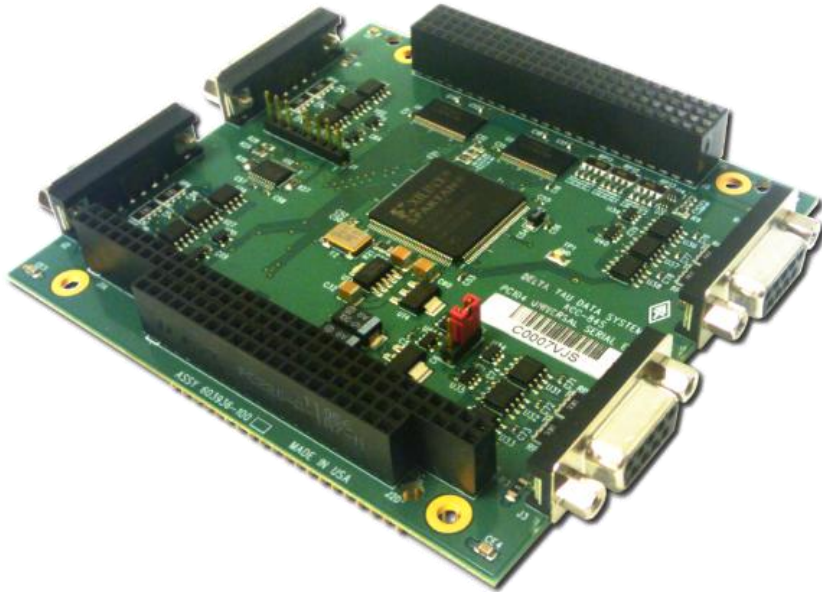


# USER MANUAL

## Accessory 84S



Universal Serial Encoder Interface for Clipper

3AX-603936-XUXX

February 24, 2014



**DELTA TAU**  
Data Systems, Inc.

*NEW IDEAS IN MOTION ...*

## Copyright Information

© 2014 Delta Tau Data Systems, Inc. All rights reserved.

This document is furnished for the customers of Delta Tau Data Systems, Inc. Other uses are unauthorized without written permission of Delta Tau Data Systems, Inc. Information contained in this manual may be updated from time-to-time due to product improvements, etc., and may not conform in every respect to former issues.

To report errors or inconsistencies, call or email:

**Delta Tau Data Systems, Inc. Technical Support**

Phone: (818) 717-5656

Fax: (818) 998-7807

Email: [support@deltatau.com](mailto:support@deltatau.com)

Website: <http://www.deltatau.com>

## Operating Conditions

All Delta Tau Data Systems, Inc. motion controller products, accessories, and amplifiers contain static sensitive components that can be damaged by incorrect handling. When installing or handling Delta Tau Data Systems, Inc. products, avoid contact with highly insulated materials. Only qualified personnel should be allowed to handle this equipment.

In the case of industrial applications, we expect our products to be protected from hazardous or conductive materials and/or environments that could cause harm to the controller by damaging components or causing electrical shorts. When our products are used in an industrial environment, install them into an industrial electrical cabinet or industrial PC to protect them from excessive or corrosive moisture, abnormal ambient temperatures, and conductive materials. If Delta Tau Data Systems, Inc. products are directly exposed to hazardous or conductive materials and/or environments, we cannot guarantee their operation.



**WARNING**

A Warning identifies hazards that could result in personal injury or death. It precedes the discussion of interest.



**Caution**

A Caution identifies hazards that could result in equipment damage. It precedes the discussion of interest.



**Note**

A Note identifies information critical to the user's understanding or use of the equipment. It follows the discussion of interest.

---

<b>REVISION HISTORY</b>				
<b>REV.</b>	<b>DESCRIPTION</b>	<b>DATE</b>	<b>CHG</b>	<b>APPVD</b>
1	MANUAL CREATION	09/08/11	DCDP	RN
2	SSI DATA RANGE, MISC FORMAT	02/24/14	RN	RN

*This page left blank intentionally*

## Table of Contents

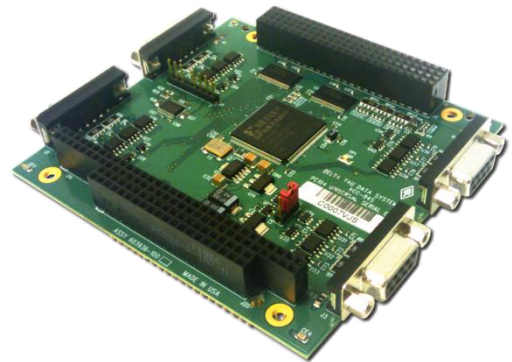
<b>INTRODUCTION.....</b>	<b>7</b>
<b>SPECIFICATIONS.....</b>	<b>8</b>
Part Number.....	8
Environmental Specifications.....	9
Physical Specifications.....	9
<b>ADDRESSING ACC-84S.....</b>	<b>10</b>
<b>USING ACC-84S WITH TURBO CLIPPER .....</b>	<b>11</b>
Using Absolute Serial Encoders with Turbo PMAC .....	11
Serial Synchronous Interface (SSI) Feedback Configuration.....	15
<i>Description of SSI Registers .....</i>	<i>16</i>
<i>Configuring SSI Encoders .....</i>	<i>20</i>
<i>Configuring SSI Encoders Example Using Technique 1 .....</i>	<i>21</i>
<i>Configuring SSI Encoders Example Using Technique 2.....</i>	<i>24</i>
EnDat 2.2 Feedback Configuration.....	31
<i>Description of EnDat Registers .....</i>	<i>32</i>
<i>Configuring EnDat Encoders.....</i>	<i>36</i>
<i>Configuring EnDat Encoders Example Using Technique 1 .....</i>	<i>37</i>
<i>Configuring EnDat Encoders Example Using Technique 2 .....</i>	<i>41</i>
BiSS-C/BiSS-B Feedback Configuration.....	48
<i>Description of BiSS Registers .....</i>	<i>49</i>
<i>Configuring BiSS Encoders .....</i>	<i>53</i>
<i>Configuring BiSS Encoders Example Using Technique 1 .....</i>	<i>54</i>
<i>Configuring BiSS Encoders Example Using Technique 2.....</i>	<i>57</i>
Yaskawa Feedback Configuration .....	63
<i>Description of Yaskawa Registers.....</i>	<i>64</i>
<i>Configuring Yaskawa Encoders.....</i>	<i>68</i>
<i>Yaskawa Sigma II 16-Bit Absolute Encoder .....</i>	<i>68</i>
<i>Yaskawa Sigma II 17-Bit Absolute Encoder .....</i>	<i>71</i>
<i>Yaskawa Sigma III 20-Bit Absolute Encoder.....</i>	<i>74</i>
<i>Yaskawa Sigma II 13-Bit Incremental Encoder.....</i>	<i>77</i>
<i>Yaskawa Sigma II 17-Bit Incremental Encoder.....</i>	<i>79</i>
<i>Yaskawa Incremental Encoder Alarm Codes .....</i>	<i>81</i>
<i>Homing with Yaskawa Incremental Encoders .....</i>	<i>82</i>
Absolute Power-On Phasing (SSI, EnDat, BiSS).....	83
Absolute Power-On Phasing (Yaskawa Absolute Encoders).....	87
<b>LAYOUT AND PINOUTS.....</b>	<b>89</b>
Board Layout .....	89

<i>ACC-84S Standalone</i> .....	89
<i>ACC-84S Mounted on the Turbo Clipper Drive</i> .....	90
Encoder Feedback Connector Pinouts .....	91
<i>J1: Encoder Feedback Channel 1</i> .....	91
<i>J2: Encoder Feedback Channel 2</i> .....	91
<i>J3: Encoder Feedback Channel 3</i> .....	92
<i>J4: Encoder Feedback Channel 4</i> .....	92
Encoder Specific Connection Information .....	93
<i>Yaskawa Sigma II Encoders</i> .....	93
<b>APPENDIX A: E-POINT JUMPERS</b> .....	<b>95</b>
<b>APPENDIX B: SCHEMATICS</b> .....	<b>96</b>
Encoder Feedback Connectors .....	96
ACC-84S Serial Encoder Feedback Input Circuitry .....	98
Stacking Connectors PinOuts .....	100

## INTRODUCTION

---

The Accessory 84S (ACC-84S) Universal Serial Encoder Interface Board provides up to four channels of serial encoders to be read by the Turbo Clipper and Turbo PC/104 controllers. ACC-84E supports different serial encoder protocols depending on the option ordered. These protocols are programmed into an on-board FPGA in the factory. Currently, ACC-84S currently supports the following protocols:



- SSI Synchronous Serial Interface
- EnDat 2.2 EnDat 2.2 interface from HEIDENHAIN
- BiSS B Renishaw Biss B Unidirectional
- BiSS C Renishaw Biss C Unidirectional
- Yaskawa Yaskawa Sigma II and Sigma III feedback support

In the future, the ACC-84S will support these protocols:

- Panasonic Panasonic Feedback Style
- Tamagawa Tamagawa Feedback Style
- HiperFace HiperFace (Serial and Sin/Cos Combined) Feedback Style (Requires Using ACC-51S in Tandem with ACC-84S)

Each ACC-84S can only support one of the protocols mentioned above for all four channels. If the customer has a number of different serial protocols in the system, the customer should use the same number of separate ACC-84S cards.

Since ACC-84S is strictly a feedback input card, if the user intends to use the card for feedback for closed loop servo control, the servo command should be sent out to the amplifier using a PC/104 bus axis interface card depending on the signal and control type required by amplifier. Turbo Clipper and Turbo PC/104 by default come with filtered PWM output. If the user requires true DAC or direct PWM, he or she can use the following axis interface cards:

- ACC-8ES 4-channel dual 18-bit true DAC output board
- ACC-8FS 4-channel direct PWM output board

Up to two ACC-84S boards can be connected to one Turbo Clipper, providing up to 8 channels of serial encoder feedback.

The ACC-84S board will take the data from the serial encoder and process it as up to four 24-bit binary parallel words, one 24-bit word per channel, depending on protocol specifications. This data can then be processed in the encoder conversion table for position and velocity feedback. With proper setup, the information can also be used to commutate brushless and AC induction motors.

## SPECIFICATIONS

---

### Part Number

---

There are currently four feedback options available with the following part numbers:

<b>Part Number</b>	<b>Feedback Type</b>	<b>Description</b>
3-3936-0-0002-000000	SSI	Synchronous Serial Interface
3-3936-0-0003-000000	EnDat 2.2	EnDat 2.2 interface from HEIDENHAIN
3-3936-0-000B-000000	Biss B/C	Renishaw Biss B/C Unidirectional
3-3936-0-0006-000000	Yaskawa	Yaskawa Sigma II and Sigma III Feedback



## Environmental Specifications

---

Description	Specification	Notes
Operating Temperature	0 °C to 45 °C	
Storage Temperature	-25 °C to 70 °C	
Humidity	10% to 95%	Non-Condensing

## Physical Specifications

---

Description	Specification	Notes
DB Option Connectors	DB9 Female	Connector: D-Sub DE-9F Mating: D-Sub DE-9M

## **ADDRESSING ACC-84S**

---

The base address for each ACC-84S is determined by jumper E1 as follows:

<b>Jumper E1 Setting</b>	<b>Base Address</b>
Pins 1–2	\$78800
Pins 2–3	\$78820

- If the user wants to use two ACC-84S cards on one Turbo Clipper, the first card must be set at base address \$78800, and the second card must be set at base address \$78820.
- If the user wants to use ACC-84S with ACC-51S on one Turbo Clipper, the ACC-51S must be set at \$78800, and the ACC-84S must be set at \$78820.

## USING ACC-84S WITH TURBO CLIPPER

### Using Absolute Serial Encoders with Turbo PMAC

With Turbo PMAC, the user must bring the absolute serial encoder data in as a parallel Y-word (unfiltered) into the Encoder Conversion Table where it is processed for the PMAC to use for ongoing position in the motor servo-loop, power-on absolute position, and (power-on and ongoing) phase referencing.

Encoder data is normally left shifted 5 bits in the Encoder Conversion Table to provide fractional data. This process can cause saturation of certain registers in Turbo PMAC (not an issue with Power PMAC) with higher resolution absolute serial encoders, thus for this type of encoders it is recommended to process the data as unshifted.

#### Technique 1: Standard 5-bit Shift

**Recommended for lower resolution serial encoders (less than 23 bits of single-turn/absolute data stream)**, this technique places the Least Significant Bit (LSB) of the serial data in Bit 5 of the result register (normal 5-bit shift), providing the standard 5 bits of (non-existent) fraction.

#### Technique 2: No Shift

**Recommended for higher resolution serial encoders (23 bits of single-turn/absolute data stream or greater)**, this technique places the Least Significant Bit (LSB) of the serial data in Bit 0 of the result register (unshifted), creating no fractional bits.

The following table shows a quick comparison between both techniques, and summarizes limitations of parameters of interest:

Parameter/Description		Technique 1 (5-bit shift)	Technique 2 (no shift)	Units
Processed resolution (Scale Factor SF)	Rotary	$2^{ST}$	$2^{ST-5}$	counts/revolution
	Linear	1/Pitch	1/32*Pitch	counts/user unit
Max. Open-Loop Velocity		$2^{18} * \text{ServoClk}$		counts/msec
Max. Closed-Loop Velocity		$2^{23} * 3 / \text{Ixx08} * 32$		counts/msec
Max. travel before rollover	Rotary	$2^{47} / 2^{ST} = 2^{47-(ST)}$	$2^{47} / 2^{ST-5} = 2^{47-(ST-5)}$	revolutions
	Linear	$2^{47} / \text{Scale Factor}$		user units
PID Position Tuning		Can be tricky	Easier	--
Power-On Position Read		Automatic (Ixx10, Ixx95)	Custom PLC	--
Power-On Phase Reference		Custom PLC	Automatic (Ixx75, Ixx81, Ixx91)	--

Where ST: Rotary (angle) encoder Single Turn resolution in bits  
 ServoClk: Servo update rate in KHz  
 Ixx08: Motor xx Position Scale Factor  
 Pitch: Linear encoder resolution in user units (i.e. mm, inches)

### Processed Resolution, Scale Factor (SF)

Turbo PMAC expects the motor count Least Significant Bit LSB to be left-shifted (5 bits). Hence, the only difference, when unshifted, is that the motor position loop will now consider 1 LSB of the source to be 1/32 of a motor count instead of 1 motor count.

For example, take a 37-bit absolute serial angle encoder (25-bit single turn, 12-bit multi-turn) and its equivalent linear scale (e.g. 10 nm pitch):

Technique 1 (5-bit shift)	<b>Rotary</b>	$2^{ST}$	$2^{25} = 33,554,432$	<b>counts/revolution</b>
	<b>Linear</b>	1/Pitch	1/0.00001 = 100,000	<b>counts/mm</b>
Technique 2 (no shift)	<b>Rotary</b>	$2^{ST-5}$	$2^{20} = 1,048,576$	<b>counts/revolution</b>
	<b>Linear</b>	32/Pitch	1/32*0.00001 = 3,125	<b>counts/mm</b>



**Note**

The servo algorithm utilizes all data bits in the stream (25 bits in this example) in either technique. The performance is not compromised.

### Maximum “Actual” Open-Loop Velocity

In open-loop mode, the actual velocity register is limited by the Encoder Conversion Table to 24 bits. Also, it requires two samples (servo cycles) to compute the velocity. Therefore, the maximum value that the actual velocity register can hold is  $2^{(24-5bit\ shift)} / (2 \cdot f_s)$ , which equals

$$2^{18} / f_s,$$

where  $f_s$  is the servo clock frequency [kHz].

Take a 37-bit absolute serial angle encoder (25-bit single turn, 12-bit multi-turn) and its equivalent linear scale (e.g. 10 nm pitch), and compare two different clock settings:

With the default servo clock of 2.258 kHz, the maximum actual open-loop velocity is  $MaxActVel = 2^{18} \cdot 2.258 = 591,921$  [counts/msec], yielding:

	<b>Rotary [rpm]</b> =MaxActVel*60000/SF	<b>Linear [mm/sec]</b> =MaxActVel*1000/SF
Technique 1 (5-bit shift)	1,058	5,919
Technique 2 (no shift)	33,870	189,414

With a servo clock setting of 4.500 kHz, the maximum actual open-loop velocity is  $MaxActVel = 2^{18} \cdot 4.500 = 1,179,648$  [counts/msec], yielding:

	<b>Rotary [rpm]</b> =MaxActVel*60000/SF	<b>Linear [mm/sec]</b> =MaxActVel*1000/SF
Technique 1 (5-bit shift)	2,109	11,796
Technique 2 (no shift)	67,500	377,487



*Note*

The maximum actual velocity attainable is directly proportional to the servo clock frequency. The faster the servo update, the higher is the actual velocity threshold.

### Maximum “Commanded” Closed-Loop Velocity

In closed-loop mode, the commanded (desired) velocity register is limited to  $2^{(24-1_{sign\ bit})} \cdot 3 / (I_{xx08} \cdot 32)$  [counts/millisecond]. In terms of motor counts per millisecond, the maximum commanded velocity will be the same with or without shifting, but since the number of counts per revolution “unshifted” is 32 times less, then the maximum programmable velocity is 32 times greater.

Take a 37-bit absolute serial angle encoder (25-bit single turn, 12-bit multi-turn) and its equivalent linear scale (e.g. 10 nm pitch). Assume a position scale factor  $I_{xx08}$  of 1 (typical with higher resolution  $\geq 23$ -bit encoders, leave at default of 96 otherwise):

The maximum “commanded” closed-loop velocity ( $I_{xx16}$ ,  $I_{xx22}$ ) programmable in Turbo PMAC is  $MaxCmdVel = 2^{23} \cdot 3 / (I_{xx08} \cdot 32) = 786,432$  [counts/msec], yielding:

	<b>Rotary [rpm]</b> = $MaxCmdVel * 60000 / SF$	<b>Linear [mm/sec]</b> = $MaxCmdVel * 1000 / SF$
Technique 1 (5-bit Shift)	1,406	7,864
Technique 2 (no Shift)	45,000	251,658

### Maximum Motor Travel

In Jog mode the rollover is handled gracefully by PMAC and jogging can be virtually performed forever. However, this can be problematic when running a motion program indefinitely in incremental mode where the motor position register (48-bit fixed) can roll over much sooner than the axis position register, which is a 48-bit floating point number.



*Note*

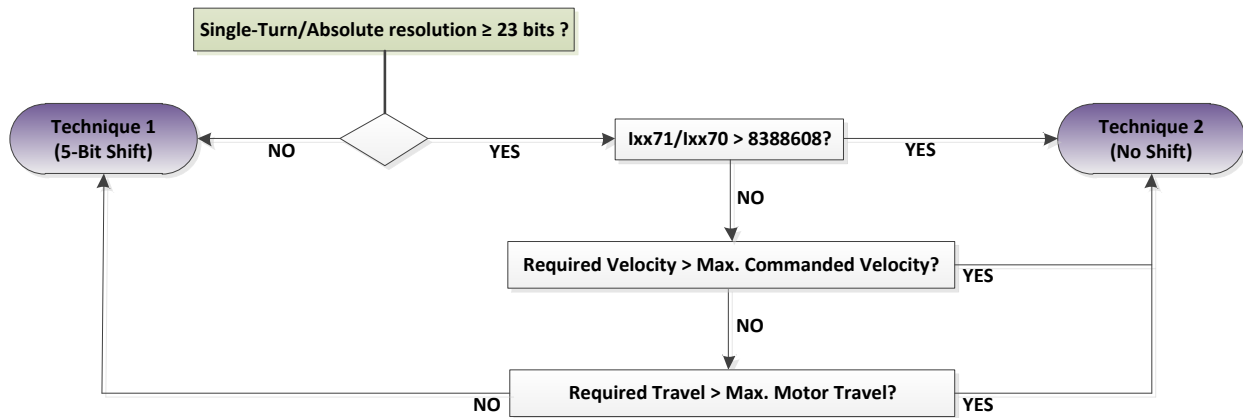
Absolute Serial Encoders with limited multi-turn range normally do roll over far before the motor position register in Turbo PMAC does (e.g. 12-bit multi-turn is 2048 revolutions in each direction).

Take a 37-bit absolute serial angle encoder (25-bit single turn, 12-bit multi-turn) and its equivalent linear scale (e.g. 10 nm pitch). Assume a position scale factor  $I_{xx08}$  of 1 (typical with higher resolution  $\geq 23$ -bit encoders, leave at default of 96 otherwise):

		Total Travel Span	In each direction = $Span/2$	Units
Technique 1 (5-bit shift)	<b>Rotary</b>	$2^{47-25} = 4,194,304$	2,097,152	<b>revolutions</b>
	<b>Linear</b>	$2^{47}/SF$	1,407,374,883	<b>mm</b>
Technique 2 (no shifting)	<b>Rotary</b>	$2^{47-20} = 134,217,728$	67,108,864	<b>revolutions</b>
	<b>Linear</b>	$2^{47}/SF$	45,035,996,274	<b>mm</b>

### Making a Decision: Technique 1 (5-bit shift) versus Technique 2 (no-shift)

The following chart shows the recommended method for using a given absolute serial encoder attached to a specific motor (number of poles):



If the application requirements are unknown or unrestrictive, it is always acceptable to set up the absolute serial encoder using:

- Technique 1 for single-turn/absolute resolutions < 23 bits
- Technique 2 for single-turn/absolute resolutions ≥ 23 bits



Some of the following examples set Ixx83 to an Encoder Conversion Table output memory location. This means the commutation happens at the servo rate, rather than the phase rate. If, when using this, the user cannot control the motor, he or she can try setting the servo clock equal to the phase clock and then use Ixx60 (Motor xx Servo Cycle Period Extension Period) to return the servo period to the desired value.

## **Serial Synchronous Interface (SSI) Feedback Configuration**

---

The SSI option allows the Turbo Clipper to connect to up to eight channels of Serial Synchronous Interface (SSI) type devices: four channels per ACC-84S, for a total of eight SSI devices when using two ACC-84S cards. Setting up SSI feedback requires the programming of two essential control registers:

- Global Control Registers
- Channel Control Registers

The resulting data is found in:

- SSI Data Registers

## Description of SSI Registers

### Global Control Registers

The Global Control Register controls the clock settings and trigger settings of the feedback protocol and it is located at X-word of the (base address) + \$F. Its default value is \$630002. The following table shows address locations for the Global Control Register for different address settings:

Card Number	Base Address	Global Control Register
1 <sup>st</sup> ACC-84S	\$78800	X:\$7880F
2 <sup>nd</sup> ACC-84S	\$78820	X:\$7882F

The Global Control register is used to program the serial encoder interface clock frequency *SER\_Clock* and configure the serial encoder interface trigger clock. *SER\_Clock* is generated from a two-stage divider clocked at 100 MHz:

$$SER\_Clock = \frac{100}{(M + 1) \times 2^N} MHz$$

M	N	Clock Frequency
49	0	2.0 MHz
99	0	1.0 MHz
99	1	500.0 KHz
99	2	250.0 KHz
...	...	

Default Settings: M=99, N=0 → 1 MHz transfer rates

There are two external trigger sources; phase and servo. Bits [9:8] in the Global Control register are used to select the source and active edge to use as the internal serial encoder trigger. The internal trigger is used by all four channels to initiate communication with the encoder. To compensate for external system delays, this trigger has a programmable 4-bit delay setting in 20 μsec increments.

23--16	15--12	11	10	9	8	7	6	5	4	3	2	1	0
M_Divisor	N_Divisor			Trigger Clock	Trigger Edge	Trigger Delay				Protocol Code			

Bit	Type	Default	Name	Description
[23:16]	R/W	0x00	M_Divisor	Intermediate clock frequency for <i>SER_Clock</i> . The intermediate clock is generated from a (M+1) divider clocked at 100 MHz.
[15:12]	R/W	0x0	N_Divisor	Final clock frequency for <i>SER_Clock</i> . The final clock is generated from a $2^N$ divider clocked by the intermediate clock.
[11:10]	R	00	Reserved	Reserved and always reads zero.
[09]	R/W	0	TriggerClock	Trigger clock select: 0= PhaseClock 1= ServoClock
[08]	R/W	0	TriggerEdge	Active clock edge select: 0= rising edge 1= falling edge
[07:04]	R/W	0x0	TriggerDelay	Trigger delay program relative to the active edge of the trigger clock. Units are in increments of 20 usec.
[03:00]	R	0x2	ProtocolCode	This read-only bit field is used to read the serial encoder interface protocol supported by the FPGA. A value of 0x2 defines this as SSI protocol.



## Channel Control Registers

Each channel also has its Channel Control Register which controls the functionality of each channel. The user uses this register to:

- Configure the number of position bits in the serial bit stream
- Enable or disable channels through the SENC\_MODE bit
- Enable or disable communication with the encoder using the trigger control bit

The Channel Specific Control Register appears at the following memory locations:

Card Number	Channel Number	Address
1 <sup>st</sup> ACC-84S	1	X:\$78800
1 <sup>st</sup> ACC-84S	2	X:\$78804
1 <sup>st</sup> ACC-84S	3	X:\$78808
1 <sup>st</sup> ACC-84S	4	X:\$7880C
2 <sup>nd</sup> ACC-84S	1	X:\$78820
2 <sup>nd</sup> ACC-84S	2	X:\$78824
2 <sup>nd</sup> ACC-84S	3	X:\$78828
2 <sup>nd</sup> ACC-84S	4	X:\$7882C

The diagram below describes the functionality of each bit range in the Channel Specific Control Registers:

[23:16]	15	14	13	12	11	10	[9:6]	[5:0]
	Parity Type		Trigger Mode	Trigger Enable	GtoB	RxData Ready /SENC_MODE	0	PositionBits/ Resolution

The following table gives descriptions and settings of each bit range:

Bit	Type	Default	Name	Description
[23:16]	R	0x00	Reserved	Reserved and always reads zero.
[15:14]	R/W	0x00	Parity Type	Parity Type of the received data: 00=None 10=Even 01=Odd 11=Reserved
[13]	R/W	0	Trigger Mode	Trigger Mode to initiate communication: 0= continuous trigger 1= one-shot trigger All triggers occur at the defined Phase/Servo clock edge and delay setting.
[12]	R/W	0	Trigger Enable	0= disabled 1= enabled This bit must be set for either trigger mode. If the Trigger Mode bit is set for one-shot mode, the hardware will automatically clear this bit after the trigger occurs.
[11]	R/W	0	Convert G to B	Gray code to Binary conversion: 0=Binary 1=Gray
[10]	R	0	RxData Ready	This read-only bit provides the received data status. It is low while the interface logic is communicating (busy) with the serial encoder. It is high when all the data has been received and processed.
	W	0	SENC_MODE	This write-only bit is used to enable the output drivers for the SENC_SDO, SENC_CLK, SENC_ENA pins for each respective channel. Writing a 0 to this bit disables the channel Writing a 1 to this bit enables the channel
[09:06]	R	0x0	Reserved	Reserved and always reads zero.
[05:00]	W	0x00	Position Bits	This bit field is used to define the number of position data bits or encoder resolution: Range is 12 – 32 (001100 –100000)



**Note**

RxData Ready and SENC\_Mode share bit 10 of the Channel Specific Control Register. Write to bit 10 to configure SENC\_MODE and then read from bit 10 thereafter for the state of RxData Ready.

## SSI Data Registers

The SSI data is stored in 4 memory locations: Serial Encoder Data Registers A, B, C, and D.

The Serial Encoder Data Register A holds the 24 bits of the encoder position data. If the data exceeds the 24 available bits in this register, the upper overflow bits are LSB justified and readable in the Serial Encoder Data Register B, which also holds the parity error flag.

Serial Encoder Data Registers C and D are reserved and always read zero.

Data D	Data C	Serial Encoder Data B		Serial Encoder Data A
N/A	N/A	23	[22:16]	[15:0]
N/A	N/A	Parity Err		Position Data [40:24]
				Position Data [23:0]

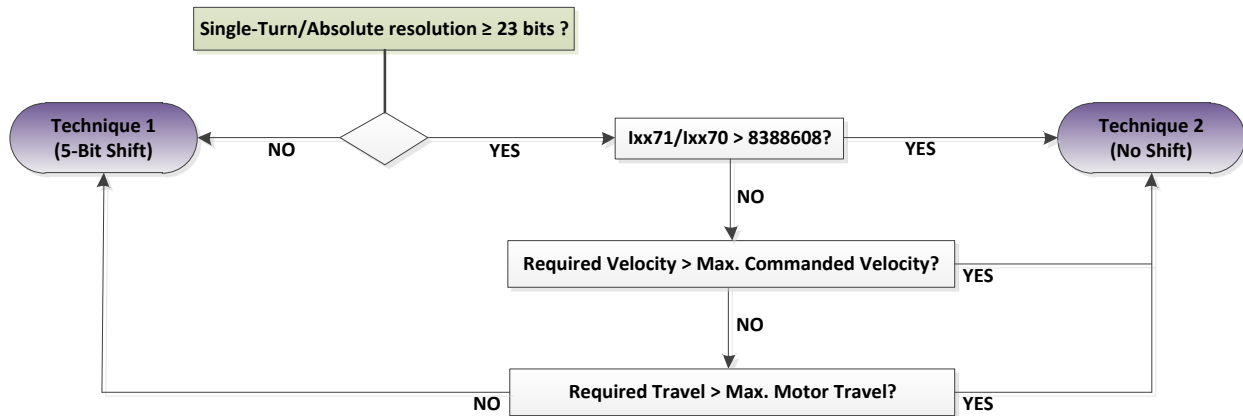
Card Number	Channel Number	SSI Data Register A	SSI Data Register B
1 <sup>st</sup> ACC-84S	1	Y:\$78800	Y:\$78801
1 <sup>st</sup> ACC-84S	2	Y:\$78804	Y:\$78805
1 <sup>st</sup> ACC-84S	3	Y:\$78808	Y:\$78809
1 <sup>st</sup> ACC-84S	4	Y:\$7880C	Y:\$7880D
2 <sup>nd</sup> ACC-84S	1	Y:\$78820	Y:\$78821
2 <sup>nd</sup> ACC-84S	2	Y:\$78824	Y:\$78825
2 <sup>nd</sup> ACC-84S	3	Y:\$78828	Y:\$78829
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C	Y:\$7882D

Data Registers C and D are listed here for future use and documentation purposes only. They do not pertain to the SSI setup and always read zero.

