IP-UL-ADC40

40 Channel Unilin™
ADC IndustryPack®

User's Manual

Manual Revision: v Hardware Revision: A

IP-UL-ADC40
Unilin 40 Channel
ADC IndustryPack

SBS GreenSpring Modular I/O
181 Constitution Drive
Menlo Park, CA 94025
(650) 327-1200
(650) 327-3808 FAX
http://www.greenspring.com

Table of Contents

List of Figures

Product Description

IP-UL-ADC40 is a member of SBS GreenSpring's Unilin™ family of analog IndustryPacks. It has forty input channels served by five ADCs, each with 12 bits of resolution and a built-in multiplexer. Five channels can be converted simultaneously every 10 µs, giving a total throughput of 500,000 samples per second. The ADCs can run continuously or under software, timer, or external control. Analog input ranges are 0-5 V, 0-10 V, ±5 V, and ±10 V, independently selectable in four groups of channels.

The analog converters operate in one of two modes: Continuous or Single-Shot. In Continuous mode, enabled converters are internally retriggered by the completion of each conversion and run at the maximum rate. In Single-Shot mode, each enabled ADC performs one conversion upon receipt of a trigger. Sources for the trigger include a software command, on-board timers, and external strobe signals.

Two additional modes, All Channels and Channel Subset are also available. In All Channels mode, on-board logic automatically controls the converters and multiplexers to convert all forty channels as fast as possible. Channel Subset mode allows software control over multiplexers and individual enabling of converters. Either mode can be used with Continuous or Single-Shot modes.

A programmable timer is provided. It has 16 bits of resolution and a prescaler that provides delays from 1 µs to 65.5s. The timer may be driven from any one of four internal and external clock sources. The timer can be used to provide interrupts, send strobe pulses to external pins, or trigger ADC conversions.

Software can be synchronized with external events and the operation of the ADCs by means of interrupts or by polling. The ADCs can provide synchronization signals for the timer, and for external hardware.

Data from the ADCs is stored by the converters in 40 16-bit words of dual-ported IndustryPack RAM (DPRAM) as sign-extended signed or zero-filled unsigned numbers, depending on the range selected. Software may read ADC data from the DPRAM asynchronously and in any order.

Interrupt sources include the timer, external strobes, and a choice of three conversion-related clock signals. Separate, prioritized interrupt handling is provided for the independent parts of the IndustryPack.

Two external strobes, one on the IndustryPack bus connector, and one on the I/O bus connector may be used as inputs or outputs for timing signals and as inputs for external clocks.

An on-board 16 Kbit EEPROM stores calibration data for each channel and each range. This memory is non-volatile and user-programmable.

A single shunt selects between IP bus or off-board power supplies. Each converter and the precision on-board reference has its own regulator to minimize noise.

Quick Start

This section is for people who don't read manuals. It describes how to get a new board out of the box and working in a system as quickly as possible. In addition to the IndustryPack, an IndustryPack carrier and a CPU are required. Access to the I/O connector of the IndustryPack is accomplished with a 50-way ribbon cable and a terminal block, supplied as part of the Engineering Kit for this product.

Set up the hardware

Without changing any shunts on the IndustryPack, plug it into a slot on the carrier board. Make sure that the IndustryPack bus is set to work at 8 MHz. Connect a 50-way ribbon between the terminal block and the carrier connector. Connect voltages to the ADC input pins. Full scale input defaults to ±10 V. The table below shows the relationship between the ADC channels and the I/O connector pins:

Signal	Pin
ADC1	1
ADC2	2
ADC3	3
ADC4	4
ADC5	5
ADC6	6
ADC7	7
ADC8	8
ADC9	9
ADC10	10
ADC11	11

Signal	Pin
ADC12	12
ADC13	13
ADC14	14
ADC15	16
ADC16	16
ADC17	17
ADC18	18
ADC19	19
ADC20	20
ADC21	21
ADC22	22

Signal	Pin
ADC23	23
ADC24	24
ADC25	25
ADC26	26
ADC27	27
ADC28	28
ADC29	29
ADC30	30
ADC31	31
ADC32	32
ADC33	33

Signal	Pin
ADC34	34
ADC35	35
ADC36	36
ADC37	37
ADC38	38
ADC39	39
ADC40	40
AGND	41
AGND	48

Figure out the memory map

The memory map of the carrier as it is configured must be known in order to have access to the IndustryPack registers and the ADC data values. Work out the base addresses of the I/O and memory areas before going to the next step. To have access to the data values in DPRAM, the memory space must be aligned on a 2 Kbyte boundary.

Turn on the system

Turn on the system. Now check that the IndustryPack is present as expected by reading some of the registers and the memory. The IndustryPack accepts only 16-bit word accesses. Byte accesses will cause bus traps and longword accesses may not be performed as expected on some carriers (notably the MVME162), so avoid these.

Read the ADC channels

The ADCs convert all the channels repeatedly soon as the board is powered up. Using a debugger, read 16-bit words from the start of the memory space. These are the results of the analog conversions as signed-extended 12-bit numbers. The table below shows the offsets to each channel in the memory space:

Signal	Data
ADC1	002
ADC2	006
ADC3	00a
ADC4	00e
ADC5	012
ADC6	016
ADC7	01a
ADC8	01e
ADC9	022
ADC10	026

Signal	Data
ADC11	02a
ADC12	02e
ADC13	032
ADC14	036
ADC15	03a
ADC16	03e
ADC17	042
ADC18	046
ADC19	04a
ADC20	04e

Signal	Data
ADC21	052
ADC22	056
ADC23	05a
ADC24	05e
ADC25	062
ADC26	066
ADC27	06a
ADC28	06e
ADC29	072
ADC30	076

Signal	Data
ADC31	07a
ADC32	07e
ADC33	082
ADC34	086
ADC35	08a
ADC36	08e
ADC37	092
ADC38	096
ADC39	09a
ADC40	09e

Since the converters are 12-bit devices, each value ranges from -2048 to 2047, corresponding to input voltages from -10 V to 10 V.

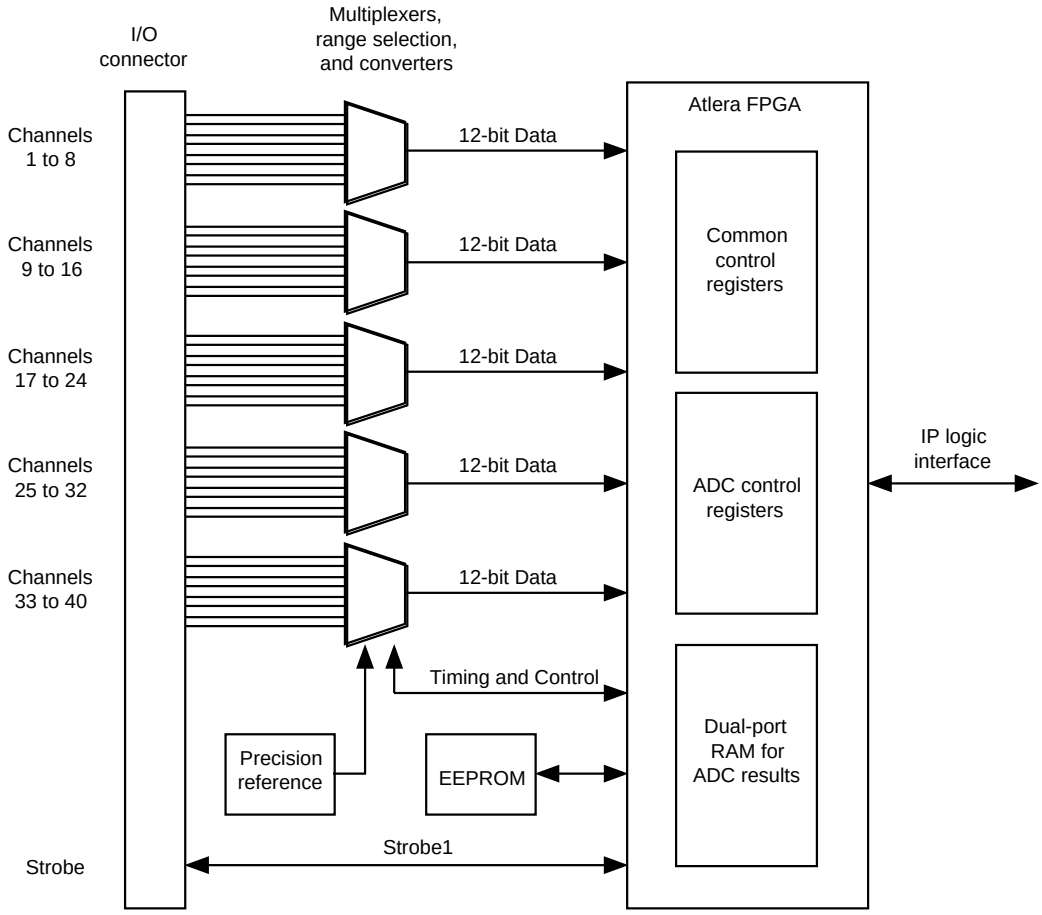

Figure 1 IP-UL-ADC40 block diagram

Programming Overview

From a programmer's perspective, IP-UL-ADC40 contains two sections: Common and ADC. Each block contains registers that control the parts of the IndustryPack associated with that section. Each section has its own interrupt circuitry and is almost completely independent of the other.

All registers are implemented in the I/O space. The memory space contains dual-ported RAM used to retrieve data values from the ADCs.

All registers and memory are word (16-bit) accessible only. None are accessible as bytes: bus traps will result. Long word accesses will work on some carriers, but notably the MVME162 fails in this respect, giving garbage in the lower word of the I/O space. Memory space may be accessed as long words on the MVME162.

Each IndustryPack in the Unilin family is designed to be as software-compatible with others as possible. For this reason IndustryPacks that implement only a few of the family's features have sparse register maps, with all other bits set to zero. The whole 128 byte I/O space and all of the memory space acknowledge accesses.

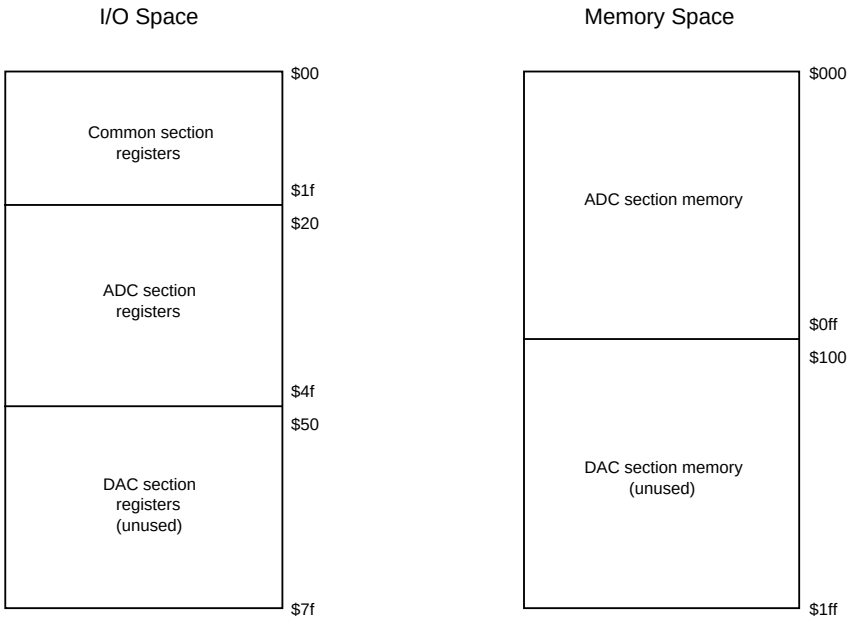

Figure 2 I/O and Memory Space Layout

Common Section

The common section occupies the first 32 bytes in I/O space. Its registers control communication with the EEPROM, give software access to the two external strobe signals, handle hardware errors, and store the interrupt vector.

ADC Section

The ADC section occupies the next 48 bytes in I/O space and the first 256 bytes of memory space. Its registers control the operation of the ADCs, the ADC interrupts, the ADC trigger, external strobe functions, and the timer. The CPU reads all ADC data from the memory space.

DAC Section

This IndustryPack has no DAC section. Its registers and memory space are unused and hard-wired to zero.

Programmable Timer

A programmable timer is provided. It resides in the ADC section. It has multiple clock sources, a prescaler, and a 16-bit down-counter. It can either repeat or perform single delays. The outputs from the timer can be used to trigger conversions, cause interrupts, and pulse external strobes.

Triggers

A trigger is provided in the ADC section. In Single-Shot mode the only way to start conversions is by asserting the trigger. In Continuous mode, the trigger can be used for other purposes. The trigger can be asserted by the external strobes, by software commands, or by the timer.

Interrupts

Eight sources of interrupt are provided, handled separately by a pair of registers in each of the two sections. The vector is modified to indicate the section with the pending interrupt.

Register Map

The table below shows the 128 byte memory map for the I/O space of IP-UL-ADC40. All locations are accessible as 16-bit words only. All unused locations read zero and ignore writes. For future compatibility, unused locations should never be written. The offsets shown are those for VME, ISA, and PCI bus carriers.

Offset	Section	Abbr.	Name	Access
00	Common	CGCR	Common General Control Register	R/W
02		CGSR	Common General Status Register	R/W
04		CICR	Common Interrupt Control Register	R/W
06		CISR	Common Interrupt Status Register	R/W
0a		CIVR	Common Interrupt Vector Register	R/W
10		CCCSR	Common Configuration Control and Status Register	R/W
12		CCDR	Common Configuration Data Register	R/W
1c		CTAR	Common Test Address Register	R/W
1e		CTDR	Common Test Data Register	R/W
20	ADC	AGCR	ADC General Control Register	R/W
22		AGSR	ADC General Status Register	R
24		AICR	ADC Interrupt Control Register	R/W
26		AISR	ADC Interrupt Status Register	R/W
2a		ACCRL	ADC Conversion Control Register Low	R/W
2c		ARSR	ADC Range Select Register	R/W
30		ATSR	ADC Trigger Source Register	R/W
3c		ATCSR	ADC Timer Control and Status Register	R/W
3e		ATPR	ADC Timer Period Register	R/W
4c		AMSR	ADC MUX Select Register	R/W

Figure 3 Register Space Layout

ID Space

The table below shows the 128 byte ID space of IP-UL-ADC40. All locations should be accessed as 16-bit words only. All unused locations should not be accessed. All writes are ignored The offsets shown are those for VME, ISA, and PCI bus carriers.

Offset	Value	Name
00	0049	'I'
02	0050	'P'
04	0041	'A'
06	0048	'H'
08	00f0	Manufacturer (GreenSpring)
0a	00ad	Model number
0c	00a1	Revision
0e	0000	(reserved)
10	0000	Driver ID Low
12	0000	Driver ID High
14	000c	Byte count
16	0067	CRC

Figure 4 ID Space Layout

Memory Space

The table below shows the 512 byte memory space of IP-UL-ADC40. All locations should be accessed as 16-bit words only. All unused locations read zero and ignore writes. For future compatibility unused locations should never be written The offsets shown are those for VME, ISA, and PCI bus carriers. The memory space repeats every 2 Kbytes.

Offset	Abbr.	Name	Access
002	ADC1L	ADC channel 1 Low Data	R/W
006	ADC2L	ADC channel 2 Low Data	R/W
00a	ADC3L	ADC channel 3 Low Data	R/W
00e	ADC4L	ADC channel 4 Low Data	R/W
012	ADC5L	ADC channel 5 Low Data	R/W
016	ADC6L	ADC channel 6 Low Data	R/W
01a	ADC7L	ADC channel 7 Low Data	R/W
01e	ADC8L	ADC channel 8 Low Data	R/W
022	ADC9L	ADC channel 9 Low Data	R/W
026	ADC10L	ADC channel 10 Low Data	R/W
02a	ADC11L	ADC channel 11 Low Data	R/W
02e	ADC12L	ADC channel 12 Low Data	R/W
032	ADC13L	ADC channel 13 Low Data	R/W
036	ADC14L	ADC channel 14 Low Data	R/W
03a	ADC15L	ADC channel 15 Low Data	R/W
03e	ADC16L	ADC channel 16 Low Data	R/W
042	ADC17L	ADC channel 17 Low Data	R/W
046	ADC18L	ADC channel 18 Low Data	R/W
04a	ADC19L	ADC channel 19 Low Data	R/W
04e	ADC20L	ADC channel 20 Low Data	R/W
052	ADC21L	ADC channel 21 Low Data	R/W
056	ADC22L	ADC channel 22 Low Data	R/W
05a	ADC23L	ADC channel 23 Low Data	R/W
05e	ADC24L	ADC channel 24 Low Data	R/W
062	ADC25L	ADC channel 25 Low Data	R/W
066	ADC26L	ADC channel 26 Low Data	R/W
06a	ADC27L	ADC channel 27 Low Data	R/W
06e	ADC28L	ADC channel 28 Low Data	R/W
072	ADC29L	ADC channel 29 Low Data	R/W
076	ADC30L	ADC channel 30 Low Data	R/W
07a	ADC31L	ADC channel 31 Low Data	R/W
07e	ADC32L	ADC channel 32 Low Data	R/W
082	ADC33L	ADC channel 33 Low Data	R/W
086	ADC34L	ADC channel 34 Low Data	R/W
08a	ADC35L	ADC channel 35 Low Data	R/W
08e	ADC36L	ADC channel 36 Low Data	R/W
092	ADC37L	ADC channel 37 Low Data	R/W
096	ADC38L	ADC channel 38 Low Data	R/W
09a	ADC39L	ADC channel 39 Low Data	R/W
09e	ADC40L	ADC channel 40 Low Data	R/W

Figure 5 Memory Space Layout

Analog Input

The Quick Start section gave a very brief description of how to operate the ADC converters on the IP-UL-ADC40. Many applications demand so little from an ADC that the QuickStart is sufficient. Applications with critical throughput, timing, or synchronization requirements need much greater control over handling data and taking samples. This section describes how to control ADC conversions, how to synchronize them with software and hardware, and how to convert between volts and converter counts.

