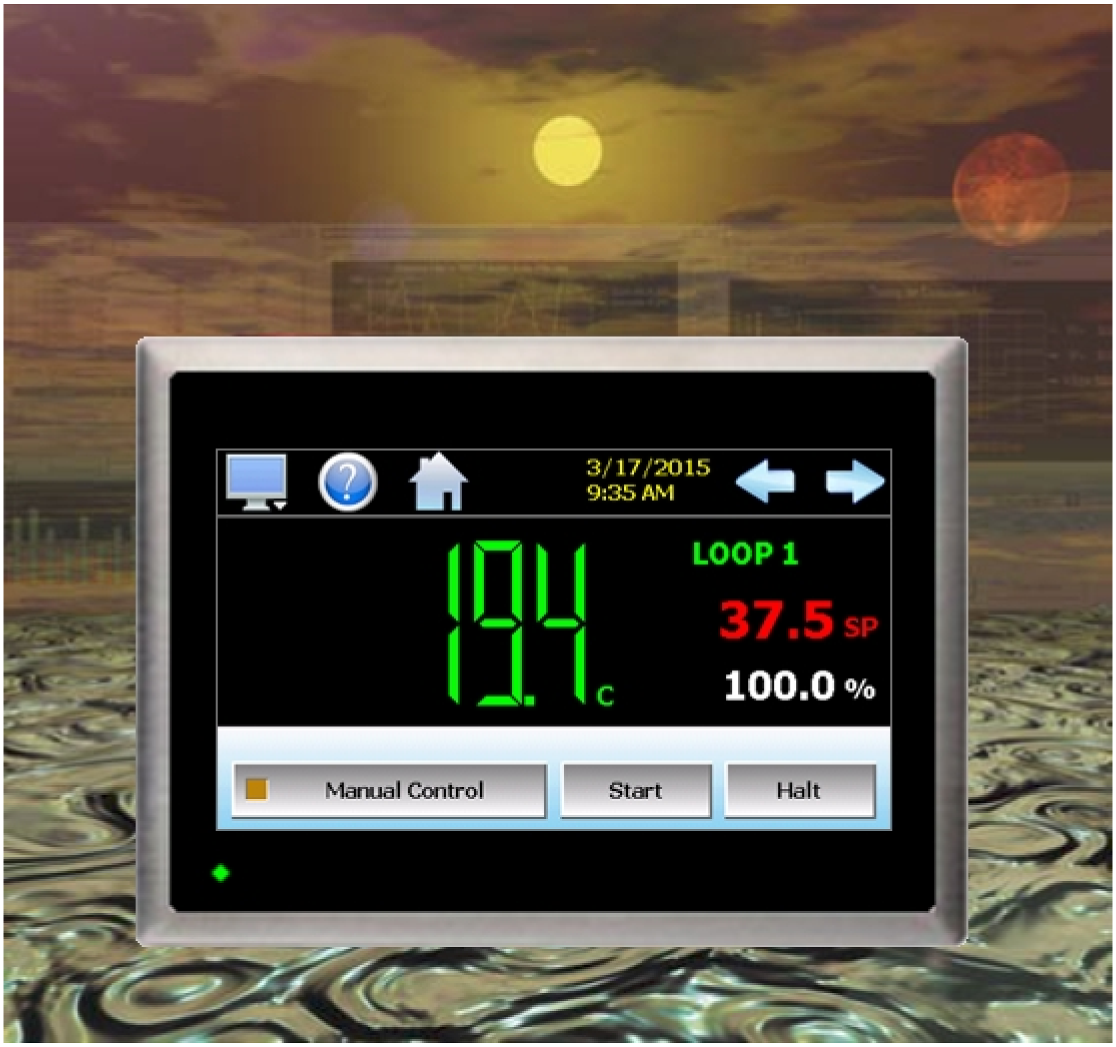
Orion *nCompass* MC i4.3

1 to 10 Loop Controller
User Communications Reference Manual



Orion nCompass MC i4.3 User Communications Reference Manual RevA
April 2015 (V1.0.0)
Supersedes: -

Safety Information in this Manual

Notes, cautions and warnings appear throughout this book to draw your attention to important operational and safety information.

A “**NOTE**” marks a short message to alert you to an important detail.

A “**CAUTION**” safety alert appears with information that is important for protecting your equipment and performance.

A “**WARNING**” safety alert appears with information that is important for protecting you, others and equipment from damage. Pay very close attention to all warnings that apply to your application.

⚠ This symbol (an exclamation point in a triangle) precedes a general CAUTION or WARNING statement.

⚡ This symbol (a lightning bolt in a lightning bolt in a triangle) precedes an electric shock hazard CAUTION or WARNING safety statement.

Technical Assistance

If you encounter a problem with your Orion nCompass controller, review all of your configuration information to verify that your selections are consistent with your application: inputs; outputs; alarms; limits; etc. If the problem persists after checking the above, you can get technical assistance by dialing +1 (866) 342-5332 or by faxing your request to +1 (866) 332-8014, Monday thru Friday, 8:00 a.m. to 5:00 p.m. Eastern Standard Time. You can also email your request to support@futuredesigncontrols.com.

An applications engineer will discuss your application with you.

Please have the following information available:

- Complete Model #’s and/or Serial #’s for Component(s) in Question
- Complete Software Version #’s
- All Configuration Information
- All User Manuals

Warranty and return information is on the back cover of this manual.

Your Comments

Your comments or suggestions on this manual are welcome. Please send them to:
Future Design Controls, P.O. Box 1196, Bridgeview, Illinois, 60455
Telephone: +1 (888) 751-5444; fax: +1 (888) 307-8014
csr@futuredesigncontrols.com

Appendix

1 What is nCompass?

The nCompass system combines all of the features of a loop controller, video/chart recorder and data logging system into a single/intuitive device. Email, SMS (text messaging), FTP (file transfer protocol for automated data backup) and remote view/control (Web server/VNC server) are standard with nCompass and can be accessed via LAN/WAN using a PC, tablet or smart phone device.

Future Designs “nCompass” provides a 4.3”color touch screen interface with standard “Smart Device” user interface features for one to ten loop OEM control applications. All loop configuration and runtime user access is configurable at the device with no PC software required. OEM’s have the ability to configure runtime features (screen availability, menus, language, etc...) to easily customize the system for their requirements. These configurations can be imported/exported to any other nCompass single/dual loop device for setup (from scratch) within minutes.

Individual high performance board level PID loop control boards (one for each loop) offer up to four control outputs each, powerful profiling capabilities with up to three events and full auto tune functionality with high resolution process inputs.

1.1 Features

Each of the nCompass loop control boards provide a single digital input that can be programmed as a automatic program control input for run, hold or abort, a manual mode or failure transfer control input or a program advance to next segment control input.

Each of the loop control boards also provide up to four control outputs which can be used as PID control outputs for heat/cool, direct outputs for controlling external equipment related to the application through software switches called events, or be programmed to act as system alarm outputs.

nCompass can be operated in single set point or automatic program control mode. Program entry is made easy through the use of copy, paste and delete menu selections. Programs can be copied to the external USB memory stick and then imported to another nCompass controller which eliminates the need to enter duplicate programs into multiple systems.

Data file analysis tools make looking at historical data a simple task. Any control variable saved to the data file can be plotted on the historical data chart for any time frame within the data file’s total time range.

The built in Ethernet functionality includes a ‘Web Server’ to provide access to all nCompass data (view only), a VNC interface for remote control and monitoring and an NTS clock, all available via a local Intranet connection (wired or wireless), or the World Wide Web using standard software like Microsoft’s Internet Explorer.

nCompass provides a rich set of tools for control interaction and process monitoring. Views include single and all loop views, charts, alarm, automated program status as well as historical data, alarm log and audit trail views. The menu driven interface eliminates screen “clutter” by providing an easy to use “Smart Device” interface for interaction between the user and nCompass.

nCompass can store more than one year of data on its SD memory card. Data logging can be enabled manually or automatically during program operation. Data backup is provided with a USB memory stick for plug and play transfer of files to any PC running Microsoft Windows XP operating systems and via the FTP back-up utility.

nCompass protects system access with 4 level security (user rights based), audit trails that document all user activity and ensures data integrity by digitally signing all data files and audit trails to meet regulatory requirements.

The nCompass controller includes the following features:

- One to ten loop controller models (automatic program operation included).
- Touch screen, “Smart Device” user interface (UI).
- Video recorder mode for view only applications.
- Email, SMS, FTP, VNC and Web functionality standard.
- Remote View/Control using PC, Tablet or Smartphone.
- Detailed maintenance, alarm monitoring and alarm history.
- User configurable data logging and historical data viewer.
- 4 level security with digitally signed audit trails and data files.
- National time server connectivity with daylight savings.
- Multi-lingual user interface supports over 25 languages.
- 30,000 hour LED display

2 Communications Wiring

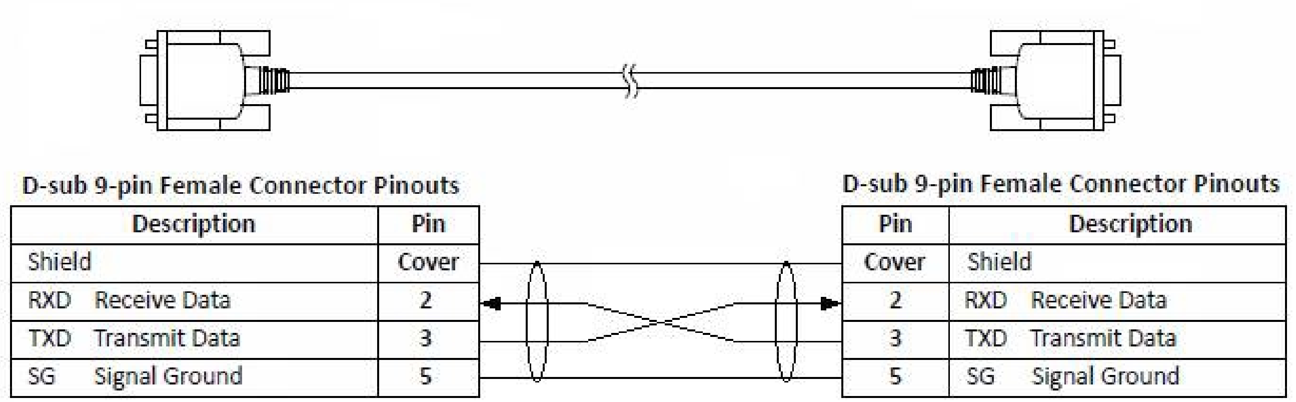
WARNING:

- To avoid potential electric shock and other hazards, all mounting and wiring for nCompass must conform to the National Electric Code (NEC) and other locally applicable codes.
- Special expertise is required to install, wire, configure and operate the nCompass controller. Personnel without such expertise should not install, wire or operate nCompass.

CAUTION:

- Prevent metal fragments and pieces of wire from dropping inside the housing of any nCompass component. If necessary, place a cover over the component during installation and wiring. Ingress of such fragments and chips may cause a fire hazard, damage or malfunction of the device.
- Locate the nCompass touch screen and all related control components away from AC power/motor wiring and sources of direct heat output such as transformers, heaters or large capacity resistors.

The nCompass touch screen provides an RS232C (COM1) user communications port for connecting nCompass to a PC running software such as FDC's Envision. In order to connect nCompass to a PC, a cable must be made according to the diagram below.