Card Number	Channel Number	SSI Data Register C	SSI Data Register D
1 <sup>st</sup> ACC-84S	1	Y:\$78802	Y:\$78803
1 <sup>st</sup> ACC-84S	2	Y:\$78806	Y:\$78807
1 <sup>st</sup> ACC-84S	3	Y:\$7880A	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880E	Y:\$7880F
2 <sup>nd</sup> ACC-84S	1	Y:\$78822	Y:\$78823
2 <sup>nd</sup> ACC-84S	2	Y:\$78826	Y:\$78827
2 <sup>nd</sup> ACC-84S	3	Y:\$7882A	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882E	Y:\$7882F

## Configuring SSI Encoders

The [Using Absolute Serial Encoders with Turbo PMAC](#) section suggests two techniques for processing absolute serial data. The following chart shows the recommended method for using a given absolute serial encoder attached to a specific motor (number of poles):



If the application requirements are unknown or unrestrictive, it is always acceptable to set up the absolute serial encoder using:

- Technique 1 for single-turn/absolute resolutions < 23 bits
- Technique 2 for single-turn/absolute resolutions ≥ 23 bits

With either technique, the following steps assure the proper setting of a motor with serial absolute encoders:

Step #	Description	Technique 1 (5-bit shift)	Technique 2 (no shift)
1	Global Control Register (per Set of 4 Axes)	√	√
2	Channel Control Register (per Channel)	√	√
3	Encoder Conversion Table (for Position)	Parallel Y-Word unfiltered normal 5-bit shift	Parallel Y-Word unfiltered no shifting
4	Encoder Conversion Table (for Commutation)	Directly from raw data	Parallel Y-Word unfiltered unshifted, limited to 23 bits
5	Absolute Power-On Position Read	Automatic using Ixx10, and Ixx95	Custom PLC (see example)
6	Absolute Power-On Phasing	Custom PLC (see example)	Automatic using Ixx75, Ixx81, and Ixx91

## Configuring SSI Encoders Example Using Technique 1

A 25-bit (13-bit Single-Turn, 12 Multi-Turn) SSI encoder is driving channel 1.

### Step 1: Global Control Register Setup (Technique 1)

[23:16]	[15:12]	11	10	9	8	7	6	5	4	3	2	1	0
M_Divisor	N_Divisor			Trigger Clock	Trigger Edge	Trigger Delay			Protocol Code				

Bit Assignment	Description	Resulting Global Control Register
M=99, N=0	Serial Clock 1.0 MHz	\$630002
Trigger Clock=0	Trigger on Phase Clock (recommended)	
Trigger Edge=0	Rising Edge (recommended)	
Trigger Delay=0	No delay	
Protocol Code=2	SSI	

### Step 2: Channel Control Register Setup (Technique 1)

[23:16]	15	14	13	12	11	10	[9:6]	[5:0]
	Parity Type	Trigger Mode	Trigger Enable	GtoB	RxData Ready /SENC_MODE			PositionBits/ Resolution

Bit Assignment	Description	Resulting Channel Control Register
Parity Type=0	None	\$001419
Trigger Mode=0	Continuous Trigger (common)	
Trigger Enable=1	Must be set to 1 (common)	
SENC_MODE=1	Enable Serial Driver (common)	
Resolution (25 bits)=11001(\$19)	Position Bits	

### Control Registers Power-On PLC (Technique 1)

The global and channel control words have to be executed once on power-up:

```
//===== NOTES ABOUT THIS PLC EXAMPLE =====//
// This PLC example utilizes: - M5990 through M5991
//                               - Coordinate system 1 Timer 1 (I5111)
// Make sure that current and/or future configurations do not create conflicts with
// these parameters.
//=====//

M5990..5991->* ; Self-referenced M-Variables
M5990..5991=0 ; Reset at download
//===== GLOBAL CONTROL REGISTERS =====//
#define SSIGlobalCtrl1_4 M5990 ; Channels 1-4 SSI global control register
SSIGlobalCtrl1_4->X:$7880F,0,24,U ; Channels 1-4 SSI global control register address
//===== CHANNEL CONTROL REGISTERS =====//
#define Ch1SSICtrl M5991 ; Channel 1 SSI control register
Ch1SSICtrl->X:$78800,0,24,U ; Channel 1 SSI control register Address

//===== POWER-ON PLC EXAMPLE, GLOBAL & CHANNEL CONTROL REGISTERS =====//
Open PLC 1 Clear
SSIGlobalCtrl1_4=$630002 ; Trigger at Phase, 1 MHz serial Clock (M=99, N=0)-User Input
Ch1SSICtrl=$001419 ; Channel 1 SSI control register -User Input

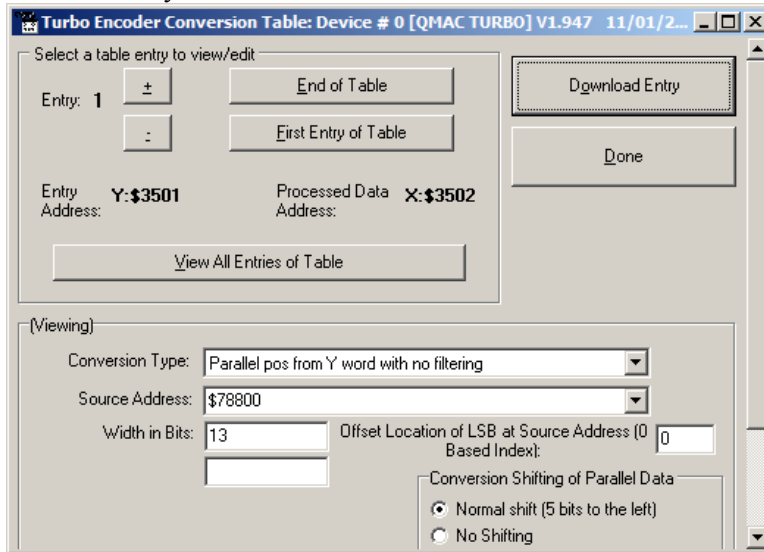
I5111=500*8388608/I10 While(I5111>0) EndWhile ; ½ sec delay
Disable PLC 1 ; Execute once on power-up or reset
Close
//=====//
```

**Step 3: Encoder Conversion Table Setup — for Position (Technique 1)**

- Conversion Type: “Parallel pos from Y word with no filtering”
- Width in Bits is the single-turn/absolute resolution in bits (13 bits in this example)
- Offset Location of LSB: leave at zero
- Normal Shift (5 bits to the left)
- Source Address

Card Number	Channel Number	Source Address (Data Register A)
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

- Click on “Download Entry”:



Record the location of the processed data (X:\$3502 in this example). This is the position and velocity pointers’ address location. This ECT configuration is a 2-line entry, and its equivalent I8000s variables would look like:

```
I8000=$278800 ; Unfiltered parallel pos of location Y:$78800
I8001=$00D000 ; Width (13 bits) and Offset (none), 5 bit left shift
; Processed result is at $3502
```

The position and velocity pointer’s address is the location of the processed data:

```
I100=1 ; Mtr #1 active - remember to activate the channel to see feedback
I103=$3502 ; Mtr #1 position loop feedback address
I104=$3502 ; Mtr #1 velocity loop feedback address
I183=$3502 ; Mtr #1 commutation feedback address
```



*Note*

At this point in the setup process, the user should be able to move the motor/encoder shaft by hand and see “motor” counts in the position window.

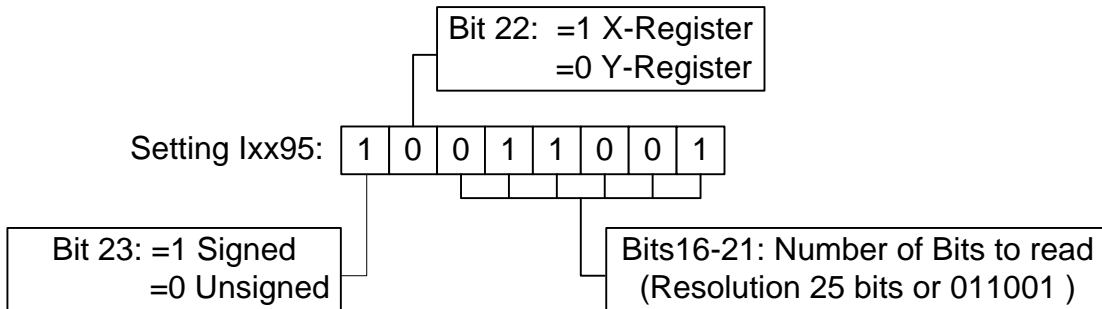
### Step 4: Absolute Power-On Position Read (Technique 1)

With the normal 5-bit shift, the absolute power-on read can be performed using the automatic feature in Turbo PMAC (Ixx80, Ixx10 and Ixx95):

**Example 1:** Channel 1 driving a 25-bit SSI (13-bit single turn, 12-bit multi-turn) encoder:

```
I180=2           ; Absolute power-on read enabled
I110=$78800     ; Absolute Servo power-on position address
I195=$990000    ; Parallel Read, 25 bits, Signed, from Y-Register -User Input
```

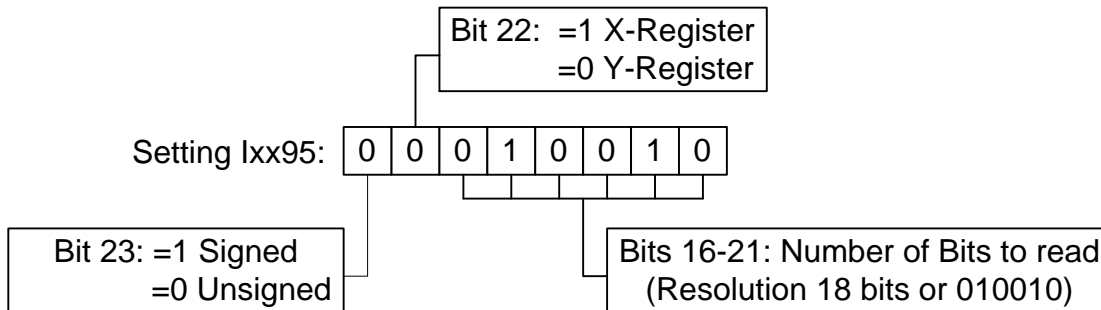
In this mode, PMAC reads and reports 25 bits: it takes the low 24 bits from Y:\$078800, and uses the lowest bit of Y:\$078801 as the 25<sup>th</sup> bit. With this setting of Ixx80, the actual position is reported automatically on power up. Otherwise, a #1\$\* command is necessary to read the absolute position.



**Example 2:** Channel 1 driving an 18-bit absolute SSI (18-bit single turn, 0 multi turn) encoder:

```
I180=2           ; Absolute power-on read enabled
I110=$78800     ; Absolute Servo power-on position address
I195=$120000    ; Parallel Read, 18 bits, Unsigned, from Y-Register -User Input
```

In this mode, PMAC reads and reports 18 bits, the low 18 bits from Y:\$078800. With this setting of Ixx80, the actual position is reported automatically on power-up. Otherwise, a #1\$\* command is necessary to read the absolute position.



*Note*

With purely absolute serial encoders (no multi-turn data), the user must configure the power-on position format for unsigned operation.

## Configuring SSI Encoders Example Using Technique 2

A 37-bit (25-bit Single-Turn, 12 Multi-Turn) SSI encoder is driving channel 1.

### Step 1: Global Control Register Setup (Technique 2)

[23:16]	[15:12]	11	10	9	8	7	6	5	4	3	2	1	0
M_Divisor	N_Divisor			Trigger Clock	Trigger Edge	Trigger Delay			Protocol Code				

Bit Assignment	Description	Resulting Global Control Register
M=99, N=0	Serial Clock 1.0 MHz	\$630002
Trigger Clock=0	Trigger on Phase Clock (recommended)	
Trigger Edge=0	Rising Edge (recommended)	
Trigger Delay=0	No delay	
Protocol Code=2	SSI	

### Step 2: Channel Control Register Setup (Technique 2)

[23:16]	15	14	13	12	11	10	[9:6]	[5:0]
	Parity Type	Trigger Mode	Trigger Enable	GtoB	RxData Ready /SENC_MODE			PositionBits/Resolution

Bit Assignment	Description	Resulting Channel Control Register
Parity Type=0	None	\$001425
Trigger Mode=0	Continuous Trigger (common)	
Trigger Enable=1	Must be set to 1 (common)	
SENC_MODE=1	Enable Serial Driver (common)	
Resolution (37 bits)=100101(\$25)	Position Bits	

## Control Registers Power-On PLC Example (Technique 2)

The Global and Channel Control words have to be executed once on power-up

```
//===== NOTES ABOUT THIS PLC EXAMPLE =====//
// This PLC example utilizes: - M5990 through M5991
//                               - Coordinate system 1 Timer 1 (I5111)
// Make sure that current and/or future configurations do not create conflicts with
// these parameters.
//=====//

M5990..5991->* ; Self-referenced M-Variables
M5990..5991=0 ; Reset at download
//===== GLOBAL CONTROL REGISTERS =====//
#define SSIGlobalCtrl1_4 M5990 ; Channels 1-4 SSI global control register
SSIGlobalCtrl1_4->X:$7880F,0,24,U ; Channels 1-4 SSI global control register address
//===== CHANNEL CONTROL REGISTERS =====//
#define Ch1SSICtrl M5991 ; Channel 1 SSI control register
Ch1SSICtrl->X:$78800,0,24,U ; Channel 1 SSI control register Address

//===== POWER-ON PLC EXAMPLE, GLOBAL & CHANNEL CONTROL REGISTERS =====//
Open PLC 1 Clear
SSIGlobalCtrl1_4=$630002 ; Trigger at Phase, 1 MHz serial Clock (M=99, N=0)-User Input
Ch1SSICtrl=$001425 ; Channel 1 SSI control register -User Input

I5111=500*8388608/I10 While (I5111>0) EndWhile ; ½ sec delay
Disable PLC 1 ; Execute once on power-up or reset
Close
//=====//
```

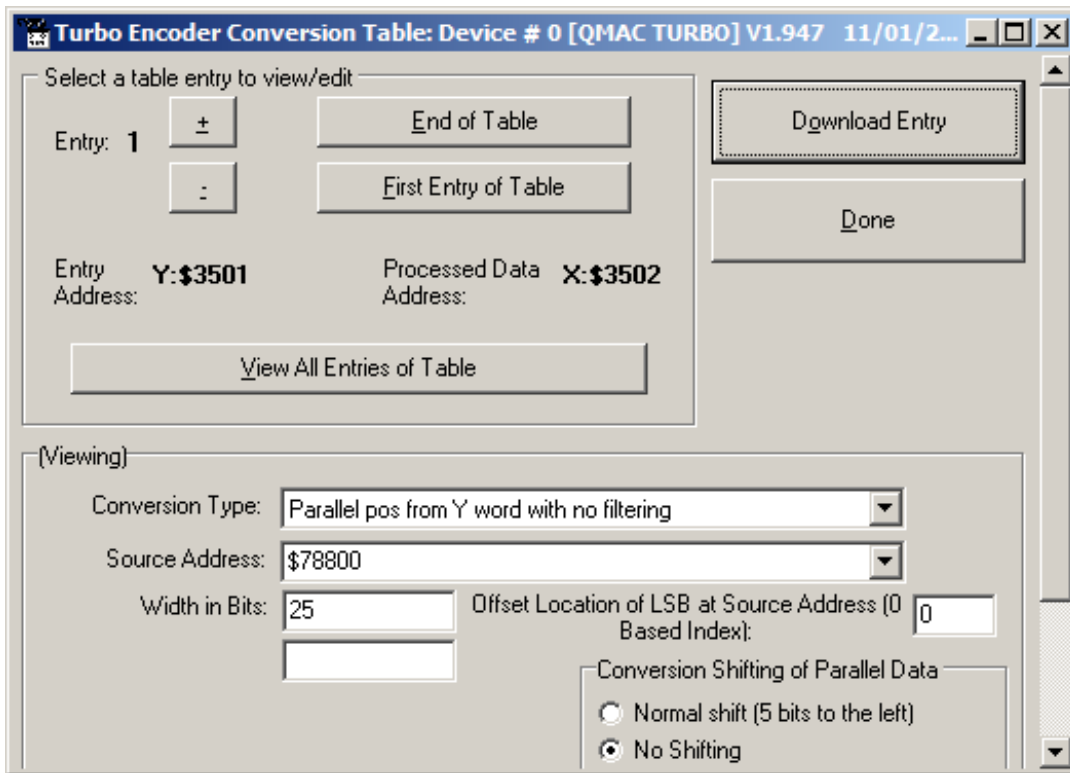


**Step 3: Encoder Conversion Table Setup — for Position (Technique 2)**

- Conversion Type: “Parallel pos from Y word with no filtering”
- Width in Bits is the single-turn/absolute resolution in bits (25 bits in this example)
- Offset Location of LSB: leave at zero
- No shifting (recommended for higher resolution encoders)
- Source Address

Card Number	Channel Number	Source Address (Data Register A)
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

- Click on “Download Entry”:



Record the location of the processed data (X:\$3502 in this example). This is the position and velocity pointers’ address location. This ECT configuration is a 2-line entry, and its equivalent I8000s variables would look like:

```
I8000=$2F8800 ; Unfiltered parallel pos of location Y:$78800
I8001=$19000 ; Width and Offset. Processed result at $3502
```

The Position and Velocity Pointer's address is the location of the processed data. Also, with higher resolution encoders, it is highly recommended to set the position- and velocity- loop scale factors to 1 to avoid saturation:

```
I100=1           ; Mtr #1 active - remember to activate the channel to see feedback
I103=$3502      ; Mtr #1 position loop feedback address -User Input
I104=$3502      ; Mtr #1 velocity loop feedback address -User Input
I108=1          ; Mtr #1 position-loop scale factor
I109=1          ; Mtr #1 velocity-loop scale factor
```



*Note*

At this point of the setup process, the user should be able to move the motor/encoder shaft by hand and see “motor” counts in the position window.

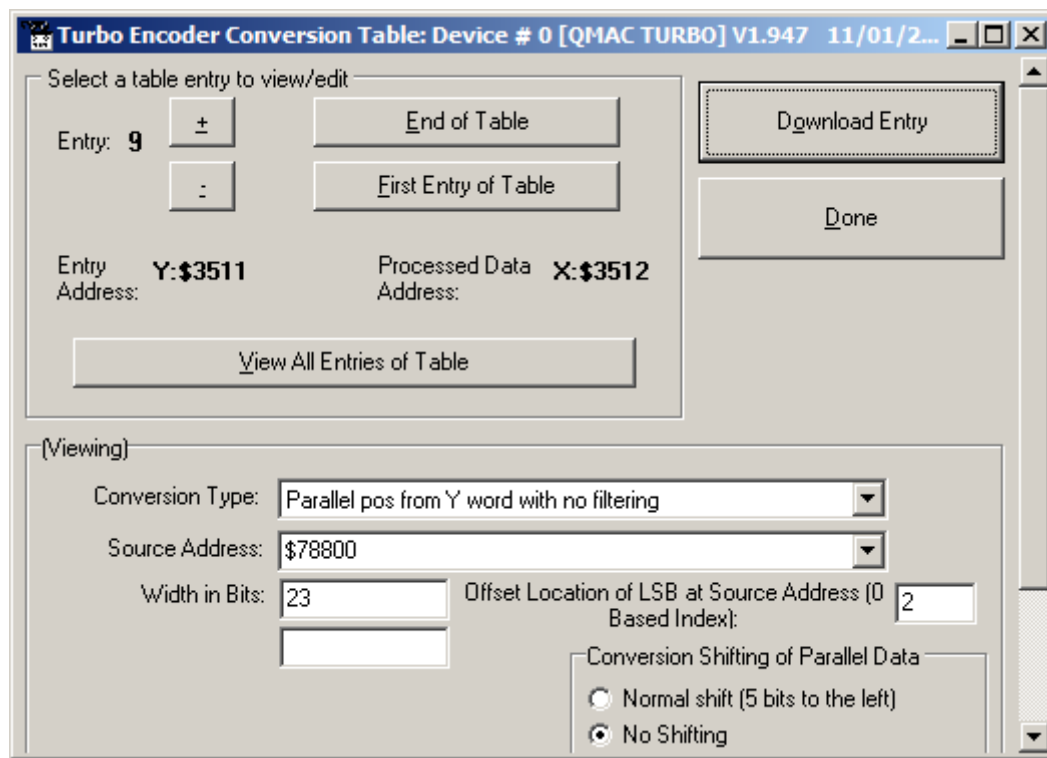
---

### Step 4: Encoder Conversion Table Setup — for Commutation (Technique 2)

Commutation with Turbo PMAC does not require high resolution data, so in order to avoid saturations with higher resolution SSI encoders one must limit the commutation to 23 bits (upper). This will also alleviate possible quantization noise effects.

This entry should preferably be inserted at the end of the Encoder Conversion Table after all position/velocity encoders in the system have been processed or have been allocated to specific entries.

- Conversion Type: “Parallel pos from Y word with no filtering”
- Width in Bits is 23 (fixed)
- Offset Location of LSB = Single Turn data stream – 23 (i.e. 25-23=2)
- No shifting
- Source Address (same as position ECT)
- Click on “Download Entry”:



Record the location of the processed data (X:\$3512 in this example). This is the commutation position address. Equivalent Turbo PMAC Script:

```
I8016=$2F8800 ; Unfiltered parallel pos of location Y:$78800 -User Input
I8017=$17002 ; Width and Offset. Processed result at X:$3512 -User Input
```

The commutation enable, and position address would then be:

```
I101=1 ; Mtr #1 Commutation Enable, from X Register
I183=$3512 ; Mtr #1 Commutation Position Address -User Input
```

### Step 5: Absolute Power-On Position Read (Technique 2)

The absolute power-on position with Technique 2 cannot be performed with the automatic Turbo PMAC feature (Ixx10, Ixx95) because the data is unshifted, and thus it must be constructed from the raw serial data registers.

The following example PLC reads and constructs the absolute position for channels 1 through 8 (channels 1–4 on the 1<sup>st</sup> ACC-84S, channels 4–8 on the 2<sup>nd</sup> ACC-84S). It is already configured for the user to input their encoder information (single turn and multi turn resolutions), and then specify which channels should perform an absolute power-on read.

### Using the Absolute Position Read Example PLC

Under the “User Input” section:

1. Enter single turn (ChxSTRes) and multi turn (ChxMTRes) resolutions in bits for each encoder. For strictly absolute single turn encoders, set the multi-turn resolution (ChxMTRes) to zero.
2. In ChAbsSel, specify which channels are desired to perform an absolute position read. This value is in hexadecimal. A field value of 1 specifies that this channel is connected, whereas 0 specifies that the channel is not connected and/or should not perform an absolute read. Examples:

Reading Absolute Position, Channels 1 through 4	<b>Channel #:</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	→ ChAbsSel=\$0F
	ChAbsSel (Binary):	0	0	0	0	1	1	1	1	
	ChAbsSel (Hex):	0				F				

Reading Absolute Position, Channels 1, 3, 5, 7	<b>Channel #:</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	→ ChAbsSel=\$55
	ChAbsSel (Binary):	0	1	0	1	0	1	0	1	
	ChAbsSel (Hex):	5				5				

```

//===== NOTES ABOUT THIS PLC EXAMPLE =====//
// This PLC example utilizes: - M6000 through M6035
//                               - P7000 through P7032
// Make sure that current and/or future configurations do not create conflicts with
// these parameters.
//=====//

M6000..6035->*      ; Self-referenced M-Variables
M6000..6035=0      ; Reset M-Variables at download
P7000..7032=0      ; Reset P-Variables at download

//===== USER INPUT =====//
#define Ch1STRes P7000      #define Ch1MTRes P7001
#define Ch2STRes P7002      #define Ch2MTRes P7003
#define Ch3STRes P7004      #define Ch3MTRes P7005
#define Ch4STRes P7006      #define Ch4MTRes P7007
#define Ch5STRes P7008      #define Ch5MTRes P7009
#define Ch6STRes P7010      #define Ch6MTRes P7011
#define Ch7STRes P7012      #define Ch7MTRes P7013
#define Ch8STRes P7014      #define Ch8MTRes P7015

Ch1STRes=25  Ch1MTRes=12    ; Ch1 Multi Turn and Single Turn Resolutions --User Input
Ch2STRes=25  Ch2MTRes=12    ; Ch2 Multi Turn and Single Turn Resolutions --User Input
Ch3STRes=25  Ch3MTRes=12    ; Ch3 Multi Turn and Single Turn Resolutions --User Input
Ch4STRes=25  Ch4MTRes=12    ; Ch4 Multi Turn and Single Turn Resolutions --User Input
Ch5STRes=25  Ch5MTRes=12    ; Ch5 Multi Turn and Single Turn Resolutions --User Input
Ch6STRes=25  Ch6MTRes=12    ; Ch6 Multi Turn and Single Turn Resolutions --User Input
Ch7STRes=25  Ch7MTRes=12    ; Ch7 Multi Turn and Single Turn Resolutions --User Input
Ch8STRes=25  Ch8MTRes=12    ; Ch8 Multi Turn and Single Turn Resolutions --User Input
#define ChAbsSel P7016      ; Select Channels using absolute read (in Hexadecimal)
ChAbsSel=$FF                ; Channels selected for absolute position read -User Input
    
```

```
//===== DEFINITIONS & SUBSTITUTIONS =====//
#define SerialRegA      M6000 ; SSI Serial Data Register A
#define SerialRegB      M6001 ; SSI Serial Data Register B
#define Two2STDec       M6002 ; 2^STRes in decimal, for shifting operations
#define Two2STHex       M6003 ; 2^STRes in Hexadecimal, for bitwise operations
#define Two2MTDec       M6004 ; 2^MTRes in decimal, for shifting operations
#define Two2MTHex       M6005 ; 2^MTRes in Hexadecimal, for bitwise operations
#define MTTemp1         M6006 ; Multi Turn Data temporary holding register 1
#define MTTemp2         M6007 ; Multi Turn Data temporary holding register 2
#define STTemp1         M6008 ; Single Turn Data temporary holding register 1
#define STTemp2         M6009 ; Single Turn Data temporary holding register 2
#define ChNoHex         M6010 ; Channel Number in Hex
#define ChAbsCalc       M6011 ; Abs. calc. flag (=1 true do read, =0 false do not do read)
#define LowerSTBits     P7017 ; Lower Single Turn Bits, RegA
#define UpperSTBits     P7018 ; Upper Single Turn Bits, RegB (where applicable)
#define LowerMTBits     P7019 ; Lower Multi Turn Bits, RegA (where applicable)
#define UpperMTBits     P7020 ; Upper Multi Turn Bits, RegB (where applicable)
#define STData          P7021 ; Single Turn Data Word
#define MTData          P7022 ; Multi Turn Data Word
#define NegTh           P7023 ; Negative Threshold
#define Temp1           P7024 ; General Temporary holding register 1
#define Temp2           P7025 ; General Temporary holding register 2
#define SerialBase      P7026 ; Indirect addressing index for serial registers, 6020
#define ChBase          P7027 ; Indirect addressing index for channel No, 162
#define ChNo            P7028 ; Current Channel Number
#define ResBase         P7029 ; Indirect Addressing index for resolution input, 6000
#define STRes           P7030 ; Single Turn Resolution of currently addressed channel
#define MTRes           P7031 ; Multi Turn Resoluition of currently addressed channel
#define PsfBase         P7032 ; Indirect addressing for position scale factor Ixx08, 108
// SSI Serial Data Registers A (left column) and B (right column)
M6020->Y:$78800,0,24,U      M6021->Y:$78801,0,24,U      ; Channel 1
M6022->Y:$78804,0,24,U      M6023->Y:$78805,0,24,U      ; Channel 2
M6024->Y:$78808,0,24,U      M6025->Y:$78809,0,24,U      ; Channel 3
M6026->Y:$7880C,0,24,U      M6027->Y:$7880D,0,24,U      ; Channel 4
M6028->Y:$78820,0,24,U      M6029->Y:$78821,0,24,U      ; Channel 5
M6030->Y:$78824,0,24,U      M6031->Y:$78825,0,24,U      ; Channel 6
M6032->Y:$78828,0,24,U      M6033->Y:$78829,0,24,U      ; Channel 7
M6034->Y:$7882C,0,24,U      M6035->Y:$7882D,0,24,U      ; Channel 8

//===== PLC SCRIPT CODE =====//
Open PLC 1 Clear
ChNo=0
While(ChNo!>7) ; Loop for 8 Channels
  ChNo=ChNo+1
  ChNoHex=exp((ChNo-1)*ln(2))
  ChAbsCalc=(ChAbsSel&ChNoHex)/ChNoHex
  If (ChAbsCalc!=0) ; Absolute read on this channel?
    SerialBase=6020+(ChNo-1)*2
    SerialRegA=M(SerialBase)
    SerialRegB=M(SerialBase+1)
    ResBase=7000+(ChNo-1)*2
    STRes=P(ResBase)
    MTRes=P(ResBase+1)
    STData=0
    MTData=0
  If (STRes!>24) ; Single Turn Res<=24
    //=====SINGLE TURN DATA=====//
    Two2STDec=exp(STRes*ln(2))
    Two2STHex=Two2STDec-1
    STData=SerialRegA&Two2STHex
    //=====MULTI TURN DATA=====//
    Two2MTDec=exp(MTRes*ln(2))
    Two2MTHex=Two2MTDec-1
    If (MTRes=0)
      LowerMTBits=0
      UpperMTBits=0
      Two2MTDec=0
      Two2MTHex=0
      MTData=0
    Else
      LowerMTBits=24-STRes
```

```
STTemp1=exp(LowerMTBits*ln(2))
STTemp2=0
UpperMTBits=MTRes-LowerMTBits
MTTemp1=exp(LowerMTBits*ln(2))
MTTemp2=exp(UpperMTBits*ln(2))
Temp1=(SerialRegA/Two2STDec)&(MTTemp1-1)
Temp2=SerialRegB&(MTTemp2-1)
MTData=Temp2*STTemp1+Temp1
EndIf
Else ; Single Turn Res>24
//=====SINGLE TURN DATA=====//
LowerSTBits=24
UpperSTBits=STRes-24
STTemp1=exp(UpperSTBits*ln(2))
STTemp2=STTemp1-1
Two2STDec=16777216*STTemp1
Two2STHex=Two2STDec-1
STData=(SerialRegB&STTemp2)*16777216+SerialRegA
//=====MULTI TURN DATA=====//
If (MTRes=0)
  LowerMTBits=0
  UpperMTBits=0
  Two2MTDec=0
  Two2MTHex=0
  MTData=0
Else
  Two2MTDec=exp(MTRes*ln(2))
  Two2MTHex=Two2MTDec-1
  LowerMTBits=0
  UpperMTBits=MTRes
  MTTemp1=exp(UpperMTBits*ln(2))
  MTTemp2=MTTemp1-1
  MTData=(SerialRegB/STTemp1)&MTTemp2
EndIf
EndIf
//=====ASSEMBLING ACTUAL POSITION=====//
ChBase=162+(ChNo-1)*100
NegTh=Two2MTDec/2
PsfBase=108+(ChNo-1)*100
If (MTData!>NegTh)
  M(ChBase)=(MTData*Two2STDec+STData)* I(PsfBase)
Else
  M(ChBase)=-((Two2MTHex-MTData)*Two2STDec)+(Two2STDec-STData)* I(PsfBase)
EndIf
EndIf
EndWhile
ChNo=0
Disable PLC 1
Close
```

