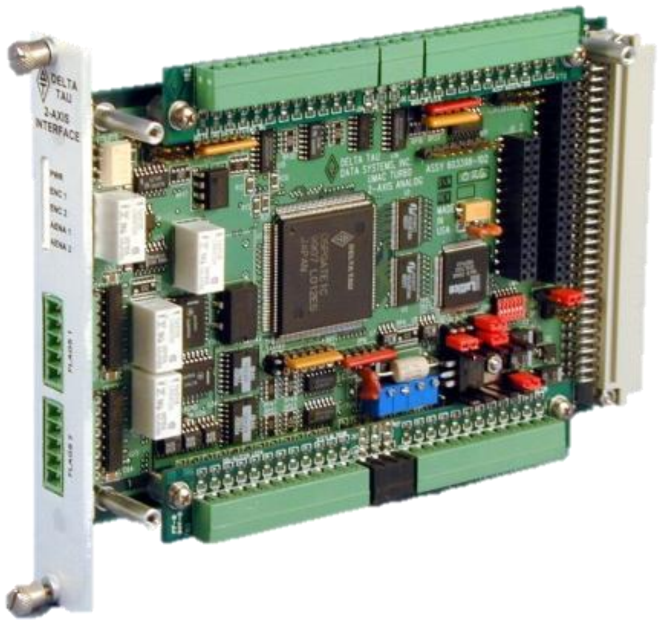
USER MANUAL

Accessory 24E2A



UMAC Axis Expansion Board

4Ax-603398-xUxx

9/24/2015

DELTA TAU
Data Systems, Inc.

NEW IDEAS IN MOTION …

Single Source Machine Control .... *Power // Flexibility // Ease of Use*
21314 Lassen St. Chatsworth, CA 91311 // Tel. (818) 998-2095 Fax. (818) 998-7807 // www.deltatau.com

Copyright Information

To report errors or inconsistencies, call or email:

Delta Tau Data Systems, Inc. Technical Support

Phone: +1 (818) 717-5656

Fax: +1 (818) 998-7807

Email: support@deltatau.com

Web: www.deltatau.com

Operating Conditions

All Delta Tau Data Systems, Inc. motion controller, accessory, and amplifier products 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 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.

Safety Instructions

Qualified personnel must transport, assemble, install, and maintain this equipment. Properly qualified personnel are persons who are familiar with the transport, assembly, installation, and operation of equipment. The qualified personnel must know and observe the following standards and regulations:

IEC364resp.CENELEC HD 384 or DIN VDE 0100
IEC report 664 or DIN VDE 0110
National regulations for safety and accident prevention or VBG 4

Incorrect handling of products can result in injury and damage to persons and machinery. Strictly adhere to the installation instructions. Electrical safety is provided through a low-resistance earth connection. It is vital to ensure that all system components are connected to earth ground.

This product contains components that are sensitive to static electricity and can be damaged by incorrect handling. Avoid contact with high insulating materials (artificial fabrics, plastic film, etc.). Place the product on a conductive surface. Discharge any possible static electricity build-up by touching an unpainted, metal, grounded surface before touching the equipment.

Keep all covers and cabinet doors shut during operation. Be aware that during operation, the product has electrically charged components and hot surfaces. Control and power cables can carry a high voltage, even when the motor is not rotating. Never disconnect or connect the product while the power source is energized to avoid electric arcing.

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 understanding or use of the equipment. It follows the discussion of interest.

REVISION HISTORY				
REV.	**DESCRIPTION**	**DATE**	**CHG.**	**APPVD**
1	Added CE declaration	06/07/06	CP	SF
2	Added UL Approval and agency safety info.	09/30/09	CP	SF
3	Corrected BEQU2 Error	02/04/10	CP	SF
4	Updated connection diagram, general formatting	07/21/15	CP	RN

Table Of Contents

INTRODUCTION 9
Overview 9
Features 9
Board Configuration 10

SPECIFICATIONS 11
Environmental Specifications 11
Physical Specifications 11
Electrical Specifications 12
Agency Approval and Safety 12

HARDWARE SETUP 13
Switch Configuration 13
UMAC Address DIP Switch S1 13
Legacy MACRO Dip Switch Settings 13
Converting Memory Address from Turbo to Power PMAC 14
ACC-24E2 Clock Settings 14
Resistor Pack Configuration 15
Differential/Single-Ended and Encoder Loss Detection 15
Termination Resistors Packs 16
Limit/Flag Voltage Level Resistor Packs 16
UBUS Specific Resistor Packs 16
UBUS Specific Resistor Packs 16
OPTO-Isolation Considerations 17
ACC-24E2 Limit and Flag Wiring 18
Connecting Limits/Flags to the ACC-24E2 19
Amplifier Fault Circuit 20
Amplifier Enable Circuit 21
Loss of Encoder Circuit 21
ACC-24E2A Encoder Loss Detection with UMAC Turbo CPU 21
ACC-24E2A Encoder Loss Detection with UMAC MACRO CPU 22
ACC-24E2A Encoder Loss Detection with Power PMAC CPU 22
Position Compare Port Driver IC 22

CONNECTIONS 23
ACC-24E2A Board Layout - Terminal Block Option 23
ACC-24E2A Board Layout - DB15 Option 23
Mating Connectors 24
Terminal Block Connectors 24
DB15 Connector Option 24

Indicators..24
Overall Wiring Diagram 25
Sample Wiring Diagrams..26
TTL Level Inputs and Outputs..*26*
Position Limits, Home Flag, and User Flag...*27*
DAC Outputs...*27*
Stepper Drive, Pulse and Direction Outputs (TTL Level)...................................*28*
Amplifier Fault Inputs...*29*
Amplifier Enable Outputs..*30*

SOFTWARE SETUP 31
Using ACC-24E2A with UMAC Turbo 31
Servo IC Numbering..*31*
Servo Channel Numbering...*31*
Multi-Channel I-Variables...*31*
Single-Channel I-Variables..*32*
Encoder Conversion Table I-Variables...*33*
Motor Addressing I-Variables..*33*
TURBO UMAC Example Setups..*36*
Using ACC-24E2A with ULTRALITE/MACRO STATION 38
Clocks and Strobe Word..*38*
Node-Specific Gate Array MI-Variables...*38*
Using ACC-24E2A with Power PMAC 42
DSPGATE1 Servo IC Data Structure Elements...*42*
Software Setup for Clock Signals...*43*
Software Setup for Position Feedback...*45*
Hall-Style Commutation Sensors...*47*
MLDT Sensors..*49*
Software Setup for Flags..*52*
Software Setup for Amplifiers..*54*
DSPGATE1 (PMAC2-Style Servo ASIC) Register Elements....................................*57*
Using ACC-24E2A for MLDT Feedback..60
MLDT Hardware Setup of the ACC-24E2A...*60*
MLDT Software Setup of the UMAC Turbo..*60*
MLDT Feedback for UMAC-MACRO..*64*

ACC-24E2A TERMINAL BLOCK DESCRIPTION 68
Connector TB1 TOP - Encoder 1 68
Connector TB2 Top - Encoder 2 68
Connector TB3 Top – EQU Outputs 69
Connector TB1 Bottom Amp – Out 1..70
Connector TB2 Bottom Amp – Out 2..70
Connector TB3 Bottom – Analog Power...70

Connector TB1 Front- Limits 1 71
Connector TB2 Front- Limits 2 71

ACC-24E2 OPTION 1A TERMINAL BLOCK DESCRIPTION 72
Connector TB1 Top - Encoder 3 72
Connector TB2 Top – Encoder 4 72
Connector TB3 Top – EQU Outputs 73
Connector TB1 Bottom Amp-Out 3 74
Connector TB2 Bottom Amp-Out 4 74
Connector TB3 Bottom-Analog Power 74
Connector TB1 Front - Limits 3 75
Connector TB2 Front - Limits 4 75

ACC-24E2A DB15 CONNECTOR OPTION 76
DB15 Style Connector J1 Top - Encoder 1 / EQU 76
DB15 Style Connector J2 Top - Encoder 2 / EQU 76
DB15 Style Connector J1 Bottom Amp – Out 1/Analog Power 78
DB15 Style Connector J2 Bottom Amp – Out 2/Analog Power 78
Connector TB1 Front-Limits 1 79
Connector TB2 Front-Limits 2 79

UBUS PINOUTS 80
P1 UBUS (96-Pin Header) 80

DECLARATION OF CONFORMITY 81

E-POINT JUMPER SETTINGS 82
ACC-24E2A Base Board (Channels* 1 and 2) 82
ACC-24E2A Option 1 Board (Channels 3 and 4) 83

INTRODUCTION

Overview

The ACC-24E2A is an axis expansion board which provides 2 or 4 channels of PMAC2-style servo interface circuitry for UMAC and Ultralite/MACRO Station controllers. The ACC-24E2A is part of the UMAC or MACRO Pack family of expansion cards and these accessory cards are designed to plug into an industrial 3U rack system.

Note

Many ACC-24E2A features are common to other accessories of the ACC-24E2 family; these common features are referred to in this manual as ACC-24E2x.

Up to eight ACC-24E2A boards can be connected to one UMAC providing up to 32 additional channels of servo interface circuitry. The 16-Axes MACRO CPU can support four ACC-24E2x cards and the 8-Axes MACRO CPU can support 2 ACC-24E2x.

The ACC-24E2x board contains no processor; it has one highly integrated 4-channel PMAC2-style Servo IC with the buffering circuitry and connectors around them. The two-axis ACC-24E2x plugs into the backplane and uses one slot in the rack. If two more axes are needed, ACC-24E2x Option 1 can be plugged into the ACC-24E2x connectors. The ACC-24E2x with its Option 1 card takes up a total of two slots.
Some new features added to the family of ACC-24E2x breakout boards include:

- Loss of encoder circuit
- 5V to 24V Flag inputs
- Pulse and direction outputs for stepper systems or MLDTs

Features

The ACC-24E2A board can be used with any UMAC or MACRO Station CPU, interfacing through the expansion port.
The ACC-24E2A supports a wide variety of servo and stepper interfaces:

- Analog +/-10V velocity commands
- Analog +/-10V torque commands
- Sinusoidal analog +/-10V phase current commands
- Pulse-and-direction commands

Board Configuration

An ACC-24E2A comes standard with one Servo IC providing four servo interface channels, which are brought out on terminal blocks (standard) or DB15 connector. Each channel of servo interface circuitry includes the following:

- Two output command signal sets, configurable as either:
- One pulse-and-direction
- Two DAC outputs
- 3-channel differential/single-ended encoder input
- Eight input flags, two output flags

Option 1A: If Option 1A is ordered, the circuitry and input/output connectors are provided for the third and fourth channels associated with the Servo IC on the main ACC-24E2A. The command signals for this option are ±10V.

Option 1D: If Option 1D is ordered, the circuitry and input/output connectors are provided for the third and fourth channels associated with the Servo IC on the main ACC-24E2A. The command signals for this option are digital PWM signals for direct PWM commutation. The option 1D description can be found in the ACC-24E2 manual.

Option DB: If the option DB is ordered the outputs and inputs to the amplifiers and encoders will be serviced from DB15 connectors. See ACC-24E2A DB15 Connector Option section for pin outs.

SPECIFICATIONS

Environmental Specifications

Description	**Specification**
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**
Dimensions w/o Option 1A	Length: 16.256 cm (6.4 in.) Height: 10 cm (3.94 in.) Width: 2.03 cm (0.8 in.)	
Dimensions with Option 1A	Length: 16.256 cm (6.4 in.) Height: 10 cm (3.94 in.) Width: 4.06 cm (1.6 in.)	
Weight w/o Option 1A	192 g	Front Plate included
Weight with Option 1A	370 g	Front Plate included
Terminal Block Connectors	FRONT-MC1,5/12-ST3,81 FRONT-MC1,5/5-ST3,81 FRONT-MC1,5/3-ST3,81	Terminal Blocks from Phoenix Contact. UL-94V0
DB Option Connectors	DB15 Female	UL-94V0
The width is the width of the front plate. The length and height are the dimensions of the PCB.		

Electrical Specifications

Description	**Specification**
ACC-24E2A Power Requirements	5V @ 0.55A (±10%) +15V @ 0.16A (±10%) -15V @ 0.07A (±10%)
ACC-24E2A with Option 1A Power Requirements	5V @ 0.95A (±10%) +15V @ 0.30A (±10%) -15V @ 0.12A (±10%)

WARNING

If more than four ACC-24E2A's with Option 1A are used in a UMAC system, the ACC-E1 or ACC-F1 power supplies will not have enough 15V power. Delta Tau recommends using an external ±15V power supply for systems with more than four ACC-24E2A boards. The external power supply should be connected to the unit from the terminal blocks (TB3 bottom) or DB connections (J1 or J2 Bottom) and jumpers E85, E87, and E88 must also be removed.

Agency Approval and Safety

Item	**Description**
CE Mark	Full Compliance
EMC	EN55011 Class A Group 1 EN61000-3-2 Class A EN61000-3-3 EN61000-4-2 EN61000-4-3 EN61000-4-4 EN61000-4-5 EN61000-4-6 EN61000-4-11
Safety	EN 61010-1
UL	UL 61010-1 File E314517
cUL	CAN/CSA C22.2 No. 1010.1-92 File E314517
Flammability Class	UL 94V-0

HARDWARE SETUP

Switch Configuration

UMAC Address DIP Switch S1

Chip Select	Base Address				SW1 Positions					
	TURBO	MACRO	POWER		6	5	4	3	2	1
			Offset	Index (n)						
CS10	$78200	$8000	$600000	4	ON	ON	ON	ON	ON	ON
	$78220	N/A	$600100	5	ON	ON	ON	ON	**OFF**	ON
	$78300	$8040	$700000	6	ON	ON	ON	ON	ON	**OFF**
	$78320	N/A	$700100	7	ON	ON	ON	ON	**OFF**	**OFF**
CS12	$79200	$9000	$608000	8	ON	ON	ON	**OFF**	ON	ON
	$79220	N/A	$608100	9	ON	ON	ON	**OFF**	**OFF**	ON
	$79300	$9040	$708000	10	ON	ON	ON	**OFF**	ON	**OFF**
	$79320	N/A	$708100	11	ON	ON	ON	**OFF**	**OFF**	**OFF**
CS14	$7A200	$A000	$610000	12	ON	ON	**OFF**	ON	ON	ON
	$7A220	N/A	$610100	13	ON	ON	**OFF**	ON	**OFF**	ON
	$7A300	$A040	$710000	14	ON	ON	**OFF**	ON	ON	**OFF**
	$7A320	N/A	$710100	15	ON	ON	**OFF**	ON	**OFF**	**OFF**
CS16	$7B200	$B000	$618000	16	ON	ON	**OFF**	**OFF**	ON	ON
	$7B220	N/A	$618100	17	ON	ON	**OFF**	**OFF**	**OFF**	ON
	$7B300	$B040	$718000	18	ON	ON	**OFF**	**OFF**	ON	**OFF**
	$7B320	N/A	$718100	19	ON	ON	**OFF**	**OFF**	**OFF**	**OFF**

Note

- ON designates Closed. OFF designates Open
- Factory default is all ON

Legacy MACRO Dip Switch Settings

S1-1	S1-2*	S1-3	S1-4	Board No.	IC No.	Base Address
ON	ON	OFF	OFF	1	2	$00C040
OFF	OFF	OFF	OFF	2	3	$00C060
* Always set to OFF for legacy MACRO Stations (part number 602804-100 thru 602804-104)						

S1-5 and S1-6 are used to determine whether the ACC-24E2 is communicating to a Turbo 3U PMAC or a MACRO Station CPU.

S1-5	S1-6	Function
OFF	OFF	3U MACRO Station use

Converting Memory Address from Turbo to Power PMAC

To convert a Turbo PMAC Y memory address to Power PMAC corresponding address the following formula can be used:

$$A_{PT} = V_0 + A_{PB} \tag{1}$$

Where A_{PT} is the target Power PMAC memory address. A_{PB} is the Power PMAC base address and V_O is the memory offset given by:

$$V_O = (\text{int})[\frac{A_{TT} - A_{TB}}{8}] \times 64 + ((A_{TT} - A_{TB})\%8) \times 4 \tag{2}$$

Where A_{TB} is the UMAC Turbo base address of the card, and A_{TT} is the address of the UMAC Turbo target memory location. Note that (int) in Equation 2 indicates that expression within the square brackets following it must be cast to an integer before use. If the expression valuates to a non-integer, the fractional part must be truncated, not rounded. The "%" sign indicates that modulo operator; e.g, a%b will yield the remainder from the quotient a/b.

If converting X memory address this formula can be used:

$$A_{PT} = V_0 + A_{PB} + 32 \tag{3}$$

ACC-24E2 Clock Settings

The Phase Clock and Servo Clock must be configured on each ACC-24E2A baseboard. Each system can have only one source for the servo and phase clocks and jumpers must be set appropriately to avoid a timing conflict or a watchdog condition.

Starting in UMAC Turbo firmware version 1.937, the firmware will set the clock settings for the ACC-24E2 cards in the UBUS automatically. To enable this feature, set jumper E13 from 2 to 3 for all of the ACC-24E2s plugged into the UMAC system. At re-initialization (either **$$$***** command or power up with E3 jumpered on UMAC), the firmware will know that all of the cards are in the auto configuration setup and will assign the card with the lowest base address setting (usually $78200) the task of sourcing the clocks by setting variable I19 to the appropriate register. The clocks will be set initially to the factory default servo update cycle and phase clock cycle. For a better understanding of this feature, refer to description of I19 in the Turbo Software Reference Manual.

For UMAC Turbo systems with firmware older than version 1.937, set one of the ACC-24E2s to transmit (E13 set 2-3) the phase and servo clock (usually the card at the lowest base address setting) and set the rest of the ACC-24E2s to receive (E13 set 1-2) the phase and servo clocks.
For MACRO systems, the clock select jumper should be set to receive servo and phase clocks because the MACRO CPU always provides the clocks. For the ACC-24E2A, E13 should be set 1-2.

Resistor Pack Configuration

Differential/Single-Ended and Encoder Loss Detection

The differential input signal pairs to the PMAC have user-configurable pull-up/pull-down resistor networks to permit the acceptance of either single-ended or differential signals in one setting, or the detection of lost differential signals in another setting.

- The '+' inputs of each differential pair each have a hard-wired 1 kΩ pull-up resistor to +5V. This cannot be changed.
- The '-' inputs of each differential pair each have a hard-wired 2.2 kΩ resistor to +5V; also each has another 2.2 kΩ resistor as part of a socketed resistor pack that can be configured as a pull-up resistor to +5V, or a pull-down resistor to GND.

If this socketed resistor is configured as a pull-down resistor (the default configuration), the combination of pull-up and pull-down resistors on this line acts as a voltage divider, holding the line at +2.5V in the absence of an external signal. This configuration is required for single-ended inputs using the '+' lines alone; it is desirable for unconnected inputs to prevent the pick-up of spurious noise; it is permissible for differential line-driver inputs.

If this socketed resistor is configured as a pull-up resistor (by reversing the SIP pack in the socket), the two parallel 2.2 kΩ resistors act as a single 1.1 kΩ pull-up resistor, holding the line at +5V in the absence of an external signal. This configuration is required if encoder-loss detection is desired; it is required if complementary open-collector drivers are used; it is permissible for differential line-driver inputs even without encoder loss detection.

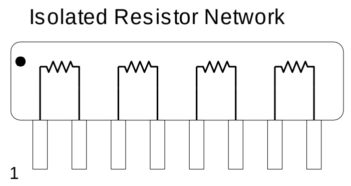
If Pin 1 of the resistor pack (marked by a dot on the pack) matches Pin 1 of the socket (marked by a wide white square solder pin on the front side of the board), then the pack is configured as a bank of pull-down resistors. If the pack is reversed in the socket, it is configured as a bank of pull-up resistors.

The following table lists the pull-up/pull-down resistor pack for each input device:

Device	Resistor Pack	Pack Size	SIP
Encoder 1	RP22	6-pin	2.2 KΩ
Encoder 2	RP24	6-pin	2.2 KΩ
Encoder 3	RP22	6-pin	2.2 KΩ
Encoder 4	RP24	6-pin	2.2 KΩ

Termination Resistors Packs

The ACC-24E2A provides sockets for termination resistors on differential input pairs coming into the board. As shipped, there are no resistor packs in these sockets. If these signals are brought long distances into the ACC-24E2A board and ringing at signal transitions is a problem, SIP resistor packs may be mounted in these sockets to reduce or eliminate the ringing.
All termination resistor packs have independent resistors (no common connection) with each resistor using two adjacent pins as shown below.



Channel 1	Channel 2	SIP	Description
RP23	RP25	220Ω	Termination resistor to reduce ringing (not installed by default).

Limit/Flag Voltage Level Resistor Packs

The ACC-24E2A limit and flag circuits also give the flexibility to wire in standard 12V to 24V limits and flags or wire in 5V level limits and flags on a channel basis. The default is set for the standard 12V to 24V inputs but if the resistor pack is added to the circuit, the card can read 5V inputs.
Channel Specific Resistor Packs

Channel 1	Channel 2	SIP	Description
RP45	RP46	1KΩ	Install for 5V limits

UBUS Specific Resistor Packs

Resistor Pack	SIP	Description
RP5	220Ω	Terminator (not installed, only used for non-UBUS)
RP6	2.2KΩ	Pull Down for old MACRO CPU Pull Up for UMAC Turbo and MACRO

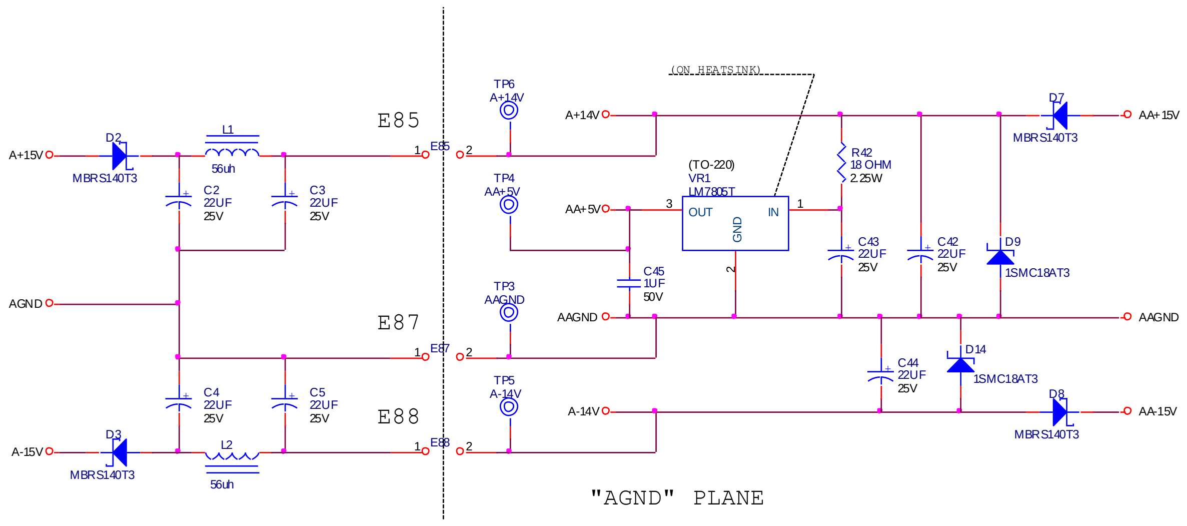
UBUS Specific Resistor Packs

Resistor Pack	SIP	Description
RP7		Installed by default, Phase/Servo signals

OPTO-Isolation Considerations

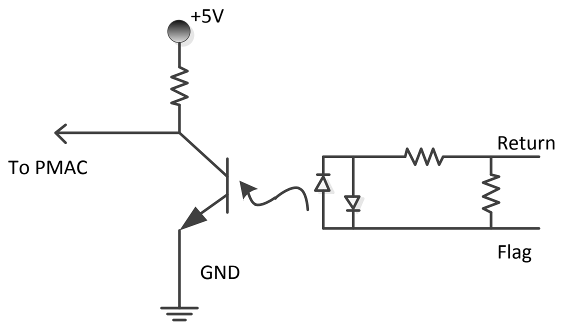
As shipped from the factory, the ACC-24E2A obtains its power from the UMAC Backplane or UBUS. Using this type of setup will defeat opto isolation since the analog ground plane will be tied directly to the digital ground plane.