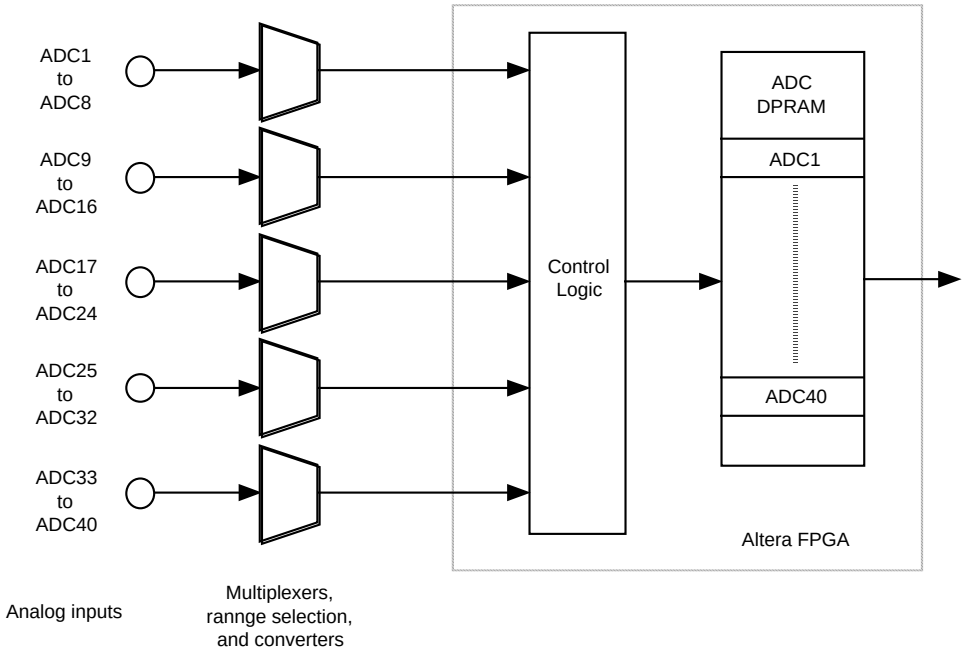

Figure 6 ADC Section Data Flow

Range Selection

Each analog input channel can be selected to operate in one of four ranges: 0-5 V, 0-10 V, ±5 V, or ±10 V. The forty input channels are arranged into four range select groups, as shown below. All channels within a group are set to the same range at once, but the four groups are independent.

Range Select Group	Channels	Ranges Available
RS1	1, 2, 3, 4, 5, 6, 7, 8	0-5 V, 0-10 V, ±5 V, ±10 V
RS2	9, 10, 11, 12, 13, 14, 15, 16	0-5 V, 0-10 V, ±5 V, ±10 V
RS3	17, 18, 19, 20, 21, 22, 23, 24	0-5 V, 0-10 V, ±5 V, ±10 V
RS4	25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40	0-5 V, 0-10 V, ±5 V, ±10 V

Figure 7 Input Range Select Groups

Range selection is carried out through the ADC Range Select Register (+2c). The default range is ±10 V for all groups, as shown by the reset bit pattern.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
RS4				RS3				RS2				RS1				Read
RS4				RS3				RS2				RS1				Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Figure 8 Range Selection in the ADC Range Select Register

The ranges are selected as follows:

RS1, 2, 3, 4	Range
0000	±10 V (default)
0001	0-10 V
0010	±5 V
0011	0-5 V

ADC Operating Modes

The ADC section of IP-UL-ADC40 operates in one of two modes: Continuous or Single-Shot.

In Continuous mode (the power-up default), conversions are carried out repeatedly. All ADC data is written to the DPRAM as it is generated. This mode is convenient, but uncontrolled: samples are not synchronized to any particular source, nor can the sampling rate be changed from 100 kHz.

In Single-Shot mode, conversions are commanded by the assertion of the ADC trigger. The trigger can be asserted by software, by a timer, or by an external strobe. The timing of conversions can be controlled precisely in Single-Shot mode, though more programming is necessary to obtain the desired result. All ADC data is written into the DPRAM, as before.

The mode is controlled by the SS bit in the AGCR (+20). Asserting the bit selects Single-Shot mode. Negating it selects Continuous mode, immediately starting the first conversion.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	STS2		STS1		0	0	0	0	0	CS	**SS**	IRQE	Read
X	X	X	X	STS2		STS1		X	X	X	X	X	CS	**SS**	IRQE	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	Reset

Figure 9 Single Shot Mode Bit in the ADC General Control Register

The following code example puts the ADC section into Single-Shot mode and programs the ADC timer to cause a trigger once every 120 µs:

```
UINT16 *agcr = (UINT16*)0xfff58020;   /* MVME162 slot A */
UINT16 *atcsr = (UINT16*)0xc000003c;
UINT16 *atpr = (UINT16*)0xc000003e;
UINT16 *atsr = (UINT16*)0xc0000030;

*agcr = 0x0002;          /* Select ADC Single Shot mode */
*atsr = 0x6021;          /* Select ADC timer as trigger input */
*atpr = 119;             /* Set timer period to 120 us */
*atcsr = 0x0023;         /* Select 1us repeating and clear timer */
```

A section operating in Continuous mode can still use the trigger to perform useful functions, since triggers can cause pulses on the strobe lines, reload the timer, and cause interrupts.

The choice of operating mode depends on the circumstances. Continuous mode is provided for convenience and speed. Single-Shot mode is provided for flexibility and control. The sections of this manual that describe the timer and the trigger contain more information on making conversions occur in Single-Shot mode.

Conversion Control

Two additional modes allow software control of the converters and the multiplexers: all Channels mode (the default) and Channel Subset mode. They can be selected in either Continuous or Single-Shot modes.

The mode is controlled by the CS bit in the AGCR (+20). Asserting the bit selects Channel Subset mode. Negating it selects All Channels mode.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	STS2		STS1		0	0	0	0	0	**CS**	SS	IRQE	Read
X	X	X	X	STS2		STS1		X	X	X	X	X	**CS**	SS	IRQE	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	Reset

Figure 10 Channel Subset Mode Bit in the ADC General Control Register

In All Channels mode all the ADC channels are always converted. This is useful, but not always convenient, since the ADC multiplexers are swept across their eight input channels and conversions performed whether the application needs the data or not. This limits the conversion rate of any one channel to 12.5 kHz.

In Channel Subset mode, each of the five ADC converters is controlled by a bit in the ACCRL (+2a). If the bit is asserted, the converter is enabled, and data flows into the DPRAM. If it is negated, the converter is disabled, and data does not flow. The ACCRL is ignored in All Channels mode.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	0	0	0	**C5**	**C4**	**C3**	**C2**	**C1**	Read
X	X	X	X	X	X	X	X	X	X	X	**C5**	**C4**	**C3**	**C2**	**C1**	Write
0	0	0	0	0	0	0	0	0	0	0	**1**	**1**	**1**	**1**	**1**	Reset

Figure 11 Converter Enable Bits in the ADC Converter Control Register Low

Multiplexers

Channel Subset mode also gives software control over the ADC multiplexers, whereas All Channels mode sweeps automatically as it converts. The AMSR (+4c) controls the multiplexer selection in this mode:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	CHANS							0	0	0	0	0	**MUX SELECT**			Read
X	X	X	X	X	X	X	X	X	X	X	X	X	**MUX SELECT**			Write
0	0	1	0	1	0	0	0	0	0	0	0	0	**0**	**0**	**0**	Reset

Figure 12 Multiplexer Bits in the ADC MUX Control Register

Changing the multiplexer selection changes the selection for all five ADC converters. Since each converter can be enabled or disabled individually, the AMSR and the ACCRL can select up to five converters in each set. The table below shows the relationship between converters, multiplexer selections, and the analog channels:

Channels Selected	Converters Enabled				
MUX Selection	C1	C2	C3	C4	C5
000	1	9	17	25	33
001	2	10	18	26	34
010	3	11	19	27	35
011	4	12	20	28	36
100	5	13	21	29	37
101	6	14	22	30	38
110	7	15	23	31	39
111	8	16	24	32	40

Figure 13 Channels Selected for Different Multiplexer and Converter Selections

Note that clearing all bits in the ACCRL while in Channel Subset mode stops all the converters from operating. In this state, the DA and SU signals for the ADC section are **not** generated.

Conversion Cycles

A conversion cycle is a complete set of conversions carried out by hardware in one indivisible operation. Depending on the mode and the ACCRL settings, this can comprise conversion of between one and forty channels:

Modes	Conversion Cycle
Continuous, All Channels (default)	One per conversion of all 40 channels. Repeats automatically.
Single-Shot, All Channels	One conversion of all 40 channels. Started by a trigger.
Continuous, Channel Subset	One per conversion of between 1 and 5 channels as set by the ACCRL. Repeats automatically.
Single-Shot, Channel Subset	One conversion of between 1 and 5 channels as set by the ACCRL. Started by a trigger.

Figure 14 ADC Section Modes and Conversion Cycle Summary

Conversion cycles determine the location of the DA and SU synchronization signals described in the next section.

Software and Hardware Synchronization

The Data Access (DA) signal in the ADC General Status Register (+22) is negated just before the first ADC conversions of a conversion cycle are written to the DPRAM and asserted just after the last is written. It can be used by software to synchronize DPRAM reads with the ADC conversions to ensure that all data values correspond to a single sample.

The Sample Update (SU) signal is pulsed once per conversion cycle. It is asserted as the first ADCs sample their inputs and negated just after the last sample their inputs. This signal is useful for synchronizing software, the timer, and conversions to external hardware. The ADC SU signal can be output on either external Strobe pin.

The sections below describe the exact timing of these signals for the four operating modes and how they can be used by software.

Note that clearing all bits in the ACCRL while in Channel Subset mode stops all the converters from operating. In this state, the DA and SU signals for the ADC section are **not** generated.

All Channels Mode

The diagram below shows the relationship between DA and SU when the ADC section is operating in Continuous, All Channels Mode:

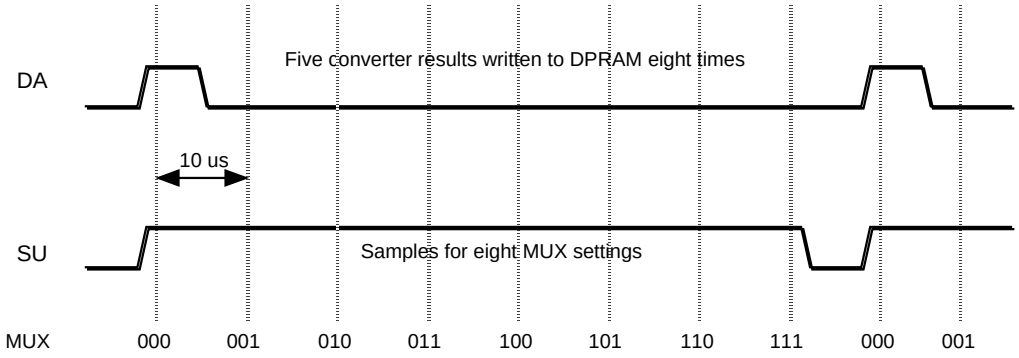

Figure 15 DA and SU Timing for ADCs in Continuous, All Channels Mode

The raw DA and SU signals in the AGSR are very brief, and will be missed most of the time:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	DPR	0	WIDTH					**DA**	**SU**	0	0	0	0	TS	IRQP	Read
X	X	X	X	X	X	X	X	**X**	**X**	X	X	X	X	X	X	Write
0	1	0	0	1	1	0	0	**DA**	**SU**	0	0	0	0	TS	0	Reset

Figure 16 DA, and SU Bits in the ADC General Status Register

Two other versions of the same signal are available in the ADC IRQ Status Register (+26).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
DDA	**DSU**	0	0	0	DTIM	DTRG	0	**PDA**	**PSU**	0	0	0	PTIM	PTRG	0	Read
X	**X**	X	X	X	X	X	X	**PDA**	**PSU**	X	X	X	PTIM	PTRG	X	Write
0	**0**	0	0	0	0	0	0	**0**	**0**	0	0	0	0	0	0	Reset

Figure 17 DA and SU Bits in the ADC IRQ Status Register

The DDA signal is a version of DA that has been passed through a rising edge detector. It pulses only very briefly and may never be visible to the CPU. The PDA signal is a latched version of DDA. A rising edge on DA will set this bit. It can be cleared by writing a one to the PDA bit. Although these bits are part of the interrupt system, they play no part in interrupt generation if the interrupts are disabled, as is the default. The DSU and PSU bits work in the same way.

This code fragment uses the PDA signal to detect the rising edge of DA:

```
UINT16 *aisr = (UINT16*)0xfff58026;   /* MVME162 slot A */
UINT16 reg;

reg = 0x80;               /* Clear the pending bit */
*aisr = reg;
do                        /* Wait for it to be set */
   reg = *aisr;
while((reg&0x80)==0);

/* Do something */
```

In Single-Shot, All Channels mode, the DA and SU signals have the same meaning as they do in Continuous, All Channels mode. Instead of the internally-generated repetition of Continuous mode, each conversion is started by the assertion of the ADC trigger. A single SU pulse coincides with the inputs being sampled and, a single DA pulse with the data being transferred to the DPRAM:

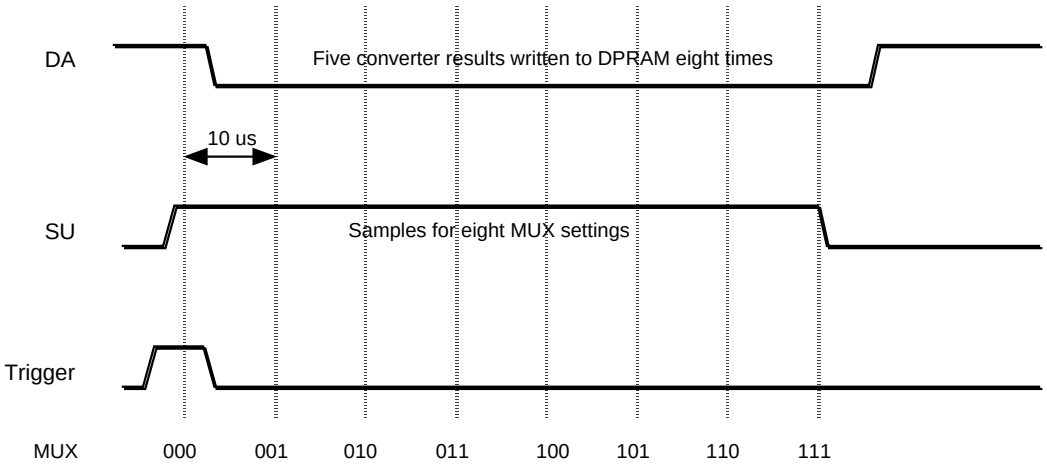

Figure 13 DA, SU, and Trigger Timing for ADCs in Single-Shot, All Channels Mode

Channel Subset Mode

In Channel Subset mode, the IndustryPack hardware does not sweep through the channels with the multiplexer. Instead, the user's setting in the AMSR (+4c) is used. Timing is similar to All Channels mode, except that only a single conversion is performed on all enabled converters.

Continuous, Channel Subset mode repeatedly performs the same channel conversions without sweeping the MUX:

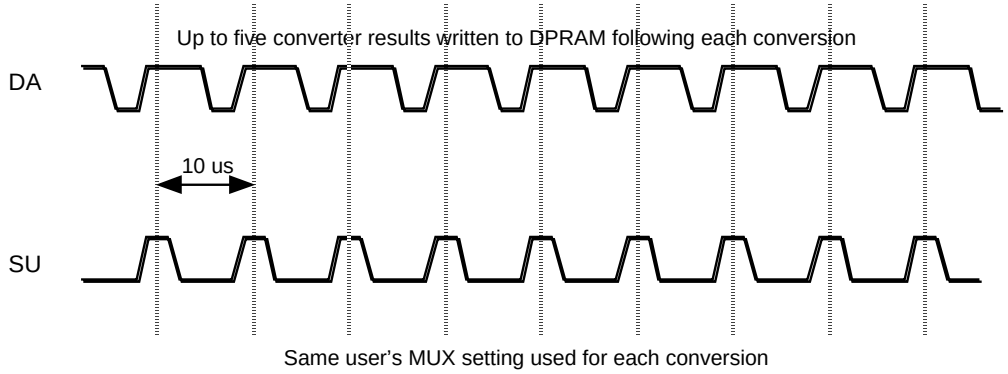

Figure 19 DA, SU Timing for ADCs in Continuous, Channel Subset Mode

Single-Shot, Channel Subset mode performs just one channel conversion for each trigger:

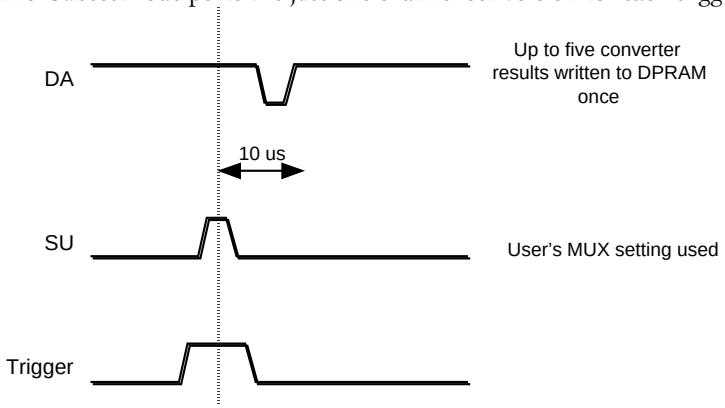

Figure 20 DA, SU, Trigger Timing for ADCs in Single-Shot, Channel Subset Mode

The timing for DA and SU on the IP-UL-ADC40 ADCs is shown in the following table:

	Mode				
	All Channels		Channel Subset		
Signal	Continuous	Single-Shot	Continuous	Single-Shot	**Units**
SU to next SU	80.0	-	10.0	-	µs
DA asserted duration	9.5	-	9.5	-	µs
DA negated duration	0.5	0.5	0.5	0.5	µs
SU asserted duration	70	70	0.125	0.125	µs
SU negated duration	10	-	9.875	-	µs
Trigger asserted to DA asserted	-	80	-	10	µs
SU asserted to DA asserted	80	80	10	10	µs
SU asserted to trigger negated	-	0.25	-	0.25	µs
SU asserted to trigger enabled	-	79.75	-	9.75	µs

Figure 21 ADC DA and SU Timing Relationship Table

The last line gives the timing for a signal not shown on the timing diagram. Since the ADCs take 10 µs to perform a conversion (eight times that for all MUX settings), the trigger is disabled for that period after the conversion starts. The result is that although the trigger is negated very quickly, it cannot be asserted until the conversion has finished.