## **EnDat 2.2 Feedback Configuration**

The EnDat 2.2 feedback option allows the Turbo Clipper to connect to up to eight channels of EnDat 2.2 type devices: four channels per ACC-84S, for a total of eight EnDat 2.2 devices when using two ACC-84S cards. Setting up the EnDat interface correctly requires the programming of two essential control registers:

- Global Control Registers
- Channel Control Registers

The resulting data is found in:

- EnDat Data Registers

## Description of EnDat Registers

### Global Control Registers

The Global Control Register controls the clock settings and trigger settings of the feedback protocol and it is located at X-word of the (base address) + \$F. Its default value is \$002003. The following table shows address locations for the Global Control Register for different address settings:

Card Number	Base Address	Global Control Register
1 <sup>st</sup> ACC-84S	\$78800	X:\$7880F
2 <sup>nd</sup> ACC-84S	\$78820	X:\$7882F

The Global Control register is used to program the serial encoder interface clock frequency *SER\_Clock* and configure the serial encoder interface trigger clock. *SER\_Clock* is generated from a two-stage divider clocked at 100 MHz as follows:

$$SER\_Clock = \frac{100}{25 \times (M + 1) \times 2^N} MHz$$

M	N	Serial Clock Frequency
0	0	4.0 MHz
0	2	1.0 MHz
0	3	500 KHz
0	4	250 KHz
...	...	...

Default Settings M=0, N=2 => 1 MHz transfer rate

There are two external trigger sources; phase and servo. Bits [9:8] in the Global Control register are used to select the source and active edge to use as the internal serial encoder trigger. The internal trigger is used by all four channels to initiate communication with the encoder. To compensate for external system delays, this trigger has a programmable 4-bit delay setting in 20 µsec increments.

23--16	15--12	11	10	9	8	7	6	5	4	3	2	1	0
M_Divisor	N_Divisor			Trigger Clock	Trigger Edge	Trigger Delay				Protocol Code			

Bit	Type	Default	Name	Description
[23:16]	R/W	0x00	M_Divisor	Intermediate clock frequency for <i>SER_Clock</i> . The intermediate clock is generated from a (M+1) divider clocked at 100 MHz.
[15:12]	R/W	0x0	N_Divisor	Final clock frequency for <i>SER_Clock</i> . The final clock is generated from a $2^N$ divider clocked by the intermediate clock.
[11:10]	R	00	Reserved	Reserved and always reads zero.
[09]	R/W	0	TriggerClock	Trigger clock select: 0= PhaseClock 1= ServoClock
[08]	R/W	0	TriggerEdge	Active clock edge select: 0= rising edge 1= falling edge
[07:04]	R/W	0x0	TriggerDelay	Trigger delay program relative to the active edge of the trigger clock. Units are in increments of 20 usec.
[03:00]	R	0x3	ProtocolCode	This read-only bit field is used to read the serial encoder interface protocol supported by the FPGA. A value of 0x3 defines this protocol as EnDat.



## Channel Control Registers

Each channel also has its Channel Control Register which controls the functionality of each channel. The user uses this register to:

- Configure the number of position bits in the serial bit stream
- Enable or disable channels through the SENC\_MODE bit
- Enable or disable communication with the encoder using the trigger control bit

The Channel Control Register appears at the following memory locations:

Card Number	Channel Number	Address
1 <sup>st</sup> ACC-84S	1	X:\$78800
1 <sup>st</sup> ACC-84S	2	X:\$78804
1 <sup>st</sup> ACC-84S	3	X:\$78808
1 <sup>st</sup> ACC-84S	4	X:\$7880C
2 <sup>nd</sup> ACC-84S	1	X:\$78820
2 <sup>nd</sup> ACC-84S	2	X:\$78824
2 <sup>nd</sup> ACC-84S	3	X:\$78828
2 <sup>nd</sup> ACC-84S	4	X:\$7882C

The diagram below describes the functionality of each bit range in the Channel Control Registers:

23	22	[21:16]	15	14	13	12	11	10	[9:6]	[5:0]
		Command Code			Trigger Mode	Trigger Enable		RxData Ready /SENC_MODE		PositionBits/ Resolution

Bit	Type	Default	Name	Description
[23:22]	R	0x000	Reserved	Reserved and always reads zero.
[21:16]	R	0x00	Command Code	111000 – Encoder to Send Position (EnDat2.2 only) 010101 – Encoder to Receive Reset (EnDat2.2 only) 000111 – Encoder to Send Position (EnDat 2.1 & 2.2) 101010 – Encoder to Receive Reset (EnDat 2.1 & 2.2)
[15:14]	R	00	Reserved	Reserved and always reads zero.
[13]	R/W	0	Trigger Mode	Trigger Mode: 0= continuous trigger 1= one-shot trigger All triggers occur at the defined Phase/Servo clock edge and delay setting. See <b>Global Control</b> register for these settings.
[12]	R/W	0	Trigger Enable	Enable trigger: 0= disabled 1= enabled This bit must be set for either trigger mode. If the Trigger Mode bit is set for one-shot mode, the hardware will automatically clear this bit after the trigger occurs.
[11]	R/W	0	Reserved	Reserved and always reads zero.
[10]	R	0	RxDData Ready	This read-only bit provides the received data status. It is low while the interface logic is communicating (busy) with the serial encoder. It is high when all the data has been received and processed.
	W	0	SENC_MODE	This write-only bit is used to enable the output drivers for the SENC_SDO, SENC_CLK, SENC_ENA pins for each respective channel. Writing a 0 to this bit disables the channel Writing a 1 to this bit enables the channel
[09:06]	R	0x0	Reserved	Reserved and always reads zero.
[05:00]	W	0x00	Position Bits	This bit field is used to define the number of position data bits or encoder resolution: Range is 12 – 40 (001100 –101000)



*Note*

RxDData Ready and SENC\_MODE share bit 10 of the Channel Specific Control Register. Write to bit 10 to configure SENC\_MODE and then read from bit 10 thereafter for the state of RxDData Ready.

## EnDat Data Registers

The EnDat data is stored in 4 memory locations: EnDat Data Registers A, B, C, and D.

The EnDat Data Register A register holds the 24 bits of the encoder position data. If the data exceeds the 24 available bits in this register, the upper overflow bits are LSB justified and readable in the EnDat Data Register B, which also holds error flags. The error bit flag is always returned by the encoder, except for a Reset command. The *CRC* error bit is set if the return data fails the *CRC* verification. The timeout error flag is set if the SEIGATE3 does not receive a response from the encoder.

EnDat Data Registers C and D are reserved and always read zero.

EnDat Data Register B				EnDat Data Register A	
23	22	21	[20:16]	[15:0]	[23:0]
TimeOut Err	CRC Err	Err flag		Position Data [39:24]	Position Data [23:0]

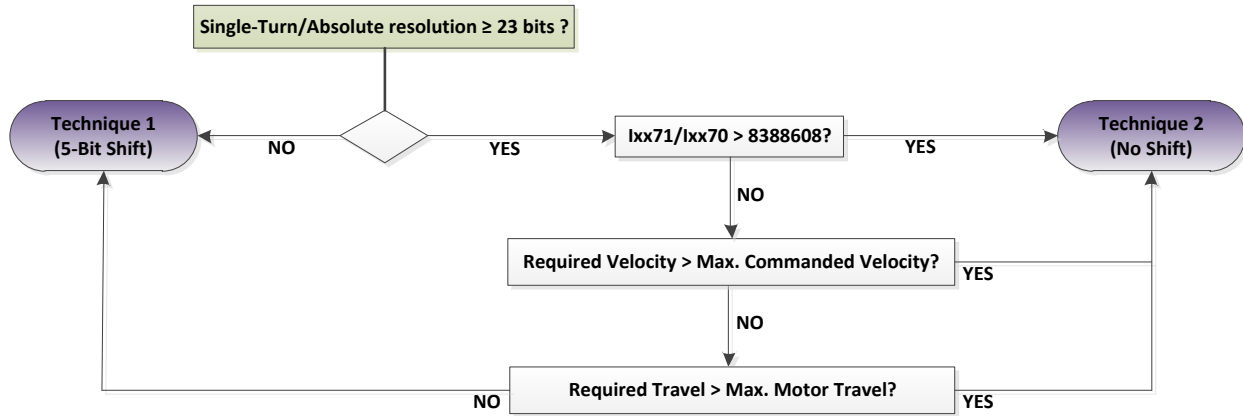
Card Number	Channel Number	EnDat Data Register A	EnDat Data Register B
1 <sup>st</sup> ACC-84S	1	Y:\$78800	Y:\$78801
1 <sup>st</sup> ACC-84S	2	Y:\$78804	Y:\$78805
1 <sup>st</sup> ACC-84S	3	Y:\$78808	Y:\$78809
1 <sup>st</sup> ACC-84S	4	Y:\$7880C	Y:\$7880D
2 <sup>nd</sup> ACC-84S	1	Y:\$78820	Y:\$78821
2 <sup>nd</sup> ACC-84S	2	Y:\$78824	Y:\$78825
2 <sup>nd</sup> ACC-84S	3	Y:\$78828	Y:\$78829
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C	Y:\$7882D

EnDat Registers C and D are listed here for future use and documentation purposes only. They do not pertain to the EnDat setup and always read zero.

Card Number	Channel Number	EnDat Data Register C	EnDat Data Register D
1 <sup>st</sup> ACC-84S	1	Y:\$78802	Y:\$78803
1 <sup>st</sup> ACC-84S	2	Y:\$78806	Y:\$78807
1 <sup>st</sup> ACC-84S	3	Y:\$7880A	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880E	Y:\$7880F
2 <sup>nd</sup> ACC-84S	1	Y:\$78822	Y:\$78823
2 <sup>nd</sup> ACC-84S	2	Y:\$78826	Y:\$78827
2 <sup>nd</sup> ACC-84S	3	Y:\$7882A	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882E	Y:\$7882F

## Configuring EnDat Encoders

The [Using Absolute Serial Encoders with Turbo PMAC](#) section suggests two techniques for processing absolute serial data. The following chart shows the recommended method for using a given absolute serial encoder attached to a specific motor (number of poles):



If the application requirements are unknown or unrestricted. It is always acceptable to setup absolute serial encoder using:

- Technique 1 for single-turn/absolute resolutions < 23 bits
- Technique 2 for single-turn/absolute resolutions ≥ 23 bits

With either technique, the following steps assure the proper setting of a motor with serial absolute encoders:

Step #	Description	Technique 1 (5-bit shift)	Technique 2 (no shift)
1	Global Control Register (per set of 4-axis)	√	√
2	Channel Control Register (per Channel)	√	√
3	Encoder Conversion Table (for Position)	Parallel Y-Word unfiltered normal 5-bit shift	Parallel Y-Word unfiltered no shifting
4	Encoder Conversion Table (for Commutation)	Directly from raw data	Parallel Y-Word unfiltered unshifted, limited to 23 bits
5	Absolute Power-On Position Read	Automatic using Ixx10, and Ixx95	Custom PLC (see example)
6	Absolute Power-On Phasing	Custom PLC (see example)	Automatic using Ixx75, Ixx81, and Ixx91

## Configuring EnDat Encoders Example Using Technique 1

A 25-bit (13-bit Single-Turn, 12 Multi-Turn) EnDat 2.2 encoder is driving channel 1.

### Step 1: Global Control Register Setup (Technique 1)

[23:16]	[15:12]	11	10	9	8	7	6	5	4	3	2	1	0
M_Divisor	N_Divisor			Trigger Clock	Trigger Edge	Trigger Delay			Protocol Code				

Bit Assignment	Description	Resulting Global Control Register
M=0, N=0	Serial Clock 4.0 MHz	\$000003
Trigger Clock=0	Trigger on Phase Clock (recommended)	
Trigger Edge=0	Rising Edge (recommended)	
Trigger Delay=0	No delay	
Protocol Code=3	EnDat2.2	

### Step 2: Channel Control Register Setup (Technique 1)

23	22	[21:16]	15	14	13	12	11	10	[9:6]	[5:0]
		Command Code			Trigger Mode	Trigger Enable		RxDData Ready /SENC_MODE		PositionBits/Resolution

Bit Assignment	Description	Resulting Channel Control Register
Command Code= 111000 (\$38)	Send Position EnDat 2.2 Only	\$381419
Trigger Mode=0	Continuous Trigger (common)	
Trigger Enable=1	Must be set to 1 (common)	
Senc Mode=1	Enable Serial Driver (common)	
Resolution (25 bits)=11001 (\$19)	EnDat 2.2	

## Control Registers Power-On PLC Example (Technique 1)

The Global and Channel Control words have to be executed once on power-up

```
//===== NOTES ABOUT THIS PLC EXAMPLE =====//
// This PLC example utilizes: - M5990 through M5991
//                               - Coordinate system 1 Timer 1 (I5111)
// Make sure that current and/or future configurations do not create conflicts with
// these parameters.
//=====//

M5990..5991->* ; Self-referenced M-Variables
M5990..5991=0 ; Reset at download
//===== GLOBAL CONTROL REGISTERS =====//
#define EnDatGlobalCtrl1_4 M5990 ; Channels 1-4 EnDat global control register
EnDatGlobalCtrl1_4->X:$7880F,0,24,U ; Channels 1-4 EnDat global control register address
//===== CHANNEL CONTROL REGISTERS =====//
#define Ch1EnDatCtrl M5991 ; Channel 1 EnDat control register
Ch1EnDatCtrl->X:$78800,0,24,U ; Channel 1 EnDat control register Address

//===== POWER-ON PLC EXAMPLE, GLOBAL & CHANNEL CONTROL REGISTERS =====//
Open PLC 1 Clear
EnDatGlobalCtrl1_4=$3 ; Trigger at Phase, 4 MHz serial Clock -User Input
Ch1EnDatCtrl=$381419 ; Channel 1 EnDat control register -User Input

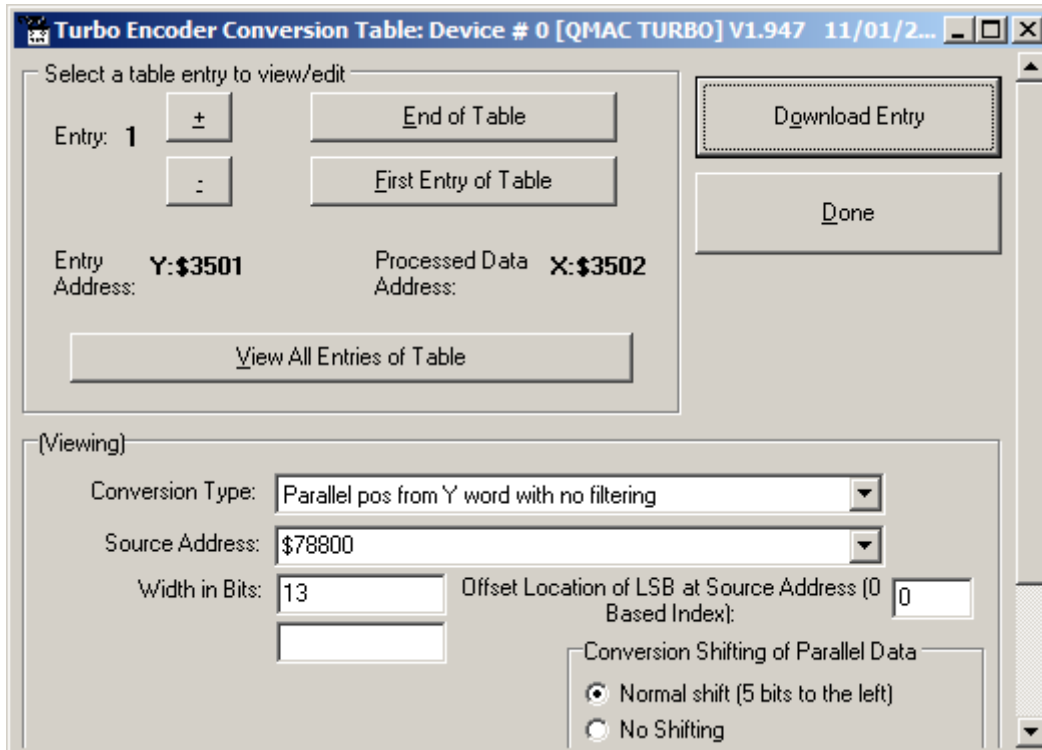
I5111=500*8388608/I10 While(I5111>0) EndWhile ; ½ sec delay
Disable PLC 1 ; Execute once on power-up or reset
Close
//=====//
```

Step3: Encoder Conversion Table Setup - for Position (Technique 1)

- Conversion Type: “Parallel pos from Y word with no filtering”
- Width in Bits is the single-turn/absolute resolution in bits (13 bits in this example)
- Offset Location of LSB: leave at zero
- Normal Shift (5 bits to the left)
- Source Address

Card Number	Channel Number	EnDat Data Register A
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

- Click on “Download Entry”



Record the location of the processed data (X:\$3502 in this example). This is the position and velocity pointers' address location. Equivalent Turbo PMAC Script:

```
I8000=$278800      ; Unfiltered parallel pos of location Y:$78800
I8001=$00D000     ; Width (13 bits) and Offset (none), 5 bit left shift
                  ; Processed result at $3502
```

The Position and Velocity Pointer's address is the location of the processed data:

I100=1	; Mtr #1 active - remember to activate the channel to see feedback
I103=\$3502	; Mtr #1 position loop feedback address
I104=\$3502	; Mtr #1 velocity loop feedback address
I183=\$3502	; Mtr #1 commutation feedback address



Note

- At this point of the setup process, the user should be able to move the motor/encoder shaft by hand and see “motor” counts in the position window.
- Some EnDat 2.2 Encoders do not support additional information (\$38 command mode in bits 24:17 of Channel Control Register). If this is the case, use \$07 instead in bits 24:17 of Channel Control Register (see Heidenhain command set table).

Heidenhain Command Set Table

No.	Mode command	Mode bit					
		M2	M1	M0	(M2)	(M1)	(M0)
1	Encoder transmit position values	0	0	0	1	1	1
2	Selection of the memory area	0	0	1	1	1	0
3	Encoder receive parameters	0	1	1	1	0	0
4	Encoder transmit parameters	1	0	0	0	1	1
5	Encoder receive reset <sup>1)</sup>	1	0	1	0	1	0
6	Encoder transmit test values	0	1	0	1	0	1
7	Encoder receive test commands	1	1	0	0	0	1
8	Encoder transmit position value with additional information	1	1	1	0	0	0
9	Encoder transmit position value and receive selection of memory area <sup>2)</sup>	0	0	1	0	0	1
10	Encoder transmit position value and receive parameters <sup>2)</sup>	0	1	1	0	1	1
11	Encoder transmit position value and transmit parameters <sup>2)</sup>	1	0	0	1	0	0
12	Encoder transmit position value and receive error reset <sup>2)</sup>	1	0	1	1	0	1
13	Encoder transmit position value and receive test command <sup>2)</sup>	1	1	0	1	1	0
14	Encoder receive communication command <sup>3)</sup>	0	1	0	0	1	0

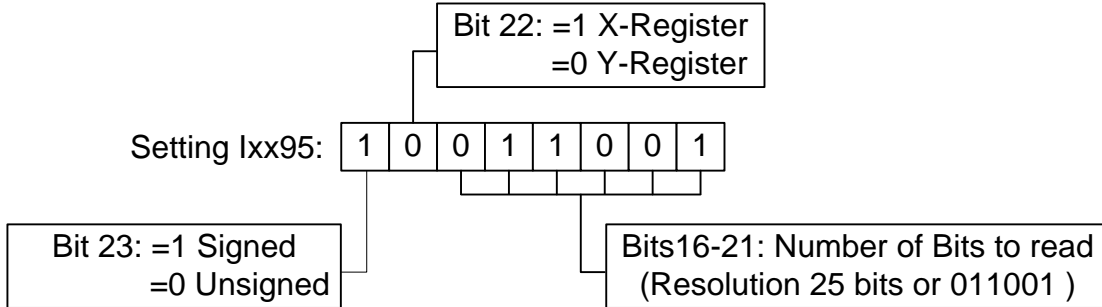
**Step 5: Absolute Power-On Position Read (Technique 1)**

With Technique 1 (normal 5-bit shift), the absolute power-on read can be configured using the automatic feature in Turbo PMAC:

**Example 1:** Channel 1 driving a 25-bit EnDat (13-bit single turn, 12-bit multi-turn) encoder:

```
I180=2 ; Absolute power-on read enabled
I110=$78800 ; Absolute Servo power-on position address
I195=$990000 ; Parallel Read, 25 bits, Signed, from Y-Register -User Input
```

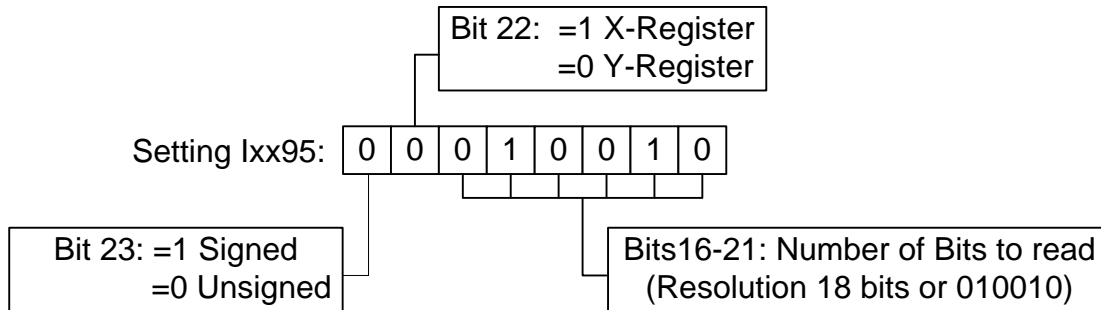
In this mode, PMAC reads and reports 25 bits: it takes the low 24 bits from Y:\$078800, and uses the lowest bit of Y:\$078801 as the 25<sup>th</sup> bit. With this setting of Ixx80, the actual position is reported automatically on power-up. Otherwise, a #1\$\* command is necessary to read the absolute position.



**Example 2:** Channel 1 driving a purely 18-bit absolute EnDat (18-bit single turn, 0 multi-turn) encoder:

```
I180=2 ; Absolute power-on read enabled
I110=$78800 ; Absolute Servo power-on position address
I195=$120000 ; Parallel Read, 18 bits, Unsigned, from Y-Register -User Input
```

In this mode, PMAC reads and reports 18 bits, the low 18 bits from Y:\$078800. With this setting of Ixx80, the actual position is reported automatically on power-up. Otherwise, a #1\$\* command is necessary to read the absolute position.



*Note*

With purely absolute serial encoders (no multi-turn data), the user must configure the power-on position format for unsigned operation.



## Configuring EnDat Encoders Example Using Technique 2

A 37-bit (25-bit Single-Turn, 12 Multi-Turn) EnDat 2.2 encoder is driving channel 1.

