VMIVME-1160A
32-bit Optically Coupled Digital Input Board with Change-of-State Detection

Product Manual

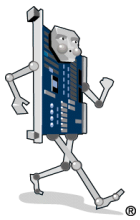
A GE Fanuc Company

12090 South Memorial Parkway
Huntsville, Alabama 35803-3308, USA
(256) 880-0444 ◆ (800) 322-3616 ◆ Fax: (256) 882-0859

500-101160-000 Rev. D

12090 South Memorial Parkway
Huntsville, Alabama 35803-3308, USA
(256) 880-0444 ◆ (800) 322-3616 ◆ Fax: (256) 882-0859

COPYRIGHT AND TRADEMARKS



(I/O man figure)

(IOWorks man figure)

12090 South Memorial Parkway
Huntsville, Alabama 35803-3308, USA
(256) 880-0444 ◆ (800) 322-3616 ◆ Fax: (256) 882-0859

Table of Contents

List of Figures

List of Tables

Overview

Contents

Introduction

The VMIVME-1160A is designed with standard Change-of-State (COS) control and interrupt logic that detects any COS and provides an interrupt vector to the byte level. It incorporates an Epic Ei68C153 Bus Interrupter Module (BIM) supporting interrupts on any of seven levels.

Each byte (8 bits) of input may have a unique interrupt vector that is generated upon a COS in any bit of that byte. The board also has an Interrupt Enable Register which allows interrupts to be enabled on a byte-by-byte basis. The input data may be accessed as a D8 or D16 transfer.

A functional block diagram of this product is shown in Figure 1 on page 12. Interrupts are generated on any COS (positive or negative transition). Polarity is determined by reading the input port after the COS interrupt.

NOTE: State changes that occur during the interrupt processing window (internal request to interrupt acknowledge cycle complete) will not be detected. The time between user input state changes must not be less than the computer interrupt processing time; otherwise, the state changes will be lost.

A Change-of-State Application Guide that describes the complete COSMODULE™ product line, VMIC's Document No. 825-000000-002, is available from VMIC. A summary of the COSMODULE™ product line is provided in Table 1, “Cosmodule™ Product Line Summary,” on page 13 for a list of reference.

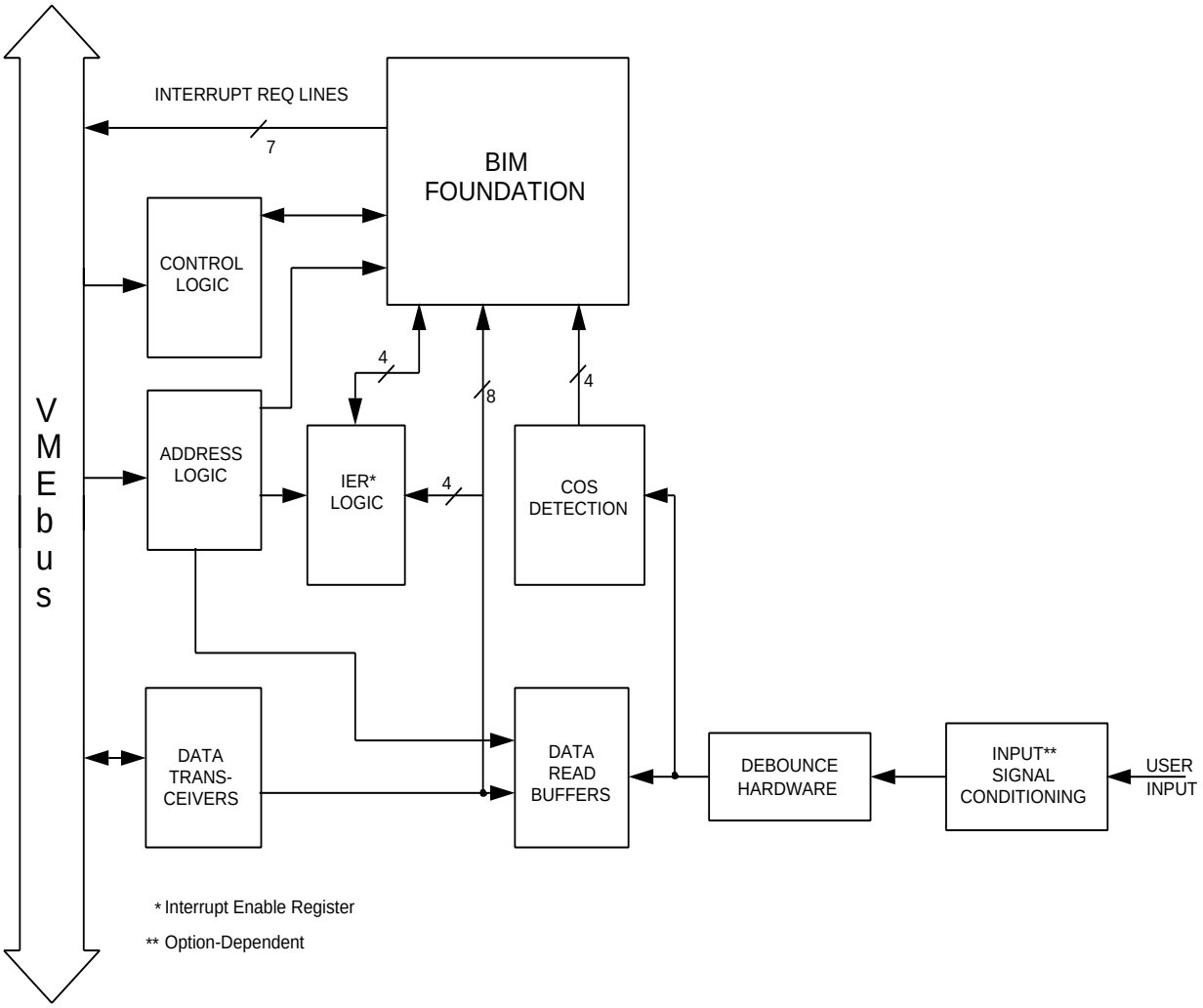


Figure 1 Typical COSMODULE™ Functional Block Diagram

Functional Description

The VMIVME-1160A provides 32 high-voltage, optically coupled digital inputs with change-of-state detection and vectoring to the byte level. The interrupt functions are supported by the BIM. The major features of the VMIVME-1160A are:

- Quad 8-bit ports
- Change-of-State port identified with interrupt vector
- Voltage sourcing or contact sensing signal conditioning
- Double-height Eurocard form factor with front panel
- 8- or 16-bit data transfers
- 64-pin DIN type input connector
- Jumper-selectable nonprivileged short I/O, supervisory short I/O, or both
- 32 optically coupled inputs

Table 1 COSMODULE™ Product Line Summary

Item No.	Description	Model No.	Transfer Type
1.	16-Channel AC or DC High Voltage (5 V to 240 V) Optically Coupled Input with Change-of-State Interrupt	VMIVME-1001	D8,D16
2.	32-bit TTL Digital Input with Change-of-State Interrupt	VMIVME-1101	D8,D16
3.	32-bit High Voltage (5 to 50 V) Digital Input with Change-of-State Interrupt	VMIVME-1180	D8,D16
4.	32-bit High Voltage (1 to 66 V) COS board with Data Capture Registers and Built-in-Test	VMIVME-1181	D8,D16,D32
5.	32-bit Optically Coupled Digital Input with Change-of-State Interrupt	VMIVME-1160A	D8,D16

Reference Material List

The reader should refer to "The VMEbus Specification" for a detailed explanation of the VMEbus. "The VMEbus Specification" is available from the following source:

VITA
VMEbus International Trade Association
7825 East Gelding Dr. Suite 104
Scottsdale, AZ 85260
(602) 951-8866
FAX: (602) 951-0720
Internet: www.vita.com

The Epic Ei68C153 Bus Interrupter Module (VME) specification is available from:

Epic Semiconductor, Inc.
4801 S. Lakeshore Dr.
Suite 203
Tempe, AZ 85282
(480) 730-1000
FAX: (480) 838-4740
Internet: www.epicsemi.com
PDF for the Ei68C153: www.epicsemi.com/153.pdf

Application and Configuration Guides

The following Application and Configuration Guides are available from VMIC to assist the user in the selection, specification and implementation of systems based on VMIC's products:

Title	Document No.
Digital Input Board Application Guide	825-000000-000
Change-of-State Application Guide	825-000000-002
Digital I/O (with Built-in-Test) Product Line Description	825-000000-003
Connector and I/O Cable Application Guide	825-000000-006

Physical Description and Specifications

Refer to VMIC Specification No. ***800-101160-000*** for a detailed explanation and physical description of the VMIVME-1160A 32-bit Optically Coupled Digital Input Board, available from the following:

VMIC
12090 South Memorial Pkwy.
Huntsville, AL 35803-3308, USA
(256) 880-0444
(800) 322-3616
FAX: (256) 882-0859
www.vmic.com

Motorola MC68153 BIM

The VMIVME-1160A was originally manufactured using the Motorola MC68153 BIM, which is now out of production. The Epic Ei68C153 BIM is being used as a replacement on all newly-manufactured VMIVME-1160A boards. Any references to the Ei68C153 in this document are also applicable to the MC68153.

CHAPTER 1

Theory of Operation

Contents

Introduction

Block Diagrams

The VMIVME-1160A consists of eight functional building blocks as illustrated in Figure 1-1 on page 18. The eight sections of the VMIVME-1160A are:

1. Address Logic
2. Control Logic
3. Data Logic
4. Bus Interrupter Module (BIM) Logic
5. Interrupt Enable Logic
6. Change-of-State Detection Logic
7. Input Data Registers
8. Input Buffers

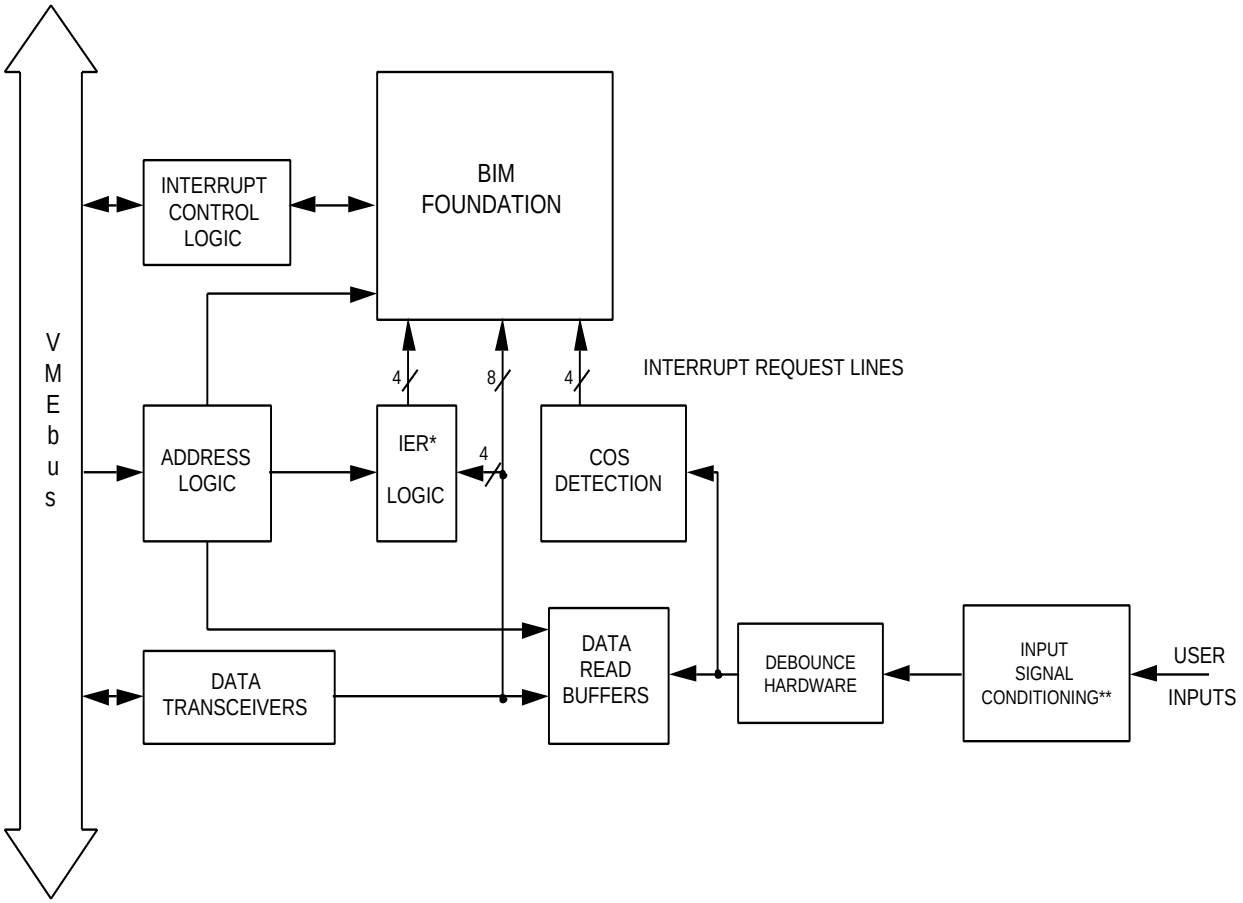
Each section of the design is illustrated in further detail in Figure 1-2 through Figure 1-10 starting on page 19.

Interrupt Functions

Interrupts are generated on any Change-of-State (positive or negative transition). The polarity is determined by reading the input port after a Change-of-State interrupt occurs. The data transfer bus, the arbitration bus, and the interrupt bus are all used in the process of generating and handling bus interrupts.

NOTE: State changes that occur during the interrupt processing window (internal request to interrupt acknowledge cycle complete) will not be detected. The time between user input state changes must not be less than the computer interrupt processing time; otherwise, the state changes will be lost.

The reader should refer to "The VMEbus Specification" for a detailed explanation of the priority interrupt bus. See “Reference Material List” on page 14.