To optically isolate the analog ground plane from the digital ground plane, connect an external power supply to one of the many AA+15V, AA-15V, and AAGND inputs on the ACC-24E2A terminal blocks or DB connectors. Also, remove the E85, E87, and E88 jumpers to isolate the external power from the UBUS power supplies.

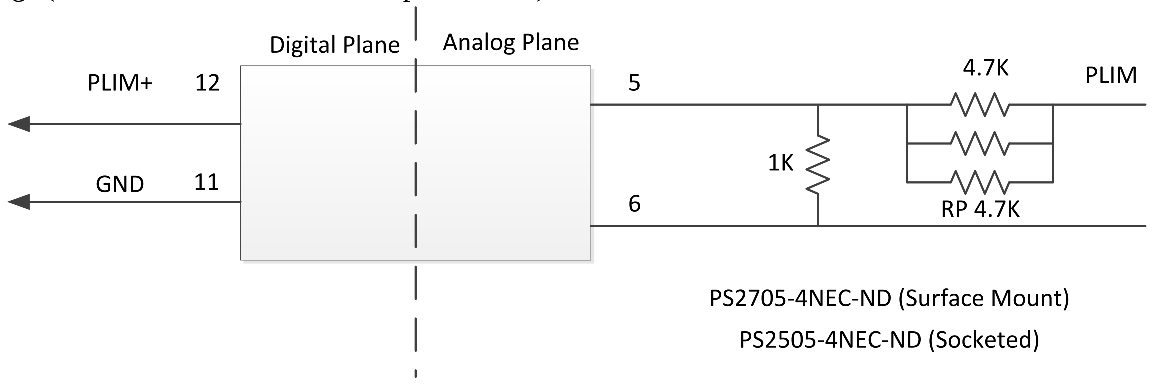


ACC-24E2 Limit and Flag Wiring

The ACC-24E2 allows the use of sinking or sourcing position limits and flags to the controller. The opto-isolator IC used is a PS2705-4NEC-ND quad photo-transistor output type. This IC allows the current to flow from return to flag (sourcing) or from flag to return (sinking).

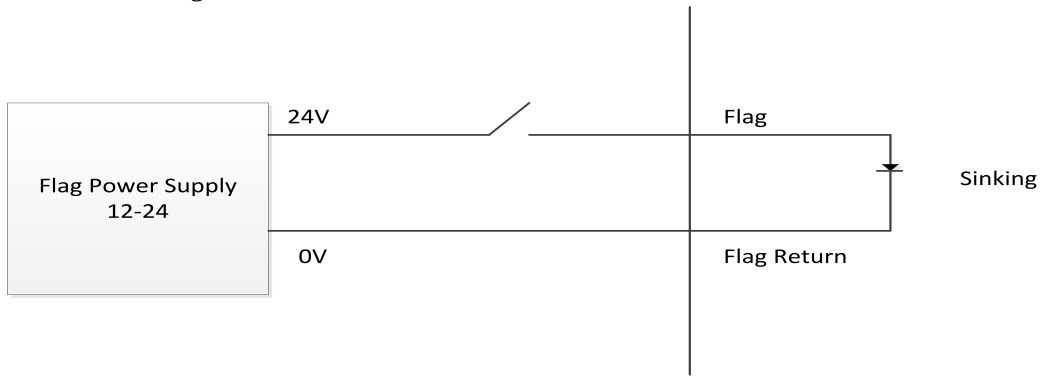


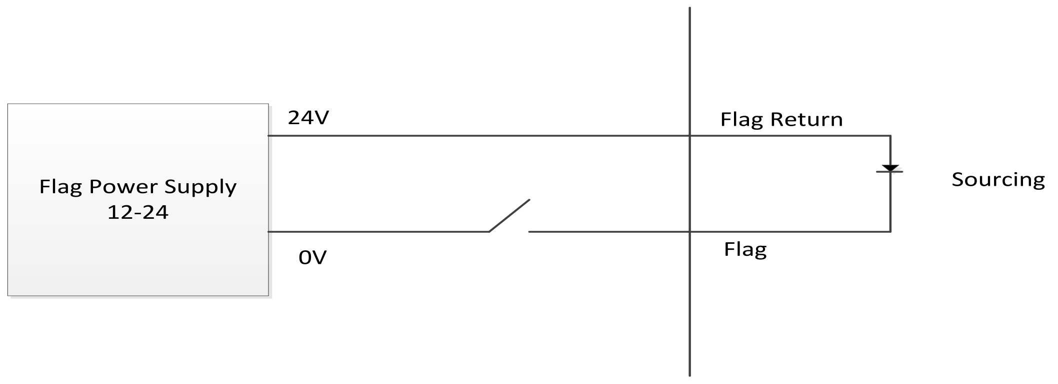
A sample of the positive limit circuit is shown below. The 4.7K resistor packs used will allow 12-24V flag inputs. If 0-5V flags are used, then a 1KΩ resistor pack (RP) can be placed in either RP45 or RP46 (refer to the Resistor Pack Configuration section of this manual). If these resistor packs are not added, all flags (±Limits, Home, User, and amplifier fault) will be referenced from 0-5V.



Connecting Limits/Flags to the ACC-24E2

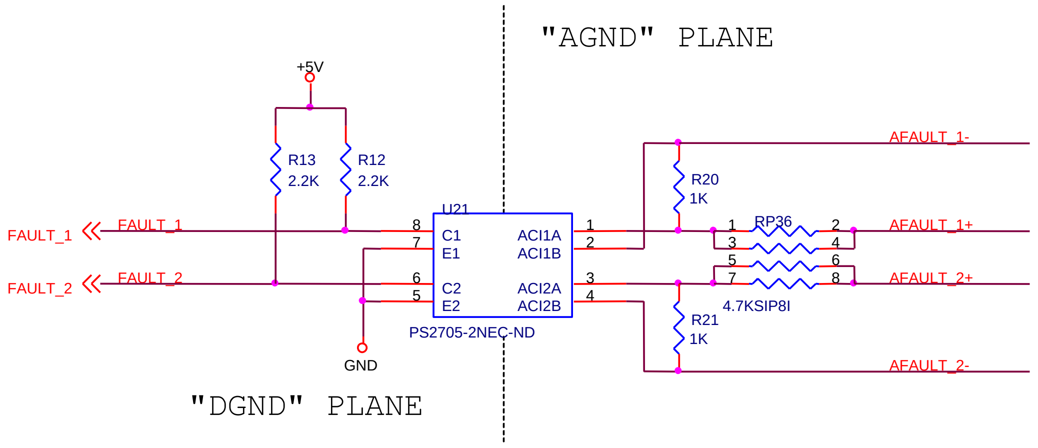
The following diagram illustrates the sinking and sourcing connections to an ACC-24E2. This example uses 12-24V flags.





Amplifier Fault Circuit

The amplifier fault circuit for the ACC-24E2A is functionally the same circuit as the limits and flag circuit.



For single-ended amplifier fault inputs, typically the AFAULT+ would be the actual signal input from the amplifier and the AFAULT- can be considered the reference.

Single Ended Amplifier Fault Inputs

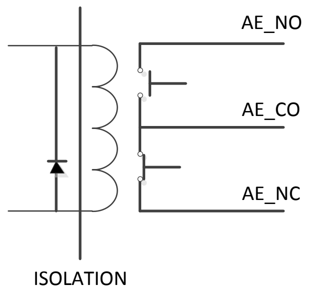
Amplifier Fault Signal	Ixx24 (Bit 23)	Fault+	Fault-	Fail Safe
Low True	1 (Recommended)	Tied to Fault Signal	Tied to Reference (GND) of Fault Signal	Yes
	0	Tied To Fault Signal	Tied to +V	No
High True	1	Tied To Fault Signal	Tied to +V	No
	0 (Recommended)	Tied To Fault Signal	Tied to Reference (GND) of Fault Signal	No

Note

- Mxx23 (Amplifier Fault Bit) is set to 0 if current is conducting from Fault+ to Fault- or vice versa and is set to 1 if current is not conducting.
- Low True Amplifier indicates that the fault signal is low, if amplifier fault triggers.
- High True Amplifier indicates that the fault signal is high, if amplifier fault triggers.
- Factory default is all
- Non recommended Ixx24s do not work with amplifiers that their low signal is an open circuit.

Amplifier Enable Circuit

Most amplifiers have an enable/disable input that permit s complete shutdown of the amplifier regardless of the voltage of the command signal. The ACC-24E2A AENA line is meant for this purpose. The amplifier enable signals of the ACC-24E2A is controlled by a relay with normal opened and normal closed dry contacts as shown in the diagram below:



Loss of Encoder Circuit

The encoder-loss detection circuitry works for differential incremental encoders only. In proper operation, the digital states of the complementary inputs for a channel (e.g. A and A/) always should be opposite: when one is high, the other is low. If for some reason, such as a cable connection coming undone, one or more of the signal lines is no longer driven, pull-up resistors on the input line pull and hold the signal high. The encoder-loss detection circuitry uses exclusive-or (XOR) gates on each complementary pair to detect whether the signals are in the same or opposite states. These results are combined to produce a single encoder-loss status bit that the processor can read.

This technique requires that both signal lines of the pair have pull-up resistors. Note that this is not the default configuration of a PMAC as it is shipped. The complementary lines (A/ and B/) are pulled to 2.5V in a voltage-divider configuration as shipped to be able to accept both single-ended and normal differential inputs. This must be changed to a pull-up configuration which involves reversing a socketed resistor pack on the ACC-24E2A.

ACC-24E2A Encoder Loss Detection with UMAC Turbo CPU

Channel	Resistor Pack	Status Bit Address (Even-Numbered Servo IC)*	Status Bit Address (Odd-Numbered Servo IC)*	Status Bit Name	Bit Error State
1	RP22	Y:$07xF08,5	Y:$07xF0C,5	QL_1-	0
2	RP24	Y:$07xF09,5	Y:$07xF0D,5	QL_2-	0
3	RP22**	Y:$07xF0A,5	Y:$07xF0E,5	QL_3-	0
4	RP24**	Y:$07xF0B,5	Y:$07xF0F,5	QL_4-	0

Note

- The x digit in this hex address matches the value (8, 9, A, or B) in the fourth digit from the right in the board's own base address (e.g. $079200). If alternate addressing of Servo ICs is used (e.g. Servo IC 2*), add $20 to these addresses.
- These resistor packs are on the Option 1A piggyback board (if present) of the module, not on the baseboard.

ACC-24E2A Encoder Loss Detection with UMAC MACRO CPU

Channel	Resistor Pack	Status Bit Address (First-Servo IC)*	Status Bit Address (Second Servo IC)*	Status Bit "Name"	Bit Error State
1	RP22	Y:$B8C8,5	Y:$B8EC,5	QL_1-	0
2	RP24	Y:$B8C9,5	Y:$B8ED,5	QL_2-	0
3	RP22	Y:$B8CA,5	Y:$B8EE,5	QL_3-	0
4	RP24	Y:$B8CB,5	Y:$B8EF,5	QL_4-	0

Note

- First Servo IC has base address $C040; second Servo IC has base address $C060.
- These resistor packs are on the Option 1A piggyback board (if present) of the module, not on the baseboard.

ACC-24E2A Encoder Loss Detection with Power PMAC CPU

Channel	Resistor Pack	Status Bit Address (Even-Numbered Servo IC)*	Status Bit Address (Odd-Numbered Servo IC)*	Status Bit Name	Bit Error State
1	RP22	u.io:$Dxx040.13.1	u.io:$Dxx050.13.1	QL_1-	0
2	RP24	u.io:$Dxx044.13.1	u.io:$Dxx054.13.1	QL_2-	0
3	RP22	u.io:$Dxx048.13.1	u.io:$Dxx058.13.1	QL_3-	0
4	RP24	u.io:$Dxx04C.13.1	u.io:$Dxx05C.13.1	QL_4-	0

Note

- The xx digit in this hex address matches the value (00, 08, 10, or 18) in the fourth and fifth digits from the right in the board's own base address (e.g. $608000). If alternate addressing of Servo ICs is used (e.g. Servo IC 2*), add $100 to these addresses.
- These resistor packs are on the Option 1A piggyback board (if present) of the module, not on the baseboard.

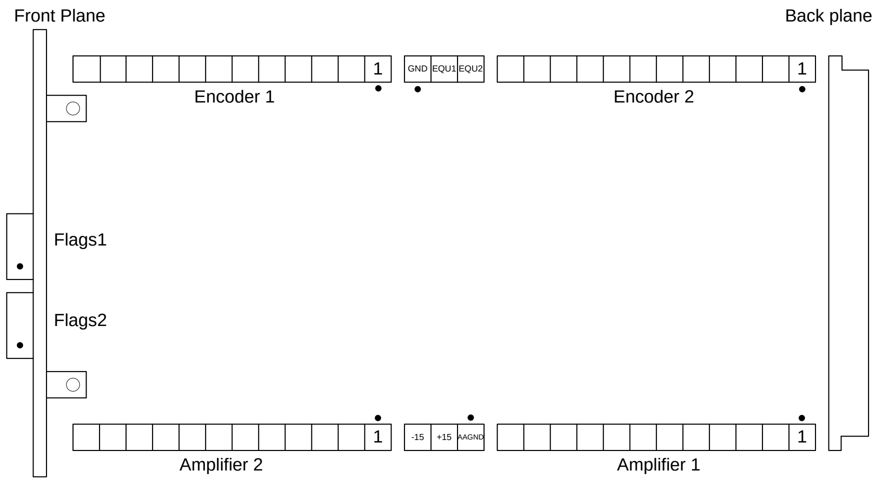
Position Compare Port Driver IC

As with the other PMAC controllers, the UMAC has the high-speed position compare outputs allowing the firing of an output based on position. This circuit will fire within 100 nsec of reaching the desired position. The position compare output port on ACC-24E2x has driver IC at component U27.
The following table lists the properties of each driver IC:

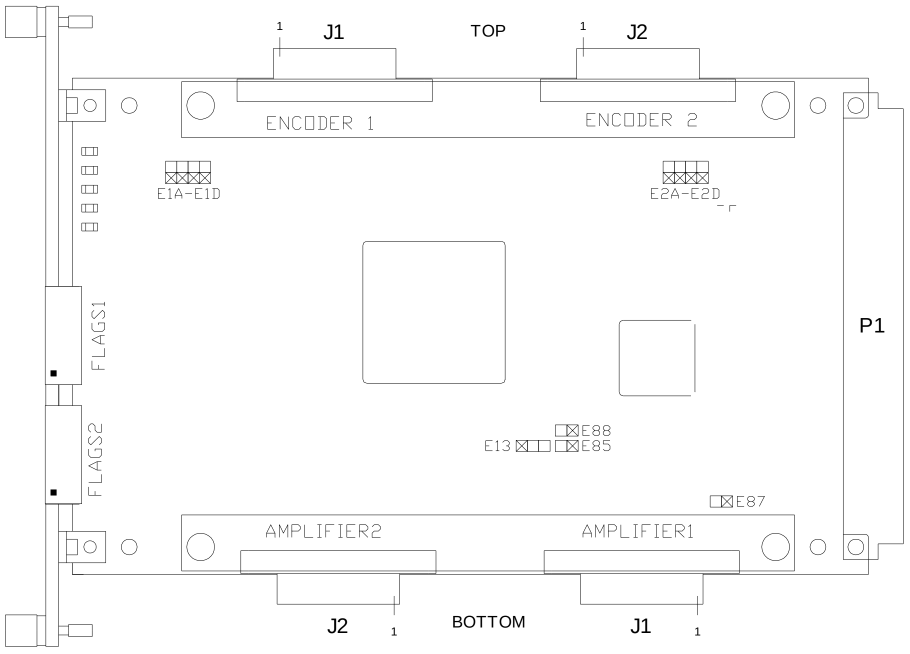
Part	# of Pins	Max Voltage and Current	Output Type	Max Frequency
DS75451N	8	5V, 10 mA	Totem-Pole	5 MHz

CONNECTIONS

ACC-24E2A Board Layout - Terminal Block Option



ACC-24E2A Board Layout - DB15 Option



Mating Connectors

Terminal Block Connectors

Name	Manufacturer	Pins	Type	Details
TB1- Top	Phoenix Contact	12	FRONT-MC1,5/12-ST3,81	Encoder 1 Inputs
TB2- Top	Phoenix Contact	12	FRONT-MC1,5/12-ST3,81	Encoder 2 Inputs
TB3- Top	Phoenix Contact	3	FRONT-MC1,5/3-ST3,81	Compare Outputs
TB1- Bottom	Phoenix Contact	12	FRONT-MC1,5/12-ST3,81	Amplifier 1 Outputs
TB2- Bottom	Phoenix Contact	12	FRONT-MC1,5/12-ST3,81	Amplifier 2 Outputs
TB3- Bottom	Phoenix Contact	3	FRONT-MC1,5/3-ST3,81	External Power Inputs
TB1- Front	Phoenix Contact	5	FRONT-MC1,5/5-ST3,81	Channel 1 Flags
TB2 Front	Phoenix Contact	5	FRONT-MC1,5/5-ST3,81	Channel 2 Flags

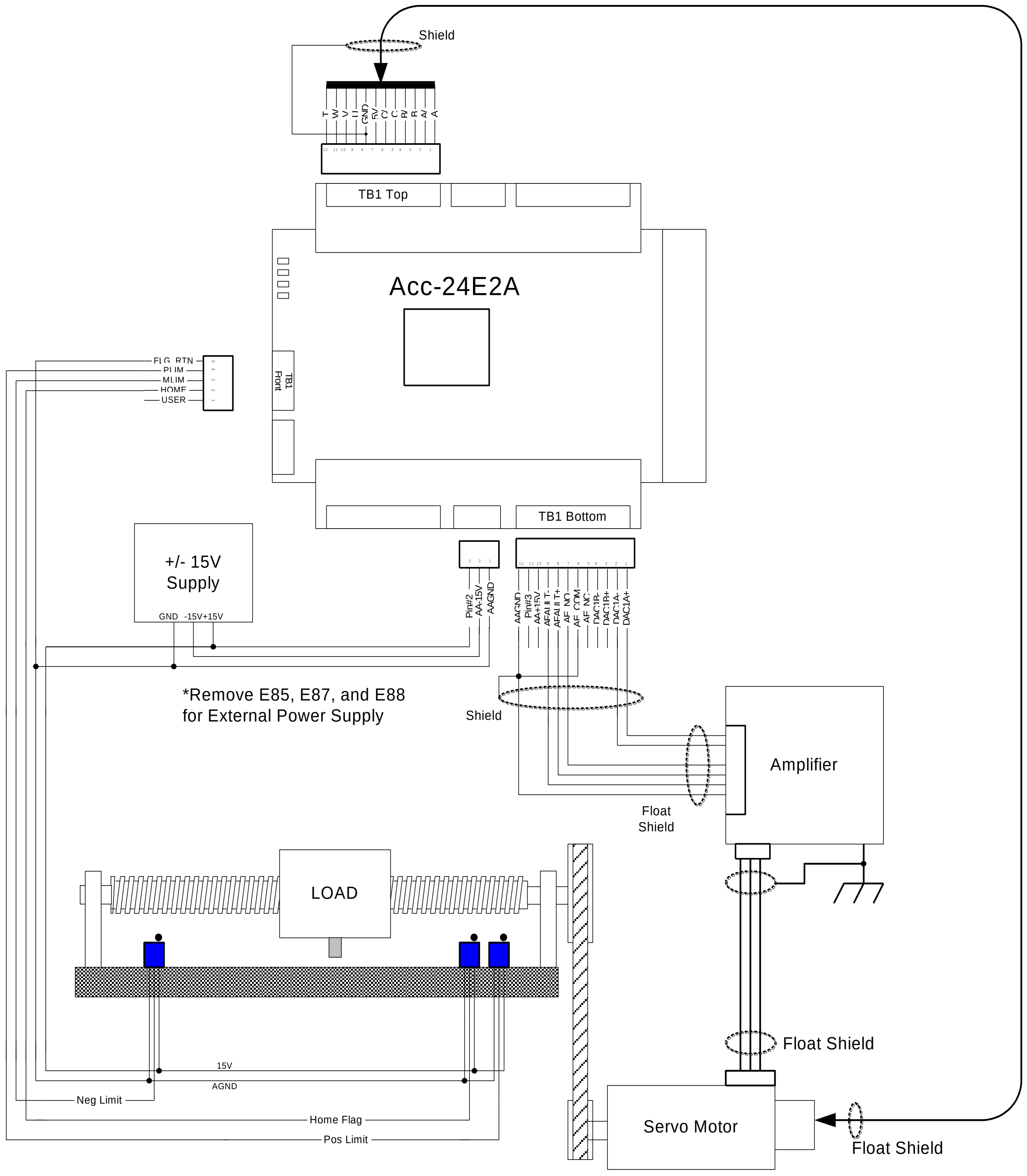
DB15 Connector Option

Name	Manufacturer	Pins	Type	Details
J1- Top	AMP	15	AMP 745072-2	Encoder 1 Inputs and Compare Outputs
J2- Top	AMP	15	AMP 745072-2	Encoder 2 Inputs and Compare Outputs
J1- Bottom	AMP	15	AMP 745072-2	Amplifier 1 Outputs and Analog Power Inputs
J2- Bottom	AMP	15	AMP 745072-2	Amplifier 2 Outputs and Analog Power Inputs

Indicators

LED	Color	Description
D5	Amber	Amplifier 1 Enabled
D6	Amber	Amplifier 2 enabled
D10	Green	Encoder 1 Power OK
D11	Green	Encoder 2 Power OK
D17	Green	Analog Power Good

Overall Wiring Diagram



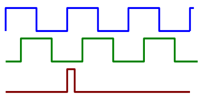
This is a general example of a system with sourcing flags and normally open amplifier enable output from the Acc-24E2A. For opto-isolation an external power supply is used and E85, E87, and E88 have been removed from the Acc-24E2A.

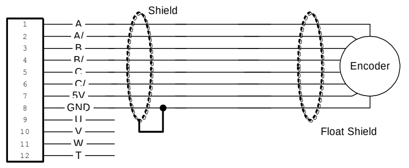
Sample Wiring Diagrams

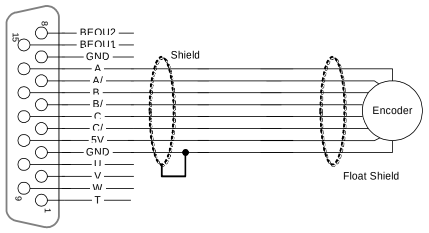
This section has typical wiring diagrams for the TTL level inputs, flags and limits, DAC and PFM outputs, amplifier enable, and amplifier fault.

