



ACC-59E3

Hardware Reference Manual

Copyright Information

© 2025 Delta Tau Data Systems, Inc. All rights reserved.

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

To report errors or inconsistencies, call or email your local Omron representative.

Technical Support

Phone: Call 1-800-556-6766, and follow the prompts for technical support on PMAC products.

Email: ODT-Support@omron.com

Alternate Email: ia.techsupport@omron.com

Type "Delta Tau Support" or "PMAC Support" in the subject line.

Operating Conditions

All Delta Tau Data Systems, Inc. motion controller products, accessories, and amplifiers contain static sensitive components that can be damaged by incorrect handling. When installing or handling Delta Tau Data Systems, Inc. products, avoid contact with highly insulated materials. Only qualified personnel should be allowed to handle this equipment. Before powering, please ensure there is no visible damage to the product.

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. Our products should not be placed in locations that can accrue a lot of dust, salt, or conductive iron-like powder. When our products are used in an industrial environment, install them into an industrial electrical cabinet or industrial PC to protect them from excessive or corrosive moisture, abnormal ambient temperatures, and conductive materials. If Delta Tau Data Systems, Inc. products are directly exposed to hazardous or conductive materials and/or environments, we cannot guarantee their operation. For your own safety, please keep the product's environmental conditions within the range outlined by the Environment Specifications section that can be located from the table of contents in this manual.

If the equipment is used in a manner not specified by Delta Tau Data Systems, the protection provided by the equipment may be impaired.

Trademarks

Other company names and product names in this document are the trademarks or registered trademarks of their respective companies.



EN Dispose in accordance with applicable regulations.



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	Created manual	09/19/11	DCDP	CW
2	Added clocking instructions and structure references; corrected jumper settings	10/19/12	DCDP	RN
3	Corrected mistake in analog input filtering section	07/10/13	DCDP	RN
4	Modified some plc examples	01/28/14	DCDP	SS
5	Corrected jumper settings for analog inputs; corrected mistakes in C-programming section	05/19/14	DCDP	VB
6	Removed unnecessary settings from example header file	09/5/14	DCDP	SS
7	Corrected analog output wiring diagrams	04/19/16	DCDP	TS
8	Added KC conformity	10/17/18	SM	RN
9	Updated “Operating temperature” in environmental specifications	09/14/20	SM	RN
10	Added Mounting and Installation section	12/15/20	SM	RN
11	Updated part number configuration section	01/14/21	SM	RN
12	Updated electrical specifications	06/02/21	SM	RN
A	Added UKCA Marking to front cover and added description in Agency of Approval section	08/11/21	AE	SM
B	Updated UKCA standard	01/31/22	AE	SF
C	Corrected fuse size information	09/12/25	TDO	AA

Table of Contents

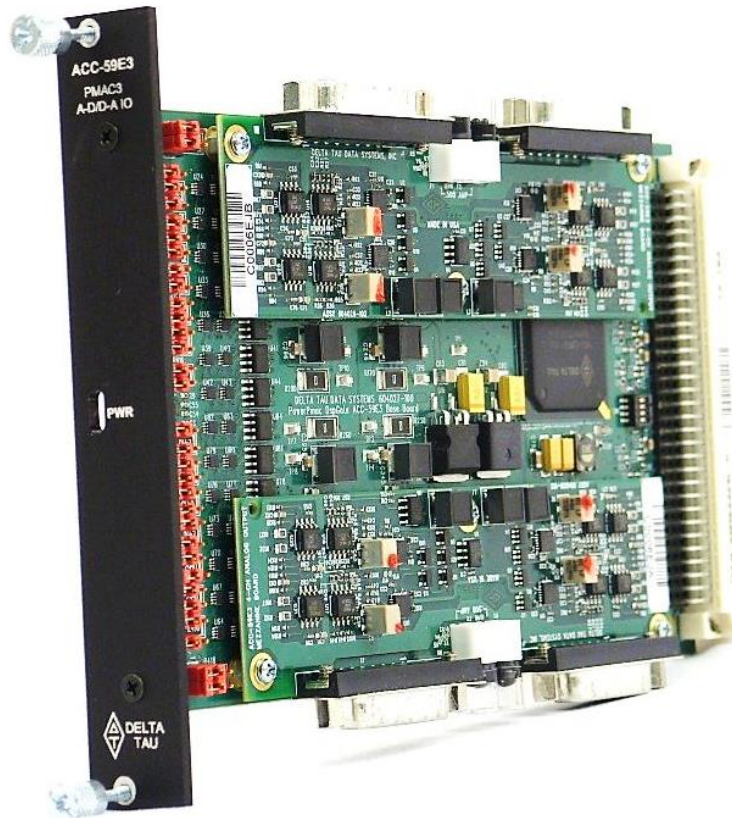
INTRODUCTION.....	6
SPECIFICATIONS.....	7
Environmental Specifications	7
Electrical Specifications.....	7
Agency Approval and Safety	8
Analog Input Specifications.....	9
Analog Voltage Output Specifications	10
Analog Current Output Specifications.....	10
Physical Specifications	10
Part Number	11
MOUNTING AND INSTALLATION	12
ADDRESSING ACC-59E3.....	13
ACC-59E3 Power UMAC Dip Switch Settings	13
USING ACC-59E3 WITH POWER UMAC	14
Initializing the ACC-59E3	14
<i>Standard Hardware Settings.....</i>	<i>14</i>
<i>Setting the Input Sampling Frequency.....</i>	<i>15</i>
Configuring the Analog Inputs	16
<i>Using Pointer Variables (M-Variables).....</i>	<i>16</i>
<i>Alternative Method: Using ADC Structures</i>	<i>19</i>
<i>Using an Analog Input for Servo Feedback.....</i>	<i>21</i>
<i>Analog Input Power-On Position.....</i>	<i>22</i>
<i>Testing the Analog Inputs</i>	<i>23</i>
<i>Example ADC Conversion PLCs</i>	<i>25</i>
Configuring the Analog Outputs.....	27
<i>Assigning Pointer Variables to Analog Output Registers.....</i>	<i>27</i>
<i>Writing to the Analog Outputs</i>	<i>28</i>
<i>Assigning Pointer Variables to Current Output Amplifier Enable Bits.....</i>	<i>30</i>
<i>Alternative Output Method: DAC Structures.....</i>	<i>33</i>
<i>Alternative Amplifier Enable Method: Amp Enable Structures.....</i>	<i>36</i>
Configuring the Analog Input Filtering	37
<i>Hardware Input Filters</i>	<i>37</i>
<i>Optional Software Low-Pass Filter</i>	<i>41</i>
ACC-59E3 LAYOUT & PINOUTS.....	44
P1: Backplane Bus	46
DB15 Breakout Option	46

<i>J1 Top</i>	46
<i>J2 Top</i>	47
<i>J2 Bottom</i>	48
<i>J1 Bottom</i>	49
Sample Wiring Diagrams.....	50
APPENDIX A: JUMPER SETTINGS.....	52
APPENDIX B: DAC OUTPUT OFFSET POTS	56
APPENDIX C: SCHEMATICS.....	57
<i>Connector Schematics</i>	57
<i>Analog Input Circuitry Schematics</i>	61
<i>Analog Voltage Output Circuitry Schematics</i>	62
<i>Analog Current Output Circuitry Schematics</i>	64
APPENDIX D: COMPLETE SUGGESTED M-VARIABLES.....	65
16-bit ADC Option	65
12-bit ADC Option	81
APPENDIX E: USING ACC-59E3 IN THE C ENVIRONMENT	97
<i>ACC-59E3 Header File (acc59e3.h)</i>	97
<i>ACC-59E3 Functions (Contained in acc59e3.h)</i>	106
<i>Examples</i>	111

INTRODUCTION

The Accessory 59E3 (ACC-59E3) is a 3U Euro bus compatible card for the Power UMAC rack. The board can be configured to have 16 Channels of 12- or 16-Bit Analog to Digital Converters (ADCs) for reading +/- 10 V voltage inputs or 4–20 mA current inputs and 8 Channels of 16-Bit Digital to Analog Converters (DACs) for voltage outputs or current outputs. The board has user-selectable sampling rates up to 30 kHz when reading all 16 channels of ADCs. Below is a picture of the board:

**ACC-59E3:
D-Sub Option**



Compared to the ACC-59E, an older but similar type of card, here are the ACC-59E3's advantages:

- ACC-59E3 can have 16-bit inputs and outputs, compared to ACC-59E's 12-bit only inputs and outputs
- ACC-59E3 can have 16 inputs, rather than ACC-59E's limit of 8 inputs
- ACC-59E3 can receive current inputs, whereas ACC-59E cannot
- ACC-59E3 has user-programmable hardware input filters, whereas ACC-59E has only one set input cutoff frequency

The only disadvantage of ACC-59E3 versus ACC-59E is that ACC-59E3 can be used only with Power PMAC, whereas ACC-59E can be used with Turbo or Power PMAC.

SPECIFICATIONS

Environmental Specifications

Description	Specification	Notes
Operating Temperature	0 °C to 55 °C	
Storage Temperature	-25 °C to 70 °C	
Humidity	5% to 95%	Non-Condensing Relative Humidity

Electrical Specifications

Power Requirements

Backplane Voltage	Current Required
+5VDC	50 mA
+15VDC	400 mA
-15VDC	175 mA

ACC-59E3 Fuse

Fuse Specification	Manufacturer	Part Number
125 V @ 0.5 A	Littelfuse	273.500

Agency Approval and Safety

Item	Description
CE Mark	EN61326-1
EMC	EN55011 Class A Group 1 EN61000-4-2 EN61000-4-3 EN61000-4-4 EN61000-4-5 EN61000-4-6
Flammability Class	UL 94V-0
KC	EMI: KN 11 EMS: KN 61000-6-2
UKCA	2016 No. 1091

사 용 자 안 내 문

이 기기는 업무용 환경에서 사용할 목적으로 적합성평가를 받은 기기로서 가정용 환경에서 사용하는 경우 전파간섭의 우려가 있습니다.

한국 EMC적용제품 준수사항

본 제품은 전파법(KC 규정)을 준수합니다. 제품을 사용하려면 다음 사항에 유의하십시오. 이 기기는 업무용 환경에서 사용할 목적으로 적합성평가를 받은 기기로서 가정용 환경에서 사용하는 경우 전파간섭의 우려가 있습니다. 입력에 EMC 필터, 서지 보호기, 페라이트 코어 또는 1차측의 케이블에 노이즈 필터를 입력으로 사용하십시오.

Analog Input Specifications

The analog inputs are provided in the base configuration of the ACC-59E3. Selection of 12-bit versus 16-bit resolution must be made when the product is ordered. Each input is selectable by the user for voltage mode or current mode by placement of hardware jumpers.

Item	Specification
# of analog inputs	16
Input type	Differential
Isolation	None
Operational input range Voltage mode Current mode	± 10 VDC (between input pins) ± 20 mA (at each input pin)
Input resolution 12-bit input version 16-bit input version	12 bits (voltage mode) over ± 10 VDC input range 12 bits (current mode) over ± 20 mA input range 10.5 bits (current mode) over 4–20 mA input range 16 bits (voltage mode) over ± 10 VDC input range 16 bits (current mode) over ± 20 mA input range 14.5 bits (current mode) over 4–20 mA input range
Sample rate	≤ 50 kHz (software selectable)
Input impedance	56.2 k Ω (voltage mode) 500 Ω (current mode)
Input filter attenuation (–3 dB frequency)	3.2 kHz, 4.3 kHz, 12.2 kHz, 24.5 kHz (software selectable) Further filtering possible with software filtering structures
Repeatability	$\pm 0.05\%$ of full scale (12-bit input version) $\pm 0.12\%$ of 4–20 mA range (12-bit input version) $\pm 0.04\%$ of full scale (16-bit input version) $\pm 0.10\%$ of 4–20 mA range (16-bit input version)
Mean accuracy over temperature range (after calibration of input range; includes INL and drift)	$\pm 0.075\%$ of full scale (12-bit input version) $\pm 0.15\%$ of 4–20 mA range (12-bit input version) $\pm 0.018\%$ of full scale (16-bit input version) $\pm 0.04\%$ of 4–20 mA range (16-bit input range)
Maximum voltage	± 12 V between input pins
ADC Chips	ADS8361: 16-bit ADC ADS7861: 12-bit ADC

Analog Voltage Output Specifications

The voltage-mode analog outputs are an optional component of the ACC-59E3 that must be specifically ordered.

Item	Specification
# of analog outputs	8
Output type	Differential
Isolation	None
Operational output range	± 10 VDC (between each pin and GND)
Output resolution	16 bits over ± 10 V output range
Update rate	≤ 50 kHz (software selectable)
Output impedance	100 Ω
Minimum load impedance	5 k Ω
Accuracy over temperature range	$\pm 0.45\%$ of full scale
Settling time	8 μ sec (to ± 1 LSB)
Slew rate	5 VDC/ μ sec
DAC Output Chips	DAC8881: 16-bit DAC DAC9881: 18-bit DAC

Analog Current Output Specifications

The current-mode analog outputs are an optional component of the ACC-59E3 that must be specifically ordered.

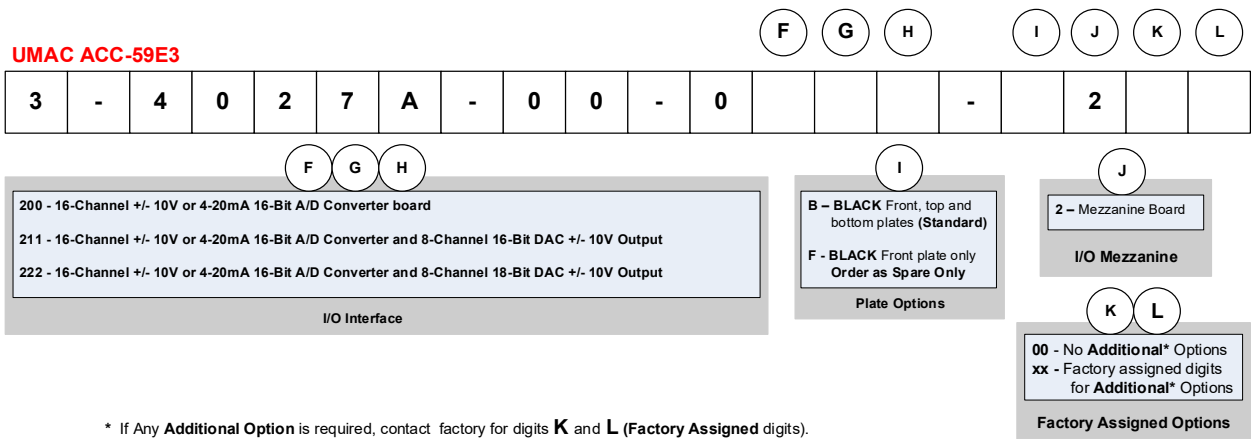
Item	Specification
# of analog outputs	8
Output type	Differential
Isolation	Optical
Operational output range	4–20 mA
Output resolution	16 bits over 4–20 mA output range
Update rate	≤ 50 kHz (software selectable)
Accuracy over temperature range	$\pm 0.45\%$ of full scale
Settling time	10 μ sec (to ± 1 LSB)
Slew rate	3 mA/ μ sec
Current Output Chip	XTR117: 16-bit Current Output

Physical Specifications

Description	Specification	Notes
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
Dimensions	100 mm x 160 mm x 20 mm	

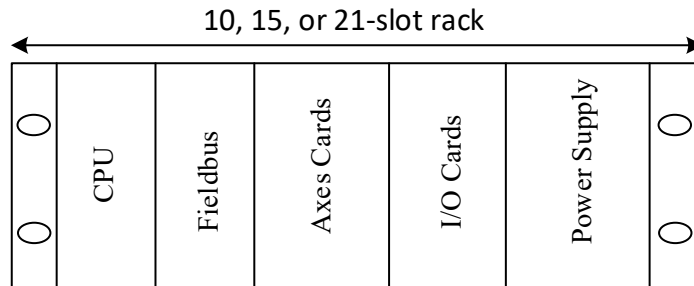
Part Number

The part number for ACC-59E3 can be generated according to the following diagram:



MOUNTING AND INSTALLATION

To connect a UMAC accessory, simply slide the board into any open slot of the UMAC rack. Customarily, accessories are installed from left to right as follows:



Prior to installation, make sure that you have set the jumpers and address settings to your desired requirements. Use the guide tracks that have been installed in the empty slots of your UMAC system when installing a board.

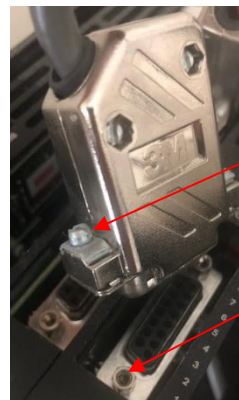
As you slide the board into the rack, use caution to ensure none of the components on the board make contact with the front plates of the boards on either side. Getting the front plate flush with the front of the rack and turning the front screws firmly will ensure a good connection with the backplane.

When removing a board from the system, the user must first pull out any wired connections from the top, bottom, and front panels then loosen the pem-nuts on the front of the rack. Next, the user can gently pull the board from the rack and use caution to ensure that none of the components on the board make contact with the boards on either side.



Caution

This product contains D-SUB style connectors. Do not overtighten any screws on mating connectors, and when possible, only tighten by hand. Overtightening, such as with manual or electric tools, may lead to the mating nut detaching when subsequently unscrewed, which may result in damage to the product, other electronics, and/or injury.



Screw

Hex nut
(Risk of falling)

ADDRESSING ACC-59E3

Power PMAC UMAC systems can accept several types of accessory boards that use the PMAC3-style “DSPGATE3” Machine-Interface IC. These include the ACC-24E3, ACC-5E3, and ACC-14E3. Each board in this class must have a unique DIP-switch setting in a single UMAC rack; otherwise, addressing conflicts will occur. ACC-59E3 is one of these boards.

Dip switch SW1 specifies the I/O base address offset of the ACC-59E3 in a 3U POWER UMAC rack. The following table shows the possible switch settings, the resulting address offsets, and the **Gate3[i]** index values “i”.

ACC-59E3 Power UMAC Dip Switch Settings

SW1-1 (Address bit 11)	SW1-2 (Address bit 12)	SW1-3 (Address bit 13)	SW1-4 (Address bit 14)	Power PMAC “Gate3” Index #i	Power PMAC I/O Base Address Offset
OFF(0)	OFF(0)	OFF(0)	OFF(0)	0	\$900000
ON(1)	OFF(0)	OFF(0)	OFF(0)	1	\$904000
OFF(0)	ON(1)	OFF(0)	OFF(0)	2	\$908000
ON(1)	ON(1)	OFF(0)	OFF(0)	3	\$90C000
OFF(0)	OFF(0)	ON(1)	OFF(0)	4	\$910000
ON(1)	OFF(0)	ON(1)	OFF(0)	5	\$914000
OFF(0)	ON(1)	ON(1)	OFF(0)	6	\$918000
ON(1)	ON(1)	ON(1)	OFF(0)	7	\$91C000
OFF(0)	OFF(0)	OFF(0)	ON(1)	8	\$920000
ON(1)	OFF(0)	OFF(0)	ON(1)	9	\$924000
OFF(0)	ON(1)	OFF(0)	ON(1)	10	\$928000
ON(1)	ON(1)	OFF(0)	ON(1)	11	\$92C000
OFF(0)	OFF(0)	ON(1)	ON(1)	12	\$930000
ON(1)	OFF(0)	ON(1)	ON(1)	13	\$934000
OFF(0)	ON(1)	ON(1)	ON(1)	14	\$938000
ON(1)	ON(1)	ON(1)	ON(1)	15	\$93C000



Note

- ON designates Closed. OFF designates Open.
- Factory default is all OFF.
- The maximum addressable number of PMAC3-style cards, including ACC-59E3, in one rack is 16.

USING ACC-59E3 WITH POWER UMAC

Initializing the ACC-59E3

Standard Hardware Settings

The ACC-59E3 requires a specific set of settings to be programmed once into its Gate3 ASIC and then saved. To do this, for each ACC-59E3 card, the user should make a configuration file to place into his or her project. The recommended location in the project tree is PMAC Script Language→Global Includes. The user need only copy the example below and change the card index from **ACC59E3[0]** as given in the example to the user's actual card index (**ACC59E3[1]** for Card Index 1, for example).

Example:

Configuring one ACC-59E3 as Card 0 (Power PMAC I/O Base Address Offset=\$900000). These settings can be stored in a file located in the IDE project under PMAC Script Language→Global Includes named “acc59E3_0_initialization.pmh”. Create this file if it does not exist already. These settings could alternatively be placed into a .cfg file in the Configuration folder in the IDE project.

```
// ACC-59E3 Initialization File
// For Card 0 (Power PMAC I/O Base Address Offset=$900000)
// Requires firmware 1.2.1.158 or newer.
Sys.WpKey=$AAAAAAA;

Acc59E3[0].DacStrobe=$FFFF0000; // Up to 16-bit DACs (works for 12-bit as well)
Acc59E3[0].GpioDir[0]=$FFFFFFFF; // All GPIO is output
Acc59E3[0].GpioPol[0]=0; // Non-inverting IO
Acc59E3[0].GpioData[0]=0; // All input filters set to 24.5 kHz cutoff freq.

Acc59E3[0].Chan[0].OutputMode=7; // Dual analog DAC output
Acc59E3[0].Chan[1].OutputMode=7; // Dual analog DAC output
Acc59E3[0].Chan[2].OutputMode=7; // Dual analog DAC output
Acc59E3[0].Chan[3].OutputMode=7; // Dual analog DAC output

Acc59E3[0].Chan[0].PackInData=0; // For AdcAmp, do not pack data
Acc59E3[0].Chan[1].PackInData=0; // For AdcAmp, do not pack data
Acc59E3[0].Chan[2].PackInData=0; // For AdcAmp, do not pack data
Acc59E3[0].Chan[3].PackInData=0; // For AdcAmp, do not pack data

Acc59E3[0].Chan[0].PackOutData=0;
Acc59E3[0].Chan[1].PackOutData=0;
Acc59E3[0].Chan[2].PackOutData=0;
Acc59E3[0].Chan[3].PackOutData=0;

Acc59E3[0].AdcAmpStrobe=$FFFFFFFC;
Acc59E3[0].AdcAmpHeaderBits=1;
```

Setting the Input Sampling Frequency

The input sampling frequency (i.e. the sampling frequency of the ADCs) on ACC-59E3 is determined by (and equal to) the global phase clock of the Power UMAC in which the ACC-59E3 has been installed. This phase clock can be generated by Gate1-, Gate2-, and Gate3-style axis control circuitry. For instructions on how to set the phase clock, please refer to the section labeled “Setting System Clock Frequencies” in the Power PMAC User’s Manual.

Configuring the Analog Inputs

Using Pointer Variables (M-Variables)

To use the Analog Inputs on ACC-59E3, follow this procedure:

1. Assign signed pointer variables (signed M-Variables) to the memory registers where the input results are stored.
2. Access the variables wherever and whenever desired, whether in the Terminal Window, a Script PLC, a Motion Program, or a CPLC, just like any M-Variable.
3. Scale the values to physical quantities (i.e. voltage [Volts] or current [mA]) if desired.

The memory registers to which the user must point the signed M-Variables can be found at the following locations:

Analog Input Channel Number	Memory Register	Bits (for 16-bit ADCs)	Bits (for 12-bit ADCs)
1	Power PMAC I/O Base Address Offset + \$20	[31:16]	[28:16]
2	Power PMAC I/O Base Address Offset + \$24	[31:16]	[28:16]
3	Power PMAC I/O Base Address Offset + \$28	[31:16]	[28:16]
4	Power PMAC I/O Base Address Offset + \$2C	[31:16]	[28:16]
5	Power PMAC I/O Base Address Offset + \$A0	[31:16]	[28:16]
6	Power PMAC I/O Base Address Offset + \$A4	[31:16]	[28:16]
7	Power PMAC I/O Base Address Offset + \$A8	[31:16]	[28:16]
8	Power PMAC I/O Base Address Offset + \$AC	[31:16]	[28:16]
9	Power PMAC I/O Base Address Offset + \$120	[31:16]	[31:20]
10	Power PMAC I/O Base Address Offset + \$124	[31:16]	[28:16]
11	Power PMAC I/O Base Address Offset + \$128	[31:16]	[28:16]
12	Power PMAC I/O Base Address Offset + \$12C	[31:16]	[28:16]
13	Power PMAC I/O Base Address Offset + \$1A0	[31:16]	[28:16]
14	Power PMAC I/O Base Address Offset + \$1A4	[31:16]	[28:16]
15	Power PMAC I/O Base Address Offset + \$1A8	[31:16]	[28:16]
16	Power PMAC I/O Base Address Offset + \$1AC	[31:16]	[28:16]



Note

- The user must define the M-Variable differently depending on whether he or she has ordered the 16-bit ADC option or the 12-bit ADC option. See the examples below for more details.
- See Appendix D for a full list of Suggested M-Variables for every possible card index.

Example M-Variables:

For a single ACC-59E3 configured as Card 0 (Power PMAC I/O Base Address Offset=\$900000) with 16-bit ADCs.

```
// ADC +/-32767 for +/-5V differential, +/-10V single ended
ptr ACC59E3_0_IN1->s.io:$900020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_0_IN2->s.io:$900024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_0_IN3->s.io:$900028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_0_IN4->s.io:$90002C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_0_IN5->s.io:$9000A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_0_IN6->s.io:$9000A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_0_IN7->s.io:$9000A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_0_IN8->s.io:$9000AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_0_IN9->s.io:$900120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_0_IN10->s.io:$900124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_0_IN11->s.io:$900128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_0_IN12->s.io:$90012C.16.16; // ACC-59E3 J1 IN12- Bottom

ptr ACC59E3_0_IN13->s.io:$9001A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_0_IN14->s.io:$9001A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_0_IN15->s.io:$9001A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_0_IN16->s.io:$9001AC.16.16; // ACC-59E3 J2 IN16 - Bottom
```

Example M-Variables:

For a single ACC-59E3 configured as Card 0 (Power PMAC I/O Base Address Offset=\$900000) with 12-bit ADCs.

```
// ADC +/-32767 for +/-5V differential, +/-10V single ended
ptr ACC59E3_0_IN1->s.io:$900020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_0_IN2->s.io:$900024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_0_IN3->s.io:$900028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_0_IN4->s.io:$90002C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_0_IN5->s.io:$9000A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_0_IN6->s.io:$9000A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_0_IN7->s.io:$9000A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_0_IN8->s.io:$9000AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_0_IN9->s.io:$900120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_0_IN10->s.io:$900124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_0_IN11->s.io:$900128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_0_IN12->s.io:$90012C.16.12; // ACC-59E3 J1 IN12- Bottom

ptr ACC59E3_0_IN13->s.io:$9001A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_0_IN14->s.io:$9001A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_0_IN15->s.io:$9001A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_0_IN16->s.io:$9001AC.16.12; // ACC-59E3 J2 IN16 - Bottom
```

Now the user can access the inputs by the above pointer names for each channel.



