 20 Default Variation 05 ☐ Point-by-point list attached	Counters Roll Over at: ☐ No Counters Reported ☐ Configurable (attach explanation) ☐ 16 Bits ☐ 32 Bits ☐ 16 Bits for 16-bit counters 32 Bits for 32-bit counters ☐ Point-by-point list attached
Sends Multi-Fragment Responses: ☒ Yes ☐ No	

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
1	0	Binary Input - All Variations	1	06		
1	1	Binary Input			129, 130	00, 01
1	2	Binary Input with Status			129, 130	00, 01
2	0	Binary Input Change - All Variations	1	06,07,08		
2	1	Binary Input Change without Time	1	06,07,08	129, 130	17, 28
2	2	Binary Input Change with Time	1	06,07,08	129, 130	17, 28
2	3	Binary Input Change with Relative Time	1	06,07,08	129, 130	17, 28
10	0	Binary Output - All Variations	1	06		
10	1	Binary Output				
10	2	Binary Output Status			129, 130	00, 01
12	0	Control Block - All Variations				
12	1	Control Relay Output Block	3, 4, 5, 6	17, 28	129	echo of request
12	2	Pattern Control Block				
12	3	Pattern Mask				
20	0	Binary Counter - All Variations	1, 7, 8, 9, 10	06		
20	1	32-Bit Binary Counter			129, 130	00, 01
20	2	16-Bit Binary Counter			129, 130	00, 01
20	3	32-Bit Delta Counter				
20	4	16-Bit Delta Counter				
20	5	32-Bit Binary Counter without Flag			129, 130	00, 01
20	6	16-Bit Binary Counter without Flag			129, 130	00, 01
20	7	32-Bit Delta Counter without Flag				

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
20	8	16-Bit Delta Counter without Flag				
21	0	Frozen Counter - All Variations	1	06		
21	1	32-Bit Frozen Counter			129, 130	00, 01
21	2	16-Bit Frozen Counter			129, 130	00, 01
21	3	32-Bit Frozen Delta Counter				
21	4	16-Bit Frozen Delta Counter				
21	5	32-Bit Frozen Counter with Time of Freeze				
21	6	16-Bit Frozen Counter with Time of Freeze				
21	7	32-Bit Frozen Delta Counter with Time of Freeze				
21	8	16-Bit Frozen Delta Counter with Time of Freeze				
21	9	32-Bit Frozen Counter without Flag			129, 130	00, 01
21	10	16-Bit Frozen Counter without Flag			129, 130	00, 01
21	11	32-Bit Frozen Delta Counter without Flag				
21	12	16-Bit Frozen Delta Counter without Flag				
22	0	Counter Change Event - All Variations	1	06,07,08		
22	1	32-Bit Counter Change Event without Time			129, 130	17, 28
22	2	16-Bit Counter Change Event without Time			129, 130	17, 28
22	3	32-Bit Delta Counter Change Event without Time				
22	4	16-Bit Delta Counter Change Event without Time				
22	5	32-Bit Counter Change Event with Time				
22	6	16-Bit Counter Change Event with Time				
22	7	32-Bit Delta Counter Change Event with Time				
22	8	16-Bit Delta Counter Change Event with Time				

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
23	0	Frozen Counter Event - All Variations				
23	1	32-Bit Frozen Counter Event without Time				
23	2	16-Bit Frozen Counter Event without Time				
23	3	32-Bit Frozen Delta Counter Event without Time				
23	4	16-Bit Frozen Delta Counter Event without Time				
23	5	32-Bit Frozen Counter Event with Time				
23	6	16-Bit Frozen Counter Event with Time				
23	7	32-Bit Frozen Delta Counter Event with Time				
23	8	16-Bit Frozen Delta Counter Event with Time				
30	0	Analog Input - All Variations	1	06		
30	1	32-Bit Analog Input			129, 130	00, 01
30	2	16-Bit Analog Input			129, 130	00, 01
30	3	32-Bit Analog Input without Flag			129, 130	00, 01
30	4	16-Bit Analog Input without Flag			129, 130	00, 01
30	5	Short Floating Point Analog Input			129, 130	00, 01
31	0	Frozen Analog Input - All Variations				
31	1	32-Bit Frozen Analog Input				
31	2	16-Bit Frozen Analog Input				
31	3	32-Bit Frozen Analog Input with Time of Freeze				
31	4	16-Bit Frozen Analog Input with Time of Freeze				
31	5	32-Bit Frozen Analog Input without Flag				
31	6	16-Bit Frozen Analog Input without Flag				
32	0	Analog Change Event - All Variations	1	06,07,08		
32	1	32-Bit Analog Change Event without Time			129,130	17,28
32	2	16-Bit Analog Change Event without Time			129,130	17,28

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
32	3	32-Bit Analog Change Event with Time			129,130	17,28
32	4	16-Bit Analog Change Event with Time			129,130	17,28
32	5	Short Floating Point Analog Change Event without Time			129,130	17,28
33	0	Frozen Analog Event - All Variations				
33	1	32-Bit Frozen Analog Event without Time				
33	2	16-Bit Frozen Analog Event without Time				
33	3	32-Bit Frozen Analog Event with Time				
33	4	16-Bit Frozen Analog Event with Time				
40	0	Analog Output Status - All Variations	1	06		
40	1	32-Bit Analog Output Status			129, 130	00, 01
40	2	16-Bit Analog Output Status			129, 130	00, 01
40	3	Short Floating Point Analog Output Status			129, 130	00, 01
41	0	Analog Output Block - All Variations				
41	1	32-Bit Analog Output Block	3, 4, 5, 6	17, 28	129	echo of request
41	2	16-Bit Analog Output Block	3, 4, 5, 6	17, 28	129	echo of request
41	3	Short Floating Point Analog Output Block	3, 4, 5, 6	17, 28	129	echo of request
50	0	Time and Date - All Variations				
50	1	Time and Date	2 (see 4.14)	07 where quantity = 1		
50	2	Time and Date with Interval				
51	0	Time and Date CTO - All Variations				
51	1	Time and Date CTO			129, 130	07, quantity=1

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
51	2	Unsynchronized Time and Date CTO			129, 130	07, quantity=1
52	0	Time Delay - All Variations				
52	1	Time Delay Coarse			129	07, quantity=1
52	2	Time Delay Fine			129	07, quantity=1
60	0					
60	1	Class 0 Data	1	06		
60	2	Class 1 Data	1 20,21	06,07,08 06		
60	3	Class 2 Data	1 20,21	06,07,08 06		
60	4	Class 3 Data	1 20,21	06,07,08 06		
70	1	File Identifier				
80	1	Internal Indications	2	00 index=7		
81	1	Storage Object				
82	1	Device Profile				
83	1	Private Registration Object				
83	2	Private Registration Object Descriptor				
90	1	Application Identifier				
100	1	Short Floating Point				
100	2	Long Floating Point				
100	3	Extended Floating Point				
101	1	Small Packed Binary-Coded Decimal				
101	2	Medium Packed Binary-Coded Decimal				
101	3	Large Packed Binary-Coded Decimal				