TTL Level Inputs and Outputs

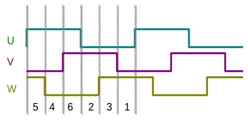
Quadrature Encoders

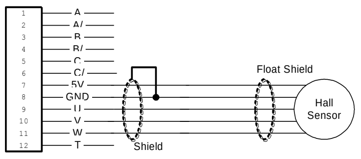


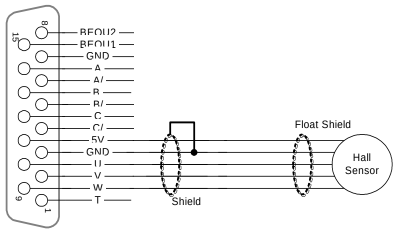




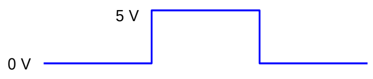
TTL Hall Effect Sensors

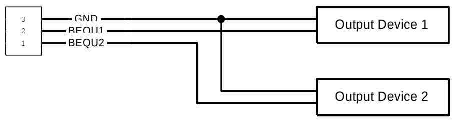


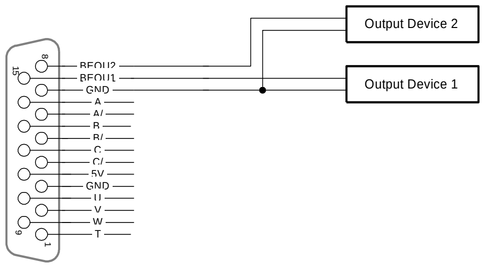




Position Compare Outputs

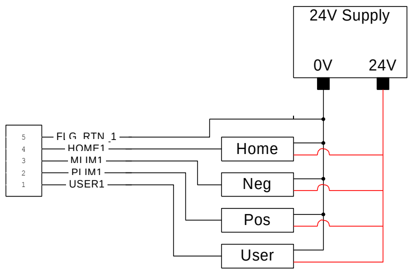




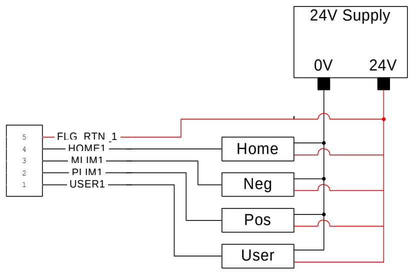


Position Limits, Home Flag, and User Flag

Acc-24E2A Sourcing Flags



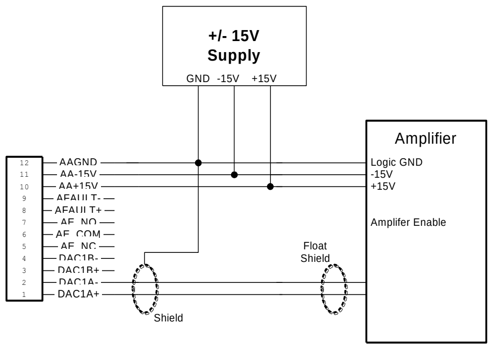
Acc-24E2A Sinking Flags

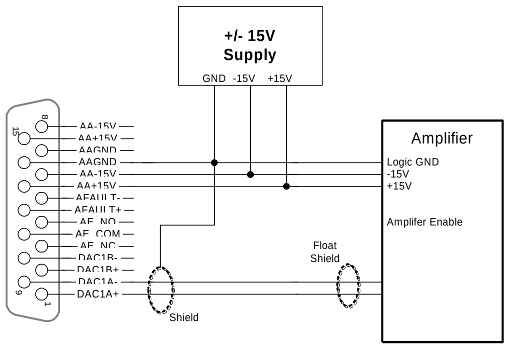


DAC Outputs

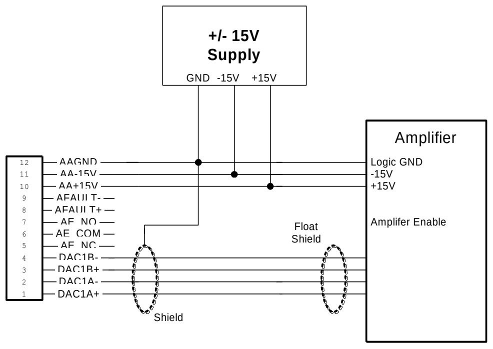
Sample diagrams shown below utilize a separate ±15V power supply for opto-isolation. E85, E87, and E88 are removed from ACC-24E2A.

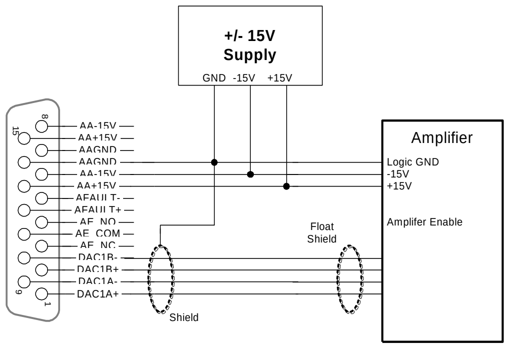
Acc-24E2A DAC-Torque/Velocity Mode





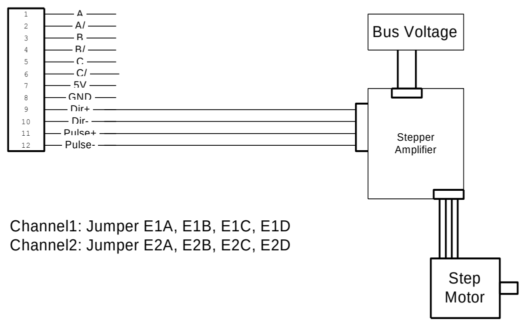
Acc-24E2A DAC - Sinusoidal Commutation Mode



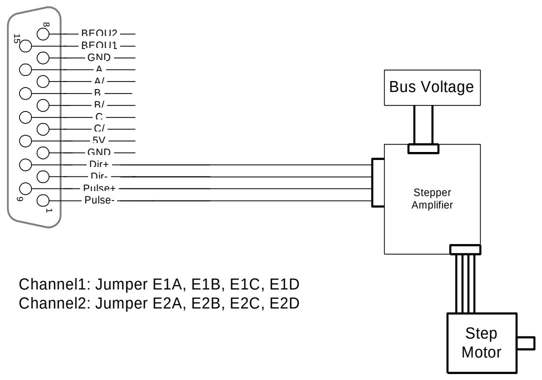


Stepper Drive, Pulse and Direction Outputs (TTL Level)

Acc-24E2A PFM-Stepper Output



Channel1: Jumper E1A, E1B, E1C, E1D
Channel2: Jumper E2A, E2B, E2C, E2D

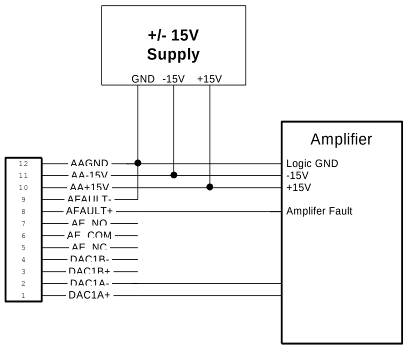


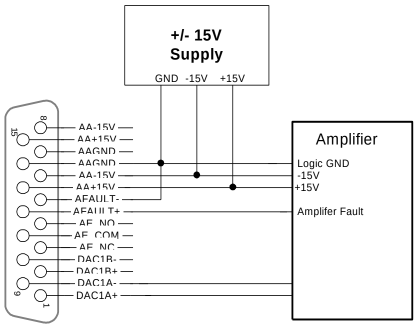
Channel1: Jumper E1A, E1B, E1C, E1D
Channel2: Jumper E2A, E2B, E2C, E2D

Amplifier Fault Inputs

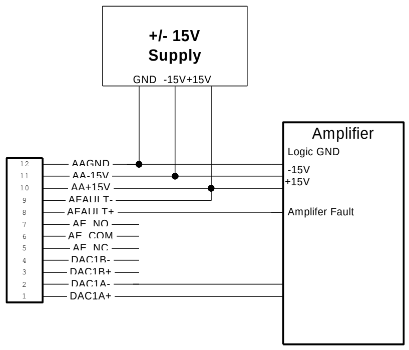
Sample diagrams shown below utilize a separate ±15V power supply for opto-isolation. E85, E87, and E88 are removed from ACC-24E2A.

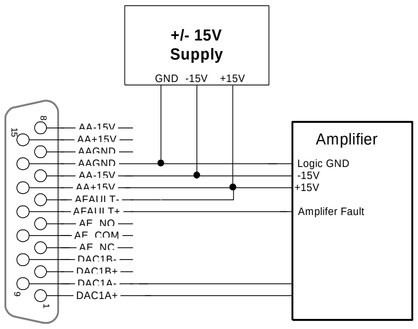
Acc-24E2A Sinking Amplifier Fault





Acc-24E2A Sourcing Amplifier Fault

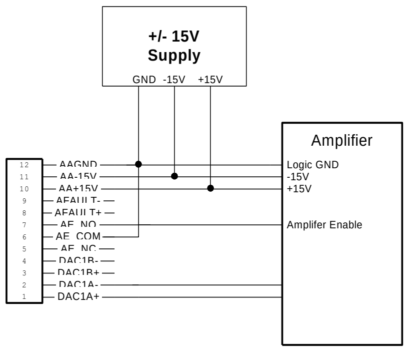


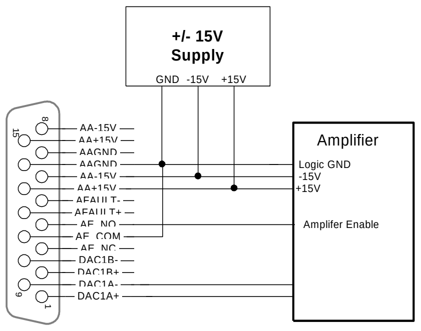


Amplifier Enable Outputs

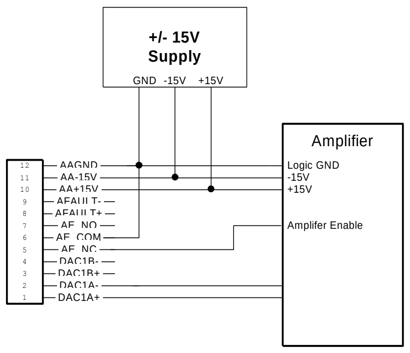
Sample diagrams shown below utilize a separate ±15V power supply for opto-isolation. E85, E87, and E88 are removed from ACC-24E2A.

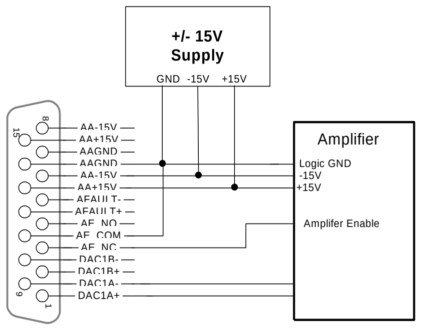
Acc-24E2A Normally Open Amplifier Enable





Acc-24E2A Normally Closed Amplifier Enable





SOFTWARE SETUP

Using ACC-24E2A with UMAC Turbo

Servo IC Numbering

The Servo IC I-variables in ACC-24E2A are addressed with I7m00-I7m99 for servo IC m and channel n (where m=2 to 9).The number m of the Servo IC on the ACC-24E2A board is dependent on the addressing of the board with DIP switches S1-1, S1-3, and S1-4, which place the board as the first through eight external devices:

• First ACC-24E2 with Option 1:	Servo IC 2 (channels 1-4)
• Second ACC-24E2 with Option 1	Servo IC 3 (channels 5-8)
• Third ACC-24E2 with Option 1:	Servo IC 4 (channels 9-12)
• Fourth ACC-24E2 with Option 1	Servo IC 5 (channels 13-16)
• Fifth ACC-24E2 with Option 1:	Servo IC 6 (channels 17-20)
• Sixth ACC-24E2 with Option 1	Servo IC 7 (channels 21-24)
• Seventh ACC-24E2 with Option 1:	Servo IC 8 (channels 25-28)
• Eighth ACC-24E2 with Option 1	Servo IC 9 (channels 29-32)

The Standard Servo IC on an ACC-24E2A occupies Channels 1-2 on the board, using connectors associated with channels 1 and 2. The Option 1 on an ACC-24E2A occupies Channels 3-4 on the board, using connectors associated with channels 3 and 4.
Example: The Standard Servo IC on the first ACC-24E2 is Servo IC 2 to Turbo PMAC and is configured by variables I7200 – I7299.

Servo Channel Numbering

Each Servo IC has four channels of servo interface circuitry. The tens digit n of the I-variable configuring the IC represents the channel number on the IC (n = 1 to 4). For example, Channel 1 of the Standard Servo IC on the first ACC-24E2A is configured by variables I7210 – I7219. These channel-specific I-variables are represented generically as I7mn0 – I7mn9, where m represents the Servo IC number (2-9) and n represents the IC channel number (1-4).
The Channels 1 – 4 on the Standard Servo IC of an ACC-24E2A correspond to Channels 1-4, respectively, on the ACC-24E2A board itself.
I-variables in the I7000s for which the tens digit is 0 (Channel 0) affect all four channels of the PMAC2-style Servo IC on the ACC-24E2A. These multi-channel I-variables are represented generically as I7m00 – I7m09.

Multi-Channel I-Variables

Several multi-channel I-variables must be set up for proper operation of the ACC-24E2A in a Turbo PMAC system. The most important are:
I7m07: Servo IC m Phase/Servo Clock Direction
This variable should be set to 0 on the ACC-24E2A generating the clocks.

I7m00: Servo IC m MaxPhase/PWM Frequency Control
Typically, this will be set to the same value as the variable that controls the system clocks: I7200 on a UMAC Turbo PMAC2. If a different PWM frequency is desired then the following constraint should be observed in setting this variable:

$$\frac{2 * PWMFreq(kHz)}{PhaseFreq} = \{ Integer \}$$

I7m01: Servo IC m Phase Clock Frequency Control
Even though the IC is receiving an external phase clock (see I7m07, above), usually it is best to create the same internal phase clock frequency in the Servo IC. This yields the following constraint:

$$I7m00 * (I7m01 + 1) = I7200 * (I7201 + 1) \qquad \text{\{UMAC Turbo\}}$$

Solving for I7m01, we get

$$I7m01 = \frac{I7200 * (I7201 + 1)}{I7m00} - 1 \qquad \text{\{UMAC Turbo\}}$$

If I7m00 is the same as I7200, I7m01 will be the same as I7201.
I7m02: Servo IC m Servo Clock Frequency Control
Even though the IC is receiving an external servo clock (see I7m07, above), usually it is best to create the same internal servo clock frequency in the Servo IC. This means that I7m02 for the IC should be set the same as I7202 on a UMAC Turbo.
I7m03: Servo IC m Hardware Clock Frequency Control
The hardware clock frequencies for the Servo IC should be set according to the devices attached to it. There is no reason that these frequencies have to be the same between ICs. There is seldom a reason to change this value from the default.

Single-Channel I-Variables

The single-channel setup I-variables for Channel n of Servo IC m work the same on an ACC-24E2 as they do on a Turbo PMAC2 itself. Each Servo IC has four channels n, numbered 1 to 4. For the first (standard) Servo IC on the ACC-24E2, the channel numbers 1 – 4 on the Servo IC are the same as the channel numbers 1 – 4 on the board. The most important variables are:
I7mn0: Servo IC m Channel n Encoder Decode Control
Typically, I7mn0 is set to 3 or 7 for x4 quadrature decode, depending on which way is up. If the channel is used for open-loop stepper drive, I7mn0 is set to 8 to accept internal pulse-and-direction, or to 0 to accept external pulse-and-direction (e.g. from an ACC-8S). It is set to 12 if the channel is used for MLDT feedback.
I7mn2: Servo IC m Channel n Capture Control
I7mn2 determines whether the encoder index channel, an input flag, or both, are used for the capture of the encoder position.
I7mn3: Servo IC m Channel n Capture Flag Select
I7mn3 determines which input flag is used for encoder capture, if one is used.
I7mn6: Servo IC m Channel n Output Mode Select
I7mn6 determines whether the A and B outputs are DAC or PWM, and whether the C output is PFM (pulse-and-direction) or PWM. Typically, it is set to 0, for 3-phase PWM, or to 3 for DACs and PFM.

Encoder Conversion Table I-Variables

To use feedback or master position data from an ACC-24E2A, add entries to the encoder conversion table (ECT) using I-variables I8000 – I8191 to address and process this data. The default conversion table in the Turbo PMAC does not contain these entries; it only contains entries for the eight channels on board the Turbo PMAC. The ECT entries for ACC-24E2 incremental encoder channels are shown in the following table:

Servo IC #	Chan. 1	Chan. 2	Chan. 3	Chan. 4	Notes
2	$m78200	$m78208	$m78210	$m78218	First ACC-24E2x Channel n Encoder Set
3	$m78300	$m78308	$m78310	$m78318	Second ACC-24E2x Channel n Encoder Set
4	$m79200	$m79208	$m79210	$m79218	Third ACC-24E2x Channel n Encoder Set
5	$m79300	$m79308	$m79310	$m79318	Fourth ACC-24E2x Channel n Encoder Set
6	$m7A200	$m7A208	$m7A210	$m7A218	Fifth ACC-24E2x Channel n Encoder Set
7	$m7A300	$m7A308	$m7A310	$m7A318	Sixth ACC-24E2x Channel n Encoder Set
8	$m7B200	$m7B208	$m7B210	$m7B218	Seventh ACC-24E2x Channel n Encoder Set
9	$m7B300	$m7B308	$m7B310	$m7B318	Eighth ACC-24E2x Channel n Encoder Set
The first hexadecimal digit in the entry, represented by m in the table, is a 0 for the most common 1/T timer-based extension of digital incremental encoders; it is an 8 for the parallel-data extension of analog incremental encoders; it is a C for no extension of an incremental encoder.					

Motor Addressing I-Variables

For a Turbo PMAC motor to use the servo interface circuitry of the ACC-24E2, several of the addressing I-variables for the motor must contain the addresses of registers in the ACC-24E2, or the addresses of encoder conversion table registers containing data processed from the ACC-24E2. These I-variables can include:

Ixx02: Motor xx Command Output Address

Ixx02 tells Turbo PMAC where to write its command outputs for Motor xx. If ACC-24E2 is to create the command signals, Ixx02 must contain the address of the register.

The following table shows the address of the A output register for each channel of each ACC-24E2. These addresses can be used for single analog outputs, double analog outputs, or direct PWM outputs.

Servo IC #	Chan. 1	Chan. 2	Chan. 3	Chan. 4	Notes
2	$078202	$07820A	$078212	$07821A	First ACC-24E2x Channel n DAC/PWMnA
3	$078302	$07830A	$078312	$07831A	Second ACC-24E2x Channel n DAC/PWMnA
4	$079202	$07920A	$079212	$07921A	Third ACC-24E2x Channel n DAC/PWMnA
5	$079302	$07930A	$079312	$07931A	Fourth ACC-24E2x Channel n DAC/PWMnA
6	$07A202	$07A20A	$07A212	$07A21A	Fifth ACC-24E2x Channel n DAC/PWMnA
7	$07A302	$07A30A	$07A312	$07A31A	Sixth ACC-24E2x Channel n DAC/PWMnA
8	$07B202	$07B20A	$07B212	$07B21A	Seventh ACC-24E2x Channel n DAC/PWMnA
9	$07B302	$07B30A	$07B312	$07B31A	Eighth ACC-24E2x Channel n DAC/PWMnA

If the C output register for a given ACC-24E2 and channel is used (primarily for pulse and direction output), simply add 2 to the address shown in the above table. For example, on the first ACC-24E2, output register 1C is at address $078204.

Ixx03: Motor xx Position-Loop Feedback Address

Ixx04: Motor xx Velocity-Loop Feedback Address

Ixx05: Motor xx Master Position Address
Usually, the Ixx03, Ixx04, and Ixx05 variables contain the address of a processed position value in the encoder conversion table, even when the raw data comes from the ACC-24E2A. The first line of the encoder conversion table is at address $003501; the last line is at address $0035C0.
Ixx10: Motor xx Power-On Position Address
Ixx10 tells the Turbo PMAC where to read absolute power-on position, if any. Typically, the only times Ixx10 will contain the address of an ACC-24E2A register is if the position is obtained from an A/D converter on an ACC-28B connected through the ACC-24E2A, or if it is obtained from an MLDT (e.g. Temposonics^{TM}) sensor excited directly from an ACC-24E2A.
The following table shows the possible values of Ixx10 for MLDT timer registers:
Ixx10 for ACC-24E2 MLDT Timer Registers (Ixx95=$170000)

Servo IC #	Chan. 1	Chan. 2	Chan. 3	Chan. 4	Notes
2	$078200	$078208	$078210	$078218	First ACC-24E2x Channel n Timer
3	$078300	$078308	$078310	$078318	Second ACC-24E2x Channel n Timer
4	$079200	$079208	$079210	$079218	Third ACC-24E2x Channel n Timer
5	$079300	$079308	$079310	$079318	Fourth ACC-24E2x Channel n Timer
6	$07A200	$07A208	$07A210	$07A218	Fifth ACC-24E2x Channel n Timer
7	$07A300	$07A308	$07A310	$07A318	Sixth ACC-24E2x Channel n Timer
8	$07B200	$07B208	$07B210	$07B218	Seventh ACC-24E2x Channel n Timer
9	$07B300	$07B308	$07B310	$07B318	Eighth ACC-24E2x Channel n Timer

Ixx24: Motor xx Flag Mode
Ixx24 defines how to read and use the flags for Motor xx that are in the register specified by Ixx25. Ixx24 is a set of independent control bits. There are two bits that must be set correctly to use a flag set on an ACC-24E2.
Bit 0 of Ixx24 must be set to 1 to tell the Turbo PMAC that this flag set is in a Type 1 PMAC2-style Servo IC. Bit 18 of Ixx24 must be set to 0 to tell the Turbo PMAC that this flag set is not transmitted over a MACRO ring. Other bits of Ixx24 may be set as desired for a particular application.

Note

See Turbo PMAC Software Reference Manual for description of Ixx24.

Ixx25: Motor xx Flag Address
Ixx25 tells Turbo PMAC where to access its flag data for Motor xx. If ACC-24E2 is interfaced to the flags, Ixx25 must contain the address of the flag register in ACC-24E2. The following table shows the address of the flag register for each channel of each ACC-24E2.

Servo IC #	Chan. 1	Chan. 2	Chan. 3	Chan. 4	Notes
2	$078200	$078208	$078210	$078218	First ACC-24E2x Channel n Flag Set
3	$078300	$078308	$078310	$078318	Second ACC-24E2x Channel n Flag Set
4	$079200	$079208	$079210	$079218	Third ACC-24E2x Channel n Flag Set
5	$079300	$079308	$079310	$079318	Fourth ACC-24E2x Channel n Flag Set
6	$07A200	$07A208	$07A210	$07A218	Fifth ACC-24E2x Channel n Flag Set
7	$07A300	$07A308	$07A310	$07A318	Sixth ACC-24E2x Channel n Flag Set
8	$07B200	$07B208	$07B210	$07B218	Seventh ACC-24E2x Channel n Flag Set
9	$07B300	$07B308	$07B310	$07B318	Eighth ACC-24E2x Channel n Flag Set

TURBO UMAC Example Setups

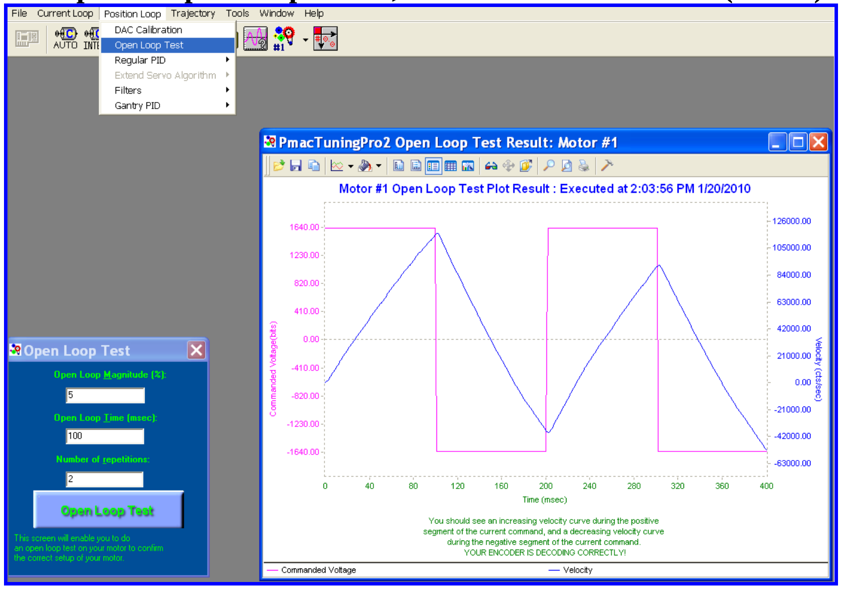
The following section shows how to quickly setup the key variables for a DAC output system and for a combination torque mode (DAC) and stepper motor (PFM) system.
For these examples, the factory defaults for the other variables will allow the command of DAC outputs and PFM outputs with a low true Amplifier Fault and ±Limits plugged in. If this is not the case then Ixx24 will have to be modified. The PID gains will also have to be modified for optimum closed loop control.