D-sub 9-pin Female Connector Pinouts

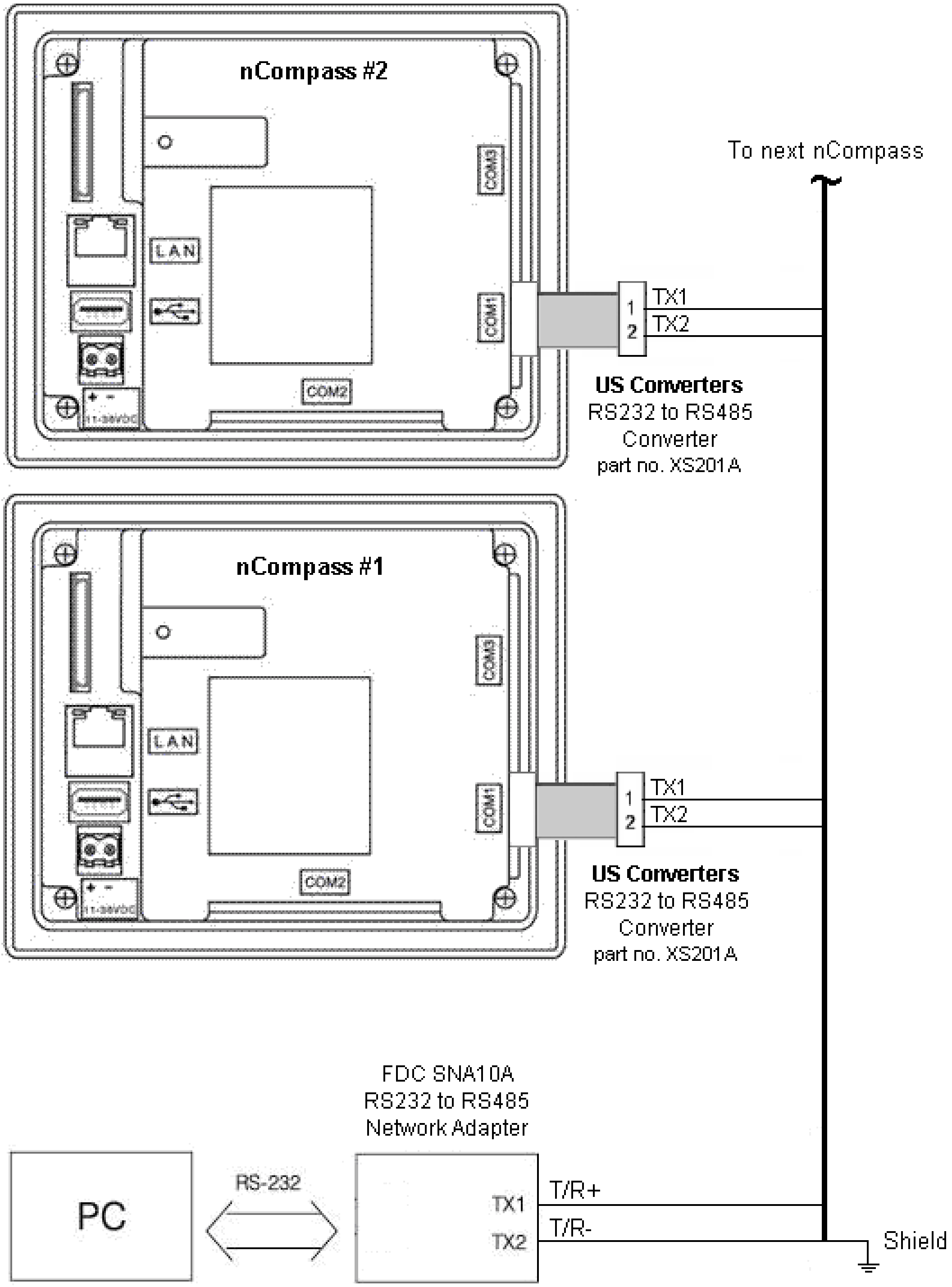
Description	Pin
Shield	Cover
RXD Receive Data	2
TXD Transmit Data	3
SG Signal Ground	5

D-sub 9-pin Female Connector Pinouts

Pin	Description
Cover	Shield
2	RXD Receive Data
3	TXD Transmit Data
5	SG Signal Ground

NOTE: *DO NOT use a standard null-modem cable to connect nCompass to a PC. Most computers do not provide a standard serial port and a USB to serial converter must be used. Incompatibilities may exist between nCompass and certain USB to serial adapters which will cause nCompass to malfunction when connected using a standard null-modem cable.*

If more than one nCompass controller is to be placed on the communication link with a PC, an RS232/485 converter will be required for each nCompass and the PC in order to convert the RS232 communications port to RS485 multi-drop communications network for connecting multiple nCompass controllers on the PC communications link.



NOTE: *The connection requires a single twisted-pair cable that is daisy-chained from one nCompass to the next. When using shielded twisted-pair cable, be sure to ground only when end of the cable, preferably at the RS232 to RS485 network adapter. Allowing any other portion of the cable shield to come in contact with ground, or grounding both ends, will cause ground loop currents to flow in that section of the cable which can cause communication errors.*

3 Communication Basics

The purpose of this document is to provide users interested in using data communications with the nCompass MC, the ability to set up and use a simple network of one or more nCompass control systems by providing a basic understanding of data communications using standard definitions, interfaces and protocols.

In this manual, numbers in the format ‘0x00’ represent values in hexadecimal. Numbers in the format ‘0’ represent values in decimal and finally, numbers in the format ‘00000000’ represent values in binary unless otherwise stated.

3.1 Explanation of Terms

Machine-to-Machine Communication
In order for machines to communicate with each other, they need a code called a character format or character set. They require rules called protocol to govern their conversation and prevent confusion and errors. Computers need a connecting interface over which to communicate. They may use one pair of wires to send information in one direction and another pair to send in the opposite direction (full duplex), or they may use one pair to send data in both directions (half duplex).

Character Format
The code or character format for nCompass data communications is shared by virtually everyone in the electronics industry. This code defines a stream of 1’s and 0’s that are created by varying a voltage signal in a regular manner. This code is the American Standard Code for Information Interchange, called ASCII.

Bits and Bytes
The word “bit” is simply the contraction of the words **bi**nary digi**t**. A bit is the basic unit in ASCII. It is either a “1” or a “0”. A byte is a string of eight bits that a computer treats as a single character. ASCII can use a single byte to represent each letter of the alphabet, each digit and each punctuation mark we use.

ASCII
The ASCII code defines 128 separate characters, one for each letter, digit and punctuation mark. ASCII also includes control characters similar to those we find on computer keys, such as backspace, shift and return. It also has nine communications control characters for identification, enquiry (inquiry), start of text, end of text, end of transmission, acknowledge, negative acknowledge and escape. The ASCII code is sometimes written in a base 16 number system that is called hexadecimal or “hex” for short. The numbers 0 through 9 represents the first ten digits of this system, and the letters A through F represents the final six digits. The 128 ASCII character codes with the decimal, binary and hexadecimal equivalents are listed in the following table.

ASCII Control Codes
ASCII Control Codes are used to give instructions to the remote device and result in specific actions, such as a line feed instruction on a printer. ASCII Control Codes, the first 33 ASCII characters (non printable), are important for the operation of communicating equipment. They give instruction to remote devices that result in specific actions such as a line feed on a printer. Holding down the keyboard control key while pressing the appropriate keyboard key is what sends these values.

ASCII Character Chart

Char	Code	Decimal	Binary	Hex	Char	Code	Decimal	Binary	Hex
NUL	Ctrl @	0	00000000	00	**@**	Shift 2	64	01000000	40
SOH	Ctrl A	1	00000001	01	**A**	Shift A	65	01000001	41
STX	Ctrl B	2	00000010	02	**B**	Shift B	66	01000010	42
ETX	Ctrl C	3	00000011	03	**C**	Shift C	67	01000011	43
EOT	Ctrl D	4	00000100	04	**D**	Shift D	68	01000100	44
ENQ	Ctrl E	5	00000101	05	**E**	Shift E	69	01000101	45
ACK	Ctrl F	6	00000110	06	**F**	Shift F	70	01000110	46
BEL	Ctrl G	7	00000111	07	**G**	Shift G	71	01000111	47
BS	Ctrl H	8	00001000	08	**H**	Shift H	72	01001000	48
TAB	Ctrl I	9	00001001	09	**I**	Shift I	73	01001001	49
LF	Ctrl J	10	00001010	0A	**J**	Shift J	74	01001010	4A
VT	Ctrl K	11	00001011	0B	**K**	Shift K	75	01001011	4B
FF	Ctrl L	12	00001100	0C	**L**	Shift L	76	01001100	4C
CR	Ctrl M	13	00001101	0D	**M**	Shift M	77	01001101	4D
SO	Ctrl N	14	00001110	0E	**N**	Shift N	78	01001110	4E
SI	Ctrl O	15	00001111	0F	**O**	Shift O	79	01001111	4F
DLE	Ctrl P	16	00010000	10	**P**	Shift P	80	01010000	50
DC1	Ctrl Q	17	00010001	11	**Q**	Shift Q	81	01010001	51
DC2	Ctrl R	18	00010010	12	**R**	Shift R	82	01010010	52
DC3	Ctrl S	19	00010011	13	**S**	Shift S	83	01010011	53
DC4	Ctrl T	20	00010100	14	**T**	Shift T	84	01010100	54
NAK	Ctrl U	21	00010101	15	**U**	Shift U	85	01010101	55
SYN	Ctrl V	22	00010110	16	**V**	Shift V	86	01010110	56
ETB	Ctrl W	23	00010111	17	**W**	Shift W	87	01010111	57
CAN	Ctrl X	24	00011000	18	**X**	Shift X	88	01011000	58
EM	Ctrl Y	25	00011001	19	**Y**	Shift Y	89	01011001	59
SUB	Ctrl Z	26	00011010	1A	**Z**	Shift Z	90	01011010	5A
ESC	Ctrl [	27	00011011	1B	**[**	[	91	01011011	5B
FS	Ctrl \	28	00011100	1C	\	\	92	01011100	5C
GS	Ctrl]	29	00011101	1D	**]**	]	93	01011101	5D
RS	Ctrl ^	30	00011110	1E	**^**	Shift 6	94	01011110	5E
US	Ctrl _	31	00011111	1F	**_**	Shift -	95	01011111	5F
SP	SPACE	32	00100000	20	**`**	`	96	01100000	60
!	Shift 1	33	00100001	21	**a**	A	97	01100001	61
"	Shift ‘	34	00100010	22	**b**	B	98	01100010	62
#	Shift 3	35	00100011	23	**c**	C	99	01100011	63
$	Shift 4	36	00100100	24	**d**	D	100	01100100	64
%	Shift 5	37	00100101	25	**e**	E	101	01100101	65
&	Shift 7	38	00100110	26	**f**	F	102	01100110	66
‘	‘	39	00100111	27	**g**	G	103	01100111	67
(	Shift 9	40	00101000	28	**h**	H	104	01101000	68
)	Shift 0	41	00101001	29	**I**	I	105	01101001	69
*****	Shift 8	42	00101010	2A	**j**	J	106	01101010	6A
+	Shift =	43	00101011	2B	**k**	K	107	01101011	6B
,	,	44	00101100	2C	**l**	L	108	01101100	6C
-	-	45	00101101	2D	**m**	M	109	01101101	6D
.	.	46	00101110	2E	**n**	N	110	01101110	6E
/	/	47	00101111	2F	**o**	O	111	01101111	6F
0	0	48	00110000	30	**p**	P	112	01110000	70
1	1	49	00110001	31	**q**	Q	113	01110001	71
2	2	50	00110010	32	**r**	R	114	01110010	72
3	3	51	00110011	33	**s**	S	115	01110011	73
4	4	52	00110100	34	**t**	T	116	01110100	74
5	5	53	00110101	35	**u**	U	117	01110101	75
6	6	54	00110110	36	**v**	V	118	01110110	76
7	7	55	00110111	37	**w**	W	119	01110111	77
8	8	56	00111000	38	**x**	X	120	01111000	78
9	9	57	00111001	39	**y**	Y	121	01111001	79
:	Shift ;	58	00111010	3A	**z**	Z	122	01111010	7A
;	;	59	00111011	3B	**{**	Shift [	123	01111011	7B
<	Shift ,	60	00111100	3C	**|**	Shift \	124	01111100	7C
=	=	61	00111101	3D	**}**	Shift]	125	01111101	7D
>	Shift .	62	00111110	3E	**~**	Shift `	126	01111110	7E
?	Shift /	63	00111111	3F	**DEL**	Delete	127	01111111	7F

4 Serial Communication

The user communications interface for nCompass employs serial communication, which is the exchange of data in a one-bit-at-a-time, sequential manner on a single data line or channel. Serial contrasts with parallel communication, which sends several bits of information simultaneously over multiple lines or channels. Not only is serial data communication simpler than parallel, it is also less costly.

Baud Rate

The baud unit is named after Jean Maurice Emile Baudot, who was an officer in the French Telegraph Service. He is credited with devising the first uniform-length 5-bit code for characters of the alphabet in the late 19th century. What baud really refers to is modulation rate or the number of times per second that a line changes state. This is not always the same as bits per second (BPS). However, if you connect two serial devices together using direct cables then baud and BPS are in fact the same. Thus, if you are running at 9600 BPS, then the line is also changing states 9600 times per second.

Typical baud rates used for computers are 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 57600 and 115200 baud. As the baud rate increases, so does the transmission rate of data. Thus you get more information in a shorter period of time. However, the faster the transmission rate, the more susceptible it is to error due to the quality of the cable and sources of electrical “noise” in the environment. To provide the best balance between speed and noise immunity, nCompass uses a baud rate of 9600 baud. *In order for a device to communicate with nCompass, it must have its serial port set for 9600 baud in order for data communications to work properly.*

Start and Stop Bits

The start bit informs the receiving device that a character is coming, and a stop bit tells it that a character is complete. The start bit is always a 0. The stop bit is always a 1. The human speech equivalent of these bits could be a clearing of the throat to get someone’s attention (start bit); and a pause at the end of a phrase (stop bit). Both help the listener understand the message.

A stop bit has a value of 1 - or a mark state - and it can be detected correctly even if the previous data bit also had a value of 1. This is accomplished by the stop bit's duration. Stop bits can be 1, 1.5, or 2 bit periods in length. nCompass uses the default – and most common – length of 1 period for the stop bit. *A device used to communicate with nCompass must also have its serial port set to use a stop bit of 1 in order for data communications to work properly.*

Parity Bit

Besides the synchronization provided by the use of start and stop bits, an additional bit called a parity bit may optionally be transmitted along with the data. A parity bit affords a small amount of error checking, to help detect data corruption that might occur during transmission. There are several defined parity selections available for serial communications. They are even parity, odd parity, mark parity, space parity or none at all can be used.