Counts/Volts Conversions

Numbers read from the ADC are 12-bit signed-extended quantities for channels set to bipolar ranges, and 12-bit zero-filled quantities for channels set to unipolar ranges. This is summarized in the table below:

			Data format	
Range	**Type**	**Values**	**D15..D12**	**D11..D0**
0-5 V	Unipolar	0..4095	Zeroes	From ADC
0-10 V	Unipolar	0..4095	Zeroes	From ADC
±5 V	Bipolar	-2048..2047	D11 repeated four times	From ADC
±10 V	Bipolar	-2048..2047	D11 repeated four times	From ADC

For unipolar ranges, the most negative voltage is represented by $0000 and the most positive by $0fff. $0000 is 0V. Input voltages outside the range will show as the most negative or most positive values.

For bipolar ranges, the most negative voltage is represented by $0800 and the most positive by $f7ff. $0000 is the center of the range, 0V. Input voltages outside the range will show as the most negative or most positive values.

The number of bits in the converters is supplied by the AGSR (+22) in five bits at location 12..8. This is hard-coded to 12 (binary 01100).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	DPR	0	**WIDTH**					DA	SU	0	0	0	0	TS	IRQP	Read
X	X	X	**X**	**X**	**X**	**X**	**X**	X	X	X	X	X	X	X	X	Write
0	1	0	**0**	**1**	**1**	**0**	**0**	DA	SU	0	0	0	0	TS	0	Reset

Figure 22 Converter Width Bits in ADC General Status Register

Use the following equation to convert counts to volts for the ADC:

$$V = (C \times m) + b$$

Make sure the variables used are floating point and the number of counts, *C*, is derived from a signed or unsigned number, as appropriate for the range setting for the channel. Unipolar ranges require that *C* be derived from an unsigned number. Bipolar ranges require that *C* be derived from a signed number.

The value of the constant, *m* depends on the range selected. The table below shows the value to use:

Section	Range	*m* (V/Count)	*b* (V)	High Count	Low Count	One Bit (μV)
ADC	0-5 V	1.2207×10^{-3}	0.0	2048	-2048	1220.7
ADC	0-10 V	2.4414×10^{-3}	0.0	2048	-2048	2441.4
ADC	±5 V	2.4414×10^{-3}	0.0	4096	0	2441.4
ADC	±10 V	4.8828×10^{-3}	0.0	4096	0	4882.8

Figure 23 Count/Volt Conversion Parameters for ADCs

The High Count and Low Count columns show the count values that correspond to the extremes of the input voltage range. The One Bit column gives the size of a single count in microvolts.

Correcting Raw Data

An on-board EEPROM contains calibration coefficients that can be used to correct the raw data read from the ADCs. The corrections are linear, comprising an offset factor and a gain factor.

For corrected data, use the following equation to convert ADC counts to volts. This is normally applied to data read from the ADC:

$$V = (G \times C) + j$$

Note that this equation is different from the raw versions. A pair of *G* and *j* coefficients is supplied for each range and channel combination for the ADC. The factors *m* and *b* present in the raw equations do not appear here because they have already been rolled into *G* and *j*.

The code example below shows data being corrected as it is read from each channel. It assumes that the *G* and *j* coefficients have already been loaded into the arrays from the EEPROM, and that a bipolar range is being used:

```
#include <math.h>

#define CHANS 40
REAL64 G[CHANS],j[CHANS],volts[CHANS];

void corrADC(INT16 *membase)
{
    INT32 i;
    for(i=0;i<CHANS;i++)
       volts[i] = (membase[i*2+1])*G[i] + j[i];
}
```

The IP-UL-ADC40 has 160 coefficient pairs for the ADC section, one per channel for each of four range settings.

The section "Calibration Data" gives detailed information about the format and extraction of calibration coefficients from the EEPROM.

Programmable Timer

Overview

The timer has a selectable clock source, a prescaler, and a 16-bit down counter with several reload options. It can set a bit and interrupt when it passes zero. The timer either counts down to zero and stops, or counts down to zero, reloads, and repeats.

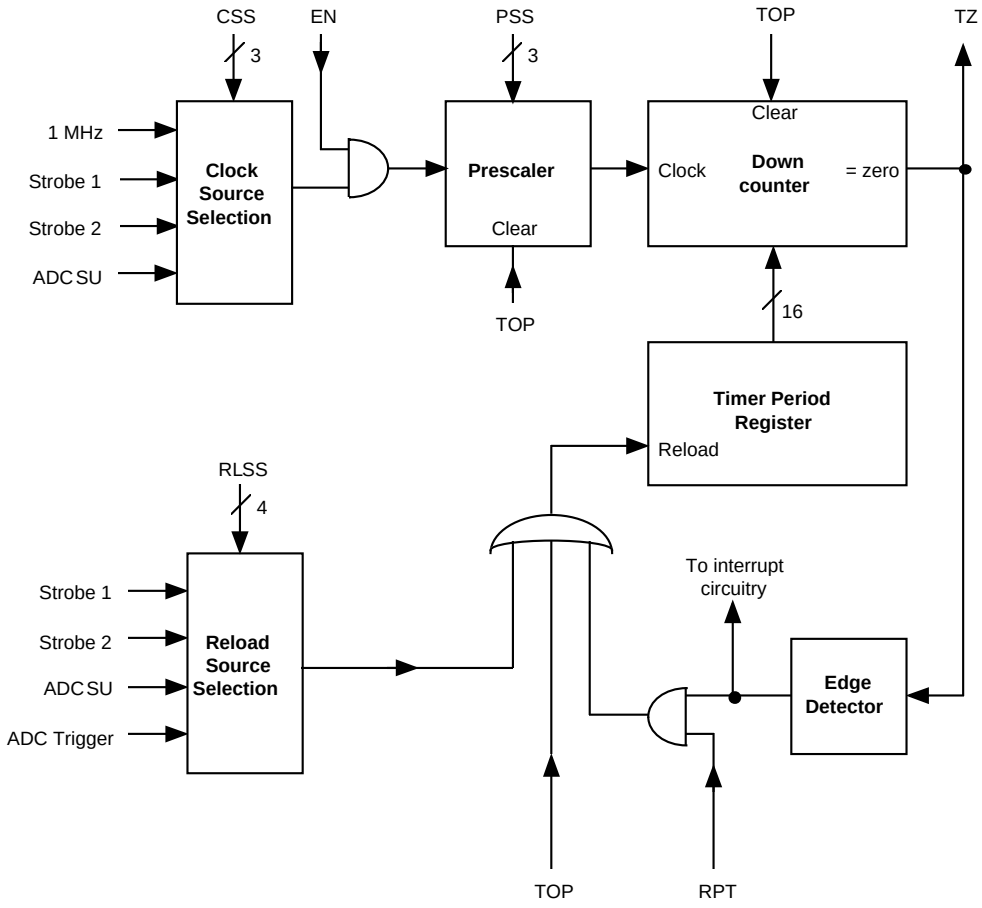

Figure 24 Block Diagram of the ADC Programmable Timer

The timer is controlled through two registers, the Timer Control and Status Register (ATCSR +3c and) and the Timer Period Register (ATPR +3e).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
RLSS				0	CSS			TZ	0	0	0	PSS		RPT	EN	Read
RLSS				X	CSS			X	X	TOP		PSS		RPT	EN	Write
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	Reset

Figure 25 The ADC Timer Control and Status Register

The various fields of the ATCSR may be programmed one at a time by reading the register, modifying it and writing it back, or may be programmed with a single write that sets all the fields and issues a timer operation at the same time.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
TIMER PERIOD																Read
TIMER PERIOD																Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Figure 26 The ADC Timer Period Register

The period register holds a 16-bit number that is loaded into the down-counter. To set a period of *n* prescaled clocks, set the ATPR to *n*-1. Valid values of *n* are 1 to 65535. A timer period of $0000 results in a count of 65536.

Starting and Stopping the Timer

Initially each timer is disabled, its enable bit (EN) being zero. Writing a one to this bit connects the clock source to the prescaler and allows the timer to count. The enable bit may be used to pause and restart the timer, but should be enabled during normal use.

The timer and prescaler can be cleared and the timer reloaded by software using the Timer Operation bits (TOP) described later. Whenever the timer is stopped at zero, it must be reloaded to start a count down.

Single and Repeated Timing

The Repeat bit (RPT) controls the behavior of the timer when it reaches zero. If RPT is zero the counter stops at zero. If RPT is one, then when the counter reaches zero it reloads the contents of the period register and continues to count. When repeating, the timer is at zero for one prescaled clock cycle.

If the period is changed while the timer is counting, the current down-count is not affected. The change will be noticed when the period register is next loaded into the timer. The Timer Operation bits (TOP) described later show how to load the timer explicitly.

Timer = Zero

The Timer = Zero (TZ) bit is the only output of the timer. It may be used to cause interrupts, start conversions, or issue pulses on the external Strobe pins.

TZ is initially asserted because the timer is sitting at zero. Since this bit is asserted only when the timer is actually at zero, if the timer is repeating then a fast clock may render it invisible to software. For this reason each timer's TZ bit is connected to an edge detector and a latch, visible in the AISR (+26) as the DTIM and PTIM bits respectively.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
DDA	DSU	0	0	0	**DTIM**	DTRG	0	PDA	PSU	0	0	0	**PTIM**	PTRG	0	Read
X	X	X	X	X	**X**	X	X	PDA	PSU	X	X	X	**PTIM**	PTRG	X	Write
0	0	0	0	0	**0**	0	0	0	0	0	0	0	**0**	0	0	Reset

Figure 27 Timer Bits in the ADC Interrupt Status Register

The DTIM bit is pulsed very briefly as TZ is asserted and is likely to be invisible. The PTIM bit is set each time the DTIM bit is pulsed. It can be reset by writing a one to PTIM. If timer interrupts are enabled, then PTIM will cause an interrupt when asserted.

The following example code fragment inputs an array of ADC values from a single channel, one per timer period:

```
void dacArray(INT16 *adc, INT16 *data, INT16 count, UINT16 us)
{
    UINT16 *aisr = (UINT16*)0xfff58026;     /* MVME162 slot A */
    UINT16 reg;

    *atpr = us-1;               /* Set the period in microseconds */
    *atcsr = 0x0003;            /* 1 MHz clock, repeating, enabled */
    while(count--) {
       *aisr = 0x0004;          /* Clear PTIM */
       do                       /* Wait for timer to pass zero */
          reg = *aisr;
       while((reg&0x04)==0);
       *data++ = *adc;          /* Input next data */
    }
}
```

This technique can be used to synchronize software to a repeating timer, or perform a single delay.

The assertion of TZ may issue a pulse on either of the two external Strobe signals. This selection is made in the AGCR (+20).

Clock Selection

By default, the clock source is a 1 MHz signal generated from the IndustryPack clock. The Clock Source Select (CSS) bits can choose other sources:

CSS	Clock source
000	1 MHz (default)
001	(reserved)
010	Strobe 1 signal
011	Strobe 2 signal
100	ADC Sample Update
101	(reserved)
110	(reserved)
111	(reserved)

010 and 011 selections (Strobes 1 or 2) select one of the external strobe pins as a clock source. An external clock can be used to implement precise delays that are not possible with a 1 MHz clock.

Selection 100 allows the timer to be clocked by the ADC sections' Sample Update signal. This allows the timers to be synchronized to conversions. The timers can be used as counters, interrupting regularly after N conversions have taken place.

Prescaler Settings

The prescaler divides the clock source by one of four fixed ratios. The default ratio is 1. The selection is made by the two Prescaler Select (PSS) bits. The table below shows the settings:

PSS	Division Ratio
00	1 (default)
01	10
10	100
11	1000

Changing the PSS setting automatically clears the prescaler. With the default clock selection, the prescaler provides period units of 1 µs, 10 µs, 100 µs, or 1 ms, allowing delays from 1 µs to 65.536s. By selecting an external clock source, periods of any length can be achieved.

Reloading and Resetting the Timer

Two mechanisms for reloading and resetting the timer are provided. The Timer Operation (TOP) bits provide a mechanism for reloading and resetting the timer and prescaler under software control. The Reload Source Select (RLSS) bits select a source that automatically reloads the timer.

The operations available via the TOP bits are shown below. The TOP bits always read as 00.

TOP	Operation
00	None
01	Reset counter and reset prescaler
10	Reload counter and reset prescaler
11	Reset prescaler

Resetting the counter will assert the TZ bit. If RPT is asserted, it will immediately reload and continue. If RPT is negated, it will stay at zero.

Reloading the counter will cause it to immediately start counting down. Reloading the counter with RPT negated provides a one-shot count down to zero initiated by software. Software can use this as a simple watchdog by making TZ generate an interrupt and issuing the reload command at a sufficiently high rate. Reloading the counter with RPT asserted lengthens the current period.

It is useful to reset the prescaler when the timer is being used as a counter. This ensures that there are no residual counts in the prescaler.

Reload can be also be performed automatically by selecting one of the Sample Update signals with the RLSS bits:

RLSS	Reload source	RLSS	Reload source
0000	Nothing (default)	1000	(reserved)
0001	(reserved)	1001	(reserved)
0010	Strobe 1 signal	1010	(reserved)
0011	Strobe 2 signal	1011	(reserved)
0100	ADC Sample Update	1100	(reserved)
0101	(reserved)	1101	(reserved)
0110	ADC trigger	1110	(reserved)
0111	(reserved)	1111	(reserved)

0010 and 0011 select one of the external strobe signals as a reload command. This can be used to start a conversion or cause an interrupt after a delay. It can also monitor an external signal and interrupt if its frequency falls below a certain threshold.

0100 causes a reload every time a conversion occurs. This can be used to make a conversion start an event after a defined delay. Examples include, an interrupt or a pulse on a Strobe pin.

0110 causes a reload when the trigger is asserted. This allows an action to take place some time after a conversion has been triggered. Such an action might be to read the ADC DPRAM or pulse some external hardware via one of the Strobe pins.

Interrupts

Section Interrupt Control

Each section of the IndustryPack has separate circuitry and registers for handling interrupts that originate from within that section. A single interrupt vector register in the Common section includes two bits that identify the section with the highest priority that has an interrupt asserted. The IndustryPack drives only one external interrupt signal: IRQ0*.

Interrupts in each section are controlled at two levels: one enable bit and pending flag for each section, and one enable bit and pending flag for each source within that section. There is no "master" interrupt enable for the whole IndustryPack. This function is carried out by the section enable bits. Below is a schematic of the high-level interrupt control. It shows only the section-level enable bits and pending flags.

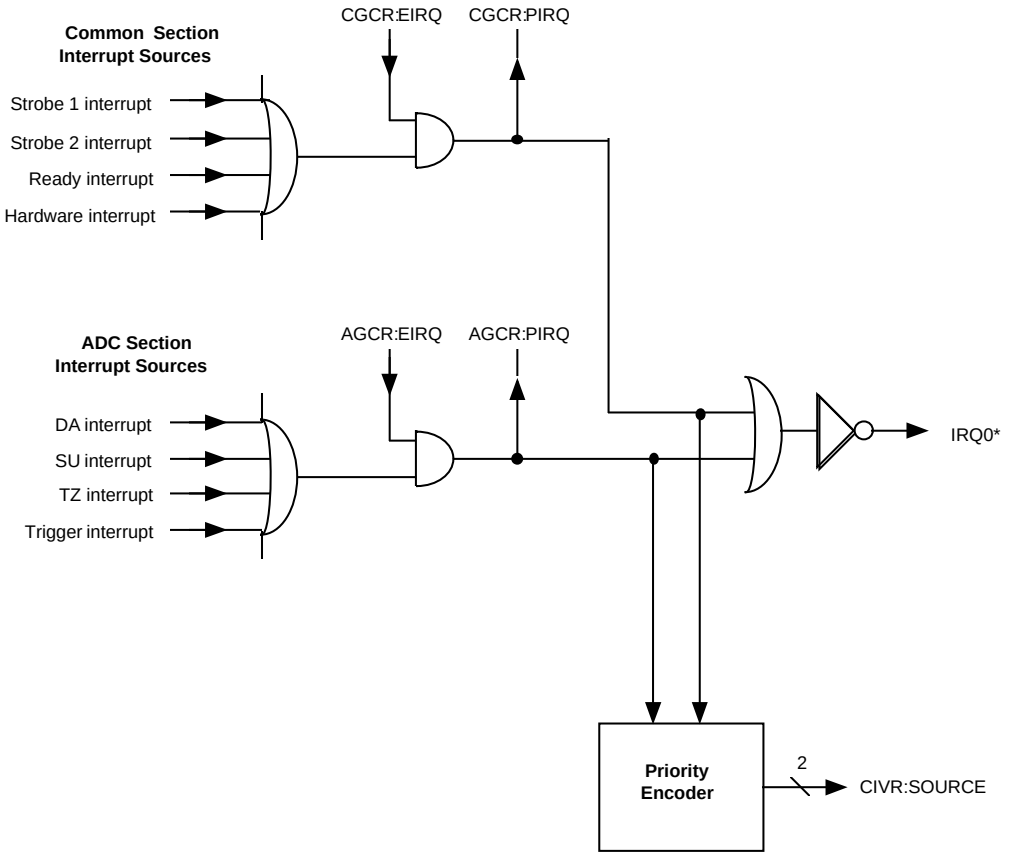

Figure 28 IP-UL-ADC40 High Level Interrupt Structure

The interrupt sources on the left are all latched levels. If that source has an interrupt pending, the source will be asserted. It is normally one of the duties of the interrupt service routine to reset the latches associated with the sources it handles.