### Step 1: Global Control Register Setup (Technique 2)

[23:16]	[15:12]	11	10	9	8	7	6	5	4	3	2	1	0
M_Divisor	N_Divisor			Trigger Clock	Trigger Edge	Trigger Delay			Protocol Code				

Bit Assignment	Description	Resulting Global Control Register
M=0, N=0	Serial Clock 4.0 MHz	\$000003
Trigger Clock=0	Trigger on Phase Clock (recommended)	
Trigger Edge=0	Rising Edge (recommended)	
Trigger Delay=0	No delay	
Protocol Code=3	EnDat 2.2	

### Step 2: Channel Control Register Setup (Technique 2)

23	22	[21:16]	15	14	13	12	11	10	[9:6]	[5:0]
		Command Code			Trigger Mode	Trigger Enable		RxDData Ready /SENC_MODE		PositionBits/Resolution

Bit Assignment	Description	Resulting Channel Control Register
Command Code= 111000 (\$38)	Send Position EnDat 2.2 Only	\$381425
Trigger Mode=0	Continuous Trigger (common)	
Trigger Enable=1	Must be set to 1 (common)	
Senc Mode=1	Enable Serial Driver (common)	
Resolution (37 bits)=100101 (\$25)	EnDat 2.2	

## Control Registers Power-On PLC Example (Technique 2)

The global and channel control words have to be executed once on power-up:

```
//===== NOTES ABOUT THIS PLC EXAMPLE =====//
// This PLC example utilizes: - M5990 through M5991
//                               - Coordinate system 1 Timer 1 (I5111)
// Make sure that current and/or future configurations do not create conflicts with
// these parameters.
//=====//

M5990..5991->* ; Self-referenced M-Variables
M5990..5991=0 ; Reset at download
//===== GLOBAL CONTROL REGISTERS =====//
#define EnDatGlobalCtrl1_4 M5990 ; Channels 1-4 EnDat global control register
EnDatGlobalCtrl1_4->X:$7880F,0,24,U ; Channels 1-4 EnDat global control register address
//===== CHANNEL CONTROL REGISTERS =====//
#define Ch1EnDatCtrl M5991 ; Channel 1 EnDat control register
Ch1EnDatCtrl->X:$78800,0,24,U ; Channel 1 EnDat control register Address

//===== POWER-ON PLC EXAMPLE, GLOBAL & CHANNEL CONTROL REGISTERS =====//
Open PLC 1 Clear
EnDatGlobalCtrl1_4=$3 ; Trigger at Phase, 4 MHz serial Clock -User Input
Ch1EnDatCtrl=$381425 ; Channel 1 EnDat control register -User Input

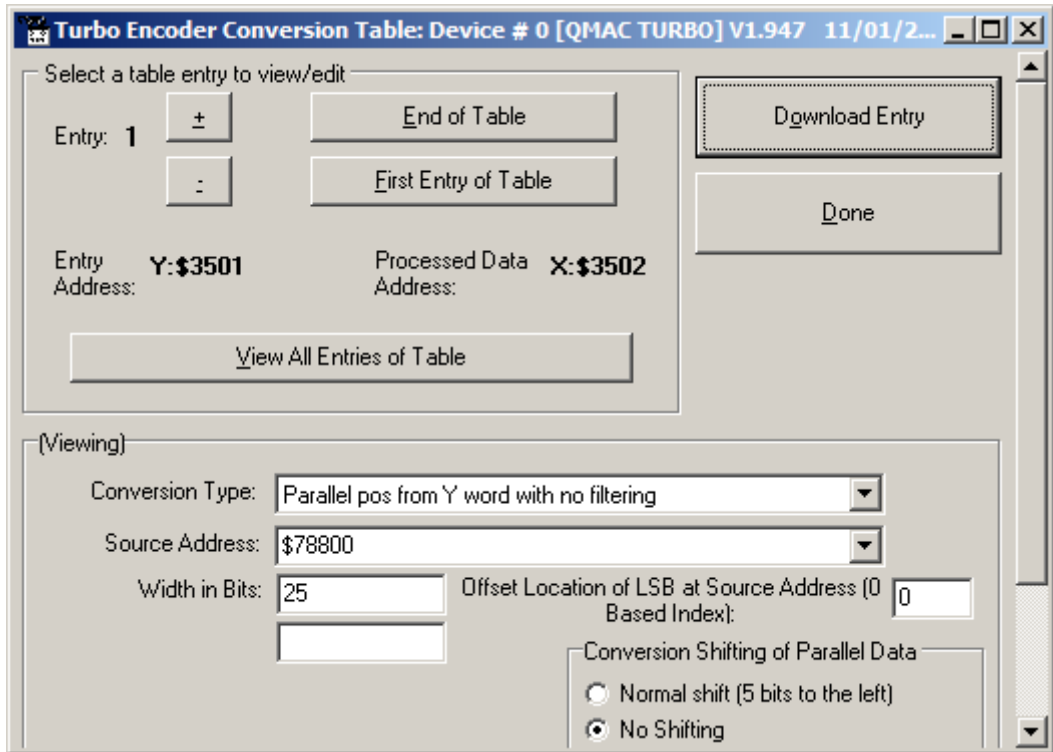
I5111=500*8388608/I10 While(I5111>0) EndWhile ; ½ sec delay
Disable PLC 1 ; Execute once on power-up or reset
Close
//=====//
```

Step3: Encoder Conversion Table Setup — for Position (Technique 2)

- Conversion Type: “Parallel pos from Y word with no filtering”
- Width in Bits is the single-turn resolution (25 bits in this example)
- Offset Location of LSB: leave at zero
- No shifting (recommended for higher resolution encoders)
- Source Address

Card Number	Channel Number	EnDat Data Register A
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

- Click on “Download Entry”:



Record the location of the processed data (X:\$3502 in this example). This is the position and velocity pointers’ address location. Equivalent Turbo PMAC Script:

```
I8000=$2F8800      ; Unfiltered parallel pos of location Y:$78800
I8001=$19000      ; Width (25 bits) and Offset (none), no shifting
                  ; Processed result at $3502
```

The Position and Velocity Pointer's address is the location of the processed data. Also, with higher resolution encoders, it is highly recommended to set the position and velocity loop scale factors to 1 to avoid saturation:

```
I100=1           ; Mtr #1 Active - remember to activate the channel to see feedback
I103=$3502      ; Mtr #1 position loop feedback address
I104=$3502      ; Mtr #1 velocity loop feedback address

I108=1          ; Mtr #1 position-loop scale factor
I109=1          ; Mtr #1 velocity-loop scale factor
```



Note

- At this point of the setup process, the user should be able to move the motor/encoder shaft by hand and see 'motor' counts in the position window.
- Some EnDat 2.2 Encoders do not support additional information (\$38 command mode in bits 24:17 of Channel Control Register). If this is the case, use \$07 instead in bits 24:17 of Channel Control Register (see Heidenhain command set table).

Heidenhain Command Set Table

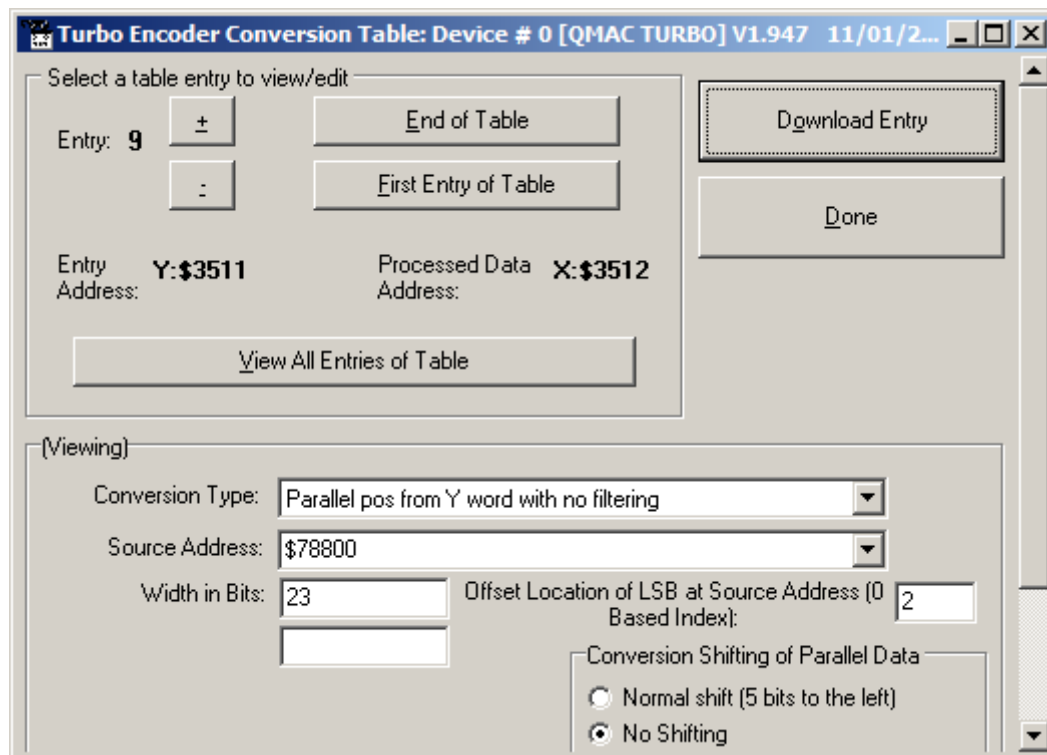
		Mode bit					
No.	Mode command	M2	M1	M0	(M2)	(M1)	(M0)
1	Encoder transmit position values	0	0	0	1	1	1
2	Selection of the memory area	0	0	1	1	1	0
3	Encoder receive parameters	0	1	1	1	0	0
4	Encoder transmit parameters	1	0	0	0	1	1
5	Encoder receive reset <sup>1)</sup>	1	0	1	0	1	0
6	Encoder transmit test values	0	1	0	1	0	1
7	Encoder receive test commands	1	1	0	0	0	1
8	Encoder transmit position value with additional information	1	1	1	0	0	0
9	Encoder transmit position value and receive selection of memory area <sup>2)</sup>	0	0	1	0	0	1
10	Encoder transmit position value and receive parameters <sup>2)</sup>	0	1	1	0	1	1
11	Encoder transmit position value and transmit parameters <sup>2)</sup>	1	0	0	1	0	0
12	Encoder transmit position value and receive error reset <sup>2)</sup>	1	0	1	1	0	1
13	Encoder transmit position value and receive test command <sup>2)</sup>	1	1	0	1	1	0
14	Encoder receive communication command <sup>3)</sup>	0	1	0	0	1	0

### Step 4: Encoder Conversion Table Setup — for Commutation (Technique 2)

Commutation with Turbo PMAC does not require high resolution data, so in order to avoid saturations with higher resolution EnDat encoders one must limit the commutation to 23 bits (upper). This will also alleviate possible quantization noise effects.

This entry should preferably be inserted at the end of the Encoder Conversion Table after all position/velocity encoders in the system have been processed or have been allocated to specific entries.

- Conversion Type: “Parallel pos from Y word with no filtering”
- Width in Bits is 23 (fixed)
- Offset Location of LSB = Single Turn data stream – 23 (i.e. 25-23=2)
- No shifting
- Source Address (same as position ECT)
- Click on “Download Entry”:



Record the location of the processed data (X:\$3512 in this example). This is the commutation position address. Equivalent Turbo PMAC Script:

```
I8016=$2F8800 ; Unfiltered parallel pos of location Y:$78800
I8017=$17002 ; Width (23 bits) and Offset (2 bits), no shifting
; Processed result at X:$3512
```

The commutation enable, and position address would then be:

```
I101=1 ; Mtr #1 Commutation enable, from X Register
I183=$3512 ; Mtr #1 Commutation Position Address
```

### Step 5: Absolute Power-On Position Read (Technique 2)

The absolute power-on position with Technique 2 cannot be performed with the automatic Turbo PMAC feature (Ixx10, Ixx95) because the data is unshifted, thus it must be constructed from the raw serial data registers.

The following example PLC reads and constructs the absolute position for channels 1 through 8 (channels 1–4 on the 1<sup>st</sup> ACC-84S, channels 4–8 on the 2<sup>nd</sup> ACC-84S). It is already configured for the user to input their encoder information (single turn and multi turn resolutions), and specify which channels should perform an absolute power-on read.

### Using the Absolute Position Read Example PLC

Under the “User Input” section:

1. Enter single turn (ChxSTRes) and multi turn (ChxMTRes) resolutions in bits for each encoder. For strictly absolute single turn encoders, set the multi-turn resolution (ChxMTRes) to zero.
2. In ChAbsSel, specify which channels are desired to perform an absolute position read. This value is in hexadecimal. A field value of 1 specifies that this channel is connected, 0 specifies that the channel is not connected and/or should not perform an absolute read. Examples:

Reading Absolute Position, Channels 1 through 4	Channel #	8	7	6	5	4	3	2	1	→ ChAbsSel=\$0F
	ChAbsSel (Binary)	0	0	0	0	1	1	1	1	
	ChAbsSel (Hex)	0				F				

Reading Absolute Position, Channels 1, 3, 5, 7	Channel #	8	7	6	5	4	3	2	1	→ ChAbsSel=\$55
	ChAbsSel (Binary)	0	1	0	1	0	1	0	1	
	ChAbsSel (Hex)	5				5				

```
//===== NOTES ABOUT THIS PLC EXAMPLE =====//
// This PLC example utilizes: - M6000 through M6035
//                               - P7000 through P7032
// Make sure that current and/or future configurations do not create conflicts with
// these parameters.
//=====//

M6000..6035->*      ; Self-referenced M-Variables
M6000..6035=0      ; Reset M-Variables at download
P7000..7032=0      ; Reset P-Variables at download

//===== USER INPUT =====//
#define Ch1STRes P7000      #define Ch1MTRes P7001
#define Ch2STRes P7002      #define Ch2MTRes P7003
#define Ch3STRes P7004      #define Ch3MTRes P7005
#define Ch4STRes P7006      #define Ch4MTRes P7007
#define Ch5STRes P7008      #define Ch5MTRes P7009
#define Ch6STRes P7010      #define Ch6MTRes P7011
#define Ch7STRes P7012      #define Ch7MTRes P7013
#define Ch8STRes P7014      #define Ch8MTRes P7015

Ch1STRes=25  Ch1MTRes=12    ; Ch1 Multi Turn and Single Turn Resolutions --User Input
Ch2STRes=25  Ch2MTRes=12    ; Ch2 Multi Turn and Single Turn Resolutions --User Input
Ch3STRes=25  Ch3MTRes=12    ; Ch3 Multi Turn and Single Turn Resolutions --User Input
Ch4STRes=25  Ch4MTRes=12    ; Ch4 Multi Turn and Single Turn Resolutions --User Input
Ch5STRes=25  Ch5MTRes=12    ; Ch5 Multi Turn and Single Turn Resolutions --User Input
Ch6STRes=25  Ch6MTRes=12    ; Ch6 Multi Turn and Single Turn Resolutions --User Input
Ch7STRes=25  Ch7MTRes=12    ; Ch7 Multi Turn and Single Turn Resolutions --User Input
Ch8STRes=25  Ch8MTRes=12    ; Ch8 Multi Turn and Single Turn Resolutions --User Input

#define ChAbsSel      P7016  ; Select Channels using absolute read (in Hexadecimal)
```

```

ChAbsSel=$FF ; Channels selected for absolute position read -User Input
//===== DEFINITIONS & SUBSTITUTIONS =====//
#define SerialRegA M6000 ; EnDat Serial Data Register A
#define SerialRegB M6001 ; EnDat Serial Data Register B
#define Two2STDec M6002 ; 2^STRes in decimal, for shifting operations
#define Two2STHex M6003 ; 2^STRes in Hexadecimal, for bitwise operations
#define Two2MTDec M6004 ; 2^MTRes in decimal, for shifting operations
#define Two2MTHex M6005 ; 2^MTRes in Hexadecimal, for bitwise operations
#define MTTemp1 M6006 ; Multi Turn Data temporary holding register 1
#define MTTemp2 M6007 ; Multi Turn Data temporary holding register 2
#define STTemp1 M6008 ; Single Turn Data temporary holding register 1
#define STTemp2 M6009 ; Single Turn Data temporary holding register 2
#define ChNoHex M6010 ; Channel Number in Hex
#define ChAbsCalc M6011 ; Abs. calc. flag (=1 true do read, =0 false do not do read)
#define LowerSTBits P7017 ; Lower Single Turn Bits, RegA
#define UpperSTBits P7018 ; Upper Single Turn Bits, RegB (where applicable)
#define LowerMTBits P7019 ; Lower Multi Turn Bits, RegA (where applicable)
#define UpperMTBits P7020 ; Upper Multi Turn Bits, RegB (where applicable)
#define STData P7021 ; Single Turn Data Word
#define MTData P7022 ; Multi Turn Data Word
#define NegTh P7023 ; Negative Threshold
#define Temp1 P7024 ; General Temporary holding register 1
#define Temp2 P7025 ; General Temporary holding register 2
#define SerialBase P7026 ; Indirect addressing index for serial registers, 6020
#define ChBase P7027 ; Indirect addressing index for channel No, 162
#define ChNo P7028 ; Current Channel Number
#define ResBase P7029 ; Indirect Addressing index for resolution input, 6000
#define STRes P7030 ; Single Turn Resolution of currently addressed channel
#define MTRes P7031 ; Multi Turn Resoluion of currently addressed channel
#define PsfBase P7032 ; Indirect addressing for position scale factor Ixx08, 108
// EnDat Serial Data Registers A (left column) and B (right column)
M6020->Y:$78800,0,24,U M6021->Y:$78801,0,24,U ; Channel 1
M6022->Y:$78804,0,24,U M6023->Y:$78805,0,24,U ; Channel 2
M6024->Y:$78808,0,24,U M6025->Y:$78809,0,24,U ; Channel 3
M6026->Y:$7880C,0,24,U M6027->Y:$7880D,0,24,U ; Channel 4
M6028->Y:$78820,0,24,U M6029->Y:$78821,0,24,U ; Channel 5
M6030->Y:$78824,0,24,U M6031->Y:$78825,0,24,U ; Channel 6
M6032->Y:$78828,0,24,U M6033->Y:$78829,0,24,U ; Channel 7
M6034->Y:$7882C,0,24,U M6035->Y:$7882D,0,24,U ; Channel 8

//===== PLC SCRIPT CODE =====//
Open PLC 1 Clear
ChNo=0
While(ChNo!>7) ; Loop for 8 Channels
  ChNo=ChNo+1
  ChNoHex=exp((ChNo-1)*ln(2))
  ChAbsCalc=(ChAbsSel&ChNoHex)/ChNoHex
  If (ChAbsCalc!=0) ; Absolute read on this channel?
    SerialBase=6020+(ChNo-1)*2
    SerialRegA=M(SerialBase)
    SerialRegB=M(SerialBase+1)
    ResBase=7000+(ChNo-1)*2
    STRes=P(ResBase)
    MTRes=P(ResBase+1)
    STData=0
    MTData=0
    If (STRes!>24) ; Single Turn Res<=24
      //=====SINGLE TURN DATA=====//
      Two2STDec=exp(STRes*ln(2))
      Two2STHex=Two2STDec-1
      STData=SerialRegA&Two2STHex
      //=====MULTI TURN DATA=====//
      Two2MTDec=exp(MTRes*ln(2))
      Two2MTHex=Two2MTDec-1
      If (MTRes=0)
        LowerMTBits=0
        UpperMTBits=0
        Two2MTDec=0
        Two2MTHex=0
        MTData=0
      Else

```

```

LowerMTBits=24-STRes
STTemp1=exp(LowerMTBits*ln(2))
STTemp2=0
UpperMTBits=MTRes-LowerMTBits
MTTemp1=exp(LowerMTBits*ln(2))
MTTemp2=exp(UpperMTBits*ln(2))
Temp1=(SerialRegA/Two2STDec)&(MTTemp1-1)
Temp2=SerialRegB&(MTTemp2-1)
MTData=Temp2*STTemp1+Temp1
EndIf
Else ; Single Turn Res>24
//=====SINGLE TURN DATA=====//
LowerSTBits=24
UpperSTBits=STRes-24
STTemp1=exp(UpperSTBits*ln(2))
STTemp2=STTemp1-1
Two2STDec=16777216*STTemp1
Two2STHex=Two2STDec-1
STData=(SerialRegB&STTemp2)*16777216+SerialRegA
//=====MULTI TURN DATA=====//
If (MTRes=0)
  LowerMTBits=0
  UpperMTBits=0
  Two2MTDec=0
  Two2MTHex=0
  MTData=0
Else
  Two2MTDec=exp(MTRes*ln(2))
  Two2MTHex=Two2MTDec-1
  LowerMTBits=0
  UpperMTBits=MTRes
  MTTemp1=exp(UpperMTBits*ln(2))
  MTTemp2=MTTemp1-1
  MTData=(SerialRegB/STTemp1)&MTTemp2
EndIf
EndIf
//=====ASSEMBLING ACTUAL POSITION=====//
ChBase=162+(ChNo-1)*100
PsfBase=108+(ChNo-1)*100
NegTh=Two2MTDec/2
If (MTData!>NegTh)
  M(ChBase)=(MTData*Two2STDec+STData)*I(PsfBase)
Else
  M(ChBase)=-((Two2MTHex-MTData)*Two2STDec+(Two2STDec-STData))*I(PsfBase)
EndIf
EndIf
EndW
ChNo=0
Disable PLC 1
Close

```

## **BiSS-C/BiSS-B Feedback Configuration**

---

The SSI option allows the Turbo Clipper to connect to up to eight channels of BiSS-C/BiSS-B type devices: four channels per ACC-84S, for a total of eight BiSS-C/BiSS-B devices when using two ACC-84S cards. Setting up the BiSS interface correctly requires the programming of two essential control registers:

- Global Control Registers
- Channel Control Registers

The resulting data is found in:

- BiSS-C/BiSS-B Data Registers



## Description of BiSS Registers

### Global Control Registers

The global register controls the clock settings and trigger settings of the feedback protocol and it is located at X-word of the (base address) + \$F. The following table shows address locations for the Global Control Register for different address settings:

Card Number	Base Address	Global Control Register
1 <sup>st</sup> ACC-84S	\$78800	X:\$7880F
2 <sup>nd</sup> ACC-84S	\$78820	X:\$7882F

The Global Control register is used to program the serial encoder interface clock frequency *SER\_Clock* and configure the serial encoder interface trigger clock. *SER\_Clock* is generated from a two-stage divider clocked at 100 MHz as follows:

$$SER\_Clock = \frac{100}{(M + 1) \times 2^N} MHz$$

M	N	Clock Frequency
49	0	2.0 MHz
99	0	1.0 MHz
99	1	500.0 KHz
99	2	250.0 KHz
...	...	

Default Settings: M=24, N=0 => 4 MHz transfer rates

There are two external trigger sources; phase and servo. Bits [9:8] in the Global Control register are used to select the source and active edge to use as the internal serial encoder trigger. The internal trigger is used by all four channels to initiate communication with the encoder. To compensate for external system delays, this trigger has a programmable 4-bit delay setting in 20 µsec increments.

23--16	15--12	11	10	9	8	7	6	5	4	3	2	1	0
M_Divisor	N_Divisor			Trigger Clock	Trigger Edge	Trigger Delay				Protocol Code			

Bit	Type	Default	Name	Description
[23:16]	R/W	0x18	M_Divisor	Intermediate clock frequency for <i>SER_Clock</i> . The intermediate clock is generated from a (M+1) divider clocked at 100 MHz.
[15:12]	R/W	0x0	N_Divisor	Final clock frequency for <i>SER_Clock</i> . The final clock is generated from a $2^N$ divider clocked by the intermediate clock.
[11:10]	R	00	Reserved	Reserved and always reads zero.
[09]	R/W	0	TriggerClock	Trigger clock select: 0= PhaseClock 1= ServoClock
[08]	R/W	0	TriggerEdge	Active clock edge select: 0= rising edge 1= falling edge
[07:04]	R/W	0x0	TriggerDelay	Trigger delay program relative to the active edge of the trigger clock. Units are in increments of 20 usec.
[03:00]	R	0xB	ProtocolCode	This read-only bit field is used to read the serial encoder interface protocol supported by the FPGA. A value of \$B defines this protocol as BiSS.

## Channel Control Registers

Each channel also has its Channel Specific Control Register which controls the functionality of each channel. The user uses this register to:

- Configure the number of position bits in the serial bit stream
- Enable or disable channels through the SENC\_MODE bit
- Enable or disable communication with the encoder using the trigger control bit

The Channel Specific Control Register appears at the following memory locations:

Card Number	Channel Number	Address
1 <sup>st</sup> ACC-84S	1	X:\$78800
1 <sup>st</sup> ACC-84S	2	X:\$78804
1 <sup>st</sup> ACC-84S	3	X:\$78808
1 <sup>st</sup> ACC-84S	4	X:\$7880C
2 <sup>nd</sup> ACC-84S	1	X:\$78820
2 <sup>nd</sup> ACC-84S	2	X:\$78824
2 <sup>nd</sup> ACC-84S	3	X:\$78828
2 <sup>nd</sup> ACC-84S	4	X:\$7882C

The diagram below describes the functionality of each bit in the Channel Specific Control Registers:

[23:16]	15	14	13	12	11	10	9	[8:6]	[5:0]
CRC Mask	=0 BiSS-C =1 BiSS-B	MCD	Trigger Mode	Trigger Enable		RxDataReady/ SENC_MODE		Status Bits	PositionBits/ Resolution

Bit	Type	Default	Name	Description
[23:16]	R/W	0x21	CRC_Mask	This bit field is used to define the CRC polynomial used for the position and status data. The 8-bit mask is to define any 4-bit to 8-bit CRC polynomial. The mask bits M[7:0] represent the coefficients [8:1], respectively, in the polynomial: $M_7x^8 + M_6x^7 + M_5x^6 + M_4x^5 + M_3x^4 + M_2x^3 + M_1x^2 + M_0x^1 + 1$ . The coefficient for $x_0$ is always 1 and therefore not included in the mask. An all zero mask indicates no CRC bits in the encoder data. Most common setting: (\$21) 00100001 = $x_6 + x_1 + 1$ (typically for Renishaw resolute) (\$09) 00001001 = $x_4 + x_1 + 1$
[15]	R/W	0	BiSS-B	This bit is used to select the BiSS-B protocol mode (=0 BiSS-C, =1 BiSS-B)
[14]	R/W	0	MCD	This bit is used to enable support for the optional MCD bit in BiSS-B mode. Setting this bit has no effect if the BiSS-B mode is not selected.
[13]	R/W	0	Trigger Mode	Trigger Mode to initiate communication: 0= continuous trigger 1= one-shot trigger All triggers occur at the defined Phase/Servo clock edge and delay setting.

[12]	R/W	0	Trigger Enable	0= disabled 1= enabled This bit must be set for either trigger mode. If the Trigger Mode bit is set for one-shot mode, the hardware will automatically clear this bit after the trigger occurs.
[11]		0	Reserved	Reserved and always reads zero.
[10]	R	0	RxData Ready	This read-only bit provides the received data status. It is low while the interface logic is communicating (busy) with the serial encoder. It is high when all the data has been received and processed.
	W	0	SENC_MODE	This write-only bit is used to enable the output drivers for the SENC_SDO, SENC_CLK, SENC_ENA pins for each respective channel. Writing a 0 to this bit disables the channel Writing a 1 to this bit enables the channel
[09]		0x0	Reserved	Reserved and always reads zero.
[08:06]	R/W	000	Status Bits	This bit field is used to define the number of status bits in the encoder data. The valid range of settings is 0 – 6 (000 – 110). The status bits are assumed to always follow after the position data and before the CRC.
[05:00]	W	0x00	Position Bits	This bit field is used to define the number of position data bits or encoder resolution: Range is 12 – 40 (001100 – 101000) The position bits are assumed to be in binary MSB-first format: \$12 for 18-bit   \$1A for 26-bit   \$20 for 32-bit

### BiSS Data Registers

The BiSS data is stored in 4 memory locations: BiSS Encoder Data Registers A, B, C, and D.

The BiSS Encoder Data Register A holds the 24 bits of the encoder position data. If the data exceeds the 24 available bits in this register, the upper overflow bits are LSB justified and readable in the BiSS Encoder Data B, which also holds the status and error bits. BiSS Encoder Data Registers C and D are reserved and always read zero.

BiSS Data Register B				BiSS Data Register A
23	22	[21:16]	[15:0]	[23:0]
TimeOut Err	CRC Err	Status Data	Position Data [39:24]	Position Data [23:0]

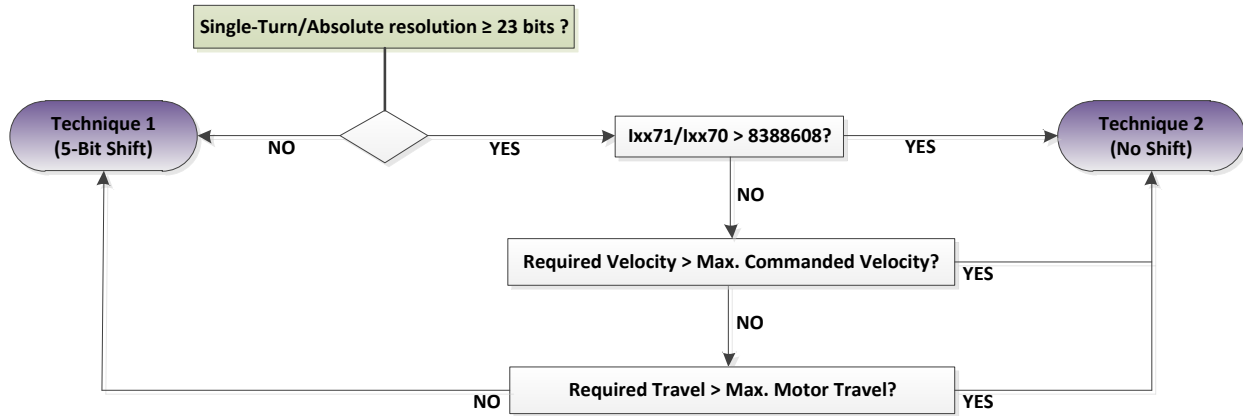
Card Number	Channel Number	BiSS Data Register A	BiSS Data Register B
1 <sup>st</sup> ACC-84S	1	Y:\$78800	Y:\$78801
1 <sup>st</sup> ACC-84S	2	Y:\$78804	Y:\$78805
1 <sup>st</sup> ACC-84S	3	Y:\$78808	Y:\$78809
1 <sup>st</sup> ACC-84S	4	Y:\$7880C	Y:\$7880D
2 <sup>nd</sup> ACC-84S	1	Y:\$78820	Y:\$78821
2 <sup>nd</sup> ACC-84S	2	Y:\$78824	Y:\$78825
2 <sup>nd</sup> ACC-84S	3	Y:\$78828	Y:\$78829
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C	Y:\$7882D

EnDat Registers C and D are listed here for future use and documentation purposes only. They do not pertain to the EnDat setup and always read zero.