Example A: Setup the first 4 motors as DAC outputs on first ACC-24E2A in UMAC Turbo:

```
I100,4,100=1                  ; Motor 1-4 enabled
I102=$78202                   ; Command output  to CH1A address (PMAC default) for DAC
I202=$7820A                   ; Command output  to CH2A address (PMAC default) for DAC
I302=$78212                   ; Command output  to CH3A address (PMAC default) for DAC
I402=$7821A                   ; Command output  to CH4A address (PMAC default) for DAC
I7216,4,10=1                  ; Servo IC 2 CHn 1-4 output mode select
I124,4,100=$800001            ; Low True Amplifier Fault and enable hard limits
I8000=$78200                  ; Encoder Convertion Table entries for 1/T extension (PMAC Default)
I8001=$78208
I8002=$78210
I8003=$78218
I103,2,1=$3501                ; Pos/Vel feedback pointing to first 4 ECT entries (PMAC Default)
I203,2,1=$3502
I303,2,1=$3503
I403,2,1=$3504
```

After this initial setup, we can use tuning software to perform open loop test. The result of open loop test should look like this:

Acceptable Open-Loop Result, Correct Encoder Decode (I7mn0)



Open Loop Magnitude (%): 5

Open Loop Time (msec): 100

Number of repetitions: 2

This screen will enable you to do an open loop test on your motor to confirm the correct setup of your motor.

You should see an increasing velocity curve during the positive segment of the current command, and a decreasing velocity curve during the negative segment of the current command.
YOUR ENCODER IS DECODING CORRECTLY!

After a successful open loop test, PID tuning should be performed by the user.

Example B: Setup the first 4 motors as Step and Direction outputs on first ACC-24E2A in UMAC Turbo (Stepper Motor):

```
I100,4,100=1                  ; Motor 1-4 enabled
I102=$78204                   ; Command output to CH1C address for PFM
I202=$7820C                   ; Command output to CH1C address for PFM
I302=$78214                   ; Command output to CH1C address for PFM
I402=$7821C                   ; Command output to CH1C address for PFM

I7216,4,10=1                  ; Servo IC 2 CHn 1-4 output mode select
I124,4,100=$800001            ; Low True Amplifier Fault and enable hard limits
I8000=$78200                  ; Encoder Convertion Table entries for 1/T extension
I8001=$78208
I8002=$78210
I8003=$78218
I103,2,1=$3501                ; Position and Velocity feedback pointing to first 4 ECT entries
I203,2,1=$3502
I303,2,1=$3503
I403,2,1=$3504
```

Using ACC-24E2A with ULTRALITE/MACRO STATION

Clocks and Strobe Word

Note

All MACRO Communication variables on the Master such as I6840/I6890/I6940/I6990, I6841/I6891/I6941/I6991, I78,I70~I77,I80,I81,I82 and on the station such as MI996,MI995 are assumed to be set correctly before configuring ACC-24E2A. For more information on how to set these variables look the User Manual for your Master (UltraLite/UMAC with ACC-5E) and MACRO 16 CPU.

There are several choices when it comes to the software setup for the MACRO Station. At the MACRO Station the ring frequency must be set up with MSn,MI992. The ACC-24E2A will have its MaxPhase Clock Frequency variables (MSn,MI900 and MSn,MI906) set to the same value as MSn,MI992 to ensure synchronous data exchange. The Delta Tau Setup software for either the standard PMAC2 Ultralite or Turbo PMAC2 Ultralite will set up all of these important MI-Variables at the MACRO Station.
The ACC-24E2A uses an 18-bit DAC and the DAC Strobe word (MSn,MI905 and MSn,MI909) must be setup for 18-bits to ensure proper operation of the DACs.
MS{anynode},MI992 - Ring Frequency Control
MS{anynode},MI900 - Channels 1-4 Frequency Control
MS{anynode},MI905 - DAC 1-4 Strobe Word
MS{anynode},MI906 - Channels 5-8 Frequency Control
MS{anynode},MI909 - DAC 5-8 Strobe Word

Note

The released MACRO Station firmware version 1.14 will set the DAC strobe variables automatically. In pre-release versions of the 1.14 firmware, the DAC strobe word must be set manually to $7FFFC0 for proper 18-bit DAC operation.

Node-Specific Gate Array MI-Variables

MI-variables MI910 through MI919 on the MACRO station control the hardware setup of the hardware interface channel on the station associated with a MACRO node. The matching of hardware interface channels to MACRO nodes is determined by the setting of the SW1 rotary switch on the CPU/Interface Board of the MACRO station.
These variables are accessed using the MS station auxiliary read and write commands. The number immediately after the MS specifies the node number, and therefore the channel number mapped to that node by the SW1 setting.

Encoder/Timer n Decode Control (MSn,MI910)

MI910 controls how the input signal for the encoder mapped to the specified node is decoded into counts. As such, this defines the sign and magnitude of a count. The following settings may be used to decode an input signal:

0:	Pulse and direction CW
1:	x1 quadrature decode CW
2:	x2 quadrature decode CW
3:	x4 quadrature decode CW
4:	Pulse and direction CCW
5:	x1 quadrature decode CCW
6:	x2 quadrature decode CCW
7:	x4 quadrature decode CCW
8:	Internal pulse and direction
9:	Not used
10:	Not used
11:	Not used
12:	MLDT pulse timer control (internal pulse resets timer; external pulse latches timer)
13:	Not used
14:	Not used
15:	Not used

In any of the quadrature decode modes, PMAC is expecting two input waveforms on CHAn and CHBn, each with approximately 50% duty cycle, and approximately one-quarter of a cycle out of phase with each other. Times-one (x1) decode provides one count per cycle; x2 provides two counts per cycle; and x4 provides four counts per cycle. Select x4 decode to get maximum resolution.

The clockwise (CW) and counter clockwise (CCW) options simply control which direction counts up. If it is the wrong direction sense, simply change to the other option (e.g., from 7 to 3 or vice versa).

WARNING

If the direction sense of an encoder with a properly working servo is changed without also changing the direction sense of the output, destabilizing positive feedback to the servo and a dangerous runaway condition will result.

In the pulse-and-direction decode modes, PMAC is expecting the pulse train on CHAn and the direction (sign) signal on CHBn. If the signal is unidirectional, the CHBn line can be allowed to pull up to a high state, or it can be hardwired to a high or low state.

If MI910 is set to 8, the decoder inputs the pulse and direction signal generated by Channel n's pulse frequency modulator (PFM) output circuitry. This permits the Compact MACRO Station to create a phantom closed loop when driving an open-loop stepper system. No jumpers or cables are needed to do this; the connection is entirely within the ASIC. The counter polarity matches the PFM output polarity automatically.

If MI910 is set to 12, the timer circuitry is set up to read magnetostrictive linear displacement transducers (MLDTs) such as Temposonics^{TM}. In this mode, the timer is cleared when the PFM circuitry sends out the excitation pulse to the sensor on PULSEn, and it is latched into the memory-mapped register when the excitation pulse is received on CHAn.

Flag Capture Control (MSn,MI911-MI913)

The flag capture registers must also be set up at the MACRO Station for proper homing, encoder capturing, and setting compare outputs.

MI911 determines which encoder input the position compare circuitry for the machine interface channel mapped to the specified node uses.

`MSn,MI911=0`	Use channel n encoder counter for position compare function
`MSn,MI911=1`	Use first encoder counter on IC (encoder 1 for channels 1 to 4; encoder 5 for channels 5 to 8) for position compare function

When MI911 is set to 0, the channel's position compare register is tied to the channel's own encoder counter, and the position compare signal appears only on the EQUn output.

When MI911 is set to 1, the channel's position compare register is tied to the first encoder counter on the ASIC (Encoder 1 for channels 1-4, Encoder 5 for channels 5-8, or Encoder 9 for channels 9-10) and the position compare signal appears both on EQUn and combined into the EQU output for the first channel on the IC (EQU1 or EQU5); executed as a logical OR.

MI911 for the first channel on an ASIC performs no effective function, so is always 1. It cannot be set to 0.

MI912 determines which signal or combination of signals, and which polarity, triggers a position capture of the counter for the encoder mapped to the specified node. If a flag input (home, limit, or user) is used, MI913 for the node determines which flag. Proper setup of this variable is essential for a successful home search, which depends on the position-capture function. The following settings may be used:

0:	Capture under software control (armed)
1:	Capture on Index (CHCn) high
2:	Capture on Flag high
3:	Capture on (Index high AND Flag high)
4:	Capture under software control (latched)
5:	Capture on Index (CHCn) low
6:	Capture on Flag high
7:	Capture on (Index low AND Flag high)
8:	Capture under software control (armed)
9:	Capture on Index (CHCn) high
10:	Capture on Flag low
11:	Capture on (Index high AND Flag low)
12:	Capture under software control (latched)
13:	Capture on Index (CHCn) low
14:	Capture on Flag low
15:	Capture on (Index low AND Flag low)

The trigger is armed when the position capture register is read. After this, as soon as the MACRO Station sees that the specified input lines are in the specified states, the trigger will occur — it is level-trigger, not edge-triggered.

MI913 parameter determines which of the Flag inputs will be used for position capture (if one is used, see MI912):

0: HMFLn (Home Flag n)
1: PLIMn (Positive End Limit Flag n)
2: MLIMn (Negative End Limit Flag n)
3: USERn (User Flag n)

Typically, this parameter is set to 0 or 3, because in actual use the LIMn flags create other effects that usually interfere with what is trying to be accomplished by the position capture. To capture on the LIMn flags, disable their normal functions with Ix25, or use a channel n where none of the flags is used for the normal axis functions.

Output Mode Select (MSn,MI916)

The PMAC2 Style outputs allow the PMAC to control up to three individual output channels based on the mode. These outputs are described as output A, output B, and output C.

MSn, MI916	Output Description	Typical Use
0	A, B, and C are PWM	Direct PWM Mode Only
1	A and B are DAC, C is PWM	±10V Outputs for torque, velocity and sinusoid input amplifiers
2	A and B are PWM, C is PFM	Stepper Systems
3	A and B are DAC, C is PFM	±10V Outputs with MLDT Feedback

DAC Output Mode Example for ACC-24E2A at MACRO Station

```
MS0,MI916=3                     ;DAC output for Channel 1
MS1,MI916=3                     ;DAC output for Channel 2
MS4,MI916=3                     ;DAC output for Channel 3
MS5,MI916=3                     ;DAC output for Channel 4
MS8,MI916=3                     ;DAC output for Channel 5
MS9,MI916=3                     ;DAC output for Channel 6
MS12,MI916=3                    ;DAC output for Channel 7
MS13,MI916=3                    ;DAC output for Channel 8
```

MACRO Station Encoder Conversion Table (MSn,MI120-MI151)

At power-up, the MACRO Station will set up all of the key memory locations and MI-Variables automatically based on the SW1 connector and firmware of the MACRO Station. The key variables set up at power-up are the encoder conversion table, servo output registers, and flag input registers. By default all encoder conversion table entries are set for 1/T of incremental encoder as:

```
MS0,MI120=$008000               ;output at X:$0010 at MACRO Station (encoder 1)
MS0,MI121=$008008               ;output at X:$0011 at MACRO Station (encoder 2)
MS0,MI122=$008010               ;output at X:$0012 at MACRO Station (encoder 3)
MS0,MI123=$008018               ;output at X:$0013 at MACRO Station (encoder 4)
MS0,MI120=$008040               ;output at X:$0014 at MACRO Station (encoder 5)
MS0,MI121=$008048               ;output at X:$0015 at MACRO Station (encoder 6)
MS0,MI122=$008050               ;output at X:$0016 at MACRO Station (encoder 7)
MS0,MI123=$008058               ;output at X:$0017 at MACRO Station (encoder 8)
MS1,MI120=$009000               ;output at X:$0090 at MACRO Station (encoder 1)
MS1,MI121=$009008               ;output at X:$0091 at MACRO Station (encoder 2)
MS1,MI122=$009010               ;output at X:$0092 at MACRO Station (encoder 3)
MS1,MI123=$009018               ;output at X:$0093 at MACRO Station (encoder 4)
MS1,MI120=$009040               ;output at X:$0094 at MACRO Station (encoder 5)
MS1,MI121=$009048               ;output at X:$0095 at MACRO Station (encoder 6)
MS1,MI122=$009050               ;output at X:$0096 at MACRO Station (encoder 7)
MS1,MI123=$009058               ;output at X:$0097 at MACRO Station (encoder 8)
```

Note

For more information on how to setup encoder conversion table look at MACRO 16 User Manual.

Using ACC-24E2A with Power PMAC

Much of the hardware on the ACC-24E2A is software configurable to provide the maximum flexibility. This section explains in detail the software setup for the Power PMAC CPU that is required to use the ACC-24E2A in different modes of operation.

When the Power PMAC is re-initialized using the **`$$$***`** command with ACC-24E2A in the rack, the most commonly used settings of the setup elements will be made automatically. Further, the hardware setup window in the Power PMAC IDE program will walk you through the setup of many of these elements. So for many users, this manual section will be primarily for reference. However, other users will want to know how to set up the board "manually".

DSPGATE1 Servo IC Data Structure Elements

The Power PMAC CPU communicates with the ACC-24E2A by accessing memory-mapped registers in the "DSPGATE1" Servo IC that was designed by Delta Tau to provide a sophisticated interface between controller software and system hardware. Settings of the control registers in this IC determine the software configuration of the ACC-24E2A for a particular application.

The Power PMAC has defined a "**Gate1**" data structure to organize the control and status settings of the ASIC. In the Script environment – either buffered program statements or on-line commands – a user can utilize this **Gate1[*i*]** data structure directly, or use its "alias" of the **Acc24E2A[*i*]** data structure. In C programs, the user must utilize the **Gate1[*i*]** data structure directly, although a "`#define`" substitution may be employed to get it. This manual will use the **Gate1[*i*]** name for maximum generality.

Elements of the data structure that affect all servo channels are of the form:

Gate1[*i*].*{element name}*

The ASIC index number "*i*" has a valid range of 4 to 19 in the UMAC rack and corresponds to the setting of the addressing DIP switches on SW1 of the 3U-format base board. No two boards with the DSPGATE1 IC (ACC-24E2, ACC-24E2A, ACC-24E2S, ACC-51E) on a single backplane may have the same DIP switch setting; otherwise an addressing conflict will be created (Look at Hardware Setup Switch Configuration).

Elements of the data structure that affect only a single servo channel are of the form:

Gate1[*i*].Chan[*j*].*{element name}*

The channel index number "*j*" has a range of 0 to 3, corresponding to hardware channels 1 to 4, respectively, for the accessory. *The index number for a channel is one less than the corresponding hardware channel number.*

Channel index numbers 0 and 1 correspond to hardware channels 1 and 2, which are on the 3U-format base board, on the left side of a 4-channel assembly. Channel index numbers 2 and 3 correspond to hardware channels 3 and 4, which are on the 3U-format piggyback board, on the right side of a 4-channel assembly.

Software Setup for Clock Signals

The clock frequencies of an ACC-24E2A are set by software data structure elements. Some of these clock frequencies can be used to control the entire Power PMAC system.

Phase and Servo Clock Direction

In a Power PMAC UMAC system, only one machine interface IC is the source of phase and servo clock signals for the entire system. This IC generates its own phase and servo clock signals and outputs them to the rest of the system over the UBUS backplane. The Power PMAC CPU uses these signals as interrupts to drive its phase-commutation and servo-loop algorithms, respectively. Other machine interface ICs use these clock signals to drive their own input and output functions.

On the re-initialization of a Power PMAC UMAC system, the CPU will automatically set up the lowest-numbered DSPGATE1 IC found as the source of the system clock signals. It does this by setting **Gate1[*i*].PhaseServoDir** for this IC to 0. It sets **Gate1[*i*].PhaseServoDir** for all of the other ICs it finds to 3, so they will input their phase and servo clock signals. (However, if any boards with DSPGATE2 or DSPGATE3 ICs are found, one of those boards will be used as the source of system clock signals and **Gate1[*i*].PhaseServoDir** for all of the boards with the DSPGATE1 IC will be set to 3 to input the clock signals.

Phase Clock Frequency

The frequency of phase clock generated internally by the IC on an ACC-24E2A is determined by the settings of **Gate1[*i*].PwmPeriod** and **Gate1[*i*].PhaseClockDiv**. If this IC is the source of the phase clock for the system, this frequency will control the entire UMAC system, hardware and software.

Gate1[*i*].PwmPeriod controls the internal "MaxPhase" clock frequency (and the PWM frequency, which is not used) on the ACC-24E2A's "DSPGATE1" Servo IC. The internally generated Phase and Servo clocks on the IC are derived from the MaxPhase clock. It controls these frequencies by setting the limits of the PWM up-down counter, which increments and decrements at the PWMCLK frequency of 117,964.8 kHz (117.9648 MHz).

To set **Gate1[*i*].PwmPeriod** for a desired "maximum phase" clock frequency, the following formula can be used:

$$Gate1[i].PwmPeriod = \frac{117{,}964.8(kHz)}{2 * MaxPhaseFreq(kHz)} - 3$$

or:

$$MaxPhaseFreq(kHz) = \frac{117{,}964.8(kHz)}{2 * \left(Gate1[i].PwmPeriod + 3\right)}$$

Gate1[*i*].PwmPeriod constitutes bits 8 – 23 of the full-word element **Gate1[i].PwmCtrl** (bits 16 – 31 of the 32-bit element in C).

Gate1[*i*].PhaseClockDiv controls how many times the Phase clock frequency is divided down from the "maximum phase" clock. The Phase clock frequency is equal to the "maximum phase" clock frequency divided by (**Gate1[*i*].PhaseClockDiv**+1). **Gate1[*i*].PhaseClockDiv** has a range of 0 to 15, so the frequency division can be by a factor of 1 to 16. The equation for **Gate1[*i*].PhaseClockDiv** is

$$Gate1[i].PhaseClockDiv = \frac{MaxPhaseFreq(kHz)}{PhaseFreq(kHz)} - 1$$

or:

$$PhaseFreq(kHz) = \frac{MaxPhaseFreq(kHz)}{\left(Gate1[i].PhaseClockDiv + 1\right)}$$

Gate1[*i*].PhaseClockDiv constitutes bits 16 – 19 of the full-word element **Gate1[*i*].ClockCtrl** (bits 24 – 27 of the 32-bit element in C).

Servo Clock Frequency

Gate1[*i*].ServoClockDiv sets the internally generated servo clock frequency for the IC by specifying how many times the frequency is divided down from the phase clock frequency. It has a range of 0 to 15, specifying a division factor of 1 to 16, respectively. The equation for **Gate1[*i*].ServoClockDiv** is:

$$Gate1[i].ServoClockDiv = \frac{PhaseFreq(kHz)}{ServoFreq(kHz)} - 1$$

or:

$$ServoFreq(kHz) = \frac{PhaseFreq(kHz)}{(Gate1[i].ServoClockDiv + 1)}$$

The default value is 3, specifying a 4-times division. With the default phase-clock frequency of 9.035 kHz, this yields a servo-clock frequency of 2.259 kHz.
For ICs receiving an external servo clock signal, the setting of this element does not really matter, but users are encouraged to set it to the same value as on the IC that is generating the system clock signals.
Gate1[*i*].ServoClockDiv constitutes bits 20 – 23 of the full-word element **Gate1[*i*].ClockCtrl** (bits 28 – 31 of the 32-bit element in C).
The global setup element **Sys.ServoPeriod** (in milliseconds) must be set to the inverse of the system servo-clock frequency expressed in kilohertz. This parameter tells the trajectory interpolation algorithms how far to advance the commanded motion equations each servo cycle. It can be calculated as:

$$Sys.ServoPeriod = \frac{(2 * PwmPeriod + 3)(PhaseClockDiv + 1)(ServoClockDiv + 1)}{117{,}964.8}$$

Hardware Clock Frequencies

Each DSPGATE1 IC has four internal clock signals that control its own hardware features. These clock signals are not shared between ICs, and do not need to be the same on different ICs. These clock signals are:

- **ADC_CLK** – Bit clock for amplifier A/D converters
- **DAC_CLK** – Bit clock for D/A converters
- **PFM_CLK** – Clock for PFM pulse-generation circuits
- **SCLK** – Sampling clock for encoder and discrete-input circuits

Each of these clock signals has a 3-bit control that can take a value "*n*" from 0 to 7, specifying a frequency division factor of 2^n (1 to 256) from the source clock frequency of 39.32 MHz. These four 3-bit controls are combined in the 12-bit element **Gate1[*i*].HardwareClockCtrl**. There is seldom reason to change this parameter from the default.
Gate1[*i*].HardwareClockCtrl constitutes bits 0 – 11 of the full-word element **Gate1[*i*].ClockCtrl** (bits 8 – 19 of the 32-bit element in C).

Software Setup for Position Feedback

The ACC-24E2A is typically used to provide interface for quadrature encoders, possibly with digital Hall-style commutation position sensors as well. It can also be used for MLDT feedback. The setup for each type of feedback is described in this section.

Quadrature Encoders

The most commonly used feedback type is the digital quadrature encoder. The Power PMAC with ACC-24E2A has many features to provide powerful and flexible, but easy-to-use, functionality with these encoders.

IC Hardware Setup

Gate1[*i*].Chan[*j*].EncCtrl determines how the IC will decode the quadrature (or similar) signal connected to the A and B inputs of the encoder connector. Most commonly, a setting of 3 or 7 is used to specify a "times-4" (x4 – four counts per encoder line) decode in either the "clockwise" or "counter-clockwise" direction sense. However, other settings are possible.

When used for servo-loop feedback, the direction sense of the feedback must match the direction sense of the servo-loop output, or a dangerous runaway condition may result when the loop is closed. That is, a positive servo-loop output must cause the position feedback to increment in the positive direction.

Changing the direction sense of the decode for the feedback encoder of a motor that is operating properly will result in unstable positive feedback and a dangerous runaway condition in the absence of other changes.

Use for Commutation Feedback

If quadrature encoder feedback is used for commutation position angle, **Motor[*x*].pPhaseEnc** should be set to **Gate1[*i*].Chan[*j*].PhaseCapt.a** to use the count value latched on the phase clock interrupt. This 24-bit value is mapped into the high 24-bits of the 32-bit Power PMAC data bus, so one quadrature count is equal to 256 LSBs of the 32-bit value read by the processor. The user must calculate how many LSBs of this 32-bit register there are per commutation cycle in order to set the commutation scale factor **Motor[*x*].PhasePosSf**, which multiplies the value in the source register to obtain the commutation angle. If there were 2000 quadrature counts per commutation cycle, there would be 512,000 LSBs per 2048-state commutation cycle. In this case, **Motor[*x*].PhasePosSf** would be set to 2048 divided by 512,000. (It is best to enter this value as an expression and let Power PMAC calculate the double-precision floating-point value itself).