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
No Object			13			
No Object			14			
No Object			23 (see 4.14)			

DNP V3.00

TIME SYNCHRONISATION PARAMETERS

This table describes the worst-case time parameters relating to time synchronisation, as required by DNP Level 2 Certification Procedure section 8.7

PARAMETER	VALUE
Time base drift	+/- 1 minute/month at 25°C +1 / -3 minutes/month 0 to 50°C
Time base drift over a 10-minute interval	+/- 14 milliseconds at 25°C +14 / -42 milliseconds 0 to 50°C
Maximum delay measurement error	+/- 100 milliseconds
Maximum internal time reference error when set from the protocol	+/- 100 milliseconds
Maximum response time	100 milliseconds

8.9 DNP Slave Device Profile Document

DNP v3.00

DEVICE PROFILE DOCUMENT

Vendor Name: Control Microsystems Inc.	
Device Name: SCADAPack controllers	
Highest DNP Level Supported: For Requests 2 For Responses 2	Device Function: ☐ Master ☒ Slave
Notable objects, functions, and/or qualifiers supported in addition to the Highest DNP Levels Supported (the complete list is described in the attached table): Function code 14 (warm restart) Function code 20 (Enable Unsolicited Messages) for class 1, 2, 3 objects only. Function code 21 (Disable Unsolicited Messages) for class 1, 2, 3 objects only. Object 41, variation 1 (32-bit analog output block)	
Maximum Data Link Frame Size (octets): Transmitted 292 Received (must be 292)	Maximum Application Fragment Size (octets): Transmitted 2048 Received 2048
Maximum Data Link Re-tries: ☐ None ☐ Fixed at ☒ Configurable, range 0 to 255	Maximum Application Layer Re-tries: ☐ None ☒ Configurable, range 0 to 255

Requires Data Link Layer Confirmation:

- ☐ Never
☐ Always
☐ Sometimes If 'Sometimes', when?

☒ Configurable for Always or Never

Requires Application Layer Confirmation:

- ☐ Never
☐ Always (not recommended)
☐ When reporting Event Data (Slave devices only)
☐ When sending multi-fragment responses (Slave devices only)

☐ Sometimes If 'Sometimes', when?

☒ Configurable for always or only when Reporting Event Data and Unsolicited Messages

Timeouts while waiting for:

Data Link Confirm	☐ None	☐ Fixed at _____	☐ Variable	☒
Configurable				
Complete Appl. Fragment	☐ None	☐ Fixed at _____	☐ Variable	☒ Configurable
Application Confirm	☐ None	☐ Fixed at _____	☐ Variable	☒ Configurable
Complete Appl. Response	☐ None	☐ Fixed at _____	☐ Variable	☒ Configurable

Others _____

Sends/Executes Control Operations:

WRITE Binary Outputs	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
SELECT/OPERATE	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
DIRECT OPERATE	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
DIRECT OPERATE - NO ACK	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Count > 1	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Pulse On	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Pulse Off	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Latch On	☐ Never	☐ Always	☐ Sometimes	☒ Configurable
Latch Off	☐ Never	☐ Always	☐ Sometimes	☒ Configurable

Queue	☒ Never	☐ Always	☐ Sometimes	☐ Configurable
Clear Queue	☒ Never	☐ Always	☐ Sometimes	☐ Configurable

FILL OUT THE FOLLOWING ITEM FOR MASTER DEVICES ONLY:

Expects Binary Input Change Events:

☐ Either time-tagged or non-time-tagged for a single event

☐ Both time-tagged and non-time-tagged for a single event

☐ Configurable (attach explanation)

FILL OUT THE FOLLOWING ITEMS FOR SLAVE DEVICES ONLY:

Reports Binary Input Change Events when no specific variation requested: ☐ Never ☐ Only time-tagged ☒ Only non-time-tagged ☐ Configurable to send both, one or the other (attach explanation)	Reports time-tagged Binary Input Change Events when no specific variation requested: ☒ Never ☐ Binary Input Change With Time ☐ Binary Input Change With Relative Time ☐ Configurable (attach explanation)
Sends Unsolicited Responses: ☐ Never ☒ Configurable by class ☐ Only certain objects ☐ Sometimes (attach explanation) ☒ ENABLE/DISABLE UNSOLICITED	Sends Static Data in Unsolicited Responses: ☒ Never ☐ When Device Restarts ☐ When Status Flags Change No other options are permitted.
Default Counter Object/Variation: ☐ No Counters Reported ☐ Configurable (attach explanation) ☒ Default Object 20 Default Variation 05 ☐ Point-by-point list attached	Counters Roll Over at: ☐ No Counters Reported ☐ Configurable (attach explanation) ☐ 16 Bits ☐ 32 Bits ☒ 16 Bits for 16-bit counters 32 Bits for 32-bit counters ☐ Point-by-point list attached

Sends Multi-Fragment Responses: ☒ Yes ☐ No

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
1	0	Binary Input - All Variations	1	06		
1	1	Binary Input			129, 130	00, 01
1	2	Binary Input with Status			129, 130	00, 01
2	0	Binary Input Change - All Variations	1	06,07,08		
2	1	Binary Input Change without Time	1	06,07,08	129, 130	17, 28
2	2	Binary Input Change with Time	1	06,07,08	129, 130	17, 28
2	3	Binary Input Change with Relative Time	1	06,07,08	129, 130	17, 28
10	0	Binary Output - All Variations	1	06		
10	1	Binary Output				
10	2	Binary Output Status			129, 130	00, 01
12	0	Control Block - All Variations				
12	1	Control Relay Output Block	3, 4, 5, 6	17, 28	129	echo of request
12	2	Pattern Control Block				
12	3	Pattern Mask				
20	0	Binary Counter - All Variations	1, 7, 8, 9, 10	06		
20	1	32-Bit Binary Counter			129, 130	00, 01
20	2	16-Bit Binary Counter			129, 130	00, 01
20	3	32-Bit Delta Counter				
20	4	16-Bit Delta Counter				
20	5	32-Bit Binary Counter without Flag			129, 130	00, 01
20	6	16-Bit Binary Counter without Flag			129, 130	00, 01
20	7	32-Bit Delta Counter without Flag				