Note

The M-Variable definitions for 16-bit ADCs start at bit 16 and have a width of 16 bits; for 12-bit ADCs, the definitions start at bit 16 and have a width of 12 bits.

Example: Using the ADC Pointers in a PLC

This PLC will copy the ADC results from the M-Variables given in the example above to a 16-element array of P-Variables, ADC(0) through ADC(15). This PLC will run in the background so that the user can just take the results from the P-Variables and do whatever he or she wants with them whenever he or she wants.

```
global ADC(16);          // Allocate array of 16 P-Variables to store the ADC results
open plc 4               // Open PLC 4
// Copy the ADC results from the M-Variables to the P-Variable Array
ADC(0)=ACC59E3_0_IN1;
ADC(1)=ACC59E3_0_IN2;
ADC(2)=ACC59E3_0_IN3;
ADC(3)=ACC59E3_0_IN4;
ADC(4)=ACC59E3_0_IN5;
ADC(5)=ACC59E3_0_IN6;
ADC(6)=ACC59E3_0_IN7;
ADC(7)=ACC59E3_0_IN8;
ADC(8)=ACC59E3_0_IN9;
ADC(9)=ACC59E3_0_IN10;
ADC(10)=ACC59E3_0_IN11;
ADC(11)=ACC59E3_0_IN12;
ADC(12)=ACC59E3_0_IN13;
ADC(13)=ACC59E3_0_IN14;
ADC(14)=ACC59E3_0_IN15;
ADC(15)=ACC59E3_0_IN16;
close                    // Close PLC 4
```

**Note**

This is simply an example. It is not necessary to first copy the **ptr** variables' values into **global** variables before using them.

Alternative Method: Using ADC Structures

An alternative method to accessing the ADCs on ACC-59E3 is through Gate3 memory structures. The advantage of this method is that there is no need to map the memory registers with M-Variables. The disadvantage is that you must manually divide these structures by 2^{16} before using their values.

The structures corresponding to each ADC channel is given in the table below, where n represents the card index as given in the “Addressing ACC-59E3” section of this manual:

Analog Input Channel Number	Structure	Bits (for 16-bit ADCs)	Bits (for 12-bit ADCs)
1	ACC59E3[i].Chan[0].ADCAmp[0]	[31:16]	[28:16]
2	ACC59E3[i].Chan[0].ADCAmp[1]	[31:16]	[28:16]
3	ACC59E3[i].Chan[0].ADCAmp[2]	[31:16]	[28:16]
4	ACC59E3[i].Chan[0].ADCAmp[3]	[31:16]	[28:16]
5	ACC59E3[i].Chan[1].ADCAmp[1]	[31:16]	[28:16]
6	ACC59E3[i].Chan[1].ADCAmp[2]	[31:16]	[28:16]
7	ACC59E3[i].Chan[1].ADCAmp[3]	[31:16]	[28:16]
8	ACC59E3[i].Chan[1].ADCAmp[4]	[31:16]	[28:16]
9	ACC59E3[i].Chan[2].ADCAmp[0]	[31:16]	[31:20]
10	ACC59E3[i].Chan[2].ADCAmp[1]	[31:16]	[28:16]
11	ACC59E3[i].Chan[2].ADCAmp[2]	[31:16]	[28:16]
12	ACC59E3[i].Chan[2].ADCAmp[3]	[31:16]	[28:16]
13	ACC59E3[i].Chan[3].ADCAmp[0]	[31:16]	[28:16]
14	ACC59E3[i].Chan[3].ADCAmp[1]	[31:16]	[28:16]
15	ACC59E3[i].Chan[3].ADCAmp[2]	[31:16]	[28:16]
16	ACC59E3[i].Chan[3].ADCAmp[3]	[31:16]	[28:16]

Example: Using the ADC Structures in a PLC

This PLC will copy the ADC results from the M-Variables given in the example above to a 16-element array of P-Variables, ADC(0) through ADC(15). This PLC will run in the background so that the user can just take the results from the P-Variables and do whatever he or she wants with them whenever he or she wants.

```
global ADC(16); // Allocate array of 16 P-Variables to store the ADC results
global ScaleFactor=1/exp2(16); // Scale factor of 2^16
open plc 4 // Open PLC 4
// Copy the ADC results from the M-Variables to the P-Variable Array
ADC(0)=ACC59E3[0].Chan[0].AdcAmp[0]*ScaleFactor;
ADC(1)= ACC59E3[0].Chan[0].AdcAmp[1]*ScaleFactor;
ADC(2)= ACC59E3[0].Chan[0].AdcAmp[2]*ScaleFactor;
ADC(3)= ACC59E3[0].Chan[0].AdcAmp[3]*ScaleFactor;
ADC(4)= ACC59E3[0].Chan[1].AdcAmp[0]*ScaleFactor;
ADC(5)= ACC59E3[0].Chan[1].AdcAmp[1]*ScaleFactor;
ADC(6)= ACC59E3[0].Chan[1].AdcAmp[2]*ScaleFactor;
ADC(7)= ACC59E3[0].Chan[1].AdcAmp[3]*ScaleFactor;
ADC(8)= ACC59E3[0].Chan[2].AdcAmp[0]*ScaleFactor;
ADC(9)= ACC59E3[0].Chan[2].AdcAmp[1]*ScaleFactor;
ADC(10)= ACC59E3[0].Chan[2].AdcAmp[2]*ScaleFactor;
ADC(11)= ACC59E3[0].Chan[2].AdcAmp[3]*ScaleFactor;
ADC(12)= ACC59E3[0].Chan[3].AdcAmp[0]*ScaleFactor;
ADC(13)= ACC59E3[0].Chan[3].AdcAmp[1]*ScaleFactor;
ADC(14)= ACC59E3[0].Chan[3].AdcAmp[2]*ScaleFactor;
ADC(15)= ACC59E3[0].Chan[3].AdcAmp[3]*ScaleFactor;
close // Close PLC 4
```

Using an Analog Input for Servo Feedback

The ACC-59E3 analog inputs can be used as a feedback device for a servo motor. This example assumes that the ADCs have already been configured and are using the suggested pointer variable names.

Example: Setting up Motor #1 with position and velocity feedback from ADC channel 1, assuming the 16-bit ADC option. The analog input is brought into the Encoder Conversion Table (ECT) as a single read of a 32-bit register (in the Power PMAC IDE Software: Delta Tau → Configure → Encoder Conversion Table):

ECT Setup: Online[192.168.0.211:SSH]

ECT entry number: 1

Type: 1: Single (32-bit) register read

Download

Start update

Display All ECT Entries

ECT entry input: 0

ECT entry output: 0

Detailed ECT Setup | PowerPMAC Structure | Encoder Plot

ECT entry Details

pEnc	Acc59E3[0].Chan[0].AdcAmp[0].a	PrevEnc	
pEnc1		PrevDelta	0
Index1	16	MaxDelta	0
Index2	16	DeltaPos	
Index3	0	SinBias	
Index4	0	CosBias	
ScaleFactor	0.0000152587890625	Counter	

Number	Type	pEnc	pEnc1	MaxDelta
1	1	Acc59E3[0].Chan[0].AdcAmp[0].a	Acc24E2A[4].Chan[0].TimeBetweenCts.a	0

The equivalent code in Power PMAC Structures:

```
EncTable[1].type = 1; // Set entry type to 32-bit word read
EncTable[1].pEnc = Sys.piom + $900020; // Set encoder address to ADC channel 1
EncTable[1].pEnc1 = Sys.pushm; // Unused; set to Sys.pushm
EncTable[1].index1 = 16; // Shift left 16 to put source MSB in bit 31
EncTable[1].index2 = 16; // Shift right 16 to put source LSB into bit 0
EncTable[1].index3 = 0; // No limit on first derivative
EncTable[1].index4 = 0; // No limit on second derivative
EncTable[1].ScaleFactor = 1/exp2(EncTable[1].index1); // Scale factor (1/(2^16))
```



Note

To clarify the example above: **Acc59E3[0].Chan[0].AdcAmp[0].a** is equivalent to **Sys.piom + \$900020**, and $1/\text{exp2}(\text{EncTable}[1].\text{index1}) = 1/\text{exp2}(16) = 0.0000152587890625$.

The position and velocity pointers are then set to the processed data address:

```
Motor[1].pEnc = EncTable[1].a; // Outer (position) loop source address
Motor[1].pEnc2 = EncTable[1].a; // Inner (velocity) loop source address
```

The user can then adjust **Motor[1].PosSf** and **Motor[1].Pos2Sf**, the position and velocity scale factors, respectively, if necessary.

Analog Input Power-On Position

Some analog devices are absolute along the travel range of the motor (e.g., in hydraulic piston applications). Generally, it is desirable to obtain the motor position (input voltage) on power up or reset.

Example:

Configuring Motor #1 to use ADC Channel #1 for the power-on position read as a signed value with a scale factor of 1.0:

```
Motor[1].pAbsPos = Sys.piom + $900020;      // Set position register to ADC channel 1
Motor[1].AbsPosFormat = $01001010;         // Use 16 bits starting at bit 16, signed
Motor[1].AbsPosSF = 1.0;                    // Scale factor of 1.0
```

Please refer to the description of **Motor[x].AbsPosFormat** in the Power PMAC Saved Data Structure Elements manual for further details on how to set this structure for motor *x*.

Testing the Analog Inputs

The Analog Inputs can be brought into the ACC-59E3 as single ended (IN+ & Ground) or differential (IN+ & IN-) signals.



Note

In single-ended mode, IN- should be tied to analog ground for full resolution and proper operation.

If the user has selected Voltage Inputs, reading the input signals in “software counts” S_{in} (either directly, with M-Variables, or with the ACC59E3 structure divided by 2^{16}) should show the following within error tolerances:

ADC Input	Single-Ended [V] IN+ □□ AGND	Differential [V] IN+ □□ IN-	Software Counts for 16-bit ADCs	Software Counts for 12-bit ADCs
Bipolar Mode	-10	-10	-32558	-2034
	0	0	0	0
	10	10	32558	2034

For 16-bit ADCs, the nominal equation to convert from software counts S_{in} to input voltage V_{in} [V] is:

$$V_{in} = \frac{10.064}{32767} S_{in}.$$

For 12-bit ADCs, the nominal equation to convert from software counts S_{in} to input voltage V_{in} [V] is:

$$V_{in} = \frac{10.064}{2048} S_{in}.$$

If the user has selected Current Inputs, reading the input signals in “software counts” should show the following within error tolerances:

Current Input	Single-Ended [mA] IN+ □□ AGND	Differential [mA] IN+ □□ IN-	Software Counts for 16-bit ADCs	Software Counts for 12-bit ADCs
Bipolar Mode	-20	-20	-32558	-2034
	0	0	0	0
	20	20	32558	2034

For 16-bit ADCs, the nominal equation to convert from software counts S_{in} to input current C_{in} [mA] is:

$$C_{in} = \frac{20.128}{32767} S_{in}.$$

For 12-bit ADCs, the nominal equation to convert from software counts S_{in} to input current C_{in} [mA] is:

$$C_{in} = \frac{20.128}{2048} S_{in}.$$

A Note about the Conversion Equations

The above equations are called “nominal” equations because they are based off of the nominal gain calculations for the ACC-59E3’s output circuitry. The actual conversion equations can vary slightly within the $\pm 0.45\%$ of full scale variance with temperature, especially since actual test results tend to produce better accuracy levels than nominal. For the best accuracy, the user can calibrate this curve according to his or her temperature environment by observing the software count values corresponding to a known, steady voltage or current value the user is putting into the input channel.

Example: Calibrating Voltage Conversion Curve for Single-Ended 16-bit Voltage Inputs (ADCs)

1. Input a known, steady 10.00 V into ADC1. This value is y_1 .
2. Observe and record the software counts shown by the ACC59E3_0_IN1 M-Variable. This value is x_1 .
3. Input a known, steady 0.00 V into ADC1. This value is y_0 .
4. Observe and record the software counts shown by the ACC59E3_0_IN1 M-Variable. This value is x_0 .

Then the calibrated equation to convert from software counts S_{in} to input voltage V_{in} [V] is:

$$V_{in} = \frac{(y_1 - y_0)}{(x_1 - x_0)} S_{in} + y_0.$$



Note

Most users will not need to perform this additional calibration step, because the temperature variance in the ACC-59E3’s input circuitry is so slight.

Example ADC Conversion PLCs

First, you can place the scale factors and arrays containing converted values that you may need to use into “global definitions.pmh” under PMAC Script Language→Global Includes:

```
global Input_Volts(16);           // Array that can contain 16 voltage values
global Input_Current(16);         // Array that can contain 16 current values

global VoltsPerCt_16bit = 10.064/32767.0; // Volts per bit scale factor for 16-bit ADCs
global VoltsPerCt_12bit = 10.064/2048.0;  // Volts per bit scale factor for 12-bit ADCs
global mAPerCt_16bit = 20.128/32767.0;    // mA per bit scale factor for 16-bit ADCs
global mAPerCt_12bit = 20.128/2048.0;     // mA per bit scale factor for 12-bit ADCs
```

16-bit Voltage Inputs

```
open plc Voltage_16bit_Conversion
Input_Volts(0) = ACC59E3_0_IN1 * VoltsPerCt_16bit;
Input_Volts(1) = ACC59E3_0_IN2 * VoltsPerCt_16bit;
Input_Volts(2) = ACC59E3_0_IN3 * VoltsPerCt_16bit;
Input_Volts(3) = ACC59E3_0_IN4 * VoltsPerCt_16bit;

Input_Volts(4) = ACC59E3_0_IN5 * VoltsPerCt_16bit;
Input_Volts(5) = ACC59E3_0_IN6 * VoltsPerCt_16bit;
Input_Volts(6) = ACC59E3_0_IN7 * VoltsPerCt_16bit;
Input_Volts(7) = ACC59E3_0_IN8 * VoltsPerCt_16bit;

Input_Volts(8) = ACC59E3_0_IN9 * VoltsPerCt_16bit;
Input_Volts(9) = ACC59E3_0_IN10 * VoltsPerCt_16bit;
Input_Volts(10) = ACC59E3_0_IN11 * VoltsPerCt_16bit;
Input_Volts(11) = ACC59E3_0_IN12 * VoltsPerCt_16bit;

Input_Volts(12) = ACC59E3_0_IN13 * VoltsPerCt_16bit;
Input_Volts(13) = ACC59E3_0_IN14 * VoltsPerCt_16bit;
Input_Volts(14) = ACC59E3_0_IN15 * VoltsPerCt_16bit;
Input_Volts(15) = ACC59E3_0_IN16 * VoltsPerCt_16bit;
close
```

12-bit Voltage Inputs

```
open plc Voltage_12bit_Conversion
Input_Volts(0) = ACC59E3_0_IN1 * VoltsPerCt_12bit;
Input_Volts(1) = ACC59E3_0_IN2 * VoltsPerCt_12bit;
Input_Volts(2) = ACC59E3_0_IN3 * VoltsPerCt_12bit;
Input_Volts(3) = ACC59E3_0_IN4 * VoltsPerCt_12bit;

Input_Volts(4) = ACC59E3_0_IN5 * VoltsPerCt_12bit;
Input_Volts(5) = ACC59E3_0_IN6 * VoltsPerCt_12bit;
Input_Volts(6) = ACC59E3_0_IN7 * VoltsPerCt_12bit;
Input_Volts(7) = ACC59E3_0_IN8 * VoltsPerCt_12bit;

Input_Volts(8) = ACC59E3_0_IN9 * VoltsPerCt_12bit;
Input_Volts(9) = ACC59E3_0_IN10 * VoltsPerCt_12bit;
Input_Volts(10) = ACC59E3_0_IN11 * VoltsPerCt_12bit;
Input_Volts(11) = ACC59E3_0_IN12 * VoltsPerCt_12bit;

Input_Volts(12) = ACC59E3_0_IN13 * VoltsPerCt_12bit;
Input_Volts(13) = ACC59E3_0_IN14 * VoltsPerCt_12bit;
Input_Volts(14) = ACC59E3_0_IN15 * VoltsPerCt_12bit;
Input_Volts(15) = ACC59E3_0_IN16 * VoltsPerCt_12bit;
close
```

16-bit Current Inputs

```
open plc Current_16bit_Conversion
Input_Current(0) = ACC59E3_0_IN1 * mAPerCt_16bit;
Input_Current(1) = ACC59E3_0_IN2 * mAPerCt_16bit;
Input_Current(2) = ACC59E3_0_IN3 * mAPerCt_16bit;
Input_Current(3) = ACC59E3_0_IN4 * mAPerCt_16bit;

Input_Current(4) = ACC59E3_0_IN5 * mAPerCt_16bit;
Input_Current(5) = ACC59E3_0_IN6 * mAPerCt_16bit;
Input_Current(6) = ACC59E3_0_IN7 * mAPerCt_16bit;
Input_Current(7) = ACC59E3_0_IN8 * mAPerCt_16bit;

Input_Current(8) = ACC59E3_0_IN9 * mAPerCt_16bit;
Input_Current(9) = ACC59E3_0_IN10 * mAPerCt_16bit;
Input_Current(10) = ACC59E3_0_IN11 * mAPerCt_16bit;
Input_Current(11) = ACC59E3_0_IN12 * mAPerCt_16bit;

Input_Current(12) = ACC59E3_0_IN13 * mAPerCt_16bit;
Input_Current(13) = ACC59E3_0_IN14 * mAPerCt_16bit;
Input_Current(14) = ACC59E3_0_IN15 * mAPerCt_16bit;
Input_Current(15) = ACC59E3_0_IN16 * mAPerCt_16bit;
close
```

12-bit Current Inputs

```
open plc Current_12bit_Conversion
Input_Current(0) = ACC59E3_0_IN1 * mAPerCt_12bit;
Input_Current(1) = ACC59E3_0_IN2 * mAPerCt_12bit;
Input_Current(2) = ACC59E3_0_IN3 * mAPerCt_12bit;
Input_Current(3) = ACC59E3_0_IN4 * mAPerCt_12bit;

Input_Current(4) = ACC59E3_0_IN5 * mAPerCt_12bit;
Input_Current(5) = ACC59E3_0_IN6 * mAPerCt_12bit;
Input_Current(6) = ACC59E3_0_IN7 * mAPerCt_12bit;
Input_Current(7) = ACC59E3_0_IN8 * mAPerCt_12bit;

Input_Current(8) = ACC59E3_0_IN9 * mAPerCt_12bit;
Input_Current(9) = ACC59E3_0_IN10 * mAPerCt_12bit;
Input_Current(10) = ACC59E3_0_IN11 * mAPerCt_12bit;
Input_Current(11) = ACC59E3_0_IN12 * mAPerCt_12bit;

Input_Current(12) = ACC59E3_0_IN13 * mAPerCt_12bit;
Input_Current(13) = ACC59E3_0_IN14 * mAPerCt_12bit;
Input_Current(14) = ACC59E3_0_IN15 * mAPerCt_12bit;
Input_Current(15) = ACC59E3_0_IN16 * mAPerCt_12bit;
close
```

Configuring the Analog Outputs

To use the analog outputs (either voltage or current outputs) with ACC-59E3, the user must:

1. Assign pointer variables (M-Variables) to the analog output registers.
2. Write the appropriate “software counts” value to the register corresponding to the desired output voltage or current value.
3. If using the Current Output option, assign pointer variables (M-Variables) to the Amplifier Enable bits.
4. If using the Current Output option, enable the Amplifier Enable bits (i.e. set the bits =1).



Note

Steps 3 and 4 of the above procedure apply only when the user is using Current Output Mezzanine Boards (ASSY 604029-100 or newer).

Assigning Pointer Variables to Analog Output Registers

The analog output registers can be found at the following locations:

Analog Output Channel	Register	Bit Range
1	Power PMAC I/O Base Address Offset + \$40	[31:16]
2	Power PMAC I/O Base Address Offset + \$44	[31:16]
3	Power PMAC I/O Base Address Offset + \$C0	[31:16]
4	Power PMAC I/O Base Address Offset + \$C4	[31:16]
5	Power PMAC I/O Base Address Offset + \$140	[31:16]
6	Power PMAC I/O Base Address Offset + \$144	[31:16]
7	Power PMAC I/O Base Address Offset + \$1C0	[31:16]
8	Power PMAC I/O Base Address Offset + \$1C4	[31:16]

Example Pointer Variables (M-Variables):

For one ACC-59E3 at Power PMAC I/O Base Address Offset = \$900000 and eight DAC output channels.

```
// DAC Output Mezzanine on Top Side
ptr ACC59E3_0_OUT1->s.io:$900040.16.16;
ptr ACC59E3_0_OUT2->s.io:$900044.16.16;
ptr ACC59E3_0_OUT3->s.io:$9000C0.16.16;
ptr ACC59E3_0_OUT4->s.io:$9000C4.16.16;

// DAC Output Mezzanine on Bottom Side
ptr ACC59E3_0_OUT5->s.io:$900140.16.16;
ptr ACC59E3_0_OUT6->s.io:$900144.16.16;
ptr ACC59E3_0_OUT7->s.io:$9001C0.16.16;
ptr ACC59E3_0_OUT8->s.io:$9001C4.16.16;
```



Note

See Appendix D for a full list of Suggested M-Variables for every possible card index.

Writing to the Analog Outputs

The Analog Voltage Outputs and the Current Outputs out of the ACC-59E3 can both be wired as single ended (DAC+ & Ground) signals. The Analog Voltage Outputs can also be wired as differential (DAC+ & DAC-) signals, but the Current Outputs cannot; Current Outputs must be wired as single-ended outputs (OUT+ and GND) only. Regardless of whether one is using Script I/O pointers (M-Variables) or C pointers, writing the following “software counts” to the upper 16 bits of the corresponding output channel registers should result in the following output voltages if the user has the analog voltage output mezzanine boards (ASSY 604028-102 or newer) within error tolerances:

Voltage DAC Output	Single-Ended [V] DAC+ □□ AGND	Differential [V] DAC+ □□ DAC-	Software Counts (Write)
Bipolar Mode	-10	-20	-32558
	0	0	0
	10	20	32558

Thus, the nominal equation to convert desired voltage output V_{out} [volts] to software counts S_{out} when using single-ended wiring is:

$$S_{out} = \left(\frac{32767}{10.064} \right) \cdot V_{out}.$$

For differential wiring:

$$S_{out} = \left(\frac{32767}{20.128} \right) \cdot V_{out}.$$

If the user has the current output mezzanine boards, the following software counts will result in the corresponding current output values (ASSY 604029-101 or newer) within error tolerances:

Current DAC Output	Single-Ended [mA] DAC+ □□ AGND	Software Counts (Write)
Bipolar Mode	24	29478
	12.634	0
	4	-22394
	0	-32768

Thus, the nominal equation to convert desired current output C_{out} [mA] to software counts S_{out} is:

$$S_{out} = \left(\frac{16.21}{100} \right) \left(\frac{65536}{4.096} \right) C_{out} - 32768$$

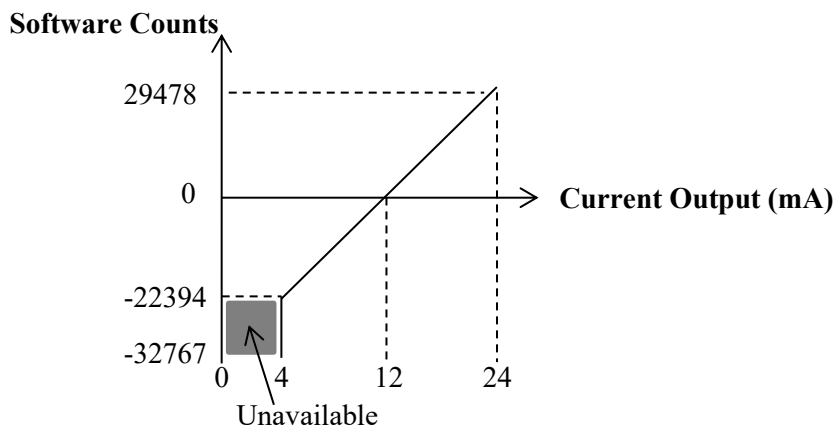
Inverting this equation, one obtains:

$$C_{out} = \left(\frac{4.096}{65536} \right) \left(\frac{100}{16.21} \right) (S_{out} + 32768)$$

**Note**

These equations are called “nominal” equations because they are based off of the nominal gain calculations for the ACC-59E3’s output circuitry. The actual conversion equations can vary slightly within the $\pm 0.45\%$ of full scale variance with temperature.