Use for Servo Feedback or Master

To process the quadrature count value latched by the IC on the servo clock interrupt for normal servo use, with "1/T" timer-based fractional-count extension (the default and most common format) for servo-loop use (feedback or master), a "Type 3" software 1/T extension entry should be specified in the Encoder Conversion Table (ECT). The IDE's setup menu for the ECT allows you to specify an entry just by specifying the method, the IC index number, and channel index number. The result will be a value scaled in counts, with 9 bits of fractional-count extension (for a resolution of 1/512 of a count).

To set up the entry manually, make settings of the following type:

```
EncTable[n].Type = 3                                  // Software 1/T extension
EncTable[n].pEnc = Gate1[i].Chan[j].ServoCapt.a       // Whole count value latched by servo
EncTable[n].pEnc1 = Gate1[i].Chan[j].TimeBetweenCts.a // Timer register
EncTable[n].ScaleFactor = 1/512                       // Scale to whole counts
```

To process the count value latched by the IC without any fractional-count extension, a "Type 1" single-register read entry should be specified in the ECT, reading the channel's "phase capture" register. (This method is most commonly used when feeding back a pulse-and-direction signal for a simulated loop, as for traditional stepper drives.) This is a 24-bit register in the high 24 bits of the 32-bit data bus. To set up this entry manually, make settings of the following type:

```
EncTable[n].Type = 1                              // Single-register read
EncTable[n].pEnc = Gate1[i].Chan[j].PhaseCapt.a   // Whole count value latched by phase
EncTable[n].index2 = 8                            // Shift right 8 to eliminate "garbage"
EncTable[n].index1 = 8                            // Shift left 8 to get MSB in high bit
EncTable[n].MaxDelta = 0                          // No derivative limiting
EncTable[n].ScaleFactor = 1/256                   // Scale to whole counts
```

In either case, to use the result of the conversion table entry for outer (position) loop motor feedback, set **Motor[*x*].pEnc** to **EncTable[*n*].a**. To use the result for inner velocity loop motor feedback, set **Motor[*x*].pEnc2** to **EncTable[*n*].a**. To use the result as a master position for the motor's position-following function, set **Motor[*x*].pMasterEnc** to **EncTable[*n*].a**. To use the result for a time-base master frequency, set **Coord[*x*].pDesTimeBase** to **EncTable[*n*].DeltaPos.a**.

If the servo loop for the motor is closed in the phase interrupt, a feature enabled by setting **Motor[*x*].PhaseCtrl** bit 3 (value 8) to 1, then the encoder conversion table, **Motor[*x*].pEnc**, and **Motor[*x*].pEnc2** are not used. Instead, Power PMAC directly reads the register whose address is contained in **Motor[*x*].pPhaseEnc** for both inner (velocity) and outer (position) loop servo feedback (regardless of whether Power PMAC is doing phase commutation for the motor or not).

To use the channel's encoder feedback for this purpose, **Motor[*x*].pPhaseEnc** should be set to **Gate1[*i*].Chan[*j*].PhaseCapt.a**. Power PMAC reads the full 32-bit register, with the whole-count value being in the high 24 bits of this register. (No fractional-count estimation is possible in this mode.) Therefore, each count of the encoder is equivalent to 256 (2^8) LSBs of this register. Unlike in the conversion table, there is no capability to shift this data around to get the result in units of counts.

Use for Trigger Position

To use the IC channel's hardware-captured position value as the "trigger position" for motor triggered moves such as homing-search moves, **Motor[*x*].pCaptPos** must be set to **Gate1[*i*].Chan[*j*].HomeCapt.a**, the address of the register that latches the present encoder position immediately on the selected input trigger condition. This latches a 24-bit value in whole counts (no fractional-count information) into a register in the high 24 bits on the 32-bit data bus.

This data must then be processed to match the scaling and offset of the servo feedback position. If 1/T extension is used for servo feedback, **Motor[*x*].CaptPosRightShift** should be set to 8 to shift out the low 8 bits of "garbage data" from the 32-bit read of the 24-bit register. **Motor[*x*].CaptPosLeftShift** should be set to 9 to match the 9 bits of fractional-count resolution of the 1/T extension used for servo feedback. **Motor[*x*].CaptPosRound** should be set to 1 to enable the half-count offset of the captured position that will match the half-count offset performed on extended servo feedback.

These settings are made automatically when the Power PMAC is re-initialized by the `$$$***` command with an ACC-24E2A present. They are also made (with the address based on **Motor[*x*].pEncStatus**) when **Motor[*x*].EncType** is assigned a value of 2 (1/T extension in a PMAC2-style IC) in the Script environment.

If only the whole-count value is used for servo feedback, **Motor[*x*].CaptPosRightShift** should be set to 8 to shift out the low 8 bits of "garbage data" from the 32-bit read of the 24-bit register. **Motor[*x*].CaptPosLeftShift** should be set to 8 to match the resolution of the whole-count data used for servo feedback. **Motor[*x*].CaptPosRound** should be set to 0 to disable half-count offset of the captured position, because the servo position does not have this offset either. These settings are made automatically when **Motor[*x*].EncType** is assigned a value of 1 (whole-count feedback PMAC2-style IC) in the Script environment.

Encoder Loss Detection

If differential encoder inputs are used, it is possible for the ACC-24E2A hardware to detect “encoder loss”, as when a cable becomes disconnected. The socketed resistor pack for the encoder channel (RP22 for the first channel on a board, RP24 for the second channel) must be reversed from its factory default configuration so that pin 1 of the pack (marked with a dot) is at the opposite end from pin 1 of the socket (indicated on the board). In this configuration, all encoder lines are pulled up to 5V if not driven from the encoder; a connected and working differential encoder always holds one signal high and its complement low.

The status bit detecting “encoder loss” for the channel is located in the “ID chip” for the board, not in the ASIC itself. There are no pre-defined data structure elements for these status bits, so the user should assign M-variables to them if he wishes to monitor them. For an ACC-24E2A in the “first” address location (IC index = 4), the following M-variable definitions (using IDE “ptr” variable declarations) could be used:

```
ptr IC4Chan0EncLoss -> u.io:$D00040.13.1   // 1st channel loss bit
ptr IC4Chan1EncLoss -> u.io:$D00044.13.1   // 2nd channel loss bit
ptr IC4Chan2EncLoss -> u.io:$D00048.13.1   // 3rd channel loss bit
ptr IC4Chan3EncLoss -> u.io:$D0004C.13.1   // 4th channel loss bit
```

The bit has a value of 1 if the encoder is present, and a value of 0 if “loss” is detected. Addresses for the ID chips for ACC-24E2A boards in other address settings are given in a table in the section on *Register Element Addresses*, below.

There is no automatic use of the encoder loss bit. Most users who wish to detect this condition will monitor the bit(s) in a PLC program and “kill” (disable) the motor on finding an encoder-loss condition.

Hall-Style Commutation Sensors

Digital Hall-style commutation sensors are frequently used with ACC-24E2A boards. These signals can be from true magnetic Hall-effect sensors, or from “Hall commutation tracks” on optical incremental encoders that mimic the signal output of the older magnetic sensors. In either case, they should provide three digital signals each of 50% duty cycle, offset either 1/3-cycle from each other (“120^o spacing” – most common), or 1/6-cycle from each other (“60^o spacing – less common). The 120^o-spacing format is strongly recommended, because the most common failure modes create states – all high or all low – that are not legal states, and so can easily be detected.

Most commonly, they are used to provide approximate power-on phase position when Power PMAC is performing the motor phase commutation. Occasionally, they are used as the only position feedback, mostly in high-speed, high-power velocity-control applications.

Use for Power-On Commutation Position

When used for power-on phase commutation position, the three Hall signals should be connected into the U, V, and W signals on the encoder connector. It does not matter which signal is connected into which input, but any change will affect the offset and potentially the direction sense of the encoded position value.

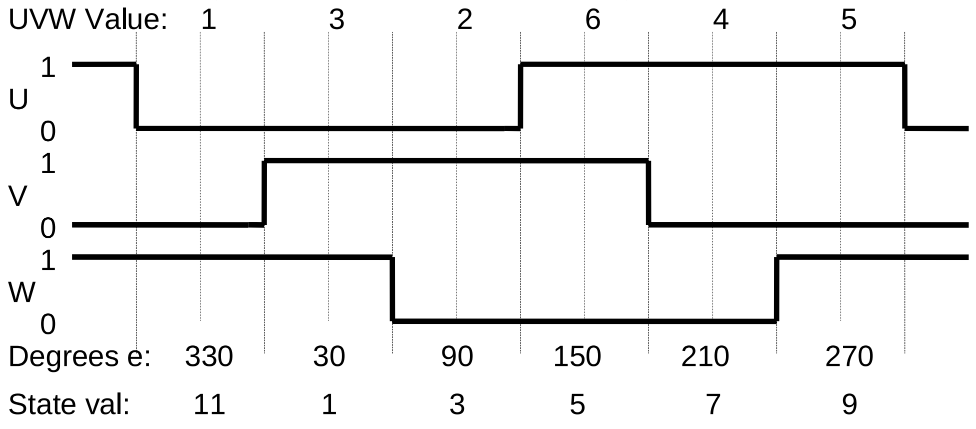
Motor[*x*].pAbsPhasePos should be set to **Gate1[*i*].Chan[*j*].Status.a** so the absolute phase position read function is enabled and uses the IC channel’s status register where the states of the U, V, and W appear.

Motor[*x*].AbsPhasePosFormat should be set to $0400031C to interpret the value in this register correctly. The “04” specifies the 120^0 Hall spacing format (“05” would specify 60^o spacing); the “00” is not used; the “03” specifies using 3 bits of this register; and the “1C” specifies starting at bit 28 (= 1C hex) of the register. This setting tells Power PMAC to read bits 28, 29, and 30 of the specified register, where the W, V, and U inputs, respectively, are found, and to interpret them as 120^o-spaced Hall-style sensors.

Motor[*x*].AbsPhasePosSf multiplies the sensor value to convert it into commutation cycle units. Power PMAC considers the position value from a 3-phase Hall sensor to have 12 units per cycle – 6 states and 6 edges. (When reading the sensor, it assumes that it will be halfway between the edges.) The commutation cycle has 2048 units per cycle, so the magnitude of **Motor[*x*].AbsPhasePosSf** for Hall sensors should be 2048/12 = 170.667.

The proper sign of **Motor[*x*].AbsPhasePosSf** is determined by the direction sense of the Hall sensors relative to the ongoing commutation position sensor. For Hall sensors with the standard 120^{o} spacing, if W leads V, and V leads U in the counting-up sense of the ongoing commutation cycle, the direction sense agrees, and **Motor[*x*].AbsPhasePosSf** should be set to (+) 170.667. However, if U leads V, and V leads W in the counting-up sense of the ongoing commutation cycle, the direction sense does not agree, and **Motor[*x*].AbsPhasePosSf** should be set to
–170.667.

After scaling the power-on position into commutation units, Power PMAC adds the value of **Motor[*x*].AbsPhasePosOffset** to this to get the power-on commutation angle. This parameter permits the Hall sensor to have a different zero position than the commutation cycle does. Power PMAC considers the zero position in the Hall cycle to be the transition of the V signal when U is low (0) and W is high (1).



The best way "manually" to determine the proper value of **Motor[*x*].AbsPhasePosOffset** is first to drive the (unloaded) motor to the commutation cycle zero point with the "stepper motor" phasing search. (This assumes that basic commutation has already been set up properly.) To do this, set **Motor[*x*].PhaseFindingTime** to a value of 256 or greater (the step time in servo cycles), and **Motor[*x*].PhaseFindingDac** to a large enough value to cause a reliable search (3000 should be enough for an unloaded motor). The phasing search can then be commanded with an on-line **`#x$`** command. (Note that you must address the motor immediately before the command – you cannot simply rely on the currently addressed motor.)

Now put **Motor[*x*].PhasePos** and **Gate1[*i*].Chan[*j*].UVW** in the IDE's Watch window so they can be continually monitored. The first element is the present commutation angle (in the range of 0 to 2048), and the second is the present value of the U, V, and W signal triplet (in the range of 1 to 6, with U having a value of 4, V of 2, and W of 1). Kill the motor with the on-line **`#xk`** command and turn the motor slowly manually until you find the position where the **UVW** value changes between 1 and 3. Note the value of **Motor[*x*].PhasePos** at this point, and put this value in **Motor[*x*].AbsPhasePosOffset**.

This test can also confirm whether the Hall sensor direction sense matches the commutation cycle direction sense or not. If the value of **Motor[*x*].PhasePos** increases as the **UVW** value changes from 1 to 3, the direction senses match, and **Motor[*x*].AbsPhasePosSf** should be set to (+) 170.667. If it decreases, the direction senses do not match, and **Motor[*x*].AbsPhasePosSf** should be set to –170.667.

Because the phase referencing with the Hall sensors is only approximate – it can have a +/-30^o error, a correction will need to be made subsequently. Most commonly, this correction is made at the motor home position, and this home position usually occurs at the index pulse of the encoder (typically the first index pulse inside the home switch). To set up this correction in this case, the value of **Motor[*x*].PhasePos** must be found at the index pulse. It is best to do a homing search on the index pulse with no home offset and read the value of **Motor[*x*].PhasePos** after the move has finished (it can be very difficult to find the index pulse manually). This value can be stored in saved setup element **Motor[*x*].AbsPhasePosForce** for future use. In the actual application, after the homing-search move is completed and the motor settled, the value saved in **Motor[*x*].AbsPhasePosForce** can be copied back into the active element **Motor[*x*].PhasePos** to provide the correction.
Remember to set **Motor[*x*].PhaseFindingTime** back to 0 before saving the other settings to non-volatile memory. Now, an on-line **#*x*$** command (or setting **Motor[*x*].PhaseFindingStep** to 1 in a program) will cause the Power PMAC to read the Hall sensors to establish the phase reference.

Use for Servo and Ongoing Commutation Feedback

In some applications of velocity control of large brushless motors, 3-phase Hall sensors provide the only feedback for the motor, as they are significantly more robust than most quadrature encoders. The ACC-24E2A provides an easy way of handling this configuration.
When used for servo and/or ongoing phase commutation position, the three Hall signals should be connected into the A+, B+, and and C+ signals on the encoder connector. It does not matter which signal is connected into which input. **Gate1[*i*].Chan[*j*].EncCtrl** should be set to 11 or 15 to specify a “times-6” (x6 – six counts per cycle) decode in either the “clockwise” or “counter-clockwise” direction sense. This permits the Hall signals to be used like a quadrature encoder, except with 3-phase input.
Note that this feedback method can provide excellent velocity control at approximately constant speeds, particularly when 1/T extension of the Hall edge count is used, but the low fundamental position resolution means that there cannot be good position control at standstill.
For servo feedback, the channel’s count value should be processed in the encoder conversion table with 1/T extension (Type = 3), just as for a quadrature encoder (see above). The motor’s position and velocity loops use the result of the table entry, again just as for a quadrature encoder, as explained above.
For ongoing commutation feedback, **Motor[*x*].PhasePos** should be set to **Gate1[*i*].Chan[*j*].PhaseCapt.a**, as for a quadrature encoder, to use the value of the counter latched on the phase interrupt. Since there are 6 counts per cycle, **Motor[*x*].PhasePosSf** should be set to 2048/(256*6) to convert the data to the 2048-state commutation cycle. The value should be entered as an expression so Power PMAC can compute the exact numerical value to double-precision floating-point accuracy, preventing errors from accumulating over many revolutions.
In this mode, the Hall sensors can also be used for power-on phase position as well, similar to the above case where they were connected to the U, V, and W inputs. **Motor[*x*].pAbsPhasePos** should be set to **Gate1[*i*].Chan[*j*].Status.a**. **Motor[*x*].AbsPhasePosFormat** should be set to $04000314 to read the A, B, and C input bits in the register as 3-phase Hall sensors. **Motor[*x*].AbsPhasePosSf** should be set to 2048/12 = 170.667, as when the U, V, and W inputs are used. **Motor[*x*].AbsPhasePosOffset** should be set in the same manner as when the U, V, and W inputs are used, explained above, except that **Gate1[*i*].Chan[*j*].ABC** should be monitored for the transition between 1 and 3, instead of **UVW**.

MLDT Sensors

The ACC-24E2A can be set up for direct interface with “externally excited” Magnetostrictive Linear Displacement Transducers (MLDTs), such as MTS’s Temposonics brand. MLDTs can provide absolute position information in rugged environments; they are particularly well suited to hydraulic applications.
In this interface, the ACC-24E2A provides a periodic electronic excitation pulse output to the MLDT, receives the echo pulse that returns at the speed of sound in the transducer, and very accurately measures the time between these pulses, which is directly proportional to the distance of the moving member from the stationary base of the transducer. The timer therefore contains a position measurement.

IC Hardware Setup

The excitation pulse is generated using the IC's pulse-frequency modulation (PFM) output on Phase C. The differential pulse is output on what is otherwise would be the T and W flag inputs on the encoder connector. Jumpers E1C and E1D must be ON to enable this output for an odd-numbered hardware channel (1 or 3; channel index 0 or 2); jumpers E2C and E2D must be ON to enable this output for an even-numbered hardware channel (2 or 4; channel index 1 or 3). **Gate1[*i*].Chan[*j*].OutputMode** must be set to 3 to put the channel's Phase C in PFM (not PWM) mode. (This setting puts Phases A and B in DAC mode as well.)

Gate1[*i*].PwmDeadTime sets the output pulse width (time) for all channels that have Phase C in PFM mode (and sets the dead time for all channels that have Phase C in PWM mode.) in units of the PFM clock period. The PFM clock period is set by **Gate1[*i*].HardwareClockCtrl**; at the default clock frequency of 9.83 MHz, the clock period is 102 nanoseconds, and at the default setting for **Gate1[*i*].PwmDeadTime** of 15, the pulse width is about 1.5 microseconds, which is suitable for almost all MLDT interfaces.

The pulse output frequency for a channel is controlled by both the PFM clock frequency for the IC and the PFM command value for the channel. When used for stepper-motor applications, the PFM command value is updated every servo cycle to control the stepper-motor velocity. However, for MLDT use, the PFM command register is just written to once, on power-up/reset (the value is *not* saved to flash memory). The desired value is simply assigned to the **Gate1[*i*].Chan[*j*].Pfm** element.

The frequency of the pulse output should produce a period just slightly longer than the longest expected response time for the echo pulse. For MLDTs, the response time is approximately 0.35 μsec/mm (9 μsec/inch). On an MLDT 1500 mm (~60 in) long, the longest response time is approximately 540 μsec; a recommended period between pulse outputs for this device is 600 μsec, for a frequency of 1667 Hz.

To produce the desired pulse output frequency from a channel on the DSPGATE1, the following formula can be used for the 24-bit element **Gate1[*i*].Chan[*j*].Pfm**:

$$OutputFreq(kHz) = \frac{Gate1[i].Chan[j].Pfm}{16{,}777{,}216} PFMCLK\_Freq(kHz)$$

or:

$$Gate1[i].Chan[j].Pfm = 16{,}777{,}216 * \frac{OutputFreq(kHz)}{PFMCLK\_Freq(kHz)}$$

To produce a pulse output frequency of 1.667 kHz with the default PFMCLK frequency of 9.83 MHz, we calculate:

$$Gate1[i].Chan[j].Pfm = 16{,}777{,}216 * \frac{1.667}{9{,}830} = 2{,}982$$

Note

The servo update time for the motor using the MLDT should be at least as high as the output time set here (the servo frequency should be as low or lower than the output frequency).

The "echo" return pulse from the sensor must be connected to the CHA+ and CHA- inputs on the encoder connector. To measure the time from the output pulse to the echo pulse, saved setup element **Gate1[*i*].Chan[*j*].EncCtrl** must be set to 12. In this mode, the timer value representing the sensor position can be found in the read-only element **Gate1[*i*].Chan[*j*].TimeBetweenCts**. This is a 24-bit element whose low 23 bits represent the time between output and echo pulses. The timer is cleared to zero at the falling edge of the output pulse. It then counts up at 117.96 MHz until a rising edge on the return pulse is received, at which point the timer's value is latched into the memory-mapped register **Gate1[*i*].Chan[*j*].TimeBetweenCts** where the processor can read it.

Each increment of the time represents a physical length along the sensor that is determined by the timer frequency and the speed of the return pulse in the MLDT, which is the speed of sound in the metal. This speed varies from device to device, but is always approximately the inverse of 0.35 μsec/mm, or 9.0 μsec/inch. With the timer frequency of 117.96 MHz, the resolution can be calculated as:

$$\mathrm{Re}\,solution\left(\frac{length}{increment}\right)=\frac{1}{TimerFreq}\left(\frac{\mu\sec}{increment}\right)*\mathrm{Re}\,turnSpeed\left(\frac{length}{\mu\sec}\right)$$

$$\cong\frac{1}{117.96}\frac{\mu\sec}{increment}*\frac{1}{0.35}\frac{mm}{\mu\sec}\cong 0.024\frac{mm}{increment}\left(41.3\frac{increments}{mm}\right)$$

$$\cong\frac{1}{117.96}\frac{\mu\sec}{increment}*\frac{1}{9.0}\frac{inches}{\mu\sec}\cong 0.00094\frac{inches}{increment}\left(1060\frac{increments}{inch}\right)$$

Use for Servo Feedback or Master

To process the timer value latched by the IC on the receipt of the return pulse, a “Type 1” single-register read conversion should be specified in the Encoder Conversion Table. **EncTable[*n*].pEnc**, which specifies the address of this register, should be set to **Gate1[*i*].Chan[*j*].TimeBetweenCts.a**. (**EncTable[*n*].pEnc1**, which specifies a second source, is not used in this mode; its setting does not matter.)
The timer data we are interested in is found in bits 8 – 30 of the 32-bit Power PMAC data bus. **EncTable[*n*].index2** should be set to 8 to shift the data right 8 bits and eliminate any “garbage data” found in the low 8 bits. **EncTable[*n*].index1** should be set to 9 to then shift the data left 9 bits and put the highest timer bit in the MSB of the 32-bit register. Since the LSB of the timer is now found in bit 9 of the 32-bit register. **EncTable[*n*].ScaleFactor** should be set to 1/512 so the output units are in timer increments.
It is strongly recommended that the entry implement a velocity limit as a protection against missed or spurious echo pulses. To do this, set **EncTable[*n*].index3** to 0 to specify a velocity (not acceleration) limit, and set **EncTable[*n*].MaxDelta** to a value slightly greater than the greatest true velocity the system will see, expressed in timer units per servo cycle. Set **EncTable[*n*].index4** to 0 so the result is not integrated.
To use the result of the conversion table entry for outer (position) loop motor feedback, set **Motor[*x*].pEnc** to **EncTable[*n*].a**. To use the result for inner velocity loop motor feedback, set **Motor[*x*].pEnc2** to **EncTable[*n*].a**. To use the result as a master position for the motor’s position-following function, set **Motor[*x*].pMasterEnc** to **EncTable[*n*].a**.

Use for Absolute Power-On Position

MLDTs provide absolute position information, but Power PMAC must be specifically set up to use it for this purpose. (Otherwise, it considers the power-on/reset to position to be zero, no matter what the sensor value is.)
Motor[*x*].pAbsPos should be set to **Gate1[*i*].Chan[*j*].TimeBetweenCts.a** to read the timer value register for absolute power-on position. **Motor[*x*].AbsPosFormat** should be set to $00001708 to read 23 (=17 hex) bits starting at bit 8 and interpret them as unsigned binary ($00). The scale factor element **Motor[*x*].AbsPosSf** should be set to the same value as the ongoing scale factor element **Motor[*x*].PosSf** (which is typically 1.0 so the motor units are timer increments). **Motor[*x*].AbsPhasePosOffset** can be used to specify the distance between “sensor zero” (which cannot be reached with an MLDT) and “motor zero”. Absolute position is read using the on-line `hmz` command or the buffered program `homez` command.