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
20	8	16-Bit Delta Counter without Flag				
21	0	Frozen Counter - All Variations	1	06		
21	1	32-Bit Frozen Counter			129, 130	00, 01
21	2	16-Bit Frozen Counter			129, 130	00, 01
21	3	32-Bit Frozen Delta Counter				
21	4	16-Bit Frozen Delta Counter				
21	5	32-Bit Frozen Counter with Time of Freeze				
21	6	16-Bit Frozen Counter with Time of Freeze				
21	7	32-Bit Frozen Delta Counter with Time of Freeze				
21	8	16-Bit Frozen Delta Counter with Time of Freeze				
21	9	32-Bit Frozen Counter without Flag			129, 130	00, 01
21	10	16-Bit Frozen Counter without Flag			129, 130	00, 01
21	11	32-Bit Frozen Delta Counter without Flag				
21	12	16-Bit Frozen Delta Counter without Flag				
22	0	Counter Change Event - All Variations	1	06,07,08		
22	1	32-Bit Counter Change Event without Time			129, 130	17, 28
22	2	16-Bit Counter Change Event without Time			129, 130	17, 28
22	3	32-Bit Delta Counter Change Event without Time				
22	4	16-Bit Delta Counter Change Event without Time				
22	5	32-Bit Counter Change Event with Time				
22	6	16-Bit Counter Change Event with Time				
22	7	32-Bit Delta Counter Change Event with Time				
22	8	16-Bit Delta Counter Change Event with Time				

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
23	0	Frozen Counter Event - All Variations				
23	1	32-Bit Frozen Counter Event without Time				
23	2	16-Bit Frozen Counter Event without Time				
23	3	32-Bit Frozen Delta Counter Event without Time				
23	4	16-Bit Frozen Delta Counter Event without Time				
23	5	32-Bit Frozen Counter Event with Time				
23	6	16-Bit Frozen Counter Event with Time				
23	7	32-Bit Frozen Delta Counter Event with Time				
23	8	16-Bit Frozen Delta Counter Event with Time				
30	0	Analog Input - All Variations	1	06		
30	1	32-Bit Analog Input			129, 130	00, 01
30	2	16-Bit Analog Input			129, 130	00, 01
30	3	32-Bit Analog Input without Flag			129, 130	00, 01
30	4	16-Bit Analog Input without Flag			129, 130	00, 01
30	5	Short Floating Point Analog Input			129, 130	00, 01
31	0	Frozen Analog Input - All Variations				
31	1	32-Bit Frozen Analog Input				
31	2	16-Bit Frozen Analog Input				
31	3	32-Bit Frozen Analog Input with Time of Freeze				
31	4	16-Bit Frozen Analog Input with Time of Freeze				
31	5	32-Bit Frozen Analog Input without Flag				
31	6	16-Bit Frozen Analog Input without Flag				
32	0	Analog Change Event - All Variations	1	06,07,08		
32	1	32-Bit Analog Change Event without Time			129,130	17,28
32	2	16-Bit Analog Change Event without Time			129,130	17,28

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
32	3	32-Bit Analog Change Event with Time			129,130	17,28
32	4	16-Bit Analog Change Event with Time			129,130	17,28
32	5	Short Floating Point Analog Change Event without Time			129,130	17,28
33	0	Frozen Analog Event - All Variations				
33	1	32-Bit Frozen Analog Event without Time				
33	2	16-Bit Frozen Analog Event without Time				
33	3	32-Bit Frozen Analog Event with Time				
33	4	16-Bit Frozen Analog Event with Time				
40	0	Analog Output Status - All Variations	1	06		
40	1	32-Bit Analog Output Status			129, 130	00, 01
40	2	16-Bit Analog Output Status			129, 130	00, 01
40	3	Short Floating Point Analog Output Status			129, 130	00, 01
41	0	Analog Output Block - All Variations				
41	1	32-Bit Analog Output Block	3, 4, 5, 6	17, 28	129	echo of request
41	2	16-Bit Analog Output Block	3, 4, 5, 6	17, 28	129	echo of request
41	3	Short Floating Point Analog Output Block	3, 4, 5, 6	17, 28	129	echo of request
50	0	Time and Date - All Variations				
50	1	Time and Date	2 (see 4.14)	07 where quantity = 1		
50	2	Time and Date with Interval				
51	0	Time and Date CTO - All Variations				
51	1	Time and Date CTO			129, 130	07, quantity=1

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
51	2	Unsynchronized Time and Date CTO			129, 130	07, quantity=1
52	0	Time Delay - All Variations				
52	1	Time Delay Coarse			129	07, quantity=1
52	2	Time Delay Fine			129	07, quantity=1
60	0					
60	1	Class 0 Data	1	06		
60	2	Class 1 Data	1 20,21	06,07,08 06		
60	3	Class 2 Data	1 20,21	06,07,08 06		
60	4	Class 3 Data	1 20,21	06,07,08 06		
70	1	File Identifier				
80	1	Internal Indications	2	00 index=7		
81	1	Storage Object				
82	1	Device Profile				
83	1	Private Registration Object				
83	2	Private Registration Object Descriptor				
90	1	Application Identifier				
100	1	Short Floating Point				
100	2	Long Floating Point				
100	3	Extended Floating Point				
101	1	Small Packed Binary-Coded Decimal				
101	2	Medium Packed Binary-Coded Decimal				
101	3	Large Packed Binary-Coded Decimal				

DNP V3.00

DEVICE PROFILE DOCUMENT

IMPLEMENTATION OBJECT

This table describes the objects, function codes and qualifiers used in the device:

OBJECT			REQUEST (slave must parse)		RESPONSE (master must parse)	
Obj	Var	Description	Func Codes (dec)	Qual Codes (hex)	Func Codes	Qual Codes (hex)
		No Object	13			
		No Object	14			
		No Object	23 (see 4.14)			

DNP V3.00

TIME SYNCHRONISATION PARAMETERS

This table describes the worst-case time parameters relating to time synchronization, as required by DNP Level 2 Certification Procedure section 8.7

PARAMETER	VALUE
Time base drift	+/- 1 minute/month at 25°C +1 / -3 minutes/month 0 to 50°C
Time base drift over a 10-minute interval	+/- 14 milliseconds at 25°C +14 / -42 milliseconds 0 to 50°C
Maximum delay measurement error	+/- 100 milliseconds
Maximum internal time reference error when set from the protocol	+/- 100 milliseconds
Maximum response time	100 milliseconds

9 TeleBUS DF1 Protocol Overview

TeleBUS communication protocols provide a standard communication interface to the controller. The protocols operate on a wide variety of serial data links. These include RS-232 serial ports, RS-485 serial ports, radios, leased line modems, and dial up modems. The protocols are generally independent of the communication parameters of the link, with a few exceptions.

TeleBUS protocol commands may be directed to a specific device, identified by its station number, or broadcast to all devices. Up to 255 devices may connect to one communication network.

The TeleBUS protocols provide full access to the I/O database in the controller. The I/O database contains user-assigned registers and general purpose registers. Assigned registers map directly to the I/O hardware or system parameter in the controller. General purpose registers can be used by ladder logic and C application programs to store processed information, and to receive information from a remote device.

Application programs can initiate communication with remote devices. A multiple port controller can be a data concentrator for remote devices, by polling remote devices on one port and responding as a slave on another port.

The protocol type, communication parameters and station address are configured separately for each serial port on a controller. One controller can appear as different stations on different communication networks. The port configuration can be set from an application program, from the TelePACE programming software, or from another Modbus or DF1 compatible device.