Note that the PIRQ bit for each section is only asserted if that section is contributing to an interrupt condition. Disabling that section's interrupt by negating the EIRQ bit also negates the PIRQ bit. Thus software can quickly determine whether a section is interrupting by examining its PIRQ bit.

The EIRQ and PIRQ bits are located in the same position (bit 0) of the General Control and General Status registers for each of the sections.

Interrupt Vector

The interrupt vector is stored in the lower eight bits of the Common Interrupt Vector Register (+0a), as shown below:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
TEST USE								VECTOR						SOURCE		Read
TEST USE								VECTOR						X	X	Write
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Reset

Figure 29 Common Interrupt Vector Register

The upper eight bits are for test use. They can be set to any value, since they are ignored. The next six store the upper six bits of the vector provided by software. The lower two bits encode the source of the interrupt as follows:

SOURCE	Interrupt Source
00	Common (highest priority)
01	(unused)
10	ADC
11	DAC (unused, lowest priority)

Note that the vector resets to $000f. This is the Motorola Uninitialized Vector Exception vector. All sixteen vector register bits are read/write until the first interrupt occurs. Only then are the SOURCE bits enabled, overriding whatever software wrote to them.

Source Interrupt Control

Each interrupt source has its own enable bit and pending flag, just as the IndustryPack sections do. In addition, the interrupt sources have a resettable latch that stores the pending status for the interrupt source. Some sources have programmable polarity control, but in most cases this is hard-wired.

The schematic below shows the interrupt structure for each individual interrupt source. The input on the left can be either a pulse or a level, depending on the source. The output on the right is one of the inputs to the interrupt circuitry for the section. It is asserted when the interrupt is pending and enabled, negated otherwise.

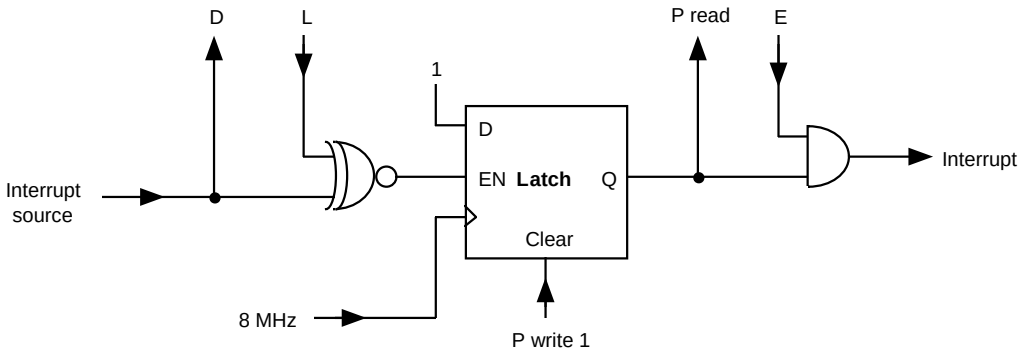

Figure 30 Interrupt Source Latch Detail

Note that an interrupt source that maintains a level will keep setting the flip flop even if the flip flop is cleared. Interrupt routines must disable the source in this case. Pulsed sources will not exhibit this behavior. Note also that the latch may be set by a power-up or reset, depending on the how the source behaves at this time.

The D, L, P, and E bits are known as the Data, Polarity, Pending, and Enable bits respectively. They occupy consistent positions in the Interrupt Control and Interrupt Status registers for each section, as shown below:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
Polarity bits (L)								Enable bits (E)								Read
Polarity bits (L)								Enable bits (E)								Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Figure 31 Interrupt Control Register Layout

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
Data bits (D)								Pending bits (P)								Read
X	X	X	X	X	X	X	X	Clear bits (P)								Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Figure 32 Interrupt Status Register Layout

Each source occupies the same position in each part of each register. The ADC Interrupt Control Register (+24) and The ADC Interrupt Status Register (+26) reproduced below show this pattern. There are no programmable polarities in the ADC section. Unimplemented sources are hard-wired to zero:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	EDA	ESU	0	0	0	ETIM	ETRG	0	Read
X	X	X	X	X	X	X	X	EDA	ESU	X	X	X	ETIM	ETRG	X	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Figure 33 ADC Interrupt Control Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
DDA	DSU	0	0	0	DTIM	DTRG	0	PDA	PSU	0	0	0	PTIM	PTRG	0	Read
X	X	X	X	X	X	X	X	PDA	PSU	X	X	X	PTIM	PTRG	X	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Figure 34 ADC Interrupt Status Register

The Polarity bits, where implemented, are read/write, as are the Enable bits. The Data bits are read-only, and in many cases not useful. Pulsed interrupt sources may produce very short pulses.

Polarity bits set to zero set the latch when the source is asserted. Polarity bits set to one set the latch when the source is negated. In this way, software can interrupt on either or both edges of a signal as it requires. The Ready signal has a programmable interrupt polarity on this IndustryPack.

The Pending bits double up as clear bits for the interrupt latch. Any Pending bit written with a one will clear the latch. Zeroes have no effect. To clear all interrupts that are asserted and enabled, read the status register, AND all sixteen bits with the contents of the control register, and write the result back to the status register.

Interrupt Handling

The code fragment below sets up two interrupt handlers, one for the Common section and one for the ADC section. If both interrupts are active simultaneously, the Common handler will be called because Common interrupts have a higher priority than do ADC interrupts:

```
#define VECTOR 0x80
#define ADCVECTOR (VECTOR+2)
#define COMVECTOR (VECTOR+0)

void installHandlers()
{
    UINT16 *civr = (UINT16*)0xfff5800a;      /* Motorola 162 slot A */
    UINT16 *cgcr = (UINT16*)0xfff58000;
    UINT16 *cicr = (UINT16*)0xfff58004;
    UINT16 *cisr = (UINT16*)0xfff58006;
    UINT16 *agcr = (UINT16*)0xfff58020;
    UINT16 *aicr = (UINT16*)0xfff58024;
    UINT16 *aisr = (UINT16*)0xfff58026;

    osInstallHandler(ADCVECTOR,isrADC);      /* Install timer handler */
    osInstallHandler(COMVECTOR,isrCOM);      /* Install common handler */

    *civr = VECTOR;        /* Set vector */
    *aicr = 0x0004;        /* Enable ADC timer IRQ */
    *gicr = 0x0004;        /* Enable Common Ready IRQ */

    *aisr = 0x0004;        /* Clear ADC timer IRQ latch */
    *gisr = 0x0004;        /* Clear Common Ready IRQ latch */
    *agcr = 0x0001;        /* Enable ADC section IRQ */
    *ggcr = 0x0001;        /* Enable Common section IRQ */
}
```

Note that the interrupt source latches are cleared before the sections are enabled. This ensures that the interrupt will not occur immediately if the latch is already set.

The next code fragment shows the timer interrupt handler. The handler reads all the ADC channels into a buffer and clears the interrupt.

```
INT16 *buffer;
INT32 count;

void isrADC()
{
    UINT16 *aisr = (UINT16*)0xfff58026;
    INT16 *adc = (INT16*)0xc0000000;
    INT32 i;

    *aisr = 0x0004;        /* Clear ADC timer IRQ latch */

     if(count>0) {               /* Read the data if room */
        for(i=0;i<CHANS;i++)
            *buffer++ = adc[i*2+1];
        count--;
     }
}
```

The Common section handler clears the not ready condition and clears the interrupt in the same way.

This final fragment shows cleanup to disable the interrupt sources and remove the handlers:

```
void removeHandlers()
{
    UINT16 *agcr = (UINT16*)0xfff58020;
    UINT16 *cgcr = (UINT16*)0xfff58000;

    *agcr = 0x0000;        /* Disable ADC section IRQ */
    *cgcr = 0x0000;        /* Disable Common section IRQ */

    osRemoveHandler(ADCVECTOR,isrADC); /* Remove timer handler */
    osRemoveHandler(COMVECTOR,isrCOM); /* Remove Common handler */
}
```

Interrupts should always be disabled *before* removing handlers, of course.

Triggers

Overview

The ADC section contains a trigger. When a section is in Single-Shot mode, a conversion can only be started by its trigger. When in Continuous mode conversions automatically occur one after the other and the trigger is not needed for this purpose. The trigger is really nothing more than a well-connected set-reset flip-flop.

The trigger is asserted by one of a number of software-selectable sources. The trigger may be asserted by a timer reaching zero, a software command, an external strobe, or a conversion. Triggers are reset automatically or manually by a software command. Triggers cannot be asserted while a conversion cycle is in progress.

Triggers may also be routed to one or both of the external Strobe pins. The Strobe will be pulsed when the trigger is asserted.

The diagram below shows the organization of the trigger circuitry:

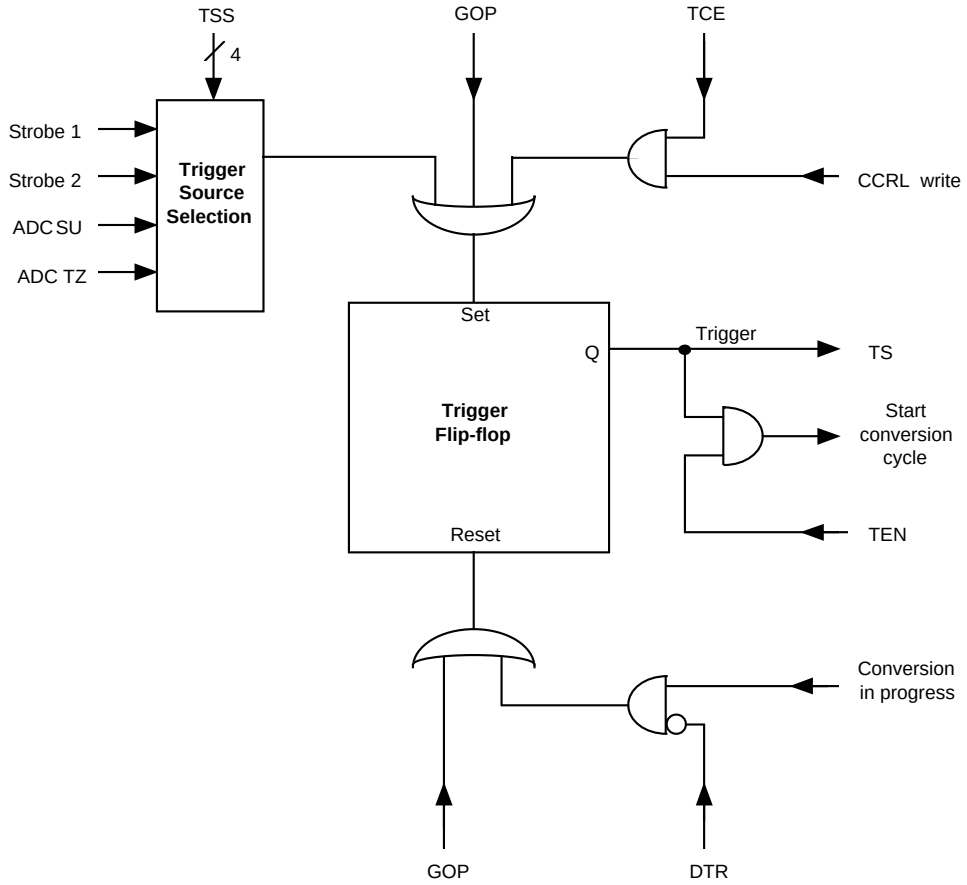

Figure 35 ADC Trigger Block Diagram

The trigger is controlled by the ADC Trigger Source Register (ATSR +30).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
TSS				0	0	0	0	0	DTR	0	0	0	TCE	TS	TEN	Read
TSS				X	X	X	X	X	DTR	GOP		X	TCE	X	TEN	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Figure 36 ADC Trigger Source Register

Trigger Source Selection

The trigger source is selected by the Trigger Source Select bits. These select one of a number of trigger sources, as below:

TSS	Trigger source
0000	Nothing (default)
0001	(reserved)
0010	Strobe 1 signal
0011	Strobe 2 signal
0100	ADC Sample Update
0101	(reserved)
0110	ADC timer = zero
0111	(reserved)

TSS	Trigger source
1000	(reserved)
1001	(reserved)
1010	(reserved)
1011	(reserved)
1100	(reserved)
1101	(reserved)
1110	(reserved)
1111	(reserved)

0010 and 0011 select the external Strobe signals as trigger sources. This gives a simple mechanism for controlling conversions with an external pacer clock.

0100 selects the SU signals from the ADC section. If a section is running in Continuous mode, its SU signal is being asserted once per conversion cycle.

0110 selects the ADC timer as a trigger source. This can be used to provide repetitive triggers, or delayed triggers based on some other event, such as an external Strobe pulse.

In addition to the hardware trigger source, the trigger can be asserted by software in two ways: a trigger command (covered below) or a write to the ACCRL. By asserting the TCE bit, a write to the ACCRL register can select a set of converters and start a conversion with a single action. Negating TCE prevents ACCRL writes from asserting the trigger.

Trigger Operations

Software may assert or negate the trigger explicitly with the GOP bits. This permits full software control of the trigger at any time.

GOP	Operation
00	(ignored)
01	Assert trigger
10	Negate trigger
11	(reserved)

If the Disable Trigger Reset (DTR) bit is asserted then the trigger is not negated automatically. Software must negate the trigger. After setting up trigger sources, it is advisable to clear the trigger in case it has inadvertently been set before enabling the trigger with the TEN bit.

Trigger State

The trigger state may be read at any time via the TS bit. This bit is mirrored in the General Status Register for each section:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	DPR	0	WIDTH					DA	SU	0	0	0	0	**TS**	IRQP	Read
X	X	X	X	X	X	X	X	X	X	X	X	X	X	**X**	X	Write
0	1	0	1	0	0	0	0	0	0	0	0	0	0	**0**	0	Reset

Figure 14 Trigger State in the ADC General Status Register

The TS bit is also connected to a rising edge detector and a latch in the interrupt circuitry. These bits two bits (DTRG and PTRG) may be used to generate an interrupt when the trigger is asserted, or provide a bit that can be polled to catch trigger assertion.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
DDA	DSU	0	0	0	DTIM	**DTRG**	0	PDA	PSU	0	0	0	PTIM	**PTRG**	0	Read
X	X	X	X	X	X	**X**	X	PDA	PSU	X	X	X	PTIM	**PTRG**	X	Write
0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	**0**	0	Reset

Figure 15 Trigger Bits in the ADC Interrupt Status Register

Starting Conversion Cycles with Triggers

In Single-Shot mode, a conversion is started by the assertion of the trigger signal, provided that the Trigger Enable (TEN) bit is asserted. Triggers that occur while a conversion cycle is underway are ignored, so care must be taken to ensure that triggers are spaced adequately.

Leaving the trigger asserted will not cause conversions to occur continuously. In Continuous mode the trigger is ignored.

Disabling the Trigger

The trigger may be prevented from starting a conversion by negating the Trigger Enable (TEN) bit. This can be useful when setting up the trigger since it prevents conversions from starting unexpectedly.

Using Triggers to Generate External Strobe Pulses

Triggers can generate pulses on the external Strobe pins if the AGCR (+00) is programmed appropriately.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	**STS2**		**STS1**		0	0	0	0	0	0	CS	SS	IRQE	Read
X	X	X	**STS2**		**STS1**		X	X	X	X	X	X	CS	SS	IRQE	Write
0	0	0	**0**	**0**	**0**	**0**	0	0	0	0	0	0	0	0	0	Reset

Figure 37 Strobe Select Bits in the ADC General Control Register

Set up the Strobe Select by writing 01. This selection generates a pulse each time the trigger is asserted.

STS	Pulse Selection
00	Nothing
01	Trigger asserted
10	SU asserted
11	Timer became zero

External Strobes

Overview

Two external Strobe signals are provided, Strobe1 and Strobe 2. These may be used as inputs to the timers as clocks or reload signals, as trigger sources, as interrupt sources, and may be read by software. They may be used as pulsed outputs, controlled by the timers, the triggers, the Sample Update signals of either analog section, or by software.

Strobe1 is located on the I/O connector at pin 42. Strobe2 is located on the IP bus connector at pin 46. These are open-drain signals, which are passively pulled high. The signal is asserted when pulled low.

Strobe2 is usable if it is supported by the carrier board holding the IP-UL-ADC40. The MVME162 has a programmable pacer clock connected to this pin, for instance. GreenSpring's VIPC616 and 618 IndustryPack carriers bring this signal out to a connector so that the user may connect it as required.

The IP-UL-ADC40 has 1 kOhm on-board pull-ups on Strobe1 and Strobe2.