*Internal Enable Register
**Board Dependent

Figure 1-1 VMIVME-1160A Functional Block Diagram

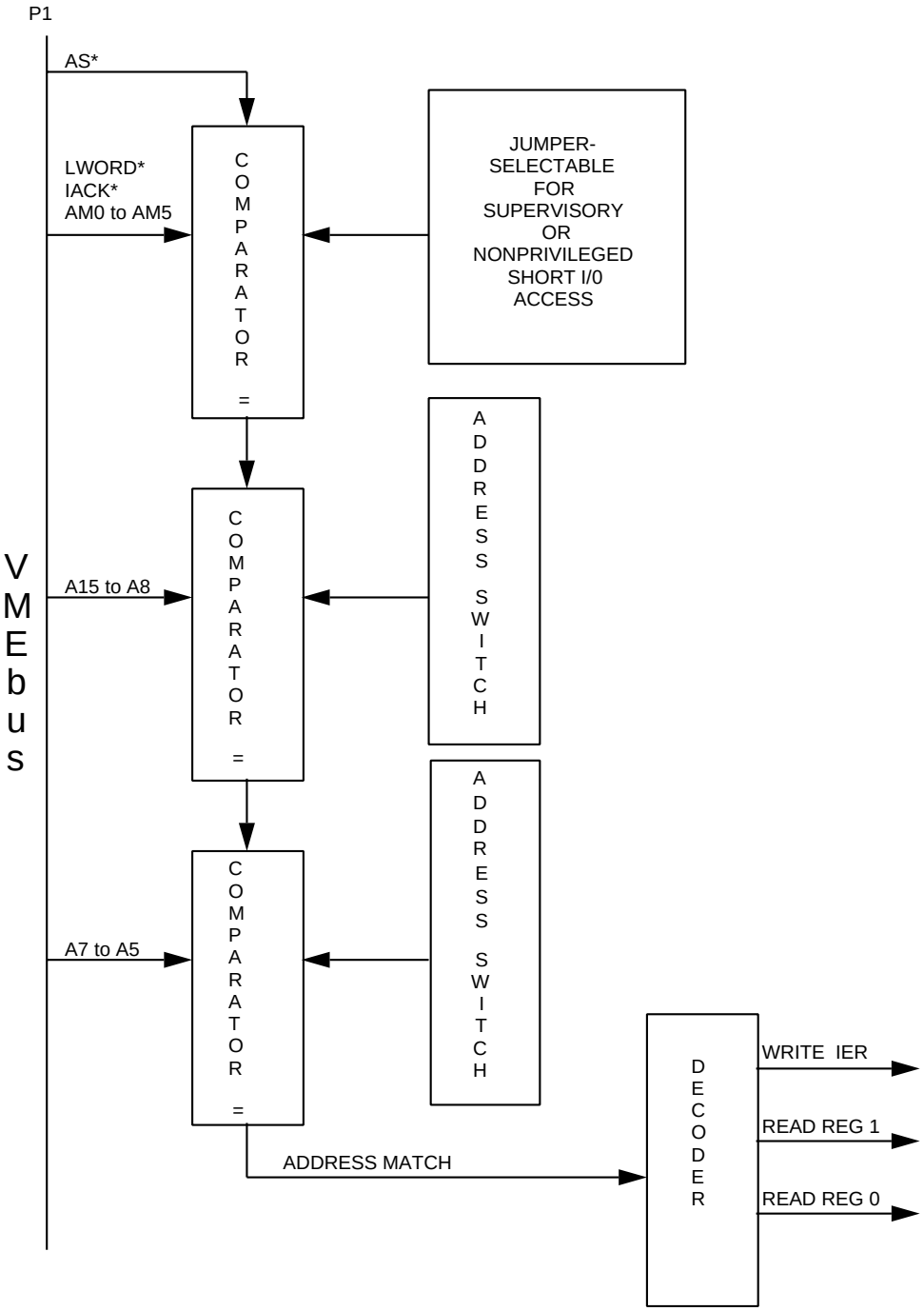


Figure 1-2 Address Section Block Diagram

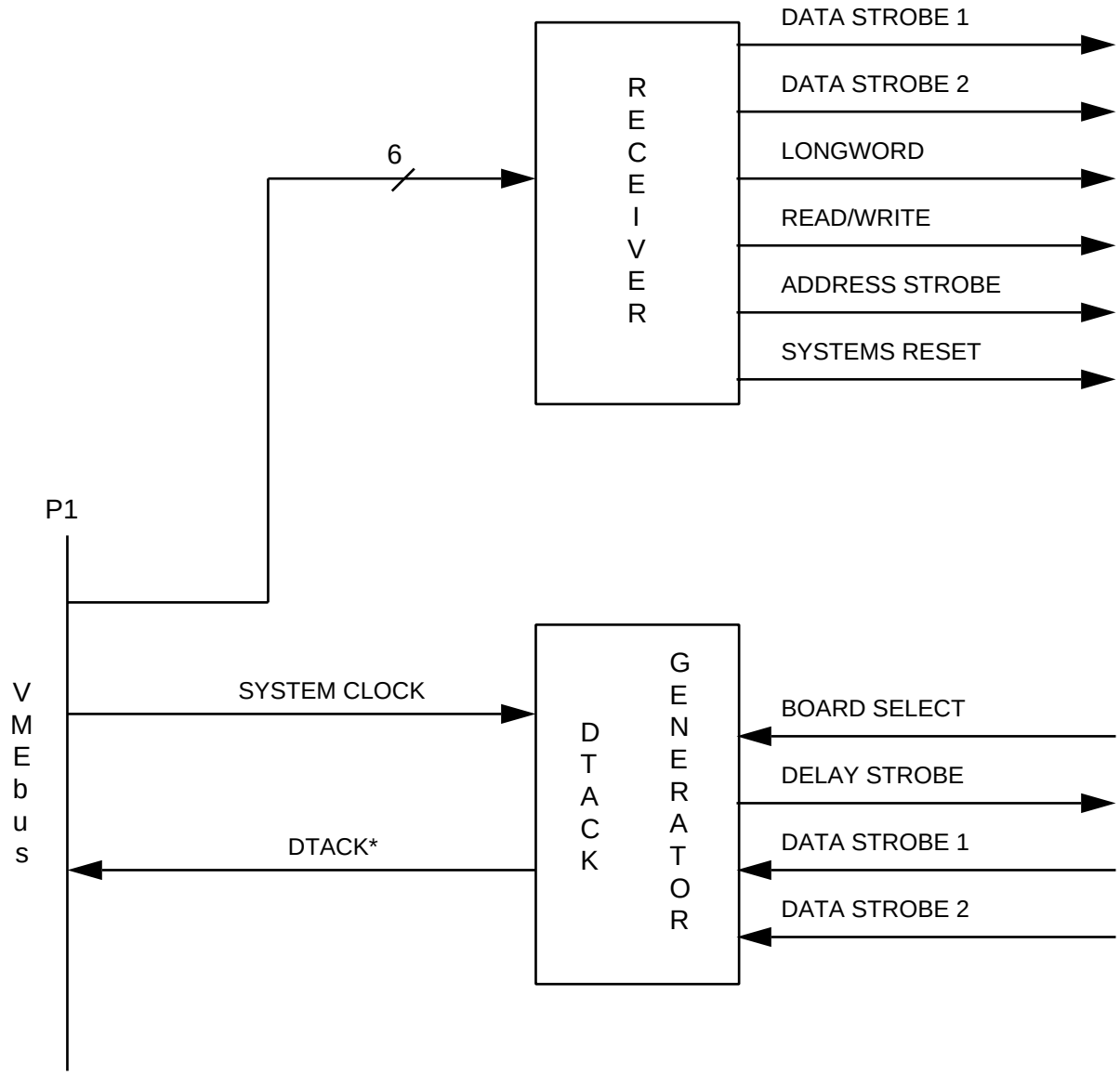


Figure 1-3 Control Section Block Diagram

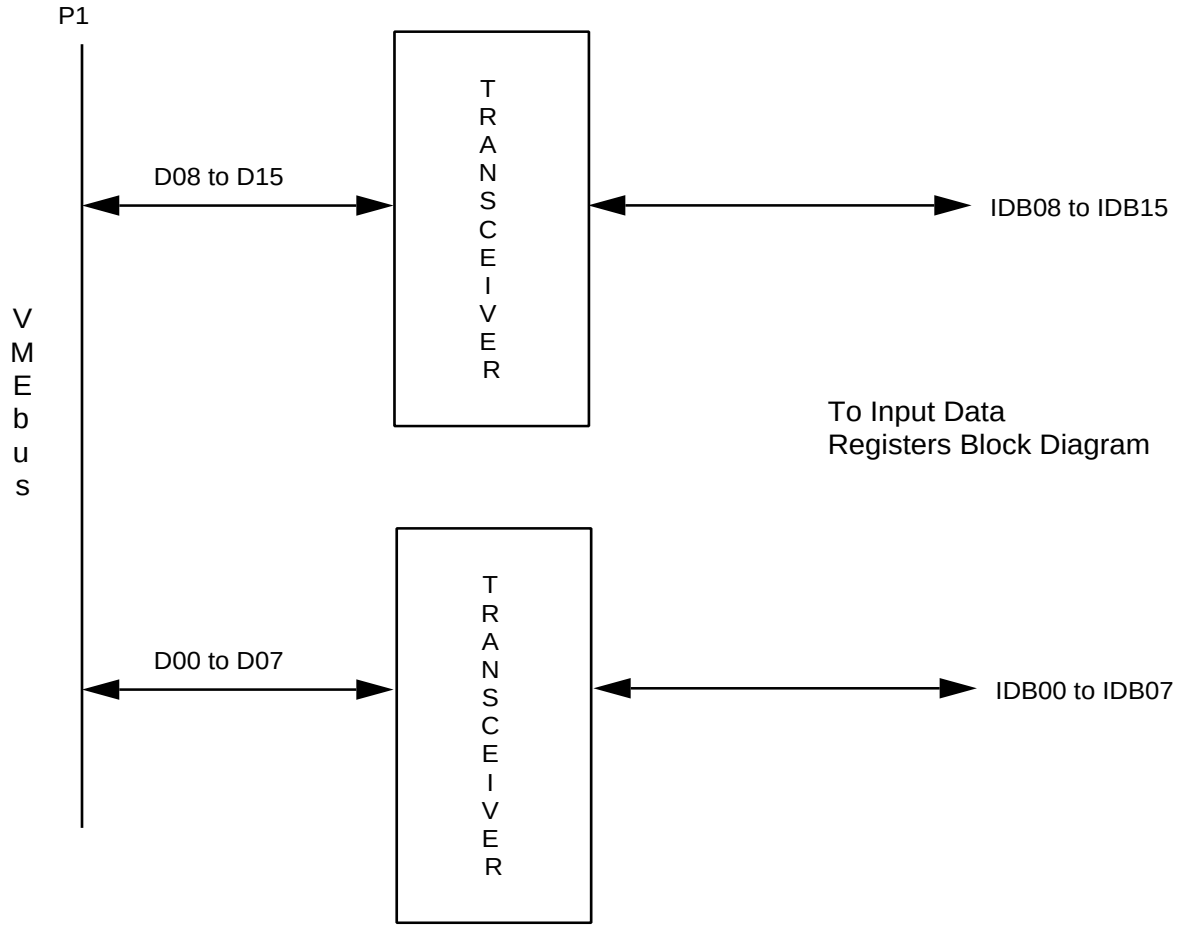


Figure 1-4 Data Section Block Diagram

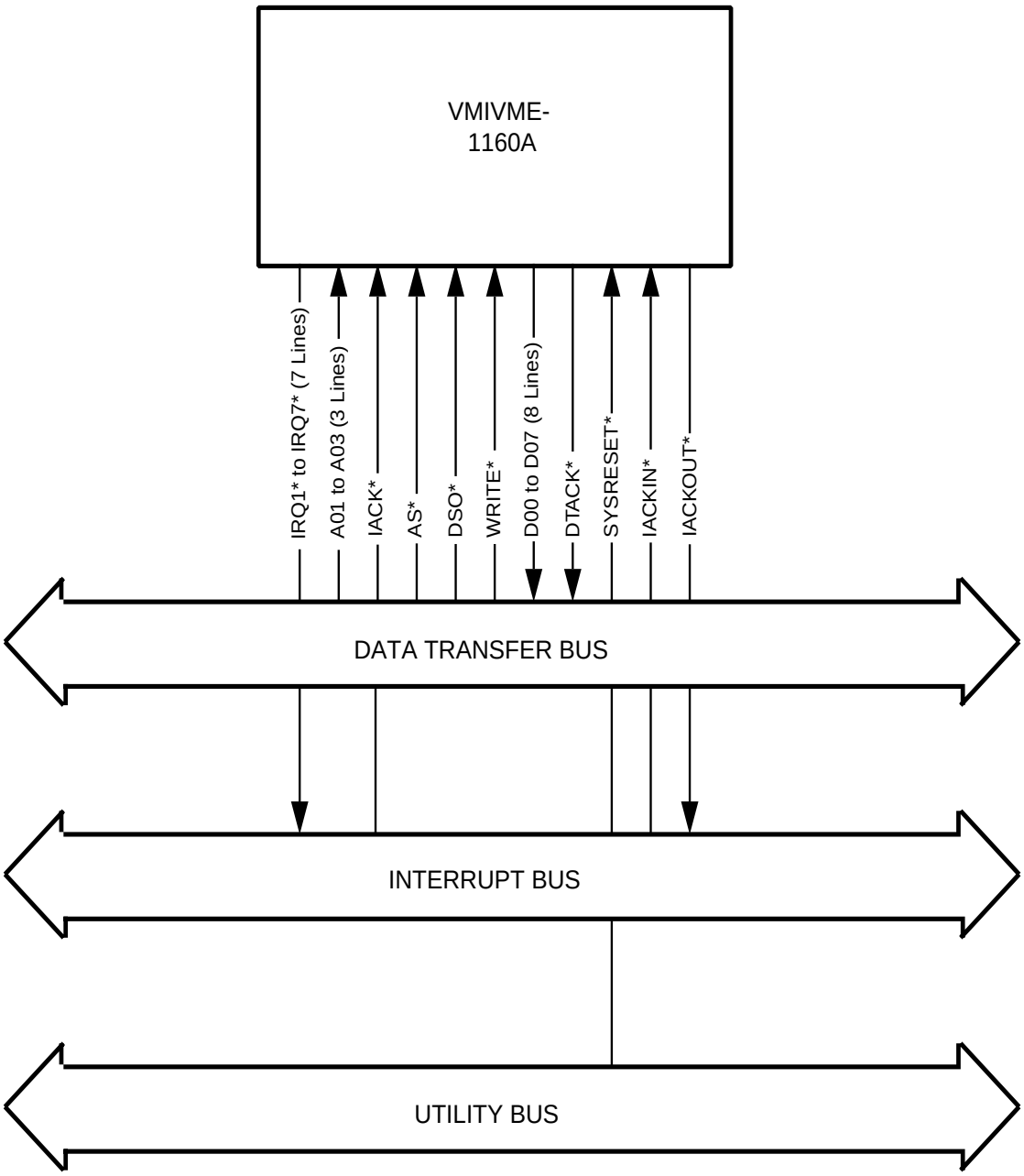