When even or odd parity is being used, the number of marks (logical 1 bits) in each data byte are counted, and a single bit is transmitted following the data bits to indicate whether the number of 1 bits just sent is even or odd. Mark parity means that the parity bit is always set to the mark signal condition and likewise space parity always sends the parity bit in the space signal condition. Since these two parity options serve no useful purpose whatsoever, they are almost never used.

nCompass offers parity settings of Even, Odd and None. *In order for a device to communicate with nCompass, it must have its serial port set to use the same parity setting as nCompass in order for data communications to work properly. nCompass is provided with a default parity setting of Even in order to be compatible with FDC’s Envision software.*

4.1 Interface Standards

An interface is a means for electronic systems to interact. It's a specific kind of electrical wiring configuration. It has nothing to do with how data is sent over that connection. The two most common interfaces used today are RS-232, which provides a simple 1 to 1 connection and RS-485, which provides a multi-drop connection where more than one device can be placed on the same line. The nCompass communications interface is RS-232, but can be changed to RS-485 through the use of external RS232/485 adapters.

EIA-232 (Full Duplex)

An EIA-232 (formerly RS-232C) interface uses three wires: a single transmit wire; a single receive wire; and a common line. Only two devices can use an EIA-232 interface. A -3 to -24 volt signal indicates a 1 and a +3 to +24 volt signal indicates a 0. The EIA-232 signal is referenced to the common line rather than to a separate wire, as in EIA-485. Thus, an EIA-232 cable is limited to a maximum of 50 feet, due to noise susceptibility.

EIA-485 (Half Duplex)

An EIA-485 interface uses two wires: a T/R+, a T/R- line. A -5-volt signal is interpreted as a 1, a +5-volt signal as a 0. As many as 31 slave devices can be connected to a master on a multi-drop network up to 4000 feet long.

Wiring

Most PCs have a standard EIA-232 port (usually referred to as RS-232). In these instances, you must use an interface converter to connect to an EIA-485 multi-drop system. The standards do not specify the wire size and type. Use of 24 AWG twisted pair provides excellent results. If shielded cable is used, terminate the shield at one end only. Always follow the manufacturer's instructions supplied with the interface converter. See Biasing of Buses next.

Biasing of Buses

The EIA-485 standard requires the bus to be biased for reliable communication. This requires termination resistors to be placed across the T/R+ and T/R- wires. One resistor is placed at the PC where it connects to the EIA-485 bus. The second resistor is placed at the last controller on the network. Do not place resistors at each controller. The impedance of the wires used for the bus determines the resistor value. For twisted pair, the value is typically 120 ohms. In addition, it may be necessary to have a pull-up and pull-down resistor between the power supply and ground of the interface adapter.

Check the documentation that came with your interface adapter. Biasing the bus reduces reflection of signals sent down the bus. These reflections are sometimes referred to as a standing wave. This condition is most notable when communicating at high baud rates over longer distances.

4.1.1 Interface Converters

The purpose of an interface converter is to allow two different buses to be connected together. Interface converters are required when connecting an EIA-232 port to an EIA-485 bus. The EIA-485 bus is a half duplex bus. This means that it can only send or receive data at any given time. Some interface converters on the market provide the ability to have full duplex with the EIA-485 bus. This is accomplished by using two receivers and transmitters tied in tandem. This type of converter will not work with the nCompass controller. Be sure that the model you purchase is designed for half duplex.

Another consideration when selecting an interface converter is how the converter handles switching between transmit and receive. Typically it is accomplished via a handshake line from the PC. When data flows into the converter from the PC, a handshake line is placed high. When data flows out of the converter to the PC, the handshake line is placed low. In this way, the handshake line controls the direction of information. Another method of achieving this is to use a built-in timer. The converter switches to transmit when a character is sent to it from the PC. After a period of time when the PC has not transmitted, the converter switches to a receive mode.

It is important that you understand how your converter accomplishes this task. You are required to wire this feature or make settings on the converter to enable this function. The PC will not talk to the controller correctly without properly setting this. Your converter may also require settings through dip switches, to set up communications parameters like baud rate, data bits, start bits, stop bits and handshaking. The converter may also require a separate power supply. Some converters get their power from the handshake lines of the PC. If you rely on this method, you will need to wire these additional lines. In addition, your software must set these lines high. A more reliable method is to use an external power supply. This is especially necessary when using a laptop computer. See the documentation that is provided with your converter for more information.

Not all converters are equal in performance. If your chamber operates in a harsh, electrically noisy environment, this can cause less robust converters to work intermittently or not at all. The following converter has been tested and is compatible with nCompass. The converter is equipped with automatic send data control circuits, driver control in the converter hardware, so you don't have to work with software at all. The circuit monitors data flow and enables the driver during transmission and automatically disables it when no data is being sent. There is no need to rework software or install new drivers.

US Converters
405 W. Fairmont Dr.
Tempe, AZ 85282
E-mail: mail@usconverters.com
www.USconverters.com

Part # **XS201A** RS232 to RS485 Converter

Future Design Controls
7524 West 98th Place
Bridgeview, IL 60455
Phone: 888-751-5444
Fax: 888-307-8014
E-mail: csr@futuredesigncontrols.com
www.futuredesigncontrols.com

Part # **SNA10A** Smart Network Adapter
Part # **DB9M-DB9F-6ft** (Cable Accessory to connect SNA10A to PC)

4.2 Protocol

Protocol describes how to initiate an exchange. It also prevents two machines from attempting to send data at the same time. There are a number of different data communications protocols, just as there are different human cultural protocols that vary according to the situation.

The protocol portion of nCompass communications is very important, because it provides a quality of communication that others often don't have. Protocol-driven communications are more accurate because they are less prone to both operator and noise errors. Protocol maintains system integrity by requiring a response to each message. It's like registered mail — you know that your letter has been received because the post office sends you a signed receipt.

In nCompass data communications, a dialog will continue successfully as long as the messages are in the correct form and responses are returned to the protocol leader. If the operator enters an incorrect message, or interference comes on to the data line, there will be no response. In that case the master must retransmit the message or go to a recovery procedure. If an operator continues to enter an incorrect message or interference continues on the data line, the system will halt until the problem is resolved. nCompass uses Modbus RTU as the protocol of choice. Modbus RTU enables a PC to read and write directly to registers containing the nCompass parameters. With it, you can read all of the controller's parameters with just a single read command.

Modbus Remote Terminal Unit (RTU)

Gould Modicon, now called AEG Schneider, created this protocol for process control systems called "Modbus". It has the advantage over other protocols of being extremely reliable in exchanging information. This protocol works on the principle of packet exchanges. The packet contains the address of the controller to receive the information, a command field that says what is to be done with the information and several fields of data. The last item sent in the packet is a field to ensure the data is received intact. This is called a cyclic redundancy check-sum. See the following example for information on how to generate this value. All information is exchanged in hex numbers. nCompass only supports the binary version of Modbus, referenced as RTU. The ASCII version is less efficient and is not supported. Therefore, you must be certain to format all data in hexadecimal.

The CRC (Cyclical Redundancy Checksum) is calculated by the following steps:

1. Load a 16-bit register (called CRC register) with 0xFFFF

2. Exclusive OR the first 8-bit byte of the command message with the low order byte of the 16-bit CRC register, putting the result in the CRC register.

3. Shift the CRC register one bit to the right with MSB zero filling. Extract and examine the LSB.

4. If the LSB of the CRC register is zero, repeat step 3, else Exclusive OR the CRC register with the polynomial value 0xA001.

5. Repeat steps 3 and 4 until eight shifts have been performed. When this is done, a complete 8-bit byte will have been processed.

6. Repeat steps 2 through 5 for the next 8-bit byte of the command message. Continue doing this until all bytes of the command message have been processed. The final contents of the CRC register is the CRC value.

When transmitting the CRC value in the message, the upper and lower bytes of the CRC value must be swapped, i.e. the lower order byte will be transmitted first.

Example Cyclical Redundancy Checksum (CRC) Algorithm

```
unsigned int calc_crc(unsigned char *start_of_packet, unsigned char *end_of_packet)
{
unsigned int crc;
unsigned char bit_count;
unsigned char *char_ptr;

/* Start at the beginning of the packet */
char_ptr = start_of_packet;
/* Initialize CRC */
crc = 0xFFFF;
/* Loop through the entire packet */
do{
    /* Exclusive-OR the byte with the CRC */
    crc ^= (unsigned int)*char_ptr;
    /* Loop through all 8 data bits */
    bit_count = 0;
    do{
        /* If the LSB is 1, shift the CRC and XOR the polynomial mask with the CRC */
        if(crc & 0x0001){
            crc >>= 1;
            crc ^= 0xA001;
            }
        /* If the LSB is 0, shift the CRC only */
        else{
        crc >>= 1;
        }
    } while(bit_count++ < 7);
} while(char_ptr++ < end_of_packet);
return(crc);
}
```

4.3 Creating your own Modbus Application

Listed below are a few of the more common software packages that claim to support the Modbus protocol. This list is provided as informational only. Contact the software manufacturer for more information on applying their software.

LabView by National Instruments
11500 N Mopac Expwy
Austin, TX 78759-3504
Phone 800-683-8411
http://www.natinst.com

Wonderware by Wonderware
26561 Rancho Pkwy. South
Lake Forest, CA 92630
Phone 949-727-3200
http://www.wonderware.com

SpecView by SpecView Corporation
13409 53rd Ave NW
Gig Harbor, WA 98332
Phone 253-853-3199
http://www.specview.com

If you already have a software application that uses Modbus, you can simply skip to the nCompass parameter table in Section 5 for the information your program requires. The rest of this section provides information on writing a software application that uses Modbus.

1. You must code messages in eight-bit bytes, with even parity, one stop bit (8, even, 1). nCompass has its parity set to even as default from the factory.

2. Negative parameter values must be written in twos' complement format. Parameters are stored in two-byte registers accessed with read and write commands to a relative address.

3. Messages are sent in packets that must be delimited by a pause at least as long as the time it takes to send 28 bits (3.5 characters). To determine this time in seconds, divide 28 by the baud rate. In the case of nCompass communications at 9600 baud, this calculates to a minimum period of ~3ms.

   In addition, the nCompass timeout period must be added to that in order to properly time the send and receive messages between the host computer and multiple nCompass controllers on the serial link. With a default timeout period in nCompass of 135ms, it makes a total pause of 138ms minimum. Thus, after you receive a response from an nCompass controller at your PC, you must wait a minimum of 138ms before sending the next command.

4. Values containing decimal points such as process values and set points, have the decimal point implied, i.e., the data exchange can only be performed using whole numbers. Thus, the value must be scaled appropriately in order to exchange the data correctly. For example, a setpoint of 78.4 degrees must be sent as a value of 784 in order for nCompass to be set correctly. Likewise, a process value read from nCompass with a value of 827 is actually 82.7 degrees. Consult the parameter table for the proper format and allowable range of each value.

5. When monitoring a process, try to keep the number of read and write commands to a minimum of 500ms between exchanges to a single controller. Continuously reading data at a faster rate consumes an excess amount of the controller's processor time and does not provided any additional benefits in process monitoring.

Handling Communication Errors

Messages with the wrong format or illegal values will receive an exception response. Messages with the wrong CRC or timing will receive no response. It is the user's responsibility to handle the error appropriately within their own software and determine whether to resend the message or halt for operator intervention.

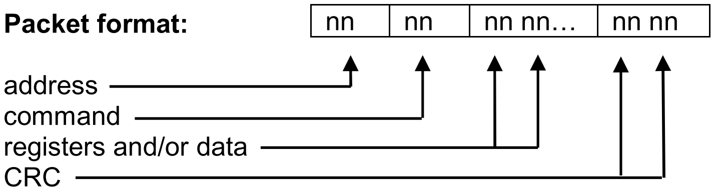
User Responsibility

Refrain from reading or writing from/to a register that does not exist or is currently disabled. Writing values to unassigned registers could cause system instability, malfunction or failure. Care must also be taken in that the process can not cause damage to property or injury to personnel if the wrong commands are sent due to operator error or equipment malfunction.

4.3.1 Packet Syntax

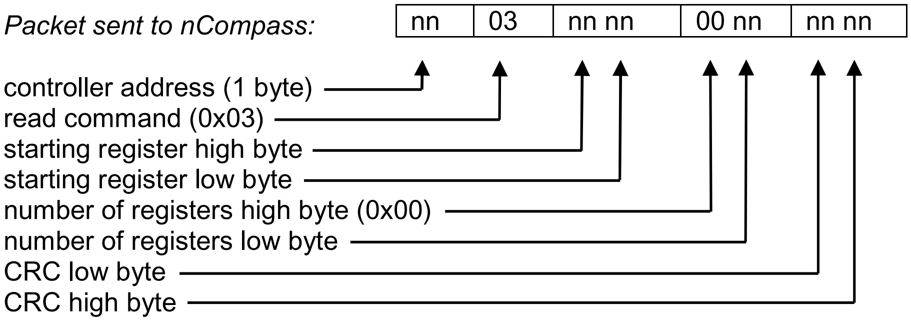
Each message packet begins with a one-byte controller address, from 0x01 to 0x1F. The second byte in the message packet identifies the message command: read (0x03); write single (0x06) or write multiple (0x10). The next “n” bytes of the message packet contain register addresses and/or data. The last two bytes in the message packet contain a two-byte Cyclical Redundancy Checksum (CRC) for error detection.