The connections to Strobe1 are shown in the schematic below. Strobe 2 is wired in the same way:

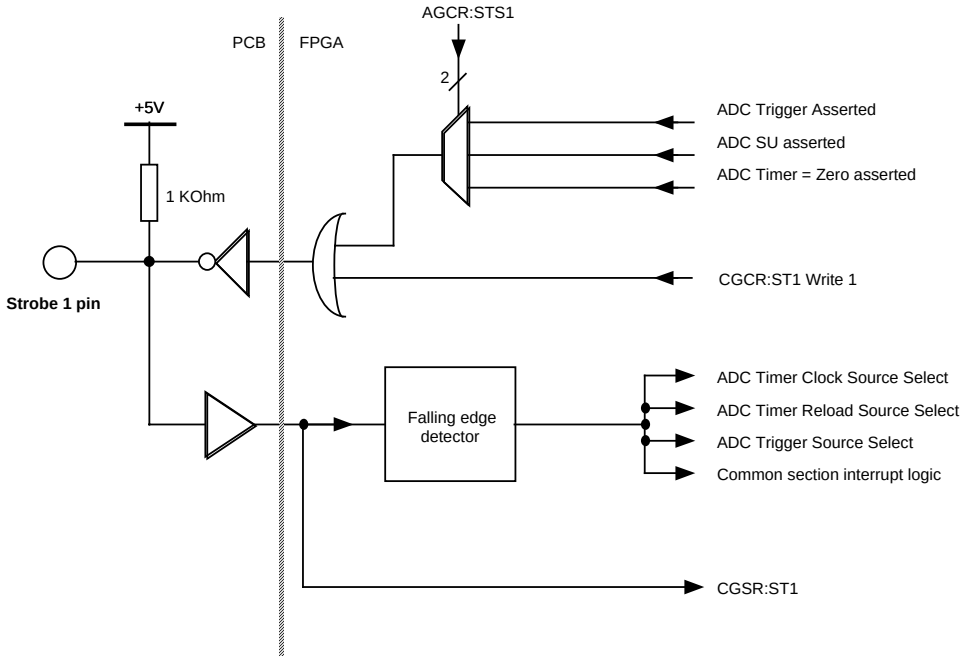

Figure 38 Strobe 1 Connection Schematic

Software-Initiated Strobes

Direct control of the levels on the Strobe pins is not provided. However, software may generate pulses by writing to the Strobe1 and Strobe2 (ST1 and ST2) bits of the CGCR (+00):

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	**0**	**0**	0	0	0	0	0	IRQE	Read
X	X	X	X	X	X	X	X	**ST2**	**ST1**	X	X	X	0	0	IRQE	Write
0	0	0	0	0	0	0	0	**0**	**0**	0	0	0	0	0	0	Reset

Figure 39 Strobe Pulse Bits in the Common General Control Register

Each write to the CGCR with bit 6 or 7 set to one creates a pulse approximately 250 ns long on the respective Strobe pin(s). Writes with the bits reset have no effect. Since the Strobe pins may be used as inputs to the timer and trigger, this is an handy method for testing the operation of software that expects external pulses.

Current Strobe State

The current state of the Strobe pins may be checked at any time by reading Strobe State bits in the CGSR (+02). A one indicates that the Strobe pin is asserted (low voltage). A zero indicates that the Strobe pin is negated (high voltage).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	**ST2**	**ST1**	0	IP32	HWE	RDY	0	IRQP	Read
X	X	X	X	X	X	X	X	**X**	**X**	X	X	X	X	X	X	Write
0	0	0	0	0	0	0	0	**ST2**	**ST1**	0	IP32	0	1	0	0	Reset

Figure 40 Strobe State Bits in the Common General Status Register

Some carriers may require programming before Strobe2 is in the negated state. The MVME162 is one of these.

Hardware-Initiated Strobes

Two bits in ADC General Control register (AGCR +20) control the automatic generation of strobe pulses by the timers, triggers, and analog conversions.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	**STS2**		**STS1**		0	0	0	0	0	0	CS	SS	IRQE	Read
X	X	X	**STS2**		**STS1**		X	X	X	X	X	X	CS	SS	IRQE	Write
0	0	0	**0**	**0**	**0**	**0**	0	0	0	0	0	0	0	0	0	Reset

Figure 41 Strobe Select Bits in the ADC General Control Register

The strobe select bits can be written as follows:

STS	Pulse Selection
00	None
01	Trigger asserted
10	SU asserted
11	Timer became zero

Each time the selected action occurs, a pulse is sent to the Strobe pin. The ADC and Common sections as well as software can send pulses to the same pin.

Using Strobes for Interrupts

Strobes can generate interrupts in the Common section via the CICR (+04) and CISR (+06).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
DST2	**DST1**	0	0	DHWE	DRDY	0	0	**PST2**	**PST1**	0	0	PHWE	PRDY	0	0	Read
X	**X**	X	X	X	X	X	X	**PST2**	**PST1**	X	X	PHWE	PRDY	X	X	Write
0	**0**	0	0	0	0	0	0	**0**	**0**	0	0	0	0	0	0	Reset

Figure 42 Strobe Bits in the Common Interrupt Status Register

The Strobe signals are passed through a rising-edge detector and a narrow pulse sent to the DST bits. These are unlikely to be visible in the register because they are so short. A latched version of the pulse is available in the PST bits. The PST bits are reset by writing a one to them.

Using Strobes to Clock or Reload Timers

Strobes can generate clocks or provide reload commands for the timer. Clock Source Select (CSS) and Reload Source Select (RLSS) fields of the ADC Timer Control and Status Register (ATCSR +3c) control the source of these signals.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
RLSS				0	**CSS**			TZ	0	0	0	PSS		RPT	EN	Read
RLSS				X	**CSS**			X	X	TOP		PSS		RPT	EN	Write
0	**0**	**0**	**0**	0	**0**	**0**	**0**	1	0	0	0	0	0	0	0	Reset

Figure 43 Reload and Clock Source Selection in the Timer Control and Status Register

To select the Strobe signals as sources, write the fields as shown below:

Source selected	CSS	RLSS
Strobe1	010	0010
Strobe2	011	0011

Figure 44 Timer Selections for Strobes

Using Strobes as Triggers

Strobes can generate a trigger for the ADC section. The Trigger Source Select (TSS) field of the ADC Trigger Source Register (ATSR +30) controls the source of this signal.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
TSS				0	0	0	0	0	DTR	0	0	0	TCE	TS	TEN	Read
TSS				X	X	X	X	X	DTR	GOP		X	TCE	X	TEN	Write
0	**0**	**0**	**0**	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Figure 45 Trigger Source Select Bits in the ADC Trigger Select Register

To select the Strobe signals as sources, write the TSS field as shown below:

Source selected	TSS
Strobe1	0010
Strobe2	0011

Figure 46 Trigger Selections for Strobes

Strobe Signal Timing

The strobe pulses output by the IndustryPack are 250 ns wide. Rise times typically show a 500 ns time constant. Pulses supplied to the IndustryPack should also be at least 250 ns wide.

Application Notes

Interconnections

The timers, triggers, synchronization signals, and strobes can be connected together in many combinations by programming the appropriate registers. This flexibility provides a wide range of options for system design.

The following sections describe some common applications.

Sampling at Regular Intervals

Regular sampling is a very common data acquisition requirement. Continuous mode samples at regular intervals, but the rate is fixed. The rate at which the IP-UL-ADC40 samples can be controlled by running the ADC section in Single-Shot mode and starting conversions with one of the timers. The diagram below shows this arrangement:

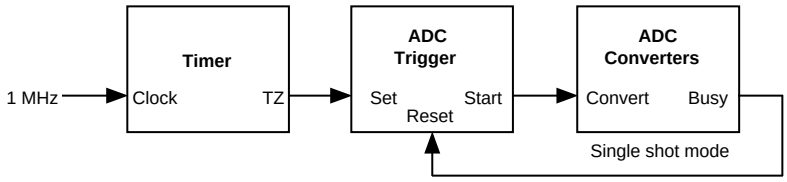

Figure 47 Sampling at Regular Intervals

Program the timer to divide the clock source to the required intersample delay. Select the Timer=zero signal as the trigger source, and ensure that the trigger is enabled. If the ADC section is in Single-Shot mode, each trigger will perform one conversion cycle. In All Channels mode, all forty channels will be converted in 80 µs.

The following code snippet sets this up:

```
*agcr = 0x0002;           /* ADC in Single-Shot, All Channels mode */
*atpr = delay-1;          /* Set sample time */
*atcsr = 0x0003;          /* Enable timer in repeat mode, 1 MHz */
*atsr = 0x6021;           /* Trigger source is ADC timer, reset, enable */

/* Read samples from DPRAM after each TZ pulse */
```

If Channel Subset mode is used, then a set of five channels can be converted in 10 µs.

Delaying a Conversion

Conversions can be delayed by using a timer. In this example, an external Strobe pulse is used to start an ADC conversion, but the application requires that the conversion start after a fixed time. In addition, the application needs an external pulse at the start of the ADC conversion to synchronize some more hardware and must run some software as soon as the original pulse arrives to control a part of the analog input section prior to the conversion.

The diagram below shows this implemented on the IP-UL-ADC40:

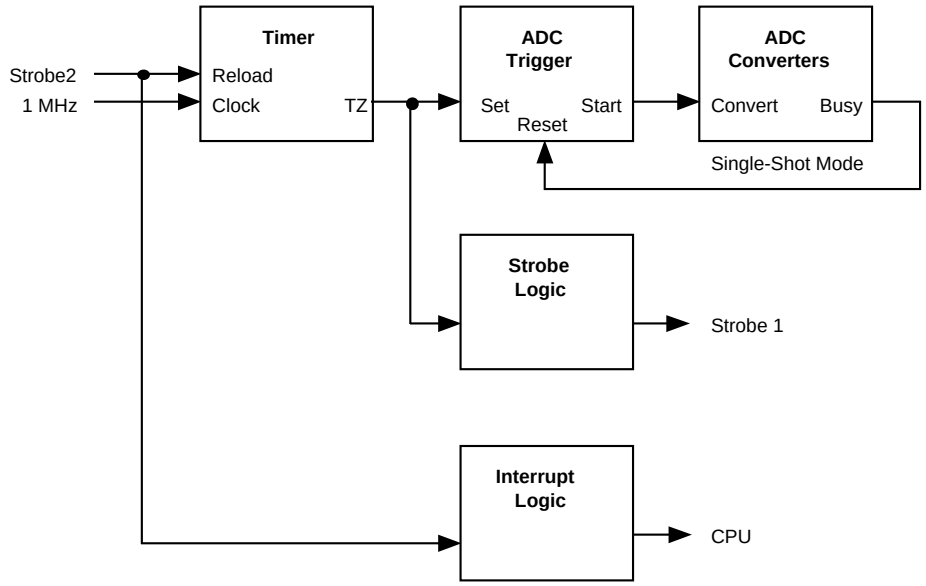

Figure 48 Delaying a Conversion

The timer is enabled, but the RPT bit is reset. It is also programmed to reload on external Strobe2. The period register is programmed with an appropriate delay. The interrupt routine is entered some (variable) time after the external strobe pulse arrives, clears the pending interrupt, manipulates the hardware, and returns. A short time after that, the timer reaches zero and the trigger causes the ADCs to sample and convert, and the external Strobe1 drives the timing of some other hardware.

Outputting Synchronized Pulses

The external Strobe signals can be driven by a variety of sources to provide external synchronizing pulses. In the example below, a free-running ADC is outputting a strobe pulse every N samples:

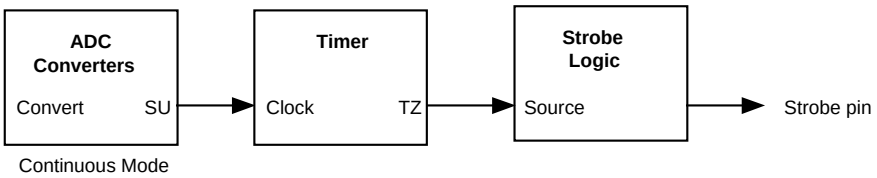

Figure 49 Outputting Synchronized Pulses

The ADC SU signal clocks the timer which counts to zero and repeats. The Strobe logic outputs a pulse on each transition to zero.

Triggering ADCs by Software

If jitter is not a problem, or if a very predictable CPU is used, software can trigger the ADC section itself via the TOP bits in the ATSR (+30).

Calibration Data

Overview

Calibration data is stored in an on-board EEPROM. The parameters it provides can be used to convert raw counts directly into voltages for each range setting.

Calibration parameters are supplied as a coefficient pair: a gain and an offset. The IP-UL-ADC40 has 160 coefficient pairs: one for each channel and range combination.

Accessing the EEPROM

The on-board EEPROM is a 16 Kbit device, arranged as 1024 words, each of 16 bits. It can be read, written, locked and unlocked. As supplied, the device is locked to prevent accidental alteration.

Access to the EEPROM is via the Common Configuration Control and Status Register (+10) and the Common Configuration Data Register (+12). These are shown below:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	SIZE					0	0	0	0	0	0	0	CE	Read
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	CE	Write
0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	Reset

Figure 50 Common Configuration Control and Status Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	CD	Read
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	CD	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

Figure 51 Common Configuration Data Register

The SIZE field encodes the size of the storage device in bytes. It is hard-wired to 01011, binary for 11, indicating that there are 2^{11} bytes of storage. CE is used to control the start and end of communication with the serial EEPROM. The least significant bit of the data register is used to pass serial commands and data to and from the device. Accesses to the data register take one microsecond each due to the timing requirements of the EEPROM device.

Copy all of the EEPROM out into RAM for manipulation. This will simplify extraction of the data.

All address, commands, and data is transferred most-significant bit first.

Read EEPROM Data

EEPROM data is read in 16-bit words. For each word, do the following:

Write CE to one to initiate the operation
Write CD three times with data 110 to indicate a read command
Write CD SIZE-1 times to supply the word address
Read CD sixteen times to get 16 bits of data
Write CE to zero to terminate operation

The CE bit may be written zero at any time to abort the operation. Valid addresses to read are in the range $0000 to $03ff.

The following code fragment provides a C routine to read an EEPROM word:

```
void readEE(UINT16 *dest, INT32 address)
{
    UINT16 *cccsr = (UINT16*)0xfff58010;  /* MVME 162 slot A */
    UINT16 *ccdr = (UINT16*)0xfff58012;
    UINT32 data;
    UINT32 mask;
    UINT16 reg;
    INT32 size;

    size = (*cccsr>>8)&0x1f;            /* Get size of device */
    *cccsr = 0x0001;                    /* Assert CE */

    data = 0x0006;
    for(mask=0x0004;mask;mask>>=1)      /* Write out all bits of command */
        *ccdr = (mask&data)?1:0;
    mask=0x0001<<(size-2);              /* Write out all bits of address */
    for(;mask;mask>>=1)
        *ccdr = (mask&address)?1:0;
    for(mask=0,data=0;mask<16;mask++) { /* Read in data word */
        reg = *ccdr;                    /* Use reg to avoid problems */
        data = (data<<1)+(reg&0x0001);
    }

    *cccsr = 0;                         /* Negate CE */
    *dest = data;                       /* Return result */
}
```

The following routine reads the whole EEPROM into RAM at the destination given.

```
void readWholeEE(VOID *dest)
{
    UINT16 *cccsr = (UINT16*)0xfff58010;  /* MVME 162 slot A */
    INT32 words,i;
    UINT16 *d = (UINT16*)dest;

    words = 1<<(((*cccsr>>8)&0x1f)-1);    /* Get words in device */

    for(i=0;i<words;i++)                  /* Read all the words */
        readEE(&d[i],i);
}
```

Unlock EEPROM

The IndustryPack is shipped with the EEPROM locked. It must be unlocked before writes are effective. A locked EEPROM may still be read.

Write CE to one to initiate the operation
Write CD five times with 10011 to indicate an unlock command
Write CD SIZE-3 times with dummy data
Write CE to zero to terminate operation

CE may be written with 0 at any time to abort the operation.

The following code fragment provides a C routine to unlock the EEPROM:

```
    void unlockEE()
    {
        UINT16 *cccsr = (UINT16*)0xfff58010;  /* MVME 162 slot A */
        UINT16 *ccdr = (UINT16*)0xfff58012;
        UINT32 data;
        UINT32 mask;
        INT32 size;

        size = (*cccsr>>8)&0x1f;            /* Get size of device */
        *cccsr = 1;                         /* Assert CE */

        data = 0x0013;                      /* Make up command */
        data <<= size-3;
        mask=0x0001<<(size+1);              /* Write out all bits of command */
        for(;mask;mask>>=1)
            *ccdr = (mask&data)?1:0;

        *cccsr = 0;                         /* Negate CE */
    }
```

Write EEPROM Data

The EEPROM must be unlocked for writes to have any effect. The IndustryPack is shipped with the EEPROM locked.

Write CE to one to initiate the operation
Write CD three times with data 101 to indicate a write command
Write CD SIZE-1 times to indicate the word address
Write CD 16 times with 16 bits of data
Hardware starts programming device
Poll CD bit to check busy status. Zero means busy, one means done.
Once not busy, write CE to zero to terminate operation

Writing may take up to 10 ms per word. If polling is not possible, the timer can be used to generate an interrupt after an appropriate period. The CE bit may be written zero at any time to abort the operation. Valid addresses to write are in the range $0000 to $03ff.

The following code fragment provides a C routine to write an EEPROM word:

```
void writeEE(UINT16 *source, INT32 address)
{
    UINT16 *cccsr = (UINT16*)0xfff58010;  /* MVME 162 slot A */
    UINT16 *ccdr = (UINT16*)0xfff58012;
    UINT32 data;
    UINT32 mask;
    UINT16 reg;
    INT32 size;

    size = (*cccsr>>8)&0x1f;            /* Get size of device */
    *cccsr = 0x0001;                    /* Assert CE */

    data = 0x0005;
    for(mask=0x0004;mask;mask>>=1)      /* Write out all bits of command */
        *ccdr = (mask&data)?1:0;
    mask=0x0001<<(size-2);              /* Write out all bits of address */
    for(;mask;mask>>=1)
        *ccdr = (mask&address)?1:0;
    data = *source;                     /* Write out all bits of data */
    for(mask=0x8000;mask;mask>>=1)
        *ccdr = (mask&data)?1:0;
    for(;;) {                           /* Poll for write not busy */
        reg = *ccdr;
        if((reg&0x0001)==1)
           break;
    }

    *cccsr = 0;                         /* Negate CE */
}
```

Lock EEPROM

The IndustryPack is shipped with the EEPROM locked. It should be relocked after being written to prevent accidental alteration.