Figure 1-5 VMEbus Signal Lines Used by the VMIVME-1160A

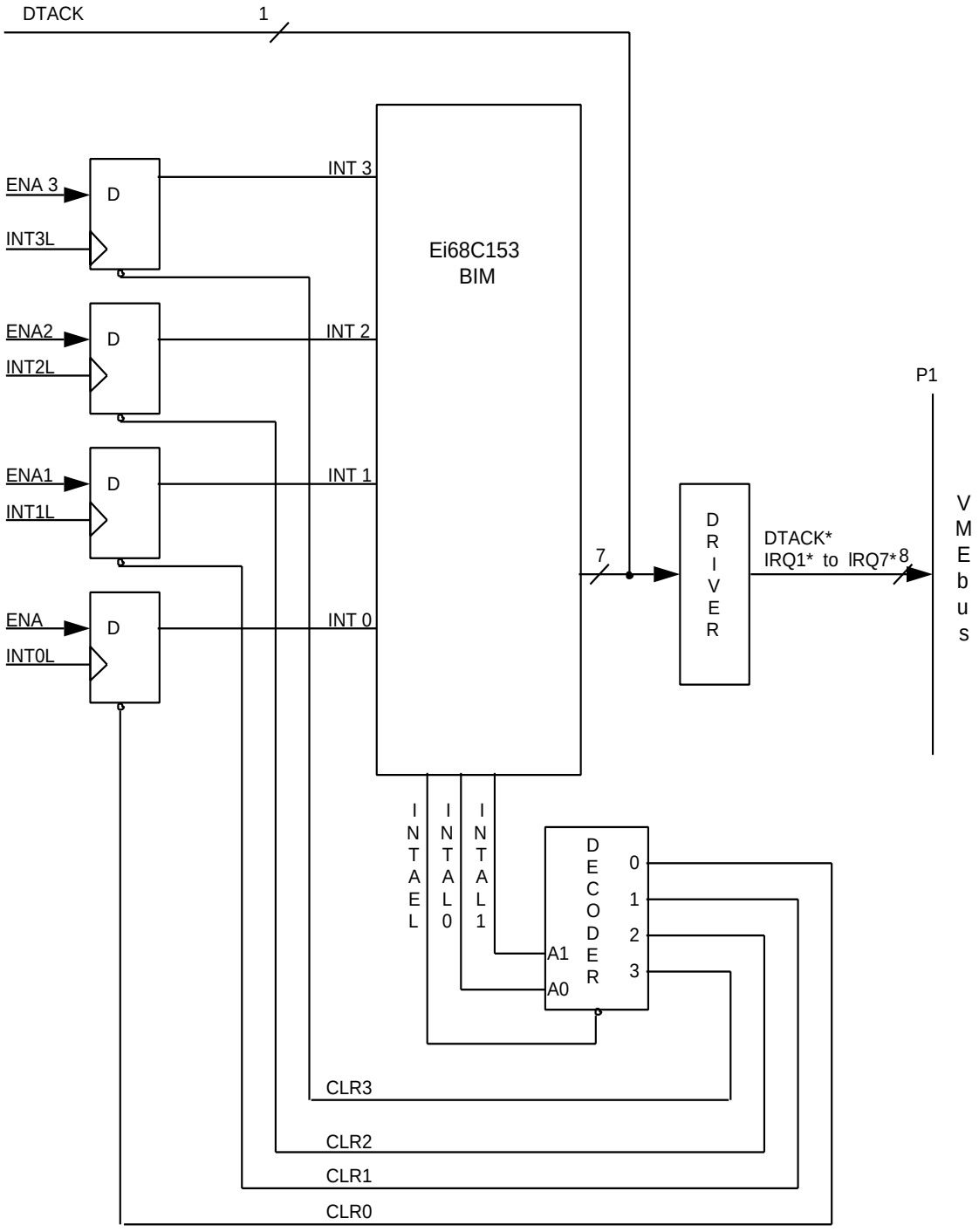


Figure 1-6 BIM Foundation Section Block Diagram

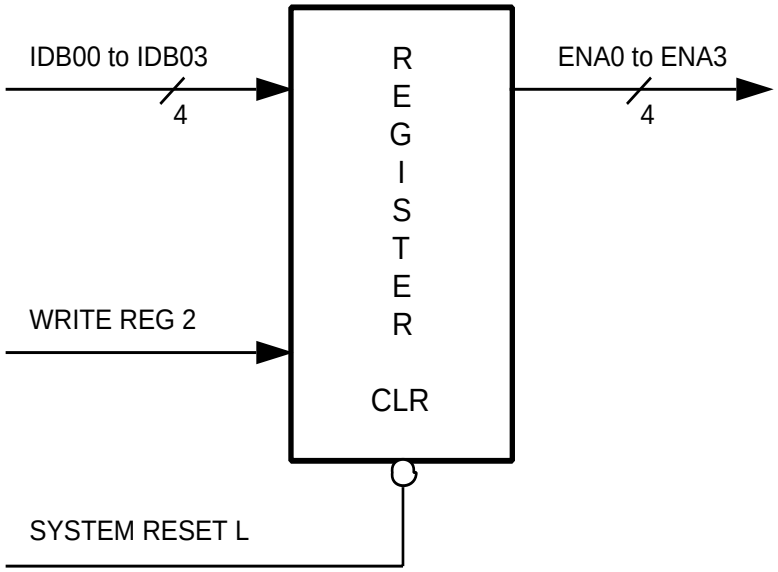


Figure 1-7 Typical IER Logic Section

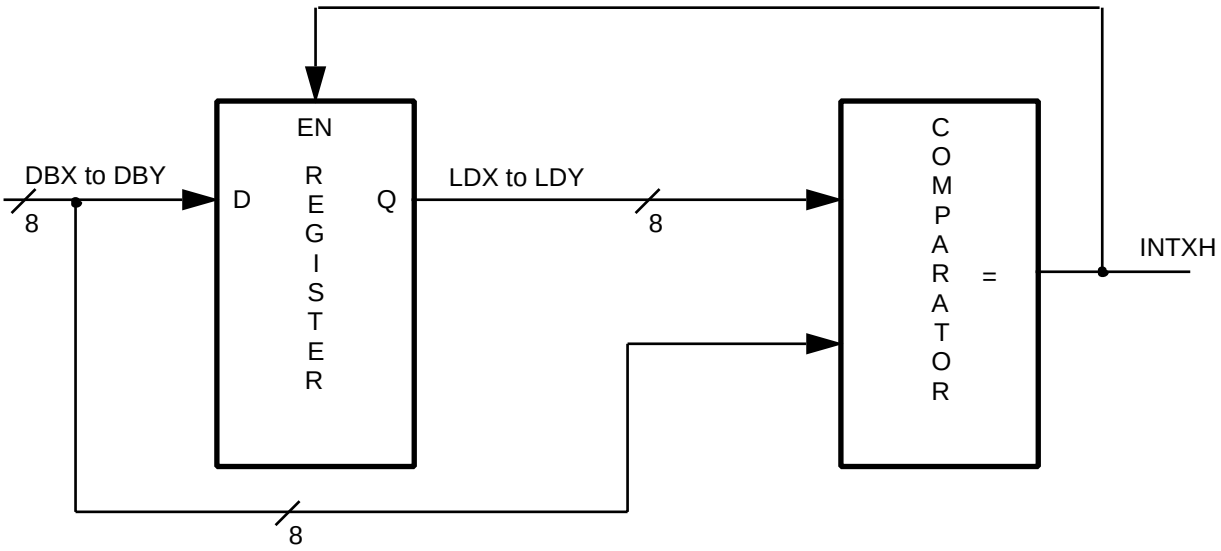


Figure 1-8 Typical Change-of-State Detection Logic Section

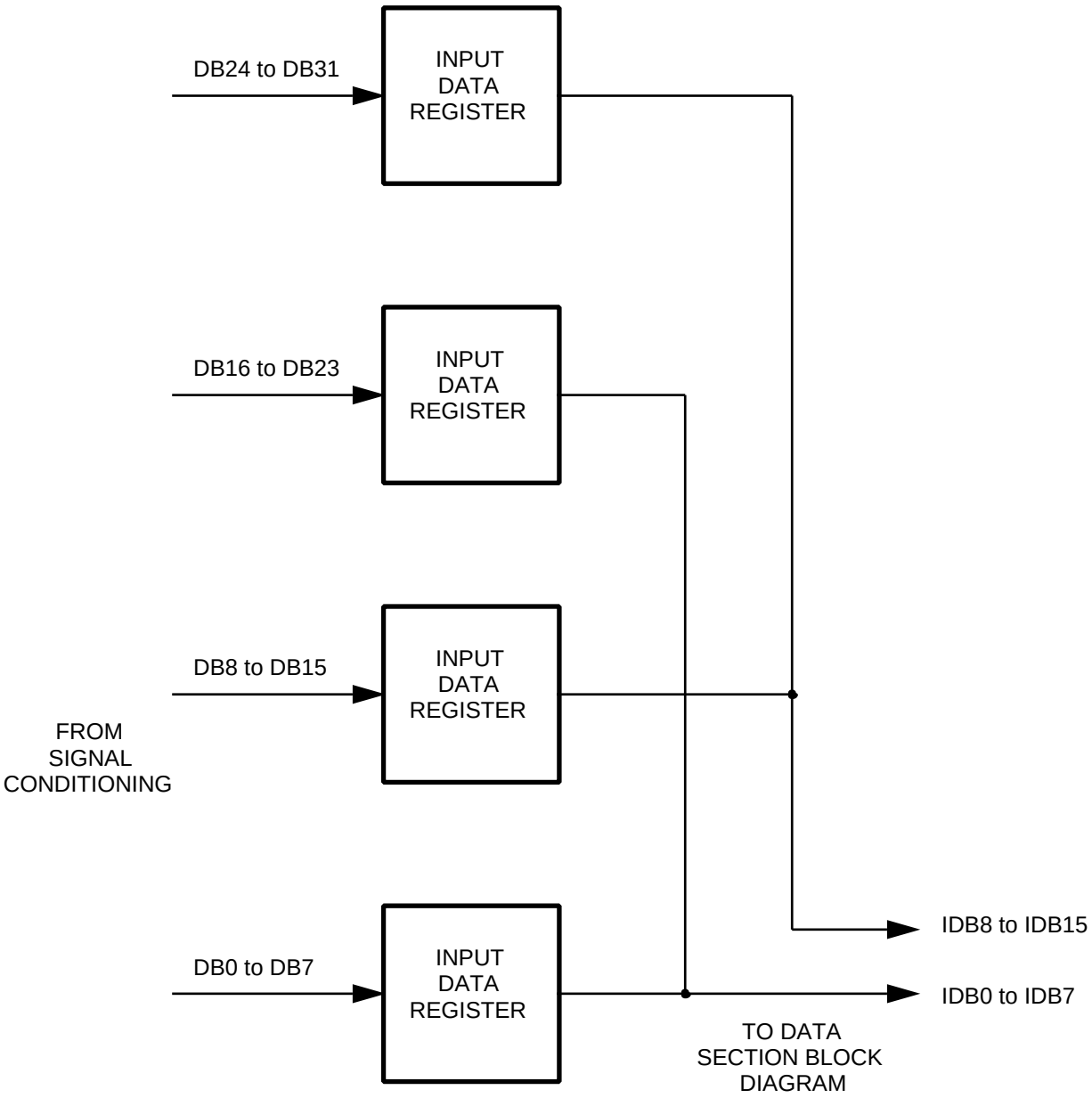