Note that the region of Current Output between 0 mA and 4 mA is not available:

**Note**

These numbers are for the latest revisions of the board or newer:

- ASSY 604028-102 for the voltage output mezzanine board
- ASSY 604029-101 for the current output mezzanine board

Assigning Pointer Variables to Current Output Amplifier Enable Bits

The Amplifier Enable bits for the Current Output circuitry are at the following memory registers:

Current Output Channels	Amplifier Enable Memory Register	Bit #
1–2	Power PMAC I/O Base Address Offset + \$6C	8
3–4	Power PMAC I/O Base Address Offset + \$EC	8
5–6	Power PMAC I/O Base Address Offset + \$16C	8
7–8	Power PMAC I/O Base Address Offset + \$1EC	8

Example: Current Output Amplifier Enable Suggested M-Variables

For ACC-59E3 addressed as Card 0.

```
ptr ACC59E3_0_AENA1->u.io:$90006C.8.1 // AENA1 output status (=1 output on, =0 output off)
ptr ACC59E3_0_AENA2->u.io:$9000EC.8.1 // AENA2 output status (=1 output on, =0 output off)
ptr ACC59E3_0_AENA3->u.io:$90016C.8.1 // AENA3 output status (=1 output on, =0 output off)
ptr ACC59E3_0_AENA4->u.io:$9001EC.8.1 // AENA4 output status (=1 output on, =0 output off)
```

Now whenever the user desires to enable his or her Current Output on a particular channel, he or she must set the Amplifier Enable bit corresponding to that channel equal to 1.

Example: Activating the Amplifier Enable Bit

```
ACC59E3_0_AENA1=1 // Enable the Current Output Amplifier for Output Channels 1-2.
```



Note

Using these bits is only necessary for current output, not for voltage output.

Example: Using ACC-59E3 DAC Pointers in “global definitions.pmh”

Commanding output from a single ACC-59E3 set at base offset \$900000 with bipolar differential outputs. Assumes the user downloaded the example pointers from the “Assigning Pointer Variables to DAC Output Registers” section of the manual.

```
// Bipolar DAC +/-32767 for +/-10V
ACC_59E3_0_OUT1=exp2(31)      // 10 V
ACC_59E3_0_OUT2=-exp2(31)    //-10 V
ACC_59E3_0_OUT3=exp2(30)     // 5 V
ACC_59E3_0_OUT4=-exp2(30)    //-5 V
ACC_59E3_0_OUT5=exp2(29)     // 2.5 V
ACC_59E3_0_OUT6=-exp2(29)    //-2.5 V
```

Example: Using Subprograms to Output Voltage

This example uses a subprogram to output voltage from the DACs of an ACC-59E3 addressed as Card 0.

```
global vosf=32767.0/10.064; // Voltage Output Scale Factor
// This subprogram converts the desired output voltage in "output_voltage" [Volts] to software counts
// and then commands the output on the specified channel in the "channel" input
open subprog OUTV(output_voltage,channel)
switch (channel){
  case 1:
    ACC59E3_0_OUT1=vosf*output_voltage;
    break;
  case 2:
    ACC59E3_0_OUT2=vosf*output_voltage;
    break;
  case 3:
    ACC59E3_0_OUT3=vosf*output_voltage;
    break;
  case 4:
    ACC59E3_0_OUT4=vosf*output_voltage;
    break;
  case 5:
    ACC59E3_0_OUT5=vosf*output_voltage;
    break;
  case 6:
    ACC59E3_0_OUT6=vosf*output_voltage;
    break;
  case 7:
    ACC59E3_0_OUT7=vosf*output_voltage;
    break;
  case 8:
    ACC59E3_0_OUT8=vosf*output_voltage;
    break;

  default:
    break;
}
close
```

The user then can call this function from a PLC, such as in the following example code:

```
global Desired_Output_Voltage;
Desired_Output_Voltage = 2.5; // Volts

open plc 10
local Channel_Number;
Channel_Number = 1;
while(Channel_Number < 9)
{
    call OUTV(Desired_Output_Voltage,Channel_Number);
    Channel_Number++;
}
close
```

Example: Using Subprograms to Output Current

This example uses a subprogram to output current from the current outputs of an ACC-59E3 addressed as Card 0.

```
global cosf=16.21/100.0*65536.0/4.096; // Current Output Scale Factor

open subprog OUTC(output_current,channel)
switch (channel){
    case 1:
        ACC59E3_0_OUT1=cosf*output_current-32768;
        break;
    case 2:
        ACC59E3_0_OUT2=cosf*output_current-32768;
        break;
    case 3:
        ACC59E3_0_OUT3=cosf*output_current-32768;
        break;
    case 4:
        ACC59E3_0_OUT4=cosf*output_current-32768;
        break;
    case 5:
        ACC59E3_0_OUT5=cosf*output_current-32768;
        break;
    case 6:
        ACC59E3_0_OUT6=cosf*output_current-32768;
        break;
    case 7:
        ACC59E3_0_OUT7=cosf*output_current-32768;
        break;
    case 8:
        ACC59E3_0_OUT8=cosf*output_current-32768;
        break;

    default:
        break;
}
close
```

The user then can call this function from a PLC, such as in the following example code:

```
global Desired_Output_Current;
Desired_Output_Current=12.0; // milliamps

open plc 11
local Channel_Number;
Channel_Number = 1;
while(Channel_Number < 9)
{
    call OUTC(Desired_Output_Current,Channel_Number);
    Channel_Number++;
}
close
```

Alternative Output Method: DAC Structures

Alternatively, you can use ACC-59E3 structures to control the DACs.



Note

These structures can be used for current output and for voltage output options.

The advantage of this is that you need not assign M-Variables; the disadvantage is that you must shift up your software counts values, S_{out} , 16 bits before writing to the DAC structures.

The DAC channels correspond to the following structures, where n corresponds to the card index as described in the “Addressing ACC-59E3” section of this manual:

Analog Output Channel	Structure	Bit Range (for 16-bit DACs)
1	ACC59E3[i].Chan[0].DAC[0]	[31:16]
2	ACC59E3[i].Chan[0].DAC[1]	[31:16]
3	ACC59E3[i].Chan[1].DAC[0]	[31:16]
4	ACC59E3[i].Chan[1].DAC[1]	[31:16]
5	ACC59E3[i].Chan[2].DAC[0]	[31:16]
6	ACC59E3[i].Chan[2].DAC[1]	[31:16]
7	ACC59E3[i].Chan[3].DAC[0]	[31:16]
8	ACC59E3[i].Chan[3].DAC[1]	[31:16]

Now you can use the same equations for calculating what to write to these structures in order to achieve your desired DAC output as used for M-Variables shown above, but make sure you first multiply your software counts value (S_{out}) that you got from those equations by 2^{16} before writing to these structures.

Example: Using Subprograms to Output Voltage

This example uses a subprogram to output voltage from the DACs of an ACC-59E3 addressed as Card 0. Note that the scale factor **vosf** differs in this example from the M-Variables-based example from earlier.

```
global vosf=32767.0/10.064*exp2(16); // Voltage Output Scale Factor
// This subprogram converts the desired output voltage in "output_voltage" [Volts] to software counts
// and then commands the output on the specified channel in the "channel" input
open subprog OUTV(output_voltage,channel)
switch (channel){
  case 1:
    ACC59E3[0].Chan[0].DAC[0]=vosf*output_voltage;
    break;
  case 2:
    ACC59E3[0].Chan[0].DAC[1]=vosf*output_voltage;
    break;
  case 3:
    ACC59E3[0].Chan[1].DAC[0]=vosf*output_voltage;
    break;
  case 4:
    ACC59E3[0].Chan[1].DAC[1]=vosf*output_voltage;
    break;
  case 5:
    ACC59E3[0].Chan[2].DAC[0]=vosf*output_voltage;
    break;
  case 6:
    ACC59E3[0].Chan[2].DAC[1]=vosf*output_voltage;
    break;
  case 7:
    ACC59E3[0].Chan[3].DAC[0]=vosf*output_voltage;
    break;
  case 8:
    ACC59E3[0].Chan[0].DAC[1]=vosf*output_voltage;
    break;

  default:
    break;
}
close
```

Example: Using Subprograms to Output Current

This example uses a subprogram to output current from the current outputs of an ACC-59E3 addressed as Card 0. Note that the scale factor **cosf** is different in the below example than in the M-Variables-based example from earlier.

```
global cosf=16.21/100.0*65536.0/4.096*exp2(16); // Current Output Scale Factor

open subprog OUTC(output_current,channel)
switch (channel){
  case 1:
    ACC59E3[0].Chan[0].DAC[0]=cosf*output_current-32768;
    break;
  case 2:
    ACC59E3[0].Chan[0].DAC[1]=cosf*output_current-32768;
    break;
  case 3:
    ACC59E3[0].Chan[1].DAC[0]=cosf*output_current-32768;
    break;
  case 4:
    ACC59E3[0].Chan[1].DAC[1]=cosf*output_current-32768;
    break;
  case 5:
    ACC59E3[0].Chan[2].DAC[0]=cosf*output_current-32768;
    break;
  case 6:
    ACC59E3[0].Chan[2].DAC[1]=cosf*output_current-32768;
    break;
  case 7:
    ACC59E3[0].Chan[3].DAC[0]=cosf*output_current-32768;
    break;
  case 8:
    ACC59E3[0].Chan[0].DAC[1]=cosf*output_current-32768;
    break;

  default:
    break;
}
close
```

Alternative Amplifier Enable Method: Amp Enable Structures

Instead of using M-Variables to turn on the DAC amplifiers, you can use the following structures, where *n* is the card index described in the “Addressing ACC-59E3” section of this manual:

Current Output Channels	Structure
1–2	ACC59E3[i].Chan[0].AmpEna
3–4	ACC59E3[i].Chan[1].AmpEna
5–6	ACC59E3[i].Chan[2].AmpEna
7–8	ACC59E3[i].Chan[3].AmpEna

Now just write a 1 to these structures when you want to activate current output for the corresponding channels listed in the above table in the leftmost column. Write a 0 to turn off current output.

**Note**

Using these structures is only necessary for current output, not for voltage output.

Configuring the Analog Input Filtering

Hardware Input Filters

The ACC-59E3 has the ability to add hardware filters in the input circuit. The user can select the desired filter for each analog input channel independently. The possible filter cutoff frequencies that the user can select are 24.5, 12.2, 4.3, and 3.2 kHz. These are switched in and out by using the general purpose I/O registers included in the ACC-59E3 ASIC:

ACC59E3[i].GpioData[n],

where *i* is the index of the ACC-59E3 card as given in the Addressing ACC-59E3 section of this manual, and *n* = 0. After the user configures this structure, he or she should put it into the setup file in his or her IDE Project (e.g. “global definitions.pmh” or “acc59e3_0_initialization.pmh”) to save permanently on the PC and to be downloaded and saved into PMAC.

The bits that the user must set are listed in the tables below, where the I/O Bit Settings are designated as GPIOxx, where xx corresponds to the bit number. See the following example for clarification.

Examples to Help Understand the Tables

GPIO03=OFF means that the user must set bit 3 of **ACC59E3[i].GpioData[n]** to a value of 0.

GPIO03=ON means that the user must set bit 3 of **ACC59E3[i].GpioData[n]** to a value of 1.

GPIO01=OFF means that the user must set bit 1 of **ACC59E3[i].GpioData[n]** to a value of 0.

GPIO01=ON means that the user must set bit 1 of **ACC59E3[i].GpioData[n]** to a value of 1.

Input Filter Configuration Bits Table

ADC Input Channel	Input Filter Cutoff Frequency (kHz)	I/O Bit Settings
1	24.5	GPIO03=OFF GPIO19=OFF
	12.2	GPIO03=ON GPIO19=OFF
	4.3	GPIO03=OFF GPIO19=ON
	3.2	GPIO03=ON GPIO19=ON
2	24.5	GPIO01=OFF GPIO17=OFF
	12.2	GPIO01=ON GPIO17=OFF
	4.3	GPIO01=OFF GPIO17=ON
	3.2	GPIO01=ON GPIO17=ON
3	24.5	GPIO02=OFF GPIO18=OFF
	12.2	GPIO02=ON GPIO18=OFF
	4.3	GPIO02=OFF GPIO18=ON
	3.2	GPIO02=ON GPIO18=ON
4	24.5	GPIO00=OFF GPIO16=OFF
	12.2	GPIO00=ON GPIO16=OFF
	4.3	GPIO00=OFF GPIO16=ON
	3.2	GPIO00=ON GPIO16=ON
5	24.5	GPIO07=OFF GPIO23=OFF
	12.2	GPIO07=ON GPIO23=OFF
	4.3	GPIO07=OFF GPIO23=ON
	3.2	GPIO07=ON GPIO23=ON
6	24.5	GPIO05=OFF GPIO21=OFF
	12.2	GPIO05=ON GPIO21=OFF
	4.3	GPIO05=OFF GPIO21=ON
	3.2	GPIO05=ON GPIO21=ON
7	24.5	GPIO06=OFF GPIO22=OFF
	12.2	GPIO06=ON GPIO22=OFF
	4.3	GPIO06=OFF GPIO22=ON
	3.2	GPIO06=ON GPIO22=ON
8	24.5	GPIO04=OFF GPIO20=OFF
	12.2	GPIO04=ON GPIO20=OFF
	4.3	GPIO04=OFF GPIO20=ON
	3.2	GPIO04=ON GPIO20=ON
9	24.5	GPIO11=OFF GPIO27=OFF
	12.2	GPIO11=ON GPIO27=OFF
	4.3	GPIO11=OFF GPIO27=ON
	3.2	GPIO11=ON GPIO27=ON

ADC Input Channel	Input Filter Cutoff Frequency (kHz)	I/O Bit Settings
10	24.5	GPIO09=OFF GPIO25=OFF
	12.2	GPIO09=ON GPIO25=OFF
	4.3	GPIO09=OFF GPIO25=ON
	3.2	GPIO09=ON GPIO25=ON

Input Filter Configuration Bits Table (Continued)

ADC Input Channel	Input Filter Cutoff Frequency (kHz)	I/O Bit Settings
11	24.5	GPIO10=OFF GPIO26=OFF
	12.2	GPIO10=ON GPIO26=OFF
	4.3	GPIO10=OFF GPIO26=ON
	3.2	GPIO10=ON GPIO26=ON
12	24.5	GPIO08=OFF GPIO24=OFF
	12.2	GPIO08=ON GPIO24=OFF
	4.3	GPIO08=OFF GPIO24=ON
	3.2	GPIO08=ON GPIO24=ON
13	24.5	GPIO15=OFF GPIO31=OFF
	12.2	GPIO15=ON GPIO31=OFF
	4.3	GPIO15=OFF GPIO31=ON
	3.2	GPIO15=ON GPIO31=ON
14	24.5	GPIO13=OFF GPIO29=OFF
	12.2	GPIO13=ON GPIO29=OFF
	4.3	GPIO13=OFF GPIO29=ON
	3.2	GPIO13=ON GPIO29=ON
15	24.5	GPIO14=OFF GPIO30=OFF
	12.2	GPIO14=ON GPIO30=OFF
	4.3	GPIO14=OFF GPIO30=ON
	3.2	GPIO14=ON GPIO30=ON
16	24.5	GPIO12=OFF GPIO28=OFF
	12.2	GPIO12=ON GPIO28=OFF
	4.3	GPIO12=OFF GPIO28=ON
	3.2	GPIO12=ON GPIO28=ON

Example of Setting the GpioData Word:

If the ACC-59E3 is addressed as Card 0 (Base Offset = \$900000) and the user only wants to set input 3 with a 3.2 kHz filter, the user must set bit-2 and bit-18 ON in **ACC59E3[0].GpioData[0]**. To do this, enter the following command either in the Terminal Window of the IDE or save and download it in the “global definitions.pmh” file:

```
ACC59E3[0].GpioData[0] = $00040004 // Select 3.2 kHz cutoff freq. for input 3
```



Note

To actually configure these filters, Delta Tau recommends using the filter setup function, **ACC59E3_FilterInit()**, described in “Appendix E: Using ACC-59E3 in the C Environment” (page 92). The example at the end of that Appendix (on page 98) shows how to use that function in a Background C Application. To use this function is much easier than manually calculating all GpioData settings.

Optional Software Low-Pass Filter

If the user desires an even lower cutoff frequency for the analog input filter — lower even than the minimum 3.2 kHz hardware filter — then the user can implement a software low-pass filter using the Encoder Conversion Table (ECT) to lower the cutoff frequency even more. To do this, follow the following procedure:

1. Create one ECT entry to process the ADC input to properly shift and scale the input data
2. Create another ECT entry two slots after the first entry to perform the tracking filter on the processed data from the first ECT entry.

Example: Low-Pass Filter in ECT for IN1

Assuming the ACC-59E3 is configured as Card 0 (Power PMAC I/O Base Address Offset=\$900000). This example will create a Low-Pass Tracking Filter on Input 1 (IN1) with a cutoff frequency of 30 Hz.

Shifting and Scaling ECT Entry

For the first filter, the analog input (assuming the 16-bit ADC option is being used) is brought into the Encoder Conversion Table (ECT) in Entry 1 as a “Single 32-bit Register Read” (in the Power PMAC IDE Software: Delta Tau → Configure → Encoder Conversion Table):

Number	Type	pEnc	pEnc1	MaxDelta
1	1	Acc59E3[0].Chan[0].AdcAmp[0].a	Acc24E2A[4].Chan[0].TimeBetweenCts.a	0

The equivalent code in Power PMAC Structures:

```
EncTable[1].type = 1;           // Set entry type to 32-bit word read
EncTable[1].pEnc = Sys.piom + $900020; // Set encoder address to ADC channel 1
EncTable[1].pEnc1 = Sys.pushm; // Unused; set to Sys.pushm
EncTable[1].index1 = 16;       // Shift left 16 to put source MSB in bit 31
EncTable[1].index2 = 16;       // Shift right 16 to put source LSB into bit 0
EncTable[1].index3 = 0;        // No limit on first derivative
EncTable[1].index4 = 0;        // No limit on second derivative
EncTable[1].ScaleFactor = 1/exp2(EncTable[1].index1); // Scale factor (1/(2^16))
```

**Note**

To clarify the example above: **Acc59E3[0].Chan[0].AdcAmp[0].a** is equivalent to **Sys.piom + \$900020**, and **1/pow(2,EncTable[1].index1) = 0.0000152587890625**.

Tracking Filter ECT Entry

The next ECT entry is also a “Single 32-bit Register Read” but in this case, the **EncTable[3].index2** field acts as the proportional gain term of the low-pass filter, and **EncTable[3].index1** serves as the integral gain term. This creates a tracking filter, which is a low-pass filter with zero steady state error at constant velocity. To design the filter and thereby determine the values to write to index2 and index1, follow the following procedure:

1. Select a cutoff frequency f_c in Hz. This is the frequency above which the filter will start attenuating the signal.
2. Compute the filter’s natural frequency $\omega_n = 2\pi f_c$.
3. Select a damping ratio ζ for the filter (usually $= \sqrt{2}/2$).
4. Compute the filter’s sample time T_s in seconds. This is the servo update period, and can be calculated as $T_s = \text{Sys.ServoPeriod} / 1000$.
5. Compute the proportional gain term **index2** $= (512) \cdot \zeta \cdot \omega_n \cdot T_s$.
6. Compute the integral gain term **index1** $= 256 \cdot \omega_n^2 \cdot T_s^2$.

Thus for this example, a cutoff frequency of $f_c = 200$ Hz will be used for the input filter. Then, the Power PMAC code used to set this low-pass filter up as ECT entry number 3 is as follows:

```
#define fc          200.0                // Desired Cutoff Frequency (Hz)
#define wn          2.0*3.14159265*fc    // Natural Frequency (rad/sec)
#define zeta        sqrt(2.0)/2.0        // Damping Ratio
#define Ts          Sys.ServoPeriod/1000.0 // Sampling Period (sec)
EncTable[3].type = 1;                    // Set entry type to 32-bit word read
EncTable[3].pEnc = EncTable[1].PrevEnc.a; // Set encoder address to output of ECT entry #1
EncTable[3].pEnc1 = Sys.pushm;           // Unused; set to Sys.pushm
EncTable[3].index1 = 256.0*pow(wn,2.0)*pow(Ts,2.0); // Calculate index1
EncTable[3].index2 = 512.0*zeta*wn*Ts;    // Calculate index2
EncTable[3].index3 = 0;                   // No limit on first derivative
EncTable[3].index4 = 0;                   // No limit on second derivative
EncTable[3].ScaleFactor = 1/exp2(16);     // Scale factor of 1/(2^16)
```

**Note**

The user should always point back two ECT slots because of the pipeline that reads the next ECT source before it writes the output. Otherwise the ECT entry would be reading the previous servo cycle’s data. Thus, ECT Entry 3 is used here for the tracking filter entry rather than ECT Entry 2.

**Note**

If the value for index1 is too small, rounding to the nearest integer may yield inaccurate results. If index1 is less than 64, double index1 enough times to make index1 greater than 64 and set the element index4 to the number of times index1 needed to be doubled.

Now, the user can access the filtered ADC channel (in this example, ADC1) by writing a PLC to scale the output of the ECT entry and store the result in a global variable (P-Variable).

Example: Scaling and Storing the Filtered ADC Value from the ECT Output

Assuming the user set up the ECT filter entry to filter ADC 1 as shown in the example above, the following PLC will properly scale the ECT entry's output and store the resulting, filtered ADC1 value in a global variable named "ACC59E3_0_IN1_Filtered".

```
global ACC59E3_0_IN1_Filtered;  
open plc 8  
ACC59E3_0_IN1_Filtered=EncTable[3].PrevEnc/65536;  
close
```

Alternatively, the user can set a motor to the output of the ECT entry as follows.

Example: Pointing a Motor to the Filtered ADC's ECT Entry's Output

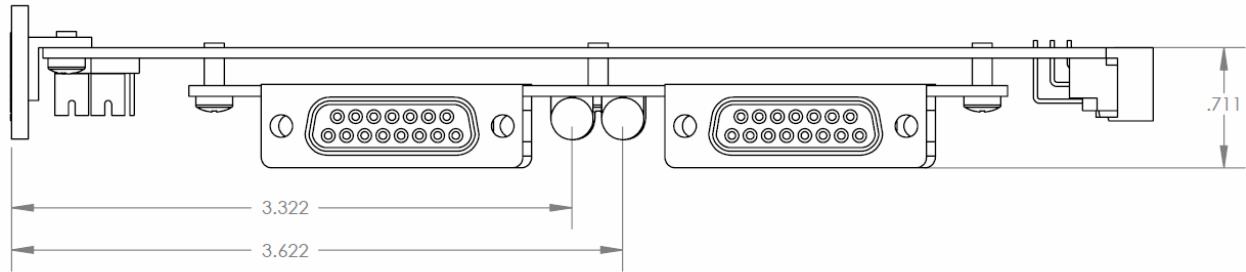
Assuming the user set up the ECT filter entry to filter ADC 1 as shown in the example above, the following example points Motor 2 to the filtered ADC1 result.

```
Motor[2].pEnc=EncTable[3].a;  
Motor[2].pEnc2=EncTable[3].a;
```

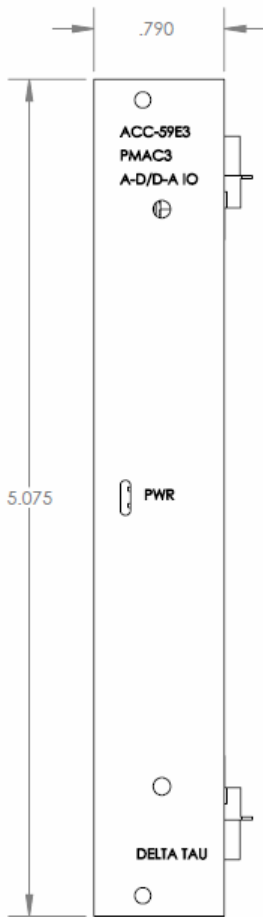
ACC-59E3 LAYOUT & PINOUTS

DB15 Option

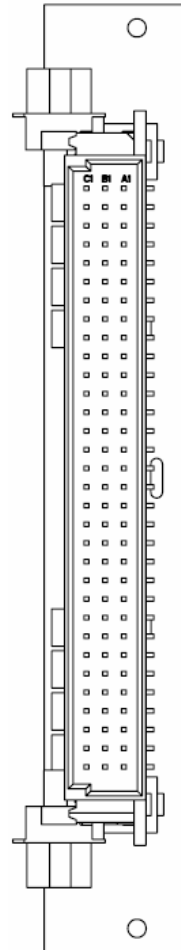
Top View



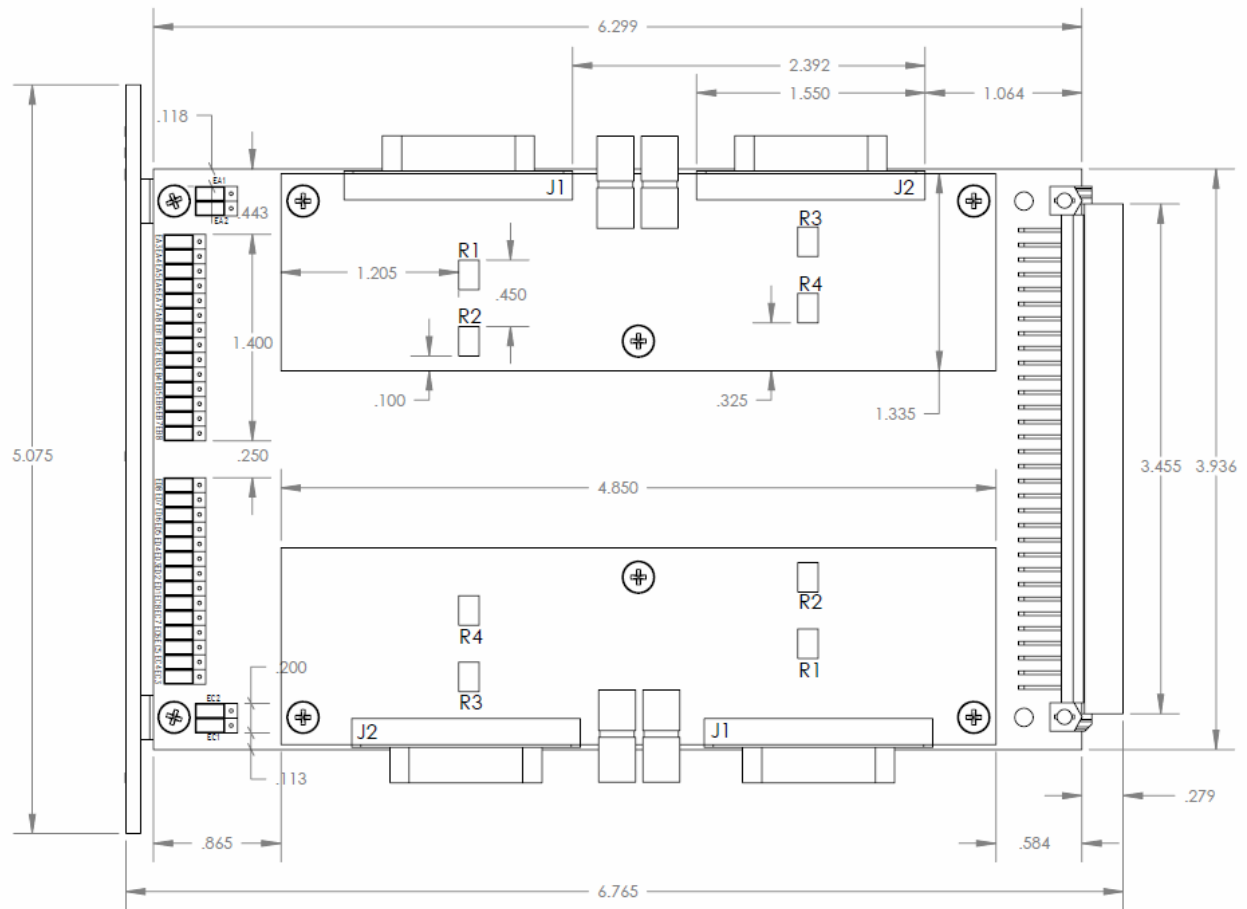
Front View



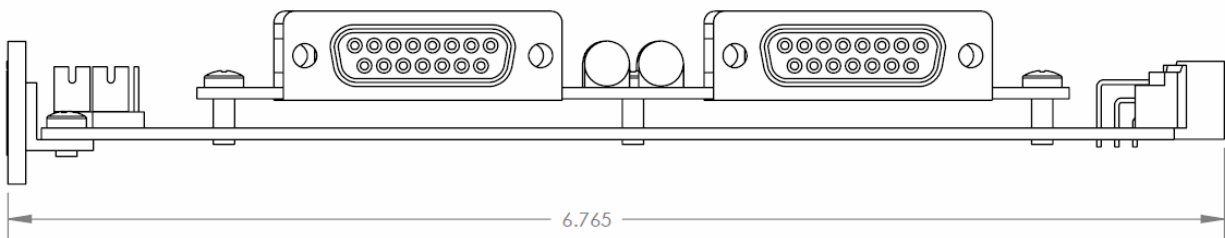
Rear View



Side View



Bottom View



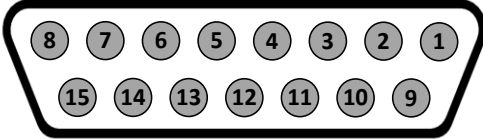
All dimensions are in inches. Drawings are not to scale.