Write CE to one to initiate the operation
Write CD five times with 10000 to indicate an unlock command
Write CD SIZE-3 times with dummy data
Write CE to zero to terminate operation

CE may be written with 0 at any time to abort the operation. The following code fragment provides a C routine to lock the EEPROM:

```
void lockEE()
{
    UINT16 *cccsr = (UINT16*)0xfff58010;  /* MVME 162 slot A */
    UINT16 *ccdr = (UINT16*)0xfff58012;
    UINT32 data;
    UINT32 mask;
    INT32 size;

    size = (*cccsr>>8)&0x1f;            /* Get size of device */
    *cccsr = 1;                         /* Assert CE */

    data = 0x0010;                      /* Make up command */
    data <<= size-3;
    mask=0x0001<<(size+1);              /* Write out all bits of command */
    for(;mask;mask>>=1)
        *ccdr = (mask&data)?1:0;

    *cccsr = 0;                         /* Negate CE */
}
```

Calibration Data Format

The calibration data is organized in a format common to all Unilin IndustryPacks. The contents comprise a header, an address table, a number of calibration tables, and free space. All values are stored in big-endian format. All addresses are given as word addresses (a word is 16 bits).

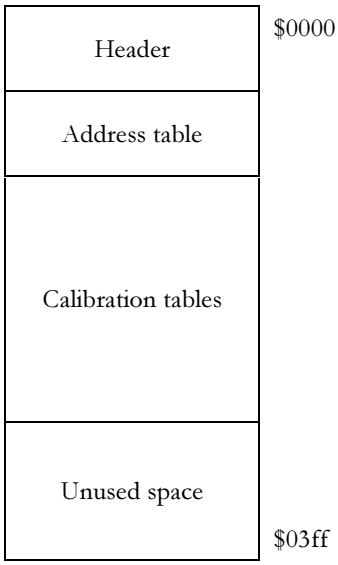

Figure 52 EEPROM Storage Layout

Data Header

The data header format is shown below as a series of 16-bit words.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address
0	1	0	1	0	1	0	1	0	1	0	0	1	1	0	0	$000
REV			SIZE					ADDRESS TABLE								$001
0	0	0	0	1	1	0	1	0	0	0	0	1	0	1	0	$002
1	1	1	1	0	0	1	0	1	1	1	1	0	1	0	1	$003
SERIAL HIGH																$004
SERIAL LOW																$005
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$006
0	0	0	0	0	0	0	0	CHECKSUM								$007

The first word is a constant that represents the ASCII for “UL”.

The second word contains revision data, currently 000, the encoded size of the EEPROM, binary 01011 meaning 2^{11} bytes, and the word address of the start of the address table.

The third and forth words contain constant values $0d0a and $f2f5. These can be used to ensure that the data has not been scrambled by end of line problems in file systems or by endian changes.

The fifth and sixth words make up the IndustryPack serial number. This is an unsigned 32-bit number that is never zero. The high word is stored at the lower address. This number matches the serial number label on the board.

The seventh word is currently unused. It is set to zero.

The eighth word contains a checksum in the lower byte. The upper word is unused. The checksum is the ones complement of the XOR of the other fifteen header bytes. The following code fragment will calculate the checksum given a pointer to an array of words containing the header:

```
UINT8 checksum;
UINT16 *head;

checksum = (((head[0]>>8)^head[0] ^
            (head[1]>>8)^head[1] ^
            (head[2]>>8)^head[2] ^
            (head[3]>>8)^head[3] ^
            (head[4]>>8)^head[4] ^
            (head[5]>>8)^head[5] ^
            (head[6]>>8)^head[6] ^
            (head[7]>>8))^0xff) & 0xff;
```

Address Table

The address table contains information that links the range selections to the calibration data. The address table can be anywhere in the first 256 words of the EEPROM, although it is likely to follow the header. Its location is given by the second word of the header.

The address table comprises a number of eight-word entries, arranged as a list. Each entry identifies the location of the calibration data for one range for either the ADC or DAC section, encoded as follows. The first word given is lowest in memory:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Word
END	0	0	SCT	RANGE				0	CHANNEL COUNT							+00
DATA ADDRESS																+01
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	+02
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	+03
NOMINAL GAIN HIGH																+04
NOMINAL GAIN LOW																+05
NOMINAL OFFSET HIGH																+06
NOMINAL OFFSET LOW																+07

The END flag is set if this word marks the end of the list. Normal entries have it reset.

The SCT flag at bit 12 indicates the section that the entry is for. 0 means ADC, 1 means DAC.

The RANGE number at bits 11..8 is the encoded range selection that this entry describes. See the ADC Range Select Register descriptions for details.

The channel count in bits 6..0 gives the number of channels for which calibration data is supplied.

The data address gives the word address of the first word of the calibration table for this section and range.

The nominal gain and nominal offset represent the uncorrected *m* and *b* values for this range as 4-byte IEEE floats. A perfect device would have all its gain and offset data in the calibration tables equal to these values. *b* is normally zero, since few devices have offsets designed in.

For an ADC use the following equation to get the nominal voltage from the counts:

Nominal voltage = (counts * *m)* + *b*

Software can find out where the calibration table for a particular range setting is stored by searching the address table for a match with the SCT and RANGE bits. The two bits in positions 14 and 13 of the word should be matched to 00 as well, since future versions may store other information in this table. The end of the list is identified by the END bit set.

Calibration Tables

Calibration tables store the actual calibration coefficients. Each table stores the coefficients for all channels for a single range setting on either the ADC or the DAC. Each table has a corresponding address table entry that points to it.

Data is organized in the calibration tables as an array of pairs of 4-byte IEEE floating-point numbers. The first in the pair is the gain. The second is the offset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Word
GAIN HIGH																+00
GAIN LOW																+01
OFFSET HIGH																+02
OFFSET LOW																+03

Each four word coefficient pair corresponds to one channel. The first (lowest in memory) is for the lowest-numbered channel. The diagram below shows the word addresses of the first three coefficient pairs in a calibration table:

Channel 1	Gain	+00
Channel 1	Offset	+02
Channel 2	Gain	+04
Channel 2	Offset	+06
Channel 3	Gain	+08
Channel 3	Offset	+0c
etc.	etc.	

Figure 54 Calibration Coefficient Organization in Calibration Table

Example EEPROM Contents

The dump below shows the first part of the contents of an EEPROM as a hex dump. Throughout, numbering is in 16-bit words, not bytes:

```
Addr    0    1     2    3      4    5     6    7
----  ---- ----  ---- ----   ---- ----  ---- ----
0000  554c 0b08  0d0a f2f5   00b8 f47c  0000 00d5
0008  0028 0030  0000 0000   3b9f ffe5  0000 0000
0010  0128 00d0  0000 0000   3b1f ffe5  0000 0000
0018  0228 0170  0000 0000   3b1f ffe5  0000 0000
0020  0328 0210  0000 0000   3a9f ffe5  0000 0000
0028  8000 0000  0000 0000   0000 0000  0000 0000
0030  3b9f ed95  3b82 69bb   3b9f ed2f  3b74 b398
0038  3b9f d926  3bd4 9a46   3b9f f003  3b57 004a
0040  3b9f ea05  3b7b 52fd   3b9f e94b  3b8f b254
0048  3b9f ed15  3b80 2679   3b9f efb7  3b49 232d
0050  3b9f e5be  3c1c dc9f   3b9f e7fa  3c10 f436
0058  3b9f e24c  3c1e 1fed   3b9f ea99  3c05 a4b6
0060  3b9f eb17  3c02 b6ae   3b9f e5a0  3c1a db72
0068  3b9f ed80  3bff 8fb9   3b9f e346  3c1c 0a45
0070  3b9f ec58  3bff cedd   3b9f ea32  3bec 1171
0078  3b9f df36  3c17 2eaf   3b9f ea66  3bf6 9a4d
0080  3b9f ed60  3bdf 4c3a   3b9f e223  3c0e 71d9
0088  3b9f eb95  3bd7 edb5   3b9f e622  3c06 a5fa
0090  3b9f e90a  3bd4 9b20   3b9f e860  3bd0 0470
...
```

The information below is a decoded version of some of the hex data:

```
EEPROM header
0x0000: 0x554c       Sync bytes
0x0001: 0x0b08       Rev 0, size 2048 bytes, address table at 0x0008
0x0002: 0x0d0a       Endian bytes 0x0d0af2f5
0x0003: 0xf2f5
0x0004: 0x00b8       Serial number 12121212
0x0005: 0xf47c
0x0006: 0x0000       (spare)
0x0007: 0x00d5       Checksum 0xd5

Address table
0x0008: 0x0028       ADC range 0x0 (+/-10 V), 40 channels
0x0009: 0x0030       -> calibration table at 0x0030
0x000c: 0x3b9f       Nominal gain    0.00488280
0x000d: 0xffe5
0x000e: 0x0000       Nominal offset  0.00000000
0x000f: 0x0000
0x0010: 0x0128       ADC range 0x1 (0 to 10 V), 40 channels
0x0011: 0x00d0       -> calibration table at 0x00d0
0x0014: 0x3b1f       Nominal gain    0.00244140
0x0015: 0xffe5
0x0016: 0x0000       Nominal offset  0.00000000
0x0017: 0x0000
0x0018: 0x0228       ADC range 0x2 (+/-5 V), 40 channels
0x0019: 0x0170       -> calibration table at 0x0170
0x001c: 0x3b1f       Nominal gain    0.00244140
0x001d: 0xffe5
0x001e: 0x0000       Nominal offset  0.00000000
0x001f: 0x0000
0x0020: 0x0328       ADC range 0x3 (0 to 5 V), 40 channels
0x0021: 0x0210       -> calibration table at 0x0210
0x0024: 0x3a9f       Nominal gain    0.00122070
0x0025: 0xffe5
0x0026: 0x0000       Nominal offset  0.00000000
0x0027: 0x0000
0x0028: 0x8000       End of address table

ADC range +/-10 V
0x0030: 0x3b9f       Channel  1: Gain    0.00488062
0x0031: 0xed95
0x0032: 0x3b82       Channel  1: Offset  0.00397989
0x0033: 0x69bb
0x0034: 0x3b9f       Channel  2: Gain    0.00488057
0x0035: 0xed2f
0x0036: 0x3b74       Channel  2: Offset  0.00373385
0x0037: 0xb398
0x0038: 0x3b9f       Channel  3: Gain    0.00487818
0x0039: 0xd926
```

```
0x003a: 0x3bd4       Channel  3: Offset  0.00648812
0x003b: 0x9a46
0x003c: 0x3b9f       Channel  4: Gain    0.00488091
0x003d: 0xf003
0x003e: 0x3b57       Channel  4: Offset  0.00328066
0x003f: 0x004a
0x0040: 0x3b9f       Channel  5: Gain    0.00488019
...
```

Applying Calibration Coefficients

For corrected data, use the following equation to convert ADC counts to volts. This is normally applied to data read from the ADC. *G* is the gain coefficient for the channel and *j* the offset coefficient for the channel. The coefficients include multipliers for the selected range:

$$V = (G \times C) + j$$

See the section on analog input for more details and example code.

Calibration Method

The ADCs are calibrated by averaging 500 readings at approximately fifteen known input voltage points between 85% negative full-scale and 85% positive full-scale. A linear curve fit produces gain and offset coefficients. This is repeated for each channel and range combination.

Ready Status

Overview

The IP-UL-ADC40 has a Ready bit in the CGSR (+02). This shows whether the IndustryPack is ready for operation or not. It can be used to generate an interrupt on either assertion or negation.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	ST2	ST1	0	IP32	**HWE**	**RDY**	0	IRQP	Read
X	X	X	X	X	X	X	X	X	X	X	X	**HWE**	**X**	X	X	Write
0	0	0	0	0	0	0	0	ST2	ST1	0	IP32	**0**	**1**	0	0	Reset

Figure 52 Ready and Hardware Error Bits in the Common General Status Register

The Ready bit is normally one, but is set to zero if a hardware error is detected.

Hardware Errors

Hardware error detection circuitry and logic on the IndustryPack can detect missing power supplies to any of the ADC converters and converters that fail to respond to conversion requests. A hardware error condition is set if any of the possible sources is in error, indicated by the Hardware Error (HWE) bit.

Once the hardware error condition is no longer present, software can reset the HWE bit by writing it with a one. Writes of zero have no effect. Once the hardware error condition is removed, the IndustryPack goes Ready again.

Generating Interrupts

Interrupts can be generated by HWE being asserted and by either state of RDY. The polarity bit in the CICR (+04) is programmable, allowing the IRQ latch to be set by either RDY asserted or RDY negated states. LRDY set to one selects interrupt on negation:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	**LRDY**	0	1	EST2	EST1	0	0	EHWE	ERDY	0	0	Read
X	X	X	X	X	**X**	X	X	EST2	EST1	X	X	EHWE	ERDY	X	X	Write
0	0	0	0	0	**1**	0	1	0	0	0	0	0	0	0	0	Reset

Figure 55 Ready Interrupt Polarity Control Bit in the Common IRQ Control Register

The Hardware Error and Ready status bits are also reflected in the CISR (+06). The pending bits are latched version of the polarity-controlled data bits:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
DST2	DST1	0	0	**DHWE**	**DRDY**	0	0	PST2	PST1	0	0	**PHWE**	**PRDY**	0	0	Read
X	X	X	X	**X**	**X**	X	X	PST2	PST1	X	X	**PHWE**	**PRDY**	X	X	Write
0	0	0	0	**0**	**0**	0	0	0	0	0	0	**0**	**0**	0	0	Reset

Figure 56 Hardware Error and Ready IRQ Status in the Common IRQ Status Register

User-Selectable Options

Shunt Overview

IP-UL-ADC40 has one shunt that controls the analog supply voltage. It is shown in the table and diagram below:

Shunt	Description
E1	Positive analog power selection

Figure 57 Shunt Descriptions

E1

This shunt controls the source of power for the analog circuitry on the IndustryPack.

E1	Description
1=2	Internal IP Bus +12 V (default)
2=3	External I/O connector +15 V

Figure 58 Shunt E1 settings

The internal supply is likely to be more noisy than a high-quality external supply. If the power supply is suspected as a source of noise, and external supply should be considered.

Fuses

The external supply rail is protected by a fuse located next to shunt E1. It can be tested with an Ohmmeter. If the fuse is blown, there will be no analog power to the converters. This will cause the board to indicate a hardware error, reflected in the HWE bit in the Common General Status Register.

Modifying the board, including replacing blown fuses will void the warranty. Contact SBS GreenSpring Modular I/O for an RMA number before shipping it back for repair.

Troubleshooting

The sections following describe possible problems and methods of tracing their sources.

Hardware Errors

Hardware errors are flagged if analog power is not present at any of the converters, or if any of the converters times out. A hardware error may be due to a component failure, or to the analog power not being present.

Fuses

The external power supply is fused. If this is blown and the external supplied used, then the analog section will not work. The HWE bit in the CGSR (+02) will be set to indicate a hardware error. See the section on User Settable Options for more information.

Shunts

Shunt E1 selects between internal and external analog power source. If the shunt is set incorrectly, or an external supply is selected, but not connected or turned on, the analog section will not work. The HWE bit in the CGSR (+02) will be set to indicate a hardware error. See the section on User Settable Options for more information.

Timer clock selection

If the timer does not run as expected, it is possible that its clock has not been selected properly. Failure to set the EN bit will also prevent the timer from working. Rapid, repeated reloads will also prevent the timer from reaching zero.

Interrupts

If interrupts do not occur when expected, it is possible that the section generating the interrupt has not been enabled to do so. Unexpected interrupts with no handler installed are probably due to the wrong section being enabled, or to the wrong vector being used to install the handler into the OS.

Triggers

In order for a trigger to start a conversion, several things must be set correctly: The trigger source must be selected and connected properly (TSS bits), the trigger must be enabled (TEN bit), and the trigger must be either automatically reset (DTR bit) or manually reset (GOP bits). The Strobe pins can be used to check whether the trigger is occurring when expected.

Modes

A section set to the wrong mode will not behave as expected. In Single-Shot mode, triggers are required to start conversions. In Channel Subset mode, ACCRL and DCCRL control the converters. To rule out incorrect mode settings, set the section into All Channels, Continuous modes and see if the expected data is present.

EEPROM and Calibration

Use the examples given to figure out whether the binary data is being read correctly from the EEPROM. Once the binary data is available, the only error is to interpret it incorrectly. Little-endian systems will have to adjust the word order for reading the floating point numbers.

I/O Connector Signals

Input and output of all external signals except Strobe2 is carried out through the 50-pin I/O connector. The pinout of the I/O connector is shown below:

Pin	IP-UL-ADC40
1	ADC1
2	ADC2
3	ADC3
4	ADC4
5	ADC5
6	ADC6
7	ADC7
8	ADC8
9	ADC9
10	ADC10
11	ADC11
12	ADC12
13	ADC13
14	ADC14
15	ADC15
16	ADC16
17	ADC17
18	ADC18
19	ADC19
20	ADC20
21	ADC21
22	ADC22
23	ADC23
24	ADC24
25	ADC25
26	ADC26
27	ADC27
28	ADC28
29	ADC29
30	ADC30
31	ADC31
32	ADC32

Pin	IP-UL-ADC40
33	ADC33
34	ADC34
35	ADC35
36	ADC36
37	ADC37
38	ADC38
39	ADC39
40	ADC40
41	AG
42	Strobe1
43	reserved
44	reserved
45	DG
46	reserved
47	DG
48	0 V in
49	+15 V in
50	n/c

AG Analog Ground

DG Digital Ground

n/c No connection

Do not make connections to pins marked “reserved”. Damage to the IndustryPack may occur.

ADC*n*

The analog to digital circuitry requires a single-ended input. This means that the measured voltage is the potential difference between the ADC*n* pin and analog ground. If the ADC*n* pin is at a positive voltage with respect to ground, then a positive number is returned by the converter.

The table below shows the absolute maximum limits on the inputs and the input impedances. The absolute voltage limit is measure with respect to analog ground.

Range	Parameter	IP-UL-ADC40
0-5 V	Input impedance	21 KΩ
	Lower absolute voltage limit	-15 V
	Upper absolute voltage limit	+15 V
0-10 V	Input impedance	21 KΩ
	Lower absolute voltage limit	-15 V
	Upper absolute voltage limit	+15 V
±5 V	Input impedance	16 KΩ
	Lower absolute voltage limit	-15 V
	Upper absolute voltage limit	+15 V
±10 V	Input impedance	16 KΩ
	Lower absolute voltage limit	-15 V
	Upper absolute voltage limit	+15 V

Figure 59 Analog Input Channel Parameters

AG

The analog ground pins are all connected to a common ground plane on the PCB. Use the AG pins for all analog signals. The 0 V in pin can also be used for analog ground.