Packet format:

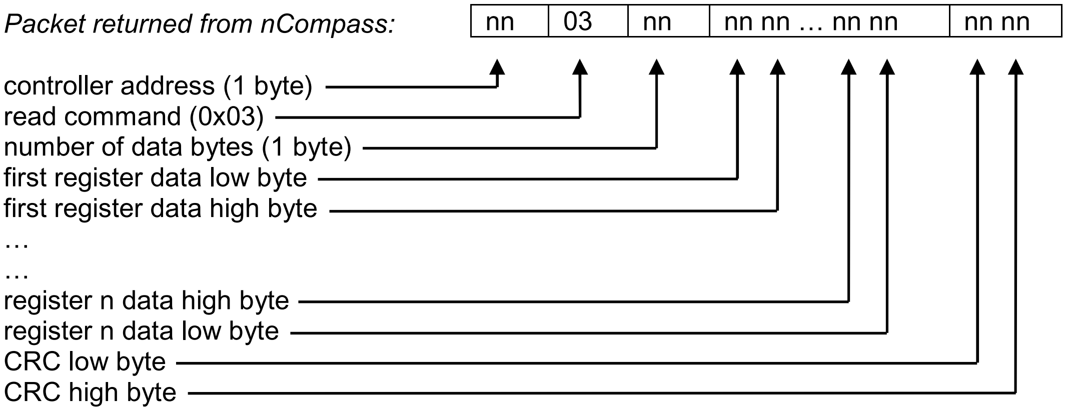


Read Register(s) Command (0x03)
This command returns from 1 to 60 registers. This command is used for reading one or more data locations from nCompass.

Packet sent to nCompass:


nn
03
nn nn
00 nn
nn nn
controller address (1 byte)
read command (0x03)
starting register high byte
starting register low byte
number of registers high byte (0x00)
number of registers low byte
CRC low byte
CRC high byte

Packet returned from nCompass:


nn
03
nn
nn nn … nn nn
nn nn
controller address (1 byte)
read command (0x03)
number of data bytes (1 byte)
first register data low byte
first register data high byte
…
…
register n data high byte
register n data low byte
CRC low byte
CRC high byte

Example: Read registers 35 and 36 (loop 1 process variable and setpoint) of controller at address 1 configured for 1 decimal point.

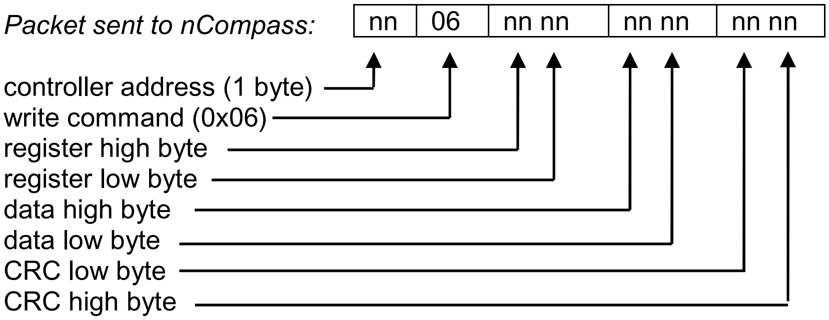
Sent: 01 03 00 23 00 02 35 C1
Received: 01 03 04 **03 0D 01 F3** 2A 61

Message data: 781 (0x**030D**) = process variable of 78.1
499 (0x**01F3**) = setpoint of 49.9

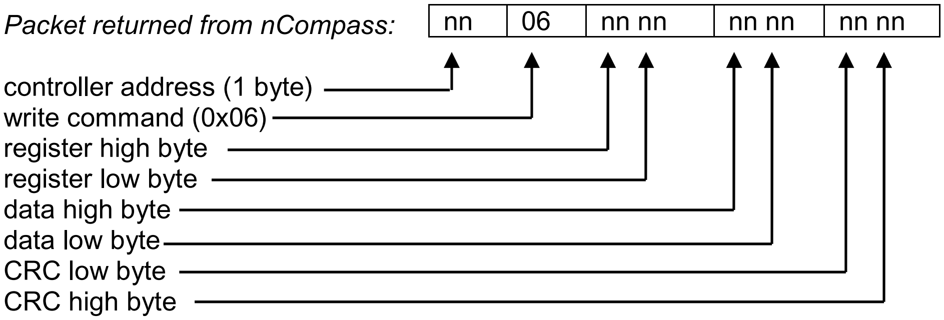
Write Register Command (0x06)

This command writes a value to a single register. This command is to be used for setting control values in nCompass. To set multiple values, repeat the command for each data location.

Packet sent to nCompass:



Packet returned from nCompass:



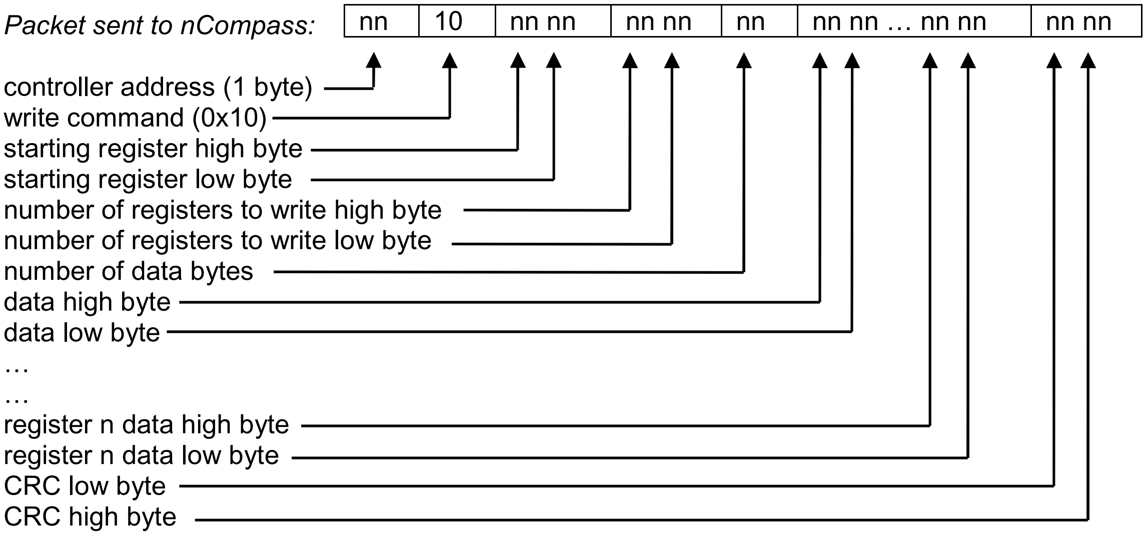
Example: Write register 41 (loop 2 set point) of controller at address one configured with no decimal point to 75 degrees (0x**004B**).

Sent: 01 06 00 29 **00 4B** 18 35
Received: 01 06 00 29 00 4B 18 35

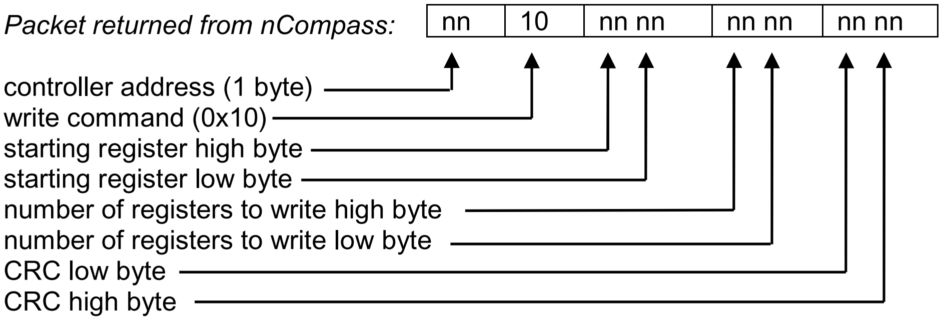
Write Registers Command (0x10)

This command writes values to multiple registers in sequential order. It is used for automatic program download only to transmit program data one step at a time to nCompass. See the Automatic Program Parameters section for the list of registers and their use. If this command is used to write to registers other than the correct program step registers, nCompass will respond with an acknowledgment that the message was received; however, the command will not be executed.

Packet sent to nCompass:


nn
10
nn nn
nn nn
nn
nn nn … nn nn
nn nn
controller address (1 byte)
write command (0x10)
starting register high byte
starting register low byte
number of registers to write high byte
number of registers to write low byte
number of data bytes
data high byte
data low byte
…
…
register n data high byte
register n data low byte
CRC low byte
CRC high byte

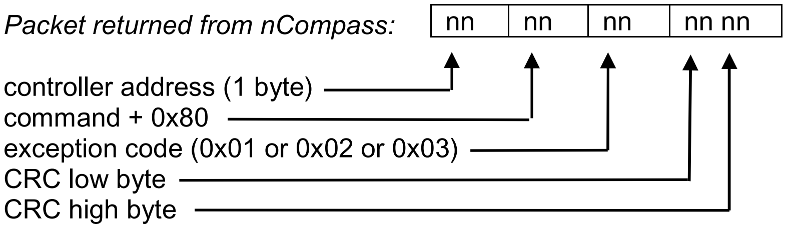
Packet returned from nCompass:


nn
10
nn nn
nn nn
nn nn
controller address (1 byte)
write command (0x10)
starting register high byte
starting register low byte
number of registers to write high byte
number of registers to write low byte
CRC low byte
CRC high byte

Exception Responses

When nCompass cannot process a command, it returns an exception response and sets the high bit (0x80) of the command.

0x01	illegal command
0x02	illegal data address
0x03	illegal data value



Packet returned from nCompass:

nn	nn	nn	nn nn

controller address (1 byte)
command + 0x80
exception code (0x01 or 0x02 or 0x03)
CRC low byte
CRC high byte

4.3.2 Error Checking

In Modbus communications, every message sent from the master (your software) receives a response from the slave (nCompass), including write commands. Thus, after each command sent, you must read the controller response before sending the next message. This provides the method of error checking in order to verify that the message you sent was received correctly, and that the controller is operating accordingly. This allows you to then determine the appropriate recovery response in case the message was not received correctly by the controller, and what action is to be taken by an operator and/or the software itself.

The exception responses provide a basic form of error checking. When an exception response is received, the code provided in the response will tell you what the error was in the sent message. However, this is only valid if the controller receives the message you sent, and there was an out-of-range value or simple transmission error in the message. It does not validate incomplete or failed transmissions. To insure that the data you receive from a read command is correct, and that the controller properly received a write command, you must parse the controller's response and validate the return message to insure it is correct.

In order to validate that the message you received is correct, you must calculate the CRC for the received message and compare it with the CRC that the controller appended to the message. This verifies that the data you received was what nCompass sent. If the CRC's do not match, there was an error in the transmission and the entire message should be ignored. This could then be followed by an attempt to resend the failed command, or halt operation and alert an operator.

Example: Read registers 35 and 36 (loop 1 process variable and setpoint) of controller at address 1.

Command sent to nCompass:	01 03 00 23 00 02 35 C1
Message received from nCompass:	01 03 04 03 0D 01 F3 2A 61

Calculated CRC: 2A61 (calculated from message 01 03 04 03 0D 01 F3)
Received CRC: 2A61

The calculated CRC matches the received CRC, the message is valid. Note that the last two bytes of the received message are not used to calculate the CRC. The last two bytes are the CRC that nCompass appended to the message. Do not include them when calculating the CRC.

4.3.3 Transmitting and Receiving Messages

In order to reliably communicate with nCompass, it is important to develop an efficient means of transmitting and receiving messages. Modbus is a structured protocol and it must be properly followed. It is recommended, if possible, to locate an existing communication driver to incorporate into your software. Developing one from scratch can be challenging. However, if one is not available, or you choose to develop one yourself, the following guidelines may be of assistance.

Transmitting Messages

When sending a message to nCompass, it is important to remember that Modbus RTU protocol does not have start-of-transmission or end-of-transmission characters. All messages are “framed” using timeouts between characters. nCompass uses its own fixed timeout setting of 135ms. Thus, if the entire message is not received by nCompass within 135ms, it will discard the data it has received and assume the next data byte received is the start of the next valid message.

The timeout must be considered carefully when developing your application. In creating your message, there are several steps that must be executed in order to build the packet and format the data properly into hexadecimal to send out the serial port of your PC. If you write code in a manner that steps byte by byte through sending the message out the serial port, formatting each piece of data prior to sending it, there is a good possibility that two much time may pass between characters, thus causing a failed transmission.

Therefore, it is recommended that the entire message, including the CRC, be created and assembled prior to being sent to the serial port. By assembling the main body of the message first, you can then pass it to the CRC algorithm which can step sequentially through the message, generate the CRC and append it to the message body. Once the message is completely assembled, it can then be sent out the serial port as a completed packet. This will insure that the message reaches nCompass within the proper time frame.