P1: Backplane Bus

This connector is used to interface to the UMAC's processor via the 3U backplane bus. The signals brought in through this connector are buffered on board.

DB15 Breakout Option

J1 Top

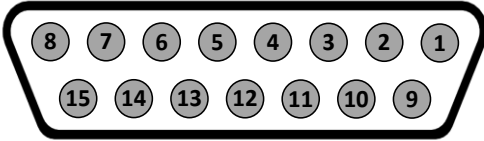
J1 Top: D-sub DA-15F Mating: D-sub DA-15M			
Pin #	Symbol	Function	Description
1	ADC1+	Input	±10 V input when EA7&8 set 2–3 ±20 mA input when EA7&8 set 1–2
2	ADC2+	Input	±10 V input when EA3&4 set 2–3 ±20 mA input when EA3&4 set 1–2
3	ADC3+	Input	±10 V input when EA5&6 set 2–3 ±20 mA input when EA5&6 set 1–2
4	ADC4+	Input	±10 V input when EA1&2 set 2–3 ±20 mA input when EA1&2 set 1–2
5	DAC1+	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
6	DAC2+	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
7	+15V	Output	0.5 Amp Slow-Blow Fuse Limited
8	AGND	Common	Ground
9	ADC1-	Input	±10 V input when EA7&8 set 2–3 ±20 mA input when EA7&8 set 1–2
10	ADC2-	Input	±10 V input when EA3&4 set 2–3 ±20 mA input when EA3&4 set 1–2
11	ADC3-	Input	±10 V input when EA5&6 set 2–3 ±20 mA input when EA5&6 set 1–2
12	ADC4-	Input	±10 V input when EA1&2 set 2–3 ±20 mA input when EA1&2 set 1–2
13	DAC1-	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
14	DAC2-	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
15	-15V	Output	0.5 Amp Slow-Blow Fuse Limited



Note

- The common ground (pin #8) is tied to the digital ground of the UMAC rack.
- Tie the ADC- pin to ground if using single-ended wiring to ensure full resolution and proper operation of the ADC channel.

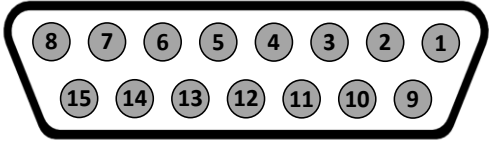
J2 Top

J2 Top: D-sub DA-15F Mating: D-sub DA-15M			
Pin #	Symbol	Function	Description
1	ADC5+	Input	±10 V input when EB7&8 set 2–3 ±20 mA input when EB7&8 set 1–2
2	ADC6+	Input	±10 V input when EB3&4 set 2–3 ±20 mA input when EB3&4 set 1–2
3	ADC7+	Input	±10 V input when EB5&6 set 2–3 ±20 mA input when EB5&6 set 1–2
4	ADC8+	Input	±10 V input when EB1&2 set 2–3 ±20 mA input when EB1&2 set 1–2
5	DAC3+	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
6	DAC4+	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
7	+15V	Output	0.5 Amp Slow-Blow Fuse Limited
8	AGND	Common	Ground
9	ADC5-	Input	±10 V input when EB7&8 set 2–3 ±20 mA input when EB7&8 set 1–2
10	ADC6-	Input	±10 V input when EB3&4 set 2–3 ±20 mA input when EB3&4 set 1–2
11	ADC7-	Input	±10 V input when EB5&6 set 2–3 ±20 mA input when EB5&6 set 1–2
12	ADC8-	Input	±10 V input when EB1&2 set 2–3 ±20 mA input when EB1&2 set 1–2
13	DAC3-	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
14	DAC4-	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
15	-15V	Output	0.5 Amp Slow-Blow Fuse Limited

**Note**

- The common ground (pin #8) is tied to the digital ground of the UMAC rack.
- Tie the ADC- pin to ground if using single-ended wiring to ensure full resolution and proper operation of the ADC channel.

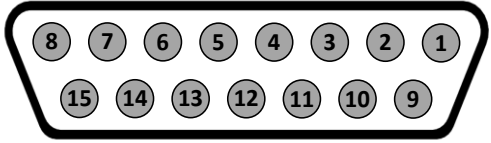
J2 Bottom

J2 Bottom: D-sub DA-15F Mating: D-sub DA-15M			
Pin #	Symbol	Function	Description
1	ADC9+	Input	±10 V input when EC7&8 set 2–3 ±20 mA input when EC7&8 set 1–2
2	ADC10+	Input	±10 V input when EC3&4 set 2–3 ±20 mA input when EC3&4 set 1–2
3	ADC11+	Input	±10 V input when EC5&6 set 2–3 ±20 mA input when EC5&6 set 1–2
4	ADC12+	Input	±10 V input when EC1&2 set 2–3 ±20 mA input when EC1&2 set 1–2
5	DAC5+	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
6	DAC6+	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
7	+15V	Output	0.5 Amp Slow-Blow Fuse Limited
8	AGND	Common	Ground
9	ADC9-	Input	±10 V input when EC7&8 set 2–3 ±20 mA input when EC7&8 set 1–2
10	ADC10-	Input	±10 V input when EC3&4 set 2–3 ±20 mA input when EC3&4 set 1–2
11	ADC11-	Input	±10 V input when EC5&6 set 2–3 ±20 mA input when EC5&6 set 1–2
12	ADC12-	Input	±10 V input when EC1&2 set 2–3 ±20 mA input when EC1&2 set 1–2
13	DAC5-	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
14	DAC6-	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
15	-15V	Output	0.5 Amp Slow-Blow Fuse Limited

**Note**

- The common ground (pin #8) is tied to the digital ground of the UMAC rack.
- Tie the ADC- pin to ground if using single-ended wiring to ensure full resolution and proper operation of the ADC channel.

J1 Bottom

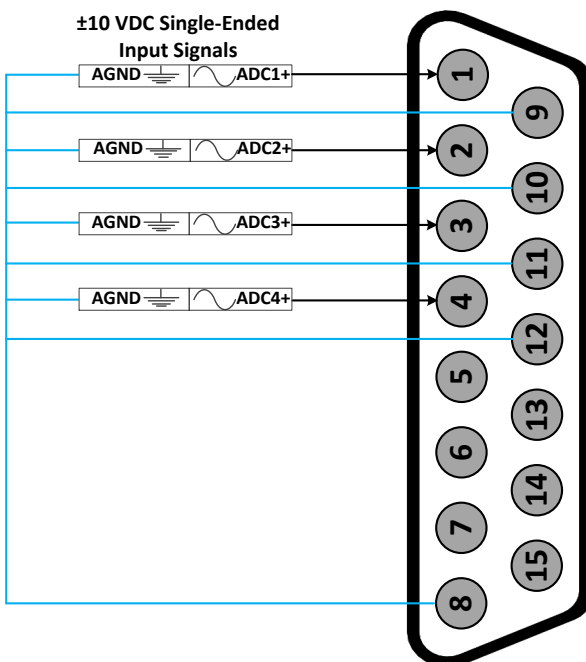
J1 Bottom: D-sub DA-15F Mating: D-sub DA-15M			
Pin #	Symbol	Function	Description
1	ADC13+	Input	±10 V input when ED7&8 set 2–3 ±20 mA input when ED7&8 set 1–2
2	ADC14+	Input	±10 V input when ED3&4 set 2–3 ±20 mA input when ED3&4 set 1–2
3	ADC15+	Input	±10 V input when ED5&6 set 2–3 ±20 mA input when ED5&6 set 1–2
4	ADC16+	Input	±10 V input when ED1&2 set 2–3 ±20 mA input when ED1&2 set 1–2
5	DAC7+	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
6	DAC8+	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
7	+15V	Output	0.5 Amp Slow-Blow Fuse Limited
8	AGND	Common	Ground
9	ADC13-	Input	±10 V input when ED7&8 set 2–3 ±20 mA input when ED7&8 set 1–2
10	ADC14-	Input	±10 V input when ED3&4 set 2–3 ±20 mA input when ED3&4 set 1–2
11	ADC15-	Input	±10 V input when ED5&6 set 2–3 ±20 mA input when ED5&6 set 1–2
12	ADC16-	Input	±10 V input when ED1&2 set 2–3 ±20 mA input when ED1&2 set 1–2
13	DAC7-	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
14	DAC8-	Output	±10 V output when voltage output option is present ±20 mA output when current output option is present
15	-15V	Output	0.5 Amp Slow-Blow Fuse Limited

**Note**

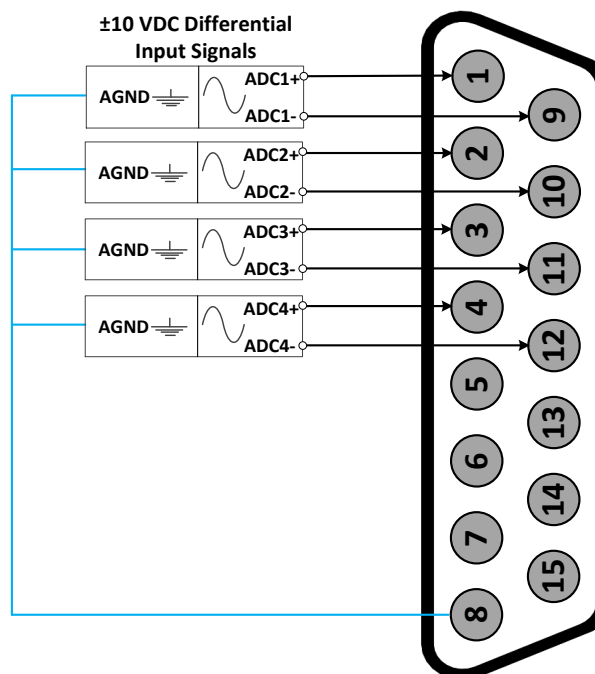
- The common ground (pin #8) is tied to the digital ground of the UMAC rack.
- Tie the ADC- pin to ground if using single-ended wiring to ensure full resolution and proper operation of the ADC channel.

Sample Wiring Diagrams

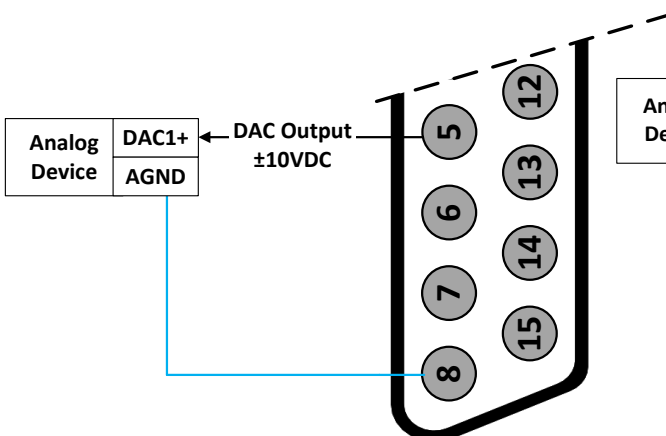
Single-Ended Analog Voltage Inputs



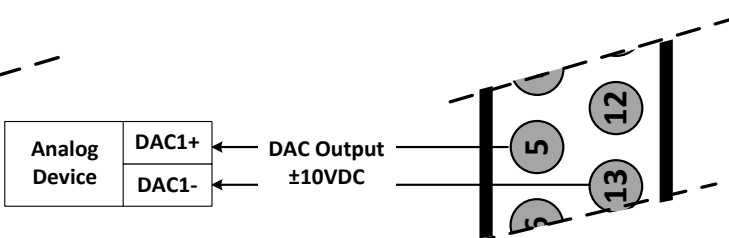
Differential Analog Voltage Inputs



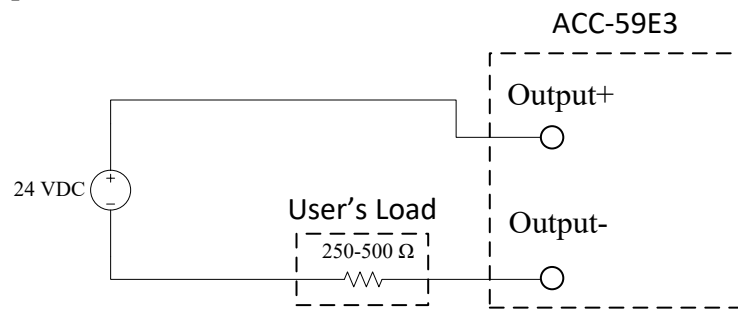
Single-Ended Analog Voltage Outputs



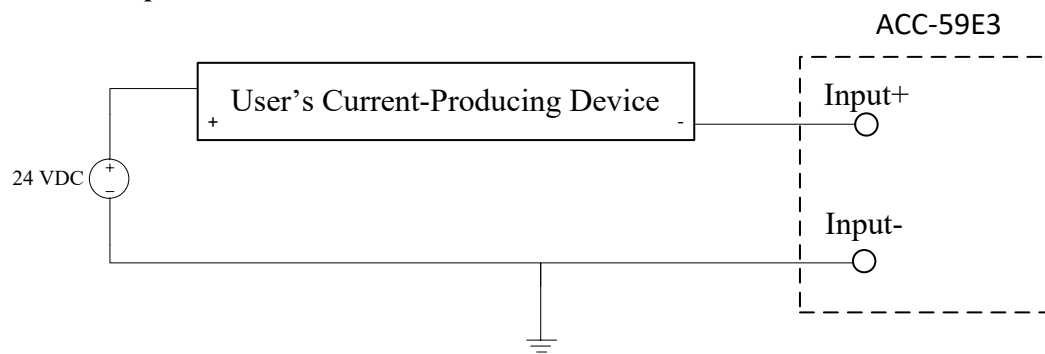
Differential Analog Voltage Outputs



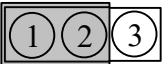
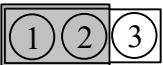
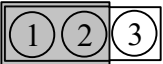
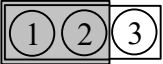
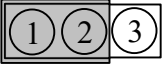
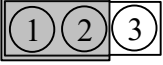
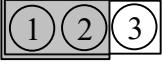
Analog Current Output

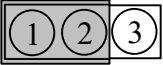
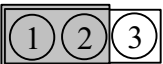
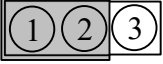
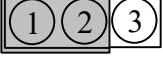
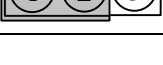


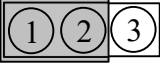
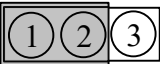
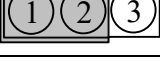
Analog Current Input

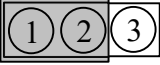
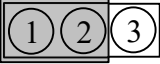
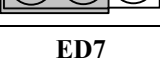
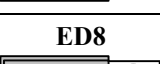


APPENDIX A: JUMPER SETTINGS

Jumper	Configuration	Default
EA1 	Set 1–2 for current sensing mode on ADC4 Set 2–3 for voltage sensing mode on ADC4	1–2 for current output option 2–3 for voltage output option/no output option
EA2 	Set 1–2 for current sensing mode on ADC4 Set 2–3 for voltage sensing mode on ADC4	1–2 for current output option 2–3 for voltage output option/no output option
EA3 	Set 1–2 for current sensing mode on ADC2 Set 2–3 for voltage sensing mode on ADC2	1–2 for current output option 2–3 for voltage output option/no output option
EA4 	Set 1–2 for current sensing mode on ADC2 Set 2–3 for voltage sensing mode on ADC2	1–2 for current output option 2–3 for voltage output option/no output option
EA5 	Set 1–2 for current sensing mode on ADC3 Set 2–3 for voltage sensing mode on ADC3	1–2 for current output option 2–3 for voltage output option/no output option
EA6 	Set 1–2 for current sensing mode on ADC3 Set 2–3 for voltage sensing mode on ADC3	1–2 for current output option 2–3 for voltage output option/no output option
EA7 	Set 1–2 for current sensing mode on ADC1 Set 2–3 for voltage sensing mode on ADC1	1–2 for current output option 2–3 for voltage output option/no output option
EA8 	Set 1–2 for current sensing mode on ADC1 Set 2–3 for voltage sensing mode on ADC1	1–2 for current output option 2–3 for voltage output option/no output option

Jumper	Configuration	Default
EB1 	Set 1–2 for current sensing mode on ADC8 Set 2–3 for voltage sensing mode on ADC8	1–2 for current output option 2–3 for voltage output option/no output option
EB2 	Set 1–2 for current sensing mode on ADC8 Set 2–3 for voltage sensing mode on ADC8	1–2 for current output option 2–3 for voltage output option/no output option
EB3 	Set 1–2 for current sensing mode on ADC6 Set 2–3 for voltage sensing mode on ADC6	1–2 for current output option 2–3 for voltage output option/no output option
EB4 	Set 1–2 for current sensing mode on ADC6 Set 2–3 for voltage sensing mode on ADC6	1–2 for current output option 2–3 for voltage output option/no output option
EB5 	Set 1–2 for current sensing mode on ADC7 Set 2–3 for voltage sensing mode on ADC7	1–2 for current output option 2–3 for voltage output option/no output option
EB6 	Set 1–2 for current sensing mode on ADC7 Set 2–3 for voltage sensing mode on ADC7	1–2 for current output option 2–3 for voltage output option/no output option
EB7 	Set 1–2 for current sensing mode on ADC5 Set 2–3 for voltage sensing mode on ADC5	1–2 for current output option 2–3 for voltage output option/no output option
EB8 	Set 1–2 for current sensing mode on ADC5 Set 2–3 for voltage sensing mode on ADC5	1–2 for current output option 2–3 for voltage output option/no output option

Jumper	Configuration	Default
EC1 	Set 1–2 for current sensing mode on ADC12 Set 2–3 for voltage sensing mode on ADC12	1–2 for current output option 2–3 for voltage output option/no output option
EC2 	Set 1–2 for current sensing mode on ADC12 Set 2–3 for voltage sensing mode on ADC12	1–2 for current output option 2–3 for voltage output option/no output option
EC3 	Set 1–2 for current sensing mode on ADC10 Set 2–3 for voltage sensing mode on ADC10	1–2 for current output option 2–3 for voltage output option/no output option
EC4 	Set 1–2 for current sensing mode on ADC10 Set 2–3 for voltage sensing mode on ADC10	1–2 for current output option 2–3 for voltage output option/no output option
EC5 	Set 1–2 for current sensing mode on ADC11 Set 2–3 for voltage sensing mode on ADC11	1–2 for current output option 2–3 for voltage output option/no output option
EC6 	Set 1–2 for current sensing mode on ADC11 Set 2–3 for voltage sensing mode on ADC11	1–2 for current output option 2–3 for voltage output option/no output option
EC7 	Set 1–2 for current sensing mode on ADC9 Set 2–3 for voltage sensing mode on ADC9	1–2 for current output option 2–3 for voltage output option/no output option
EC8 	Set 1–2 for current sensing mode on ADC9 Set 2–3 for voltage sensing mode on ADC9	1–2 for current output option 2–3 for voltage output option/no output option

Jumper	Configuration	Default
ED1 	Set 1–2 for current sensing mode on ADC16 Set 2–3 for voltage sensing mode on ADC16	1–2 for current output option 2–3 for voltage output option/no output option
ED2 	Set 1–2 for current sensing mode on ADC16 Set 2–3 for voltage sensing mode on ADC16	1–2 for current output option 2–3 for voltage output option/no output option
ED3 	Set 1–2 for current sensing mode on ADC14 Set 2–3 for voltage sensing mode on ADC14	1–2 for current output option 2–3 for voltage output option/no output option
ED4 	Set 1–2 for current sensing mode on ADC14 Set 2–3 for voltage sensing mode on ADC14	1–2 for current output option 2–3 for voltage output option/no output option
ED5 	Set 1–2 for current sensing mode on ADC15 Set 2–3 for voltage sensing mode on ADC15	1–2 for current output option 2–3 for voltage output option/no output option
ED6 	Set 1–2 for current sensing mode on ADC15 Set 2–3 for voltage sensing mode on ADC15	1–2 for current output option 2–3 for voltage output option/no output option
ED7 	Set 1–2 for current sensing mode on ADC13 Set 2–3 for voltage sensing mode on ADC13	1–2 for current output option 2–3 for voltage output option/no output option
ED8 	Set 1–2 for current sensing mode on ADC13 Set 2–3 for voltage sensing mode on ADC13	1–2 for current output option 2–3 for voltage output option/no output option

APPENDIX B: DAC OUTPUT OFFSET POTS

For each individual output mezzanine board (ASSY 604028-102 or newer for analog voltage output, ASSY 604029-100 or newer for analog current output), the following 5 kOhm potentiometers (pots) adjust the output offsets of the DAC output channels:

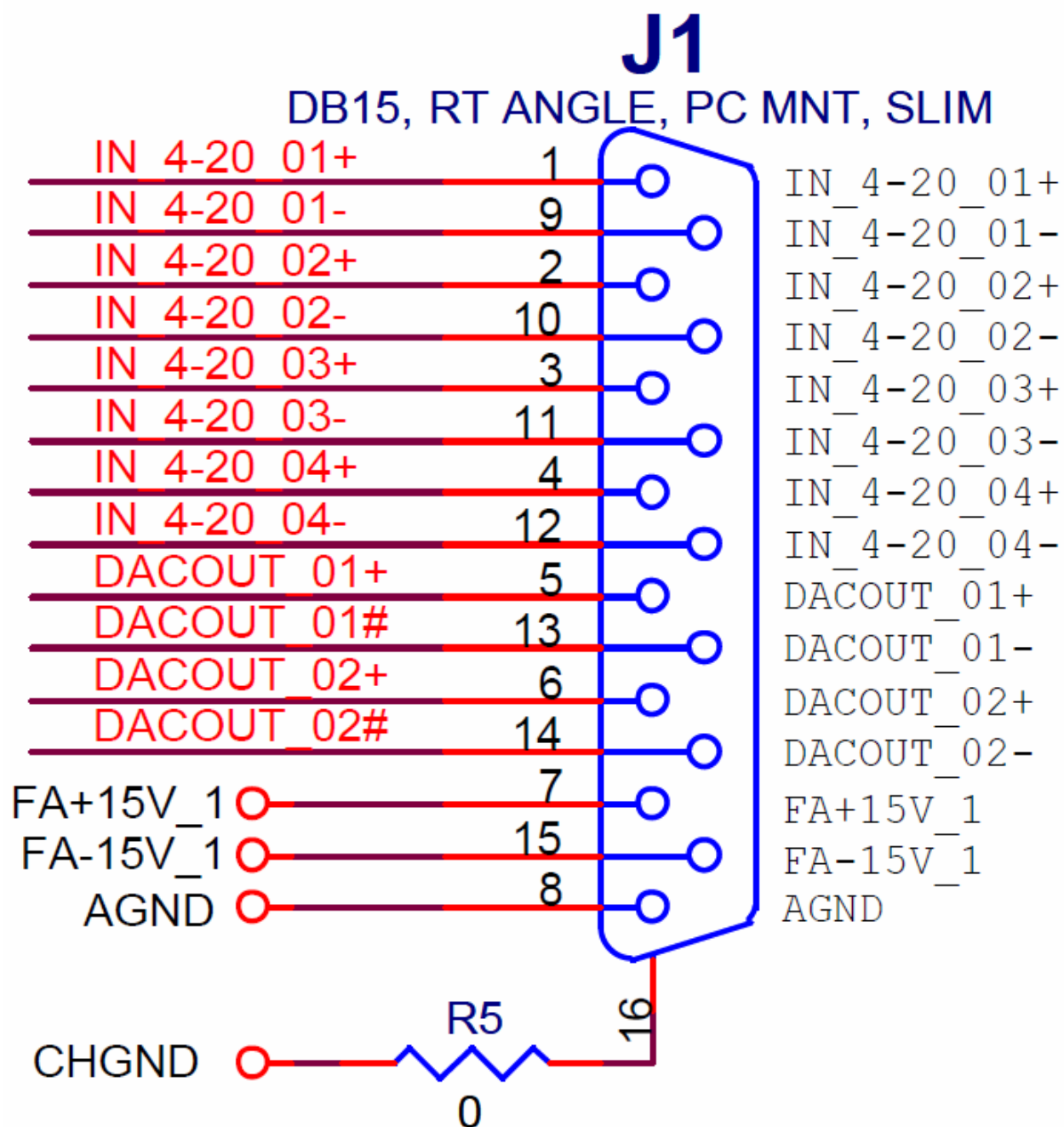
DAC Channel (Per Board)	Potentiometer Number
1	R1
2	R2
3	R3
4	R4