9.1 Compatibility

There are four TeleBUS DF1 protocols:

- The **TeleBUS DF1 Full-Duplex BCC** protocol is compatible with the DF1 Full-Duplex data link layer protocol with block check character (BCC) error checking.
- The **TeleBUS DF1 Full-Duplex CRC** protocol is compatible with the DF1 Full-Duplex data link layer protocol with 16-bit cyclic redundancy check (CRC-16) error checking.
- The **TeleBUS DF1 Half-Duplex BCC** protocol is compatible with the DF1 Half-Duplex data link layer protocol with block check character (BCC) error checking.
- The **TeleBUS DF1 Half-Duplex CRC** protocol is compatible with the DF1 Half-Duplex data link layer protocol with 16-bit cyclic redundancy check (CRC-16) error checking.

Compatibility refers to communication only. The protocol defines communication aspects such as commands, syntax, message framing, error handling and addressing. The controllers do not mimic the internal functioning of any programmable controller. Device specific functions – those that relate to the hardware or programming of a specific programmable controller – are not implemented.

9.2 Serial Port Configuration

9.2.1 Communication Parameters

The TeleBUS DF1 protocols are, in general, independent of the serial communication parameters. The baud rate and parity may be chosen to suit the host computer and the characteristics of the data link.

The port configuration can be set in four ways:

- using the TelePACE program;
- using the **set_port** function from a C application program;
- writing to the I/O database from a C or ladder logic application program; or
- writing to the I/O database remotely from a Modbus or DF1 compatible device.

To configure a serial port through the I/O database, add the module, CNFG Serial port settings, to the Register Assignment.

9.2.1.1 Protocol Parameters

The TeleBUS DF1 protocols are eight bit character-oriented protocols. The table below shows possible and recommended communication parameters.

Parameter	Possible Settings	Recommended Setting
Baud Rate	see <i>Baud Rate</i> section below	see <i>Baud Rate</i> section below
Data Bits	8 data bits	8 data bits
Parity	none even odd	none
Stop bits	1 stop bit 2 stop bits ¹	1 stop bit
Flow control	disabled	disabled
Duplex	see <i>Duplex</i> section below	see <i>Duplex</i> section below

¹ Not applicable on the SCADAPack 33x or SCADAPack 350.

9.2.1.2 Baud Rate

The baud rate sets the communication speed. The possible settings are determined by the type of serial data link used. The table below shows the possible settings for the controller. Note that not all port types and baud rates are available on all controller ports.

Port Type	Possible Settings	Recommended Setting
RS-232 or RS-232 Dial-up modem	75 baud 110 baud 150 baud 300 baud 600 baud 1200 baud 2400 baud 4800 baud 9600 baud 19200 baud 38400 baud 57600 baud 115200 baud	Use the highest rate supported by all devices on the network.
RS-485	75 baud 110 baud 150 baud 300 baud 600 baud 1200 baud 2400 baud 4800 baud 9600 baud 19200 baud 38400 baud	Use the highest rate supported by all devices on the network.

9.2.1.3 Duplex

The TeleBUS DF1 protocols communicate in one direction at a time. However the duplex setting is determined by the type of serial data link used. The table below shows the possible settings for the controller. Note that not all port types are available on all controllers.

Port Type	Possible Settings	Recommended Setting
RS-232 or RS-232 Dial-up modem	half duplex full duplex	Use full duplex wherever possible. Use half duplex for most external modems.

Port Type	Possible Settings	Recommended Setting
RS-485	half duplex full duplex	Slave stations always use half duplex. Master stations can use full duplex only on 4 wire systems.

9.2.2 Protocol Parameters

The TeleBUS DF1 protocols operate independently on each serial port. Each port may set the protocol type, station number, protocol task priority and store-and-forward messaging options.

The protocol configuration can be set in four ways:

- using the TelePACE or ISaGRAF programs;
- using the **set_protocol** function from a C application program;
- writing to the I/O database from a C or ladder logic application program; or
- writing to the I/O database remotely from a Modbus or DF1 compatible device.

To configure protocol settings through the I/O database, add the module, CNFG Protocol settings, to the Register Assignment.

9.2.2.1 Protocol Type

The protocol type may be set to emulate the DF1 or Modbus protocols, or it may be disabled. When the protocol is disabled, the port functions as a normal serial port.

9.2.2.2 Station Number

The TeleBUS DF1 protocols allow up to 255 devices on a network. Station numbers identify each device. A device responds to commands addressed to it, or to commands broadcast to all stations.

The station number is in the range 0 to 254. Address 255 indicates a command broadcast to all stations, and cannot be used as a station number. Each serial port may have a unique station number.

9.2.2.3 Task Priority

A task is responsible for monitoring each serial port for messages. The real time operating system (RTOS) schedules the tasks with the application program tasks according to the task priority. The priority can be changed only with the **set_protocol** function from a C application program.

The default task priority is 3. Changing the priority is not recommended.

9.2.2.4 Store and Forward Messaging

Store and forward messaging is not supported by the TeleBUS DF1 protocols.

9.2.3 Default Parameters

All ports are configured at reset with default parameters when the controller is started in SERVICE mode. The ports use user-defined parameters when the controller is reset in the RUN mode. The default parameters are listed below.

Parameter	com1	com2	Com3	com4
Baud rate	9600	9600	9600	9600
Parity	none	none	None	none
Data bits	8	8	8	8
Stop bits	1	1	1	1
Duplex	full	full	Half	half
Protocol	Modbus RTU	Modbus RTU	Modbus RTU	Modbus RTU
Station	1	1	1	1
Rx flow control	none	none	Rx disable	Rx disable
Tx flow control	none	none	None	none
Serial time out	60 s	60 s	60 s	60 s
Type	RS-232	RS-232	RS-232	RS-232
Minimum Protected DF1 Address	0	0	0	0
Maximum Protected DF1 Address	11534	11534	11534	11534

Notes

com3 is supported only when the SCADAPack Lower I/O module is installed. com4 is supported only when the SCADAPack Upper I/O module is installed.

To optimize performance, minimize the length of messages on com3 and com4. Examples of recommended uses for com3 and com4 are for local operator display terminals, and for programming and diagnostics using the TelePACE program.

9.3 I/O Database

The TeleBUS protocols read and write information from the I/O database. The I/O database is an area of memory that can be accessed by C programs, Ladder Logic programs, ISaGRAF programs and communication protocols. The I/O database allows data to be shared between these programs. The I/O database contains user-assigned registers and general purpose registers.

User-assigned registers map directly to the I/O hardware or system parameter in the controller. Assigned registers are initialized to the default hardware state or system parameter when the controller is reset. Assigned output registers do not maintain their values

during power failures. However, output registers do retain their values during application program loading.

General purpose registers are used by ladder logic, ISaGRAF and C application programs to store processed information, and to receive information from remote devices. General purpose registers retain their values during power failures and application program loading. The values change only when written by an application program or a communication protocol.

9.3.1 Coil and Status Registers

Coil and status registers contain one bit of information, that is whether a signal is off or on.