<b>Card Number</b>	<b>Channel Number</b>	<b>BiSS Data Register C</b>	<b>BiSS Data Register D</b>
1 <sup>st</sup> ACC-84S	1	Y:\$78802	Y:\$78803
1 <sup>st</sup> ACC-84S	2	Y:\$78806	Y:\$78807
1 <sup>st</sup> ACC-84S	3	Y:\$7880A	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880E	Y:\$7880F
2 <sup>nd</sup> ACC-84S	1	Y:\$78822	Y:\$78823
2 <sup>nd</sup> ACC-84S	2	Y:\$78826	Y:\$78827
2 <sup>nd</sup> ACC-84S	3	Y:\$7882A	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$782E	Y:\$7882F

## Configuring BiSS Encoders

The [Using Absolute Serial Encoders with Turbo PMAC](#) section suggests two techniques for processing absolute serial data. The following chart shows the recommended method for using a given absolute serial encoder attached to a specific motor (number of poles):



If the application requirements are unknown or unrestrictive. It is always acceptable to setup absolute serial encoder using:

- Technique 1 for single-turn/absolute resolutions < 23 bits
- Technique 2 for single-turn/absolute resolutions ≥ 23 bits

With either technique, the following steps assure the proper setting of a motor with serial absolute encoders:

Step #	Description	Technique 1 (5-bit shift)	Technique 2 (no shift)
1	Global Control Register (per set of 4-axis)	√	√
2	Channel Control Register (per Channel)	√	√
3	Encoder Conversion Table (for Position)	Parallel Y-Word unfiltered normal 5-bit shift	Parallel Y-Word unfiltered no shifting
4	Encoder Conversion Table (for Commutation)	Directly from raw data	Parallel Y-Word unfiltered unshifted, limited to 23 bits
5	Absolute Power-On Position Read	Automatic using Ixx10, and Ixx95	Custom PLC (see example)
6	Absolute Power-On Phasing	Custom PLC (see example)	Automatic using Ixx75, Ixx81, and Ixx91

## Configuring BiSS Encoders Example Using Technique 1

A 18-bit resolute BiSS-C encoder is driving channel 1.

### Step 1: Global Control Register Setup (Technique 1)

	M_divisor								N_divisor						Trig Clk	Trig Edge	Trigger Delay				Protocol			
Bit#	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Binary	0	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
Hex \$	6				3				0				0				0				B			

Bit Assignment	Description	Resulting Global Control Register
M_divisor (99)=01100011	Serial Clock 1.0 MHz	\$63000B
Trigger Clock=0	Trigger on Phase Clock (recommended)	
Trigger Edge=0	Rising Edge (recommended)	
Trigger Delay=0	No delay	
Protocol Code (\$B)=1011	BiSS	

### Step 2: Channel Control Register Setup (Technique 1)

	CRC_Mask								BiSS Protocol	Trigger Mode	Trigger Enable		Senc		Status Bits	Resolution								
Bit#	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Binary	0	0	1	0	0	0	0	1	0	0	0	1	0	1	0	0	1	0	0	1	0	0	1	0
Hex \$	2				1				1				4				9						2	

Bit Assignment	Description	Resulting Channel Control Register
CRC_Mask=100001	Typically \$21 for BiSS	\$211492
Trigger Mode=0	Continuous Trigger (common)	
BiSS protocol =0	bits 14-15=0 for BiSS-C	
Trigger Enable=1	Must be set to 1 (common)	
Senc Mode=1	Enable Serial Driver (common)	
Resolution (18 bits)=10010 (\$1A)	Serial data stream	

## Control Registers Power-On PLC Example (Technique 1)

The Global and Channel Control words have to be executed once on power-up

```
//===== NOTES ABOUT THIS PLC EXAMPLE =====//
// This PLC example utilizes: - M5990 through M5991
//                               - Coordinate system 1 Timer 1 (I5111)
// Make sure that current and/or future configurations do not create conflicts with
// these parameters.
//=====//
M5990..5991->* ; Self-referenced M-Variables
M5990..5991=0 ; Reset at download
//===== GLOBAL CONTROL REGISTERS =====//
#define SSIGlobalCtrl1_4 M5990 ; Channels 1-4 BiSS global control register
SSIGlobalCtrl1_4->X:$7880F,0,24,U ; Channels 1-4 BiSS global control register address
//===== CHANNEL CONTROL REGISTERS =====//
#define Ch1SSICtrl M5991 ; Channel 1 BiSS control register
Ch1SSICtrl->X:$78800,0,24,U ; Channel 1 BiSS control register Address
//===== POWER-ON PLC EXAMPLE, GLOBAL & CHANNEL CONTROL REGISTERS =====//
Open PLC 1 Clear
SSIGlobalCtrl1_4=$63000B ; Trigger at Phase, 1 MHz serial Clock (M=99, N=0) -User Input
Ch1SSICtrl=$211492 ; Channel 1, BiSS-C protocol, 18-bit resolution -User Input

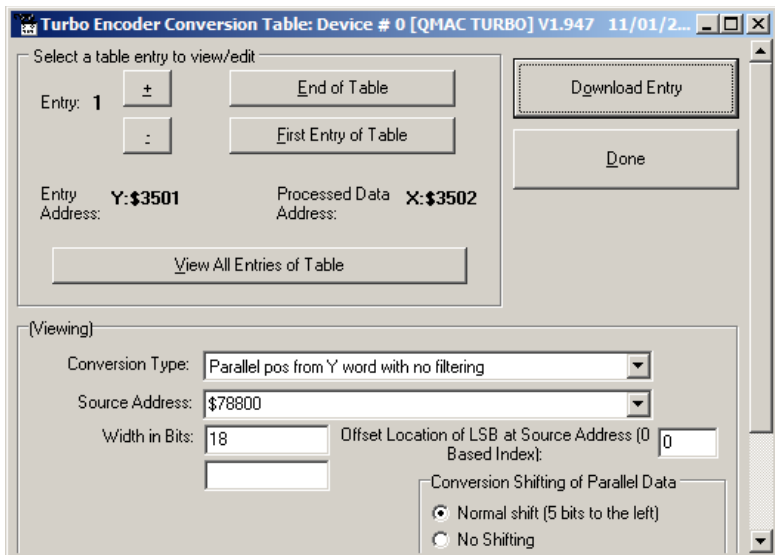
I5111=500*8388608/I10 While (I5111>0) EndWhile ; ½ sec delay
Disable PLC 1 ; Execute once on power-up or reset
Close
//=====//
```

Step 3: Encoder Conversion Table Setup - for Position (Technique 1)

- Conversion Type: “Parallel position from Y word with no filtering”
- Width in Bits is the single-turn/absolute resolution in bits (18 bits in this example)
- Offset Location of LSB: leave at zero
- Normal Shift (5 bits to the left)
- Source Address

Card Number	Channel Number	BiSS Data Register A
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

- Click on “Download Entry”:



Record the location of the processed data (X:\$3502 in this example). This is the position and velocity pointers’ address location. Equivalent Turbo PMAC Script:

```
I8000=$278800 ; Unfiltered parallel pos of location Y:$78800
I8001=$12000 ; Width (18 bits) and Offset (none), 5 bit left shift
; Processed result at $3502
```

The Position and Velocity Pointer’s address is the location of the processed data:

```
I100=1 ; Mtr #1 active - remember to activate the channel to see feedback
I103=$3502 ; Mtr #1 position loop feedback address
I104=$3502 ; Mtr #1 velocity loop feedback address
```



*Note*

At this point of the setup, the user should be able to move the motor/encoder shaft by hand and see “motor” counts in the position window.

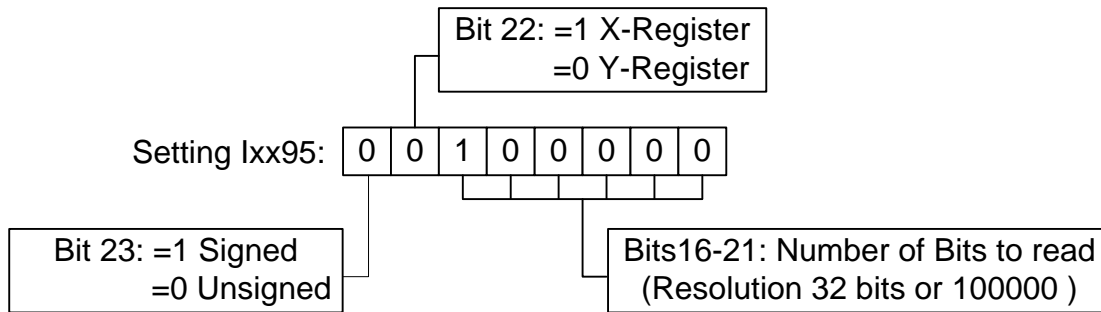
### Step 4: Absolute Power-On Position Read (Technique 1)

With Technique 1 (normal 5-bit shift), the absolute power-on read can be setup using the automatic feature in Turbo PMAC:

**Example:** Channel 1 driving a 32-bit BiSS (32-bit single turn, 0-bit multi-turn) encoder:

```
I180=2 ; Absolute power-on read enabled
I110=$78800 ; Absolute Servo power-on position address
I195=$200000 ; Parallel Read, 32 bits, Unsigned, from Y-Register -User Input
```

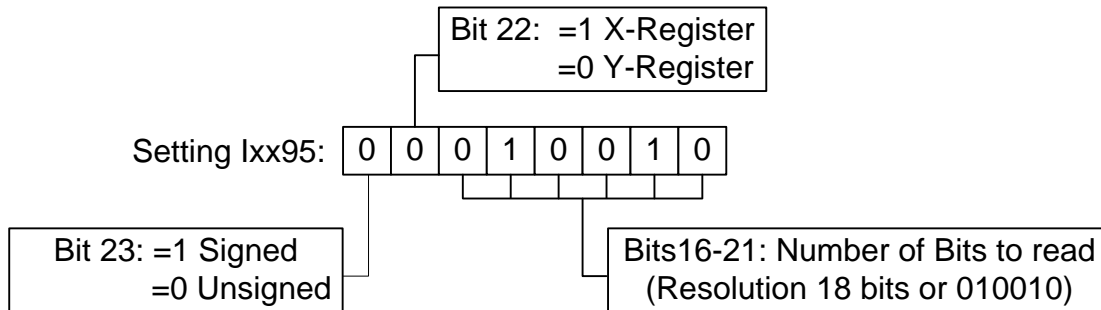
In this mode, PMAC reads and reports 32 bits: PMAC takes bits 23:0 from Y:\$078800, and then treats the low bits of Y:\$078801 as bits 31:24 of the position word. With this setting of Ixx80, the actual position is reported automatically on power-up. Otherwise, a #1\$\* command is necessary to read the absolute position.



**Example:** Channel 1 driving a 18-bit BiSS-C (18-bit single turn, 0 multi-turn) encoder:

```
I180=2 ; Absolute power-on read enabled
I110=$78800 ; Absolute Servo power-on position address
I195=$120000 ; Parallel Read, 18 bits, Unsigned, from Y-Register -User Input
```

In this mode, PMAC reads and reports 18 bits: the low 18 bits from Y:\$078800. With this setting of Ixx80, the actual position is reported automatically on power-up. Otherwise, a #1\$\* command is necessary to read the absolute position.



With purely absolute serial encoders (no multi-turn data), the user must configure the power-on position format for unsigned operation.



## Configuring BiSS Encoders Example Using Technique 2

A 26-bit resolute BiSS-C encoder is driving channel 1.

### Step 1: Global Control Register Setup (Technique 2)

Bit#	M_divisor								N_divisor				Trig Clk	Trig Edge	Trigger Delay				Protocol					
	23	22	21	20	19	18	17	16	15	14	13	12			11	10	9	8	7	6	5	4	3	2
Binary	0	1	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
Hex \$	6								0				0				B							

Bit Assignment	Description	Resulting Global Control Register
M_divisor (99)=01100011	Serial Clock 1.0 MHz	\$63000B
Trigger Clock=0	Trigger on Phase Clock (recommended)	
Trigger Edge=0	Rising Edge (recommended)	
Trigger Delay=0	No delay	
Protocol Code (\$B)=1011	BiSS-C	

### Step 2: Channel Control Register Setup (Technique 2)

Bit#	CRC_Mask								BiSS Protocol		Trigger Mode	Trigger Enable	Senc	Status Bits				Resolution						
	23	22	21	20	19	18	17	16	15	14	13	12		11	10	9	8	7	6	5	4	3	2	1
Binary	0	0	1	0	0	0	0	1	0	0	0	1	0	1	0	0	1	0	0	1	1	0	1	0
Hex \$	2								1		1		4				9				A			

Bit Assignment	Description	Resulting Channel Control Register
CRC_Mask=100001	Typically \$21 for BiSS	\$21149A
BiSS protocol =0	Bits 14-15=0 for BiSS-C	
Trigger Mode=0	Continuous Trigger (common)	
Trigger Enable=1	Must be set to 1 (common)	
SENC_MODE=1	Enable Serial Driver (common)	
Resolution (26 bits)=11010 (\$1A)	Serial data stream bits	

## Control Registers Power-On PLC Example (Technique 2)

The Global and Channel Control words have to be executed once on power-up

```
//===== NOTES ABOUT THIS PLC EXAMPLE =====//
// This PLC example utilizes: - M5990 through M5991
//                               - Coordinate system 1 Timer 1 (I5111)
// Make sure that current and/or future configurations do not create conflicts with
// these parameters.
//=====//
M5990..5991->* ; Self-referenced M-Variables
M5990..5991=0 ; Reset at download
//===== GLOBAL CONTROL REGISTERS =====//
#define SSIGlobalCtrl1_4 M5990 ; Channels 1-4 BiSS global control register
SSIGlobalCtrl1_4->X:$7880F,0,24,U ; Channels 1-4 BiSS global control register address
//===== CHANNEL CONTROL REGISTERS =====//
#define Ch1SSICtrl M5991 ; Channel 1 BiSS control register
Ch1SSICtrl->X:$78800,0,24,U ; Channel 1 BiSS control register Address

//===== POWER-ON PLC EXAMPLE, GLOBAL & CHANNEL CONTROL REGISTERS =====//
Open PLC 1 Clear
SSIGlobalCtrl1_4=$63000B ; Trigger at Phase, 1 MHz serial Clock (M=99, N=0) -User Input
Ch1SSICtrl=$21149A ; Channel 1, BiSS-C protocol, 26-bit resolution -User Input

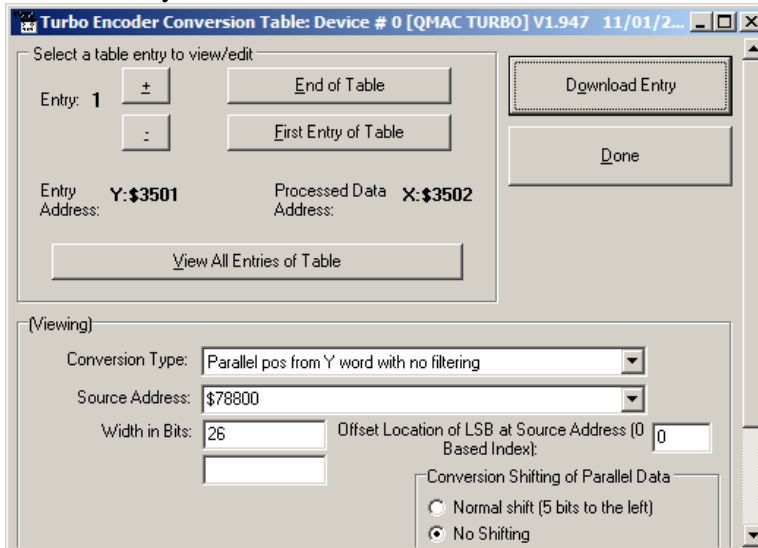
I5111=500*8388608/I10 While(I5111>0) EndWhile ; ½ sec delay
Disable PLC 1 ; Execute once on power-up or reset
Close
```

Step 3: Encoder Conversion Table Setup - for Position (Technique 2)

- Conversion Type: “Parallel pos from Y word with no filtering”
- Width in Bits is the single-turn/absolute resolution in bits (26 bits in this example)
- Offset Location of LSB: leave at zero
- No shifting (recommended for higher resolution encoders)
- Source Address

Card Number	Channel Number	Data Register A
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

- Click on “Download Entry”:



Record the location of the processed data (X:\$3502 in this example). This is the position and velocity pointers’ address location. Equivalent Turbo PMAC Script:

```
I8000=$2F8800 ; Unfiltered parallel pos of location Y:$78800
I8001=$01A000 ; Width (26 bits) and Offset (none), no shifting
; Processed result at $3502
```

The Position and Velocity Pointer’s address is the location of the processed data. Also, with higher resolution encoders, it is highly recommended to set the position- and velocity- loop scale factors to 1 to avoid saturation.

```
I100=1 ; Mtr#1 Active. Remember to activate the channel to see feedback
I103=$3502 ; Mtr#1 position loop feedback address
I104=$3502 ; Mtr#1 velocity loop feedback address
I108=1 ; Mtr#1 position-loop scale factor
I109=1 ; Mtr#1 velocity-loop scale factor
```



**Note**

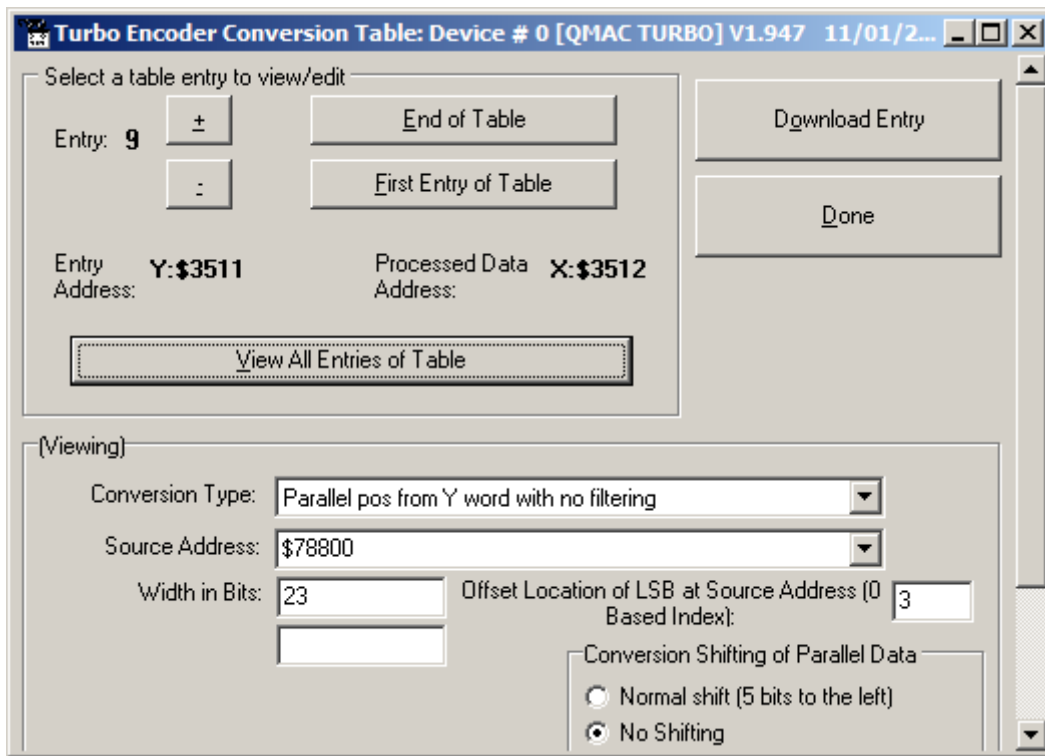
At this point of the setup, you should be able to move the motor/encoder shaft by hand and see ‘motor’ counts in the position window.

**Step 4: Encoder Conversion Table Setup — for Commutation (Technique 2)**

Commutation with Turbo PMAC does not require high resolution data, so in order to avoid saturations with higher resolution BiSS encoder one must limit the commutation to 23 bits (upper). This will also alleviate possible quantization noise effects.

This entry is preferably inserted at the end of the Encoder Conversion Table after all position/velocity encoders in the system have been processed or have been allocated to specific entries.

- Conversion Type: “Parallel position from Y word with no filtering”
- Width in Bits is 23 (fixed)
- Offset Location of LSB = Single Turn data stream– 23 (i.e. 26-23=3)
- No shifting
- Source Address (same as position ECT)
- Click on “Download Entry”:



Record the location of the processed data (X:\$3512 in this example). This is the commutation position address. Equivalent Turbo PMAC Script:

```
I8016=$2F8800 ; Unfiltered parallel pos of location Y:$78800
I8017=$017003 ; Width (23 bits) and Offset (3 bits), no shifting
                ; Processed result at X:$3512
```

The commutation enable, and position address would then be:

```
I101=1 ; Mtr #1 Commutation Enable, from X Register
I183=$3512 ; Mtr #1 Commutation Position Address
```

### Step 5: Absolute Power-On Position Read (Technique 2)

The absolute power-on position with Technique 2 cannot be performed with the automatic Turbo PMAC feature (Ixx10, Ixx95) because the data is unshifted, and thus it must be constructed from the raw serial data registers.

The following example PLC reads and constructs the absolute position for channels 1 through 8. It is already configured for the user to input their encoder information (single turn and multi turn resolutions), and specify which channels are to perform an absolute power-on read.

### Using the Absolute Position Read Example PLC

Under the “User Input” section:

1. Enter the single-turn (ChxSTRes) and multi-turn (ChxMTRes) resolutions in bits for each encoder. For strictly absolute single turn encoders, the user must set the multi turn resolution to zero.
2. In ChAbsSel, specify which channels the user desires to perform an absolute position read. This value is in hexadecimal. A field value of 1 specifies that this channel is connected, 0 specifies that the channel is not connected and/or should not perform an absolute read. Examples:

Channel #	8	7	6	5	4	3	2	1
ChAbsSel (Binary)	0	0	0	0	1	1	1	1
ChAbsSel (Hex)	0				F			

→ ChAbsSel=\$0F

Channel #	8	7	6	5	4	3	2	1
ChAbsSel (Binary)	0	1	0	1	0	1	0	1
ChAbsSel (Hex)	5				5			

→ ChAbsSel=\$55

```

//===== NOTES ABOUT THIS PLC EXAMPLE =====//
// This PLC example utilizes: - M6000 through M6035
//                               - P7000 through P7032
// Make sure that current and/or future configurations do not create conflicts with
// these parameters.
//=====//

M6000..6035->*      ; Self-referenced M-Variables
M6000..6035=0      ; Reset M-Variables at download
P7000..7032=0      ; Reset P-Variables at download

//===== USER INPUT =====//
#define Ch1STRes P7000      #define Ch1MTRes P7001
#define Ch2STRes P7002      #define Ch2MTRes P7003
#define Ch3STRes P7004      #define Ch3MTRes P7005
#define Ch4STRes P7006      #define Ch4MTRes P7007
#define Ch5STRes P7008      #define Ch5MTRes P7009
#define Ch6STRes P7010      #define Ch6MTRes P7011
#define Ch7STRes P7012      #define Ch7MTRes P7013
#define Ch8STRes P7014      #define Ch8MTRes P7015

Ch1STRes=25  Ch1MTRes=12    ; Ch1 Multi Turn and Single Turn Resolutions --User Input
Ch2STRes=25  Ch2MTRes=12    ; Ch2 Multi Turn and Single Turn Resolutions --User Input
Ch3STRes=25  Ch3MTRes=12    ; Ch3 Multi Turn and Single Turn Resolutions --User Input
Ch4STRes=25  Ch4MTRes=12    ; Ch4 Multi Turn and Single Turn Resolutions --User Input
Ch5STRes=25  Ch5MTRes=12    ; Ch5 Multi Turn and Single Turn Resolutions --User Input
Ch6STRes=25  Ch6MTRes=12    ; Ch6 Multi Turn and Single Turn Resolutions --User Input
Ch7STRes=25  Ch7MTRes=12    ; Ch7 Multi Turn and Single Turn Resolutions --User Input
Ch8STRes=25  Ch8MTRes=12    ; Ch8 Multi Turn and Single Turn Resolutions --User Input
#define ChAbsSel P7016      ; Select Channels using absolute read (in Hexadecimal)
ChAbsSel=$FF                ; Channels selected for absolute position read -User Input
    
```

```
//===== DEFINITIONS & SUBSTITUTIONS =====//
#define SerialRegA      M6000   ; BiSS Serial Data Register A
#define SerialRegB      M6001   ; BiSS Serial Data Register B
#define Two2STDec       M6002   ; 2^STRes in decimal, for shifting operations
#define Two2STHex       M6003   ; 2^STRes in Hexadecimal, for bitwise operations
#define Two2MTDec       M6004   ; 2^MTRes in decimal, for shifting operations
#define Two2MTHex       M6005   ; 2^MTRes in Hexadecimal, for bitwise operations
#define MTTemp1         M6006   ; Multi Turn Data temporary holding register 1
#define MTTemp2         M6007   ; Multi Turn Data temporary holding register 2
#define STTemp1         M6008   ; Single Turn Data temporary holding register 1
#define STTemp2         M6009   ; Single Turn Data temporary holding register 2
#define ChNoHex         M6010   ; Channel Number in Hex
#define ChAbsCalc       M6011   ; Abs. calc. flag (=1 true do read, =0 false do not do read)
#define LowerSTBits    P7017   ; Lower Single Turn Bits, RegA
#define UpperSTBits    P7018   ; Upper Single Turn Bits, RegB (where applicable)
#define LowerMTBits    P7019   ; Lower Multi Turn Bits, RegA (where applicable)
#define UpperMTBits    P7020   ; Upper Multi Turn Bits, RegB (where applicable)
#define STData         P7021   ; Single Turn Data Word
#define MTData         P7022   ; Multi Turn Data Word
#define NegTh          P7023   ; Negative Threshold
#define Temp1          P7024   ; General Temporary holding register 1
#define Temp2          P7025   ; General Temporary holding register 2
#define SerialBase     P7026   ; Indirect addressing index for serial registers, 6020
#define ChBase         P7027   ; Indirect addressing index for channel No, 162
#define ChNo           P7028   ; Current Channel Number
#define ResBase        P7029   ; Indirect Addressing index for resolution input, 6000
#define STRes          P7030   ; Single Turn Resolution of currently addressed channel
#define MTRes          P7031   ; Multi Turn Resolution of currently addressed channel
#define PsfBase        P7032   ; Indirect addressing for position scale factor Ixx08, 108
// BiSS Serial Data Registers A (left column) and B (right column)
M6020->Y:$78800,0,24,U      M6021->Y:$78801,0,24,U      ; Channel 1
M6022->Y:$78804,0,24,U      M6023->Y:$78805,0,24,U      ; Channel 2
M6024->Y:$78808,0,24,U      M6025->Y:$78809,0,24,U      ; Channel 3
M6026->Y:$7880C,0,24,U      M6027->Y:$7880D,0,24,U      ; Channel 4
M6028->Y:$78820,0,24,U      M6029->Y:$78821,0,24,U      ; Channel 5
M6030->Y:$78824,0,24,U      M6031->Y:$78825,0,24,U      ; Channel 6
M6032->Y:$78828,0,24,U      M6033->Y:$78829,0,24,U      ; Channel 7
M6034->Y:$7882C,0,24,U      M6035->Y:$7882D,0,24,U      ; Channel 8

//===== PLC SCRIPT CODE =====//
Open PLC 1 Clear
ChNo=0
While(ChNo!>7) ; Loop for 8 Channels
  ChNo=ChNo+1
  ChNoHex=exp((ChNo-1)*ln(2))
  ChAbsCalc=(ChAbsSel&ChNoHex)/ChNoHex
  If (ChAbsCalc!=0) ; Absolute read on this channel?
    SerialBase=6020+(ChNo-1)*2
    SerialRegA=M(SerialBase)
    SerialRegB=M(SerialBase+1)
    ResBase=7000+(ChNo-1)*2
    STRes=P(ResBase)
    MTRes=P(ResBase+1)
    STData=0
    MTData=0
  If (STRes!>24) ; Single Turn Res<=24
    //=====SINGLE TURN DATA=====//
    Two2STDec=exp(STRes*ln(2))
    Two2STHex=Two2STDec-1
    STData=SerialRegA&Two2STHex
    //=====MULTI TURN DATA=====//
    Two2MTDec=exp(MTRes*ln(2))
    Two2MTHex=Two2MTDec-1
    If (MTRes=0)
      LowerMTBits=0
      UpperMTBits=0
      Two2MTDec=0
      Two2MTHex=0
      MTData=0
    Else
      LowerMTBits=24-STRes
```

```
STTemp1=exp(LowerMTBits*ln(2))
STTemp2=0
UpperMTBits=MTRes-LowerMTBits
MTTemp1=exp(LowerMTBits*ln(2))
MTTemp2=exp(UpperMTBits*ln(2))
Temp1=(SerialRegA/Two2STDec) & (MTTemp1-1)
Temp2=SerialRegB&(MTTemp2-1)
MTData=Temp2*STTemp1+Temp1
EndIf
Else ; Single Turn Res>24
//=====SINGLE TURN DATA=====//
LowerSTBits=24
UpperSTBits=STRes-24
STTemp1=exp(UpperSTBits*ln(2))
STTemp2=STTemp1-1
Two2STDec=16777216*STTemp1
Two2STHex=Two2STDec-1
STData=(SerialRegB&STTemp2)*16777216+SerialRegA
//=====MULTI TURN DATA=====//
If (MTRes=0)
  LowerMTBits=0
  UpperMTBits=0
  Two2MTDec=0
  Two2MTHex=0
  MTData=0
Else
  Two2MTDec=exp(MTRes*ln(2))
  Two2MTHex=Two2MTDec-1
  LowerMTBits=0
  UpperMTBits=MTRes
  MTTemp1=exp(UpperMTBits*ln(2))
  MTTemp2=MTTemp1-1
  MTData=(SerialRegB/STTemp1) & MTTemp2
EndIf
EndIf
//=====ASSEMBLING ACTUAL POSITION=====//
ChBase=162+(ChNo-1)*100
PsfBase=108+(ChNo-1)*100
NegTh=Two2MTDec/2
If (MTData!>NegTh)
  M(ChBase)=(MTData*Two2STDec+STData)*I(PsfBase)
Else
  M(ChBase)=-((Two2MTHex-MTData)*Two2STDec)+(Two2STDec-STData)*I(PsfBase)
EndIf
EndIf
EndWhile
ChNo=0
Disable PLC 1
Close
```

## **Yaskawa Feedback Configuration**

---

This option allows the ACC-84S to connect to up to eight Yaskawa feedback devices. Setting up the Yaskawa Sigma interface correctly requires the programming of two essential control registers:

- Global Control Registers
- Channel Control Registers

The resulting data is found in:

- Yaskawa Data Registers



*Note*

All Yaskawa Sigma II & Sigma III protocols, whether incremental or absolute, regardless of resolution, are supported.