Figure 1-9 Input Data Register Block Diagram

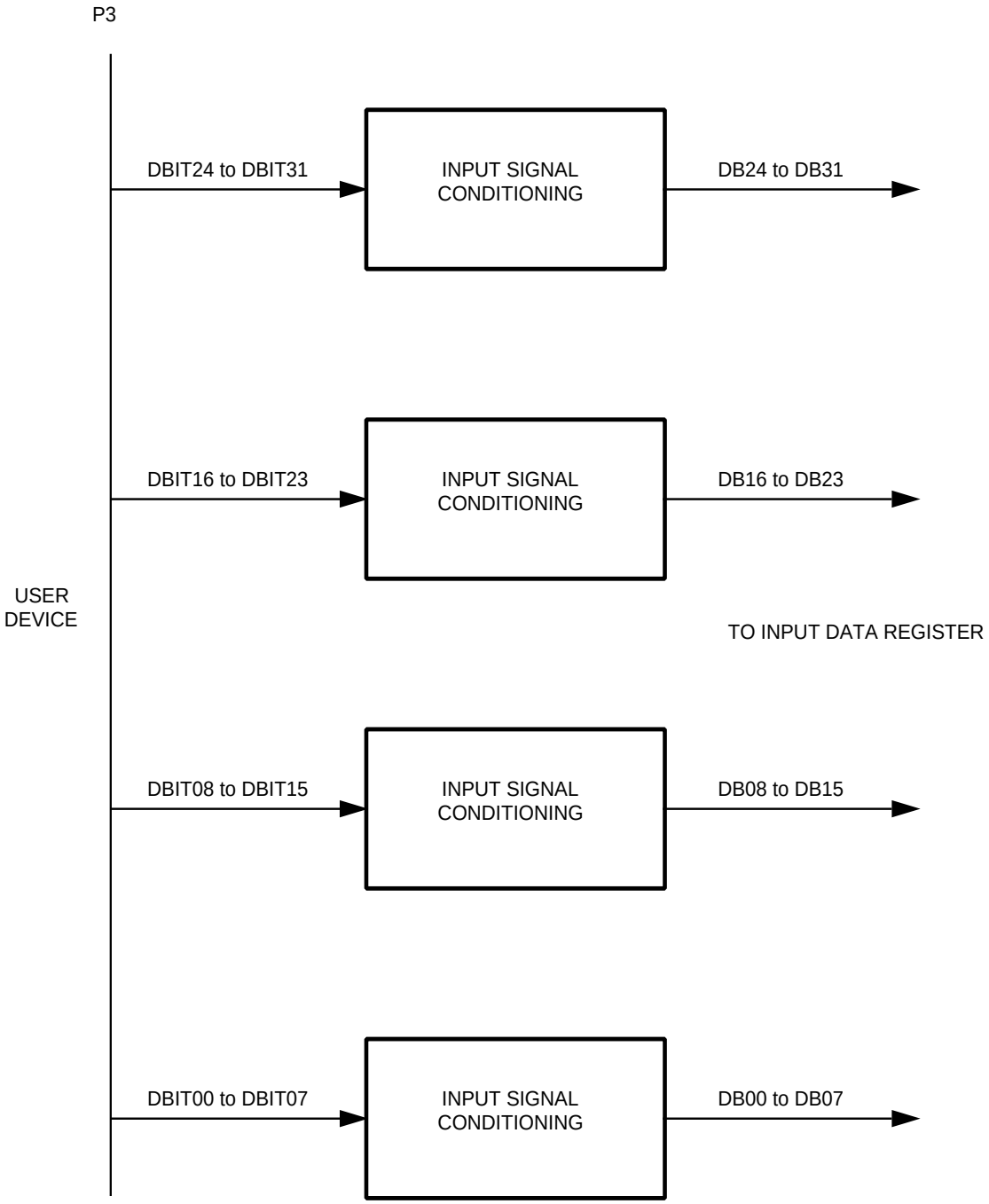


Figure 1-10 Signal Conditioning Block Diagram

Priority Interrupt Subsystem

The following overview of the priority interrupt subsystem assumes that the reader understands the operation of both the data transfer bus and the arbitration bus.

The interrupt bus consists of seven interrupt request signal lines, one daisy-chain signal line, and one interrupt acknowledge line:

IRQ1*	IRQ4*	IRQ7*
IRQ2*	IRQ5*	IACK*
IRQ3*	IRQ6*	IACKIN*/IACKOUT*

Each interrupt request line may be driven low by the VMIVME-1160A Board or other interrupter to request an interrupt. In a single handler system, these interrupt request lines are prioritized, with IRQ7* having the highest priority.