For TelePACE firmware coil registers are single bits, which the protocols can read and write. There are 4096 coil registers located in the digital output section of the I/O database. Coil registers are contained within the DF1 16-bit addresses 0 to 255.

For TelePACE firmware status registers are single bits, which the protocols can read. There are 4096 status registers located in the digital input section of the I/O database. Status registers are contained within the DF1 16-bit addresses 256 to 511.

For ISaGRAF firmware coil registers are single bits, which the protocols can read and write. There are 9999 coil registers located in the digital output section of the I/O database. Coil registers are contained within the DF1 16-bit addresses 0 to 624.

For ISaGRAF firmware status registers are single bits, which the protocols can read. There are 9999 status registers located in the digital input section of the I/O database. Status registers are contained within the DF1 16-bit addresses 625 to 1249.

Coil and status registers are accessed 16 at a time or individually (in some commands) using a bitmask. Writing a one to a bit within the 16-bit address turns the bit on. Writing zero to the bit turns the bit off. If the register is assigned to an I/O module, the bit status is written to the module output hardware or parameter.

Reading a coil or status register returns 1 if the bit is on, or 0 if the bit is off. The stored value is returned from general purpose registers. The I/O module point status is returned from assigned registers.

9.3.2 Input and Holding Registers

Input and holding registers contain 16-bit values.

Input registers are 16-bit registers, which the protocol can read. For TelePACE firmware, there are 1024 input registers located in the analog input section of the I/O database. Input registers are contained within the DF1 addresses 512 to 1535.

For ISaGRAF firmware there are 9999 input registers located in the analog input section of the I/O database. Input registers are contained within the DF1 addresses 1250 to 11247.

Holding registers are 16-bit registers that the protocol can read and write. For TelePACE firmware there are 9999 holding registers located in the analog output section of the I/O database. Holding registers are contained within the DF1 addresses 1536 to 11534.

For ISaGRAF firmware there are 9999 holding registers located in the analog output section of the I/O database. Holding registers are contained within the DF1 addresses 11248 to 21247.

Writing any value to a general purpose register stores the value in the register. Writing a value to an assigned register, writes the value to the assigned I/O module.

Reading a general purpose register returns the value stored in the register. Reading an assigned register returns the value read from the I/O module.

9.3.3 Accessing the I/O Database Using TelePACE

Ladder logic programs access the I/O database through function blocks. All function blocks can access the I/O database. The function blocks in ladder logic use only the Modbus addressing scheme. Refer to the ***TelePACE Ladder Logic Reference and User Manual*** for details.

C language programs access the I/O database with two functions. The **dbase** function reads a value from the I/O database. The **setdbase** function writes a value to the I/O database. These functions use either Modbus or DF1 physical addressing. Refer to the ***TelePACE C Tools Reference and User Manual*** for full details on these functions.

9.3.3.1 Modbus Addressing

Modbus addressing is used in all ladder logic program functions. The controller's Register Assignment is also configured using Modbus addresses. The C functions **dbase** and **setdbase** support Modbus addressing. When the specified port is configured for one of the Modbus protocols, the function **master_message** uses Modbus addressing.

The range of Modbus registers available depends on the controller being used. The following sections list the Modbus registers available for each controller type.

9.3.3.1.1 SCADAPack, SCADAPack 100: 1024K and SCADASense 4202 Series

Coil registers are single bit addresses ranging from 00001 to 04096.

Status registers are single bit addresses ranging from 10001 to 14096.

Input registers are 16-bit addresses in the range 30001 to 31024.

Holding registers are 16-bit addresses in the range 40001 to 49999.

9.3.3.1.2 SCADAPack 100: 256K Controllers

Coil registers are single bit addresses ranging from 00001 to 04096.

Status registers are single bit addresses ranging from 10001 to 14096.

Input registers are 16-bit addresses in the range 30001 to 30512.

Holding registers are 16-bit addresses in the range 40001 to 44000.

9.3.3.1.3 SCADAPack 330, SCADAPack 334, SCADAPack 350, SCADAPack 32, and SCADASense 4203 Series

Coil registers are single bit addresses ranging from 00001 to 04096.

Status registers are single bit addresses ranging from 10001 to 14096.

Input registers are 16-bit addresses in the range 30001 to 39999.

Holding registers are 16-bit addresses in the range 40001 to 49999.

9.3.3.2 DF1 Addressing

DF1 addressing is used by the **MSTR** ladder logic function when the specified port is configured for one of the DF1 protocols. Modbus addressing must be used in all other ladder logic program functions.

DF1 addressing is used by function **master_message** when the specified port is configured for one of the DF1 protocols. The functions **dbase**, **setdbase**, **setABConfiguration** and **getABConfiguration** also support DF1 addressing.

All DF1 addresses are absolute word addresses beginning with the first 16-bit register in the I/O database at address 0.

9.3.3.3 Converting Modbus to DF1 Addresses

I/O database registers are assigned within the controller's Register Assignment using Modbus addressing. When polling the controller from a DF1 device it is necessary to know the DF1 address corresponding to each assigned register. Refer to the section for the controller type being used.

9.3.3.3.1 SCADAPack, SCADAPack 100: 1024K and SCADASense 4202 Series

In general, the cross-reference between Modbus and DF1 addressing is shown in the following table. DF1 addresses in this table are described in the format **word/bit** where **word** is the address of a 16-bit word and **bit** is the bit within that word. The bit address is optional.

Address Range		Description
Modbus DF1	00001 to 04096 0/0 to 255/15	Digital Output Database (single bit registers)
Modbus DF1	10001 to 14096 256/0 to 511/15	Digital Input Database (single bit registers)
Modbus DF1	30001 to 31024 512 to 1535	Analog Input Database (16-bit registers)
Modbus DF1	40001 to 49999 1536 to 11534	Analog Output Database (16-bit registers)

9.3.3.3.1.1 Coil Registers

To convert a Modbus coil register, *ModbusCoil*, to a DF1 address word/bit:

word = (*ModbusCoil* - 00001) / 16

bit = remainder of { (*ModbusCoil* - 00001) / 16 }

9.3.3.3.1.2 Status Registers

To convert a Modbus status register, *ModbusStatus*, to a DF1 address word/bit:

word = (*ModbusStatus* - 10001) / 16 + 256
bit = remainder of { (*ModbusStatus* - 10001) / 16 }

9.3.3.3.1.3 Input Registers

To convert a Modbus input register, *ModbusInput*, to a DF1 address word:

word = (*ModbusInput* - 30001) + 512

9.3.3.3.1.4 Holding Registers

To convert a Modbus holding register, *ModbusHolding*, to a DF1 address word:

word = (*ModbusHolding* - 40001) + 1536

9.3.3.3.1.5 Example

In this example the equivalent DF1 addresses are shown next to a sample SCADAPack Register Assignment specified with Modbus addresses.