---

## Description of Yaskawa Registers

### Global Control Registers

X:\$788nF (default value: \$002003),

where n=0 for the 1<sup>st</sup> ACC-84S

n=2 for the 2<sup>nd</sup> ACC-84S

Card Number	Base Address	Global Control Register
1 <sup>st</sup> ACC-84S	\$78800	X:\$7880F
2 <sup>nd</sup> ACC-84S	\$78820	X:\$7882F



*Note*

With the Yaskawa option, the Global Control Register is set already at power-up and need not be changed.

The Global Control register is used to program the frequency for *SER\_Clock* and configure the serial encoder interface trigger clock. *SER\_Clock* is generated from a two-stage divider clocked at 100 MHz. The frequency for *SER\_Clock* is shown below:

$$SER\_Clock = \frac{100\text{ MHz}}{(M+1) \cdot 2^N},$$

where M and N are variables set in the M\_Divisor and N\_Divisor fields, respectively, of the Global Control register as shown in the following tables:

[23-16]								[15-12]				11	10	9	8	7	6	5	4	3	2	1	0
M Divisor								N Divisor				Reserved		Trig. Clock	Trig. Edge	Trigger Delay				Protocol Code			
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
0				0				0				0				0				6			



Bit	Type	Default	Name	Description
[23:16]	R/W	0x00	M_Divisor	Intermediate clock frequency for SER_Clock, the encoder clock. The intermediate clock is generated from a (M+1) divider clocked at 100 MHz.
[15:12]	R/W	0x0	N_Divisor	Final clock frequency for SER_Clock. The final clock is generated from a $2^N$ divider clocked by the intermediate clock.
[11:10]	R	00	Reserved	Reserved and always reads zero.
[09]	R/W	0	TriggerClock	Trigger clock select for initiating serial encoder communications: 0= PhaseClock 1= ServoClock
[08]	R/W	0	TriggerEdge	Active clock edge select for the trigger clock: 0= rising edge 1= falling edge
[07:04]	R/W	0x0	TriggerDelay	Trigger delay program relative to the active edge of the trigger clock. Units are in increments of 20 usec.
[03:00]	R		ProtocolCode	This read-only bit field is used to read the serial interface protocol supported by the FPGA. A value of \$5 defines this protocol as Yaskawa Sigma I. A value of \$6 defines this protocol as Yaskawa Sigma II, III, or V.

## Channel Control Registers

X:\$788n0, X:\$788n4, X:\$788n8, X:\$788nC, where: n=0 for the 1<sup>st</sup> ACC-84S  
n=2 for the 2<sup>nd</sup> ACC-84S

Card Number	Channel Number	Address
1 <sup>st</sup> ACC-84S	1	X:\$78800
1 <sup>st</sup> ACC-84S	2	X:\$78804
1 <sup>st</sup> ACC-84S	3	X:\$78808
1 <sup>st</sup> ACC-84S	4	X:\$7880C
2 <sup>nd</sup> ACC-84S	1	X:\$78820
2 <sup>nd</sup> ACC-84S	2	X:\$78824
2 <sup>nd</sup> ACC-84S	3	X:\$78828
2 <sup>nd</sup> ACC-84S	4	X:\$7882C

Bits 10, 12, and 13 are the only fields to be configured in the Channel Control Registers with the Yaskawa option. The rest is protocol information. The user must configure bits 10, 12, and 13 in a startup PLC to execute once on power up.

[23:14]	13	12	11	10	[9:0]
Reserved	Trig. Mode	Trig. Enable		RxData Ready/ SENC_MODE	Reserved

Bit	Type	Default	Name	Description
[23:14]	R	0x000	Reserved	Reserved and always reads zero.
[13]	R/W	0	Trigger Mode	Trigger Mode to initiate communication: 0= continuous trigger 1= one-shot trigger All triggers occur at the defined Phase/Servo clock edge and delay setting. See Global Control register for these settings.
[12]	R/W	0	Trigger Enable	Enable trigger for serial encoder communications: 0= disabled 1= enabled This bit must be set for either trigger mode. If the Trigger Mode bit is set for one-shot mode, the hardware will automatically clear this bit after the trigger occurs.
[11]	R/W	0	Reserved	Reserved and always reads zero.
[10]	R	0	RxData Ready	This read-only bit provides the received data status. It is low while the interface logic is communicating (busy) with the serial encoder. It is high when all the data has been received and processed.
	W	0	SENC_MODE	This write-only bit is used to enable the output drivers for the SENC_SDO, SENC_CLK, SENC_ENA pins for each respective channel. It also directly drives the respective SENC_MODE pin for each channel.
[09:00]	R	0x0	Reserved	Reserved and always reads zero.

### Yaskawa Feedback Channel Control Power-On Example PLC (Motors 1–8)

This code statement must be added to the PMAC's existing initialization PLC.

```
End Gat
Del Gat
Close

Open PLC 1 Clear
CMD"WX:$78800,$1400"
CMD"WX:$78084,$1400"
CMD"WX:$78808,$1400"
CMD"WX:$7880C,$1400"
CMD"WX:$78820,$1400"
CMD"WX:$78824,$1400"
CMD"WX:$78828,$1400"
CMD"WX:$7882C,$1400"
Disable PLC 1
Close
```

### Yaskawa Data Registers

Card Number	Channel Number	Yaskawa Data Register
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

## Configuring Yaskawa Encoders Yaskawa Sigma II 16-Bit Absolute Encoder

<b>Y:\$78801</b>		<b>Y:\$78800</b>		
[23-12]	[11-0]	[23-20]	[19-4]	[3:0]
Multi-Turn Position (16-bits)		Absolute Single Turn Data (16-bits)		

Card Number	Channel Number	Yaskawa Data Register
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

The ongoing servo and commutation position data is configured using a 2-line Entry in the Encoder Conversion Table (ECT). The first line represents a “Parallel pos from Y word with no filtering” entry (with a value of \$200000) from the corresponding Yaskawa data register/channel. The second line represents the width of the data to be read and bit location of the LSB of the data in the source word.

### Channel 1, Yaskawa Sigma II 16-bit Absolute Encoder Setup Example

The screenshot shows the 'Turbo Encoder Conversion Table: Device # 0 [UMAC TURBO] V1.947 11/01/2...' window. The 'Entry: 1' is selected. The 'Entry Address' is Y:\$3501 and the 'Processed Data Address' is X:\$3502. The 'View All Entries of Table' button is visible. In the '(Viewing)' section, the 'Conversion Type' is 'Parallel pos from Y word with no filtering', the 'Source Address' is '\$78800', the 'Width in Bits' is '32', and the 'Offset Location of LSB at Source Address (0 Based Index)' is '4'. The 'Conversion Shifting of Parallel Data' options are 'Normal shift (5 bits to the left)' (selected) and 'No Shifting'.

## Encoder Conversion Table Setup (Motors 1–8)

The ECT automatic entry is equivalent to:

I8000=\$278800	; Entry 1 Unfiltered parallel pos of location Y:\$78800
I8001=\$020004	; Width and Bias, total of 32-bits LSB starting at bit#4
I8002=\$278804	; Entry 2 Unfiltered parallel pos of location Y:\$78804
I8003=\$020004	; Width and Bias, total of 32-bits LSB starting at bit#4
I8004=\$278808	; Entry 3 Unfiltered parallel pos of location Y:\$78808
I8005=\$020004	; Width and Bias, total of 32-bits LSB starting at bit#4
I8006=\$27880C	; Entry 4 Unfiltered parallel pos of location Y:\$7880C
I8007=\$020004	; Width and Bias, total of 32-bits LSB starting at bit#4
I8008=\$278820	; Entry 5 Unfiltered parallel pos of location Y:\$78820
I8009=\$020004	; Width and Bias, total of 32-bits LSB starting at bit#4
I8010=\$278824	; Entry 6 Unfiltered parallel pos of location Y:\$78824
I8011=\$020004	; Width and Bias, total of 32-bits LSB starting at bit#4
I8012=\$278828	; Entry 7 Unfiltered parallel pos of location Y:\$78828
I8013=\$020004	; Width and Bias, total of 32-bits LSB starting at bit#4
I8014=\$27882C	; Entry 8 Unfiltered parallel pos of location Y:\$7882C
I8015=\$020004	; Width and Bias, total of 32-bits LSB starting at bit#4

## Position (Ixx03) and Velocity (Ixx04) Pointers

I103=\$3502	; Motor 1 Position feedback address, ECT processed data
I104=\$3502	; Motor 1 Velocity feedback address, ECT processed data
I203=\$3504	; Motor 2 Position feedback address, ECT processed data
I204=\$3504	; Motor 2 Velocity feedback address, ECT processed data
I303=\$3506	; Motor 3 Position feedback address, ECT processed data
I304=\$3506	; Motor 3 Velocity feedback address, ECT processed data
I403=\$3508	; Motor 4 Position feedback address, ECT processed data
I404=\$3508	; Motor 4 Velocity feedback address, ECT processed data
I503=\$350A	; Motor 5 Position feedback address, ECT processed data
I504=\$350A	; Motor 5 Velocity feedback address, ECT processed data
I603=\$350C	; Motor 6 Position feedback address, ECT processed data
I604=\$350C	; Motor 6 Velocity feedback address, ECT processed data
I703=\$350E	; Motor 7 Position feedback address, ECT processed data
I704=\$350E	; Motor 7 Velocity feedback address, ECT processed data
I803=\$3510	; Motor 8 Position feedback address, ECT processed data
I804=\$3510	; Motor 8 Velocity feedback address, ECT processed data

## Motor Activation

I100,8,100=1	; Motors 1-8 Activated
--------------	------------------------



*Note*

At this point of the setup process, the user should be able to move the motor/encoder shaft by hand and see encoder counts in the Position Window in PeWin32Pro2.

## Absolute Power-On Position Read (Yaskawa 16-bit) Channel 1 Example PLC, 16-bit Absolute Sigma II Encoder

```
End Gat
Del Gat
Close

#define STD0_15      M7000    ; Single-turn Data 0-15 (16-bits)
#define MTD0_3       M7001    ; Multi-Turn Data 0-3 (4-bits)
#define MTD4_15     M7002    ; Multi-Turn Data 4-15 (12-bits)
#define MTD0_15     M7003    ; Multi-Turn Data 0-15 (16-bits)

STD0_15->Y:$78800,4,16
MTD0_3->Y:$78800,20,4
MTD4_15->Y:$78801,0,12
MTD0_15->*

#define MtrlActPos   M162
MtrlActPos->D:$00008B ; #1 Actual position (1/[Ixx08*32] cts)

Open PLC 1 Clear
MTD0_15 = MTD4_15 * $10 + MTD0_3
If (MTD0_15 > $7FFF)
    MTD0_15 = (MTD0_15 ^ $FFFF + 1) * (-1)
    If (STD0_15 != 0)
        STD0_15 = (STD0_15 ^ $FFFF + 1) * (-1)
    EndIf
EndIf
MtrlActPos = ((MTD0_15 * $10000) + STD0_15) * I108 * 32
Disable PLC 1
Close
```

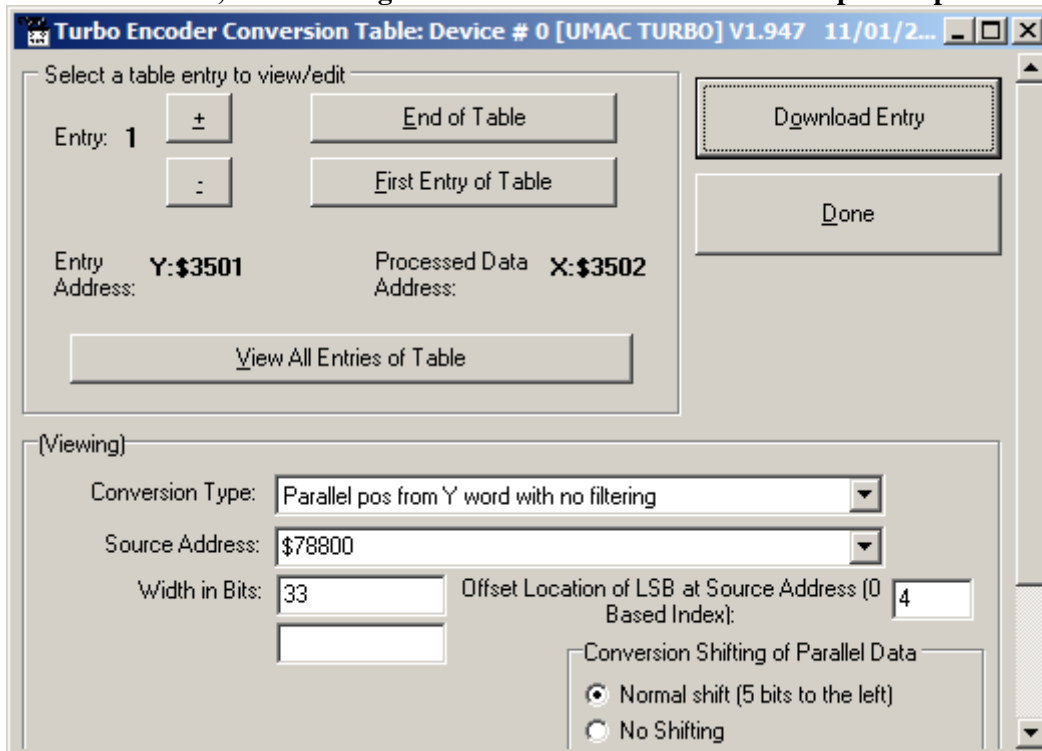
### Yaskawa Sigma II 17-Bit Absolute Encoder

<b>Y:\$78801</b>		<b>Y:\$78800</b>		
[23-13]	[12-0]	[23-21]	[20-4]	[3:0]
Multi-Turn Position (16-bits)		Absolute Single Turn Data (17-bits)		

Card Number	Channel Number	Yaskawa Data Register
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

The ongoing servo and commutation position data is setup using a 2-line Entry in the Encoder Conversion Table. The first line represents a “Parallel pos from Y word with no filtering” entry (with a value of \$200000) from the corresponding Yaskawa data register/channel. The second line represents the width of the data to be read and bit location of the LSB of the data in the source word.

#### Channel 1, Yaskawa Sigma II 17-bit Absolute Encoder Setup Example



## Encoder Conversion Table Setup (Motors 1–8)

The ECT automatic entry is equivalent to:

I8000=\$278800	; Entry 1 Unfiltered parallel pos of location Y:\$78800
I8001=\$021004	; Width and Bias, total of 33-bits LSB starting at bit#4
I8002=\$278804	; Entry 2 Unfiltered parallel pos of location Y:\$78804
I8003=\$021004	; Width and Bias, total of 33-bits LSB starting at bit#4
I8004=\$278808	; Entry 3 Unfiltered parallel pos of location Y:\$78808
I8005=\$021004	; Width and Bias, total of 33-bits LSB starting at bit#4
I8006=\$27880C	; Entry 4 Unfiltered parallel pos of location Y:\$7880C
I8007=\$021004	; Width and Bias, total of 33-bits LSB starting at bit#4
I8008=\$278820	; Entry 5 Unfiltered parallel pos of location Y:\$78820
I8009=\$021004	; Width and Bias, total of 33-bits LSB starting at bit#4
I8010=\$278824	; Entry 6 Unfiltered parallel pos of location Y:\$78824
I8011=\$021004	; Width and Bias, total of 33-bits LSB starting at bit#4
I8012=\$278828	; Entry 7 Unfiltered parallel pos of location Y:\$78828
I8013=\$021004	; Width and Bias, total of 33-bits LSB starting at bit#4
I8014=\$27882C	; Entry 8 Unfiltered parallel pos of location Y:\$7882C
I8015=\$021004	; Width and Bias, total of 33-bits LSB starting at bit#4

## Position (Ixx03) and Velocity (Ixx04) Pointers

I103=\$3502	; Motor 1 Position feedback address, ECT processed data
I104=\$3502	; Motor 1 Velocity feedback address, ECT processed data
I203=\$3504	; Motor 2 Position feedback address, ECT processed data
I204=\$3504	; Motor 2 Velocity feedback address, ECT processed data
I303=\$3506	; Motor 3 Position feedback address, ECT processed data
I304=\$3506	; Motor 3 Velocity feedback address, ECT processed data
I403=\$3508	; Motor 4 Position feedback address, ECT processed data
I404=\$3508	; Motor 4 Velocity feedback address, ECT processed data
I503=\$350A	; Motor 5 Position feedback address, ECT processed data
I504=\$350A	; Motor 5 Velocity feedback address, ECT processed data
I603=\$350C	; Motor 6 Position feedback address, ECT processed data
I604=\$350C	; Motor 6 Velocity feedback address, ECT processed data
I703=\$350E	; Motor 7 Position feedback address, ECT processed data
I704=\$350E	; Motor 7 Velocity feedback address, ECT processed data
I803=\$3510	; Motor 8 Position feedback address, ECT processed data
I804=\$3510	; Motor 8 Velocity feedback address, ECT processed data

## Motor Activation

I100,8,100=1	; Motors 1-8 Activated
--------------	------------------------



*Note*

At this point of the setup process, the user should be able to move the motor/encoder shaft by hand and see encoder counts in the Position Window in PeWin32Pro2.



## Absolute Power-On Position Read (Yaskawa 17-bit) Channel 1 Example PLC, 17-bit Absolute Sigma II Encoder

```
End Gat
Del Gat
Close

#define FirstWord      M7000 ; Yaskawa Data Register1, 1st word
#define SecondWord    M7001 ; Yaskawa Data Register1, 2nd word
#define STD0_16      M7002 ; Single-Turn Data 0-16 (17-bits)
#define MTD0_15      M7003 ; Multi-Turn Data 0-15 (16-bits)

FirstWord->Y:$78800,0,24
SecondWord->Y:$78801,0,4
STD0_16->*
MTD0_15->*

#define MtrlActPos    M162
MtrlActPos->D:$00008B ; #1 Actual position (1/[Ixx08*32] cts)

Open PLC 1 Clear
MTD0_15 = (SecondWord & $1FFF) * $8 + int(FirstWord / 2097152)
STD0_16 = int((FirstWord & $1FFFF0) / 16)
If(MTD0_15>$7FFF)
    MTD0_15 = (MTD0_15^$FFFF + 1)*(-1)

    If(STD0_16 !=0)
        STD0_16 = (STD0_16^$1FFFF + 1)*(-1)
    EndIf
EndIf
MtrlActPos = ((MTD0_15 * $20000) + STD0_16) * I108 * 32
Disable PLC 1
Close
```

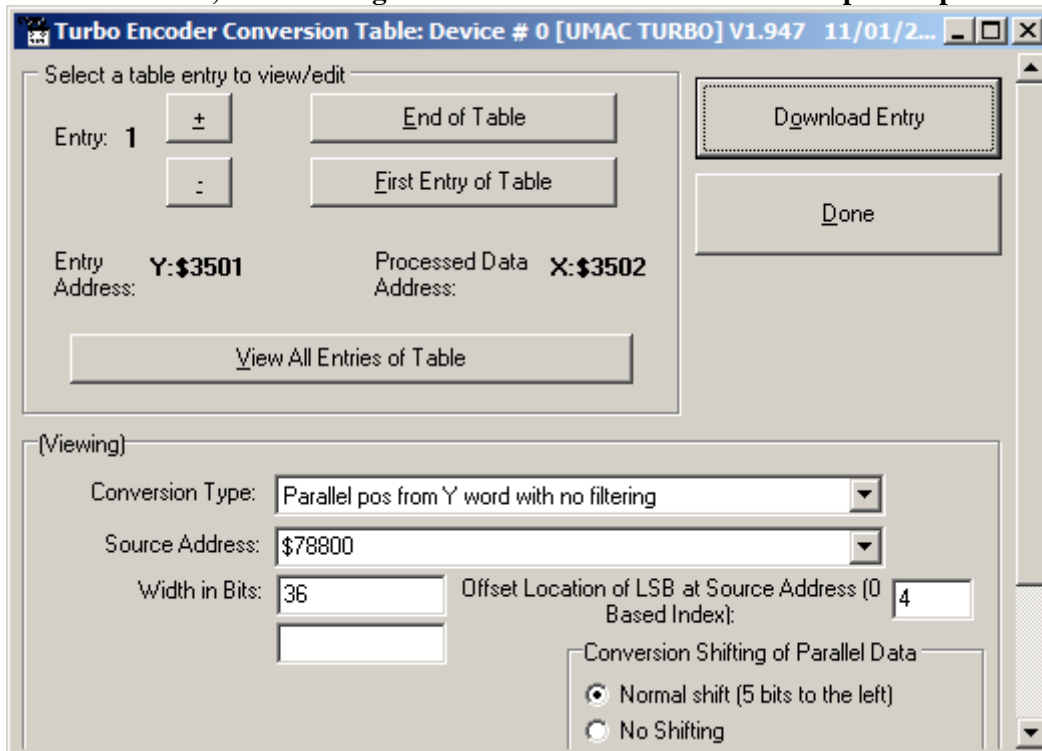
### Yaskawa Sigma III 20-Bit Absolute Encoder

<b>Y:\$78801</b>		<b>Y:\$78800</b>	
[23-16]	[15-0]	[23-4]	[3:0]
Multi-Turn Position (16-bits)		Absolute Single Turn Data (20-bits)	

Card Number	Channel Number	Yaskawa Data Register
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

The ongoing servo and commutation position data is setup using a 2-line Entry in the Encoder Conversion Table. The first line represents a “Parallel pos from Y word with no filtering” entry (with a value of \$200000) from the corresponding Yaskawa data register/channel. The second line represents the width of the data to be read and bit location of the LSB of the data in the source word.

#### Channel 1, Yaskawa Sigma III 20-bit Absolute Encoder Setup Example



## Encoder Conversion Table Setup (Motors 1–8)

The ECT automatic entry is equivalent to:

I8000=\$278800	; Entry 1 Unfiltered parallel pos of location Y:\$78800
I8001=\$024004	; Width and Bias, total of 36-bits LSB starting at bit#4
I8002=\$278804	; Entry 2 Unfiltered parallel pos of location Y:\$78804
I8003=\$024004	; Width and Bias, total of 36-bits LSB starting at bit#4
I8004=\$278808	; Entry 3 Unfiltered parallel pos of location Y:\$78808
I8005=\$024004	; Width and Bias, total of 36-bits LSB starting at bit#4
I8006=\$27880C	; Entry 4 Unfiltered parallel pos of location Y:\$7880C
I8007=\$024004	; Width and Bias, total of 36-bits LSB starting at bit#4
I8008=\$278820	; Entry 5 Unfiltered parallel pos of location Y:\$78820
I8009=\$024004	; Width and Bias, total of 36-bits LSB starting at bit#4
I8010=\$278824	; Entry 6 Unfiltered parallel pos of location Y:\$78824
I8011=\$024004	; Width and Bias, total of 36-bits LSB starting at bit#4
I8012=\$278828	; Entry 7 Unfiltered parallel pos of location Y:\$78828
I8013=\$024004	; Width and Bias, total of 36-bits LSB starting at bit#4
I8014=\$27882C	; Entry 8 Unfiltered parallel pos of location Y:\$7882C
I8015=\$024004	; Width and Bias, total of 36-bits LSB starting at bit#4

## Position (Ixx03) and Velocity (Ixx04) Pointers

I103=\$3502	; Motor 1 Position feedback address, ECT processed data
I104=\$3502	; Motor 1 Velocity feedback address, ECT processed data
I203=\$3504	; Motor 2 Position feedback address, ECT processed data
I204=\$3504	; Motor 2 Velocity feedback address, ECT processed data
I303=\$3506	; Motor 3 Position feedback address, ECT processed data
I304=\$3506	; Motor 3 Velocity feedback address, ECT processed data
I403=\$3508	; Motor 4 Position feedback address, ECT processed data
I404=\$3508	; Motor 4 Velocity feedback address, ECT processed data
I503=\$350A	; Motor 5 Position feedback address, ECT processed data
I504=\$350A	; Motor 5 Velocity feedback address, ECT processed data
I603=\$350C	; Motor 6 Position feedback address, ECT processed data
I604=\$350C	; Motor 6 Velocity feedback address, ECT processed data
I703=\$350E	; Motor 7 Position feedback address, ECT processed data
I704=\$350E	; Motor 7 Velocity feedback address, ECT processed data
I803=\$3510	; Motor 8 Position feedback address, ECT processed data
I804=\$3510	; Motor 8 Velocity feedback address, ECT processed data

## Motor Activation

I100,8,100=1	; Motors 1-8 Activated
--------------	------------------------



*Note*

At this point of the setup process, the user should be able to move the motor/encoder shaft by hand and see encoder counts in the Position Window in PeWin32Pro2.

## Absolute Power-On Position Read (Yaskawa 20-bit) Channel 1 Example PLC, 20-bit Absolute Sigma III Encoder

```
End Gat
Del Gat
Close

#define FirstWord      M1000 ; Yaskawa Data Register1, 1st word
#define SecondWord    M1001 ; Yaskawa Data Register1, 2nd word
#define STD0_19       M1002 ; Single-Turn Data 0-19 (20-bits)
#define MTD0_15       M1003 ; Multi-Turn Data 0-15 (16-bits)

FirstWord->Y:$78800,0,24
SecondWord->Y:$78801,0,4
STD0_19->*
MTD0_15->*

#define MtrlActPos    M162
MtrlActPos->D:$00008B ; #1 Actual position (1/[Ixx08*32] cts)

Open PLC 1 Clear
MTD0_15 = (SecondWord & $FFFF)
STD0_19 = int((FirstWord & $FFFFFF0) / 16)
If (MTD0_15>$7FFF)
    MTD0_15 = (MTD0_15^$FFFF + 1)*(-1)

    If (STD0_19 !=0)
        STD0_19 = (STD0_19^$FFFFFF + 1)*(-1)
    EndIf
EndIf
MtrlActPos = ((MTD0_15 * $100000)+ STD0_19) * I108 * 32
Disable PLC 1
Close
```

## Yaskawa Sigma II 13-Bit Incremental Encoder

Y:\$78801		Y:\$78800						
[23-11]	[10-0]	23	[22-11]	[10:4]	3	2	1	0
	Incremental Compensation (11-bits)		Incremental Position in Single Turn (13-bits)		U	V	W	Z

Card Number	Channel Number	Yaskawa Data Register
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

The on-going servo and commutation position data is setup using a 2-line Entry in the Encoder Conversion Table. The first line represents a “Parallel pos from Y word with no filtering” entry (with a value of \$200000) from the corresponding Yaskawa data register/channel. The second line represents the width of the data to be read and bit location of the LSB of the data in the source word.

### Channel 1, Yaskawa Sigma II 13-bit Incremental Encoder Setup Example

The screenshot shows the 'Turbo Encoder Conversion Table: Device # 0 [UMAC TURBO] V1.947 11/01/2...' window. It features a 'Select a table entry to view/edit' section with 'Entry: 1' selected. Below this, the 'Entry Address' is set to 'Y:\$3501' and the 'Processed Data Address' is 'X:\$3502'. A 'View All Entries of Table' button is present. The 'Viewing' section shows the 'Conversion Type' as 'Parallel pos from Y word with no filtering', 'Source Address' as '\$78800', 'Width in Bits' as '13', and 'Offset Location of LSB at Source Address (0 Based Index)' as '6'. Under 'Conversion Shifting of Parallel Data', 'Normal shift (5 bits to the left)' is selected.