DG

The digital ground pins are connected to analog ground at a single point on the PCB. The Strobe signals are the only signals that should be referenced to digital ground.

+15 V in, 0 V in

These pins are for the connection of an optional power source for the analog circuitry. In order to use an external source, shunt E1 must be in the 2=3 position. As shipped, the IndustryPack uses the ±12 V supply from the IP bus connector. A high-quality external supply may be preferred, since this can carry much less noise.

0 V in can also be used as an analog ground.

The IndustryPack is protected by a 0.5 A fuse on the external power source pin.

The current requirements for the external power is shown below:

Rail	Current (mA)
+15 V	50

Figure 60 External Power Supply Current

Power requirements for the carrier board are shown in the specifications at the end of this manual.

Strobe

The Strobe signals need a pulse at least 250 µs long. They output 250 µs pulses. Voltage levels are TTL.

Reserved pins

Reserved pins should not be used. Make no connections to these pins. Failure to follow this directive could damage the IndustryPack.

IP Bus Connector Signals

IP Bus connector signals are shown below. Signals that are not connected are marked as "n/c". Signals that are connected but not used are marked as "Reserved". See the IndustryPack logic specification for more information. The Strobe2 signal is carried on the pin marked STROBE*, pin 46.

GND	GND	1		26	
CLK	+5V		2		27
Reset*	R/W*	3		28	
D0	IDSEL*		4		29
D1	Reserved	5		30	
D2	MEMSEL*		6		31
D3	Reserved	7		32	
D4	INTSEL*		8		33
D5	Reserved	9		34	
D6	IOSEL*		10		35
D7	Reserved	11		36	
D8	A1		12		37
D9	Reserved	13		38	
D10	A2		14		39
D11	Reserved	15		40	
D12	A3		16		41
D13	IRQREQ0*	17		42	
D14	A4		18		43
D15	Reserved	19		44	
BS0*	A5		20		45
BS1*	STROBE*	21		46	
-12V	A6		22		47
+12V	ACK*	23		48	
+5V	n/c		24		49
GND	GND	25		50	

The layout of the pin numbers in this table corresponds to the physical placement of pins on the IP connector. Thus this table may be used to easily locate the physical pin corresponding to a desired signal. Pin 1 is marked with a square pad on the IndustryPack.

Register Description Reference

This section describes the registers in detail. Follow the guidelines below:

- Access registers as 16-bit words only. Byte accesses will bus trap. Long-word accesses to registers are not supported by all IndustryPack carriers, the MVME162 being a notable example.
- All bits have a write function, a read function, or they are read/write. This makes it safe to read a register, modify it, and write it back.
- Always write zeroes to unused locations, even if they are "don't care". Zeroes will ensure compatibility with other Unilin products and upgrades to this product.
- Don't use bitwise operators directly on registers. This can lead to compilers creating the 68000 `btst`, `bset`, and `bclr` instructions which use byte accesses and consequently bus trap. Instead, read a register, modify it, and write it back, each time with a separate C expression.

Each register described has a page to itself, with a header similar to the example below:

Common Example Test Register CETR +78

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
RW BITS				RO	0	0	0	0	0	0	0	XYZ				Read
RW BITS				X	WO	X	X	X	XOP			X	X	X	X	Write
0	0	0	1	0	0	0	0	0	0	0	0	XYZ				Reset

The title shows the name of the register, its abbreviation, and its offset when addressed via the VME, ISA, or PCI busses.

The first line of the register layout shows the bit number, least significant on the right. The next line shows the read function of each bit. The next line shows the write function of each bit. The last line shows the state following a reset.

In the example shown, the bits marked RW BITS are read/write. Whatever is written to them can be read back. The name in the Read row is the same as the name in the Write row. The reset status of these bits is 0001.

The bit in location 11, RO, is read-only. The X indicates that whatever is written is ignored.

Bit 10, WO, is write-only. It always reads zero. Bits 6..4 are also write-only. Every read returns 000 from them.

Bits 3..0 are read-only, but their reset state is given as XYZ. If the reset state is named then the state is not expected to change, but it does depend on some hardware setting. Bits connected to shunts are a good example of read-only bits of this type.

The register name and layout is followed by a header for each bit or group of bits, as in the following example:

15..12	Read Write Bits	RW BITS	R/W

A description of the bits follows the header. Bits with no function are not described.

Common General Control Register CGCR +00

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	IRQE	Read
X	X	X	X	X	X	X	X	ST2	ST1	X	X	X	0	0	IRQE	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The CGSR controls aspects of the IndustryPack that are not specific to either the ADC or DAC sections.

7	Strobe 2	ST2	W

If written with a one, issues a pulse to the external Strobe2 pin. The pulse is a minimum of 250 ns long. If written with zero, nothing happens.

6	Strobe 1	ST1	W

If written with a one, issues a pulse to the external Strobe2 pin. The pulse is a minimum of 250 ns long. If written with zero, nothing happens.

0	Common Section IRQ Enable	IRQE	R/W

While IRQE is asserted, interrupts from the common section can assert the IRQ0* pin on the IP bus connector. While IRQE is negated, such interrupts are masked. CGSR:IRQP cannot be asserted while IRQE is negated.

This bit does not affect interrupts from either the ADC or DAC sections.

See the section on interrupts for more details.

Common General Status Register CGSR +02

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	ST2	ST1	0	IP32	HWE	RDY	0	IRQP	Read
X	X	X	X	X	X	X	X	X	X	X	X	HWE	X	X	X	Write
0	0	0	0	0	0	0	0	ST2	ST1	0	IP32	0	1	0	0	Reset

The CGSR shows the status of parts of the IndustryPack that are not specific to either the ADC or DAC sections.

7	Strobe 2 State	ST2	R

This bit is a one if the external Strobe 2 pin is asserted (low voltage). It is zero otherwise. This bit shows the instantaneous state of the pin and cannot be used to detect pulses.

6	Strobe 1 State	ST1	R

This bit is a one if the external Strobe 1 pin is asserted (low voltage). It is zero otherwise. This bit shows the instantaneous state of the pin and cannot be used to detect pulses.

4	32 MHz IP Bus Clock	IP32	R

This bit is one if the IndustryPack is plugged into a 32 MHz bus It is zero if the IndustryPack is plugged into an 8 MHz bus.

3	Hardware Error	HWE	R/W

This bit is one if the IndustryPack has detected a hardware error. It is zero otherwise. A hardware error may be generated if the analog section supply is missing or below 12V, if one of the regulators fails, or if one of the converters does not complete a conversion. A hardware error will result in the IndustryPack going not Ready. See the description of the Ready bit below.

To reset this bit, write it with a one. Writing it with a zero has no effect. If the source of the error is still present, the bit will be set again immediately. See the section on Error States for more information.

2	Ready	RDY	R

This bit is one if the IndustryPack is ready. It is zero otherwise. The IndustryPack will be ready as long as any detected hardware errors have been acknowledged by writing the HWE bit with a one.

0	Common Section IRQ Pending	IRQP	R

IRQP asserted indicates that the Common section has its interrupts enabled and is asserting the IRQ0* pin on the IP bus connector. IRQP negated indicates that the Common section is not contributing to the status of the IRQ0* pin. IRQP cannot be asserted while CGCR:IRQE is negated.

This bit does not reflect interrupts from either the ADC or DAC sections.

See the section on interrupts for more details.

Common Interrupt Control Register CICR +04

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	LRDY	0	1	EST2	EST1	0	0	EHWE	ERDY	0	0	Read
X	X	X	X	X	X	X	X	EST2	EST1	X	X	EHWE	ERDY	X	X	Write
0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	Reset

The CICR controls the individual interrupt sources in the Common section.

10	Polarity Ready IRQ	LRDY	R/W

When zero selects the assertion of the Ready Data bit as setting the Ready IRQ latch. When one, selects the negation of the Ready Data bit as setting the Ready IRQ latch. This bit can be programmed by interrupt routines so that interrupts can be generated on both transitions.

7	Enable Strobe2 IRQ	EST2	R/W

When zero, prevents the state of the Pending Strobe2 bit from contributing to the ADC interrupt. When one, permits the state of the Pending Strobe2 bit to contribute to the ADC interrupt.

6	Enable Strobe1 IRQ	EST1	R/W

When zero, prevents the state of the Pending Strobe1 bit from contributing to the ADC interrupt. When one, permits the state of the Pending Strobe1 bit to contribute to the ADC interrupt.

3	Enable Hardware Error IRQ	EHWE	R/W

When zero, prevents the state of the Pending Hardware Error bit from contributing to the ADC interrupt. When one, permits the state of the Pending Hardware Error bit to contribute to the ADC interrupt.

2	Enable Ready IRQ	ERDY	R/W

When zero, prevents the state of the Pending Ready bit from contributing to the ADC interrupt. When one, permits the state of the Pending Ready bit to contribute to the ADC interrupt

Common Interrupt Status Register CISR +06

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
DST2	DST1	0	0	DHWE	DRDY	0	0	PST2	PST1	0	0	PHWE	PRDY	0	0	Read
X	X	X	X	X	X	X	X	PST2	PST1	X	X	PHWE	PRDY	X	X	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The CISR give access to the flip-flops for the individual interrupt sources in the Common section.

15	Data Strobe2 IRQ	DST2	R

Pulsed to a one briefly when the Strobe2 signal in the Common General Status Register is asserted. The pulse is so brief as to be invisible to software, therefore always reads as zero. The pulse is sufficient to set the Strobe2 interrupt flip-flop, so the Pending Strobe2 bit may be used to detect assertions of the Strobe2 signal.

14	Data Strobe1 IRQ	DST1	R

Pulsed to a one briefly when the Strobe1 signal in the Common General Status Register is asserted. The pulse is so brief as to be invisible to software, therefore always reads as zero. The pulse is sufficient to set the Strobe1 interrupt flip-flop, so the Pending Strobe1 bit may be used to detect assertions of the Strobe1 signal.

11	Data Hardware Error IRQ	DHWE	R

Reflects the state of the Hardware Error signal in the Common General Status Register. While one, the Hardware Error IRQ latch is repeatedly set.

10	Data Ready IRQ	DRDY	R

Reflects the state of the Ready signal in the Common General Status Register. While one, the Ready IRQ latch is repeatedly set.

7	Pending Strobe2 IRQ	PST2	R/W

Reflects the state of the Strobe2 interrupt flip-flop, where one indicates an interrupt condition. Since the polarity of this interrupt is hard-wired to 0, and the data source is a pulse, this bit is set once each time the Strobe2 signal is asserted. If written with one, the flip-flop is reset. If written with zero, nothing happens.

7	Pending Strobe1 IRQ	PST1	R/W

Reflects the state of the Strobe1 interrupt flip-flop, where one indicates an interrupt condition. Since the polarity of this interrupt is hard-wired to 0, and the data source is a pulse, this bit is set once each time the Strobe1 signal is asserted. If written with one, the flip-flop is reset. If written with zero, nothing happens.

3	Pending Hardware Error IRQ	PHWE	R/W

Reflects the state of the Hardware Error interrupt flip-flop, where one indicates an interrupt condition. Since the data source is a level, this bit is set repeatedly as long as the data source is asserted. If written with one with the source negated, the flip-flop is reset. If written with zero, nothing happens.

2	Pending Ready IRQ	PRDY	R/W

Reflects the state of the Ready interrupt flip-flop, where one indicates an interrupt condition. Since the data source is a level, this bit is set repeatedly as long as the data source is one (asserted or negated, as selected by the polarity bit). If written with one with the source zero one (negated or asserted, as selected by the polarity bit), the flip-flop is reset. If written with zero, nothing happens.

Common Interrupt Vector Register CIVR +0a

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
TEST USE								VECTOR						SOURCE		Read
TEST USE								VECTOR						X	X	Write
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	Reset

The CIVR supplies a user-programmable vector during interrupt acknowledge cycles.

Following a reset the register reads $000F, the Motorola Uninitialized Interrupt vector. All sixteen bits are read/write. Following the first interrupt, the lower two bits encode the interrupt source and are read-only.

15..8	Test Bits	TEST USE	R/W

The Test Bits are read/write. They have no internal function and may be used to help test the data bus.

7..2	Vector	VECTOR	R/W

The Vector bits supply the upper six bits of the interrupt vector to the CPU during an interrupt acknowledge cycle. The lower two bits of the vector encode the interrupt source, so are not under the programmer's control.

1..0	Interrupt Source	SOURCE	R

Prior to the first interrupt, the Interrupt Source bits are read/write bits. Following the first interrupt, the Interrupt Source bits encode the interrupt source, as below:

SOURCE	Interrupt Source
00	Common (highest priority)
01	(unused)
10	ADC
11	DAC (lowest priority)

A source with a higher priority will those with lower priorities, resulting in a different vector. The Common section of this IndustryPack has no interrupts, so source 00 will never be encoded.

See the section on interrupts for more details.

Common Configuration Control and Status Register CCCSR +10

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	SIZE					0	0	0	0	0	0	0	CE	Read
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	CE	Write
0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	Reset

The CCCSR provides access to the on-board EEPROM. The EEPROM stores configuration information, notably calibration coefficients.

12..8	Configuration Area Size	SIZE	R

The SIZE field encodes the size of the storage device in bytes. It is hard-wired to 01011, binary for 11, indicating that there are 2^{11} bytes of storage in the EEPROM.

0	Configuration Enable	CE	R/W

CE controls the start and end of communication with the serial EEPROM. Each command starts with CE asserted and ends with CE negated. CE may be negated at any time during a command to abort the command. Each command may pass sixteen bits of data.

See the section on Configuration Data for more detailed information.

Common Configuration Data Register CCDR +12

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	CD	Read
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	CD	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The CCDR transfers data to and from the on-board EEPROM. The EEPROM stores configuration information, notably calibration coefficients. Whenever the register is written or read, on-board logic generates a clock signal with the correct timing for the EEPROM, so no delays are necessary. Accesses to the data register take one microsecond each due to the timing requirements of the EEPROM device.

0	Configuration Data	CD	R/W

When written, the data in the CD bit is passed to the EEPROM. When read, data is passed from the EEPROM to the CPU. Used in conjunction with CCCSR:CE, the CD bit passes commands, addresses, and data to accomplish reading, writing, locking, and unlocking of the EEPROM.

During write operations, the CD bit reflects the EEPROM busy status. The EEPROM is busy if CD is negated.

See the section on Configuration Data for more detailed information.

Common Test Address Register CTAR +1c

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Read
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The CTAR gives access to test functions built into the Industry Pack.

For normal operation this register must not be changed from its default value of 0x0000.

Common Test Data Register CTDR +1e

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Read
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The CTDR gives access to test functions built into the Industry Pack.

This register works in conjunction with the CTAR to provide software access to test functions built into the IndustryPack.

For normal operation this register must not be changed from its default value of 0x0000.

ADC General Control Register AGCR +20

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	STS2		STS1		0	0	0	0	0	CS	SS	IRQE	Read
X	X	X	X	STS2		STS1		X	X	X	X	X	CS	SS	IRQE	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The AGCR controls parts of the IndustryPack that are specific to the ADC section.

11..10	Strobe2 Select	STS2	R/W

These two bits control the automatic generation of pulses on the external Strobe2 signal by the ADC timer, ADC trigger, and ADC conversions. The bits select the source as below:

STS	Pulse Selection
00	Nothing (default)
01	ADC Trigger asserted
10	ADC SU asserted
11	ADC Timer became zero

Once pulse of approximately 250 ns duration is output each time the condition becomes true.

9..8	Strobe1 Select	STS1	R/W

These two bits control the automatic generation of pulses on the external Strobe1 signal by the ADC timer, ADC trigger, and ADC conversions. The bits select the source as for Strobe2.

2	Channel Subset	CS	R/W

While CS is one, only data from converters enabled by bits in the ADC Converter Control Register Low is written into the ADC DPRAM. Disabled converters produce no data and the multiplexers are not swept across all the channels. Up to five channels can be converted. This is Channel Subset mode.

While CS is zero (the reset state), data from all converters is written into the ADC DPRAM regardless of the contents of the ACCRL. The multiplexer is swept across all inputs to convert all forty channels. This is All Channels mode.

1	Single-Shot	SS	R/W

While SS is one, ADC conversions can be started only by the assertion of the ADC trigger. This is Single-Shot mode. While SS is zero, conversions occur continuously, regardless of the state of the ADC trigger. This is Continuous mode. Switching modes automatically starts or stops conversions.

0	ADC Section IRQ Enable	IRQE	R/W

While IRQE is asserted, interrupts from the ADC section can assert the IRQ0* pin on the IP bus connector. While IRQE is negated, such interrupts are masked. AGSR:IRQP cannot be asserted while IRQE is negated.

This bit does not affect interrupts from either the Common or DAC sections.

See the section on interrupts for more details.

ADC General Status Register AGSR +22

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	DPR	0	WIDTH					DA	SU	0	0	0	0	TS	IRQP	Read
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	Write
0	0	0	0	1	1	0	0	DA	SU	0	0	0	0	TS	0	Reset

The AGSR shows the status of parts of the IndustryPack that are specific to the ADC section.

14	Dual Port RAM	DPR	R

This bit is one if DPRAM exists in the ADC section, zero otherwise. This IndustryPack has no DPRAM in the ADC section, so this bit is hard-wired to zero.

12..8	ADC Data Width	WIDTH	R

The ADC Data Width expresses the width of the ADC data in bits. This IndustryPack reads a constant binary 01100: twelve.

7	Data Access	DA	R

Data Access is asserted when the last ADC in the conversion cycle has written its data into the FIFO and negated when the first of the next cycle is about to do so. This signal may be very short indeed and may never be visible to a CPU in this location. For this reason a latched version is available in the AISR. DA is not generated when all converters are disabled by the ACCRL when the ADC is in Channel Subset mode.