Module	Start Register	End Register	DF1 Address Range
SCADAPack			
5601 I/O module			
digital outputs	00001	00012	0/0 to 0/11
digital inputs	10001	10016	256/0 to 256/15
analog inputs	30001	30008	512 to 519
DIN Controller digital inputs	10017	10019	257/0 to 257/2
DIAG Force LED	10020	10020	257/3
SCADAPack AOUT module	40001	40002	1536 to 1537

9.3.3.3.2 SCADAPack 330, SCADAPack 334, SCADAPack 350 and SCADAPack 32 Controllers

In general, the cross-reference between Modbus and DF1 addressing is shown in the following table. DF1 addresses in this table are described in the format **word/bit** where **word** is the address of a 16-bit word and **bit** is the bit within that word. The bit address is optional.

Address Range		Description
Modbus	00001 to 04096	Digital Output Database (single bit registers)
DF1	0/0 to 255/15	
Modbus	10001 to 14096	Digital Input Database (single bit registers)
DF1	256/0 to 511/15	
Modbus	30001 to 39999	Analog Input Database (16-bit registers)
DF1	512 to 10510	
Modbus	40001 to 49999	Analog Output Database (16-bit registers)
DF1	10511 to 20511	

9.3.3.3.2.1 Coil Registers

To convert a Modbus coil register, *ModbusCoil*, to a DF1 address word/bit:

word = (*ModbusCoil* - 00001) / 16
bit = remainder of { (*ModbusCoil* - 00001) / 16 }

9.3.3.3.2.2 Status Registers

To convert a Modbus status register, *ModbusStatus*, to a DF1 address word/bit:

word = (*ModbusStatus* - 10001) / 16 + 256

bit = remainder of { (*ModbusStatus* - 10001) / 16 }

9.3.3.3.2.3 Input Registers

To convert a Modbus input register, *ModbusInput*, to a DF1 address word:

word = (*ModbusInput* - 30001) + 512

9.3.3.3.2.4 Holding Registers

To convert a Modbus holding register, *ModbusHolding*, to a DF1 address word:

word = (*ModbusHolding* - 40001) + 10511

9.3.3.3.2.5 Example

In this example the equivalent DF1 addresses are shown next to a sample SCADAPack Register Assignment specified with Modbus addresses.

Module	Start Register	End Register	DF1 Address Range
SCADAPack			
5601 I/O module			
digital outputs	00001	00012	0/0 to 0/11
digital inputs	10001	10016	256/0 to 256/15
analog inputs	30001	30008	512 to 519
DIN Controller digital inputs	10017	10019	257/0 to 257/2
DIAG Force LED	10020	10020	257/3
SCADAPack AOUT module	40001	40002	10511 to 10512

9.3.4 Accessing the I/O Database Using ISaGRAF

ISaGRAF programs access the I/O database through function blocks. The function blocks in ISaGRAF may use the Modbus addressing scheme when a Network Address is defined for a variable. Refer to the *IEC 61131 User Manual* for details.

C language programs access the I/O database with two functions. The **dbase** function reads a value from the I/O database. The **setdbase** function writes a value to the I/O database. These functions use either Modbus or DF1 physical addressing. Refer to the *IEC 61131 User Manual* for full details on these functions.

9.3.4.1 Modbus Addressing

Modbus addressing can be used in ISaGRAF program functions.

The C functions **dbase** and **setdbase** support Modbus addressing. When the specified port is configured for one of the Modbus protocols, the function **master_message** uses Modbus addressing.

Modbus addresses coil registers with a single bit address ranging from 00001 to 09999. Status registers are also addressed with single bit addresses ranging from 10001 to 19999.

Input registers are addressed with 16-bit addresses in the range 30001 to 39999. And holding registers are addressed with a 16-bit address in the range 40001 to 49999.

9.3.4.2 DF1 Addressing

DF1 addressing is used by the **master** function when the specified port is configured for one of the DF1 protocols. Modbus addressing must be used in all other ladder logic program functions.

DF1 addressing is used by function **master_message** when the specified port is configured for one of the DF1 protocols. The functions **dbase**, **setdbase**, **setABConfiguration** and **getABConfiguration** also support DF1 addressing.

All DF1 addresses are absolute word addresses beginning with the first 16-bit register in the I/O database at address 0 and ending at address 21247.

9.3.4.3 Converting Modbus to DF1 Addresses

I/O database registers are assigned within the controller's Register Assignment using Modbus addressing. When polling the controller from an DF1 device it is necessary to know the DF1 address corresponding to each assigned register.

In general, the cross-reference between Modbus and DF1 addressing is shown in the following table. DF1 addresses in this table are described in the format **word/bit** where **word** is the address of a 16-bit word and **bit** is the bit within that word. The bit address is optional.

Address Range		Description
Modbus DF1	00001 to 09999 0/0 to 624/15	Digital Output Database (single bit registers)
Modbus DF1	10001 to 19999 625/0 to 1249/15	Digital Input Database (single bit registers)
Modbus DF1	30001 to 39999 1250 to 11248	Analog Input Database (16-bit registers)
Modbus DF1	40001 to 49999 11249 to 21247	Analog Output Database (16-bit registers)

9.3.4.3.1 Coil Registers

To convert a Modbus coil register, *ModbusCoil*, to an DF1 address word/bit:

$$\text{word} = (\text{ModbusCoil} - 00001) / 16$$

$$\text{bit} = \text{remainder of } \{ (\text{ModbusCoil} - 00001) / 16 \}$$

9.3.4.3.2 Status Registers

To convert a Modbus status register, *ModbusStatus*, to an DF1 address word/bit:

$$\text{word} = (\text{ModbusStatus} - 10001) / 16 + 625$$

bit = remainder of { (*ModbusStatus* - 10001) / 16 }

9.3.4.3.3 Input Registers

To convert a Modbus input register, *ModbusInput*, to an DF1 address word:

word = (*ModbusInput* - 30001) + 1250

9.3.4.3.4 Holding Registers

To convert a Modbus holding register, *ModbusHolding*, to an DF1 address word:

word = (*ModbusHolding* - 40001) + 11249

9.3.4.3.5 Example

In this example the equivalent DF1 addresses are shown next to a sample SCADAPack Register Assignment specified with Modbus addresses.

Module	Start Register	End Register	DF1 Address Range
SCADAPack			
5601 I/O module			
digital outputs	00001	00012	0/0 to 0/11
digital inputs	10001	10016	625/0 to 625/15
analog inputs	30001	30008	1250 to 1257
DIN Controller digital inputs	10017	10019	626/0 to 626/2
DIAG Force LED	10020	10020	626/3
SCADAPack AOUT module	40001	40002	11249 to 111250

9.4 Slave Mode

The TeleBUS DF1 protocols operate in slave and master modes simultaneously. In slave mode the controller responds to commands sent by another device. Commands may be sent to a specific device or broadcast to all devices.

The TeleBUS DF1 protocols emulate the protocol functions required for communication with a host device which uses the Non-Privileged commands from the DF1 Basic Command Set. These functions are described below.

Consult the DF1 I/O driver documentation included with the SCADA package, or specific DF1 documentation, for details on these commands. In most cases a knowledge of the actual commands is not required to set up the host system.