## Encoder Conversion Table Setup (Motors 1–8)

The ECT automatic entry is equivalent to:

I8000=\$278800	; Entry 1 Unfiltered parallel pos of location Y:\$78800
I8001=\$00D006	; Width and Bias, total of 13-bits LSB starting at bit#6
I8002=\$278804	; Entry 2 Unfiltered parallel pos of location Y:\$78804
I8003=\$00D006	; Width and Bias, total of 13-bits LSB starting at bit#6
I8004=\$278808	; Entry 3 Unfiltered parallel pos of location Y:\$78808
I8005=\$00D006	; Width and Bias, total of 13-bits LSB starting at bit#6
I8006=\$27880C	; Entry 4 Unfiltered parallel pos of location Y:\$7880C
I8007=\$00D006	; Width and Bias, total of 13-bits LSB starting at bit#6
I8008=\$278820	; Entry 5 Unfiltered parallel pos of location Y:\$78820
I8009=\$00D006	; Width and Bias, total of 13-bits LSB starting at bit#6
I8010=\$278824	; Entry 6 Unfiltered parallel pos of location Y:\$78824
I8011=\$00D006	; Width and Bias, total of 13-bits LSB starting at bit#6
I8012=\$278828	; Entry 7 Unfiltered parallel pos of location Y:\$78828
I8013=\$00D006	; Width and Bias, total of 13-bits LSB starting at bit#6
I8014=\$27882C	; Entry 8 Unfiltered parallel pos of location Y:\$7882C
I8015=\$00D006	; Width and Bias, total of 13-bits LSB starting at bit#6

## Position (Ixx03) and Velocity (Ixx04) Pointers

I103=\$3502	; Motor 1 Position feedback address, ECT processed data
I104=\$3502	; Motor 1 Velocity feedback address, ECT processed data
I203=\$3504	; Motor 2 Position feedback address, ECT processed data
I204=\$3504	; Motor 2 Velocity feedback address, ECT processed data
I303=\$3506	; Motor 3 Position feedback address, ECT processed data
I304=\$3506	; Motor 3 Velocity feedback address, ECT processed data
I403=\$3508	; Motor 4 Position feedback address, ECT processed data
I404=\$3508	; Motor 4 Velocity feedback address, ECT processed data
I503=\$350A	; Motor 5 Position feedback address, ECT processed data
I504=\$350A	; Motor 5 Velocity feedback address, ECT processed data
I603=\$350C	; Motor 6 Position feedback address, ECT processed data
I604=\$350C	; Motor 6 Velocity feedback address, ECT processed data
I703=\$350E	; Motor 7 Position feedback address, ECT processed data
I704=\$350E	; Motor 7 Velocity feedback address, ECT processed data
I803=\$3510	; Motor 8 Position feedback address, ECT processed data
I804=\$3510	; Motor 8 Velocity feedback address, ECT processed data

## Motor Activation

I100,8,100=1	; Motors 1-8 Activated
--------------	------------------------



*Note*

At this point of the setup process, the user should be able to move the motor/encoder shaft by hand and see encoder counts in the Position Window in PeWin32Pro2.

## Yaskawa Sigma II 17-Bit Incremental Encoder

Y:\$78801		Y:\$78800						
[23-11]	[10-0]	23	[22-6]	[5:4]	3	2	1	0
	Incremental Compensation (11-bits)		Incremental Position in Single Turn (17-bits)		U	V	W	Z

Card Number	Channel Number	Yaskawa Data Register
1 <sup>st</sup> ACC-84S	1	Y:\$78800
1 <sup>st</sup> ACC-84S	2	Y:\$78804
1 <sup>st</sup> ACC-84S	3	Y:\$78808
1 <sup>st</sup> ACC-84S	4	Y:\$7880C
2 <sup>nd</sup> ACC-84S	1	Y:\$78820
2 <sup>nd</sup> ACC-84S	2	Y:\$78824
2 <sup>nd</sup> ACC-84S	3	Y:\$78828
2 <sup>nd</sup> ACC-84S	4	Y:\$7882C

The on-going servo and commutation position data is setup using a 2-line Entry in the Encoder Conversion Table. The first line represents a “Parallel pos from Y word with no filtering” entry (with a value of \$200000) from the corresponding Yaskawa data register/channel. The second line represents the width of the data to be read and bit location of the LSB of the data in the source word.

### Channel 1, Yaskawa Sigma II 17-bit Incremental Encoder Setup Example

The screenshot shows the 'Turbo Encoder Conversion Table' software interface. The main window displays the following configuration for Channel 1:

- Entry:** 1
- Entry Address:** Y:\$3501
- Processed Data Address:** X:\$3502
- Conversion Type:** Parallel pos from Y word with no filtering
- Source Address:** \$78800
- Width in Bits:** 17
- Offset Location of LSB at Source Address (0 Based Index):** 6
- Conversion Shifting of Parallel Data:** Normal shift (5 bits to the left)

## Encoder Conversion Table Setup (Motors 1–8)

The ECT automatic entry is equivalent to:

I8000=\$278800	; Entry 1 Unfiltered parallel pos of location Y:\$78800
I8001=\$011006	; Width and Bias, total of 17-bits LSB starting at bit#6
I8002=\$278804	; Entry 2 Unfiltered parallel pos of location Y:\$78804
I8003=\$011006	; Width and Bias, total of 17-bits LSB starting at bit#6
I8004=\$278808	; Entry 3 Unfiltered parallel pos of location Y:\$78808
I8005=\$011006	; Width and Bias, total of 17-bits LSB starting at bit#6
I8006=\$27880C	; Entry 4 Unfiltered parallel pos of location Y:\$7880C
I8007=\$011006	; Width and Bias, total of 17-bits LSB starting at bit#6
I8008=\$278820	; Entry 5 Unfiltered parallel pos of location Y:\$78820
I8009=\$011006	; Width and Bias, total of 17-bits LSB starting at bit#6
I8010=\$278824	; Entry 6 Unfiltered parallel pos of location Y:\$78824
I8011=\$011006	; Width and Bias, total of 17-bits LSB starting at bit#6
I8012=\$278828	; Entry 7 Unfiltered parallel pos of location Y:\$78828
I8013=\$011006	; Width and Bias, total of 17-bits LSB starting at bit#6
I8014=\$27882C	; Entry 8 Unfiltered parallel pos of location Y:\$7882C
I8015=\$011006	; Width and Bias, total of 17-bits LSB starting at bit#6

## Position (Ixx03) and Velocity (Ixx04) Pointers

I103=\$3502	; Motor 1 Position feedback address, ECT processed data
I104=\$3502	; Motor 1 Velocity feedback address, ECT processed data
I203=\$3504	; Motor 2 Position feedback address, ECT processed data
I204=\$3504	; Motor 2 Velocity feedback address, ECT processed data
I303=\$3506	; Motor 3 Position feedback address, ECT processed data
I304=\$3506	; Motor 3 Velocity feedback address, ECT processed data
I403=\$3508	; Motor 4 Position feedback address, ECT processed data
I404=\$3508	; Motor 4 Velocity feedback address, ECT processed data
I503=\$350A	; Motor 5 Position feedback address, ECT processed data
I504=\$350A	; Motor 5 Velocity feedback address, ECT processed data
I603=\$350C	; Motor 6 Position feedback address, ECT processed data
I604=\$350C	; Motor 6 Velocity feedback address, ECT processed data
I703=\$350E	; Motor 7 Position feedback address, ECT processed data
I704=\$350E	; Motor 7 Velocity feedback address, ECT processed data
I803=\$3510	; Motor 8 Position feedback address, ECT processed data
I804=\$3510	; Motor 8 Velocity feedback address, ECT processed data

## Motor Activation

I100,8,100=1	; Motors 1-8 Activated
--------------	------------------------



*Note*

At this point of the setup process, the user should be able to move the motor/encoder shaft by hand and see encoder counts in the Position Window in PeWin32Pro2.



## Yaskawa Incremental Encoder Alarm Codes

Yaskawa Incremental Encoder Alarm Codes		
Card Number	Channel Number	Yaskawa Data Register
1 <sup>st</sup> ACC-84S	1	Y:\$78802,8,8
1 <sup>st</sup> ACC-84S	2	Y:\$78806,8,8
1 <sup>st</sup> ACC-84S	3	Y:\$7880A,8,8
1 <sup>st</sup> ACC-84S	4	Y:\$7880E,8,8
2 <sup>nd</sup> ACC-84S	1	Y:\$78822,8,8
2 <sup>nd</sup> ACC-84S	2	Y:\$78826,8,8
2 <sup>nd</sup> ACC-84S	3	Y:\$7882A,8,8
2 <sup>nd</sup> ACC-84S	4	Y:\$7882E,8,8

Bit#	Error Name	Type	Alarm Type	Clear Action	Notes
8	Fixed at "1"	-	-	-	
9	Encoder Error	Alarm	Session Flag	Power cycle	Encoder Error
10	Fixed at "0"	-	-	-	
11	Position Error	Alarm	Session Flag	Power cycle	Possible error in position or Hall sensor
12	Fixed at "0"	-	-	-	
13	Fixed at "0"	-	-	-	
14	Origin not passed flag	Warning	-	-	The origin has not been passed in this session yet
15	Fixed at "0"				Set at zero

## Homing with Yaskawa Incremental Encoders

Hardware capture is not available with serial data encoders, so software capture (Ixx97=1) is required. Setting Ixx97 to 1 tells Turbo PMAC to use the register whose address is specified by Ixx03 for the trigger position. The disadvantage is that the software capture can have up to 1 background cycle delay (typically 2–3 msec), which limits the accuracy of the capture. To alleviate homing inaccuracies with serial encoders, it is recommended to perform home search moves at low speeds.

Homing to a flag (i.e. Home, Overtravel Limit, and User) is done using the traditional capture parameters I7mn2, and I7mn3. Remember to (temporarily) disable the end of travel limit use (bit#17 of Ixx24) when homing to one of the hardware limit flags, and then reenable it when homing is finished.

### Example: Homing channel 1 to the negative limit (high true)

```
I124=I124|$20001      ; Flag Mode, Disable hardware over travel limits
I197=1                ; Channel 1 position capture, software
I7012=2               ; Channel 1 capture control, capture on flag high
I7012=2               ; Channel 1 capture flag select, minus or negative end limit
```

Homing to the index pulse, normally performed after referencing to a hardware flag, is an internal function of the Yaskawa encoder. Bit 14 of the alarm code indicates whether the index has been detected since last power-up. The motor should be jogged until bit 14 is low, the encoder will then place the “incremental compensation” value in the lower 11 bits of the second data word. Subtracting the “incremental compensation” from the “incremental position” results into the true position of the index.

### Motor 1 Index Detection Example PLC:

```
#define FirstWord      M7025
#define SecondWord     M7026
#define OriginNotPassed M7027

FirstWord->Y:$78800,0,24
SecondWord->Y:$78801,0,24
OriginNotPassed->Y:$78802,14

#define Mtr1ActPos      M162 ; Suggested M-Variable Definition, Motor 1 Actual Position
Mtr1ActPos->D:$00008B      ; #1 Actual position (1/[Ixx08*32] cts)

Open PLC 1 Clear
If(OriginNotPassed = 1)
  CMD"#1j+"                ; Jog in positive direction looking for index
  While(OriginNotPassed = 1); wait until index is detected
  EndWhile
  CMD"#1k"                 ; Kill Motor
EndIf
While(SecondWord & $8FF = 0) ; Incremental Compensation takes up to 2 msec to execute
EndWhile
Mtr1ActPos = int(((FirstWord & $8FFFC0) / $40)-((SecondWord & $8FF) * $40))* I108 * 32
Disable PLC 1
Close
```

## Absolute Power-On Phasing (SSI, EnDat, BiSS)

With absolute serial encoders, the absolute serial data can be used to establish a phase reference position on power-up without moving the motor.

For parallel processed data such as with absolute serial encoders, Turbo PMAC's automatic power-on phasing feature (Ixx75, Ixx81, and Ixx91) expects the least significant bit of the data stream to be at bit 0. This is possible with Technique 2 which processes the data without shifting in the Encoder Conversion Table however it is unsuitable for Technique 1 which processes the data with the normal 5-bit shift. Therefore, a custom PLC is suggested for performing an absolute power-on phasing with Technique 1.



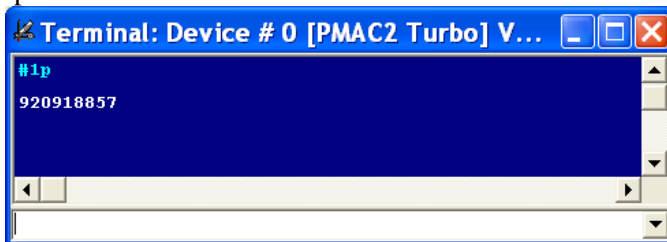
**Note**

Prior to implementing a power-on phasing routine, the user should verify that the motor can be phased manually, is able to execute open-loop moves successfully (output and encoder direction matching), and can possibly perform jog commands (requires PID tuning).

### Technique 1 (5-bit shift)

A one-time simple test (per installation) is performed, preferably on an unloaded motor, to find the motor phase position offset:

1. Issue a #1\$\* to ensure that the absolute position is correct and up to date
2. Record the values of Ixx29, and Ixx79 to restore them at the end of test (if applicable)
3. Set Ixx29=0, and write a positive value to Ixx79 then issue a #n00 (where n is the motor number). 500 is a conservative initial value for Ixx79. Adjust appropriately (most likely to increase) to force the motor to lock tightly onto a phase
4. Wait for the motor to settle
5. Record the absolute position from the position window or issue a #nP to return the motor position in the terminal window



6. Issue a #nK to kill the motor
7. Restore Ixx29, and Ixx79 to their original values (if applicable)
8. Enter the recorded value in the corresponding motor/channel definition in the example PLC below

The following example PLC computes and corrects for the phase position register (Mxx71) for channels 1 through 8. It is preconfigured for the user to input his or her encoder/motor information, and also to specify which channels will perform an absolute power-on phasing.

### Using the Absolute Power-On Phasing Example PLC

Under User Input section:

1. Enter motor scale factor (MtrxSF) for each corresponding motor.  
 For rotary encoders, this is the number of counts per revolution =  $2^{\text{Single-Turn Resolution}}$   
 For linear encoders, this is the number of counts per user units (i.e. mm) =  $1/(\text{Encoder Resolution})$

- In ChPhaseSel, specify which channels are desired to perform an absolute power-on phasing. This value is in hexadecimal. A value of 1 in the corresponding field specifies that this channel is connected, 0 specifies that it is not connected and should not perform phasing. Examples:

Absolute Power-On Phasing, Channels 1 through 4	Channel #	8	7	6	5	4	3	2	1	→ ChPhaseSel =\$0F
	ChPhaseSel (Binary)	0	0	0	0	1	1	1	1	
	ChPhaseSel (Hex)	0				F				

Absolute Power-On Phasing, Channels 1,3,5,7	Channel #	8	7	6	5	4	3	2	1	→ ChPhaseSel =\$55
	ChPhaseSel (Binary)	0	1	0	1	0	1	0	1	
	ChPhaseSel (Hex)	5				5				

```
//===== NOTES ABOUT THIS PLC EXAMPLE =====//
// This PLC example utilizes: - P7050 through P7079
//                               - Suggested M-Variables (make sure they are downloaded)
// Make sure that current and/or future configurations do not create conflicts with
// these parameters.
//=====//

P7050..7079=0 ; Reset P-Variables at download

//===== USER INPUT =====//
#define Mtr1SF P7050 #define Mtr5SF P7054 ; Motors scale factor
#define Mtr2SF P7051 #define Mtr6SF P7055 ; cts/rev for rotary encoders
#define Mtr3SF P7052 #define Mtr7SF P7056 ; cts/user units (i.e. mm, inches) for linear
#define Mtr4SF P7053 #define Mtr8SF P7057 ;
Mtr1SF=0 Mtr5SF=0 ; --User Input
Mtr2SF=0 Mtr6SF=0 ; --User Input
Mtr3SF=0 Mtr7SF=0 ; --User Input
Mtr4SF=0 Mtr8SF=0 ; --User Input

#define Mtr1PhaseTest P7058 #define Mtr5PhaseTest P7062 ; Phase force test values
#define Mtr2PhaseTest P7059 #define Mtr6PhaseTest P7063 ;
#define Mtr3PhaseTest P7060 #define Mtr7PhaseTest P7064 ;
#define Mtr4PhaseTest P7061 #define Mtr8PhaseTest P7065 ;
Mtr1PhaseTest=0 Mtr5PhaseTest=0 ; --User Input
Mtr2PhaseTest=0 Mtr6PhaseTest=0 ; --User Input
Mtr3PhaseTest=0 Mtr7PhaseTest=0 ; --User Input
Mtr4PhaseTest=0 Mtr8PhaseTest=0 ; --User Input

#define ChPhaseSel P7066 ; Select channels to perform power-on phasing (in Hexadecimal)
ChPhaseSel=$0 ; Channels selected for power-on phasing --User Input

//===== DEFINITIONS & SUBSTITUTIONS =====//
#define ChNo P7067 ; Present addressed channel
#define PhaseOffset P7068 ; Holding register for computing phase position offset
#define ActPos P7069 ; Indirect addressing index for actual position, 162
#define PresPhasePos P7070 ; Holding register for computing present phase position
#define Ixx70 P7071 ; Indirect addressing index for No of commutation cycles, 170
#define Ixx71 P7072 ; Indirect addressing index for commutation cycle size, 171
#define Mxx71 P7073 ; Indirect addressing index for phase position register, 171
#define PhaseErrBit P7074 ; Indirect addressing index for phasing search error bit, 148
#define PhaseTest P7075 ; Indirect addressing index for force phase test values, 7058
#define MtrSF P7076 ; Indirect addressing index for motor scale factor, 7050
#define ChNoHex P7077 ; Channel number in hex
#define Ixx08 P7078 ; Indirect addressing index for position scale factor, 108
#define ChPhaseTrue P7079 ; Present channel power-on phasing flag, =1 true =0 false

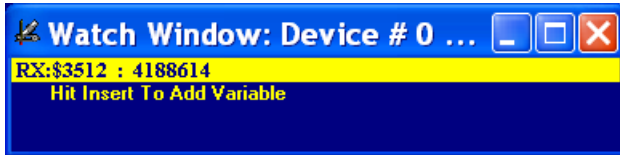
//===== PLC SCRIPT CODE =====//
Open PLC 1 Clear
ChNo=0 ; Reset channel number
While (ChNo!>7) ; Loop for 8 channels
ChNo=ChNo+1
ChNoHex=exp((ChNo-1)*ln(2))
ChPhaseTrue=(ChPhaseSel&ChNoHex)/ChNoHex
If (ChPhaseTrue!=0) ; Absolute read on this channel?
```

```
MtrSF=7050+(ChNo-1)*1
PhaseTest=7058+(ChNo-1)*1
Ixx70=170+(ChNo-1)*100
Ixx71=171+(ChNo-1)*100
ActPos=162+(ChNo-1)*100
Ixx08=108+(ChNo-1)*100
Mxx71=171+(ChNo-1)*100
PhaseErrBit=148+(ChNo-1)*100
I5111= 100*8388608/I10 While(I5111>0) EndW
// Compute position offset from user force phase test input
PhaseOffset=P(PhaseTest)%P(MtrSF)
PhaseOffset=PhaseOffset*I(Ixx70)
PhaseOffset=PhaseOffset*I(Ixx71)
I5111= 100*8388608/I10 While(I5111>0) EndW
// Compute present phase position
PresPhasePos=M(ActPos)/(I(Ixx08)*32)
PresPhasePos=PresPhasePos%P(MtrSF)
PresPhasePos=PresPhasePos*I(Ixx70)
PresPhasePos=PresPhasePos*I(Ixx71)
I5111= 100*8388608/I10 While(I5111>0) EndW
// Correct for Mxx71 to apply power-on phasing, and clear phase error search bit
M(Mxx71)=(PresPhasePos-PhaseOffset)%I(Ixx71)
M(PhaseErrBit)=0
I5111= 100*8388608/I10 While(I5111>0) EndW
EndIf
EndWhile
Disable PLC 1
Close
//=====//
```

**Technique 2 (No shift):**

A one-time simple test (per installation) is performed, preferably on an unloaded motor, to find the motor phase position offset:

1. Make sure the Absolute Position Read PLC has run once reporting the correct position
2. Record the values of Ixx29, and Ixx79 to restore them at the end of test (if applicable)
3. Set Ixx29=0, and write a positive value to Ixx79 then issue a #nOO (where n is the motor number). 500 is a conservative initial value for Ixx79. Adjust appropriately (most likely to increase) to force the motor to lock tightly onto a phase.
4. Wait for the motor to settle.
5. Record the value from the ECT entry for commutation result. This is the same address location to which Ixx83 was set. Issue a read command in the Watch Window of PeWin32Pro2 if desired:



6. Issue a #nK to kill the motor
7. Restore Ixx29, and Ixx79 to their original values (if applicable)
8. Set up the following (Motor 1 example):

```
#define Mtr1PhaseOffset 4188614 ; This is the recorded value from step 5 -User Input
I180=4 ; Mtr#1 Power-up mode, no automatic absolute position or phase
I181=I183 ; Mtr#1 Power-On Phase Position Address, same as for commutation
I191=$570000 ; Mtr#1 Power-On Phase Format. X Register, 23 bits.
I175=(-Mtr1PhaseOffset*I170)%I171 ; Mtr#1 Phase Position Offset
```

In this mode, issuing a #1\$ command will phase motor 1.

## Absolute Power-On Phasing (Yaskawa Absolute Encoders)

With absolute encoders, the single turn data is used to find an absolute phase position offset per electrical cycle and thus an absolute phase reference position.



**Note**

Prior to implementing a power-on phasing routine, the user should try to phase the motor manually, successfully execute open-loop moves (output and encoder direction matching), and issue jog commands (which require PID tuning). Remember to increase the fatal following error limit with high resolution encoders when executing closed-loop moves

The U-phase in the Yaskawa motor/encoder assemblies is usually aligned with the index pulse, which should result in the same motor phase offset per one revolution for each encoder type (i.e. 16, 17, or 20-bit).

Yaskawa Absolute Encoders' Single-Turn Data M-Variables		
16-bit	17-bit	20-bit
#define Mtr1STD4_15 M180	#define Mtr1STD0_23 M180	#define Mtr1STD4_23 M180
#define Mtr2STD4_15 M280	#define Mtr2STD0_23 M280	#define Mtr2STD4_23 M280
#define Mtr3STD4_15 M380	#define Mtr3STD0_23 M380	#define Mtr3STD4_23 M380
#define Mtr4STD4_15 M480	#define Mtr4STD0_23 M480	#define Mtr4STD4_23 M480
#define Mtr5STD4_15 M580	#define Mtr5STD0_23 M580	#define Mtr5STD4_23 M580
#define Mtr6STD4_15 M680	#define Mtr6STD0_23 M680	#define Mtr6STD4_23 M680
#define Mtr7STD4_15 M780	#define Mtr7STD0_23 M780	#define Mtr7STD4_23 M780
#define Mtr8STD4_15 M880	#define Mtr8STD0_23 M880	#define Mtr8STD4_23 M880
Mtr1STD4_15->Y:\$278800,4,16	Mtr1STD0_23->Y:\$278800,0,24	Mtr1STD4_23->Y:\$278800,4,20
Mtr2STD4_15->Y:\$278804,4,16	Mtr2STD0_23->Y:\$278804,0,24	Mtr2STD4_23->Y:\$278804,4,20
Mtr3STD4_15->Y:\$278808,4,16	Mtr3STD0_23->Y:\$278808,0,24	Mtr3STD4_23->Y:\$278808,4,20
Mtr4STD4_15->Y:\$27880C,4,16	Mtr4STD0_23->Y:\$27880C,0,24	Mtr4STD4_23->Y:\$27880C,4,20
Mtr5STD4_15->Y:\$278800,4,16	Mtr5STD0_23->Y:\$278800,0,24	Mtr5STD4_23->Y:\$278800,4,20
Mtr6STD4_15->Y:\$278824,4,16	Mtr6STD0_23->Y:\$278824,0,24	Mtr6STD4_23->Y:\$278824,4,20
Mtr7STD4_15->Y:\$278828,4,16	Mtr7STD0_23->Y:\$278828,0,24	Mtr7STD4_23->Y:\$278828,4,20
Mtr8STD4_15->Y:\$27882C,4,16	Mtr8STD0_23->Y:\$27882C,0,24	Mtr8STD4_23->Y:\$27882C,4,20

A one-time simple test (per installation) is performed on an unloaded motor to find the motor phase position offset:

Enable the Absolute position read PLC which was previously created in the Feedback section.

- Record the values of Ixx29, and Ixx79 to restore them at the end of test.
- Set Ixx29=0, and write a positive value to Ixx79 then issue a #nO0. 500 is a reasonably conservative initial value for Ixx79. Adjust appropriately (most likely increase) to force the motor (unloaded) to lock tightly onto a phase.
- Record the Single-Turn Data value (defined in the table above) and store in the user defined motor phase offset.
- Issue a #nK to kill the motor
- Restore Ixx29, and Ixx79 to their original values

Yaskawa Absolute Encoders Motor Phase Offset (found from above test procedure)		
16-bit	17-bit	20-bit
#define PhaseOffset_16Bit P184 PhaseOffset_16Bit=5461	#define PhaseOffset_17Bit P184 PhaseOffset_17Bit=10922	#define PhaseOffset_20Bit P184 PhaseOffset_20Bit=30000



**Note**

Appropriate masking is required with 17-bit encoders to process the data correctly.

## Absolute Power-On Phasing Example PLCs (Yaskawa)

With the motor phase position offset established, the phase position register can now be modified on power-up to compensate for the calculated offset. This allows the user to issue jog commands or close the loop and run a motion program on power-up or reset.

### Channel 1 Driving a 16-bit Yaskawa Absolute Encoder

```
#define Mtr1PhasePos      M171    ; Suggested M-Variables
Mtr1PhasePos->X:$B4,24,S
#define Mtr1PhaseErr     M148    ; Suggested M-Variables
Mtr1PhaseErr->Y:$C0,8
#define Mtr1CommSize     I171    ;
#define Mtr1CommCycles  I170    ;
#define Mtr1CommRatio    P170    ; Motor 1 commutation cycle size (Ixx71/Ixx70 counts)
Mtr1CommRatio=Mtr1CommSize/Mtr1CommCycles

Open PLC 1 Clear
Mtr1PhasePos = ((Mtr1STD4_15 % Mtr1CommRatio) - PhaseOffset_16Bit) * 32 * Mtr1CommCycles
Mtr1PhaseErr = 0
Disable PLC 1
Close
```

### Channel 1 Driving a 17-bit Yaskawa Absolute Encoder

```
#define Mtr1PhasePos      M171    ; Suggested M-Variables
Mtr1PhasePos->X:$B4,24,S
#define Mtr1PhaseErr     M148    ; Suggested M-Variables
Mtr1PhaseErr->Y:$C0,8
#define Mtr1CommSize     I171    ;
#define Mtr1CommCycles  I170    ;
#define Mtr1CommRatio    P170    ; Motor 1 commutation cycle size (Ixx71/Ixx70 counts)
Mtr1CommRatio=Mtr1CommSize/Mtr1CommCycles

Open PLC 1 Clear
Mtr1PhasePos = ((Int((Mtr1STD0_23&$1FFFF0)/$F) % Mtr1CommRatio) - PhaseOffset_17Bit) * 32 *
Mtr1CommCycles
Mtr1PhaseErr = 0
Disable PLC 1
Close
```

### Channel 1 Driving a 20-bit Yaskawa Absolute Encoder

```
#define Mtr1PhasePos      M171    ; Suggested M-Variables
Mtr1PhasePos->X:$B4,24,S
#define Mtr1PhaseErr     M148    ; Suggested M-Variables
Mtr1PhaseErr->Y:$C0,8
#define Mtr1CommSize     I171    ;
#define Mtr1CommCycles  I170    ;
#define Mtr1CommRatio    P170    ; Motor 1 commutation cycle size (Ixx71/Ixx70 counts)
Mtr1CommRatio=Mtr1CommSize/Mtr1CommCycles

#define TwoToThe20th      1048576

Open PLC 1 Clear
If (Mtr1STD4_23 !< PhaseOffset_20Bit)
    Mtr1PhasePos = (Mtr1STD4_23 - PhaseOffset_20Bit) * 32
Else
    Mtr1PhasePos = (TwoToThe20th - PhaseOffset_20Bit + Mtr1STD4_23) * 32
EndIf
Mtr1PhaseErr = 0;
Disable PLC 1
Close
```



#### Note

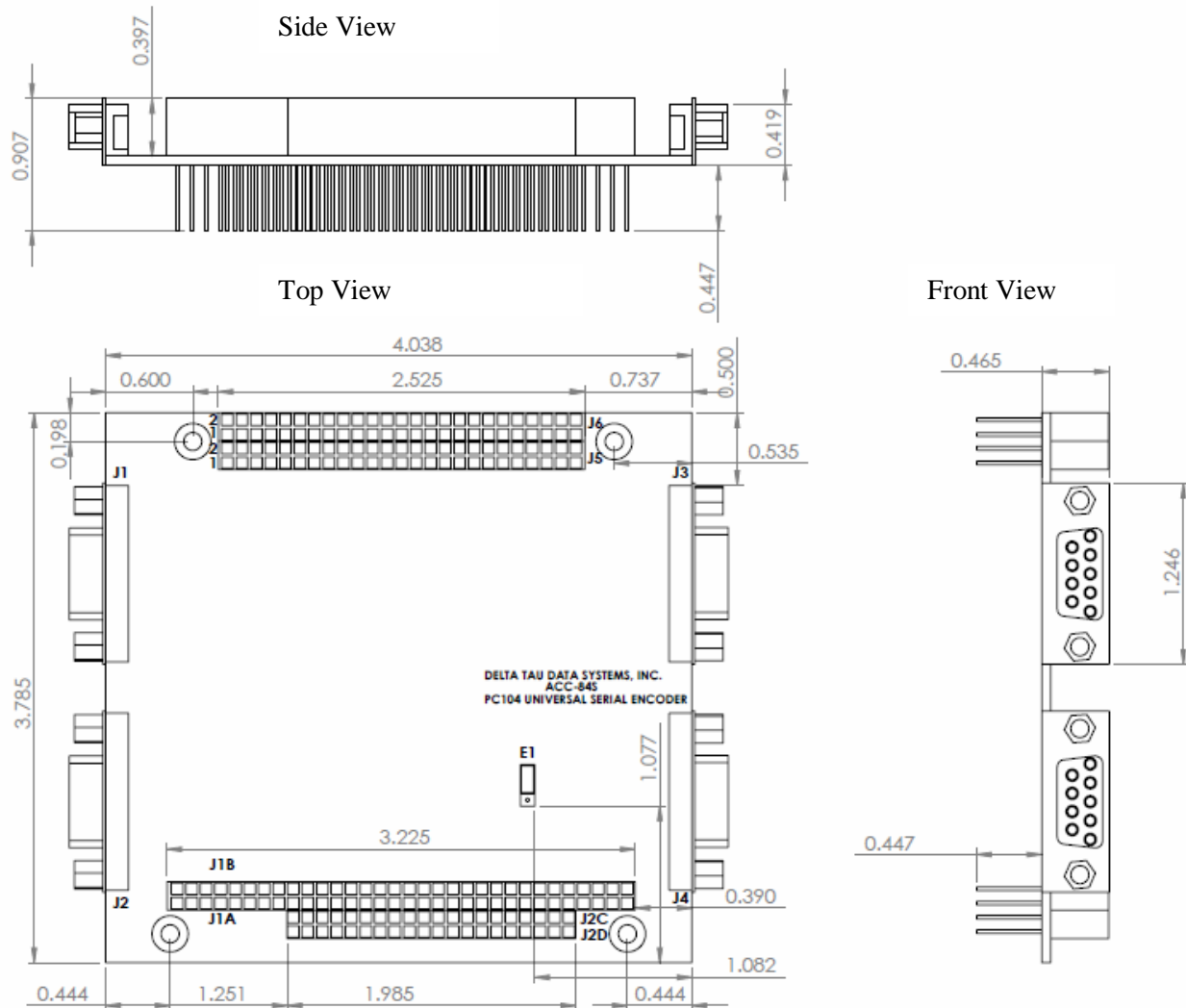
It is highly recommended to try the sequence in this PLC manually at first (using the Terminal Window). In some cases, the Motor Phase Position Offset has to be added instead of subtracted depending on the direction of the encoder mounting/decoding. Turbo PMAC has no ability to change the direction of serial encoder data.