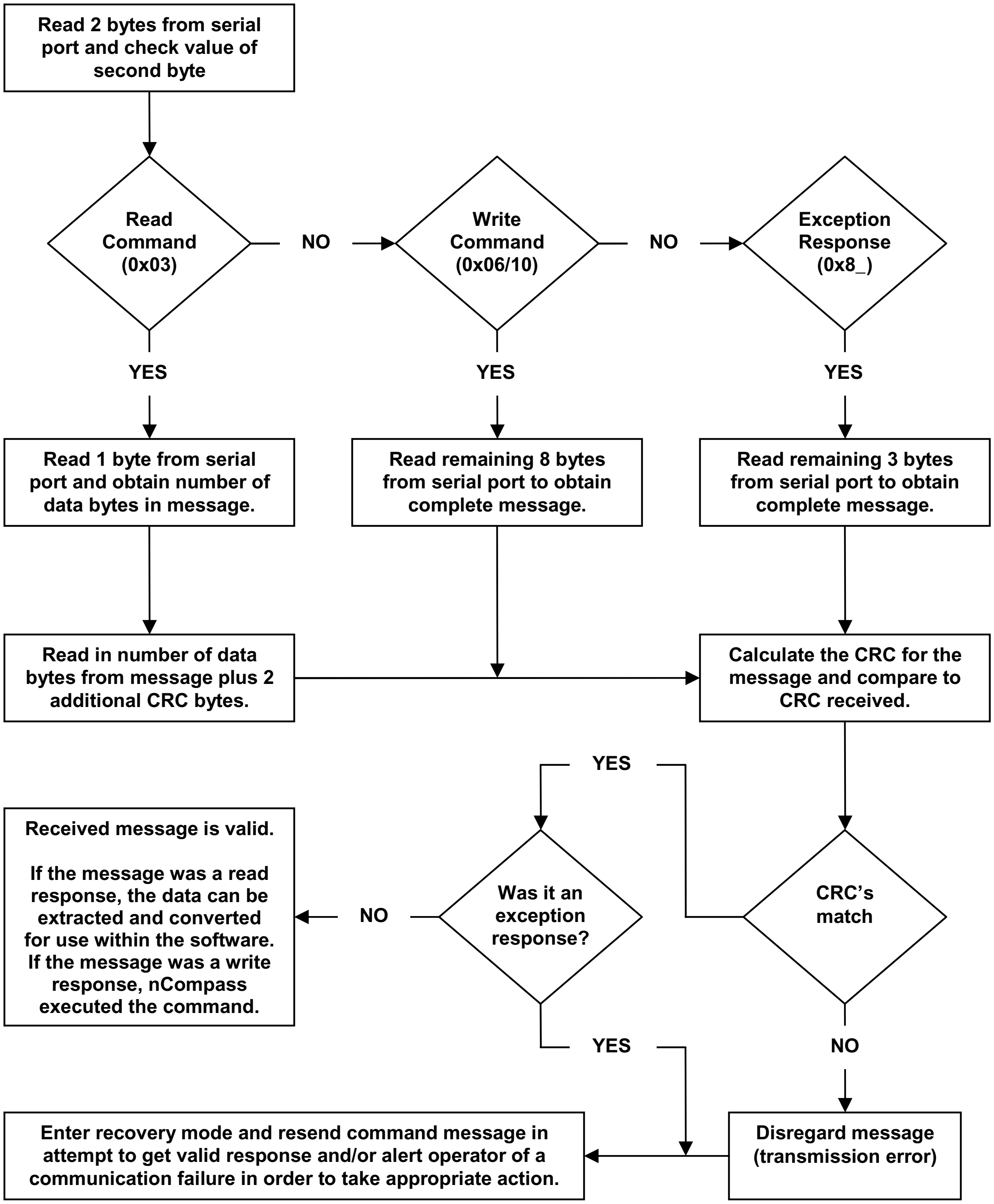
Receiving Messages

Due to the fact that Modbus RTU protocol does not have start-of-transmission or end-of-transmission characters, if the serial port driver you are using does not support an interval timeout setting allowing you to automatically terminate a read after a specified time passes between bytes (signaling the end of a message), you must know how long the message will be that you are receiving. That allows you to know how many bytes to read from your serial port and when you have received the entire message. If you rely on a maximum timeout period to terminate the read, depending upon the length of the received message, you will either loose a portion of the message or have to set the timeout period so high, that it will greatly affect the throughput of your code.

As can be seen from the previous examples for read and write commands in Section 4.3.1, the length of the returned message will very based on the type of command, and for read commands, how many registers are being returned. Response messages can vary in length from as little as 5 bytes for an exception response to as many as 125 bytes for a read command. Therefore, in order to read in the message efficiently, you need to know what type of command it is in response to.

The response messages are always coded with the first two bytes of the message as the controller address and command type. When executing a read, read in only the first 2 bytes of data at the serial port. Examine the second byte and determine what the command is. If it is a write command (0x06 or 0x10), you know the response message is 8 bytes long. You can then read in the next 6 bytes of data from the serial port to complete the message. You can then calculate the CRC for the first 6 bytes of that message, and compare it to the last 2 bytes. If they match, then the communication completed successfully.

If the response is to a read command (0x03), you must then perform a single byte read from your serial port in order to get the next byte of the message. The third byte in a read response message is the number of data bytes in the message. By reading in this value, you then know how many data bytes follow. Note that this value does not include the 2 bytes for the CRC. Thus, when reading in the rest of the message, you will read in the number of data bytes plus an additional two, in order to get the CRC. You can then calculate the CRC for the message and compare it to the last two bytes. If they match, the data you received is valid.


Read 2 bytes from serial port and check value of second byte
Read Command (0x03)
NO
Write Command (0x06/10)
NO
Exception Response (0x8_)
YES
YES
YES
Read 1 byte from serial port and obtain number of data bytes in message.
Read remaining 8 bytes from serial port to obtain complete message.
Read remaining 3 bytes from serial port to obtain complete message.
Read in number of data bytes from message plus 2 additional CRC bytes.
Calculate the CRC for the message and compare to CRC received.
YES
Received message is valid.
If the message was a read response, the data can be extracted and converted for use within the software. If the message was a write response, nCompass executed the command.
NO
Was it an exception response?
CRC's match
YES
NO
Enter recovery mode and resend command message in attempt to get valid response and/or alert operator of a communication failure in order to take appropriate action.
Disregard message (transmission error)

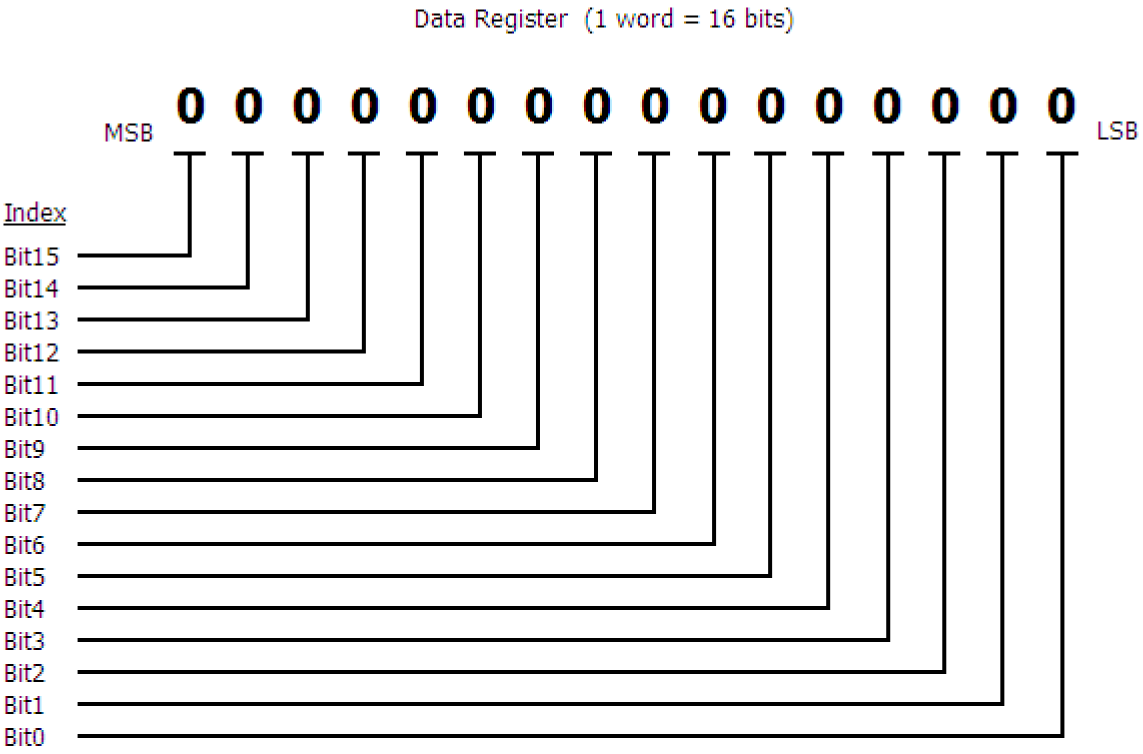
.

5 nCompass MC Loop Controller Data Registers

Some of the values contained in the nCompass loop controller register base contain bit oriented values. This means that each bit of the word indicates an on/off status for a specific setting or condition. In handling these values, it is recommended that the word be converted to its binary equivalent.

By converting the value to its binary equivalent, it produces a Boolean array of true [bit on (1)] and false [bit off (0)] values. This allows each bit to be examined individually. In the same manner, creating a Boolean array of 16 bits produces an equivalent hexadecimal value that can be sent to nCompass in order to set a control register.

For the purpose of this manual, parameters defined as bit oriented will have the function of each bit associated with the bit's index number in the data word. The index number is equal to that of a typical array function. Thus, an index number of zero, selects the first bit in the word (LSB). An index number of 1 selects the second bit in the word, and so on. This helps eliminate offset selection errors that may occur when coding software and using array functions to select which bit in the word that is required for examination.



⚠ *Adhere to the following lists of registers and their allowable data ranges. DO NOT attempt to write to any other register number than those listed. DO NOT write to registers that are for options your controller does not have. Failure to adhere to this requirement can result in erratic control and/or damage to equipment.*

<u>All register numbers are listed in relative values. To convert to absolute values, add 400001.</u>

5.1 Control Registers

Register Address	Parameter Description	Data *A Type	Range *B		*C
			Low	High	Unit
0 **(0x0000)**	Controller Set Point (Static Mode)	W	-32768	32767	*C1
1 **(0x0001)**	Current Program/Segment Number	R/W	*B1	*B1	-
5 **(0x0005)**	Set Point for Alarm 1	R/W	-32768	32767	*C1
6 **(0x0006)**	Set Point for Alarm 2	R/W	-32768	32767	*C1
7 **(0x0007)**	Set Point for Alarm 3	R/W	-32768	32767	*C1
9 **(0x0009)**	Input Unit Selection	R	*B2	*B2	-
10 **(0x000A)**	Decimal Point Selection	R	0	3	-
11 **(0x000B)**	Operation Mode	R/W	*B3	*B3	-
127 **(0x007F)**	Manual Event Output	R/W	*B4	*B4	-
128 **(0x0080)**	Process Value (PV)	R	-32768	32767	*C1
129 **(0x0081)**	Set Point Value (SV) for Control	R	-32768	32767	*C1
130 **(0x0082)**	Output 1 Percentage Value (MV1)	R/W	0.00	100.00	%
131 **(0x0083)**	Output 2 Percentage Value (MV2)	R/W	0.00	100.00	%
132 **(0x0084)**	Mode and Operational Status	R	*B5	*B5	-
133 **(0x0085)**	Error Code	R	*B6	*B6	-
134 **(0x0086)**	Current Program and Segment Running	R	*B1	*B1	-
135 **(0x0087)**	Total Number of Steps	R	1	64	-
136 **(0x0088)**	Total Time for Step Running	R	0	99.59	-
137 **(0x0089)**	Set Point for Current Step	R	-32768	32767	*C1
138 **(0x008A)**	Time Remaining in Current Step	R	0	99.59	-
139 **(0x008B)**	Cycles Remaining for Current Loop	R	1	9999 10000=infinite	-
142 **(0x008E)**	Command Code	W	*B7	*B7	-

Notes:

*A R/W Specifies readable/writable data, R specifies read only data and W specifies a write only control value.

*B The range of certain parameters are dependent upon system options. Consult the following range tables for information regarding the use of these parameters.

Reading bit oriented parameters

The value contained in these parameters is dependant upon the combination of "on" bits (1). Therefore, only the individual status of each bit has meaning, not the value of the parameter.

Setting bit oriented parameters

The value that must be written to these parameters is dependant upon the combination of “on” bits. Therefore, it is necessary to know the current value of the parameter before setting it so that only the bit status you want to update is changed. Otherwise, sending a value derived from only the bit you wish to set will turn off all other functions related to the other bits in the parameter.

*B1

Parameter Value	
Range Low	9.00
Range High	9.63

Note: nCompass uses program 9 only in each loop control for automatic ramp/soak program operation. The “whole number” of the value (9) designates program 9. The fractional portion of the value (.00-.63) designates the segment number of the program which corresponds to steps 1 to 64.

Example

Values read from and written to nCompass are integer values (no decimal) so the decimal point is implied. The values read from and written to nCompass will be in the range of 900 to 963. To set the program step number to 3, you would write a value of 902 to nCompass.

902 = 900 (for program number) + 2 (step number of 3 minus 1 to convert step number to segment number)

*B2

Parameter Value	Description
0	Centigrade
1	Fahrenheit
2	Process Units (linear input type only)

*B3

Parameter Value	Description
0	Program Run
1	Program Hold
2	Static Mode
3	Automatic Tune
5	Manual Mode

Note: Auto tune may not be available depending upon the loop configuration. If auto tune operation is not available, the loop controller will revert back to static mode. When auto tune operation completes normally, the loop controller will revert back to static mode.

Only one mode can be set at time. The program hold mode can not be set unless a program is currently running. The program mode can then be switched between run and hold by setting a value of 0 or 1 to the operation mode register.