Software Setup for Flags

The ACC-24E2A provides multiple input and output "flag" functions for a motor on each interface channel. Use of these flag functions by the Power PMAC requires proper setup of several saved data structure elements. Typically, the default values generated by re-initializing the Power PMAC with the ACC-24E2A installed will enable the flag functions as desired by most users, but some users may want to change these defaults.

Motor Flag Addresses

The ACC-24E2A provides multiple input and output "flag" functions for a motor on each interface channel. For each function, a motor addressing setup element must contain the address of the channel's input ("**Status**") flag register or output ("**Ctrl**") register, as appropriate. In typical use, all of the flags for a motor use the same servo interface channel on the ACC-24E2A, but separate addressing elements provide flexibility for the cases where this is not so.

The following flag address-element settings should be made to use the flags of a channel on the ACC-24E2A. These are the default settings made when the Power PMAC is re-initialized using the `$$$***` command with an ACC-24E2A present.

Motor[*x*].pAmpEnable = **Gate1[*i*].Chan[*j*].Ctrl.a**	// Amp enable in output flag register
Motor[*x*].pAmpFault = **Gate1[*i*].Chan[*j*].Status.a**	// Amp fault in input flag register
Motor[*x*].pCaptFlag = **Gate1[*i*].Chan[*j*].Status.a**	// Trigger flags in input flag register
Motor[*x*].pEncStatus = **Gate1[*i*].Chan[*j*].Status.a**	// Encoder status in input flag register
Motor[*x*].pLimits = **Gate1[*i*].Chan[*j*].Status.a**	// Limit flags in input flag register

Motor[*x*].pCaptFlag is automatically set to the same value as **Motor[*x*].pEncStatus** on re-initialization, and when **Motor[*x*].EncType** is assigned a value of 1, 2, or 3 in the Script environment to specify use of a PMAC2 interface channel.

For the amplifier-enable, amplifier-fault, and limit flag functions, if the address-element value is set to 0, that automatic flag function is disabled.

It is also necessary to specify which bit number(s) in the specified registers (on the full 32-bit data bus) are used for the respective functions. The following settings specify the standard values to use the named flags on an ACC-24E2A for their named functions. These are the default setting made when the Power PMAC is re-initialized using the `$$$***` command with an ACC-24E2A present. They are also set automatically when **Motor[*x*].EncType** is assigned a value of 1, 2, or 3 in the Script environment to specify use of a PMAC2 interface channel.

Motor[*x*].AmpEnableBit = 22	// Amp enable bit in output flag register
Motor[*x*].AmpFaultBit = 23	// Amp fault bit in input flag register
Motor[*x*].CaptFlagBit = 19	// Trigger bit in input flag register
Motor[*x*].LimitBits = 25	// First limit flag bit in input flag register

Flag Polarity

The Power PMAC always writes a 0 value to the selected amplifier-enable output bit to disable the amplifier and a 1 value to enable the amplifier. Hardware resets or failures such as watchdog timer trips automatically force 0 values into the standard amplifier-enable output bits to disable the attached amplifiers. For this reason, the software polarity of the amplifier-enable output is not user-programmable.

The Power PMAC always considers 0 values in the selected limit input bits to mean that the motor is not in the limit, and 1 values to mean that the motor has hit the limit. With the opto-isolation circuitry on the ACC-24E2A, current must be flowing through the isolators (in either direction) to produce a 0 value in the register, meaning that normally-closed limit switches (opening on the limit) must be used. Since the most likely circuit failures produce an open circuit, this is the more fail-safe configuration. For this reason, the software polarity of the overtravel-limit inputs is not user-programmable.

Since there is no standardization in general of the fault outputs of servo amplifiers, the software polarity of the amplifier-fault input to the Power PMAC is user programmable. Power PMAC considers the value in the selected amplifier-fault bit that is equal to **Motor[*x*].AmpFaultLevel** to be the fault state, and the opposite logical level to mean there is no fault.

Capture Trigger Control

The "trigger bit" selected by **Motor[*x*].pCaptFlag** and **Motor[*x*].CaptFlagBit**, discussed above, can be determined by a variety of combinations of input flags and encoder index channels for the interface channel. **Gate1[*i*].Chan[*j*].CaptCtrl** for the channel specifies whether a flag is used or not to create the trigger, whether the encoder index is used or not, and if used, which level of those signals causes a trigger. If a flag is used, then **Gate1[*i*].Chan[*j*].CaptFlagSel** specifies which of the 4 flags (HOME, PLIM, MLIM, USER) is selected.

Position-Compare Outputs

The action of the position-compare ("EQU") output for each channel, which permits the creation of a digital output very precisely related to counter position, is specified by three 24-bit registers, each with its own data structure element, and three control bits, mapped into two elements. Note that the values of these elements (except for **Equ1Ena**) are *not* saved to non-volatile memory.

Compare Control Registers

When the instantaneous encoder-counter value for the channel passes the value in either compare-position register **Gate1[*i*].Chan[*j*].CompA** or **Gate1[*i*].Chan[*j*].CompB**, the digital state of the EQU*n* output toggles. In addition, the "auto-increment" value in **Gate1[*i*].Chan[*j*].CompAdd** is added to the value of the compare-position register *not* presently matching the counter value, which permits a repeated set of "compare pulses" at even intervals.

CompA, **CompB**, and **CompAdd** cover 24-bit registers in the IC, with units of counts. In the script environment, they are 24-bit elements. They are reported as unsigned values (range of 0 to 16,777,215), but can be treated as signed values (range of -8,388,608 to 8,388,607) for ease of use. For example, **CompAdd** can be set to -100 to permit auto-incrementing in the negative direction, but its value will be reported when queried as 16,777,116. These positions are referenced to the position at power-on/reset, not necessarily the motor zero position (the difference between these is stored in **Motor[*x*].HomePos**).

When these three elements are used in a C program, they are 32-bit values with real data only in the high 24 bits, so their effective units are 1/256 of a quadrature count.

Compare Control Bits

Three control bits for the channel are used for the position-compare function. **Gate[*i*].Chan[*j*].Equ1Ena**, if set to 1, specifies that the channel's compare circuitry operate with the IC's first channel's counter; if set to the default value of 0, it operates with its own channel's counter. (For the first channel's compare circuitry, this obviously does nothing!)

Gate1[*i*].Chan[*j*].EquWrite permits the user to force the present state of the compare output. It is a 2-bit value. Bit 1 (value 2) is the value to be set on the output; bit 0 (value 1) is the "forcing" bit. When it is set to 1, the value in bit 1 is forced onto the output, and then the IC automatically resets the forcing bit to 0. So to set the output to 1, **EquWrite** should be set to 3 (and it will then report as 2); to set the output to 0, **EquWrite** should be set to 1 (and it will then report as 0). Note that this element can be used to make the compare output operate as a software-controlled general-purpose output.

These two elements are part of the full-word channel control element **Gate1[*i*].Chan[*j*].Ctrl**. C programs can only access this full-word element. **Equ1Ena** is bit 21 of the full word; **EquWrite** comprises bits 19 and 20.

Status Bit

The present state of the position compare output can be determined by accessing the status bit **Gate1[*i*].Chan[*j*].Equ**. In a C program, the full-word status element **Gate1[*i*].Chan[*j*].Status** should be accessed; the "**Equ**" bit is bit 17 of the 32-bit word.

Software Setup for Amplifiers

Analog-Input Amplifiers

The ACC-24E2A is primarily used to command amplifiers with +/-10V inputs. There are three common classes of these analog-input amplifiers: velocity-mode and torque-mode, which require a single analog command; and "sine-wave" amplifiers, which require dual analog commands.

DAC Signal Setup:

The DSPGATE1 IC used in the ACC-24E2A can support several different output signal formats. To drive the digital-to-analog converters (DACs) on the ACC-24E2A, each channel of the IC must be configured for DAC signal output on Phases A and B of the channel. This is done by setting saved channel element **Gate1[*i*].Chan[*j*].OutputMode** to 1 or 3. (The difference between these settings is in the Phase C output signal – PWM if 1, PFM if 3 – which is not commonly used on the ACC-24E2A.)

Gate1[*i*].Chan[*j*].OutputPol should be set to the default value of 0 for the non-inverting polarity of the data outputs to the DACs on the ACC-24E2A.

For the 18-bit DACs used on the ACC-24E2A, the saved multi-channel element **Gate1[*i*].DacStrobe** must be set to $7FFFC0 to create the proper strobe signal each cycle to all of the DACs commanded by the IC. This is the default value for the element.

The frequency of the DAC_CLK bit-clocking signal generated by the IC is controlled by 3 bits of the saved multi-channel element **Gate1[*i*].HardwareClockCtrl**. This frequency should be set to the default value of 4.9152 MHz for the DACs on the ACC-24E2A.

Motor Setup for Velocity and Torque-Mode Amplifiers:

Analog velocity and torque-mode amplifiers are usually commanded using the Phase A DAC signal (DacA) from a channel of the ACC-24E2A. With these amplifiers, the motor commutation is either performed in the motor (as for a DC brush motor) or in the amplifier (as for brushless servo motors), but not in the Turbo PMAC.

For these amplifiers, **Motor[*x*].PhaseCtrl** should be set to 0 to disable the phase commutation tasks in the Power PMAC for the motor (or to 8 if the servo loop is closed in the phase interrupt).

Saved setup element **Motor[*x*].pDac**, which specifies the address of the register where the servo command output is written, should be set to **Gate1[*i*].Chan[*j*].Dac[0].a** to use the Phase A DAC output for the IC channel.

Motor Setup for Analog Sine-Wave Amplifiers:

Analog "sine-wave input" amplifiers are commanded using both the Phase A and Phase B DAC signals (DacA and DacB) from a channel of the ACC-24E2A. With these amplifiers, Power PMAC is performing the motor phase commutation, but the amplifier is closing the current loop. The analog output voltages represent the commanded current levels for two phases of the motor; if there is a third phase, its command is generated inside the amplifier in a "balance loop".

For these amplifiers **Motor[*x*].PhaseCtrl** should be set to 4 to enable phase commutation tasks in the Power PMAC for the motor, using two separate registers for the DAC commands ("unpacked" mode).

Motor[*x*].pAdc should be set to 0 to disable the current-loop algorithm in Power PMAC.

Saved setup element **Motor[*x*].pDac**, which specifies the address of the register where the servo command output is written, should be set to **Gate1[*i*].Chan[*j*].Dac[0].a** to use the Phase A DAC output for the IC channel to command the first phase of the motor. In this mode, the Phase B DAC for the same channel will be used to command the second phase of the motor.

Motor[*x*].PwmSf controls the scaling of the commutation outputs (whether PWM or DAC format). For full-range +/-10V DAC A and B output ranges from the commutation, this should be set to a magnitude of 32,768. Changing the sign of **Motor[*x*].PwmSf** effectively flips the direction sense of the commutation. Remember that this value acts as a gain term in the feedback loop, so changing its value changes the tuning of the overall feedback loop. **Motor[*x*].MaxDac** can be used to limit the outputs without affecting the scaling or loop gain.

Amplifier Enable and Fault Signals:
Users are strongly encouraged to utilize the amplifier-enable output and amplifier-fault input on the ACC-24E2A when interfacing to amplifiers. The use of these signals is discussed in the section "*Software Setup for Flags*", above.

Pulse-and-Direction Amplifiers

The ACC-24E2A can also be used to command drives with pulse-and-direction inputs. These drives are commonly used with stepper motors, where each pulse represents an increment of one step or microstep. They are also sometimes used with servo motors, where each pulse represents an increment of one encoder count, in what are sometimes called "stepper-replacement" servo drives.
On the ACC-24E2A, the differential pulse-and-direction signals can be output on the encoder (not amplifier) connectors on what are normally the T, U, V, and W flag-input lines. Jumpers E1A, E1B, E1C, and E1D must be ON to enable this output for an odd-numbered hardware channel (1 or 3; channel index 0 or 2); jumpers E2A, E2B, E2C and E2D must be ON to enable this output for an even-numbered hardware channel (2 or 4; channel index 1 or 3).
IC Hardware Setup:
Gate1[*i*].Chan[*j*].OutputMode must be set to 3 to put the channel's Phase C in PFM (not PWM) mode. (This setting puts Phases A and B in DAC mode as well.)
Gate1[*i*].PwmDeadTime sets the output pulse width (time) for all channels that have Phase C in PFM mode (and sets the dead time for all channels that have Phase C in PWM mode.) in units of the PFM clock period. The PFM clock period is set by **Gate1[*i*].HardwareClockCtrl**; at the default clock frequency of 9.83 MHz, the clock period is 102 nanoseconds. At the default setting for **Gate1[*i*].PwmDeadTime** of 15, the pulse width is about 1.5 microseconds.
The pulse width must at least as large as the minimum required pulse width for the drive. However, the minimum gap between pulses is equal to the pulse width, so the minimum pulse cycle period is twice the pulse width set here. This sets a maximum frequency of the PFM output. (If a higher frequency is requested, the IC cannot produce it, and pulses will be skipped.)
Generally, bit 1 of **Gate1[*i*].Chan[*j*].OutputPol** is set to the default value of 0 for high-true pulses on the PULSE+ line. The polarity can be inverted either by using the PULSE- line or by setting this bit to 1.
The single-bit element **Gate1[*i*].Chan[*j*].PfmDirPol** can be used to invert the sense of the direction output in software.
Gate1[*i*].Chan[*j*].EncCtrl should be set to 8 for the channel's encoder input circuitry to use and decode the channel's internally generated pulse-and-direction signals instead of any external signals. (Note that no cabling is required – the connection is made inside the IC.) No external encoder can be used on the channel in this mode.
(Use of the channel's PFM output to close a simulated servo loop is strongly recommended in this mode. However, it is possible to use an external encoder instead. That encoder is processed as for any other class of amplifier, as described above. In general, though, it is difficult to get a stable and dither-free feedback loop using an external encoder. The real position loop in these cases is closed inside the drive.)
Encoder Conversion Table Setup:
To process the count value latched by the IC without any fractional-count extension, a "Type 1" single-register read entry should be specified in the ECT, reading the channel's "phase capture" register. (This method is recommended for the simulated feedback from pulse-and-direction outputs, as 1/T fractional-

count extension tends to lead to dithering when holding position.) This is a 24-bit register in the high 24 bits of the 32-bit data bus. To set up this entry manually, make settings of the following type:

EncTable[*n*].Type = 1	// Single-register read
EncTable[*n*].pEnc = **Gate1[*i*].Chan[*j*].PhaseCapt.a**	// Whole count value latched by phase
EncTable[*n*].index2 = 8	// Shift right 8 to eliminate "garbage"
EncTable[*n*].index1 = 8	// Shift left 8 to get MSB in high bit
EncTable[*n*].MaxDelta = 0	// No derivative limiting
EncTable[*n*].ScaleFactor = 1/256	// Scale to whole counts

Motor Addressing and Mode Setup:

The motor element **Motor[*x*].pDac** should be set to **Gate3[*i*].Chan[*j*].Pfm.a** to specify the use of the channel's Phase C PFM output.

Motor[*x*].pEnc and **Motor[*x*].pEnc2** should be set to **EncTable[*n*].a**, where "*n*" is the index of the table entry that processed the IC channel's "phase capture" count value.

It is recommended that the following servo gain values be used to close a simulated loop:

- **Motor[*x*].Servo.Kp** = 40
- **Motor[*x*].Servo.Kvfb** = 0
- **Motor[*x*].Servo.Kvff** = 40
- **Motor[*x*].Servo.Ki** = 0.001

DSPGATE1 (PMAC2-Style Servo ASIC) Register Elements

This section documents the addresses of the data structure elements for registers and parts of registers in the DSPGATE1 IC in a Power PMAC system. This is primarily for reference, as most uses of the elements and registers do not require the user to know the numerical address of the register.

To calculate the address offset of a register element from the I/O base address:

1. Add the offset of the base address of the IC in question from the I/O base address, based on the IC type and index number. This offset value is found in the boxed table at the top of the section for each IC type.
2. If a channel-specific register, add the offset of the channel's base address from the IC's base address.
3. Add the offset of the register in question from the IC's base address, or for a channel-specific register, from the channel's base address.

This value can be used to declare "io" format M-variables.

To find the absolute numerical address of the register element in the Power PMAC memory map:

1. Find the starting address for I/O by querying the value of **Sys.piom**.
2. Add this to the address offset calculated above

This value will match the numerical value reported when the address of the element is queried.

Notes:

1. Bit numbers are given in "Intel" (little-endian) style, where the value of Bit *n* in the word is 2^n. All bit numbers are for a full 32-bit word, even if the hardware uses less than 32 bits.

2. Names given in bold font are those of saved setup elements. Names given in italics are not actual elements that can be used directly, but express components within full-word elements with distinct functions.

In the Power PMAC script-language environment, both with buffered program statements and on-line commands, elements that are less than 32 bits are accessed using the actual bit width of the element. For example **Gate1[*i*].Chan[*j*].Status** occupies bits 08 – 31 of a 32-bit word. In the script environment, it is treated as a 24-bit value. **Gate1[*i*].Chan[*j*].HomeFlag** occupies bit 24 of this same word. It can be accessed directly as a single-bit value in the script environment

In the C-language, IC registers can be accessed only as full 32-bit words, even if the registers do not occupy all 32-bit words. So **Gate1[*i*].Chan[*j*].Status** is treated as a 32-bit value, with the low 8 bits being meaningless. The single-bit value **Gate1[*i*].Chan[*j*].HomeFlag** cannot be accessed directly from a C program; it would have to be derived with an expression such as

(Gate1[*i*].Chan[*j*].Status & $01000000) >> 24

Gate1[#]	4	5	6	7	8	9	10	11
IC Base Offset	$600000	$600100	$700000	$700100	$608000	$608100	$708000	$708100
Gate1[#]	12	13	14	15	16	17	18	19
IC Base Offset	$610000	$610100	$710000	$710100	$618000	$618100	$718000	$718100

Data Structure	Full-Word Element	Partial-Word Element	Offset from ASIC Base	Bits
`Gate1[i].`	**`ClockCtrl`**		`$030`	
		`HardwareClockCtrl`		`08-19`
		`PhaseServoDir`		`20-21`
		`PhaseClockDiv`		`24-27`
		`ServoClockDiv`		`28-31`
	`DacStrobe`		`$070`	`08-31`
	`AdcStrobe`		`$0B0`	`08-31`
	`PwmCtrl`		`$0F0`	`08-31`
		`PwmDeadTime`		`08-15`
		`PwmPeriod`		`16-31`
	`Chan[0].`		`$000`	
	`Chan[1].`		`$040`	
	`Chan[2].`		`$080`	
	`Chan[3].`		`$0C0`	

Chan[#]	0	1	2	3
Offset from IC Base	$000	$040	$080	$0C0

Data Structure	Full-Word Element	Partial-Word Element	Offset from Channel Base	Bits
`Gate1[i].Chan[j].`	`TimeBetweenCts`		`$000`	`08-31`
		`CompA fraction*`		`08-19`
		`ServoCapt fraction*`		`20-31`
	`TimeSinceCts`		`$004`	`08-31`
		`CompB fraction*`		`08-19`
		`HomeCapt fraction*`		`20-31`
	`Pwm[0] / Dac[0]`		`$008`	`08-31`
	`Pwm[1] / Dac[1]`		`$00C`	`08-31`
	`Pwm[2] / Pfm`		`$010`	`08-31`
	`Adc[0]`		`$014`	`08-31`
	`Adc[1]`		`$018`	`08-31`
	`CompB`		`$01C`	`08-31`
	`Status`		`$020`	`08-31`
		`HallState`		`08-10`
		`CountError`		`16`
		`Equ`		`17`
		`PosCapt`		`18-19`
		`ABC`		`20-22`
		`Fault`		`23`
		`HomeFlag`		`24`
		`PlusLimit`		`25`

```
            MinusLimit                    26
            UserFlag                      27
            UVW                           28-30
            T                             31
PhaseCapt                      $024       08-31
ServoCapt                      $028       08-31
HomeCapt                       $02C       08-31

Ctrl                           $034       08-31
            EncCtrl                       08-11
            CaptCtrl                      12-15
            CaptFlagSel                   16-17
            CountReset                    18
            EquWrite                      19-20
            Equ1Ena                       21
            AmpEna                     22
            GatedIndexSel                 23
            IndexGateState                24-25
            OneOverTEna                   26
            PfmDirPol                     27
            OutputPol                     28-29
            OutputMode                    30-31
CompAdd                        $038       08-31
CompA                          $03C       08-31
```

(* if **OneOverTEna** is set to 1)

Each UMAC card with a DSPGATE1 IC has an “ID chip” with information about the card. The base address offset of the ID chip is given in the table below, followed by information about the registers in chip.

Gate1[#]	4	5	6	7	8	9	10	11
Cid[#]	2	10	3	11	18	26	19	27
ID Chip Base Offset	$D00040	$D00140	$D00050	$D00150	$D08040	$D08140	$D08050	$D08150
Gate1[#]	12	13	14	15	16	17	18	19
Cid[#]	34	42	35	43	50	58	51	59
ID Chip Base Offset	$D10040	$D10140	$D10050	$D10150	$D18040	$D18140	$D18050	$D18150

Full-Word Register	Partial-Word Component	Offset from IC Base	Bits
Reg0		$000	08-15
	*Vendor ID bits 0-3**		08-11
	*Revision number***		08-11
	Clock buffer direction (1=out)	12	
	1st chan encoder loss		13
Reg1		$004	08-15
	*Vendor ID bits 4-7**		08-11
	*Card number bits 0-3***		08-11
	Bank select output	12	
	2nd chan encoder loss		13

```
            Reg2                                $008        08-15
                        Card options bits 0-4*              08-12
                        Card number bits 4-8**              08-12
                        3rd chan encoder loss               13
            Reg3                                $00C        08-15
                        Card options bits 5-9*              08-12
                        Card number bits 9-13**             08-12
                        4th chan encoder loss               13

(* when bank select output set to 0)
(** when bank select output set to 1)
```

Using ACC-24E2A for MLDT Feedback

The ACC-24E2A can provide direct interface to magnetostrictive linear displacement transducers (MLDTs) through its encoder connectors. This interface is for MLDTs with an external excitation format (often called RS-422 format) because of the signal levels, because the ACC-24E2A provides the excitation pulse, and receives the echo pulse, both with RS-422 signal formats.

This section provides basic information for using MLDTs with the ACC-24E2A. More information can be found in the User Manuals for the Turbo PMAC or the MACRO Station.

MLDT Hardware Setup of the ACC-24E2A

The ACC-24E2A must be set up to output the differential pulse on what is normally the T and W input flags on the encoder connector. This is done by putting jumpers on E-points E1C and E1D for the first channel on the board, or E2C and E2D for the second channel on the board. These jumpers are OFF by default.

The PULSE+ (high during the pulse) and PULSE- (low during the pulse) outputs from the encoder connector are connected to the differential pulse inputs on the MLDT. The echo pulse differential outputs from the MLDT are connected to the CHA+ and CHA- input pins on the same encoder connector.

If the MLDT uses RPM format, in which there is a brief start echo pulse, and a brief stop echo pulse, the "+" output from the MLDT should be connected to the CHA+ input on the ACC-24E2A, and the "-" output should be connected to the CHA- input.

If the MLDT uses DPM format, in which there is a single long echo pulse, with the delay to the trailing edge measuring the position, the "+" output from the MLDT should be connected to the CHA- input on the ACC-24E2A, and the "-" output should be connected to the CHA+ input.

MLDT Software Setup of the UMAC Turbo

When the ACC-24E2A is used for MLDT feedback in a UMAC Turbo system, a few I-variables must be set up properly.

Hardware Setup I-Variables for Servo IC m

I7m03 (PFM Clock Frequency): In almost all cases, the clock frequency driving the pulse-generation circuitry for all channels on Servo IC m can be left at its default value of 9.83 MHz (0.102 µsec). I7m03 also controls other clock signals, has a default value of 2258 and rarely needs to be changed.