## LAYOUT AND PINOUTS

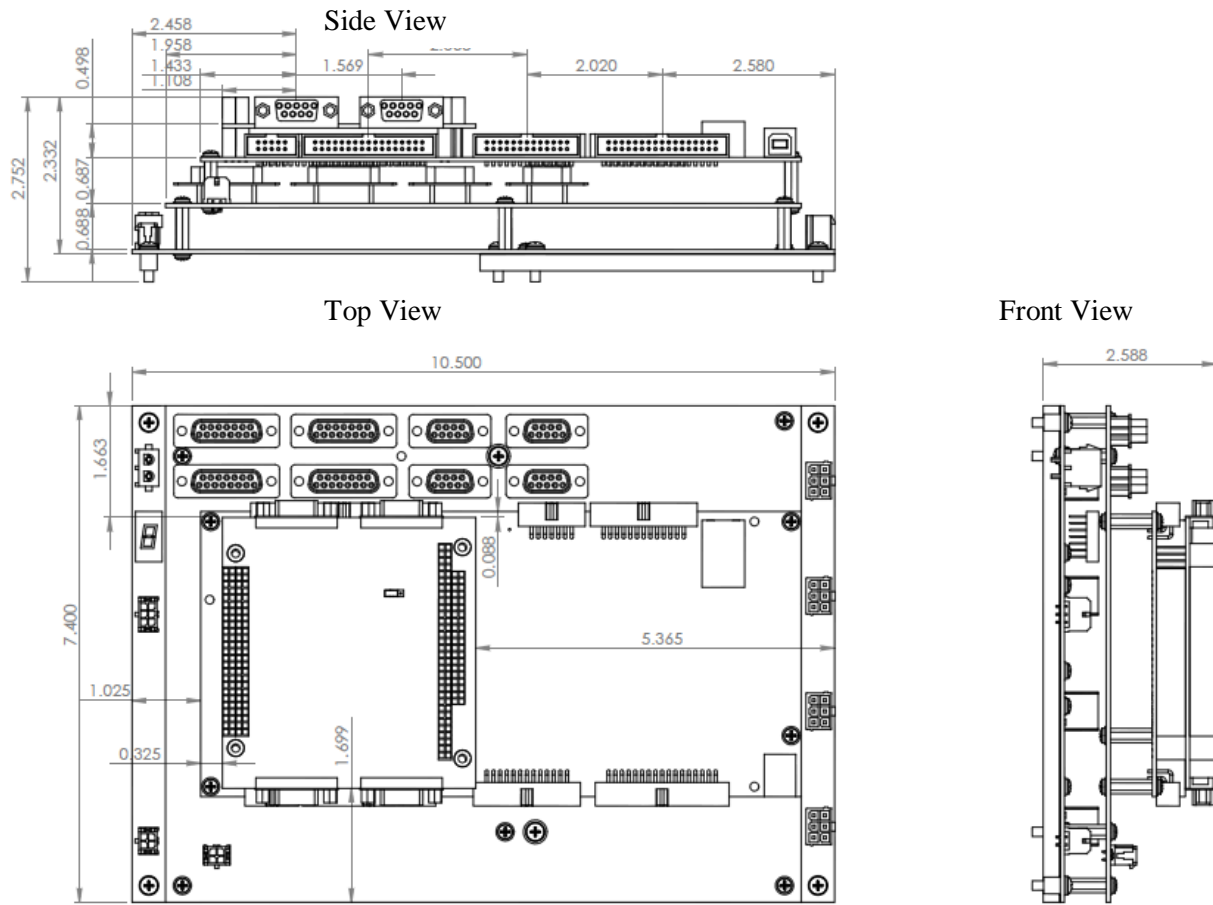
### Board Layout

#### ACC-84S Standalone



All dimensions are in inches.

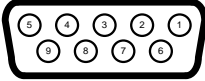
## ACC-84S Mounted on the Turbo Clipper Drive



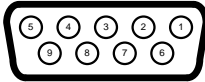
All dimensions are in inches.

## Encoder Feedback Connector Pinouts

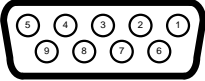
### J1: Encoder Feedback Channel 1

J1: D-Sub DE-9F Mating: D-Sub DE-9M							
Pin #	Symbol	Function	SSI	EnDat	Yaskawa	BiSS B/C	Description
1	Clock-	Output	CLK-	CLK-	-	MA-	
2	Data-	In/Out	DAT-	DAT-	SDI (blu/blk)	SLO-	
3	ENA-		-	-		-	
4	GND	Common	GND	GND	GND (blk)	GND	Common Ground
5	GND	Common	GND	GND	GND (blk)	GND	
6	Clock+	Output	CLK+	CLK+	-	MA+	
7	Data+	In/Out	DAT+	DAT+	SDO (blu)	SLO+	
8	ENA+						
9	5V	Output	+5VDC	+5VDC	+5VDC (red)	+5VDC	Encoder Power

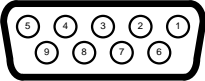
### J2: Encoder Feedback Channel 2

J2: D-Sub DE-9F Mating: D-Sub DE-9M							
Pin #	Symbol	Function	SSI	EnDat	Yaskawa	BiSS B/C	Description
1	Clock-	Output	CLK-	CLK-	-	MA-	
2	Data-	Input/Output	DAT-	DAT-	SDI (blu/blk)	SLO-	
3	ENA-		-	-		-	
4	GND	Common	GND	GND	GND (blk)	GND	Common Ground
5	GND	Common	GND	GND	GND (blk)	GND	
6	Clock+	Output	CLK+	CLK+	-	MA+	
7	Data+	Input/Output	DAT+	DAT+	SDO (blu)	SLO+	
8	ENA+						
9	5V	Output	+5VDC	+5VDC	+5VDC (red)	+5VDC	Encoder Power

### J3: Encoder Feedback Channel 3

J3: D-Sub DE-9F Mating: D-Sub DE-9M							
Pin #	Symbol	Function	SSI	EnDat	Yaskawa	BiSS B/C	Description
1	Clock-	Output	CLK-	CLK-	-	MA-	
2	Data-	Input/Output	DAT-	DAT-	SDI (blu/blk)	SLO-	
3	ENA-		-	-		-	
4	GND	Common	GND	GND	GND (blk)	GND	Common Ground
5	GND	Common	GND	GND	GND (blk)	GND	
6	Clock+	Output	CLK+	CLK+	-	MA+	
7	Data+	Input/Output	DAT+	DAT+	SDO (blu)	SLO+	
8	ENA+						
9	5V	Output	+5VDC	+5VDC	+5VDC (red)	+5VDC	Encoder Power

### J4: Encoder Feedback Channel 4

J4: D-Sub DE-9F Mating: D-Sub DE-9M							
Pin #	Symbol	Function	SSI	EnDat	Yaskawa	BiSS B/C	Description
1	Clock-	Output	CLK-	CLK-	-	MA-	
2	Data-	Input/Output	DAT-	DAT-	SDI (blu/blk)	SLO-	
3	ENA-		-	-		-	
4	GND	Common	GND	GND	GND (blk)	GND	Common Ground
5	GND	Common	GND	GND	GND (blk)	GND	
6	Clock+	Output	CLK+	CLK+	-	MA+	
7	Data+	Input/Output	DAT+	DAT+	SDO (blu)	SLO+	
8	ENA+						
9	5V	Output	+5VDC	+5VDC	+5VDC (red)	+5V DC	Encoder Power



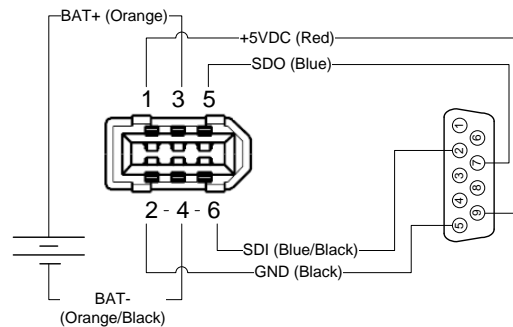
*Note*

- Some SSI devices require 24 V<sub>DC</sub> power which has to be brought in externally. Pins #4, #5, and #9 are unused in this case; leave floating.
- Hardware capture is not available with Serial Data encoders (SSI, EnDat 2.2, BiSS B/C)

## Encoder Specific Connection Information

### Yaskawa Sigma II Encoders

Yaskawa Sigma II absolute encoders require a 3.6 V<sub>DC</sub> battery to maintain the multi-turn data while the controller is powered down. This battery should be placed outside of the ACC-84S and the Yaskawa Sigma II encoder, possibly on the cable. The battery should be installed between the orange (+3.6 V<sub>DC</sub>) and the orange/black wires (GND). Use of ready-made cables by Yaskawa is recommended (Yaskawa part number: UWR00650).



Yaskawa Encoder Cable Has a Female Connector by Default

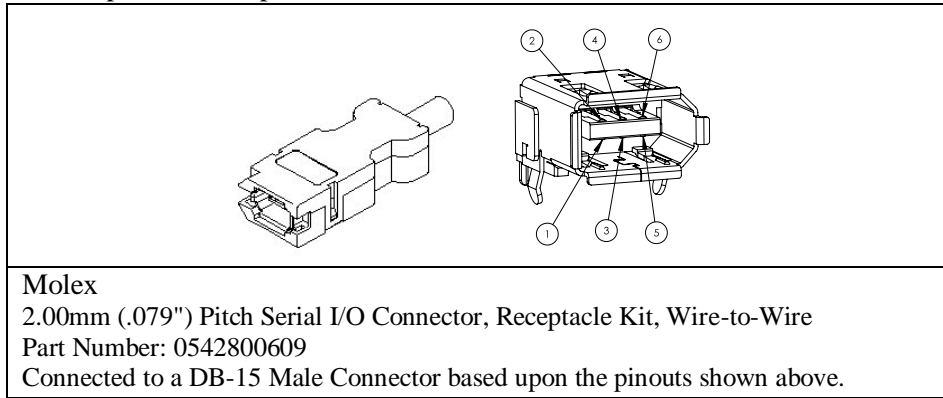
The previous diagram shows the pin assignment from mating IEEE 1394 Yaskawa Sigma II connector to ACC-84S encoder input. The Molex connector required for IEEE 1394 can be acquired in a receptacle kit from Molex: 2.00mm (.079") Pitch Serial I/O Connector, Receptacle Kit, Wire-to-Wire, Molex Part Number: 0542800609.



**Note**

Use an encoder cable with high quality shield. Connect the shield to chassis ground, and use a ferrite core in noise-sensitive environments.

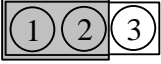
The part number and pin number specifications for the Molex Male connector are as follows:



Pin Number	Function	Wire Color code
1	+5VDC	RED
2	GND	BLACK
3	BAT+	Orange
4	BAT-	Orange/Black (Orange/White)
5	SDO	Blue
6	SDI	Blue/Black (Blue/White)

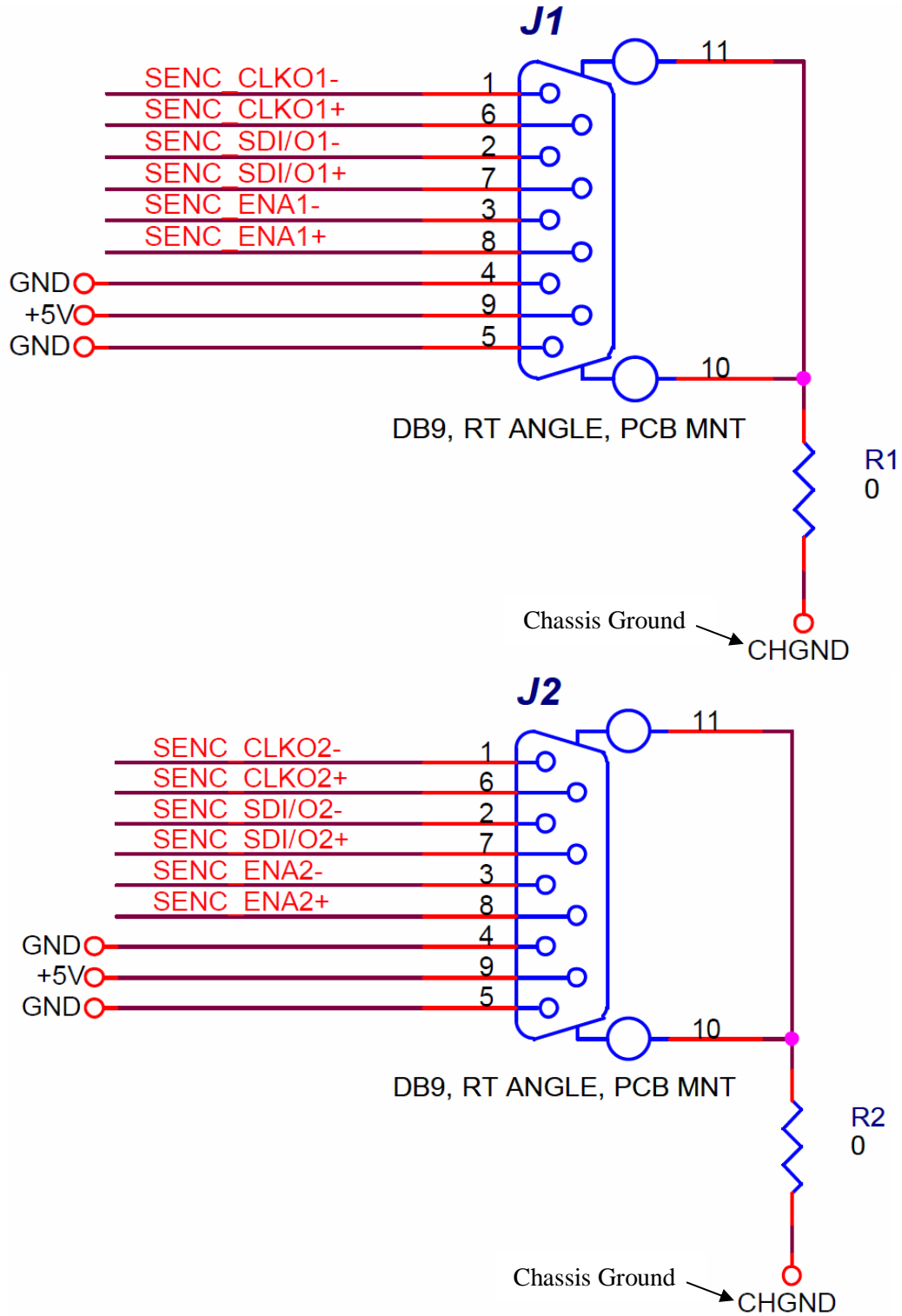
## APPENDIX A: E-POINT JUMPERS

---

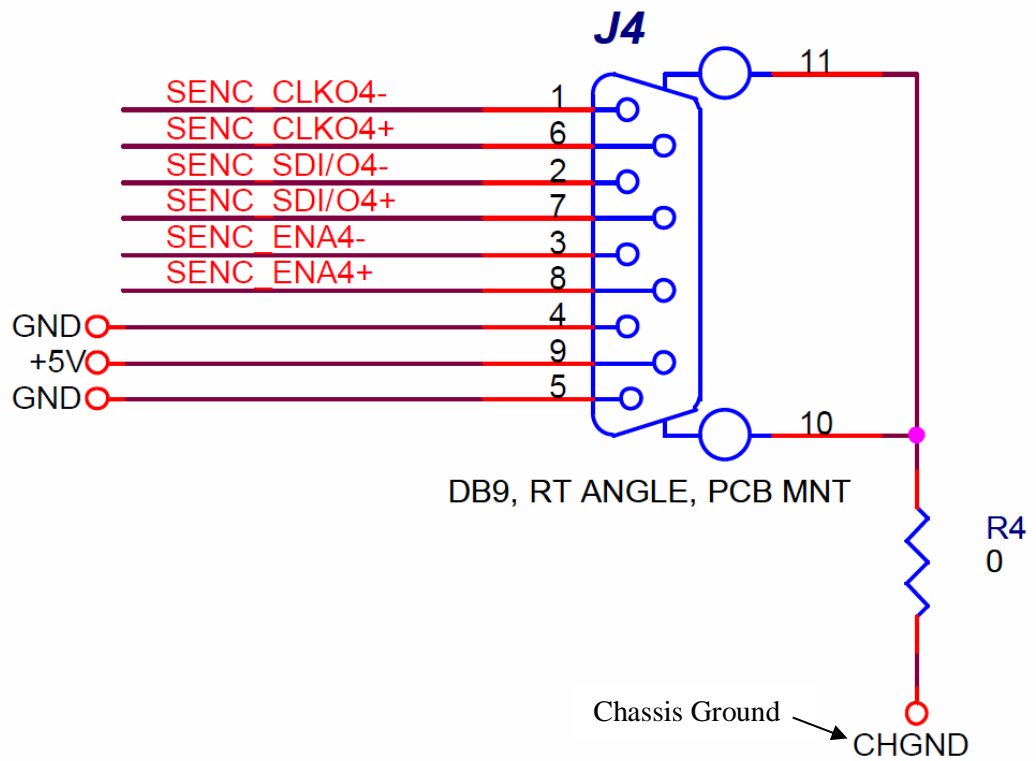
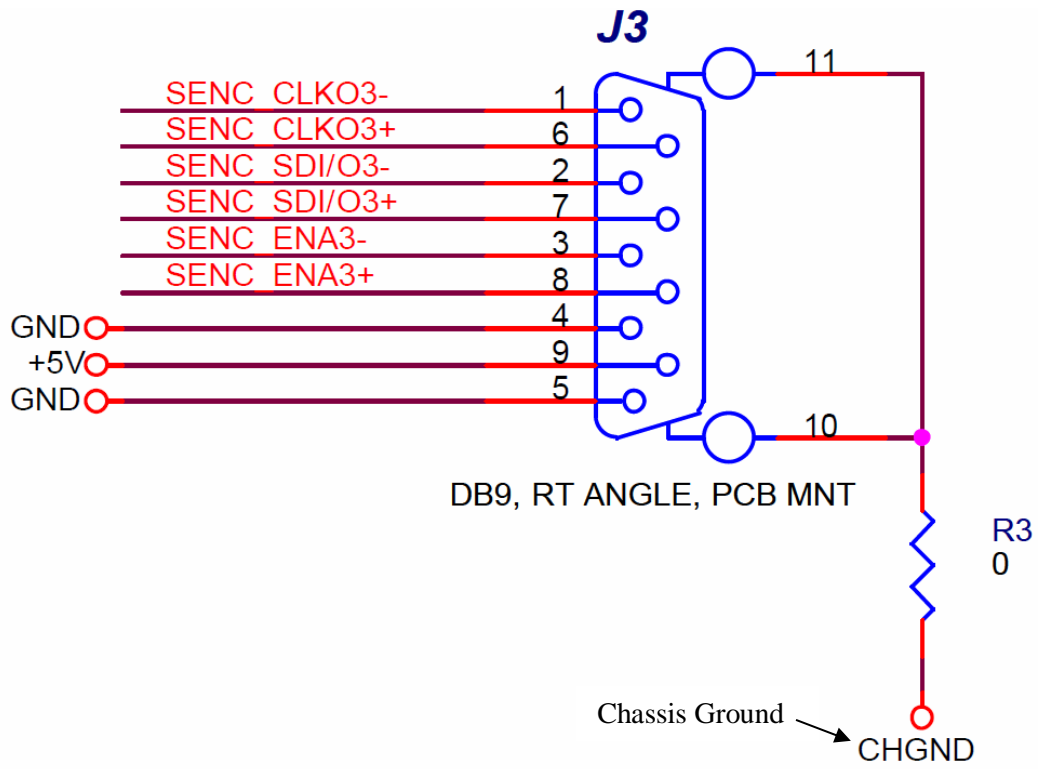
Jumper	Configuration	Default
<p data-bbox="300 331 332 359">E1</p> 	Jump pins 1 to 2 for base address \$78800	Set by factory
	Jump pins 2 to 3 for base address \$78820	

## APPENDIX B: SCHEMATICS

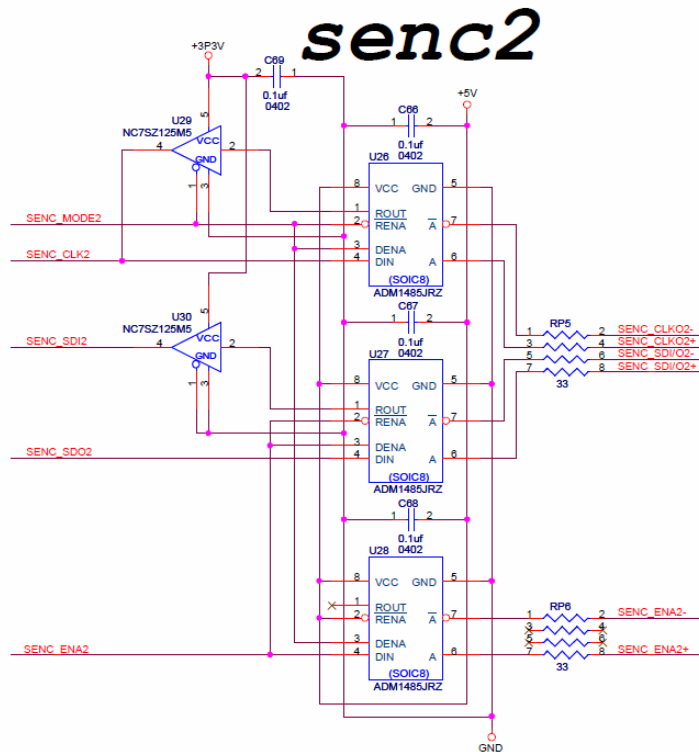
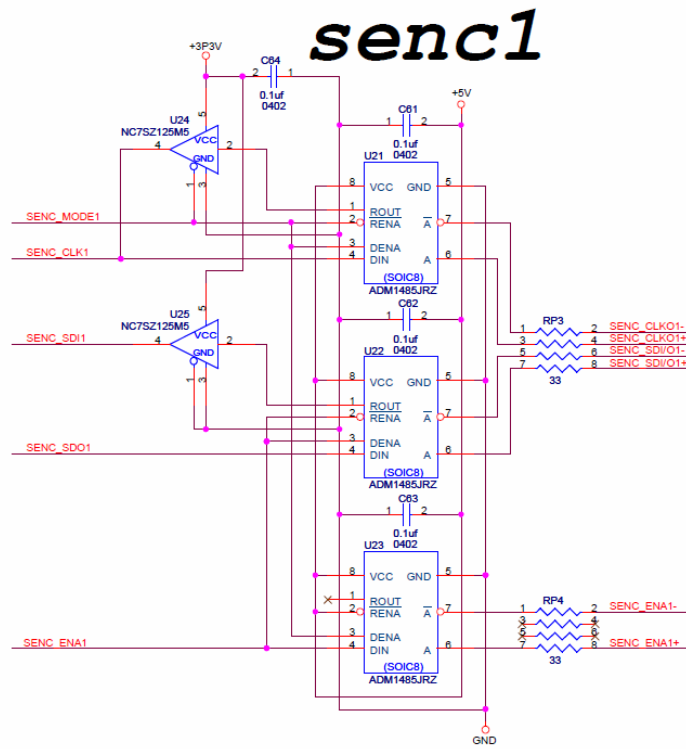
### Encoder Feedback Connectors



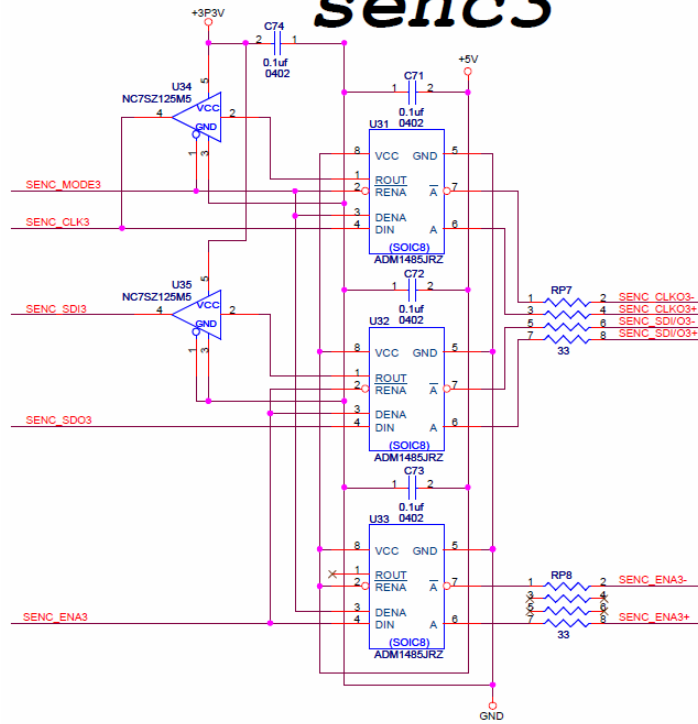




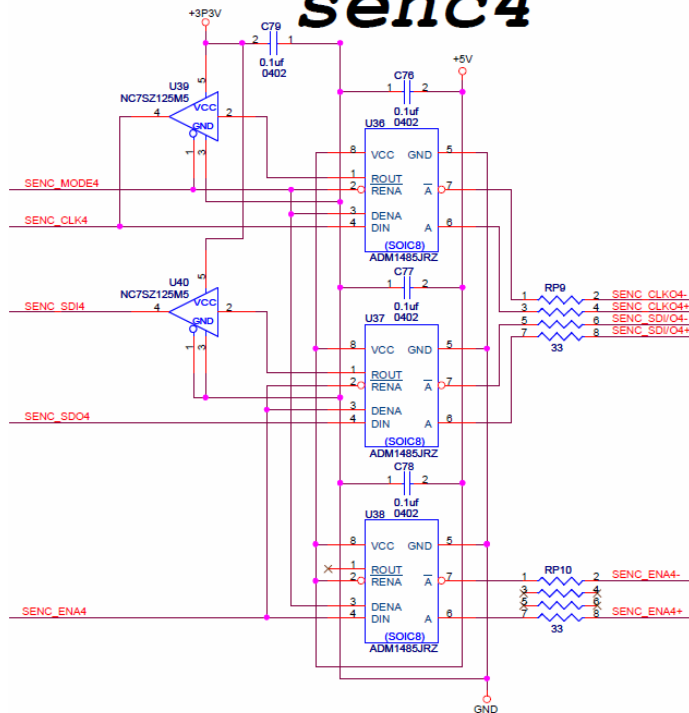
## ACC-84S Serial Encoder Feedback Input Circuitry



# senc3



# senc4



## Stacking Connectors PinOuts