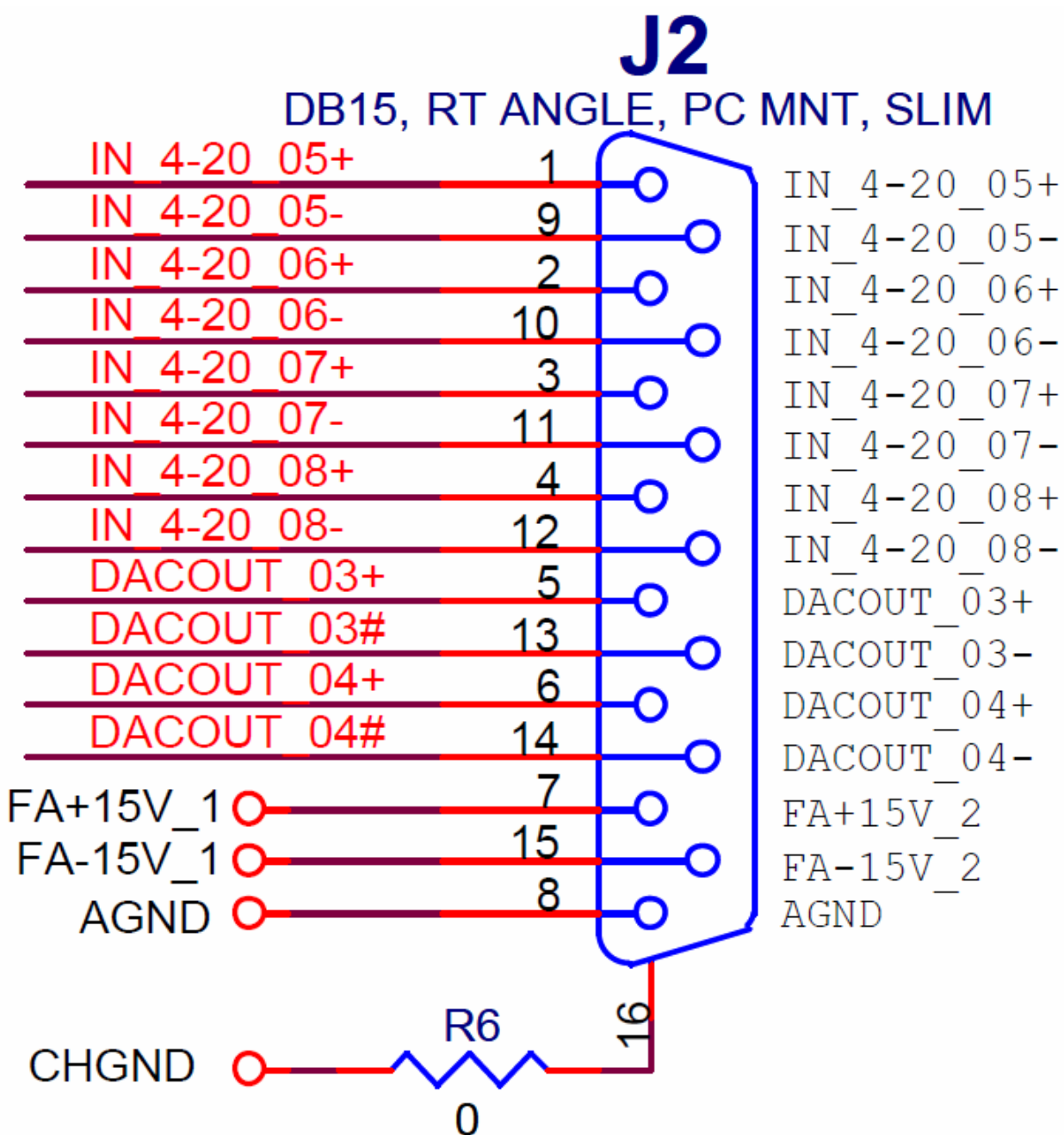
Note that these potentiometer number labels are per board; i.e., each mezzanine board only supports four DAC channels, each with R1–R4 for channels 1–4, respectively. With two mezzanine boards installed, the board obtains eight DAC channels, but the potentiometer number labels are per board and are always labeled R1–R4.

APPENDIX C: SCHEMATICS

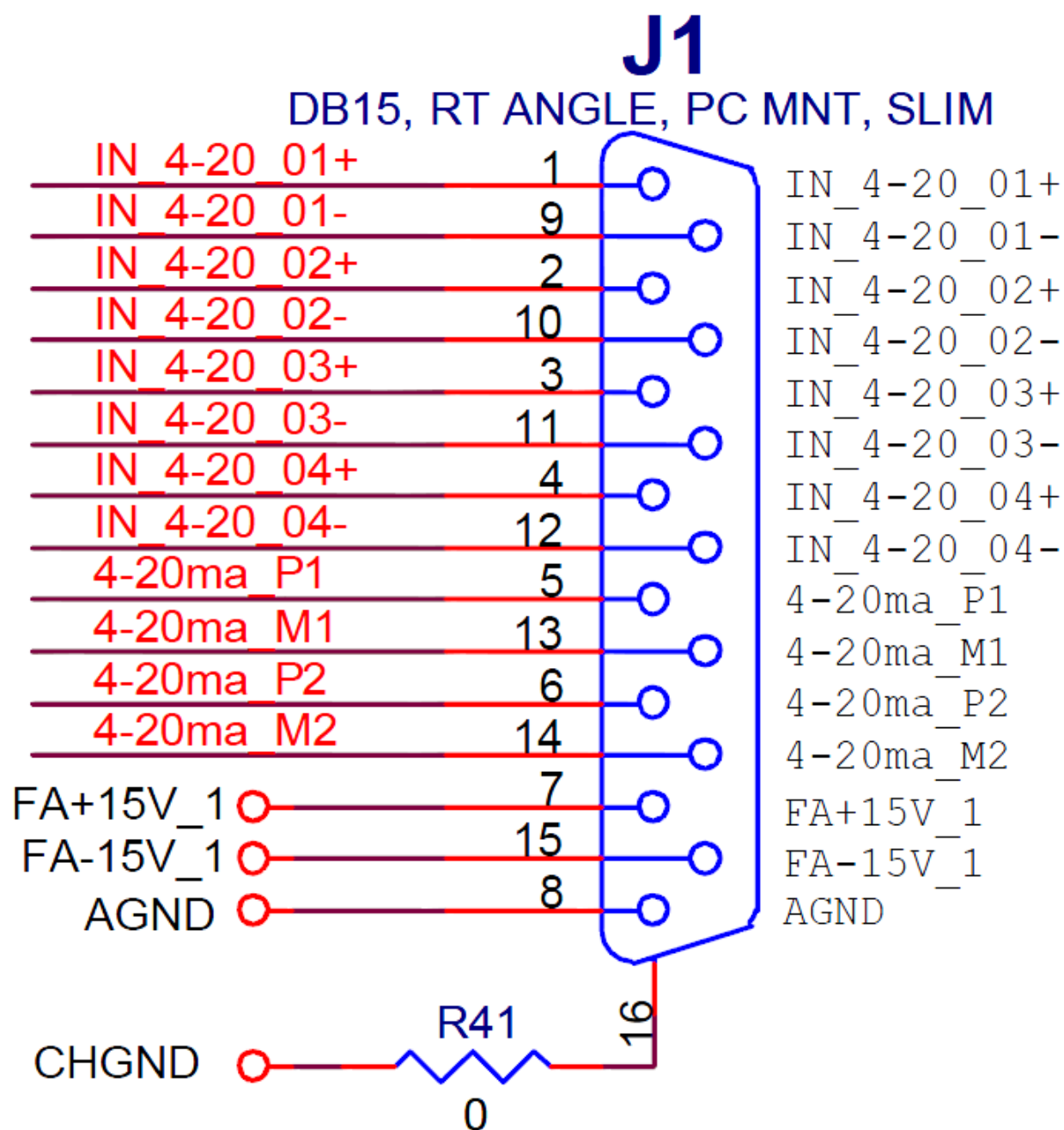
Connector Schematics

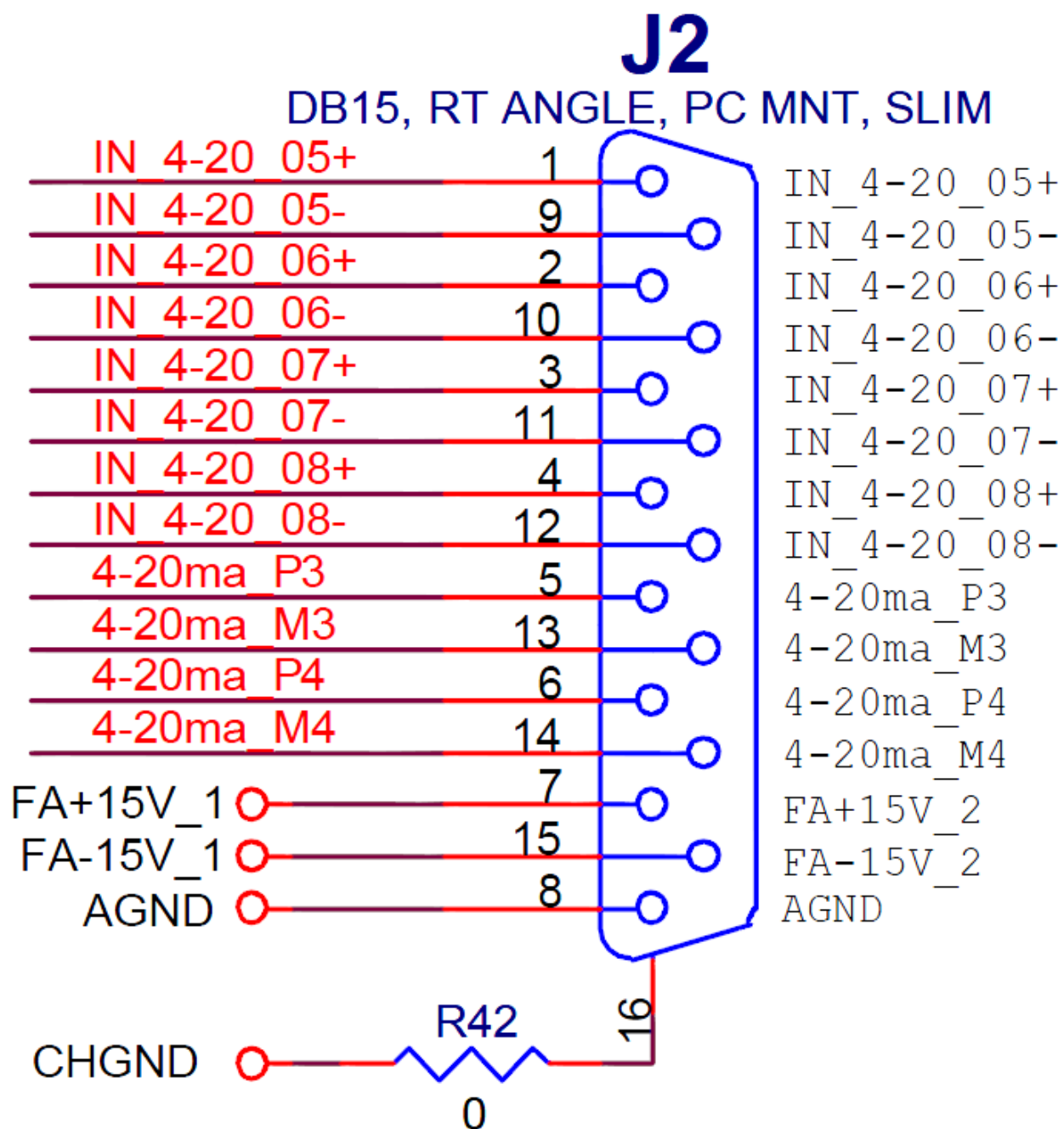
ACC-59E3 4-Channel Analog Output Voltage Mezzanine Board (ASSY 604028-102)





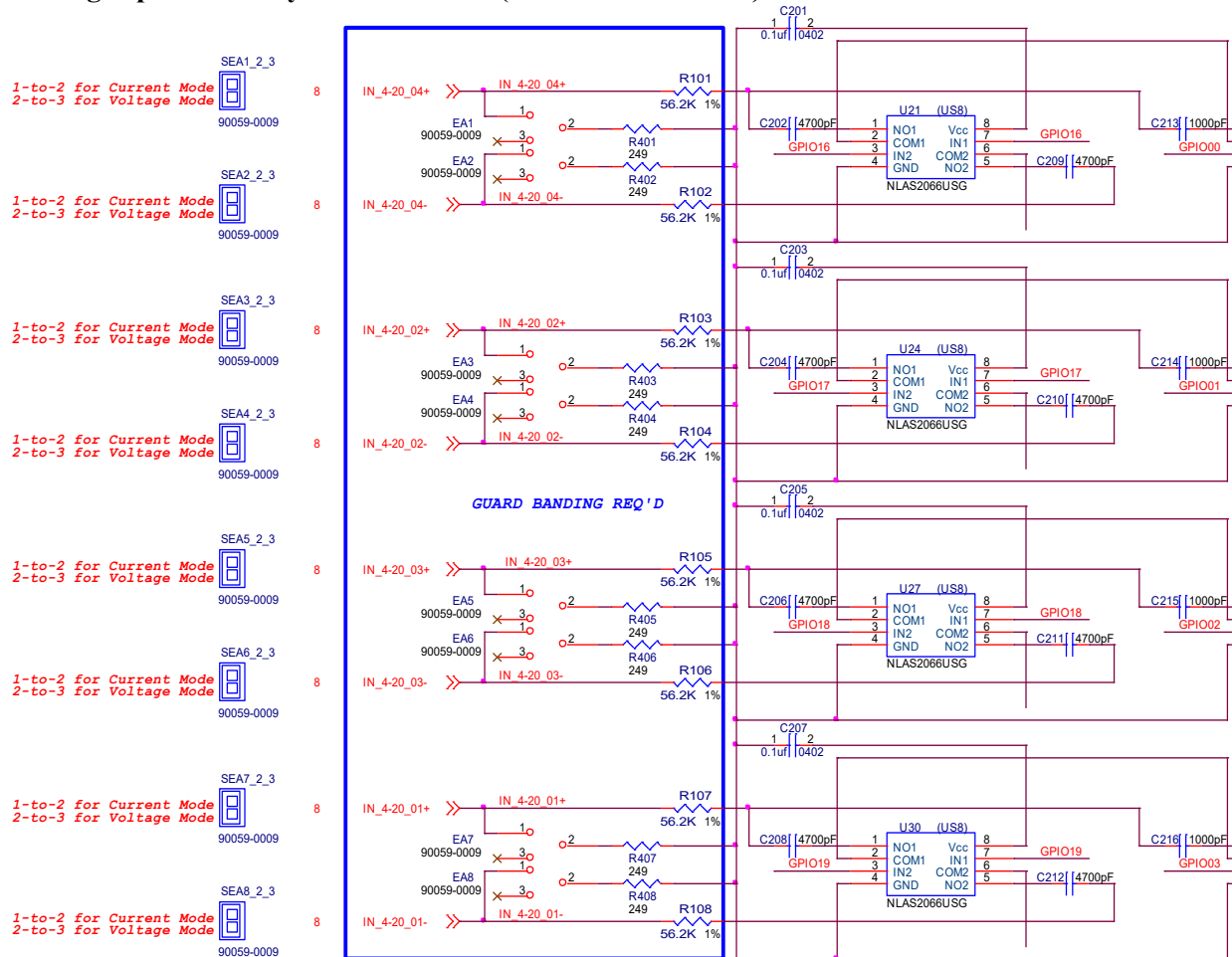
ACC-59E3 4-CHANNEL 4-20mA OUTPUT MEZZANINE BOARD (ASSY 604029-100)





Analog Input Circuitry Schematics

Analog Input Circuitry on Base Board (ASSY 604027-100A)

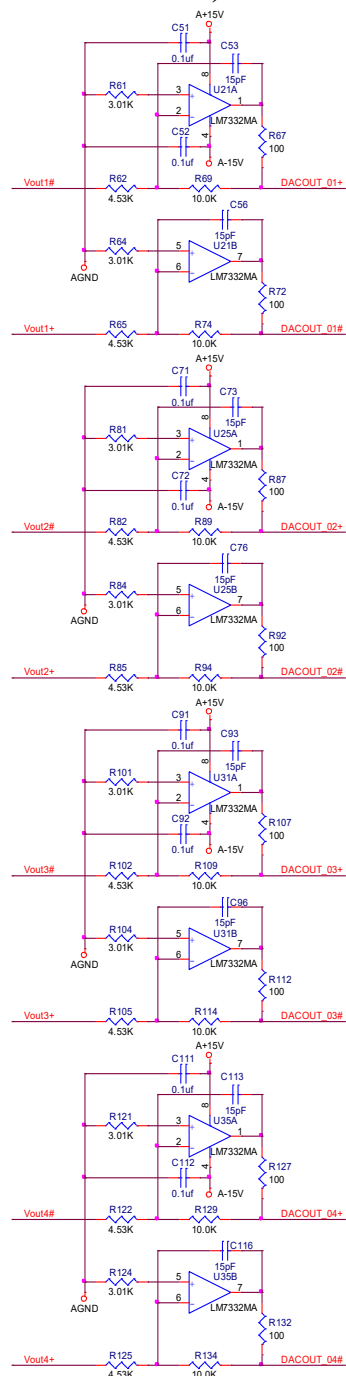
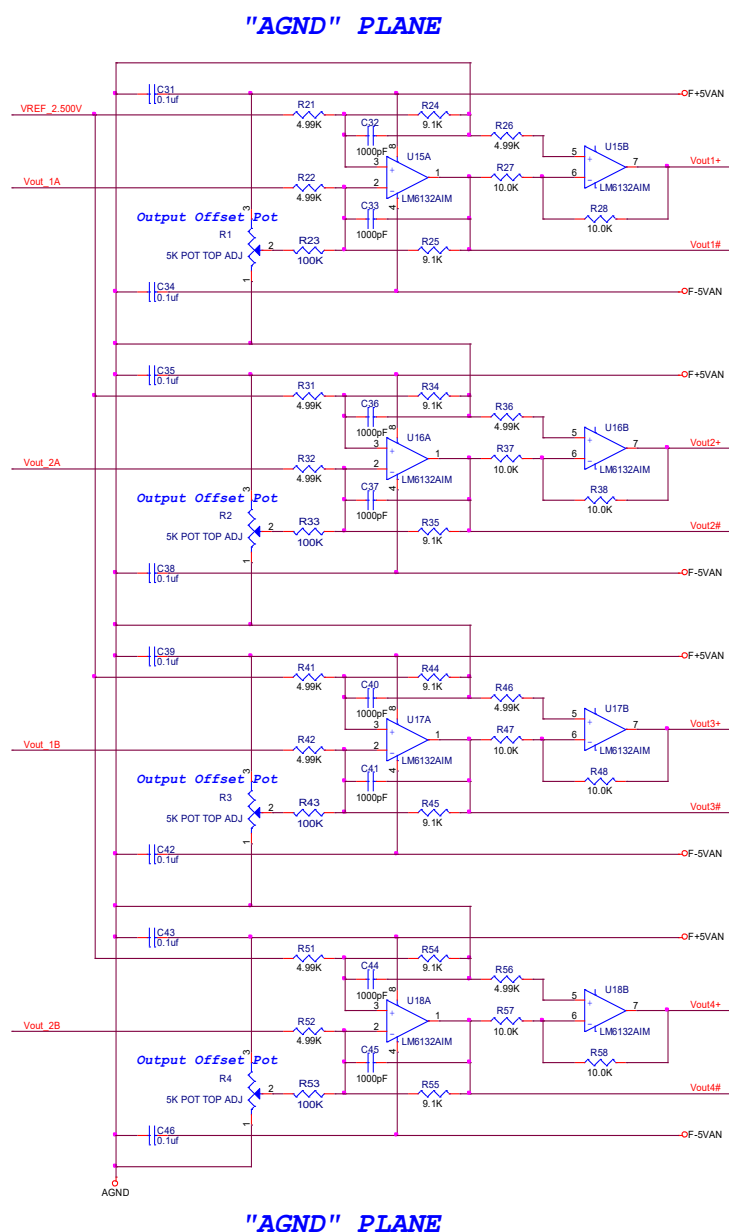


Note

The schematic shown above is for ADC channels 1–4, but channels 5–16 are identical in design.

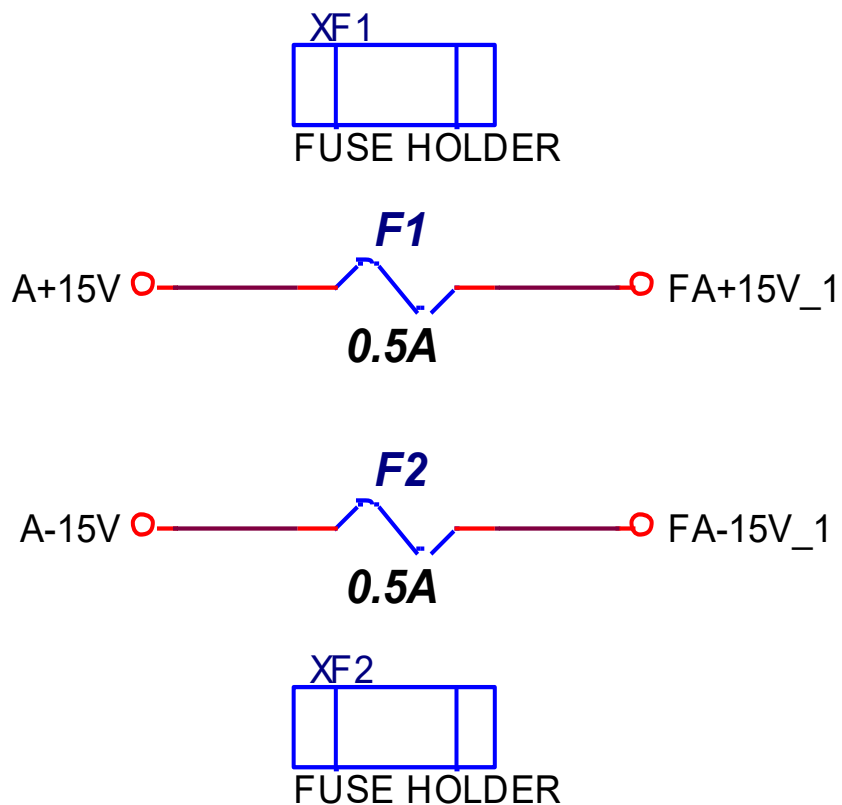
Analog Voltage Output Circuitry Schematics

ACC-59E3 4-Channel Analog Output Voltage Mezzanine Board (ASSY 604028-102)



Note

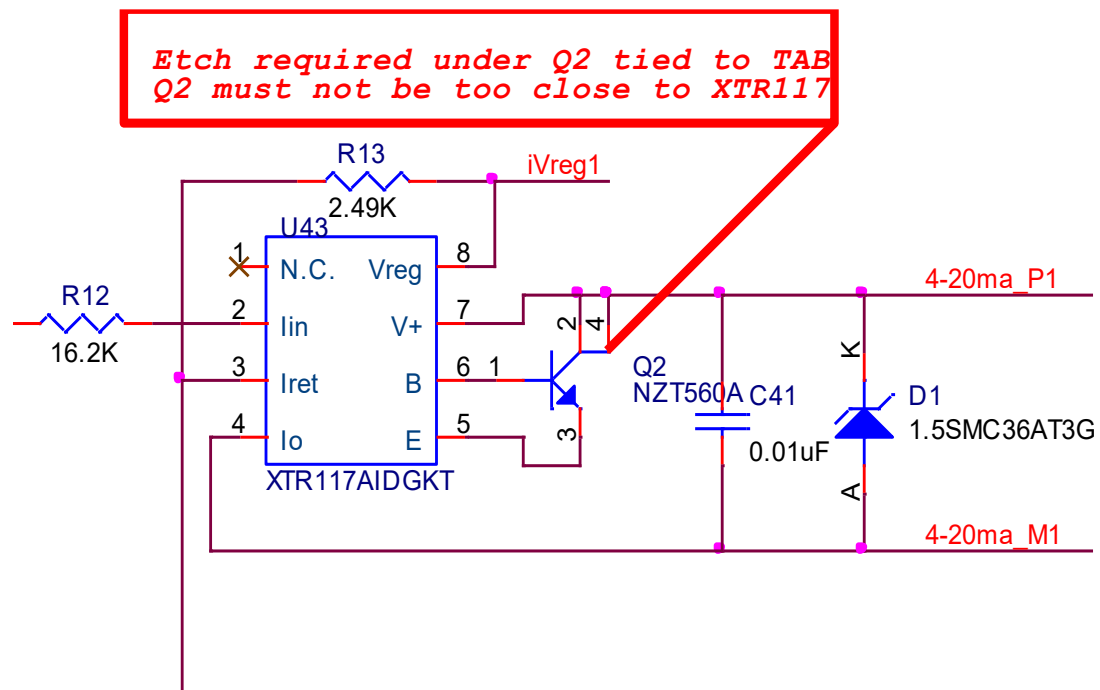
The schematic shown above is for DAC voltage output channels 1–2, but channels 3–8 are identical in design.