The IACK* line runs the full length of the bus and is connected to the IACKIN* pin of slot A1. When it is driven low, it initiates a low-going transition down the interrupt acknowledge daisy-chain. This may not occur immediately, since additional constraints are placed on the propagation of IACKIN*/IACKOUT*.

Each of the seven interrupt request lines may be shared by two or more interrupter boards. Because of this, some method must be provided to ensure that only one of the boards is acknowledged. This is done by means of the interrupt acknowledge daisy-chain. The daisy-chain line passes through each board on the VMEbus. When an interrupt is acknowledged, IACKIN* is driven low at slot A1. Each board that is driving an interrupt request line low must wait for the low level down the daisy-chain, thereby guaranteeing that only one board will be acknowledged.

The VMIVME-1160A uses one of the seven IRQX* lines to request an interrupt. It then monitors the DTB address bus, IACK*, and the IACKIN*/IACKOUT* daisy-chain to determine when its interrupt is being acknowledged. When acknowledged, it places its status/ID byte on the lower eight lines of the data bus and signals the byte's validity to the interrupt handler via the DTACK* line.

The VMEbus signal lines used by the VMIVME-1160A are shown in Figure 1-5 on page 22.

CHAPTER
2

Configuration and Installation

Contents

Unpacking Procedures

CAUTION: Some of the components assembled on VMIC's products may be sensitive to electrostatic discharge and damage may occur on boards that are subjected to a high-energy electrostatic field. When the board is placed on a bench for configuring, etc., it is suggested that conductive material should be inserted under the board to provide a conductive shunt. Unused boards should be stored in the same protective boxes in which they were shipped.

Upon receipt, any precautions found in the shipping container should be observed. All items should be carefully unpacked and thoroughly inspected for damage that might have occurred during shipment. The board(s) should be checked for broken components, damaged printed circuit board(s), heat damage, and other visible contamination. All claims arising from shipping damage should be filed with the carrier and a complete report sent to VMIC, together with a request for advice concerning the disposition of the damaged item(s).

Physical Installation

CAUTION: Do not install or remove the board while power is applied.

De-energize the equipment and insert the board into an appropriate slot of the chassis. While ensuring that the card is properly aligned and oriented in the supporting card guides, slide the card smoothly forward against the mating connector until firmly seated.

Configuration Switches

Refer to Figure 2-1 below for the location of jumpers and address switches.

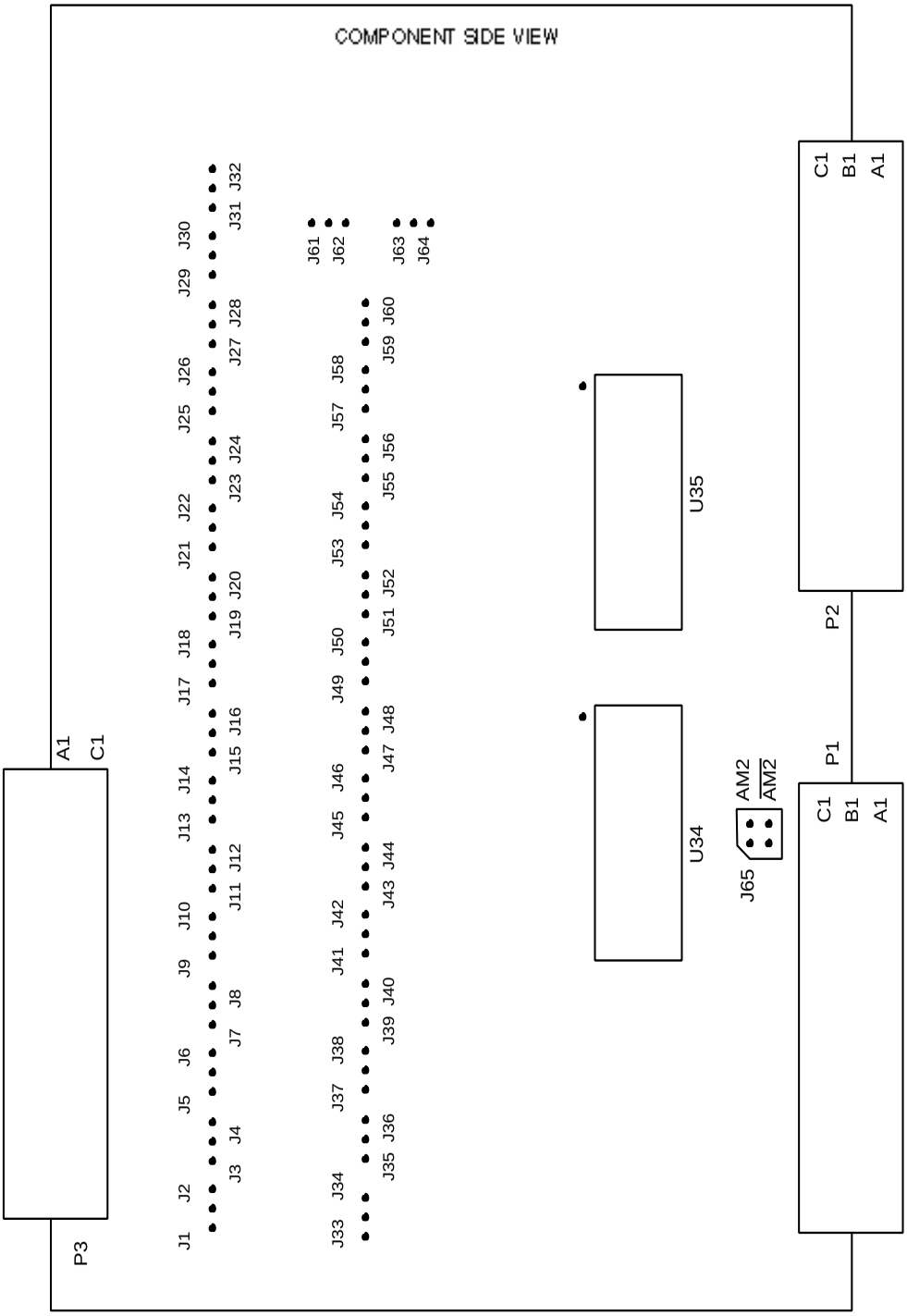


Figure 2-1 Location of Jumpers and Address Switches

Contact Sense, Voltage Source Selection

The VMIVME-1160A may be ordered with its input signal conditioning electronics factory configured for voltage source or contact sense input options. A wide range of input voltages (5 to 48 V) are supported, and you may select from a variety of input filters. Refer to “Product Specification 800-101160-000” for the complete ordering information.

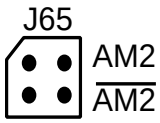
In Figure 2-4 on page 33, jumper Jx is the contact sense jumper. This jumper is installed at the factory when a contact sense board is ordered. Jumper Jx corresponds to the even numbered jumpers (J2, J4, etc.). Jumper Jy is the voltage source jumper. It is installed for voltage source boards and corresponds to the odd numbered jumpers (J3, J5, etc.) with the exception of jumper J65, which is the address modifier jumper.

External Pull-up Voltage

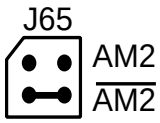
External voltage is connected through the P2 connector on row C, pins 30 and 31 (VPOS uses P2 row C; pin 30, VNEG uses P2 row C, pin 31). This voltage is required on contact sense configurations only.