9.4.1 Broadcast Messages

A broadcast message is sent to all devices on a network. Each device executes the command. No device responds to a broadcast command. The device sending the command must query each device to determine if the command was received and processed. Broadcast messages are supported for function codes that write information.

A broadcast message is sent to station number 255.

9.4.2 Function Codes

The table summarizes the implemented function codes. Note that slave commands at the protocol layer access the I/O database in physical byte addresses. However, in master mode the interface to the TeleBUS DF1 protocol accesses the I/O database in physical 16-bit word addresses. The interface function block **MSTR** and C function **master_message** are described in the *Master Mode* section below.

The maximum number of 16-bit words that can be read or written with one slave command message is shown in the maximum column.

Function	Name	Description	Maximum
00	Protected Write	Writes words of data to limited areas of the database.	121
01	Unprotected Read	Reads words of data from any area of the database.	122
02	Protected Bit Write	Sets or resets individual bits within limited areas of the database.	30
05	Unprotected Bit Write	Sets or resets individual bits in any area of the database.	30
08	Unprotected Write	Writes words of data to any area of the database.	121

Functions 0, 2, 5 and 8 support broadcast messages. The functions are described in detail below.

9.4.2.1 Protected Write

The Protected Write function writes 8 bit values into limited areas of the I/O database. Access to the I/O database is limited to the protected address range.

Any number of bytes may be written up to the maximum number. The write may start at any byte address, provided the entire block is within the protected address range.

The protected address range is set using the **setABConfiguration** function from a C application program.

9.4.2.2 Unprotected Read

The Unprotected Read function reads 8 bit values from any area of the I/O database. Access to the I/O database is limited to the unprotected address range.

Any number of bytes may be read up to the maximum number. The read may start at any byte address, provided the entire block is within the unprotected address range.

9.4.2.3 Protected Bit Write

The Protected Bit Write function sets or resets individual bits within limited areas of the I/O database. Access to the I/O database is limited to the protected address range.

Bits are accessed one byte at a time and may be written up to the maximum number of bytes. The write may start at any byte address, provided the entire block is within the protected address range.

9.4.2.4 Unprotected Bit Write

The Unprotected Bit Write function sets or resets individual bits in any area of the I/O database. Access to the I/O database is limited to the unprotected address range.

Bits are accessed one byte at a time and may be written up to the maximum number of bytes. The write may start at any byte address, provided the entire block is within the unprotected address range.

9.4.2.5 Unprotected Write

The Unprotected Write function writes 8 bit values into any area of the I/O database. Access to the I/O database is limited to the unprotected address range.

Any number of bytes may be written up to the maximum number. The write may start at any byte address, provided the entire block is within the unprotected address range.

9.5 Master Mode

The TeleBUS DF1 protocols may act as a communication master on any serial port. In master mode, the controller sends commands to other devices on the network. Simultaneous master messages may be active on all ports.

9.5.1 Function Codes

The table shows the implemented function codes. Note that slave commands at the protocol layer access the I/O database in physical byte addresses. However, in master mode the interface to the TeleBUS DF1 protocol accesses the I/O database in physical 16-bit word registers. The interface function block **MSTR** and C function **master_message** are described in the *Sending Messages* section below.

The maximum number of 16-bit registers that can be read or written with one message is shown in the maximum column. The slave device may support fewer registers than shown; consult the manual for the device for details.

Function	Name	Description	Maximum
00	Protected Write	Writes words of data to limited areas of the database.	121
01	Unprotected Read	Reads words of data from any area of the database.	122
02	Protected Bit Write	Sets or resets individual bits within limited areas of the database.	1
05	Unprotected Bit Write	Sets or resets individual bits in any area of the database.	1
08	Unprotected Write	Writes words of data to any area of the database.	121

9.5.1.1 Protected Write

The Protected Write function writes 16-bit values into the I/O database of the slave device. Access to the I/O database is limited to the protected address range of the slave device. The data may come from any area of the master I/O database within its unprotected address range.

Any number of 16-bit registers may be written up to the maximum number supported by the slave device or the maximum number above, which ever is less. The write may start at any 16-bit register, provided the entire block is within the protected address range of the slave device.

9.5.1.2 Unprotected Read

The Unprotected Read function reads 16-bit values from any area of the I/O database of the slave device. Access to the I/O database is limited to the unprotected address range of the slave device. Data can be written into any area of the master I/O database within its unprotected address range.

Any number of 16-bit registers may be read up to the maximum number supported by the slave device or the maximum number above, which ever is less. The read may start at any 16-bit register, provided the entire block is within the unprotected address range of the slave device.

9.5.1.3 Protected Bit Write

The Protected Bit Write function sets or resets individual bits in the I/O database of the slave device. Access to the I/O database is limited to the protected address range of the slave device. The data may come from any area of the master I/O database within the unprotected address range.

One 16-bit register with a bitmask is used to write up to 16 bits of data. The register must be within the protected address range of the slave device.

9.5.1.4 Unprotected Bit Write

The Unprotected Bit Write function sets or resets individual bits in any area of the I/O database of the slave device. Access to the I/O database is limited to the unprotected address range of the slave device. The data may come from any area of the master I/O database within its unprotected address range.

One 16-bit register with a bitmask is used to write up to 16 bits of data. The register must be within the unprotected address range of the slave device.

9.5.1.5 Unprotected Write

The Unprotected Write function writes 16-bit values into any area of the I/O database of the slave device. Access to the I/O database is limited to the unprotected address range of the slave device. The data may come from any area of the master I/O database within its unprotected address range.

Any number of 16-bit registers may be written up to the maximum number supported by the slave device or the maximum number above, which ever is less. The write may start at any 16-bit register, provided the entire block is within the unprotected address range of the slave device.

9.5.2 Sending Messages

A master message is initiated in two ways:

- using the **master_message** function from a C application program; or
- using the **MSTR** function block from a ladder logic program.

These functions specify the port on which to issue the command, the function code, the slave station number, and the location and size of the data in the slave and master devices. The protocol driver, independent of the application program, receives the response to the command.

The application program detects the completion of the transaction by:

- calling the **get_protocol_status** function in a C application program; or
- using the output of the **MSTR** function block in a ladder logic program.

A communication error has occurred if the slave does not respond within the expected maximum time for the complete command and response. The application program is responsible for detecting this condition. When errors occur, it is recommended that the application program retry several times before indicating a communication failure.

The completion time depends on the length of the message, the length of the response, the number of transmitted bits per character, the transmission baud rate, and the maximum message turn-around time. One to three seconds is usually sufficient. Radio systems may require longer delays.

9.5.2.1 Polling DF1 PLCs

All DF1 PLCs, except the PLC-5/VME, will support some portion of the basic commands implemented.