When the loop control is in manual mode, the output percentage values for output 1 (MV1) and output 2 (MV2) can be set by writing to the corresponding output register.

*B4

Parameter Value	Description
Bit0	Event 1 (output 2)
Bit1	Event 2 (output 3)
Bit2	Event 3 (output 4)
Bit3 – Bit15	Not Assigned

Note: The events provided by nCompass are a combination of all events provided by all loop controls combined. In order to turn events on and off, the events assigned to each specific control must be accessed. The proper event bits must be turned on in each corresponding loop control.

The first event in nCompass may actually be output 3 of loop controller number 2. This requires bit 1 of the event register to be set on loop controller 2 of nCompass.

*B5

Parameter Value	Description
Bit0	Automatic Program Run Mode
Bit1	Automatic Program Hold Mode
Bit2	Static Mode
Bit3	Auto Tune Mode
Bit4	Manual Mode
Bit5	Off Mode
Bit6	Failure Transfer
Bit7	Automatic Program Ramp Up
Bit8	Automatic Program Ramp Down
Bit9	Automatic Program Soak
Bit10	Alarm 1 Active
Bit11	Alarm 2 Active
Bit12	Alarm 3 Active
Bit13	Event 1 On
Bit14	Event 2 On
Bit15	Event 3 On

*B6

Parameter Value	Description
0	No Error
4	Illegal Setup Values
10	Comm Error – Bad Function Code
11	Comm Error – Register Out of Range
14	Comm Error – Write Read Only Data
15	Comm Error – Out of Range Data
25	Holdback Time Out
26	Auto Tune Error
27	Input Type Requires Calibration
29	EEPROM Error
30	Cold Junction Failure
39	Sensor Break
40	A to D Failure

*B7

Parameter Value	Description
26660	Reset Single Set Point (Static Mode)
26661	Reset Alarms

Note: Upon completion of an automatic ramp/soak program, the loop control will not return control to the static set point until the “reset single set point (static mode)” command value is written to the loop control. Even though the static set point can be changed, the loop controller will continue to use the last program set point until the reset command is received.

*C1 The units of measure and range of a loop input is dependant upon the configuration of the input and/or the units of temperature selection (Celsius or Fahrenheit) of the nCompass loop controller. The decimal point position for the value is an implied value based on the configuration of the input. Thus, a register value of 345 can represent an actual process value of 345, 34.5, 3.45 or 0.345 depending upon the decimal point configuration of the loop.

5.1.1 Communicating with Loop Controllers on nCompass

The loop control register data table provides the list of available data for each loop controller attached to nCompass. In order to read the process value of each loop for example, you must send separate read commands addressed specifically to each controller.

The loop controls are addressed in sequential order starting with the Modbus address entered on the communications settings screen on nCompass. If the Modbus address is set to 3 and there are 5 loop controls attached to nCompass, the loop control addresses would be 3, 4, 5, 6 and 7. Sending the read register command to nCompass to read register 128 with an address of 5 would result in nCompass returning the process value of loop number 3. The third loop of nCompass based on the example, responds to the Modbus address of 5.

For an nCompass system of 1 to 10 loops, simply changing the address of the command allows you to send or receive the same data from the next loop controller. Utilizing a read register command starting at register 127, and reading a total of 13 registers, allows you to retrieve the current operating status of the loop control in a single command. By repeating the command for each loop attached to nCompass, your software can continually monitor the status of all loops attached to nCompass quickly and efficiently.

The design of the serial interface also allows the nCompass MC to connect to FDC'S Envision software. The software is free of charge and provides a host of features and is compatible with all FDC products. Using the P-Series controller selection in EnVision, each loop controller of the nCompass MC can be configured in EnVision as individual P-Series controls. This allows each one to be independently monitored and controlled over the nCompass interface.

5.2 Automatic Program Registers

The automatic program parameters are a separate group of registers that are used to download programs to each individual loop controller attached to nCompass. The manner in which the program steps are configured and sent to nCompass is specific and must be followed exactly.

Each program consists of a "header step" of 5 registers and up to 64 steps consisting of 10 data registers. The program must be written one step at a time, using a multiple write command (0x10) to write all the data for each step at once. This allows programs to be stored as two-dimensional arrays, of which code can be written to simply index through the array step-by-step, and transmit the program file to nCompass.

The first 5 registers of the program (header step) contain specific settings related to the program. These include the program number, hold back band, ramp units and dwell units.

Register Address		Parameter Description	Data *D Type	Range *E Low	High	*F Unit
86	**(0x0056)**	Program Selected for View	W	9	9	-
87	**(0x0057)**	Hold Back Band	W	*E1	*E1	-
88	**(0x0058)**					
89	**(0x0059)**	Ramp Units	W	*E2	*E2	-
90	**(0x005A)**	Dwell Units	W	*E3	*E3	-

The following 10 registers of the automatic program contain the data for step 1 of the program.

Register Address		Parameter Description	Data D Type	Range *E Low	High	*F Unit
91	**(0x005B)**	Step Number*	W	0	63	-
92	**(0x005C)**	Step Type	W	*E4	*E4	-
93	**(0x005D)**	Ramp Target Setpoint	W	-32768	32767	PV
94	**(0x005E)**	Ramp Time or Ramp Rate	W	0	5999	-
95	**(0x005F)**	Event Selections 1, 2, 3	W	*E5	*E5	-
96	**(0x0060)**	Holdback Type	W	*E6	*E6	-
97	**(0x0061)**	Dwell Time	W	0	9999	-
98	**(0x0062)**	Jump Step	W	0	63	-
99	**(0x0063)**	Jump Cycles	W	1	9999	-
100	**(0x0064)**	Final Setpoint	W	-32768	32767	PV

The **Step Number must be offset by 1 when writing step data to nCompass. Steps 1-64 will be sent with the step number as a value of 0-63 (-1 offset).

All remaining steps of the program follow the same format and data structure as is represented for step one above. Each step of the program is downloaded to the same registers (91 to 100). The step number in register 91 designates the step that the data is for on each write command. Since few if any programs will contain the maximum of 64 steps, it is only necessary to write the step data for the number steps used in the automatic program.

Notes:

*D W Specifies write only data.

*E1

Parameter Value	Temperature Units °C	Temperature Units °F	Process Units
Range Low	1	1	1
Range High	555	999	999

*E2

Parameter Value	Description
0	Hours and Minutes
1	Minutes and Seconds
2	Units/Minute
3	Units/Hour

*E3

Parameter Value	Description
0	Hours and Minutes
1	Minutes and Seconds

*E4

Parameter Value	Description
0	Ramp
1	Soak
2	Jump
3	End

Note: The last step of an automatic program must be an end step. If the last step is not an end step, the program will not run correctly and/or a program download error will occur and the program will not operate.

*E5

Parameter Value	Description
Bit0	Event 1
Bit1	Event 2
Bit2	Event 3
Bit3 – Bit15	Not Assigned

Note: The available system events are based on the configuration of the specific loop controller that the program is being downloaded to.

*E6

Parameter Value	Description
0	Holdback Disabled
1	Deviation Low Holdback
2	Deviation High Holdback
3	Deviation Band Holdback

*F The unit PV means that the unit of the parameter is the same as the unit of PV (the loop configuration).

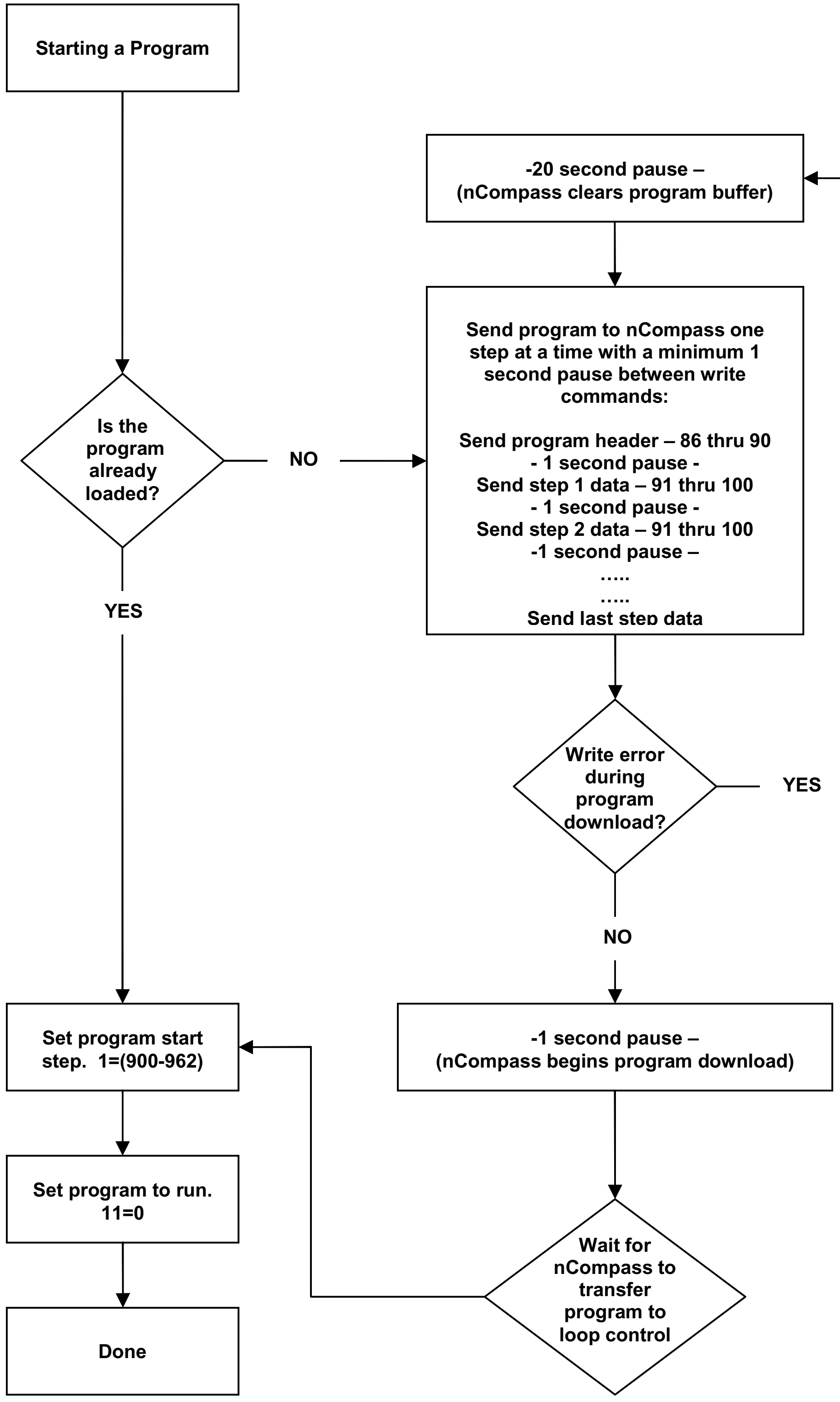
Use caution when loading an automatic program to nCompass. You must insure that the decimal point scaling and units of measurement in the program match the loop setting. Loading a program with a temperature setpoint of 80 will result in a control temperature of 80°F if the nCompass control is in degrees Fahrenheit. However, if the nCompass control is set for degrees Centigrade, it will result in a control temperature of 80°C (176°F).

5.2.1 Starting an Automatic Program on nCompass

Automatic programs are sent to nCompass in a step-by-step process. The download sequence must be followed in proper order and must complete without errors to be valid. If a write error is detected during the transfer of a program from a PC to nCompass (no response from nCompass or NACK returned), the program download must be aborted and restarted.

nCompass is automatically placed into program transfer mode when the header step of a program is first sent (registers 86-90). nCompass then begins looking for the following program steps to be sent. Once nCompass sees the end step, it then downloads the program to the loop controller assigned to the Modbus address that the program steps were written to.

If an error occurs during the transfer process from the PC to nCompass, the program transfer process should be stopped at the PC. The data sent to nCompass was either corrupted in transmission or not received properly. It is not possible to resend the failed step because it is not known if any of the previous data was received by nCompass properly. On the transmission error, nCompass will enter a 15 second timeout process. At the end of the timeout period, the buffer will be cleared and the program can be resent. In order to insure that the new download begins properly, induce a 20 second wait period on the host PC after the failed transmission attempt to insure that enough time has elapsed.


Starting a Program
Is the program already loaded?
NO
YES
-20 second pause –
(nCompass clears program buffer)
Send program to nCompass one step at a time with a minimum 1 second pause between write commands:
Send program header – 86 thru 90
- 1 second pause -
Send step 1 data – 91 thru 100
- 1 second pause -
Send step 2 data – 91 thru 100
-1 second pause –
.....
.....
Send last step data
Write error during program download?
YES
NO
-1 second pause –
(nCompass begins program download)
Wait for nCompass to transfer program to loop control
Set program start step. 1=(900-962)
Set program to run. 11=0
Done

Appendix

Terms and Definitions

address – A unique designator for a location of data or a controller that allows each location or controller on a single communications bus to respond to its own message.

ASCII (pronounced AS-KEY) – American Standard Code for Information Interchange. A universal standard for encoding alphanumeric characters into 7 or 8 binary bits.

Asynchronous – Communications where characters can be transmitted at an unsynchronized point in time. In other words, it can start and stop anytime. The time between transmitted characters may be of varying lengths. Communication is controlled by “start” and “stop” bits at the beginning and end of each character.