DAC Fuses**Note**

J1 Top and J2 Top (ADC 1-8 and DAC 1-4) share a single +15V Fuse and a single -15V Fuse. J1 Bottom and J2 Bottom (ADC 9-16 and DAC 5-8) also share a single +15V Fuse and a single -15V Fuse.

Analog Current Output Circuitry Schematics

ACC-59E3 4-CHANNEL 4-20mA OUTPUT MEZZANINE BOARD (ASSY 604029-100)



Note

The schematic shown above is for DAC current output channel 1, but channels 2–8 are identical in design.

APPENDIX D: COMPLETE SUGGESTED M-VARIABLES

16-bit ADC Option

ACC-59E3 Addressed as Card 0 with 16-bit ADCs

```
// Suggested M-Variables for ACC-59E3 Addressed as Card 0
// Power PMAC I/O Base Address Offset=$900000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_0_OUT1->s.io:$900040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_0_OUT2->s.io:$900044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_0_OUT3->s.io:$9000C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_0_OUT4->s.io:$9000C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_0_OUT5->s.io:$900140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_0_OUT6->s.io:$900144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_0_OUT7->s.io:$9001C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_0_OUT8->s.io:$9001C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_0_IN1->s.io:$900020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_0_IN2->s.io:$900024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_0_IN3->s.io:$900028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_0_IN4->s.io:$90002C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_0_IN5->s.io:$9000A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_0_IN6->s.io:$9000A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_0_IN7->s.io:$9000A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_0_IN8->s.io:$9000AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_0_IN9->s.io:$900120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_0_IN10->s.io:$900124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_0_IN11->s.io:$900128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_0_IN12->s.io:$90012C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_0_IN13->s.io:$9001A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_0_IN14->s.io:$9001A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_0_IN15->s.io:$9001A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_0_IN16->s.io:$9001AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_0_AENA1->u.io:$90006C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_0_AENA2->u.io:$9000EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_0_AENA3->u.io:$90016C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_0_AENA4->u.io:$9001EC.8.1; // AENA4 output status, output channels 7-8
```

ACC-59E3 Addressed as Card 1 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 1
// Power PMAC I/O Base Address Offset=$904000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_1_OUT1->s.io:$904040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_1_OUT2->s.io:$904044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_1_OUT3->s.io:$9040C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_1_OUT4->s.io:$9040C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_1_OUT5->s.io:$904140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_1_OUT6->s.io:$904144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_1_OUT7->s.io:$9041C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_1_OUT8->s.io:$9041C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_1_IN1->s.io:$904020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_1_IN2->s.io:$904024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_1_IN3->s.io:$904028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_1_IN4->s.io:$90402C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_1_IN5->s.io:$9040A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_1_IN6->s.io:$9040A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_1_IN7->s.io:$9040A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_1_IN8->s.io:$9040AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_1_IN9->s.io:$904120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_1_IN10->s.io:$904124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_1_IN11->s.io:$904128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_1_IN12->s.io:$90412C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_1_IN13->s.io:$9041A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_1_IN14->s.io:$9041A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_1_IN15->s.io:$9041A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_1_IN16->s.io:$9041AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_1_AENA1->u.io:$90406C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_1_AENA2->u.io:$9040EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_1_AENA3->u.io:$90416C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_1_AENA4->u.io:$9041EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 2 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 2
// Power PMAC I/O Base Address Offset=$908000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_2_OUT1->s.io:$908040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_2_OUT2->s.io:$908044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_2_OUT3->s.io:$9080C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_2_OUT4->s.io:$9080C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_2_OUT5->s.io:$908140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_2_OUT6->s.io:$908144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_2_OUT7->s.io:$9081C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_2_OUT8->s.io:$9081C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_2_IN1->s.io:$908020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_2_IN2->s.io:$908024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_2_IN3->s.io:$908028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_2_IN4->s.io:$90802C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_2_IN5->s.io:$9080A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_2_IN6->s.io:$9080A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_2_IN7->s.io:$9080A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_2_IN8->s.io:$9080AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_2_IN9->s.io:$908120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_2_IN10->s.io:$908124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_2_IN11->s.io:$908128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_2_IN12->s.io:$90812C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_2_IN13->s.io:$9081A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_2_IN14->s.io:$9081A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_2_IN15->s.io:$9081A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_2_IN16->s.io:$9081AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_2_AENA1->u.io:$90806C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_2_AENA2->u.io:$9080EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_2_AENA3->u.io:$90816C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_2_AENA4->u.io:$9081EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 3 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 3
// Power PMAC I/O Base Address Offset=$90C000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_3_OUT1->s.io:$90C040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_3_OUT2->s.io:$90C044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_3_OUT3->s.io:$90C0C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_3_OUT4->s.io:$90C0C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_3_OUT5->s.io:$90C140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_3_OUT6->s.io:$90C144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_3_OUT7->s.io:$90C1C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_3_OUT8->s.io:$90C1C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_3_IN1->s.io:$90C020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_3_IN2->s.io:$90C024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_3_IN3->s.io:$90C028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_3_IN4->s.io:$90C02C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_3_IN5->s.io:$90C0A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_3_IN6->s.io:$90C0A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_3_IN7->s.io:$90C0A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_3_IN8->s.io:$90C0AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_3_IN9->s.io:$90C120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_3_IN10->s.io:$90C124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_3_IN11->s.io:$90C128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_3_IN12->s.io:$90C12C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_3_IN13->s.io:$90C1A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_3_IN14->s.io:$90C1A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_3_IN15->s.io:$90C1A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_3_IN16->s.io:$90C1AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_3_AENA1->u.io:$90C06C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_3_AENA2->u.io:$90C0EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_3_AENA3->u.io:$90C16C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_3_AENA4->u.io:$90C1EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 4 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 4
// Power PMAC I/O Base Address Offset=$910000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_4_OUT1->s.io:$910040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_4_OUT2->s.io:$910044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_4_OUT3->s.io:$9100C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_4_OUT4->s.io:$9100C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_4_OUT5->s.io:$910140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_4_OUT6->s.io:$910144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_4_OUT7->s.io:$9101C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_4_OUT8->s.io:$9101C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_4_IN1->s.io:$910020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_4_IN2->s.io:$910024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_4_IN3->s.io:$910028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_4_IN4->s.io:$91002C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_4_IN5->s.io:$9100A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_4_IN6->s.io:$9100A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_4_IN7->s.io:$9100A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_4_IN8->s.io:$9100AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_4_IN9->s.io:$910120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_4_IN10->s.io:$910124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_4_IN11->s.io:$910128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_4_IN12->s.io:$91012C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_4_IN13->s.io:$9101A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_4_IN14->s.io:$9101A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_4_IN15->s.io:$9101A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_4_IN16->s.io:$9101AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_4_AENA1->u.io:$91006C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_4_AENA2->u.io:$9100EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_4_AENA3->u.io:$91016C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_4_AENA4->u.io:$9101EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 5 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 5
// Power PMAC I/O Base Address Offset=$914000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_5_OUT1->s.io:$914040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_5_OUT2->s.io:$914044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_5_OUT3->s.io:$9140C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_5_OUT4->s.io:$9140C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_5_OUT5->s.io:$914140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_5_OUT6->s.io:$914144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_5_OUT7->s.io:$9141C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_5_OUT8->s.io:$9141C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_5_IN1->s.io:$914020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_5_IN2->s.io:$914024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_5_IN3->s.io:$914028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_5_IN4->s.io:$91402C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_5_IN5->s.io:$9140A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_5_IN6->s.io:$9140A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_5_IN7->s.io:$9140A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_5_IN8->s.io:$9140AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_5_IN9->s.io:$914120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_5_IN10->s.io:$914124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_5_IN11->s.io:$914128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_5_IN12->s.io:$91412C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_5_IN13->s.io:$9141A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_5_IN14->s.io:$9141A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_5_IN15->s.io:$9141A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_5_IN16->s.io:$9141AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_5_AENA1->u.io:$91406C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_5_AENA2->u.io:$9140EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_5_AENA3->u.io:$91416C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_5_AENA4->u.io:$9141EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 6 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 6
// Power PMAC I/O Base Address Offset=$918000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_6_OUT1->s.io:$918040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_6_OUT2->s.io:$918044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_6_OUT3->s.io:$9180C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_6_OUT4->s.io:$9180C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_6_OUT5->s.io:$918140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_6_OUT6->s.io:$918144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_6_OUT7->s.io:$9181C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_6_OUT8->s.io:$9181C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_6_IN1->s.io:$918020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_6_IN2->s.io:$918024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_6_IN3->s.io:$918028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_6_IN4->s.io:$91802C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_6_IN5->s.io:$9180A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_6_IN6->s.io:$9180A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_6_IN7->s.io:$9180A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_6_IN8->s.io:$9180AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_6_IN9->s.io:$918120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_6_IN10->s.io:$918124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_6_IN11->s.io:$918128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_6_IN12->s.io:$91812C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_6_IN13->s.io:$9181A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_6_IN14->s.io:$9181A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_6_IN15->s.io:$9181A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_6_IN16->s.io:$9181AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_6_AENA1->u.io:$91806C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_6_AENA2->u.io:$9180EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_6_AENA3->u.io:$91816C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_6_AENA4->u.io:$9181EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 7 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 7
// Power PMAC I/O Base Address Offset=$91C000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_7_OUT1->s.io:$91C040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_7_OUT2->s.io:$91C044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_7_OUT3->s.io:$91C0C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_7_OUT4->s.io:$91C0C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_7_OUT5->s.io:$91C140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_7_OUT6->s.io:$91C144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_7_OUT7->s.io:$91C1C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_7_OUT8->s.io:$91C1C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_7_IN1->s.io:$91C020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_7_IN2->s.io:$91C024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_7_IN3->s.io:$91C028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_7_IN4->s.io:$91C02C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_7_IN5->s.io:$91C0A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_7_IN6->s.io:$91C0A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_7_IN7->s.io:$91C0A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_7_IN8->s.io:$91C0AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_7_IN9->s.io:$91C120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_7_IN10->s.io:$91C124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_7_IN11->s.io:$91C128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_7_IN12->s.io:$91C12C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_7_IN13->s.io:$91C1A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_7_IN14->s.io:$91C1A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_7_IN15->s.io:$91C1A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_7_IN16->s.io:$91C1AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_7_AENA1->u.io:$91C06C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_7_AENA2->u.io:$91C0EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_7_AENA3->u.io:$91C16C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_7_AENA4->u.io:$91C1EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 8 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 8
// Power PMAC I/O Base Address Offset=$920000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_8_OUT1->s.io:$920040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_8_OUT2->s.io:$920044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_8_OUT3->s.io:$9200C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_8_OUT4->s.io:$9200C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_8_OUT5->s.io:$920140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_8_OUT6->s.io:$920144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_8_OUT7->s.io:$9201C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_8_OUT8->s.io:$9201C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_8_IN1->s.io:$920020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_8_IN2->s.io:$920024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_8_IN3->s.io:$920028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_8_IN4->s.io:$92002C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_8_IN5->s.io:$9200A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_8_IN6->s.io:$9200A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_8_IN7->s.io:$9200A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_8_IN8->s.io:$9200AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_8_IN9->s.io:$920120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_8_IN10->s.io:$920124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_8_IN11->s.io:$920128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_8_IN12->s.io:$92012C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_8_IN13->s.io:$9201A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_8_IN14->s.io:$9201A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_8_IN15->s.io:$9201A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_8_IN16->s.io:$9201AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_8_AENA1->u.io:$92006C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_8_AENA2->u.io:$9200EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_8_AENA3->u.io:$92016C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_8_AENA4->u.io:$9201EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 9 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 9
// Power PMAC I/O Base Address Offset=$924000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_9_OUT1->s.io:$924040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_9_OUT2->s.io:$924044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_9_OUT3->s.io:$9240C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_9_OUT4->s.io:$9240C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_9_OUT5->s.io:$924140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_9_OUT6->s.io:$924144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_9_OUT7->s.io:$9241C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_9_OUT8->s.io:$9241C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_9_IN1->s.io:$924020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_9_IN2->s.io:$924024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_9_IN3->s.io:$924028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_9_IN4->s.io:$92402C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_9_IN5->s.io:$9240A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_9_IN6->s.io:$9240A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_9_IN7->s.io:$9240A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_9_IN8->s.io:$9240AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_9_IN9->s.io:$924120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_9_IN10->s.io:$924124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_9_IN11->s.io:$924128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_9_IN12->s.io:$92412C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_9_IN13->s.io:$9241A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_9_IN14->s.io:$9241A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_9_IN15->s.io:$9241A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_9_IN16->s.io:$9241AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_9_AENA1->u.io:$92406C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_9_AENA2->u.io:$9240EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_9_AENA3->u.io:$92416C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_9_AENA4->u.io:$9241EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 10 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 10
// Power PMAC I/O Base Address Offset=$928000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_10_OUT1->s.io:$928040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_10_OUT2->s.io:$928044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_10_OUT3->s.io:$9280C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_10_OUT4->s.io:$9280C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_10_OUT5->s.io:$928140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_10_OUT6->s.io:$928144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_10_OUT7->s.io:$9281C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_10_OUT8->s.io:$9281C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_10_IN1->s.io:$928020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_10_IN2->s.io:$928024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_10_IN3->s.io:$928028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_10_IN4->s.io:$92802C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_10_IN5->s.io:$9280A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_10_IN6->s.io:$9280A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_10_IN7->s.io:$9280A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_10_IN8->s.io:$9280AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_10_IN9->s.io:$928120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_10_IN10->s.io:$928124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_10_IN11->s.io:$928128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_10_IN12->s.io:$92812C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_10_IN13->s.io:$9281A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_10_IN14->s.io:$9281A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_10_IN15->s.io:$9281A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_10_IN16->s.io:$9281AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_10_AENA1->u.io:$92806C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_10_AENA2->u.io:$9280EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_10_AENA3->u.io:$92816C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_10_AENA4->u.io:$9281EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 11 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 11
// Power PMAC I/O Base Address Offset=$92C000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_11_OUT1->s.io:$92C040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_11_OUT2->s.io:$92C044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_11_OUT3->s.io:$92C0C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_11_OUT4->s.io:$92C0C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_11_OUT5->s.io:$92C140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_11_OUT6->s.io:$92C144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_11_OUT7->s.io:$92C1C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_11_OUT8->s.io:$92C1C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_11_IN1->s.io:$92C020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_11_IN2->s.io:$92C024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_11_IN3->s.io:$92C028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_11_IN4->s.io:$92C02C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_11_IN5->s.io:$92C0A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_11_IN6->s.io:$92C0A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_11_IN7->s.io:$92C0A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_11_IN8->s.io:$92C0AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_11_IN9->s.io:$92C120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_11_IN10->s.io:$92C124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_11_IN11->s.io:$92C128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_11_IN12->s.io:$92C12C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_11_IN13->s.io:$92C1A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_11_IN14->s.io:$92C1A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_11_IN15->s.io:$92C1A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_11_IN16->s.io:$92C1AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_11_AENA1->u.io:$92C06C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_11_AENA2->u.io:$92C0EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_11_AENA3->u.io:$92C16C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_11_AENA4->u.io:$92C1EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 12 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 12
// Power PMAC I/O Base Address Offset=$930000
// 16-bit ADCs

//***** Output Pointers *****//
// Mezzanine on top side
ptr ACC59E3_12_OUT1->s.io:$930040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_12_OUT2->s.io:$930044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_12_OUT3->s.io:$9300C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_12_OUT4->s.io:$9300C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_12_OUT5->s.io:$930140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_12_OUT6->s.io:$930144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_12_OUT7->s.io:$9301C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_12_OUT8->s.io:$9301C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****//
ptr ACC59E3_12_IN1->s.io:$930020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_12_IN2->s.io:$930024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_12_IN3->s.io:$930028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_12_IN4->s.io:$93002C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_12_IN5->s.io:$9300A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_12_IN6->s.io:$9300A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_12_IN7->s.io:$9300A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_12_IN8->s.io:$9300AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_12_IN9->s.io:$930120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_12_IN10->s.io:$930124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_12_IN11->s.io:$930128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_12_IN12->s.io:$93012C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_12_IN13->s.io:$9301A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_12_IN14->s.io:$9301A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_12_IN15->s.io:$9301A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_12_IN16->s.io:$9301AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****//
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****//
ptr ACC59E3_12_AENA1->u.io:$93006C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_12_AENA2->u.io:$9300EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_12_AENA3->u.io:$93016C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_12_AENA4->u.io:$9301EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 13 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 13
// Power PMAC I/O Base Address Offset=$934000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_13_OUT1->s.io:$934040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_13_OUT2->s.io:$934044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_13_OUT3->s.io:$9340C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_13_OUT4->s.io:$9340C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_13_OUT5->s.io:$934140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_13_OUT6->s.io:$934144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_13_OUT7->s.io:$9341C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_13_OUT8->s.io:$9341C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_13_IN1->s.io:$934020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_13_IN2->s.io:$934024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_13_IN3->s.io:$934028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_13_IN4->s.io:$93402C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_13_IN5->s.io:$9340A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_13_IN6->s.io:$9340A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_13_IN7->s.io:$9340A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_13_IN8->s.io:$9340AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_13_IN9->s.io:$934120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_13_IN10->s.io:$934124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_13_IN11->s.io:$934128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_13_IN12->s.io:$93412C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_13_IN13->s.io:$9341A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_13_IN14->s.io:$9341A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_13_IN15->s.io:$9341A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_13_IN16->s.io:$9341AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_13_AENA1->u.io:$93406C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_13_AENA2->u.io:$9340EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_13_AENA3->u.io:$93416C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_13_AENA4->u.io:$9341EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 14 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 14
// Power PMAC I/O Base Address Offset=$938000
// 16-bit ADCs

//***** Output Pointers *****//
// Mezzanine on top side
ptr ACC59E3_14_OUT1->s.io:$938040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_14_OUT2->s.io:$938044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_14_OUT3->s.io:$9380C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_14_OUT4->s.io:$9380C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_14_OUT5->s.io:$938140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_14_OUT6->s.io:$938144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_14_OUT7->s.io:$9381C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_14_OUT8->s.io:$9381C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****//
ptr ACC59E3_14_IN1->s.io:$938020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_14_IN2->s.io:$938024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_14_IN3->s.io:$938028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_14_IN4->s.io:$93802C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_14_IN5->s.io:$9380A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_14_IN6->s.io:$9380A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_14_IN7->s.io:$9380A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_14_IN8->s.io:$9380AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_14_IN9->s.io:$938120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_14_IN10->s.io:$938124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_14_IN11->s.io:$938128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_14_IN12->s.io:$93812C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_14_IN13->s.io:$9381A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_14_IN14->s.io:$9381A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_14_IN15->s.io:$9381A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_14_IN16->s.io:$9381AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****//
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****//
ptr ACC59E3_14_AENA1->u.io:$93806C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_14_AENA2->u.io:$9380EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_14_AENA3->u.io:$93816C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_14_AENA4->u.io:$9381EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 15 with 16-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 15
// Power PMAC I/O Base Address Offset=$93C000
// 16-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_15_OUT1->s.io:$93C040.16.16; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_15_OUT2->s.io:$93C044.16.16; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_15_OUT3->s.io:$93C0C0.16.16; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_15_OUT4->s.io:$93C0C4.16.16; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_15_OUT5->s.io:$93C140.16.16; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_15_OUT6->s.io:$93C144.16.16; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_15_OUT7->s.io:$93C1C0.16.16; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_15_OUT8->s.io:$93C1C4.16.16; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_15_IN1->s.io:$93C020.16.16; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_15_IN2->s.io:$93C024.16.16; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_15_IN3->s.io:$93C028.16.16; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_15_IN4->s.io:$93C02C.16.16; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_15_IN5->s.io:$93C0A0.16.16; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_15_IN6->s.io:$93C0A4.16.16; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_15_IN7->s.io:$93C0A8.16.16; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_15_IN8->s.io:$93C0AC.16.16; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_15_IN9->s.io:$93C120.16.16; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_15_IN10->s.io:$93C124.16.16; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_15_IN11->s.io:$93C128.16.16; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_15_IN12->s.io:$93C12C.16.16; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_15_IN13->s.io:$93C1A0.16.16; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_15_IN14->s.io:$93C1A4.16.16; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_15_IN15->s.io:$93C1A8.16.16; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_15_IN16->s.io:$93C1AC.16.16; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_15_AENA1->u.io:$93C06C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_15_AENA2->u.io:$93C0EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_15_AENA3->u.io:$93C16C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_15_AENA4->u.io:$93C1EC.8.1; // AENA4 output status, output channels 7-8

```

12-bit ADC Option

ACC-59E3 Addressed as Card 0 with 12-bit ADCs