I7m04 (PFM Pulse Width): The pulse width, set by I7m04 in units of PFM clock cycles must be set long enough for the MLDT to see, and long enough to contain the rising edge of the RPM start echo pulse, or the rising edge of the single DPM echo pulse. For example, if this edge can come up to 2 µsec after the start of the excitation pulse, and the PMAC clock cycle is at its default of about 0.1 µsec, then I7m04 must be set at least to 20.

I7mn6 (Output Format Select): For Servo IC m Channel n to be used for MLDT feedback, I7mn6 must be set to 1 or 3 for the C sub-channel to be used for PFM-format output. On an ACC-24E2A, I7mn6 must then be set to 3 for the A and B sub-channels to be used for DAC-format output.

I7mn0 (MLDT Feedback Select): For Servo IC m Channel n to be used for MLDT feedback, I7mn0 must be set to 12. In this mode, the pulse timer is cleared on the output pulse, and latched on the echo pulse, counting in between at 117.96 MHz.

Conversion Table Processing I-Variables

The pulse timer for Servo IC m Channel n holds a number proportional to the time and therefore the position. This must be processed in the conversion table before it can be used by the servo loop. It is best to use the filtered parallel data conversion, a 3-line entry in the table (three consecutive I-variables.
Line 1 (Method and Address): This 24-bit value (6 hex digits) should begin with a "3" (filtered parallel data) followed by the address of the timer register. The possible values for this line are shown in the following table:

Encoder Conversion Table Parallel Filtered Data Format First Line for ACC-24E2A Boards with Servo IC m Channel n

ACC-24 #	Servo IC #	Channel 1	Channel 2	Channel 3	Channel 4
1A	2	$378200	$378208	$378210	$378218
1B	3	$378300	$378308	$378310	$378318
2A	4	$379200	$379208	$379210	$379218
2B	5	$379300	$379308	$379310	$379318
3A	6	$37A200	$37A208	$37A210	$37A218
3B	7	$37A300	$37A308	$37A310	$37A318
4A	8	$37B200	$37B208	$37B210	$37B218
4B	9	$37B300	$37B308	$37B310	$37B318

Line 2 (Width and Start): This 24-bit value should be set to $013000 to specify the use of 19 bits ($013) starting at bit 0.
Line 3 (Max Change): This 24-bit value should be set to a value slightly greater than the maximum true velocity ever expected, expressed in timer LSBs per servo cycle. With a typical MLDT, the 117.96 MHz timer LSB represents 0.024 mm (0.00094 inches); the default servo cycle is 0.442 msec.
The result of this conversion is in the X-register of the third line. Any functions using this value should address this register. For example, if this were the first entry in the table, which starts at $003501, the result would be in X:$003503.

Motor I-Variables

Ixx03 (Position Loop Feedback Address): To use the result of the conversion table for position-loop feedback for Motor xx, Ixx03 should contain the address of the result register in the conversion table - $003503 in the above example.
Ixx04 (Velocity Loop Feedback Address): To use the result of the conversion table for velocity-loop feedback for Motor xx, Ixx04 should contain the address of the result register in the conversion table - $003503 in the above example.
Ixx05 (Master Position Address): To use the result of the conversion table for the master position for Motor xx, Ixx05 should contain the address of the result register in the conversion table - $003503 in the above example.
Ixx10 and Ixx95 (Power-On Position Address and Format): To use the MLDT for absolute power-on position for Motor xx, Ixx95 should be set to $180000 (up to 24 bits of parallel Y-data) and Ixx10 should be set to the address of the timer register used:

Ixx10 for ACC-24E2A MLDT Timer Registers (Ixx95=$180000)

ACC-24 #	Servo IC #	Channel 1	Channel 2	Channel 3	Channel 4
1	2	$078200	$078208	$078210	$078218
2	3	$078300	$078308	$078310	$078318
3	4	$079200	$079208	$079210	$079218
4	5	$079300	$079308	$079310	$079318
5	6	$07A200	$07A208	$07A210	$07A218
6	7	$07A300	$07A308	$07A310	$07A318
7	8	$07B200	$07B208	$07B210	$07B218
8	9	$07B300	$07B308	$07B310	$07B318

Ixx80 (Power-On Mode): Set Ixx80 to 4 to delay the absolute power-on position read until the pulse-output frequency can be set.
Ixx81 and Ixx91 (Power-On Phase Position Address and Format): Occasionally the MLDT is used to establish an absolute phase reference position for Turbo-PMAC-commutated motors. In this case, Ixx81 and Ixx91 are set to the same values as Ixx10 and Ixx95, respectively (see above).

Pulse Output Frequency

The pulse-output frequency is established by assigning an M-variable to the C sub-channel command register, and writing a value to that M-variable after every power-up/reset. The suggested M-variable for the Motor xx using this register is:

```
Mxx07->Y:{address},8,16,S
```

where **`{address}`** is specified according to the following table:

Mxx07 for ACC-24E2A MLDT Pulse-Output Registers

ACC-24 #	Servo IC #	Channel 1	Channel 2	Channel 3	Channel 4
1	2	$078204	$07820C	$078214	$07821C
2	3	$078304	$07830C	$078314	$07831C
3	4	$079204	$07920C	$079214	$07921C
4	5	$079304	$07930C	$079314	$07931C
5	6	$07A204	$07A20C	$07A214	$07A21C
6	7	$07A304	$07A30C	$07A314	$07A31C
7	8	$07B204	$07B20C	$07B214	$07B21C
8	9	$07B304	$07B30C	$07B314	$07B31C

The frequency of the pulse output should produce a period just slightly longer than the longest expected response time for the echo pulse. For MLDTs, the response time is approximately 0.35 µsec/mm (9 µsec/inch). On an MLDT 1500 mm (~60 in) long, the longest response time is approximately 540 µsec; a recommended period between pulse outputs for this device is 600 µsec, for a frequency of 1667 Hz.
To produce the desired pulse output frequency, the following formula can be used (assuming a 16-bit M-variable definition):

$$OutputFreq(kHz) = \frac{Mxx07}{65{,}536} PFMCLK\_Freq(kHz)$$

or:

$$Mxx07 = 65{,}536 * \frac{OutputFreq(kHz)}{PFMCLK\_Freq(kHz)}$$

To produce a pulse output frequency of 1.667 kHz with the default PFMCLK frequency of 9.83 MHz, we calculate:

$$Mxx07 = 65{,}536 * \frac{1.667}{9{,}380} \cong 11$$

To write this value to the register, a power-on PLC routine is suggested; this can also be done with on-line commands from the host computer. Sample PLC code to do this for Channel 1, using the above example value, is:

```
OPEN PLC 1                    ; PLC 1 is first program to execute
CLEAR
  M107=11                     ; Set pulse frequency
  CMD"$*"                     ; Absolute Position Read
DISABLE PLC 1                 ; So will not execute again
CLOSE
```

PMAC2/Turbo PMAC2 Conversion Table and Motor I-Variables

Once the MACRO Station has been set up to process the MLDT feedback, the PMAC2 or Turbo PMAC2 can process the ongoing position feedback with its conversion table, Ix03, and Ix04 just as for any other feedback from a MACRO Station.

If the MLDT is used for absolute power-on position for the servo loop, the proper variables must be set on the PMAC2 or Turbo PMAC2:

PMAC2 Ix10 (Power-On Position Address and Format): To get the absolute position in this format for Motor x through MACRO node n (n = 0 to 15 decimal), Ix10 should be set to $74000n, where n here is the hexadecimal representation of the node number (n = 0 to F hex).

Turbo PMAC2 Ixx10 & Ixx95 (Power-On Position Address and Format): To get the absolute position for Motor xx through MACRO node n (n = 0 to 63 decimal), Ixx10 should be set to n; in hex-format $0000nn, where nn is the hexadecimal representation of the node number (nn = 00 to 3F hex). If node 0 is used, Ixx10 should be set to $000100 (256 decimal). Ixx95 should be set to $740000 to specify parallel data through a MACRO node.

If the MLDT is used for absolute power-on phase position for commutation, the proper variables must be set on the PMAC2 or Turbo PMAC2:

PMAC2 Ix81 (Power-On Phase Position Address and Format): To get the absolute phase position in this format for Motor x through MACRO node n (n = 0 to 15 decimal), Ix81 should be set to $74000n, where n here is the hexadecimal representation of the node number (n = 0 to F hex).

Turbo PMAC2 Ixx81 & Ixx91 (Power-On Phase Position Address and Format): To get the absolute phase position for Motor xx through MACRO node n (n = 0 to 63 decimal), Ixx81 should be set to n; in hex-format $0000nn, where nn is the hexadecimal representation of the node number (nn = 00 to 3F hex). If node 0 is used, Ixx81 should be set to $000100 (256 decimal). Ixx91 should be set to $740000 to specify parallel data through a MACRO node.

MLDT Feedback for UMAC-MACRO

The data from the MLDT is processed as a parallel word input at the MACRO Station and then transmitted back to the Ultralite using the traditional Servo Node. The encoder conversion table at the MACRO Station must be modified to process this data. From the Ultralite standpoint, nothing needs to be modified to read the position and velocity data.

Since the data is also absolute, the data can also be sent at the Ultralite as absolute data for correct position at power-up. This is accomplished with the proper setup of MSn,MI11x at the MACRO Station, and Ix10 at the Ultralite or Ix10 and Ix95 with the Turbo Ultralite. Regardless of the type of Ultralite, retrieving the power-on-position is the same. The information must be retrieved from MACRO Station variable MSn,MI920 for each node transfer as specified by Ix10 at the Ultralite. MSn,MI920 does not need to be set up because the MACRO Station will place the power-on position the appropriate register at power-up.

MLDT Software Setup of the UMAC MACRO

When the ACC-24E2A is used for MLDT feedback in a UMAC MACRO system, there are a few MI-variables in the MACRO Station, and a few in the PMAC2 or Turbo PMAC2 driving the Station, that must be set up properly.

Station Hardware Setup I-Variables for Servo IC

MS{anynode},MI903/MI907 (PFM Clock Frequency): In almost all cases, the clock frequency driving the pulse-generation circuitry for all channels on the Servo IC can be left at its default value of 9.83 MHz (0.102 μsec). Few will need to change MI903/MI907, which also controls other clock signals, from its default value of 2258.

MS{anynode},MI904/MI908 (PFM Pulse Width): The pulse width, set by MI904/MI908 in units of PFM clock cycles must be set long enough for the MLDT to see, and long enough to contain the rising edge of the RPM start echo pulse, or the rising edge of the single DPM echo pulse. For example, if this edge can come up to 2 μsec after the start of the excitation pulse, and the PMAC clock cycle is at its default of about 0.1 μsec, then I7m04 must be set at least to 20.

MS{node},MI916 (Output Format Select): For the channel associated with this node to be used for MLDT feedback, MI916 must be set to 1 or 3 for the C sub-channel to be used for PFM-format output. On an ACC-24E2A, I7mn6 must then be set to 3 for the A and B sub-channels to be used for DAC-format output.

MS{node},MI910 (MLDT Feedback Select): For the channel associated with this node to be used for MLDT feedback, MI910 must be set to 12. In this mode, the pulse timer is cleared on the output pulse, and latched on the echo pulse, counting in between at 117.96 MHz.

Station Conversion Table Processing I-Variables

The pulse timer for Servo IC m Channel n holds a number proportional to the time and therefore the position. This must be processed in the conversion table before it can be used by the servo loop. It is best to use the filtered parallel data conversion, a 3-line entry in the table (three consecutive MI-variables. The MI-variables for the conversion table start at MI120.

Line 1 (Method and Address): This 24-bit value (6 hex digits) should begin with a 3 (filtered parallel data) followed by the address of the timer register. The possible values for this line are shown in the following table:

Encoder Conversion Table Parallel Filtered Data Format First Line for ACC-24E2A Boards

ACC-24 #	Channel 1	Channel 2	Channel 3	Channel 4
1	$30C040	$30C048	$30C050	$30C058
2	$30C060	$30C068	$30C070	$30C078

Line 2 (Bits Used Mask): This 24-bit value should be set to $07FFFF to specify the use of the low 19 bits of the 24-bit source word.
Line 3 (Max Change): This 24-bit value should be set to a value slightly greater than the maximum true velocity ever expected, expressed in timer LSBs per servo cycle. With a typical MLDT, the 117.96 MHz timer LSB represents 0.024 mm (0.00094 inches); the default servo cycle is 0.442 msec.
The result of this conversion is in the X-register of the third line. Any functions using this value should address this register. For example, if this were the first entry in the table, which starts at $000010, the result would be in X:$0012.

Station Motor Node I-Variables

MS{anynode}, MI10x (xth Motor Node Position Loop Feedback Address): To use the result of the conversion table for position-loop feedback for the xth motor node, MI10x should contain the address of the result register in the conversion table - $0012 in the previous example.
MS{anynode}, MI11x (xth Motor Node Absolute Position Address): To use the MLDT for absolute power-on position for the xth motor node, set MI11x to $18xxxx (up to 24 bits of parallel Y-data) from Station address xxxx, where xxxx is the address of the timer register.

MS{anynode},MI11x xth Motor Node Absolute Position

ACC-24 #	Channel 1	Channel 2	Channel 3	Channel 4
1	$30C042	$30C04A	$30C052	$30C05A
2	$30C062	$30C06A	$30C072	$30C07A

MS{anynode}, MI16x (xth Motor Node MLDT Frequency Control): This variable establishes the frequency of the excitation pulse sent to the MLDT. Its value is written automatically to the full 24-bit C sub-channel command register for the channel assigned to this node, so the PFM circuit will create a pulse train at this frequency.
To compute the output frequency as a function of MI16x, the following formula can be used:

$$OutputFreq(kHz) = \frac{MI16x}{16{,}777{,}216} PFMCLK\_Freq(kHz)$$

To compute the required value of MI16x as a function of the desired output frequency, the following formula can be used:

$$MI16x = 16{,}777{,}216 * \frac{OutputFreq(kHz)}{PFMCLK\_Freq(kHz)}$$

Power-On Feedback Address for PMAC2 Ultralite

Both the Ultralite and the Turbo Ultralite can obtain absolute position at power up or upon request (#n$*). The Ultralite must have Ix10 setup and the Turbo Ultralite needs both Ixx10 and Ixx95 setup to enable this power on position function. For power on position reads as specified in this document, MACRO firmware version 1.114 or newer is needed, the Turbo Ultralite firmware must be 1.936 or newer, and lastly the standard Ultralite must have firmware version 1.16H or newer.
Ix10 permits an automatic read of an absolute position sensor at power-on/reset. If Ix10 is set to 0, the power-on/reset position for the motor will be considered to be 0, regardless of the type of sensor used. There are specific settings of PMAC's/PMAC2's Ix10 for each type of MACRO interface. If a Turbo Ultralite is used, Ixx95 must also be set appropriately. The Compact MACRO Station has a corresponding variable I11x for each node that must be set.

Absolute Position for Ultralite

Compact MACRO Station Feedback Type (firmware version 1.16H and above)	Ix10 (Unsigned)	Ix10 (Signed)
ACC-8D Opt 7 Resolver/Digital Converter	$73000n	$F3000n
ACC-8D Opt 9 Yaskawa Absolute Encoder Converter	$72000n	$F2000n
ACC-8D Opt 10 Sanyo Absolute Encoder Converter	$74000n	$F4000n
ACC-28B or ACC-28E Analog/Digital Converter	$74000n	$F4000n
MACRO Station Option 1C/ACC-6E A/D Converter	$74000n	$F4000n
MACRO Station Parallel Input	$74000n	$F4000n
MACRO Station MLDT Input	$74000n	$F4000n
n is the MACRO node number used for Motor x: 0, 1, 4, 5, 8, 9, C(12), or D(13).		

Absolute Position for Turbo Ultralite (Ixx95=$720000 - $740000, $F20000 - $F40000) Addresses are MACRO Node Numbers

MACRO Node Number	Ixx10 for MACRO IC 0	Ixx10 for MACRO IC 1	Ixx10 for MACRO IC 2	Ixx10 for MACRO IC 3
0	$000100	$000010	$000020	$000030
1	$000001	$000011	$000021	$000031
4	$000004	$000014	$000024	$000034
5	$000005	$000015	$000025	$000035
8	$000008	$000018	$000028	$000038
9	$000009	$000019	$000029	$000039
12	$00000C	$00001C	$00002C	$00003C
13	$00000D	$00001D	$00002D	$00003D

Compact MACRO Station Feedback Type	Ixx95 (Unsigned)	Ixx95 (Signed)
ACC-8D Opt 7 Resolver/Digital Converter	$730000	$F30000
ACC-8D Opt 9 Yaskawa Absolute Encoder Converter	$720000	$F20000
ACC-8D Opt 10 Sanyo Absolute Encoder Converter	$740000	$F40000
ACC-28B Analog/Digital Converter	$740000	$F40000
MACRO Station Option 1C/ACC-6E A/D Converter	$740000	$F40000
MACRO Station Parallel Input, MLDT, SSI	$740000	$F40000

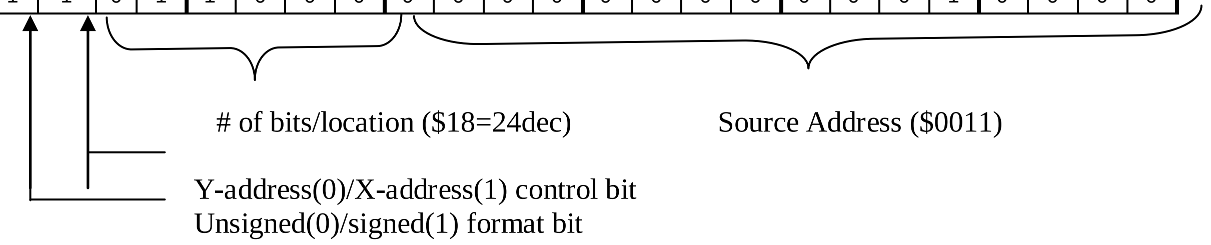
When PMAC or PMAC2 has Ix10 set to get absolute position over MACRO, it executes a station auxiliary read command MS{node},I920 to request the absolute position from the Compact MACRO Station. The station then references its own I11x value to determine the type, format, and address of the data to be read.

MACRO Parallel Absolute Position Setup

MI111 through MI118 (MI11*x*) specify whether, where, and how absolute position is to be read on the Compact MACRO Station for a motor node (MI11*x* controls the *x*th motor node, usually which corresponds to Motor *x* on PMAC) and sent back to the Ultralite.
If MI11*x* is set to 0, no power-on reset absolute position value will be returned to PMAC. If MI11*x* is set to a value greater than 0, then when the PMAC requests the absolute position because its Ix10 and/or Ix81 values are set to obtain absolute position through MACRO (sending an auxiliary **`MS{node},MI920`** command), the Compact MACRO Station will use MI11*x* to determine how to read the absolute position, and report that position back to PMAC as an auxiliary response.
For an MLDT, take the output from the encoder conversion table (ECT) at the MACRO Station and process it as an absolute position because the information in the ECT is synchronized properly. Remember, the output from the encoder conversion table will reside in the X register. For example, with the following entry:
MS0,MI120=$30C040 ($10 of ECT)
MS0,MI121=$FFFFFF ($11 of ECT)

MS0,MI122=32 ($12 of ECT)
The output from the ECT will reside in X:$12 and this will be the register to obtain the absolute data from.
MI11*x* consists of two parts. The low 16 bits (last four hexadecimal digits) specify the address on the MACRO Station from which the absolute position information is read. The high eight bits (first two hexadecimal digits) tell the Compact MACRO Station how to interpret the data at that address (the method.
MACRO MI11x Parallel Word Example: Signed 24-bit Absolute MLDT $0010

HEX($)	**D**				**8**				**0**				**0**				**1**				**0**			
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
VALUE	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0


# of bits/location ($18=24dec)
Source Address ($0011)
Y-address(0)/X-address(1) control bit
Unsigned(0)/signed(1) format bit

X/Y Address Bit: If bit 22 of Ix10 is 0, the PMAC looks for the parallel sensor in its Y address space. This is the standard choice, since all I/O ports map into the Y address space. If this bit is 1, PMAC looks for the parallel sensor in its X address space.
Signed/Unsigned Bit: If the most significant bit (MSB -- bit 23) of MI11x is 0, the value read from the absolute sensor is treated as an unsigned quantity. If the MSB is 1, which adds $80 to the high eight bits of MI11x, the value read from the sensor is treated as a signed, twos-complement quantity.

```
MS0,MI111=$D80010       ;read signed 24-bit absolute power on position
                        ;from X:$0010
```

Example MLDT Setup for UMAC MACRO

Ultralite	Turbo Ultralite	Description

I110=$740000	I110=$000100
	I195=$740000
		Power on position read from MACRO Node 0 as an unsigned value

```
MS0,i161=3825  ;(15*255)
Ms0,i903=2258                ;default
MS0,I904=25    ;might need to increase from factory default
MS0,I910=12
MS0,I916=3
MS0,i120=$30C040
MS0,I121=$FFFFFF             ;24-bit
MS0,I122=32    ;output at $12
MS0,i101=$12
Ms0,i111=$D80010             ;grab data from 1st entry of ECT X register
```

ACC-24E2A TERMINAL BLOCK DESCRIPTION

The terminal blocks on the ACC-24E2A are described as TB1 Top, TB2 Top, TB3 Top, TB1 Bottom, TB2 Bottom, TB3 Bottom, TB1 Front and TB2 Front. The top connectors have the Encoder signals, the bottom connectors have the Amplifier signals, and the front connectors contain the Limit and Flag signals.

Connector TB1 TOP - Encoder 1

Pin#	Symbol	Function	Description	Notes
1	CHA1+	Input	Encoder 1 Positive A Channel	
2	CHA1-	Input	Encoder 1 Negative A Channel	
3	CHB1+	Input	Encoder 1 Positive B Channel	
4	CHB1-	Input	Encoder 1 Negative B Channel	
5	CHC1+	Input	Encoder 1 Positive C Channel	Index channel
6	CHC1-	Input	Encoder 1 Negative C Channel	Index channel
7	ENCPWR	Output	Digital Supply	Power for encoder
8	GND	Common	Digital Reference	
9	CHU1+/DIR_1+	I/O	Supplemental Flag U or Direction 1+	Also Direction Output
10	CHV1+/DIR_1-	I/O	Supplemental Flag V or Direction 1-	Also Direction Output
11	CHW1+/PUL_1+	I/O	Supplemental Flag W or Pulse Output 1+	Also Pulse Output
12	CHT1+/PUL_1-	I/O	Supplemental Flag T or Pulse Output 1-	Also Pulse Output

Connector TB2 Top - Encoder 2

Pin#	Symbol	Function	Description	Notes
1	CHA2+	Input	Encoder 2 Positive A Channel	
2	CHA2-	Input	Encoder 2 Negative A Channel	
3	CHB2+	Input	Encoder 2 Positive B Channel	
4	CHB2-	Input	Encoder 2 Negative B Channel	
5	CHC2+	Input	Encoder 2 Positive C Channel	Index channel
6	CHC2-	Input	Encoder 2 Negative C Channel	Index channel
7	ENCPWR	Output	Digital Supply	Power for encoder
8	GND	Common	Digital Reference	
9	CHU1+/DIR_2+	I/O	Supplemental Flag U or Direction 2+	Also Direction Output
10	CHV1+/DIR_2-	I/O	Supplemental Flag V or Direction 2-	Also Direction Output
11	CHW1+/PUL_2+	I/O	Supplemental Flag W or Pulse Output 2+	Also Pulse Output
12	CHT1+/PUL_2-	I/O	Supplemental Flag T or Pulse Output 2-	Also Pulse Output

Connector TB3 Top – EQU Outputs

Pin#	Symbol	Function	Description	Notes
1	GND	Common	Reference Voltage	
2	BEQU1	Output	Compare Output 1	
3	BEQU2	Output	Compare Output 2	

Connector TB1 Bottom Amp – Out 1

Pin#	Symbol	Function	Description	Notes
1	DAC1A+	Output	Phase A Analog Out	+/-10V, ref to AGND
2	DAC1A-	Output	Phase A Analog Out	-/+10V; ref to AGND
3	DAC1B+	Output	Phase B Analog Out	+/-10V, ref to AGND
4	DAC1B-	Output	Phase B Analog Out	-/+10V; ref to AGND
5	AE_NC_1	Output	Amplifier Enable	Normally closed
6	AE_COM_1	Input	Amplifier Enable	
7	AE_NO_1	Output	Amplifier Enable	Normally open
8	AFAULT_1+	Input		
9	AFAULT_1-	Input		
10	AA+15V	Input*	Analog Positive Supply Voltage	Remove jumpers E85, E87, E88 if using external power
11	AA-15V	Input*	Analog Negative Supply Voltage	Remove jumpers E85, E87, E88 if using external power
12	AAGND	Input*	Analog Reference Voltage	Remove jumpers E85, E87, E88 if using external power
*External power supply inputs for opto-isolation from the digital ground plane.				