Baud – Unit of signaling speed derived from the number of events per second (i.e., bits per second).

Baud rate – The rate of information transfer in serial communications, measured in bits per second.

Binary – Number based system where only two characters exist, 0 and 1. Counting is 0, 1, 10, 11...

Bit – Derived from “**Bi**nary digi**t**”, a one or zero condition in the binary system.

Byte – A term referring to eight associated bits of information, sometimes called a “character”.

Character – Letter, numeral, punctuation, control figure or any other symbol contained in a message. Typically this is encoded in one byte.

Communications – The use of digital computer messages to link components. (See serial communications and baud rate)

Converter – This device will convert from one hardware interface to another such as from EIA-232 to EIA-485. The converter may be transparent to the software, which means you do not have to give any special considerations to software programming.

CRC – When data is corrupted during transmission, a method is used to return the data to its correct value. This can be accomplished through several methods: parity, checksum and CRC (cyclic redundancy checksum) are three of these. **C**yclic **R**edundancy **C**hecksum is an error-checking mechanism using a polynomial algorithm based on the content of a message frame at the transmitter and included in a field appended to the frame. At the receiver, it is then compared with the results of the calculation that is performed by the receiver.

Data – The information that is transferred across the communications bus. This may be a setpoint, setup parameter, or any character. This information is transferred to an address or register.

DB-9 – A standardized connector shaped like the letter “D” when viewed on edge. This connector has 9 contacts. It is utilized on most IBM AT compatible PCs as the serial port.

Decode – This is the reverse of encode. When a piece of data has information embedded in it, decode is to extract that information. Example: to extract an “A” from 01000001.

Duplex – The ability to send and receive data at the same time. “To listen and talk at the same time.”

EIA-232 – Electronic Industries Association developed this standard hardware interface to allow one device to talk to another device in full duplex mode. This method uses a differential voltage between one wire and ground. Also called an unbalanced system since the ground wire carries the sum of current of all lines. Transmission is limited to about 50 feet.

EIA-485 – Electronic Industries Association developed this standard hardware interface to allow up to 32 devices to be on a bus at one time. This method uses a differential voltage between two wires. Also called a balanced system since each wire carries the same current value. This has the advantage of being immune to outside electrical disturbances.

EIA/TIA -232 and -485 – Data communications standards set by the Electronic Industries Association and Telecommunications Industry Association. Formerly referred to as RS- (Recommended Standard). (See EIA-232 and EIA-485)

Electronic Industries Association (EIA) – An association in the US that establishes standards for electronics and data communications.

Encode – To embed information into a piece of data. This is the reverse of decode. Example: let 01000001 stand for an “A”.

Error Correction – When an inconsistency is in the data, a method is used to detect and/or return the data to its correct value. This can be done through several methods, parity, checksum and CRC (cyclic redundancy checksum) area three of these.

Even – This term is used with parity. See parity.

Firmware – Instruction or data stored in an IC (integrated circuit) or on a read only disk. This data is programmed once and cannot easily be changed as software can.

Full Duplex – Full is used to mean the duplex’s full capability. The ability to send and receive data at the same time. The same as duplex.

GPIB – See IEEE488

Half Duplex – The ability to send or receive data, but not at the same time. “To listen or talk, but not both at the same time.”

Handshake (Handshaking) – Exchange of predetermined signals between two devices establishing a connection. Using extra wires or software signals to coordinate communications, signals can be sent to tell the transmitter the current status of the other device receiver. Example: Are you busy or are you ready?

Hex or Hexadecimal – Number based system where sixteen characters exist, 0 to 9, A to F. Counting is 0..9,A,B,C...

Integer – Two bytes make an integer. This contains 16 bits. An integer can represent a decimal value of -32768 to 32767.

Logic Level – A voltage measurement system where only two stable voltage values exist. Example: 0v and 5V, or -3v and +3v.

Mark – Represents the transmission of data bit logic 1 (see logic level). Usually this is the most negative voltage value in serial communications.

Master – The device on the bus that controls all communications. Only the master can initiate conversation.

Modbus – A software protocol developed by Gould Modicon (now AEG) for process control systems. No hardware interface is defined. Modbus is accessed on the master/slave principle, the protocol providing for one master and up to 247 slaves. Only the master can initiate a transaction. This is a half duplex protocol.

Network – When two or more devices share communication lines, the devices are “networked”.

Node – A point of interconnection to a network.

Noise Immunity – The ability of communication lines to ignore electrical noise generated in the lines by nearby magnetic and electrostatic fields.

Odd – This term is used with parity. See parity.

Parallel – Communication using this method, transfers eight bits or one byte at a time over eight data wires and one ground wire. This method is eight times faster than using serial but utilizes more hardware.

Parity – A bit is assigned at the beginning of a byte to stand for parity. When the ‘1’ bits are counted, the number will be even or odd. A parity bit is used to ensure that the answer is always even if even parity or odd if odd parity. If the receiving end counts the ‘1’ bits and the sum is not the same odd or even, an error is generated. Parity is used to detect errors caused by noise in data transmission.

Protocol – A set of rules for communication. This will specify what method to transfer information, packet size, information headers and who should talk when. It is used to coordinate communication activity.

Receive – To accept data sent from another device. The device that receives the data is the receiver.

Register – An area of memory that provides temporary storage of digital data.

Slave – A device that only responds to commands. This device never starts communication on it’s own. Only the Master can do this. (See Master)

SCADA – Supervisory Control and Data Acquisition

Serial – To process something in order. First item, second item, etc.

Serial Communications – A method of transmitting information between devices by sending all bits serially (see serial) over a single communication channel.

Software – Information of data or program stored in an easily changeable format. (RAM, Floppy Disk, Hard Disk)

Space – Represents the transmission of a data bit logic 0 (see logic level). Usually this is the most positive voltage value in serial communications.

Start Bit – A binary bit or logic level that represents when the serial data information is about to start (at the beginning of a character or byte). This voltage level is positive.

Stop Bit – A binary bit or logic level that represents when the serial data information is complete (at the end of a character or byte). This voltage level is negative.

Synchronous – When data is transmitted on a data line and a clock signal is used on another line to determine when to check the data line for a logic level. This clock is said to “synchronize” the data.
Transmit – To send data from one device to another. The device that sends the data is the transmitter.

Word – Two bytes make a word. This contains 16 bits. A word can represent a decimal value of 0 to 65535.

How to Order

nCompass MC is ordered as 3 to 12 components dependant upon the number of control loops, one to ten.

nCompass MC Sample Part Numbers

(minimum of 3 component part numbers is required; (1) display, (2) firmware, (3) loop #1)

Item #	Product	Sample Part Number	Description
1	Display	FDC-0450-1011-000BN	4.3" display, MC software, 11-36 VDC power input, SD slot, Ethernet, standard enclosure
2	MC Firmware	SD-MC	MC Loop Control Firmware (inserted into display SD slot)
3	Loop #1	B42-4166-11D0	90-250VAC power input, T/C or RTD input, output #1 1A Triac, output #2 1A Triac, output #3 relay, output #4 relay, comms, no display
• • •			
12	Loop #10	B42-4366-11D0	90-250VAC power input, mA input, output #1 1A Triac, output #2 1A Triac, output #3 relay, output #4 relay, comms, no display
	Options		
	Power Supply	PS5R-SD24	85-264VAC power input, Output 24VDC 15W (0.6A)
	Cable	CA2011-3D	Cable from display to B42 (DB25 connector (display), twisted pair leads (B42), shielded (10ft)

Item #1 Model FDC - 0450 4.3" Color Touch Screen

FDC - 0450 -	1	0	1	1	0	0	0	B	N
Order Matrix #	1	2	3	4	5	6	7	8	9

(1) Power Input
1: 11 to 36 VDC

(2) Sound Output
0: None

(3) SD Card Slot
1: Yes

(4) Ethernet
1: Yes

(5) Network
0: None

(6) Software
0: None

(7) Enclosure
0: Standard

(8) Overlay Color
B: Black Overlay

(9) Special Code
N: Neutral Overlay (no name/logo)

Item #2: Loop Control Firmware

Order Matrix #	SD	-	MC

(Fixed characters SD-MC: SD card with MC (1 to 10 Loop Control) Display Configuration and SD plugged into display)

Control System Options *(ordered separately as appropriate)*

Cable: Display to B42 Control Board
CA2011-8D: Cable (10ft) from Display to B42 controller.

Note: Consult factory for other lengths & options

Display Power Supply (input 100-240VAC / Output 24VDC)

DIN Rail Mount:	PS5R-SB24:	15W power supply (0.6A)
Open Frame:	PS3X-B24AFC:	15W power supply (0.6A)

Note: Either of the above will power the FDC-0450 touch screen display

USB Memory Stick
UDF115-2GB: (2GB High Capacity USB Memory Stick (3VDC))

System Reset Timer
GE1A-C10MA110/SR2P-06: Reset Timer and socket (DIN Rail)

Note: Timer is recommended for proper system restart due to momentary power interruptions (<500ms) which can erroneous operation.

B42 Programming Display Board

B42 Display Board 300 mm cable:	3020B42-00300A-00/300
B42 Display Board 1000mm cable:	3020B42-00300A-00/1K

Printed Operators Manual
Part Number (Orion-nCompass_MC_i4.3_User_Manual.pdf)
FDC-Orion nCompass MC i4.3 User Manual

Printed Configuration Manual
Part Number (Orion-nCompass_MC_i4.3_Config_Manual.pdf)
FDC-Orion nCompass MC i4.3 Configuration Manual

Printed Communication Reference Manual
Part Number (Orion-nCompass_MC_i4.3_Comm Reference.pdf)
FDC-Orion nCompass MC i4.3 Communication Reference Manual

USB Cables & Accessories

IStick-Panel	USB Panel Mount Adapter
IStick-4X-CVR	USB Panel Mount Adapter-Nema4x

Item #3: B42 Loop Control Board - Loop 1

B42 -	□	□	□	□	□	□	D	□	□
Order Matrix #	1	2	3	4	5	6	7	8	9

(1) Power Input

4: 90-250 VAC, 47-63 HZ
5: 11-26 VAC or VDC *(consult factory for availability)*

(2) Signal Input (18-bit A/D)

1: Standard Input
Thermocouple: J, K, T, E, B, R, S, N, L, C, P
RTD: PT100 DI, PT100 JIS
Voltage: 0-60mV
5: 0-10V, 0-1V, 0-5V, 1-5V
6: 0-20/4-20mA
9: Special Order

(3) Output 1

0: None
1: Relay rated 2A/240VAC (SPST)
2: SSR Driver 5 VDC @ 30 Ma
3: 4-20mA / 0-20mA Isolated
4: 1-5V / 0-5V/0 - 10V Isolated
6: Triac output 1A / 240VAC,SSR
C: SSR Driver 14 VDC @ 40 Ma

(4) Output 2

0: None
1: Relay rated 2A/240VAC (SPST)
2: SSR Driver 5 VDC @ 30 Ma
3: 4-20mA / 0-20mA Isolated
4: 1-5V / 0-5V/0-10V Isolated
6: Triac output 1A / 240VAC,SSR
7: Transmitter power supply 20 VDC/25 ma Isolated
8: Transmitter power supply 12 VDC/40 ma Isolated
A: Transmitter power supply 5 VDC/80 ma Isolated
C: SSR Driver 14 VDC @ 40 Ma

(5) Output 3

0: None
1: Relay rated 2A/240VAC (SPST)
2: SSR Driver 5 VDC @ 30 Ma
6: Triac output 1A / 240VAC,SSR
7: Transmitter power supply 20 VDC/25 ma Isolated
8: Transmitter power supply 12VDC/40 ma Isolated
A: Transmitter power supply 5VDC/80 ma Isolated
C: SSR Driver 14 VDC @ 40 Ma

(6) Output 4

0: None
1: Relay rated 2A/240VAC (SPST)
2: SSR Driver 5 VDC @ 30 Ma
3. Retransmission 4-20 / 0-20mA, isolated
4. Retransmission 1-5 / 0-5 / 0-10VDC, isolated
6: Triac output 1A / 240VAC,SSR
7: Transmitter power supply 20 VDC/25 ma Isolated
8: Transmitter power supply 12VDC/40 ma Isolated
A: Transmitter power supply 5VDC/80 ma Isolated
C: SSR Driver 14 VDC @ 40 Ma

(7) Output 5

D: RS-485 Modbus RTU interface Isolated

(8) Display Board and cable

0: None
3: Display Board with 300mm connection cable
4: Display Board with 1000mm connection cable

(9) Modbus Addressing: (if blank factory default is Modbus address #1)
CA: Modbus Comm Address #1
CB: Modbus Comm Address #2
CC: Modbus Comm Address #3
Note that field configuration of Comm address requires the optional display.