```
// Suggested M-Variables for ACC-59E3 Addressed as Card 0
// Power PMAC I/O Base Address Offset=$900000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_0_OUT1->s.io:$900040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_0_OUT2->s.io:$900044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_0_OUT3->s.io:$9000C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_0_OUT4->s.io:$9000C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_0_OUT5->s.io:$900140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_0_OUT6->s.io:$900144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_0_OUT7->s.io:$9001C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_0_OUT8->s.io:$9001C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_0_IN1->s.io:$900020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_0_IN2->s.io:$900024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_0_IN3->s.io:$900028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_0_IN4->s.io:$90002C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_0_IN5->s.io:$9000A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_0_IN6->s.io:$9000A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_0_IN7->s.io:$9000A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_0_IN8->s.io:$9000AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_0_IN9->s.io:$900120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_0_IN10->s.io:$900124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_0_IN11->s.io:$900128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_0_IN12->s.io:$90012C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_0_IN13->s.io:$9001A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_0_IN14->s.io:$9001A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_0_IN15->s.io:$9001A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_0_IN16->s.io:$9001AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_0_AENA1->u.io:$90006C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_0_AENA2->u.io:$9000EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_0_AENA3->u.io:$90016C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_0_AENA4->u.io:$9001EC.8.1; // AENA4 output status, output channels 7-8
```

ACC-59E3 Addressed as Card 1 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 1
// Power PMAC I/O Base Address Offset=$904000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_1_OUT1->s.io:$904040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_1_OUT2->s.io:$904044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_1_OUT3->s.io:$9040C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_1_OUT4->s.io:$9040C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_1_OUT5->s.io:$904140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_1_OUT6->s.io:$904144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_1_OUT7->s.io:$9041C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_1_OUT8->s.io:$9041C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_1_IN1->s.io:$904020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_1_IN2->s.io:$904024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_1_IN3->s.io:$904028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_1_IN4->s.io:$90402C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_1_IN5->s.io:$9040A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_1_IN6->s.io:$9040A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_1_IN7->s.io:$9040A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_1_IN8->s.io:$9040AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_1_IN9->s.io:$904120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_1_IN10->s.io:$904124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_1_IN11->s.io:$904128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_1_IN12->s.io:$90412C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_1_IN13->s.io:$9041A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_1_IN14->s.io:$9041A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_1_IN15->s.io:$9041A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_1_IN16->s.io:$9041AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_1_AENA1->u.io:$90406C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_1_AENA2->u.io:$9040EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_1_AENA3->u.io:$90416C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_1_AENA4->u.io:$9041EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 2 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 2
// Power PMAC I/O Base Address Offset=$908000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_2_OUT1->s.io:$908040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_2_OUT2->s.io:$908044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_2_OUT3->s.io:$9080C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_2_OUT4->s.io:$9080C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_2_OUT5->s.io:$908140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_2_OUT6->s.io:$908144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_2_OUT7->s.io:$9081C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_2_OUT8->s.io:$9081C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_2_IN1->s.io:$908020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_2_IN2->s.io:$908024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_2_IN3->s.io:$908028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_2_IN4->s.io:$90802C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_2_IN5->s.io:$9080A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_2_IN6->s.io:$9080A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_2_IN7->s.io:$9080A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_2_IN8->s.io:$9080AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_2_IN9->s.io:$908120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_2_IN10->s.io:$908124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_2_IN11->s.io:$908128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_2_IN12->s.io:$90812C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_2_IN13->s.io:$9081A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_2_IN14->s.io:$9081A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_2_IN15->s.io:$9081A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_2_IN16->s.io:$9081AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_2_AENA1->u.io:$90806C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_2_AENA2->u.io:$9080EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_2_AENA3->u.io:$90816C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_2_AENA4->u.io:$9081EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 3 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 3
// Power PMAC I/O Base Address Offset=$90C000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_3_OUT1->s.io:$90C040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_3_OUT2->s.io:$90C044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_3_OUT3->s.io:$90C0C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_3_OUT4->s.io:$90C0C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_3_OUT5->s.io:$90C140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_3_OUT6->s.io:$90C144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_3_OUT7->s.io:$90C1C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_3_OUT8->s.io:$90C1C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_3_IN1->s.io:$90C020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_3_IN2->s.io:$90C024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_3_IN3->s.io:$90C028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_3_IN4->s.io:$90C02C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_3_IN5->s.io:$90C0A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_3_IN6->s.io:$90C0A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_3_IN7->s.io:$90C0A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_3_IN8->s.io:$90C0AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_3_IN9->s.io:$90C120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_3_IN10->s.io:$90C124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_3_IN11->s.io:$90C128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_3_IN12->s.io:$90C12C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_3_IN13->s.io:$90C1A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_3_IN14->s.io:$90C1A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_3_IN15->s.io:$90C1A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_3_IN16->s.io:$90C1AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_3_AENA1->u.io:$90C06C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_3_AENA2->u.io:$90C0EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_3_AENA3->u.io:$90C16C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_3_AENA4->u.io:$90C1EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 4 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 4
// Power PMAC I/O Base Address Offset=$910000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_4_OUT1->s.io:$910040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_4_OUT2->s.io:$910044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_4_OUT3->s.io:$9100C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_4_OUT4->s.io:$9100C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_4_OUT5->s.io:$910140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_4_OUT6->s.io:$910144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_4_OUT7->s.io:$9101C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_4_OUT8->s.io:$9101C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_4_IN1->s.io:$910020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_4_IN2->s.io:$910024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_4_IN3->s.io:$910028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_4_IN4->s.io:$91002C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_4_IN5->s.io:$9100A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_4_IN6->s.io:$9100A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_4_IN7->s.io:$9100A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_4_IN8->s.io:$9100AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_4_IN9->s.io:$910120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_4_IN10->s.io:$910124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_4_IN11->s.io:$910128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_4_IN12->s.io:$91012C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_4_IN13->s.io:$9101A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_4_IN14->s.io:$9101A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_4_IN15->s.io:$9101A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_4_IN16->s.io:$9101AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_4_AENA1->u.io:$91006C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_4_AENA2->u.io:$9100EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_4_AENA3->u.io:$91016C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_4_AENA4->u.io:$9101EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 5 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 5
// Power PMAC I/O Base Address Offset=$914000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_5_OUT1->s.io:$914040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_5_OUT2->s.io:$914044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_5_OUT3->s.io:$9140C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_5_OUT4->s.io:$9140C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_5_OUT5->s.io:$914140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_5_OUT6->s.io:$914144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_5_OUT7->s.io:$9141C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_5_OUT8->s.io:$9141C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_5_IN1->s.io:$914020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_5_IN2->s.io:$914024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_5_IN3->s.io:$914028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_5_IN4->s.io:$91402C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_5_IN5->s.io:$9140A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_5_IN6->s.io:$9140A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_5_IN7->s.io:$9140A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_5_IN8->s.io:$9140AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_5_IN9->s.io:$914120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_5_IN10->s.io:$914124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_5_IN11->s.io:$914128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_5_IN12->s.io:$91412C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_5_IN13->s.io:$9141A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_5_IN14->s.io:$9141A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_5_IN15->s.io:$9141A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_5_IN16->s.io:$9141AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_5_AENA1->u.io:$91406C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_5_AENA2->u.io:$9140EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_5_AENA3->u.io:$91416C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_5_AENA4->u.io:$9141EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 6 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 6
// Power PMAC I/O Base Address Offset=$918000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_6_OUT1->s.io:$918040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_6_OUT2->s.io:$918044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_6_OUT3->s.io:$9180C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_6_OUT4->s.io:$9180C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_6_OUT5->s.io:$918140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_6_OUT6->s.io:$918144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_6_OUT7->s.io:$9181C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_6_OUT8->s.io:$9181C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_6_IN1->s.io:$918020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_6_IN2->s.io:$918024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_6_IN3->s.io:$918028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_6_IN4->s.io:$91802C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_6_IN5->s.io:$9180A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_6_IN6->s.io:$9180A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_6_IN7->s.io:$9180A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_6_IN8->s.io:$9180AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_6_IN9->s.io:$918120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_6_IN10->s.io:$918124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_6_IN11->s.io:$918128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_6_IN12->s.io:$91812C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_6_IN13->s.io:$9181A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_6_IN14->s.io:$9181A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_6_IN15->s.io:$9181A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_6_IN16->s.io:$9181AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_6_AENA1->u.io:$91806C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_6_AENA2->u.io:$9180EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_6_AENA3->u.io:$91816C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_6_AENA4->u.io:$9181EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 7 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 7
// Power PMAC I/O Base Address Offset=$91C000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_7_OUT1->s.io:$91C040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_7_OUT2->s.io:$91C044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_7_OUT3->s.io:$91C0C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_7_OUT4->s.io:$91C0C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_7_OUT5->s.io:$91C140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_7_OUT6->s.io:$91C144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_7_OUT7->s.io:$91C1C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_7_OUT8->s.io:$91C1C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_7_IN1->s.io:$91C020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_7_IN2->s.io:$91C024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_7_IN3->s.io:$91C028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_7_IN4->s.io:$91C02C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_7_IN5->s.io:$91C0A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_7_IN6->s.io:$91C0A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_7_IN7->s.io:$91C0A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_7_IN8->s.io:$91C0AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_7_IN9->s.io:$91C120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_7_IN10->s.io:$91C124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_7_IN11->s.io:$91C128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_7_IN12->s.io:$91C12C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_7_IN13->s.io:$91C1A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_7_IN14->s.io:$91C1A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_7_IN15->s.io:$91C1A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_7_IN16->s.io:$91C1AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_7_AENA1->u.io:$91C06C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_7_AENA2->u.io:$91C0EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_7_AENA3->u.io:$91C16C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_7_AENA4->u.io:$91C1EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 8 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 8
// Power PMAC I/O Base Address Offset=$920000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_8_OUT1->s.io:$920040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_8_OUT2->s.io:$920044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_8_OUT3->s.io:$9200C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_8_OUT4->s.io:$9200C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_8_OUT5->s.io:$920140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_8_OUT6->s.io:$920144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_8_OUT7->s.io:$9201C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_8_OUT8->s.io:$9201C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_8_IN1->s.io:$920020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_8_IN2->s.io:$920024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_8_IN3->s.io:$920028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_8_IN4->s.io:$92002C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_8_IN5->s.io:$9200A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_8_IN6->s.io:$9200A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_8_IN7->s.io:$9200A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_8_IN8->s.io:$9200AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_8_IN9->s.io:$920120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_8_IN10->s.io:$920124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_8_IN11->s.io:$920128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_8_IN12->s.io:$92012C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_8_IN13->s.io:$9201A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_8_IN14->s.io:$9201A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_8_IN15->s.io:$9201A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_8_IN16->s.io:$9201AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_8_AENA1->u.io:$92006C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_8_AENA2->u.io:$9200EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_8_AENA3->u.io:$92016C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_8_AENA4->u.io:$9201EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 9 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 9
// Power PMAC I/O Base Address Offset=$924000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_9_OUT1->s.io:$924040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_9_OUT2->s.io:$924044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_9_OUT3->s.io:$9240C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_9_OUT4->s.io:$9240C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_9_OUT5->s.io:$924140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_9_OUT6->s.io:$924144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_9_OUT7->s.io:$9241C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_9_OUT8->s.io:$9241C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_9_IN1->s.io:$924020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_9_IN2->s.io:$924024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_9_IN3->s.io:$924028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_9_IN4->s.io:$92402C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_9_IN5->s.io:$9240A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_9_IN6->s.io:$9240A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_9_IN7->s.io:$9240A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_9_IN8->s.io:$9240AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_9_IN9->s.io:$924120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_9_IN10->s.io:$924124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_9_IN11->s.io:$924128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_9_IN12->s.io:$92412C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_9_IN13->s.io:$9241A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_9_IN14->s.io:$9241A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_9_IN15->s.io:$9241A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_9_IN16->s.io:$9241AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_9_AENA1->u.io:$92406C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_9_AENA2->u.io:$9240EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_9_AENA3->u.io:$92416C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_9_AENA4->u.io:$9241EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 10 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 10
// Power PMAC I/O Base Address Offset=$928000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_10_OUT1->s.io:$928040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_10_OUT2->s.io:$928044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_10_OUT3->s.io:$9280C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_10_OUT4->s.io:$9280C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_10_OUT5->s.io:$928140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_10_OUT6->s.io:$928144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_10_OUT7->s.io:$9281C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_10_OUT8->s.io:$9281C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_10_IN1->s.io:$928020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_10_IN2->s.io:$928024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_10_IN3->s.io:$928028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_10_IN4->s.io:$92802C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_10_IN5->s.io:$9280A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_10_IN6->s.io:$9280A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_10_IN7->s.io:$9280A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_10_IN8->s.io:$9280AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_10_IN9->s.io:$928120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_10_IN10->s.io:$928124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_10_IN11->s.io:$928128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_10_IN12->s.io:$92812C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_10_IN13->s.io:$9281A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_10_IN14->s.io:$9281A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_10_IN15->s.io:$9281A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_10_IN16->s.io:$9281AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_10_AENA1->u.io:$92806C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_10_AENA2->u.io:$9280EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_10_AENA3->u.io:$92816C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_10_AENA4->u.io:$9281EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 11 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 11
// Power PMAC I/O Base Address Offset=$92C000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_11_OUT1->s.io:$92C040.16.12;    // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_11_OUT2->s.io:$92C044.16.12;    // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_11_OUT3->s.io:$92C0C0.16.12;    // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_11_OUT4->s.io:$92C0C4.16.12;    // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_11_OUT5->s.io:$92C140.16.12;    // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_11_OUT6->s.io:$92C144.16.12;    // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_11_OUT7->s.io:$92C1C0.16.12;    // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_11_OUT8->s.io:$92C1C4.16.12;    // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_11_IN1->s.io:$92C020.16.12;    // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_11_IN2->s.io:$92C024.16.12;    // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_11_IN3->s.io:$92C028.16.12;    // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_11_IN4->s.io:$92C02C.16.12;    // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_11_IN5->s.io:$92C0A0.16.12;    // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_11_IN6->s.io:$92C0A4.16.12;    // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_11_IN7->s.io:$92C0A8.16.12;    // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_11_IN8->s.io:$92C0AC.16.12;    // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_11_IN9->s.io:$92C120.16.12;    // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_11_IN10->s.io:$92C124.16.12;    // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_11_IN11->s.io:$92C128.16.12;    // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_11_IN12->s.io:$92C12C.16.12;    // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_11_IN13->s.io:$92C1A0.16.12;    // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_11_IN14->s.io:$92C1A4.16.12;    // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_11_IN15->s.io:$92C1A8.16.12;    // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_11_IN16->s.io:$92C1AC.16.12;    // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_11_AENA1->u.io:$92C06C.8.1;    // AENA1 output status, output channels 1-2
ptr ACC59E3_11_AENA2->u.io:$92C0EC.8.1;    // AENA2 output status, output channels 3-4
ptr ACC59E3_11_AENA3->u.io:$92C16C.8.1;    // AENA3 output status, output channels 5-6
ptr ACC59E3_11_AENA4->u.io:$92C1EC.8.1;    // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 12 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 12
// Power PMAC I/O Base Address Offset=$930000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_12_OUT1->s.io:$930040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_12_OUT2->s.io:$930044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_12_OUT3->s.io:$9300C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_12_OUT4->s.io:$9300C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_12_OUT5->s.io:$930140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_12_OUT6->s.io:$930144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_12_OUT7->s.io:$9301C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_12_OUT8->s.io:$9301C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_12_IN1->s.io:$930020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_12_IN2->s.io:$930024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_12_IN3->s.io:$930028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_12_IN4->s.io:$93002C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_12_IN5->s.io:$9300A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_12_IN6->s.io:$9300A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_12_IN7->s.io:$9300A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_12_IN8->s.io:$9300AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_12_IN9->s.io:$930120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_12_IN10->s.io:$930124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_12_IN11->s.io:$930128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_12_IN12->s.io:$93012C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_12_IN13->s.io:$9301A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_12_IN14->s.io:$9301A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_12_IN15->s.io:$9301A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_12_IN16->s.io:$9301AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_12_AENA1->u.io:$93006C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_12_AENA2->u.io:$9300EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_12_AENA3->u.io:$93016C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_12_AENA4->u.io:$9301EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 13 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 13
// Power PMAC I/O Base Address Offset=$934000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_13_OUT1->s.io:$934040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_13_OUT2->s.io:$934044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_13_OUT3->s.io:$9340C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_13_OUT4->s.io:$9340C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_13_OUT5->s.io:$934140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_13_OUT6->s.io:$934144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_13_OUT7->s.io:$9341C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_13_OUT8->s.io:$9341C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_13_IN1->s.io:$934020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_13_IN2->s.io:$934024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_13_IN3->s.io:$934028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_13_IN4->s.io:$93402C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_13_IN5->s.io:$9340A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_13_IN6->s.io:$9340A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_13_IN7->s.io:$9340A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_13_IN8->s.io:$9340AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_13_IN9->s.io:$934120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_13_IN10->s.io:$934124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_13_IN11->s.io:$934128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_13_IN12->s.io:$93412C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_13_IN13->s.io:$9341A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_13_IN14->s.io:$9341A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_13_IN15->s.io:$9341A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_13_IN16->s.io:$9341AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_13_AENA1->u.io:$93406C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_13_AENA2->u.io:$9340EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_13_AENA3->u.io:$93416C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_13_AENA4->u.io:$9341EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 14 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 14
// Power PMAC I/O Base Address Offset=$938000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_14_OUT1->s.io:$938040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_14_OUT2->s.io:$938044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_14_OUT3->s.io:$9380C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_14_OUT4->s.io:$9380C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_14_OUT5->s.io:$938140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_14_OUT6->s.io:$938144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_14_OUT7->s.io:$9381C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_14_OUT8->s.io:$9381C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_14_IN1->s.io:$938020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_14_IN2->s.io:$938024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_14_IN3->s.io:$938028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_14_IN4->s.io:$93802C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_14_IN5->s.io:$9380A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_14_IN6->s.io:$9380A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_14_IN7->s.io:$9380A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_14_IN8->s.io:$9380AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_14_IN9->s.io:$938120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_14_IN10->s.io:$938124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_14_IN11->s.io:$938128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_14_IN12->s.io:$93812C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_14_IN13->s.io:$9381A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_14_IN14->s.io:$9381A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_14_IN15->s.io:$9381A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_14_IN16->s.io:$9381AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_14_AENA1->u.io:$93806C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_14_AENA2->u.io:$9380EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_14_AENA3->u.io:$93816C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_14_AENA4->u.io:$9381EC.8.1; // AENA4 output status, output channels 7-8

```

ACC-59E3 Addressed as Card 15 with 12-bit ADCs

```

// Suggested M-Variables for ACC-59E3 Addressed as Card 15
// Power PMAC I/O Base Address Offset=$93C000
// 12-bit ADCs

//***** Output Pointers *****/
// Mezzanine on top side
ptr ACC59E3_15_OUT1->s.io:$93C040.16.12; // ACC-59E3 J1 OUT1 - Top
ptr ACC59E3_15_OUT2->s.io:$93C044.16.12; // ACC-59E3 J1 OUT2 - Top
ptr ACC59E3_15_OUT3->s.io:$93C0C0.16.12; // ACC-59E3 J2 OUT3 - Top
ptr ACC59E3_15_OUT4->s.io:$93C0C4.16.12; // ACC-59E3 J2 OUT4 - Top

// Mezzanine on bottom side
ptr ACC59E3_15_OUT5->s.io:$93C140.16.12; // ACC-59E3 J1 OUT5 - Bottom
ptr ACC59E3_15_OUT6->s.io:$93C144.16.12; // ACC-59E3 J1 OUT6 - Bottom
ptr ACC59E3_15_OUT7->s.io:$93C1C0.16.12; // ACC-59E3 J2 OUT7 - Bottom
ptr ACC59E3_15_OUT8->s.io:$93C1C4.16.12; // ACC-59E3 J2 OUT8 - Bottom

//***** Input Pointers *****/
ptr ACC59E3_15_IN1->s.io:$93C020.16.12; // ACC-59E3 J1 IN1 - Top
ptr ACC59E3_15_IN2->s.io:$93C024.16.12; // ACC-59E3 J1 IN2 - Top
ptr ACC59E3_15_IN3->s.io:$93C028.16.12; // ACC-59E3 J1 IN3 - Top
ptr ACC59E3_15_IN4->s.io:$93C02C.16.12; // ACC-59E3 J1 IN4 - Top

ptr ACC59E3_15_IN5->s.io:$93C0A0.16.12; // ACC-59E3 J2 IN5 - Top
ptr ACC59E3_15_IN6->s.io:$93C0A4.16.12; // ACC-59E3 J2 IN6 - Top
ptr ACC59E3_15_IN7->s.io:$93C0A8.16.12; // ACC-59E3 J2 IN7 - Top
ptr ACC59E3_15_IN8->s.io:$93C0AC.16.12; // ACC-59E3 J2 IN8 - Top

ptr ACC59E3_15_IN9->s.io:$93C120.16.12; // ACC-59E3 J1 IN9 - Bottom
ptr ACC59E3_15_IN10->s.io:$93C124.16.12; // ACC-59E3 J1 IN10 - Bottom
ptr ACC59E3_15_IN11->s.io:$93C128.16.12; // ACC-59E3 J1 IN11 - Bottom
ptr ACC59E3_15_IN12->s.io:$93C12C.16.12; // ACC-59E3 J1 IN12 - Bottom

ptr ACC59E3_15_IN13->s.io:$93C1A0.16.12; // ACC-59E3 J2 IN13 - Bottom
ptr ACC59E3_15_IN14->s.io:$93C1A4.16.12; // ACC-59E3 J2 IN14 - Bottom
ptr ACC59E3_15_IN15->s.io:$93C1A8.16.12; // ACC-59E3 J2 IN15 - Bottom
ptr ACC59E3_15_IN16->s.io:$93C1AC.16.12; // ACC-59E3 J2 IN16 - Bottom

//***** Current Output Amplifier Enable Bit Pointers *****/
//***** NECESSARY FOR CURRENT OUTPUT OPTION ONLY *****/
ptr ACC59E3_15_AENA1->u.io:$93C06C.8.1; // AENA1 output status, output channels 1-2
ptr ACC59E3_15_AENA2->u.io:$93C0EC.8.1; // AENA2 output status, output channels 3-4
ptr ACC59E3_15_AENA3->u.io:$93C16C.8.1; // AENA3 output status, output channels 5-6
ptr ACC59E3_15_AENA4->u.io:$93C1EC.8.1; // AENA4 output status, output channels 7-8

```

APPENDIX E: USING ACC-59E3 IN THE C ENVIRONMENT

ACC-59E3 Header File (acc59e3.h)

Delta Tau has developed a set of functions that make using the ACC-59E3 with the C Programming Language easier. The user can choose to include these functions in his or her C application if desired. In the IDE Project, the user must first create a folder in the C Language→Libraries folder called “acc59e3”, and then create a header file in that folder called “acc59e3.h”. Then, the user must copy the following header file code into that file he or she just created:

```
// acc59e3.h
// Put into C Language-->Include folder
// Contains the ACC59E3 structure definition and functions associated with
// using ACC-59E3 in Power PMAC in the C Programming Language

double SixteenBitVoltsPerSoftwareCount=10.064/32767.0;
double TwelveBitVoltsPerSoftwareCount=10.064/2048.0;
double SixteenBitmAPerSoftwareCount=20.128/32767.0;
double TwelveBitmAPerSoftwareCount=20.128/2048.0;

double SoftwareCountsPerVolt_16bit=32767.0/10.064;
double SoftwareCountsPerVolt_12bit=2048.0/10.064;
double SoftwareCountsPermA_16bit=32767.0/20.128;
double SoftwareCountsPermA_12bit=2048.0/20.128;
#define SoftwareCountsPerSingleEndedVolt SoftwareCountsPerVolt_16bit
#define SoftwareCountsPerDifferentialVolt SoftwareCountsPermA_16bit

double CurrentSF=16.21/100.0*65536.0/4.096;

#define Card0 0
#define Card1 1
#define Card2 2
#define Card3 3
#define Card4 4
#define Card5 5
#define Card6 6
#define Card7 7
#define Card8 8
#define Card9 9
#define Card10 10
#define Card11 11
#define Card12 12
#define Card13 13
#define Card14 14
#define Card15 15

#define SingleEndedWiring 0
#define DifferentialWiring 1

typedef struct ACC59E3
{
    char CardNumber;

    volatile int* OUT1;
    volatile int* OUT2;
    volatile int* OUT3;
    volatile int* OUT4;
    volatile int* OUT5;
    volatile int* OUT6;
    volatile int* OUT7;
    volatile int* OUT8;

    volatile int* IN1;
    volatile int* IN2;
    volatile int* IN3;
    volatile int* IN4;
    volatile int* IN5;
    volatile int* IN6;
    volatile int* IN7;
    volatile int* IN8;
```

```

    volatile int* IN9;
    volatile int* IN10;
    volatile int* IN11;
    volatile int* IN12;
    volatile int* IN13;
    volatile int* IN14;
    volatile int* IN15;
    volatile int* IN16;
} ACC59E3;

// Function Prototypes
struct ACC59E3 ACC59E3_PtrInit(struct ACC59E3 x, unsigned int CardNumberInit);
int ACC59E3_VOUT(struct ACC59E3 x, double Output_Voltage, unsigned int Output_Channel, unsigned
int Wiring_Mode);
int ACC59E3_COUT(struct ACC59E3 x, double Output_Current, unsigned int Output_Channel);
int ACC59E3_IN(struct ACC59E3 x, unsigned int Input_Channel);
double ACC59E3_SC2INV(int Software_Counts, unsigned int Resolution);
double ACC59E3_SC2INC(int Software_Counts, unsigned int Resolution);
int ACC59E3_INV2SC(int Voltage, unsigned int Resolution);
int ACC59E3_INC2SC(int Current, unsigned int Resolution);
int ACC59E3_OUTV2SC(double Output_Voltage, unsigned int Wiring_Mode);
int ACC59E3_OUTC2SC(double Output_Current);
void ACC59E3_FilterInit(char CardNumber, unsigned int FSA[16]);
int ACC59E3_FilterMask(unsigned int Input_Channel, unsigned int FilterType);

struct ACC59E3 ACC59E3_PtrInit(struct ACC59E3 x, unsigned int CardNumberInit)
// Initializes the ACC-59E3 card by pointing its input and output pointers to the appropriate
registers
// Input:
// x: The ACC59E3 structure corresponding to the card the user wants to initialize
// CardNumberInit: The card number which the user is initializing. This is the same number as
//                  the "Power PMAC "Gate3" Index #" from the address table in the
//                  Addressing ACC-59E3 section of the ACC-59E3 manual
// Output:
// Returns the initialized ACC59E3 structure
{
    x.CardNumber=CardNumberInit;
    x.IN1 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x20)/4; // IN Channel 1
    x.IN2 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x24)/4; // IN Channel 2
    x.IN3 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x28)/4; // IN Channel 3
    x.IN4 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x2C)/4; // IN Channel 4
    x.IN5 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0xA0)/4; // IN Channel 5
    x.IN6 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0xA4)/4; // IN Channel 6
    x.IN7 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0xA8)/4; // IN Channel 7
    x.IN8 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0xAC)/4; // IN Channel 8

    x.IN9 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x120)/4; // IN Channel 9
    x.IN10 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x124)/4; // IN Channel 10
    x.IN11 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x128)/4; // IN Channel 11
    x.IN12 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x12C)/4; // IN Channel 12
    x.IN13 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x1A0)/4; // IN Channel 13
    x.IN14 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x1A4)/4; // IN Channel 14
    x.IN15 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x1A8)/4; // IN Channel 15
    x.IN16 = (int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x1AC)/4; // IN Channel 16

    x.OUT1=(int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x40)/4; //OUT Channel 1
    x.OUT2=(int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x44)/4; //OUT Channel 2
    x.OUT3=(int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0xC0)/4; //OUT Channel 3
    x.OUT4=(int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0xC4)/4; //OUT Channel 4
    x.OUT5=(int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x140)/4; //OUT Channel 5
    x.OUT6=(int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x144)/4; //OUT Channel 6
    x.OUT7=(int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x1C0)/4; //OUT Channel 7
    x.OUT8=(int*)piom + (pshm->OffsetGate3[x.CardNumber] + 0x1C4)/4; //OUT Channel 8
    return x;
}

int ACC59E3_VOUT(struct ACC59E3 x, double Output_Voltage, unsigned int Output_Channel, unsigned
int Wiring_Mode)
// This function will output the appropriate voltage from the DACs (for current, use
ACC59E3_COUT())

```

```

// **Inputs:
// x: ACC59E3 structure being used for the present card
// Output_Voltage: Desired output voltage for this DAC channel (volts)
// Output_Channel: The number of the DAC channel out of which the user desired to put voltage
// (indexed 1-16)
// Wiring_Mode: 0 = Single-Ended Wiring
//               1 = Differential Wiring
// **Outputs:
// Returns 0 if everything went well
// Returns -1 if incorrect Output_Channel was given
// **Notes: Will saturate output voltages to  $\pm 10$  V for single-ended wiring,  $\pm 20$  V for
// differential
{
    if(Wiring_Mode == SingleEndedWiring)
    {
        if(Output_Voltage > 10.0)
            Output_Voltage = 10.0;
        else if(Output_Voltage < -10.0)
            Output_Voltage = -10.0;
    }
    else
    {
        if(Output_Voltage > 20.0)
            Output_Voltage = 20.0; // Volts
        else if(Output_Voltage < -20.0)
            Output_Voltage = -20.0; // Volts
    }

    switch (Output_Channel)
    {
        case 1:
            *(x.OUT1) = ((int)ACC59E3_OUTV2SC(Output_Voltage, Wiring_Mode)) << 16;
            break;

        case 2:
            *(x.OUT2) = ((int)ACC59E3_OUTV2SC(Output_Voltage, Wiring_Mode)) << 16;
            break;

        case 3:
            *(x.OUT3) = ((int)ACC59E3_OUTV2SC(Output_Voltage, Wiring_Mode)) << 16;
            break;

        case 4:
            *(x.OUT4) = ((int)ACC59E3_OUTV2SC(Output_Voltage, Wiring_Mode)) << 16;
            break;

        case 5:
            *(x.OUT5) = ((int)ACC59E3_OUTV2SC(Output_Voltage, Wiring_Mode)) << 16;
            break;

        case 6:
            *(x.OUT6) = ((int)ACC59E3_OUTV2SC(Output_Voltage, Wiring_Mode)) << 16;
            break;

        case 7:
            *(x.OUT7) = ((int)ACC59E3_OUTV2SC(Output_Voltage, Wiring_Mode)) << 16;
            break;

        case 8:
            *(x.OUT8) = ((int)ACC59E3_OUTV2SC(Output_Voltage, Wiring_Mode)) << 16;
            break;

        default:
            return -1;
            break;
    }
    return 0;
}

int ACC59E3_COUT(struct ACC59E3 x, double Output_Current, unsigned int Output_Channel)