Connector TB2 Bottom Amp – Out 2

Pin#	Symbol	Function	Description	Notes
1	DAC2A+	Output	Phase A Analog Out	+/-10V, ref to AGND
2	DAC2A-	Output	Phase A Analog Out	-/+10V; ref to AGND
3	DAC2B+	Output	Phase B Analog Out	+/-10V, ref to AGND
4	DAC2B-	Output	Phase B Analog Out	-/+10V; ref to AGND
5	AE_NC_2	Output	Amplifier Enable	Normally closed
6	AE_COM_2	Output	Amplifier Enable	
7	AE_NO_2	Output	Amplifier Enable	Normally open
8	AFAULT_2+	Input		
9	AFAULT_2-	Input		
10	AA+15V	Input*	Analog Positive Supply Voltage	Remove jumpers E85, E87, E88 if using external power
11	AA-15V	Input*	Analog Negative Supply Voltage	Remove jumpers E85, E87, E88 if using external power
12	AAGND	Input*	Analog Reference Voltage	Remove jumpers E85, E87, E88 if using external power
* External power supply inputs for opto-isolation from the digital ground plane.				

Connector TB3 Bottom – Analog Power

Pin#	Symbol	Function	Description	Notes
1	AAGND	Input*	Analog Reference Voltage	Remove jumpers E85, E87, E88 if using external power
2	AA+15V	Input*	Analog Positive Supply Voltage	Remove jumpers E85, E87, E88 if using external power
3	AA-15V	Input*	Analog Negative Supply Voltage	Remove jumpers E85, E87, E88 if using external power
* External power supply inputs for opto-isolation from the digital ground plane.				

Connector TB1 Front- Limits 1

Pin#	Symbol	Function	Description	Notes
1	USER1	Input	General Capture Flag	Sinking or sourcing
2	PLIM1	Input	Positive Limit Flag	Sinking or sourcing
3	MLIM1	Input	Negative Limit Flag	Sinking or sourcing
4	HOME1	Input	Home Flag	Sinking or sourcing
5	FLG_1_RET	Input	Return For All Flags	+V (12 to 24V) or 0V

Connector TB2 Front- Limits 2

Pin#	Symbol	Function	Description	Notes
1	USER2	Input	General Capture Flag	Sinking or sourcing
2	PLIM2	Input	Positive Limit Flag	Sinking or sourcing
3	MLIM2	Input	Negative Limit Flag	Sinking or sourcing
4	HOME2	Input	Home Flag	Sinking or sourcing
5	FLG_2_RET	Input	Return For All Flags	+V (12 to 24V) or 0V

ACC-24E2 OPTION 1A TERMINAL BLOCK DESCRIPTION

The terminal blocks on the ACC-24E2 option 1A are described as TB1 Top, TB2 Top, TB3 Top, TB1 Bottom, TB2 Bottom, TB3 Bottom, TB1 Front and TB2 Front. The top connectors have the Encoder signals, the bottom connectors have the Amplifier signals, and the front connectors contain the Limit and Flag signals.

Connector TB1 Top - Encoder 3

Pin#	Symbol	Function	Description	Notes
1	CHA3+	Input	Encoder 3 Positive A Channel	Also pulse input
2	CHA3-	Input	Encoder 3 Negative A Channel	Also pulse input
3	CHB3+	Input	Encoder 3 Positive B Channel	Also direction input
4	CHB3-	Input	Encoder 3 Negative B Channel	Also direction input
5	CHC3+	Input	Encoder 3 Positive C Channel	Index channel
6	CHC3-	Input	Encoder 3 Negative C Channel	Index channel
7	ENCPWR	Output	Digital Supply	Power for encoder
8	GND	Common	Digital Reference	
9	CHU3+/DIR_3+	I/O	Supplemental Flag U or Direction 3+	Also direction output
10	CHV3+/DIR_3-	I/O	Supplemental Flag V or Direction 3-	Also direction output
11	CHW3+/PUL_3+	I/O	Supplemental Flag W or Pulse Output 3+	Also pulse output
12	CHT3+/PUL_3-	I/O	Supplemental Flag T or Pulse Output 3-	Also pulse output

Connector TB2 Top – Encoder 4

Pin#	Symbol	Function	Description	Notes
1	CHA4+	Input	Encoder 4 Positive A Channel	
2	CHA4-	Input	Encoder 4 Negative A Channel	
3	CHB4+	Input	Encoder 4 Positive B Channel	
4	CHB4-	Input	Encoder 4 Negative B Channel	
5	CHC4+	Input	Encoder 4 Positive C Channel	Index channel
6	CHC4-	Input	Encoder 4 Negative C Channel	Index channel
7	ENCPWR	Output	Digital Supply	Power for encoder
8	GND	Common	Digital Reference	
9	CHU1+/DIR_4+	I/O	Supplemental Flag U or Direction 4+	Also direction output
10	CHV1+/DIR_4-	I/O	Supplemental Flag V or Direction 4-	Also direction output
11	CHW1+/PUL_4+	I/O	Supplemental Flag W or Pulse Output 4+	Also pulse output
12	CHT1+/PUL_4-	I/O	Supplemental Flag T or Pulse Output 4-	Also pulse output

Connector TB3 Top – EQU Outputs

Pin#	Symbol	Function	Description	Notes
1	GND	Common	Reference Voltage	
2	BEQU3	Output	Compare Output 3	
3	BEQU4	Output	Compare Output 4	

Connector TB1 Bottom Amp-Out 3

Pin#	Symbol	Function	Description	Notes
1	DAC3A+	Output	Phase A Analog out	+/-10V, ref to AGND
2	DAC3A-	Output	Phase A Analog out	-/+10V; ref to AGND
3	DAC3B+	Output	Phase B Analog out	+/-10V, ref to AGND
4	DAC3B-	Output	Phase B Analog out	-/+10V; ref to AGND
5	AE_NC_3	Output	Amplifier Enable	Normally closed
6	AE_COM_3	Output	Amplifier Enable	
7	AE_NO_3	Output	Amplifier Enable	Normally open
8	AFAULT_3+	Input		
9	AFAULT_3-	Input		
10	AA+15V	Input*	Analog Positive Supply Voltage	Remove jumpers E85, E87, E88 if using external power
11	AA-15V	Input*	Analog Negative Supply Voltage	Remove jumpers E85, E87, E88 if using external power
12	AGND	Input*	Analog Reference Voltage	Remove jumpers E85, E87, E88 if using external power
* External power supply inputs for opto-isolation from the digital ground plane.				

Connector TB2 Bottom Amp-Out 4

Pin#	Symbol	Function	Description	Notes
1	DAC4A+	Output	Phase A Analog out	+/-10V, ref to AGND
2	DAC4A-	Output	Phase A Analog out	-/+10V; ref to AGND
3	DAC4B+	Output	Phase B Analog out	+/-10V, ref to AGND
4	DAC4B-	Output	Phase B Analog out	-/+10V; ref to AGND
5	AE_NC_4	Output	Amplifier Enable	Normally closed
6	AE_COM_4	Output	Amplifier Enable	
7	AE_NO_4	Output	Amplifier Enable	Normally open
8	AFAULT_4+	Input		
9	AFAULT_4-	Input		
10	AA+15V	Input*	Analog Positive Supply Voltage	Remove jumpers E85, E87, E88 if using external power
11	AA-15V	Input*	Analog Negative Supply Voltage	Remove jumpers E85, E87, E88 if using external power
12	AAGND	Input*	Analog Reference Voltage	Remove jumpers E85, E87, E88 if using external power
* External power supply inputs for opto-isolation from the digital ground plane.				

Connector TB3 Bottom-Analog Power

Pin#	Symbol	Function	Description	Notes
1	AAGND	Input	Analog Reference Voltage	Remove jumpers E85, E87, E88 if using external power
2	AA+15V	Input	Analog Positive Supply Voltage	Remove jumpers E85, E87, E88 if using external power
3	AA-15V	Input	Analog Negative Supply Voltage	Remove jumpers E85, E87, E88 if using external power
* External power supply inputs for opto-isolation from the digital ground plane.				

Connector TB1 Front - Limits 3

Pin#	Symbol	Function	Description	Notes
1	USER3	Input	General Capture Flag	Sinking or sourcing
2	PLIM3	Input	Positive Limit Flag	Sinking or sourcing
3	MLIM3	Input	Negative Limit Flag	Sinking or sourcing
4	HOME3	Input	Home Flag	Sinking or sourcing
5	FLG_3_RET	Input	Return for All Flags	+V (12 to 24V) or 0V

Connector TB2 Front - Limits 4

Pin#	Symbol	Function	Description	Notes
1	USER4	Input	General Capture Flag	Sinking or sourcing
2	PLIM4	Input	Positive Limit Flag	Sinking or sourcing
3	MLIM4	Input	Negative Limit Flag	Sinking or sourcing
4	HOME4	Input	Home Flag	Sinking or sourcing
5	FLG_4_RET	Input	Return for All Flags	+V (12 to 24V) or 0V

ACC-24E2A DB15 CONNECTOR OPTION

DB15 Style Connector J1 Top - Encoder 1 / EQU

Pin#	Symbol	Function	Description	Notes
1	CHT1+/PUL_1-	I/O	Supplemental Flag T or Pulse Output 1-	Also pulse output
2	CHV1+/DIR_1-	I/O	Supplemental Flag V or Direction 1-	Also direction output
3	GND	Common	Digital Reference	
4	CHC1-	Input	Enc 1 Neg. C Chan.	Index channel
5	CHB1-	Input	Enc 1 Neg. B Chan.	
6	CHA1-	Input	Enc 1 Neg. A Chan.	
7	GND	Common	Reference Voltage	
8	BEQU2	Output	Compare Output 2	
9	CHW1+/PUL_1+	I/O	Supplemental Flag W or Pulse Output 1+	Also pulse output
10	CHU1+/DIR_1+	I/O	Supplemental Flag U or Direction 1+	Also direction output
11	ENCPWR	Output	Digital Supply	Power for encoder
12	CHC1+	Input	Enc 1 Pos. C Chan.	Index channel
13	CHB1+	Input	Enc 1 Pos. B Chan.	
14	CHA1+	Input	Enc 1 Pos. A Chan.	
15	BEQU1	Output	Compare Output 1	

DB15 Style Connector J2 Top - Encoder 2 / EQU

Pin#	Symbol	Function	Description	Notes
1	CHT2+/PUL_2-	I/O	Supplemental Flag T or Pulse Output 2-	Also pulse output
2	CHV2+/DIR_2-	I/O	Supplemental Flag V or Direction 2-	Also direction output
3	GND	Common	Digital Reference	
4	CHC2-	Input	Enc 2 Neg. C Chan.	Index channel
5	CHB2-	Input	Enc 2 Neg. B Chan.	
6	CHA2-	Input	Enc 2 Neg. A Chan.	
7	GND	Common	Reference Voltage	
8	BEQU2	Output	Compare Output 2	
9	CHW2+/PUL_2+	I/O	Supplemental Flag W or Pulse Output 2+	Also pulse output
10	CHU2+/DIR_2+	I/O	Supplemental Flag U or Direction 2+	Also direction output
11	ENCPWR	Output	Digital Supply	Power for encoder
12	CHC2+	Input	Enc 2 Pos. C Chan.	Index channel
13	CHB2+	Input	Enc 2 Pos. B Chan.	
14	CHA2+	Input	Enc 2 Pos. A Chan.	

Pin#	Symbol	Function	Description	Notes
15	BEQU1	Output	Compare Output 1	

DB15 Style Connector J1 Bottom Amp – Out 1/Analog Power

Pin#	Symbol	Function	Description	Notes
1	DAC1A+	Output	Phase A Analog Out	+/-10V, ref to AGND
2	DAC1B+	Output	Phase B Analog Out	+/-10V, ref to AGND
3	AE_NC_1	Output	Amplifier Enable	Normally closed
4	AE_NO_1	Output	Amplifier Enable	Normally open
5	AFAULT_1-	Input		
6	AA-15V	Input*	Analog Negative Supply Voltage	Remove jumpers E85, E87, E88 if using external power
7	AAGND	Input*	Analog Reference Voltage	Remove jumpers E85, E87, E88 if using external power
8	AA-15V	Input*	Analog Negative Supply Voltage	Remove jumpers E85, E87, E88 if using external power
9	DAC1A-	Output	Phase A Analog Out	-/+10V; ref to AGND
10	DAC1B-	Output	Phase B Analog Out	-/+10V; ref to AGND
11	AE_COM_1	Input	Amplifier Enable	
12	AFAULT_1+	Input		
13	AA+15V	Input*	Analog Positive Supply Voltage	Remove jumpers E85, E87, E88 if using external power
14	AAGND	Input*	Analog Reference Voltage	Remove jumpers E85, E87, E88 if using external power
15	AA+15V	Input*	Analog Positive Supply Voltage	Remove jumpers E85, E87, E88 if using external power
*External power supply inputs for opto-isolation from the digital ground plane.				

DB15 Style Connector J2 Bottom Amp – Out 2/Analog Power

Pin#	Symbol	Function	Description	Notes
1	DAC2A+	Output	Phase A Analog Out	+/-10V, ref to AGND
2	DAC2B+	Output	Phase B Analog Out	+/-10V, ref to AGND
3	AE_NC_2	Output	Amplifier Enable	Normally closed
4	AE_NO_2	Output	Amplifier Enable	Normally open
5	AFAULT_2-	Input		
6	AA-15V	Input*	Analog Negative Supply Voltage	Remove jumpers E85, E87, E88 if using external power
7	AAGND	Input*	Analog Reference Voltage	Remove jumpers E85, E87, E88 if using external power
8	AA-15V	Input*	Analog Negative Supply Voltage	Remove jumpers E85, E87, E88 if using external power
9	DAC2A-	Output	Phase A Analog Out	-/+10V; ref to AGND
10	DAC2B-	Output	Phase B Analog Out	-/+10V; ref to AGND
11	AE_COM_2	Input	Amplifier Enable	
12	AFAULT_2+	Input		
13	AA+15V	Input*	Analog Positive Supply Voltage	Remove jumpers E85, E87, E88 if using external power
14	AAGND	Input*	Analog Reference Voltage	Remove jumpers E85, E87, E88 if using external power
15	AA+15V	Input*	Analog Positive Supply Voltage	Remove jumpers E85, E87, E88 if using external power
*External power supply inputs for opto-isolation from the digital ground plane.				

Connector TB1 Front-Limits 1

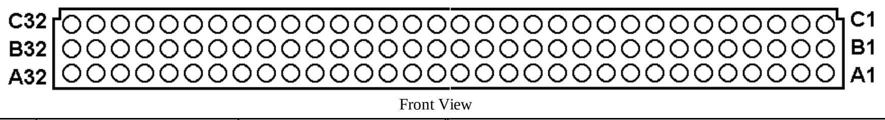
Pin#	Symbol	Function	Description	Notes
1	USER1	Input	General Capture Flag	Sinking or sourcing
2	PLIM1	Input	Positive Limit Flag	Sinking or sourcing
3	MLIM1	Input	Negative Limit Flag	Sinking or sourcing
4	HOME1	Input	Home Flag	Sinking or sourcing
5	FLG_1_RET	Input	Return For All Flags	+V (12 to 24V) or 0V

Connector TB2 Front-Limits 2

Pin#	Symbol	Function	Description	Notes
1	USER2	Input	General Capture Flag	Sinking or sourcing
2	PLIM2	Input	Positive Limit Flag	Sinking or sourcing
3	MLIM2	Input	Negative Limit Flag	Sinking or sourcing
4	HOME2	Input	Home Flag	Sinking or sourcing
5	FLG_2_RET	Input	Return For All Flags	+V (12 to 24V) or 0V

UBUS PINOUTS

P1 UBUS (96-Pin Header)



Pin #	Row A	Row B	Row C
1	+5Vdc	+5Vdc	+5Vdc
2	GND	GND	GND
3	BD01	DAT0	BD00
4	BD03	SEL0	BD02
5	BD05	DAT1	BD04
6	BD07	SEL1	BD06
7	BD09	DAT2	BD08
8	BD11	SEL2	BD10
9	BD13	DAT3	BD12
10	BD15	SEL3	BD14
11	BD17	DAT4	BD16
12	BD19	SEL4	BD18
13	BD21	DAT5	BD20
14	BD23	SEL5	BD22
15	BS1	DAT6	BS0
16	BA01	SEL6	BA00
17	BA03	DAT7	BA02
18	BX/Y	SEL7	BA04
19	CS3-	BA06	CS2-
20	BA05	BA07	CS4-
21	CS12-	BA08	CS10-
22	CS16-	BA09	CS14-
23	BA13	BA10	BA12
24	BRD-	BA11	BWR-
25	BS3	MEMCS0-	BS2
26	WAIT-	MEMCS1-	RESET
27	PHASE+	IREQ1-	SERVO+
28	PHASE-	IREQ2-	SERVO-
29	ANALOG	GND IREQ3-	ANALOG GND
30	-15Vdc	PWRGND	+15Vdc
31	GND	GND	GND
32	+5Vdc	+5Vdc	+5Vdc

For more details about the JEXP, see the UBUS Specification Document.

DECLARATION OF CONFORMITY

Application of Council Directive: 89/336/EEC, 72/23/EEC

Manufacturers Name:	Delta Tau Data Systems, Inc.
Manufacturers Address:	21314 Lassen Street Chatsworth, CA 91311 USA

We, Delta Tau Data Systems, Inc. hereby declare that the product

Product Name:	Accessory 24E2A
Model Number:	603398

And all of its options conforms to the following standards:

EN61326: 1997	Electrical equipment for measurement, control, and laboratory use-EMC requirements
EN55011: 1998	Limits and methods of measurements of radio disturbance characteristics of information technology equipment
EN61010-1	Electrical equipment for measurement, control, and laboratory use- Safety requirements
EN61000-3-2 :1995 A14:1998	Limits for harmonic current emissions. Criteria A
EN61000-3-3: 1995	Limitation of voltage fluctuations an d flicker in low-voltage supply systems for equipment with rated current ≤ 16A. Criteria B.
EN61000-4-2:1995 A1: 1998	Electro Static Discharge immunity test. Criteria B
EN61000-4-3: 1995 A1: 1998	Radiated, radio-frequency, electromagnetic field immunity test. Criteria A
EN61000-4-4: 1995	Electrical fast transients/burst immunity test. Criteria B
EN61000-4-5: 1995	Surge Test. Criteria B
EN61000-4-6: 1996	Conducted immunity test. Criteria A
EN61000-4-11: 1994	Voltage dips test. Criteria B and C

Date Issued:	11 May 2006
Place Issued:	Chatsworth, California USA

Yolande Cano

Yolande Cano
Quality Assurance Manager

Mark of Compliance

CE

E-POINT JUMPER SETTINGS

ACC-24E2A Base Board (Channels* 1 and 2)

Jumper	Config.	Description	Default
E1A	1-2	No Jumper for TTL Level input for CHU1 flag Jumper 1-2 for DIR1+ output in Stepper Mode	No jumper
E1B	1-2	No Jumper for TTL Level input for CHV1 flag Jumper 1-2 for DIR1- output in Stepper Mode	No jumper
E1C	1-2	No Jumper for TTL Level input for CHW1 flag Jumper 1-2 for PUL1+ output in Stepper Mode	No jumper
E1D	1-2	No Jumper for TTL Level input for CHT1 flag Jumper 1-2 for PUL1- output in Stepper Mode	No jumper
E2A	1-2	No Jumper for TTL Level input for CHU2 flag Jumper 1-2 for DIR2+ output in Stepper Mode	No jumper
E2B	1-2	No Jumper for TTL Level input for CHV2 flag Jumper 1-2 for DIR2- output in Stepper Mode	No jumper
E2C	1-2	No Jumper for TTL Level input for CHW2 flag Jumper 1-2 for PUL2+ output in Stepper Mode	No jumper
E2D	1-2	No Jumper for TTL Level input for CHT2 flag Jumper 1-2 for PUL2- output in Stepper Mode	No jumper
E5	1-2-3	Jump 1-2 for Turbo 3U CPU and MACRO CPU ** Jump 2-3 for legacy MACRO CPU (before 6/00)	Jump 1-2
E13	1-2-3	Jump 1-2 to receive phase and servo clocks Jump 2-3 to transmit phase and servo clocks	Factory set
E85	1-2	Jump 1-2 for Backplane Supplied +15V No Jumper for External Supplied +15V	Jump 1-2
E87	1-2	Jump 1-2 for Backplane Supplied AGND No Jumper for External Supplied AGND	Jump 1-2
E88	1-2	Jump 1-2 for Backplane Supplied -15V No Jumper for External Supplied -15V	Jump 1-2
OPT1	1-2	For factory use only	
OPT2	1-2	For factory use only	

* The channels refer to the Servo IC associated with the ACC-24E2 base board. For example, an eight-axis application would have two ACC-24E2s with option 1. The first ACC-24E2 would have axes 1-4 and the second ACC-24E2 would contain axes 5-8.

** For legacy MACRO Stations (part number 602804-100 thru 602804-104)

ACC-24E2A Option 1 Board (Channels 3 and 4)

Jumper	Config.	Description	Default
E1A	1-2	No Jumper for TTL Level input for CHU3 flag Jumper 1-2 for DIR3+ output in Stepper Mode	No jumper
E1B	1-2	No Jumper for TTL Level input for CHV3 flag Jumper 1-2 for DIR3- output in Stepper Mode	No jumper
E1C	1-2	No Jumper for TTL Level input for CHW3 flag Jumper 1-2 for PUL3+ output in Stepper Mode	No jumper
E1D	1-2	No Jumper for TTL Level input for CHT3 flag Jumper 1-2 for PUL3- output in Stepper Mode	No jumper
E2A	1-2	No Jumper for TTL Level input for CHU4 flag Jumper 1-2 for DIR4+ output in Stepper Mode	No jumper
E2B	1-2	No Jumper for TTL Level input for CHV4 flag Jumper 1-2 for DIR4- output in Stepper Mode	No jumper
E2C	1-2	No Jumper for TTL Level input for CHW4 flag Jumper 1-2 for PUL4+ output in Stepper Mode	No jumper
E2D	1-2	No Jumper for TTL Level input for CHT4 flag Jumper 1-2 for PUL4- output in Stepper Mode	No jumper
E85	1-2	Jump 1-2 for Backplane Supplied +15V No Jumper for External Supplied +15V	Jump 1-2
E87	1-2	Jump 1-2 for Backplane Supplied AGND No Jumper for External Supplied AGND	Jump 1-2
E88	1-2	Jump 1-2 for Backplane Supplied -15V No Jumper for External Supplied -15V	Jump 1-2