AB PLC	Unprotected Read	Unprotected Write	Protected Write	Protected Bit Write	Unprotected Bit Write
MicroLogix ⁴ 1000	√	√			
SLC500 ^{1,3}	√	√			
SLC5/01 ^{1,3}	√	√			
SLC5/02 ^{1,2,3}	√	√			
SLC5/03 ^{2,3}	√	√			
SLC5/04 ^{2,3}	√	√			
1774-PLC	√	√	√	√	√
PLC-2	√	√	√	√	√
PLC-3 ⁵	√	√	√	√	√
PLC-5 ⁵	√	√	√	√	√
PLC-5/250 ⁵	√	√		√	√
PLC-5/VME					

9.5.2.1.1 Notes

¹ At the protocol level these commands convert and send the slave word address as a byte address. The SLC500, 5/01 (and 5/02, prior to Series C FRN 3) treat this byte address as if it

were a word address in the SLC500. This means that the (desired word address)/2 must be specified for the Slave Register Address in MSTR or master_message. Note that this always results in an even starting word address in the slave SLC. (E.g. Slave word address of 7 specified in MSTR accesses SLC word address 14.)

² The SLC5/02, 5/03 and 5/04 have a status selection bit (S:2/8) which allows selection of either word or byte addressing when interpreting only these commands. (Setting SLC bit S:2/8 = 1 selects byte addressing so that, for example, a slave word address of 7 specified in MSTR accesses SLC word address 7.)

³ These commands can access SLC memory only in the CIF or Common Interface File: N9. See further details in section 12-15 of the SLC500 Instruction Set manual.

⁴ These commands can access MicroLogix memory only in the CIF or Common Interface File: N7. See further details in section 12-15 of the MicroLogix 1000 Instruction Set manual.

⁵ These commands can access memory only in the CIF or “PLC-2 compatibility file”: N7. See further details in section 16-7 of the PLC-5 manual.

10 Modem Commands

The DIAL and INIM elements, and the TelePACE program dial-up connection operate with Hayes AT command set compatible external modems. The commands specify how the modem behaves. Each modem manufacturer has a different set of modem commands required to initialize the modem.

This appendix lists modem initialization command strings for a number of modems. If your modem is not on the list try the generic modem settings.

10.1 Modem Settings

If generic modem settings do not work, fill out the table below by looking up the commands in your modem manual. Construct a command string from the results. Your modem may not require all the commands or may require additional commands. Please contact our Technical Support department if you require assistance.

<i>Command</i>	<i>Description</i>
	reset modem to factory or default settings (This command should be sent first.)
	select extended result code set
	return verbal responses
	do not echo characters in command state
	return actual state of carrier detect (CD) signal
	hang up when DTR signal is dropped
	disable (local) flow control
	Communicate at a constant speed between DTE and modem
	answer incoming calls on the first ring
	carrier wait time = 50 seconds

Note that command strings do not begin with AT. The AT will be inserted automatically when the commands are sent to the modem.

10.2 Generic Modem

This command string works with many Hayes compatible modems.

<i>String</i>	X4 V1 E0 &C1 &D2 S0=1 S7=50
<i>Command</i>	<i>Description</i>
X4	select extended result code set
V1	return verbal responses
E0	do not echo characters in command state
&C1	return actual state of carrier detect (CD) signal
&D2	hang up when DTR signal is dropped
S0=1	answer incoming calls on the first ring
S7=50	carrier wait time = 50 seconds

10.3 5901 High Speed Dial-up Modem

Command	Description
&F0	reset modem to factory or default settings (This command should be sent first.)
X4	select extended result code set
V1	return verbal responses
E0	do not echo characters in command state
&C1	return actual state of carrier detect (CD) signal
&D2	hang up when DTR signal is dropped
&K0	disable (local) flow control
Nn and S37 see user manual	Communicate at a constant speed between DTE and modem
S0=1	answer incoming calls on the first ring
S7=50	carrier wait time = 50 seconds

10.4 ATI 14400 ETC-Express

String	&F0 X4 &C1 &D2 &K0 &Q0
Command	Description
&F0	recall factory (configuration) profile 0
X4	select extended result code set
&C1	return actual state of carrier detect (CD) signal
&D2	hang up when DTR signal is dropped
&K0	disable flow control
&Q0	select direct asynchronous operation

10.5 ATI 14400 ETC-E, ETC-I

String	&F0 &B1 &C1 &D2 &E0
Command	Description
&F0	recall factory (configuration) profile 0
&B1	DTE speed equals default value or the AT command speed
&C1	return actual state of carrier detect (CD) signal
&D2	hang up when DTR signal is dropped
&E0	Automatic retrying disabled

10.6 Hayes Smartmodem 1200

String	F1 Q0 V1 X1
Command	Description
F1	full duplex
Q0	result codes sent
V1	result codes transmitted as words (verbal)
X1	extended result code set

10.7 Hayes ACCURA 96, 144 and 288 Modems

String	&F &C1 &D2 &K0 &Q6
Command	Description
&F	recall factory configuration
&C1	return actual state of carrier detect (CD) signal
&D2	hang up when DTR signal is dropped
&K0	disable local flow control
&Q6	Communicate in asynchronous mode with automatic speed buffering (constant speed between DTE and modem)

10.8 Kama 2400 EI

String	&F &C1 &D2
Command	Description
&F	factory settings
&C1	DCD asserted when carrier detected
&D2	DTR controls the modem

10.9 Megahertz XJ4288 28.8 PC Card Modem

String	&F0 &Q0 %C0 S7=60
Command	Description
&F0	recall factory profile 0
&Q0	select direct asynchronous mode
%C0	disable data compression
S7=60	wait time for carrier = 60 seconds

10.10 Multitech 224E7B

String	F Q0 V1 X4 &C1 &D2 &E3
Command	Description
F	factory settings
Q0	send result codes
V1	verbal result codes
X4	extended result code set
&C1	DCD asserted when carrier detected
&D2	DTR controls the modem
&E3	no flow control correction

10.11 TeleSAFE 6901 Bell 212 Modem

1	String	
	Command	Description
	S0 register	Rings to answer
	S2 register	CTS-on delay measured in units of 8.3 ms.
	S3 register	Anti-streaming timer measured in 10's of seconds.
	S4 register	CTS-off delay measured in units of 8.3 ms.
	S6 register	Dial tone wait time measured in 10ths of a second.
	S7 register	Carrier wait time measured in seconds.
	S9 register	Carrier detect delay time measured in units of 8.3 ms.
	S10 register	Carrier loss time measured in units of 8.3 ms.
	S11 register	Tone dial speed measured in 0.050 seconds.
		Default
		1
		14 (116 ms)
		0 (disabled)
		4 (33 ms)
		30 (3 seconds)
		15 (15 seconds)
		10 (83 ms)
		10 (83 ms)
		2 (5 p/p second)

10.12 US Robotics Sportster 28,800

String	&F1 &A0 &H0 &K0 &M0 &R1 &B1 S0=1
Command	Description
&F1	generic factory configuration
&A0	ARQ result codes disabled
&K0	data compression disabled
&M0	error control (ARQ) disabled
&B1	fixed serial port rate
&H0	Flow control disabled
&R1	Modem ignores RTS
S0=1	answer phone on first ring (optional)