```

```

// This function will output the appropriate current values from the output channels (for
// voltage, use ACC59E3_VOUT())
// **Inputs:
// x: ACC59E3 structure being used for the present card
// Output_Current: Desired output current for this Current Output channel (mA)
// Output_Channel: The number of the output channel out of which the user desired to put voltage
// (indexed 1-8)
// **Outputs:
// Returns 0 if everything went well
// Returns -1 if incorrect Output_Channel was given
// **Notes: Will saturate output current values to 4-20 mA
{
    if(Output_Current > 24.0)
        Output_Current = 24.0; // mA
    else if(Output_Current < 4.0)
        Output_Current = 4.0; // mA

    switch (Output_Channel)
    {
        case 1:
            *(x.OUT1) = ((int)ACC59E3_OUTC2SC(Output_Current)) << 16;
            break;

        case 2:
            *(x.OUT2) = ((int)ACC59E3_OUTC2SC(Output_Current)) << 16;
            break;

        case 3:
            *(x.OUT3) = ((int)ACC59E3_OUTC2SC(Output_Current)) << 16;
            break;

        case 4:
            *(x.OUT4) = ((int)ACC59E3_OUTC2SC(Output_Current)) << 16;
            break;

        case 5:
            *(x.OUT5) = ((int)ACC59E3_OUTC2SC(Output_Current)) << 16;
            break;

        case 6:
            *(x.OUT6) = ((int)ACC59E3_OUTC2SC(Output_Current)) << 16;
            break;

        case 7:
            *(x.OUT7) = ((int)ACC59E3_OUTC2SC(Output_Current)) << 16;
            break;

        case 8:
            *(x.OUT8) = ((int)ACC59E3_OUTC2SC(Output_Current)) << 16;
            break;

        default:
            return -1;
            break;
    }
    return 0;
}

int ACC59E3_IN(struct ACC59E3 x, unsigned int Input_Channel)
// This function will return the ADC result on an ADC Channel of ACC59E3
// Works for both voltage and current ADCs, must convert to voltage or current later
// x: ACC59E3 structure being used for the present card
// Input_Channel: The ADC channel the user desires to read (indexed 1-16)
// Output:
// Returns the ADC result if there were no errors
// Returns 0 if ADC_Channel was input improperly
{
    switch (Input_Channel)
    {
        case 1:
            return ((int)(*(x.IN1))) >> 16;

```

```
        break;

    case 2:
        return ((int) (* (x.IN2))) >> 16;
        break;

    case 3:
        return ((int) (* (x.IN3))) >> 16;
        break;

    case 4:
        return ((int) (* (x.IN4))) >> 16;
        break;

    case 5:
        return ((int) (* (x.IN5))) >> 16;
        break;

    case 6:
        return ((int) (* (x.IN6))) >> 16;
        break;

    case 7:
        return ((int) (* (x.IN7))) >> 16;
        break;

    case 8:
        return ((int) (* (x.IN8))) >> 16;
        break;

    case 9:
        return ((int) (* (x.IN9))) >> 16;
        break;

    case 10:
        return ((int) (* (x.IN10))) >> 16;
        break;

    case 11:
        return ((int) (* (x.IN11))) >> 16;
        break;

    case 12:
        return ((int) (* (x.IN12))) >> 16;
        break;

    case 13:
        return ((int) (* (x.IN13))) >> 16;
        break;

    case 14:
        return ((int) (* (x.IN14))) >> 16;
        break;

    case 15:
        return ((int) (* (x.IN15))) >> 16;
        break;

    case 16:
        return ((int) (* (x.IN16))) >> 16;
        break;

    default:
        return 0;
        break;
}

double ACC59E3_SC2INV(int Software_Counts, unsigned int Resolution)
// Converts Software Counts from ADCs to Volts
```

```
// Input:
// Software_Counts: The ADC result in software counts from an ADC channel (e.g. from the
ACC59E3_IN() function).
// Resolution: The resolution of the ADCs: 16 for 16-bit ADCs, 12 for 12-bit ADCs
// Output:
// Voltage on the ADC channel corresponding to the Software_Counts input value
{
    if(Resolution == 16)
        return Software_Counts*SixteenBitVoltsPerSoftwareCount;
    else if (Resolution == 12)
        return Software_Counts*TwelveBitVoltsPerSoftwareCount;
}

double ACC59E3_SC2INC(int Software_Counts, unsigned int Resolution)
// Converts Software Counts from ADCs to Current Values
// Input:
// Software_Counts: The ADC result in software counts from an ADC channel (e.g. from the
ACC59E3_IN() function).
// Output:
// Current (mA) on the ADC channel corresponding to the Software_Counts input value
{
    if(Resolution == 16)
        return Software_Counts*SixteenBitmAPerSoftwareCount;
    else if (Resolution == 12)
        return Software_Counts*TwelveBitmAPerSoftwareCount;
}

int ACC59E3_INV2SC(int Voltage, unsigned int Resolution)
// Converts Volts to ADC Software Counts
// Input:
// Voltage: The input voltage the user desires to convert to software counts
// Resolution: The resolution of the ADCs: 16 for 16-bit ADCs, 12 for 12-bit ADCs
// Output:
// Voltage on the ADC channel corresponding to the Software_Counts input value
{
    if(Resolution == 16)
        return Voltage*SoftwareCountsPerVolt_16bit;
    else if (Resolution == 12)
        return Voltage*SoftwareCountsPerVolt_12bit;
}

int ACC59E3_INC2SC(int Current, unsigned int Resolution)
// Converts Current Values to ADC Software Counts
// Input:
// Current: The current value the user desires to convert to software counts
// Resolution: The resolution of the ADCs: 16 for 16-bit ADCs, 12 for 12-bit ADCs
// Output:
// Current (mA) on the ADC channel corresponding to the Software_Counts input value
{
    if(Resolution == 16)
        return Current*SoftwareCountsPermA_16bit;
    else if (Resolution == 12)
        return Current*SoftwareCountsPermA_12bit;
}

int ACC59E3_OUTV2SC(double Output_Voltage, unsigned int Wiring_Mode)
// Converts DAC Voltage Values to DAC Software Counts
// Input:
// Output_Voltage: Desired DAC Voltage Output Value (Volts)
// Wiring_Mode: 0 = Single-Ended Wiring
//              1 = Differential Wiring
// Output:
// Returns the software counts value (in software counts) corresponding to the desired DAC
voltage
{
    if(Wiring_Mode == SingleEndedWiring)
        return Output_Voltage*SoftwareCountsPerSingleEndedVolt;
    else
        return Output_Voltage*SoftwareCountsPerDifferentialVolt;
}
```

```

int ACC59E3_OUTC2SC(double Output_Current)
// Converts Current Output Values to DAC Software Counts
// Input:
// Output_Current: Desired Current Output Value (mA)
// Output:
// Returns the software counts value (in software counts) corresponding to the desired current
{
    return CurrentSF*Output_Current - 32768.0;
}

void ACC59E3_FilterInit(char CardNumber, unsigned int FSA[16])
// This function initializes the input filters of the ADCs of the ACC-59E3.
// Inputs:
// CardNumber: The card number for the ACC-59E3 whose input filters the user wants
// FSA: Filter Setup Array; a 16-element array with the input filter setting for each ADC channel
//      Each element 0 to 15 corresponds to ADC 1 to 16, respectively, and
//      should contain one of the following values:
//      0 = 24.5 kHz input filter cutoff frequency for the ADC channel
//      1 = 12.2 kHz input filter cutoff frequency for the ADC channel
//      2 = 4.3 kHz input filter cutoff frequency for the ADC channel
//      3 = 3.2 kHz input filter cutoff frequency for the ADC channel
// Output:
// Returns void, but the function will write to the ACC59E3[i].GpioData[0] structure
{
    unsigned int FilterWord=0;
    unsigned int index;
    char buff[64]="";
    for(index = 0; index < 16; index++)
    {
        FilterWord |= ACC59E3_FilterMask(index+1,FSA[index]);
    }

    sprintf(buff,"Gate3[%u].GpioData[0]",CardNumber);
    SetPmacVar(buff,(double)FilterWord);
    return;
}

int ACC59E3_FilterMask(unsigned int Input_Channel, unsigned int FilterType)
// Creates the mask for the filter word for each channel and user-specified filter type
// Note: The user should never call this function directly; the ACC59E3_FilterInit() function
// should be the only function calling this function
// Input:
// ADC_Channel: ADC Channel whose filter the user desires to configure
// FilterType: the integer value corresponding to the desired type of filter as follows:
//      0 = 24.5 kHz input filter cutoff frequency for the ADC channel
//      1 = 12.2 kHz input filter cutoff frequency for the ADC channel
//      2 = 4.3 kHz input filter cutoff frequency for the ADC channel
//      3 = 3.2 kHz input filter cutoff frequency for the ADC channel
// Output:
// Returns the filter word mask "FilterMask".
{
    unsigned int BitA=0, BitB=0;
    unsigned int FilterMask=0;
    switch(Input_Channel){
        case 1:
            BitA=3;
            BitB=19;
            break;

        case 2:
            BitA=1;
            BitB=17;
            break;
    }
}

```

```
case 3:
    BitA=2;
    BitB=18;
    break;

case 4:
    BitA=0;
    BitB=16;
    break;

case 5:
    BitA=7;
    BitB=23;
    break;

case 6:
    BitA=5;
    BitB=21;
    break;

case 7:
    BitA=6;
    BitB=22;
    break;

case 8:
    BitA=4;
    BitB=20;
    break;

case 9:
    BitA=11;
    BitB=27;
    break;

case 10:
    BitA=9;
    BitB=25;
    break;

case 11:
    BitA=10;
    BitB=26;
    break;

case 12:
    BitA=8;
    BitB=24;
    break;

case 13:
    BitA=15;
    BitB=31;
    break;

case 14:
    BitA=13;
    BitB=29;
    break;

case 15:
    BitA=14;
    BitB=30;
    break;

case 16:
    BitA=12;
    BitB=28;
    break;
```

```
        default:
            return -1; // Invalid Input_Channel value; must be between 1 and 16
            break;
    }

    switch(FilterType){
        case 0: // 24.5 kHz input filter
            return 0;
            break;

        case 1: // 12.2 kHz input filter
            FilterMask = 1 << BitA;
            break;

        case 2: // 4.3 kHz input filter
            FilterMask = 1 << BitB;
            break;

        case 3: // 3.2 kHz input filter
            FilterMask = 1 << BitA;
            FilterMask |= 1 << BitB;
            break;

        default:
            return 0;
            break;
    }

    return FilterMask;
}
```

After copying this header file, the user must simply write this line in his or her C source code file (of file extension type “*.c”) in the header section (above the main function):

```
#include "../../Libraries/acc59e3/acc59e3.h"
```

The user will then have access to the functions in that header file. The functions in this header file are described below.

ACC-59E3 Functions (Contained in acc59e3.h)

ACC59E3_PtrInit()

Function Declaration

```
struct ACC59E3 ACC59E3_PtrInit(struct ACC59E3 x, unsigned int CardNumberInit)
```

Purpose

This function initializes the ACC-59E3 card by assigning to the card its card number and then pointing the ADC and DAC pointers to their appropriate memory registers.

Input

- x: The ACC59E3 structure corresponding to the card the user wants to initialize
- CardNumberInit: The card number which the user is initializing. This is the same number as the "Power PMAC 'Gate3' Index #" from the address table in the Addressing ACC-59E3 section of this manual

Output

Returns the initialized ACC-59E3 structure.

ACC59E3_VOUT()

Function Declaration

```
int ACC59E3_VOUT(struct ACC59E3 x, double Output_Voltage, unsigned int Output_Channel, unsigned int Wiring_Mode)
```

Purpose

This function will output the appropriate voltage from the DACs. Note: For current output, use ACC59E3_COUT() instead.

Input

- x: ACC59E3 structure being used for the present card
- Output_Voltage: Desired output voltage for this DAC channel (volts)
- Output_Channel: The number of the DAC channel out of which the user desired to put voltage
- Wiring_Mode: 0 = Single-Ended Wiring
1 = Differential Wiring

Output

- Returns 0 if everything went well
- Returns -1 if incorrect Output_Channel was given



Note

The ACC59E3_VDAC() function will saturate output voltages at ± 10 V for single-ended wiring and ± 20 V for differential to prevent writing incorrect software count values to the DAC registers.

ACC59E3_COUT()

Function Declaration

```
int ACC59E3_COUT(struct ACC59E3 x, double Output_Current, unsigned int Output_Channel)
```

Purpose

This function will output the appropriate current values from the DACs. Note: for voltage outputs, use ACC59E3_VOUT() instead.

Input

- x: ACC-59E3 structure being used for the present card
- Output_Current: Desired output current for this DAC channel (mA)
- DAC_Channel: The number of the DAC channel out of which the user desired to put voltage

Output

- Returns 0 if everything went well
- Returns -1 if incorrect DAC_Channel was given



Note

The ACC59E3_COUT() function will saturate output currents to 4–20 mA to prevent writing incorrect software count values to the output registers.

ACC59E3_IN()

Function Declaration

```
int ACC59E3_IN(struct ACC59E3 x, unsigned int Input_Channel)
```

Purpose

This function will return the ADC result on an ADC Channel of ACC-59E3. This function works for both voltage and current inputs, but the user must convert to voltage or current later, using the functions ACC59E3_SC2INV() or ACC59E3_SC2INC(), respectively.

Input

- x: ACC59E3 structure being used for the present card
- Input_Channel: The input channel the user desires to read

Output

- Returns the ADC result if there were no errors
- Returns -400000 if ADC_Channel was input improperly

ACC59E3_SC2INV()

Function Declaration

```
double ACC59E3_SC2INV(int Software_Counts, unsigned int Resolution)
```

Purpose

Converts Software Counts from the ADCs to Volts.

Input

- Software_Counts: The ADC result in software counts from an ADC channel (e.g. from the ACC59E3_ADC() function).
- Resolution: The resolution of the ADCs: 16 for 16-bit ADCs, 12 for 12-bit ADCs.

Output

Voltage on the ADC channel corresponding to the Software_Counts input value.

ACC59E3_SC2INC()

Function Declaration

```
double ACC59E3_SC2INC(int Software_Counts, unsigned int Resolution)
```

Purpose

Converts Software Counts from the ADCs to Current Values [mA].

Input

- Software_Counts: The ADC result in software counts from an ADC channel.
- Resolution: The resolution of the ADCs: 16 for 16-bit ADCs, 12 for 12-bit ADCs.

Output

The current value [mA] on the ADC channel corresponding to the Software_Counts input value.

ACC59E3_INV2SC

Function Declaration

```
int ACC59E3_INV2SC(double Voltage, unsigned int Resolution)
```

Purpose

Converts Volts to ADC Software Counts.

Input

- Voltage: The input voltage the user desires to convert to software counts.
- Resolution: The resolution of the ADCs: 16 for 16-bit ADCs, 12 for 12-bit ADCs.

Output

Voltage on the ADC channel corresponding to the Software_Counts input value.

ACC59E3_INC2SC()

Function Declaration

```
int ACC59E3_INC2SC(double Current, unsigned int Resolution)
```

Purpose

Converts Current Values to ADC Software Counts.

Input

- Current: The input current value [mA] the user desires to convert to software counts.
- Resolution: The resolution of the ADCs: 16 for 16-bit ADCs, 12 for 12-bit ADCs.

Output

Software counts on the ADC channel corresponding to the current input value [mA].

ACC59E3_OUTV2SC()

Function Declaration

```
int ACC59E3_OUTV2SC(double Output_Voltage, unsigned int Wiring_Mode)
```

Purpose

Converts DAC Voltage Values to DAC Software Counts

Input

- Output_Voltage: Desired DAC Voltage Output Value (Volts)
- Wiring_Mode: 0 = Single-Ended Wiring
1 = Differential Wiring

Output

Returns the software counts value (in software counts) corresponding to the desired DAC Voltage.

ACC59E3_OUTC2SC()

Function Declaration

```
int ACC59E3_OUTC2SC(double Output_Current)
```

Purpose

Converts DAC Current Output Values to DAC Software Counts.

Input

Output_Current: Desired Current Output Value [mA].

Output

Returns the software counts value (in software counts) corresponding to the desired DAC Current.

ACC59E3_FilterInit()

Function Declaration

```
void ACC59E3_FilterInit(char CardNumber, unsigned int FSA[16])
```

Purpose

This function initializes the input filters of the ADCs of the ACC-59E3.

Input

- CardNumber: The card number for the ACC-59E3 whose input filters the user wants
- FSA: Filter Setup Array; a 16-element array with the input filter setting for each ADC channel. Each element 0 to 15 corresponds to input channel 1 to 16, respectively, and should contain one of the following values:
 - 0 = 24.5 kHz input filter cutoff frequency for the ADC channel corresponding to this array element
 - 1 = 12.2 kHz input filter cutoff frequency for the ADC channel corresponding to this array element
 - 2 = 4.3 kHz input filter cutoff frequency for the ADC channel corresponding to this array element
 - 3 = 3.2 kHz input filter cutoff frequency for the ADC channel corresponding to this array element

Output

Returns void, but the function will write to the Acc59E3[i].GpioData[0] structure to configure the card's ADC input filters.

ACC59E3_FilterMask()

Function Declaration

```
int ACC59E3_FilterMask(unsigned int ADC_Channel, unsigned int FilterType)
```

Purpose

Creates the mask for the filter word for each channel and user-specified filter type.



Note

The user should never call this function directly; the ACC59E3_FilterInit() function should be the only function calling this function.

Input

- ADC_Channel: ADC Channel whose filter the user desires to configure
- FilterType: the integer value corresponding to the desired type of filter as follows:
 - 0 = 24.5 kHz input filter cutoff frequency for the ADC channel corresponding to this array element
 - 1 = 12.2 kHz input filter cutoff frequency for the ADC channel corresponding to this array element
 - 2 = 4.3 kHz input filter cutoff frequency for the ADC channel corresponding to this array element
 - 3 = 3.2 kHz input filter cutoff frequency for the ADC channel corresponding to this array element

Output

Returns the filter word mask "FilterMask".

Examples

Example: Converting ADC Results to Voltages and then Copying to P-Variables Using a BGCPLC

This example will obtain the ADC results for all 16 channels of an ACC-59E3 addressed as Card 0 (For one ACC-59E3 as Card 0 (Power PMAC I/O Base Address Offset=\$900000) with 16-bit ADCs, converts to voltages [Volts], and then stores the results in P2000–P2015 for ADC1–ADC16, respectively.

```
#include <gplib.h>
#include <stdio.h>
#include <dlfcn.h>

#include "../Include/pp_proj.h"
#include "../Libraries/acc59e3/acc59e3.h"

void user_plcc()
{
    unsigned int Channel_Number;
    ACC59E3 ACC59E3_0;    // Allocate a structure for the ACC-59E3
    ACC59E3_0=ACC59E3_PtrInit(Card0); // Initialize the card's pointers

    for(Channel_Number=1;Channel_Number<=16;Channel_Number++)
    { // Loop through all 16 inputs
        pshm->P[2000+Channel_Number-1]=(double)ACC59E3_IN(ACC59E3_0,Channel_Number);
    }
    return;
}
```

Example: Converting ADC Results to Current and then Copying to P-Variables Using a BGCPLC

This example will obtain the ADC results for all 16 channels of an ACC-59E3 addressed as Card 0 (For one ACC-59E3 as Card 0 (Power PMAC I/O Base Address Offset=\$900000) with 16-bit ADCs, converts to current [mA], and then stores the results in P3000–P3015 for Current Inputs IN1–IN16, respectively.

```
#include <gplib.h>
#include <stdio.h>
#include <dlfcn.h>

#include "../Include/pp_proj.h"
#include "../Libraries/acc59e3/acc59e3.h"

#define SixteenBitResolution 16

void user_plcc()
{
    struct ACC59E3 ACC59E3_0;
    unsigned int Channel_Number;
    int ADC_Result;

    ACC59E3_0=ACC59E3_PtrInit(Card0);
    for(Channel_Number=1; Channel_Number<=16; Channel_Number++)
    {
        ADC_Result=ACC59E3_IN(ACC59E3_0,Channel_Number);
        pshm->P[3000+Channel_Number-1]=ACC59E3_SC2INC(ADC_Result,SixteenBitResolution);
    }

    return;
}
```

Example: Accessing Predefined ADC M-Variables Using a BGCPLC

For one ACC-59E3 as Card 0 (Power PMAC I/O Base Address Offset=\$900000). Assumes the user has already downloaded the example M-Variables suggested for Card 0 earlier into his or her “global definitions.pmh” file. This example reads the ADC results to which the suggested M-Variables point and then stores the results in P4000 through P4015. Note: This example actually does not need to use “acc59e3.h”.

```
#include <gplib.h>
#include <stdio.h>
#include <dlfcn.h>

#include "../..../Include/pp_proj.h"

void user_plcc()
{
    unsigned int index; // Define index counter
    for(index=0; index<16; index++) // Iterate through all 16 input channels
    {
        // Copy all 16 ADCs to P4000 through P4015
        // P4000 contains IN1's result
        // P4001 contains IN2's result
        // ...
        // P4015 contains IN16's result
        pshm->P[4000+index]=GetPtrVar(ACC59E3_0_IN1 + index);
    }

    return;
}
```

Example: DAC Voltage Output Using C with Card 0 Using a BGCPLC

For a single ACC-59E3 addressed as Card 0 (Card 0 is at Power PMAC I/O Base Address Offset = \$900000). The program takes voltage commands stored in P5000–P5007 for DAC1–DAC8, respectively, and converts to software counts appropriate for Single-Ended Output Wiring, and then commands the output on the DACs.

```
#include <gplib.h>
#include <stdio.h>
#include <dlfcn.h>

#include "../..../Include/pp_proj.h"
#include "../..../Libraries/acc59e3/acc59e3.h"

void user_plcc()
{
    struct ACC59E3 ACC59E3_0; // Allocate ACC-59E3 structure
    int flag; // Allocate storage variable for ACC59E3_VOUT's error code
    unsigned int Output_Channel; // Variable to select output channel
    double Output_Voltage; // Variable to store voltage value to output

    ACC59E3_0=ACC59E3_PtrInit(Card0); // Initialize the card's ADC/DAC pointers

    for(Output_Channel=1; Output_Channel <= 8; Output_Channel++)
    {
        Output_Voltage=pshm->P[5000+Output_Channel-1]; // Look in P5000-P5007 for voltage values

        // Command voltage to the desired DAC channel
        flag=ACC59E3_VOUT(ACC59E3_0, Output_Voltage, Output_Channel, SingleEndedWiring);

        // Store flag in P-Variable (P10000) for external monitoring
        pshm->P[10000]=(double) flag;
    }

    return;
}
```

Example: DAC Current Output Using C with Card 1 Using a BGCPLC

This example defines a structure for ACC59E3 and then points to Card 1's DAC registers (Card 1 is at Power PMAC I/O Base Address Offset = \$904000). The program takes voltage commands stored in P6000–P6007 for OUT1–OUT8, respectively and converts to software counts appropriate for Single-Ended Output Wiring, and then commands the output on the DACs.

```
#include <gplib.h>
#include <stdio.h>
#include <dlfcn.h>

#include "../Include/pp_proj.h"
#include "../Libraries/acc59e3/acc59e3.h"

void user_plcc()
{
    struct ACC59E3 ACC59E3_1;    // Allocate ACC-59E3 structure
    unsigned int index;
    ACC59E3_1=ACC59E3_PtrInit(Card1);    // Initialize card's pointers

    for(index=0; index<8; index++)
    {
        // Cycle through all outputs
        ACC59E3_COUT(ACC59E3_1, pshm->P[6000+index], index+1);
    }
    return;
}
```

Example: Configuring the Input Filters for an ACC-59E3 Addressed as Card 0

This example, written as a Background C Application (from within the IDE Project Manager, go to C Language→Background Programs to create a Background C Application) configures the following ADCs to have the following input filter cutoff frequencies:

ADCs	Input Filter Cutoff Frequency [kHz]
1–4	24.5
5–8	12.2
9–12	4.3
13–16	3.2

This application runs once when called and then stops.

```
#include <gplib.h>    // Global Gp Shared memory pointer

#include "../Include/pp_proj.h"
#include "../Libraries/acc59e3/acc59e3.h"

int main(void)
{
    unsigned int FSA[]=    {0,0,0,0,    // Configure ADC1-ADC4 as 24.5 kHz input cutoff freq.
                           1,1,1,1,    // Configure ADC5-ADC8 as 12.2 kHz input cutoff freq.
                           2,2,2,2,    // Configure ADC9-ADC12 as 4.3 kHz input cutoff freq.
                           3,3,3,3};   // Configure ADC13-ADC16 as 3.2 kHz input cutoff freq.

    InitLibrary();
    ACC59E3_FilterInit(Card0, FSA);
    CloseLibrary();
    return 0;
}
```

OMRON Corporation Industrial Automation Company

Kyoto, JAPAN

Contact : www.ia.omron.com

Regional Headquarters

OMRON EUROPE B.V.
Wegalaan 67-69, 2132 JD Hoofddorp
The Netherlands
Tel: (31) 2356-81-300 Fax: (31) 2356-81-388

OMRON ASIA PACIFIC PTE. LTD.
438B Alexandra Road, #08-01/02 Alexandra
Technopark, Singapore 119968
Tel: (65) 6835-3011 Fax: (65) 6835-3011

OMRON ELECTRONICS LLC
2895 Greenspoint Parkway, Suite 200
Hoffman Estates, IL 60169 U.S.A.
Tel: (1) 847-843-7900 Fax: (1) 847-843-7787

OMRON (CHINA) CO., LTD.
Room 2211, Bank of China Tower,
200 Yin Cheng Zhong Road,
PuDong New Area, Shanghai, 200120, China
Tel: (86) 21-6023-0333 Fax: (86) 21-5037-2388

Authorized Distributor:

©OMRON Corporation 2025 All Rights Reserved.
In the interest of product improvement,
specifications are subject to change without notice.

Cat. No. O070-E-01 Rev C

0925 (0325)