Item #4-12: B42 Loop Control Board - Loop 2-10

B42 -	□	□	□	□	□	□	D	□	□
Order Matrix #	1	2	3	4	5	6	7	8	9

(1) Power Input

4: 90-250 VAC, 47-63 HZ
5: 11-26 VAC or VDC *(consult factory for availability)*

(2) Signal Input (18-bit A/D)

1: Standard Input
Thermocouple: J, K, T, E, B, R, S, N, L, C, P
RTD: PT100 DI, PT100 JIS
Voltage: 0-60mV
5: 0-10V, 0-1V, 0-5V, 1-5V
6: 0-20/4-20mA
9: Special Order

(3) Output 1

0: None
1: Relay rated 2A/240VAC (SPST)
2: SSR Driver 5 VDC @ 30 Ma
3: 4-20mA / 0-20mA Isolated
4: 1-5V / 0-5V/0 - 10V Isolated
6: Triac output 1A / 240VAC,SSR
C: SSR Driver 14 VDC @ 40 Ma

(4) Output 2

0: None
1: Relay rated 2A/240VAC (SPST)
2: SSR Driver 5 VDC @ 30 Ma
3: 4-20mA / 0-20mA Isolated
4: 1-5V / 0-5V/0-10V Isolated
6: Triac output 1A / 240VAC,SSR
7: Transmitter power supply 20 VDC/25 ma Isolated
8: Transmitter power supply 12 VDC/40 ma Isolated
A: Transmitter power supply 5 VDC/80 ma Isolated
C: SSR Driver 14 VDC @ 40 Ma

(5) Output 3

0: None
1: Relay rated 2A/240VAC (SPST)
2: SSR Driver 5 VDC @ 30 Ma
6: Triac output 1A / 240VAC,SSR
7: Transmitter power supply 20 VDC/25 ma Isolated
8: Transmitter power supply 12VDC/40 ma Isolated
A: Transmitter power supply 5VDC/80 ma Isolated
C: SSR Driver 14 VDC @ 40 Ma

(6) Output 4

0: None
1: Relay rated 2A/240VAC (SPST)
2: SSR Driver 5 VDC @ 30 Ma
3. Retransmission 4-20 / 0-20mA, isolated
4. Retransmission 1-5 / 0-5 / 0-10VDC, isolated
6: Triac output 1A / 240VAC,SSR
7: Transmitter power supply 20 VDC/25 ma Isolated
8: Transmitter power supply 12VDC/40 ma Isolated
A: Transmitter power supply 5VDC/80 ma Isolated
C: SSR Driver 14 VDC @ 40 Ma

(7) Output 5

D: RS-485 Modbus RTU interface Isolated

(8) Display Board and cable

0: None
3: Display Board with 300mm connection cable
4: Display Board with 1000mm connection cable

(9) Modbus Addressing: (if blank factory default is Modbus address #1)
CA: Modbus Comm Address #1
CB: Modbus Comm Address #2
CC: Modbus Comm Address #3
Note that field configuration of Comm address requires the optional display.

NOTES: SCADA (Supervisory Control & Data Acquisition)

FDC-nCompass Series Graphic User Interface (GUI) touch screen provides a full SCADA feature set providing ease of use with either an icon menu system with finger navigation or traditional Menu bar, data acquisition, alarm manager, operator audit trail, multi-level security with user rights, LAN connections and more.

The GUI provides ease of configuration, use & support.

- System Configuration for loop, alarm, input & outputs assignment, Help language selection and more, all without an external device or PC.
- Loop Views: view 1 to 10 controls in single or overall view as well as in a Trend format (trend up to loop controls PV, SP & Percent Out).
- Profile: Virtually unlimited number of profiles with each profile having up to 64 steps with up to 3 events.
- File Management: View, print, copy/move Profile, Alarm, Historical Data (data log files) and operator audit trial files. File transfer via LAN features or USB flash memory.
- LAN: Remote Access & touch screen operation (VNC), email/SMS on alarm, email historical, alarm & audit trail files on-demand, Web Page (view only) and FTP of alarm, audit and historical data files automatically or on-demand.

Data Acquisition:

- Data log PV, SP and PID Percent output for 1 to10 loop controls.
- Log interval: configurable 6 seconds to 31 minutes with configurable number of days to auto start & name next file (1 to 31 days).
- File Start/Stop: Configurable; operator on-demand, on system boot, profile ramp-soak start/end or digital inputs
- File Interval: Once started a data log file is configurable to auto end and start new file with the same name as previous file with an appended time/date name. Configurable time interval is from 1 to 31 days.
- File name: Operator entered file name, batch & lot number or if running a profile, file name same as profile name. (all file names appended with date-time to file name)
- Operator Comments/Events: Unlimited operator comments/events linked to each file entered manually or via Bar Code Scanner.
- Digital Signatures: full support for user based digital signatures for each data file (data encryption).
- Historical Data File: View & print the data directly from the display (auto scale on X & Y axis with each channel selectable for right or left axis values), from a PC after data is copied/moved via LAN (FTP or email) or USB Flash Memory card provided.

NOTES: B42 Control Board (1 to 10 may be used)

B42 Loop Control: http://www.futuredesigncontrols.com/B42.HTM

- Analog Input: 200ms scan rate with 18-bit A/D resolution providing high accuracy and PID control performance
- Outputs: Maximum of 4 control or auxiliary outputs
- PID Control Outputs: up to 2 with Auto Tune capability (heat-cool) Relay, SSR driver 5V, SSR driver 14V, 1A SSR, mA* or VDC*
- Alarm or Event: up to 3 outputs (outputs #2, #3 & #4) Relay, SSR driver 5V, SSR driver 14V or 1A SSR
- Auxiliary Outputs: up to 2 outputs, one or two Transmitter power supplies and/or one Retransmission* output
  *Note: Analog mA and VDC PID and retransmit outputs are 15-bit, contributing to high performance PID control and highly accurate retransmitted PV or SP values.
- Alarm Types: Process, Deviation and Deviation Band, all High or Low
- PID Parameters: Two sets of PID values may be configured and selected by Profile Step and/or by digital event input. (see next)
- Event Inputs: Configurable for one of the following functions.
  - Profile Run (momentary)
  - Profile Hold (closed profile hold, open profile run)
  - Profile Run/Hold (closed profile run, open profile hold)
  - Profile Abort (momentary)
  - Segment Advance (momentary)
  - Manual Mode (closed manual mode, open normal control)
  - Failure Transfer: (closed failure transfer, open normal control) (transfer active %Out is a pre-configured value for each output)

NOTES: Profile Ramp / Soak Specifications

Profile Ramp/Soak:

- Profiles: Virtually an unlimited number of profiles in system memory
- Profile Type: System configurable for separate profiles per loop or common profile with slave set point operation for other loops.
- Profile Start: via touch screen or via Event input (see above).
- Profile Name: Free form 16 character (max) naming convention with auto append time/date stamp.
- Global Profile Configuration
  - Profile Time Base: Time Ramp (hh:mm or mm:ss)
    Ramp Rate (degrees/minute or hour)
  - Starting and Ending SP values (may be individually active or not).
  - Guaranteed Soak & Ramp (GS):
    - Range 1 to 999F (555C), configurable per step as low, high, band or inactive (disable).
- Power Fail / Recovery when Profile is active:
  - Continue from last SP value.
  - Continue from last PV value.
  - Static mode (abort profile) with last static SP entered.
- Profile Segments: (maximum of 64)
  - Jump To Segment: configurable by segment (also nested looping)
  - Events: up to 3 events* per segment.
  - Guaranteed Soak & Ramp: Configurable active or not by segment.

  * Note: The number of events available is predicated on the control board hardware, number of control boards and configuration.

- Event Input: (profile specific; refer to B42 Control Board for all options)
  - Configurable functions below (mutually exclusive to each other)
    - Profile Run (momentary)
    - Profile Hold (closed profile hold, open profile run)
    - Profile Run/Hold (closed profile run, open profile hold)
    - Step Advance (momentary)
    - Profile Abort (momentary)

Although the Event Input provides a manner to control Profile functions remotely, typically Profile actions would have the Operator using the intuitive easy to use 4.3” color touch screen (operator interface).

NOTES: Power Supply – Optional

Power Capacity Required (Watts) - A base system requires ~10 watts of 24Vdc power for the nCompass user interface. The 15W power supplies offered are sufficient to power display only.

Note: The power supply should be used to power nCompass components only and not OEM or other end user components or devices.

P.O. Box 1196 Bridgeview, IL 60455
888.751.5444 Sales 888.307.8014 Fax

Technical Support: 866.342.5332
http://www.futuredesigncontrols.com

RoHS Compliant

FUTURE DESIGN SOFTWARE LICENSE FOR FDC Orion nCompass Control System

Future Design Controls Windows CE based nCompass display and Control Module (Idec PLC) software (listed as “SOFTWARE” in this document) is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE is licensed at no charge to the end user when all components, including but not limited to the Control Module (Idec PLC components) and CE Color Touch Screen display are purchased from Future Design Controls. All ownership and rights remain with Future Design Controls.

I. LICENSE GRANT. This LICENSE grants you the following rights:

A. You may use Future Design’s SOFTWARE with Future Design Controls products or products marketed by Future Design Controls only. Products are considered Future Design Controls products (and products marketed by Future Design Controls) when the sale or shipment originates from our main headquarters in Bridgeview, IL or one of our authorized office locations. Products purchased from Future Design Controls distributors or OEM's with brand labels other than Future Design controls are not considered Future Design Controls products and cannot be used with Future Designs SOFTWARE. Using Future Design's SOFTWARE with any other manufacturer (or distributor) of hardware is a violation of this license and applicable copyright laws. The SOFTWARE is considered in "use" when it is installed into permanent or temporary memory (e.g. CE display unit/PLC, or other storage device)

B. Solely with respect to electronic documents included with the SOFTWARE, you may make a copy (either in hardcopy or electronic form), provided that the software is not resold without the knowledge and acceptance of terms by Future Design Controls.

II. TITLE; COPYRIGHT.

All title and copyrights in and to the SOFTWARE (including but not limited to any images, photographs, animation, video, audio, music, text and "applets" incorporated into the SOFTWARE), the accompanying printed materials, and any copies of the SOFTWARE are owned by Future Design Controls or its suppliers. The SOFTWARE is protected by copyright laws and international treaty provisions. Therefore, you must treat the SOFTWARE like any other copyrighted material, except that you may either (A) make one copy of the SOFTWARE solely for backup or archival purposes or (B) install the SOFTWARE on each system purchased from Future Design Controls (CE display/PLC) provided you keep the original solely for backup or archival purposes. You may not copy the printed materials accompanying the SOFTWARE.

III. ADDITIONAL RIGHTS AND LIMITATIONS.

A. Reverse Engineering, De-compilation, and Disassembly: You may not reverse engineer, decompile, disassemble or modify the SOFTWARE (CE display software and/or Control Module (PLC) code).

B. No Separation of Components. The SOFTWARE is licensed as a single product and the software programs comprising the SOFTWARE may not be separated for use on any hardware not supplied by Future Design Controls.

C. Rental. You may not rent or lease the SOFTWARE.

D. Software Transfer. You may NOT transfer any of your rights under this LICENSE.

E. Termination. Without prejudice to any other rights, Future Design Controls may terminate this LICENSE if you fail to comply with the terms and conditions of this LICENSE. In such event, you must destroy all copies of the SOFTWARE and agree not to modify/distribute the software or download to any hardware not purchased from Future Design Controls.

F. The selection, application and use of Future Design products and/or software is the sole responsibility of the purchaser or end user. No claims will be allowed for any damages or losses, whether direct, indirect, incidental, special or consequential. In addition, Future Design reserves the right to make changes without notification to purchaser or user to materials or processing that do not affect compliance with any applicable specification. Future Design Controls makes no warranties when using Future Design Controls SOFTWARE system.

Software Usage Note:

The selection, application and use of Future Design Control products or software is the sole responsibility of the purchaser or end user. No claims will be allowed for any damages or losses, whether direct, indirect, incidental, special or consequential.

In addition, Future Design reserves the right to make changes without notification to purchaser or user to materials or processing that do not affect compliance with any applicable specification. Future Design makes no warranties when using the nCompass system.

Warranty:

Future Design Controls products described in this book are warranted to be free from functional defects in material and workmanship at the time the products shipped from Future Design Controls facilities and to conform at that time to the specifications set forth in the relevant Future Design Controls manual, sheet or sheets for a period of one year after delivery to the first purchaser.

Future Design Controls FDC-0450 products are warranted to be free from functional defects in materials and workmanship at the time the products shipped from Future Design Controls facilities and to conform at that time to the specifications set forth in the relevant Future Design Controls manual, sheet or sheets for a period of one year after delivery to the first purchaser for use.

There are no expressed or implied Warranties extending beyond the Warranties herein and above set forth. Limitations: Future Design Controls provides no warranty or representations of any sort regarding the fitness of use or application of its products by the purchaser. Users are responsible for the selection, suitability of the products for their application or use of Future Design Controls products.

Future Design Controls shall not be liable for any damages or losses, whether direct, indirect, incidental, special, consequential or any other damages, costs or expenses excepting only the cost or expense of repair or replacement of Future Design Control products as described below.

Future Design Controls sole responsibility under the warranty, at Future Design Controls option, is limited to replacement or repair, free of charge, or refund of purchase price within the warranty period specified. This warranty does not apply to damage resulting from transportation, alteration, misuse or abuse.

Future Design Controls reserves the right to make changes without notification to purchaser to materials or processing that do not affect compliance with any applicable specifications.

Return Material Authorization:

Contact Future Design Controls for Return Material Authorization Number prior to returning any product to our facility:

FUTUREDESIGN CONTROLS

7524 West 98th Place – Bridgeview, IL 60455 – Phone 888.751.5444 – Fax 888.307.8014

http://www.futuredesigncontrols.com