See the section on analog input and output for more information.

6	Sample Update	SU	R

Sample Update is asserted when the first ADC input in the conversion set samples the input data and negated after the last. It represents the time during which inputs are being sampled. This signal may be very short indeed and may never be visible to a CPU in this location. For this reason a latched version is available in the AISR. SU is not generated when all converters are disabled by the ACCRL when the ADC is in Channel Subset mode.

See the section on analog input and output for more information.

1	Trigger State	TS	R

This bit is one if the ADC Trigger is asserted, zero otherwise. It identical to the TS bit in the ATSR.

0	ADC Section IRQ Pending	IRQP	R

IRQP asserted indicates that the ADC section has its interrupts enabled and is asserting the IRQ0* pin on the IP bus connector. IRQP negated indicates that the ADC section is not contributing to the status of the IRQ0* pin. IRQP cannot be asserted while AGCR:IRQE is negated.

This bit does not reflect interrupts from either the Common or DAC sections.

See the section on interrupts for more details.

ADC IRQ Control Register AICR +24

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	EDA	ESU	0	0	0	ETIM	ETRG	0	Read
X	X	X	X	X	X	X	X	EDA	ESU	X	X	X	ETIM	ETRG	X	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The AICR controls the individual interrupt sources in the ADC section.

For more information, see the sections on analog input, the ADC timer, the ADC trigger, and interrupts.

7	Enable ADC DA IRQ	EDA	R/W

When zero, prevents the state of the Pending Data Access bit from contributing to the ADC interrupt. When one, permits the state of the Pending Data Access bit to contribute to the ADC interrupt.

6	Enable ADC SU IRQ	ESU	R/W

When zero, prevents the state of the Pending Sample Update bit from contributing to the ADC interrupt. When one, permits the state of the Pending Sample Update bit to contribute to the ADC interrupt.

2	Enable ADC Timer IRQ	ETIM	R/W

When zero, prevents the state of the Pending ADC Timer Became Zero bit from contributing to the ADC interrupt. When one, permits the state of the Pending ADC Timer Became Zero bit to contribute to the ADC interrupt.

1	Enable ADC Trigger IRQ	ETRG	R/W

When zero, prevents the state of the Pending ADC Trigger bit from contributing to the ADC interrupt. When one, permits the state of the Pending ADC Trigger bit to contribute to the ADC interrupt.

ADC IRQ Status Register AISR +26

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
DDA	DSU	0	0	0	DTIM	DTRG	0	PDA	PSU	0	0	0	PTIM	PTRG	0	Read
X	X	X	X	X	X	X	X	PDA	PSU	X	X	X	PTIM	PTRG	X	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The AISR gives access to the flip flops for the individual interrupt sources in the ADC section.

For more information, see the sections on analog input, the ADC timer, ADC trigger, and interrupts.

15	Data ADC Data Access IRQ	DDA	R

Pulsed to a one briefly when the DA signal in the ADC General Control and Status Register is asserted. The pulse is so brief as to be invisible to software, therefore always reads as zero. The pulse is sufficient to set the Data Access interrupt flip-flop, so the Pending ADC Data Access bit may be used to detect assertions of the DA signal.

DA is not generated when all converters are disabled by the ACCRL when the ADC is in Channel Subset mode.

14	Data ADC Sample Update IRQ	DSU	R

Pulsed to a one briefly when the SU signal in the ADC General Control and Status Register is asserted. The pulse is so brief as to be invisible to software, therefore always reads as zero. The pulse is sufficient to set the Sample Update interrupt flip-flop, so the Pending ADC Sample Update bit may be used to detect assertions of the SU signal.

SU is not generated when all converters are disabled by the ACCRL when the ADC is in Channel Subset mode.

10	Data ADC Timer IRQ	DTIM	R

Pulsed to a one briefly when the TZ signal in the ADC Timer Control and Status Register is asserted. The pulse is so brief as to be invisible to software, therefore always reads as zero. The pulse is sufficient to set the Timer interrupt flip-flop, so the Pending ADC Timer bit may be used to detect assertions of the TZ signal.

9	Data ADC Trigger IRQ	DTRG	R

Pulsed to a one briefly when the ADC Trigger is asserted. The pulse is so brief as to be invisible to software, therefore always reads as zero. The pulse is sufficient to set the Trigger interrupt flip-flop, so the Pending ADC Trigger bit may be used to detect assertions of the Trigger signal.

7	Pending ADC DA IRQ	PDA	R/W

Reflects the state of the ADC Data Access interrupt flip-flop, where one indicates an interrupt condition. Since the polarity of this interrupt is hard-wired to 0, and the data source is a pulse, this bit is set once each time the Data Access signal is asserted. If written with one, the flip-flop is reset. If written with zero, nothing happens.

6	Pending ADC SU IRQ	PSU	R/W

Reflects the state of the ADC Sample Update interrupt flip-flop, where one indicates an interrupt condition. Since the polarity of this interrupt is hard-wired to 0, and the data source is a pulse, this bit is set once each time the Sample Update signal is asserted. If written with one, the flip-flop is reset. If written with zero, nothing happens.

2	Pending ADC Timer IRQ	PTIM	R/W

Reflects the state of the ADC Timer interrupt flip-flop, where one indicates an interrupt condition. Since the polarity of this interrupt is hard-wired to 0, and the data source is a pulse, this bit is set once each time the Timer Equals Zero signal is asserted. If written with one, the flip-flop is reset. If written with zero, nothing happens.

1	Pending ADC Trigger IRQ	PTRG	R/W

Reflects the state of the ADC Trigger interrupt flip-flop, where one indicates an interrupt condition. Since the polarity of this interrupt is hard-wired to 0, and the data source is a pulse, this bit is set once each time the Trigger is asserted. If written with one, the flip-flop is reset. If written with zero, nothing happens.

ADC Converter Control Register Low ACCRL +2a

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	0	0	0	0	0	0	0	0	0	0	C5	C4	C3	C2	C1	Read
X	X	X	X	X	X	X	X	X	X	X	C5	C4	C3	C2	C1	Write
0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	Reset

The ACCRL controls the flow of data from individual ADC converters to the ADC DPRAM.

When the ADC section is operating in Single-Shot mode, the ACCRL controls the flow of data from individual ADC converters to the DPRAM. When operating in Continuous mode, this register is ignored.

1..0	Converter *n* Enable	C*n*	R/W

Each bit controls one converter: C*n* controls ADC*n*. If the bit is a one, data from the corresponding converter is enabled. If it is zero, it is disabled. Disabling all converters will prevent any data being written into the ADC DPRAM.

DA and SU are not generated when all converters are disabled by the ACCRL when the ADC is in Channel Subset mode.

Writes to the register assert the trigger if the TCE bit in the ATSR is one.

ADC Range Select Register ARSR +2c

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
RS4				RS3				RS2				RS1				Read
RS4				RS3				RS2				RS1				Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The ARSR selects the input ranges for different groups of channels.

15..12	Range Select 4	RS4	R/W

Selects the input range for channels 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, and 40. The default range is ±10 V.

11..8	Range Select 3	RS3	R/W

Selects the input range for channels 17, 18, 19, 20, 21, ,22, 23, and 24. The default range is ±10 V.

4..7	Range Select 2	RS2	R/W

Selects the input range for channels 9, 10, 11, 12, 13, 14, 15, and 16. The default range is ±10 V.

3..0	Range Select 1	RS1	R/W

Selects the input range for channels 1, 2, 3, 4 ,5 ,6, 7, and 8. The default range is ±10 V.

The ranges for all groups are selected as follows. Each group can be set independently:

RS1, 2, 3, 4	Range
0000	±10 V (default)
0001	0-10 V
0010	±5 V
0011	0-5 V

ADC Trigger Source Register ATSR +30

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
TSS				0	0	0	0	0	DTR	0	0	0	TCE	TS	TEN	Read
TSS				X	X	X	X	X	DTR	GOP		X	TCE	X	TEN	Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The ATSR controls the trigger in the ADC section. See the section about the trigger for more information.

15..12	Trigger Source Select	TSS	R/W

These four bits select from one of a number of hardware trigger sources, as given below:

TSS	Trigger source
0000	Nothing (default)
0001	(reserved)
0010	Strobe 1 signal
0011	Strobe 2 signal
0100	ADC Sample Update
0101	(reserved)
0110	ADC timer = zero
0111	(reserved)

TSS	Trigger source
1000	(reserved)
1001	(reserved)
1010	(reserved)
1011	(reserved)
1100	(reserved)
1101	(reserved)
1110	(reserved)
1111	(reserved)

The assertion of the selected signal asserts the ADC trigger.

6	Disable Trigger Reset	DTR	R/W

While this bit is one, the ADC trigger is not reset at the end of a conversion cycle. While this bit is zero, the end of a conversion cycle resets the trigger.

5..4	Trigger Operation	GOP	W

These two bits explicitly control the ADC trigger state, as below. They always read as zeros:

GOP	Operation
00	Nothing
01	Assert ADC trigger
10	Negate ADC trigger
11	(reserved)

Asserting the trigger will start a conversion if the ADC section is in Single-Shot mode. The trigger is normally self-resetting, but this behavior can be changed with the DTR bit.

2	Trigger on Converter Enable	TCE	R/W

While this bit is one, writes to the ACCRL assert the ADC trigger. If it is zero, writes to the ACCRL do not affect the trigger state. This feature can be used to enable converters and start a conversion with a single write.

1	Trigger State	TS	R

This bit shows the state of the ADC trigger. If it is one, the trigger is asserted. If it is zero, the trigger is negated. This bit is also available in the AGSR.

0	Trigger Enable	TEN	R/W

While this bit is one, assertion of the trigger starts a conversion cycle if the ADC section is in Single-Shot mode. While this bit is zero, trigger assertion does not start conversions.

ADC Timer Control and Status Register ATCSR +3c

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
RLSS				0	CSS			TZ	0	0	0	PSS		RPT	EN	Read
RLSS				X	CSS			X	X	TOP		PSS		RPT	EN	Write
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	Reset

The ATCSR controls the timer in the ADC section. See the section about the timer for more information.

15..12	Reload Source Select	RLSS	R/W

The Reload Source Select bits select a source for a reload signal, as shown below. If the selected signal is asserted, the timer's down-counter is reloaded:

RLSS	Reload source
0000	Nothing (default)
0001	(reserved)
0010	Strobe 1 signal
0011	Strobe 2 signal
0100	ADC Sample Update
0101	(reserved)
0110	ADC trigger
0111	(reserved)

RLSS	Reload source
1000	(reserved)
1001	(reserved)
1010	(reserved)
1011	(reserved)
1100	(reserved)
1101	(reserved)
1110	(reserved)
1111	(reserved)

10..8	Clock Source Select	CSS	R/W

By default, the clock source is a 1 MHz signal generated from the IndustryPack clock. The Clock Source Select (CSS) bits can choose other sources:

CSS	Clock source
000	1 MHz (default)
001	(reserved)
010	Strobe 1 signal
011	Strobe 2 signal
100	ADC Sample Update
101	(reserved)
110	(reserved)
111	(reserved)

The speed of the Sample Update signals depends on how often conversion cycles are performed.

7	Timer is Zero	TZ	R

When asserted, indicates that the down-counter is at zero. When negated, indicates that the down-counter is at zero. If the counter is not set to reload, this bit can be polled. If the Repeat bit is set then this bit will only pulse very briefly. The latched (pending) version available in the ADC Interrupt Status Register can be polled more reliably.

5..4	Timer Operation	TOP	W

The Timer Operation bits control explicit reset and reload of the prescaler and down-counter. The operations available via the TOP bits are shown below. The TOP bits always read as 00.

TOP	Operation
00	Nothing
01	Reset counter and reset prescaler
10	Reload counter and reset prescaler
11	Reset prescaler

3..2	Prescale Select	PSS	R/W

The prescaler divides the clock source by one of four fixed ratios. The default ratio is 1. The selection is made by the two Prescaler Select bits. The table below shows the settings:

PSS	Division Ratio
00	1 (default)
01	10
10	100
11	1000

1	Repeat	RPT	R/W

While negated, prevents the timer from reloading when it counts down to zero. While asserted, allows the timer to reload when it counts down to zero. The timer may be reloaded before that time by software via the TOP bits or by a reload source (see RLSS). If this bit is asserted while the timer is zero, it loads and starts immediately.

0	Enable Timer	EN	R/W

While negated, the timer is disabled. When asserted, the timer is enabled. The disabled timer simply has its clock removed. The timer may be re-enabled at any time. The source defaults to 1 MHz, derived from the IP clock.

ADC Timer Period Register ATPR +3e

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
PERIOD																Read
PERIOD																Write
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Reset

The ATPR stores the ADC timer period.

15..0	ADC Timer Period	PERIOD	R/W

The value in this register is loaded into the timer's down-counter whenever a reload is performed. This occurs when the Repeat is enabled and the counter reaches zero, when a reload is performed due to the assertion of a selected source, or when the timer operation includes a reload.

The timer period is the prescaled clock period divided by this number plus 1. The timer period resets to zero, corresponding to the maximum period of 65536 counts. Valid register values are 1 to 65535.

ADC MUX Select Register AMSR +4c

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Bit
0	CHANS							0	0	0	0	0	MUX			Read
X	X	X	X	X	X	X	X	X	X	X	X	X	MUX			Write
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	Reset

The AMSR selects input sources for the converters in the ADC section.

14..8	Number of Channels	CHANS	R

These seven bits give the number of channels in the ADC section. There are forty on this IndustryPack so this is hard-wired to 40 (binary 0101000).

2..0	Multiplexer	MUX	R/W

These three bits select the multiplexer setting for all five ADCs when the ADC section is in Channel Subset Mode. It is ignored in All Channels mode. See the section on Analog Input for more information about the settings for these bits.

Warranty and Repair

GreenSpring Computer warrants this product to be free from defects in workmanship and materials under normal use and service and in its original, unmodified condition, for a period of one year from the time of purchase. If the product is found to be defective within the terms of this warranty, GreenSpring Computer's sole responsibility shall be to repair, or at GreenSpring Computer's sole option to replace, the defective product. The product must be returned by the original customer, insured, and shipped prepaid to SBS GreenSpring Modular I/O. All replaced products become the sole property of SBS GreenSpring Modular I/O.

GreenSpring Computer's warranty of and liability for defective products is limited to that set forth herein. SBS GreenSpring Modular I/O disclaims and excludes all other product warranties and product liability, expressed or implied, including but not limited to any implied warranties of merchandisability or fitness for a particular purpose or use, liability for negligence in manufacture or shipment of product, liability for injury to persons or property, or for any incidental or consequential damages.

GreenSpring's products are not authorized for use as critical components in life support devices or systems without the express written approval of the president of SBS GreenSpring Modular I/O.

Service Policy

Before returning a product for repair, verify as well as possible that the suspected unit is at fault. Then call the Customer Service Department for a RETURN MATERIAL AUTHORIZATION (RMA) number. Carefully package the unit, in the original shipping carton if this is available, and ship prepaid and insured with the RMA number clearly written on the outside of the package. Include a return address and the telephone number of a technical contact. For out-of-warranty repairs, a purchase order for repair charges must accompany the return. SBS GreenSpring Modular I/O will not be responsible for damages due to improper packaging of returned items. For service on GreenSpring Products not purchased directly from SBS GreenSpring Modular I/O contact your reseller. Products returned to SBS GreenSpring Modular I/O for repair by other than the original customer will be treated as out-of-warranty.

Out of Warranty Repairs

Out of warranty repairs will be billed on a material and labor basis. The current minimum repair charge is $100. Customer approval will be obtained before repairing any item if the repair charges will exceed one half of the quantity one list price for that unit. Return transportation and insurance will be billed as part of the repair and is in addition to the minimum charge.

For Service Contact:

Customer Service Department
SBS GreenSpring Modular I/O
181 Constitution Drive
Menlo Park, CA 94025
(650) 327-1200
FAX: (650) 327-3808
Internet Address support@greenspring.com

Specifications

General

IP Type	Type II, single high
Dimensions	1.800 x 3.900 x 0.340 inches maximum
Conformance	ANSI/VITA 4-1995 IP Module Specification
Factory Calibration	Coefficients include gain and offset for all channel and range combinations
Power Requirements	+5 V @ 100 mA +12 V @ 135 mA -12 V not required

ADC

Number of ADC Channels	40
Resolution	12-bit
Throughput	100 kSample/sec max, on each of five channels simultaneously
On-board Conversion Rates	Software programmable
Full Scale Analog Inputs	0-5 V, 0-10 V, ±5 V, and ±10 V nominal, single ended
Input Impedance	16 kΩ Bipolar, 21 kΩ Unipolar
Overvoltage Protection	Up to ±16.5 V

Environmental

Operating Temp Range	0°C to 70°C, 5% to 95% relative humidity, non-condensing
Storage Temp Range	–10°C to +85°C, non-condensing

"Typical" characteristics are measured at room temperature, and are from component vendor data sheets. "Nominal" characteristics will have typical unit to unit variation of a few percent from stated values.

Order Information

IP-UL-ADC40	40 Channel, 12-bit, 100 kHz A/D IndustryPack
NTDRV-IP-UL-ADC40	Windows NT driver. Binary. Works with GS PCI carriers only. Call for availability.
VXDRV-IP-UL-ADC40	VxWorks driver. Binary. For MVME162 and GS VME carriers only. Call for availability.
ENG KIT-IP-UL-ADC40	Includes: • Terminal block • Cable • Technical documentation

Order Engineering Kit with first purchase.

Schematics

Schematics are provided here for customer *reference only*. This information was current at the time the printed circuit board was last revised. This revision letter is shown on the front of this manual as "Corresponding Hardware Revision." This information is not necessarily current or complete manufacturing data, nor is it part of the product specification. All information following is Copyright SBS GreenSpring Modular I/O.

Current manufacturing information, including schematics, programmed device listings, bills-of-material, and assembly diagrams are available from SBS GreenSpring Modular I/O as part of the Engineering Kit option or from your international distributor.