Address Modifiers

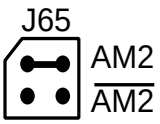
I/O Access Mode is configured by a dual header jumper J65. Figure 2-2 below illustrates how jumper J65 can be configured for short supervisory I/O access, short nonprivileged I/O access or both. The VMIVME-1160A is factory configured to respond to short nonprivileged I/O access.



a. Configuration for Short Supervisory I/O Access, AM = 2D



b. Configuration for Short Nonprivileged I/O Access, AM = 29



c. Configuration for both Short Supervisory I/O Access or Short Nonprivileged I/O Access

Figure 2-2 I/O Access Mode Selection

Address Selection Switches

Figure 2-3 below shows the two addressing DIP switches on-board the VMIVME-1160A and their use in the addressing scheme.

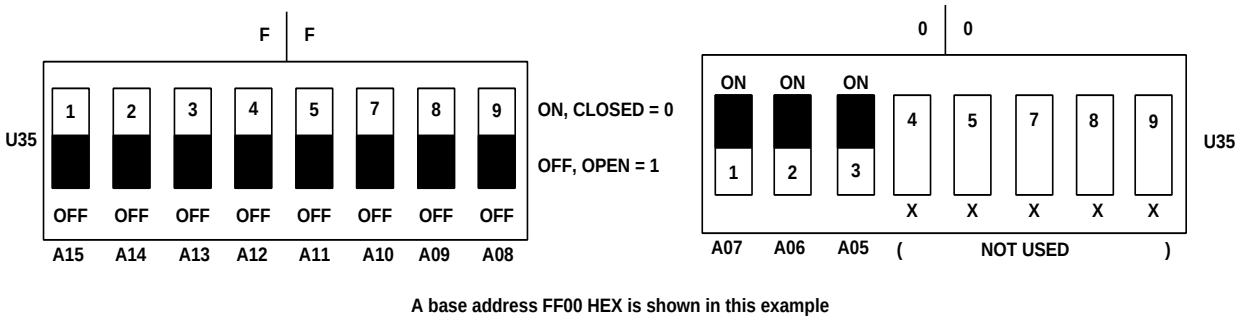


Figure 2-3 Address Selection Switches

Connector Pin Configuration

Figure 2-4 below shows a typical input channel. The input connector, P3, is a 64-pin connector designed with pins in two rows, Row A and Row C. Connection is accomplished by connecting the appropriate row pin to the appropriate user signal. For example, in the voltage source case the user is recommended to connect the equipment ground to the Row C pins, and bring each signal input in through Row A pins.

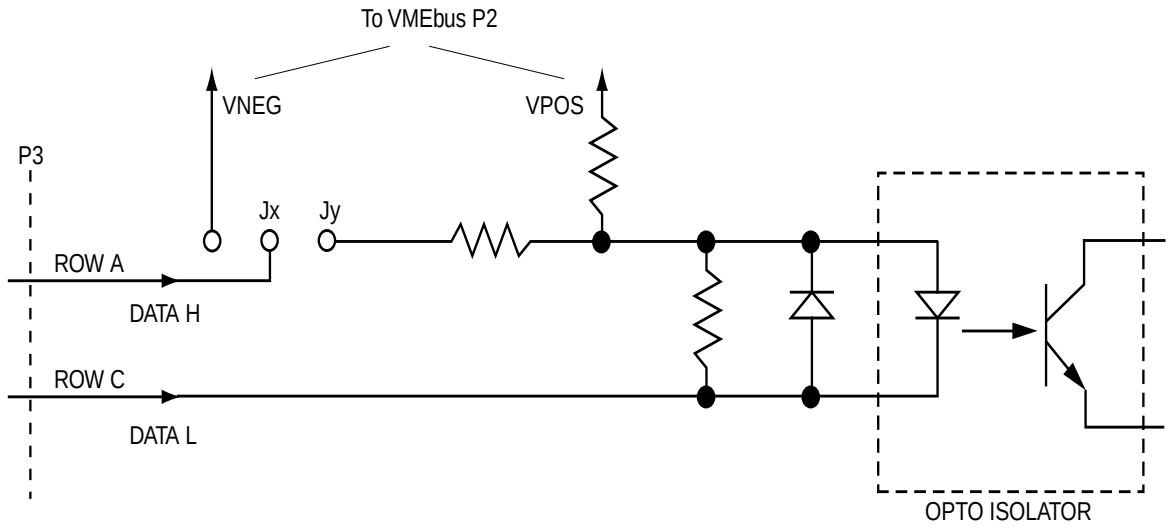


Figure 2-4 Typical Input Channel

I/O Cable and Front Panel Connector Configuration

The I/O connector (P3) on the VMIVME-1160A is a 64-pin DIN standard and was selected by VMIC because of its high quality. Although these connectors are generally used with flat-ribbon cables, a variety of cables and mating connectors are available for most user requirements. The user should refer to VMIC's Connector and I/O Cable Application Guide (VMIC's Document 825-000000-006) for additional information concerning the variety of possible cabling and connector types available.

Details concerning I/O connections are shown in Figure 2-5 below. Conductor No. 1 is shown at the bottom of the connector as it plugs into the header, corresponding to pin No. 1 on the P3 header connector.

A compatible flat-ribbon cable connector for the VMIVME-1160A is Panduit No. 120-964-435E, and strain relief, Panduit No. 100-000-032. The Header Connector soldered to the PC board is Panduit No. 120-964-033A. Figure 2-6 on page 35 shows the pin out of connector P3. Table 2-2 on page 38 shows the P3 connector pin assignments for the inputs to the board. Figure 2-7 on page 36 shows the pin out of connector P2. Table 2-1 on page 37 shows the pertinent pin assignments for connector P2.

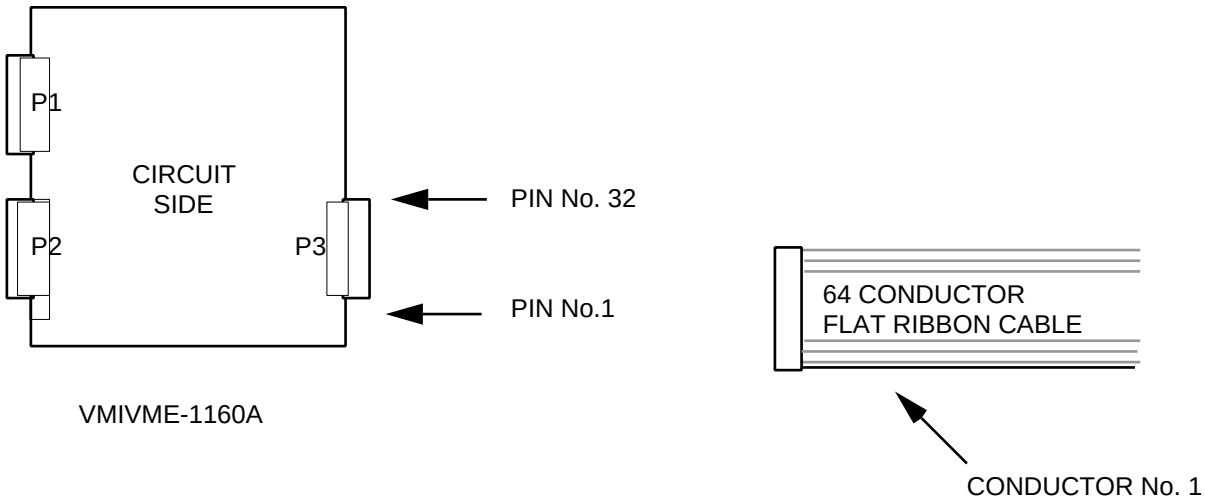


Figure 2-5 Cable Connector Configuration

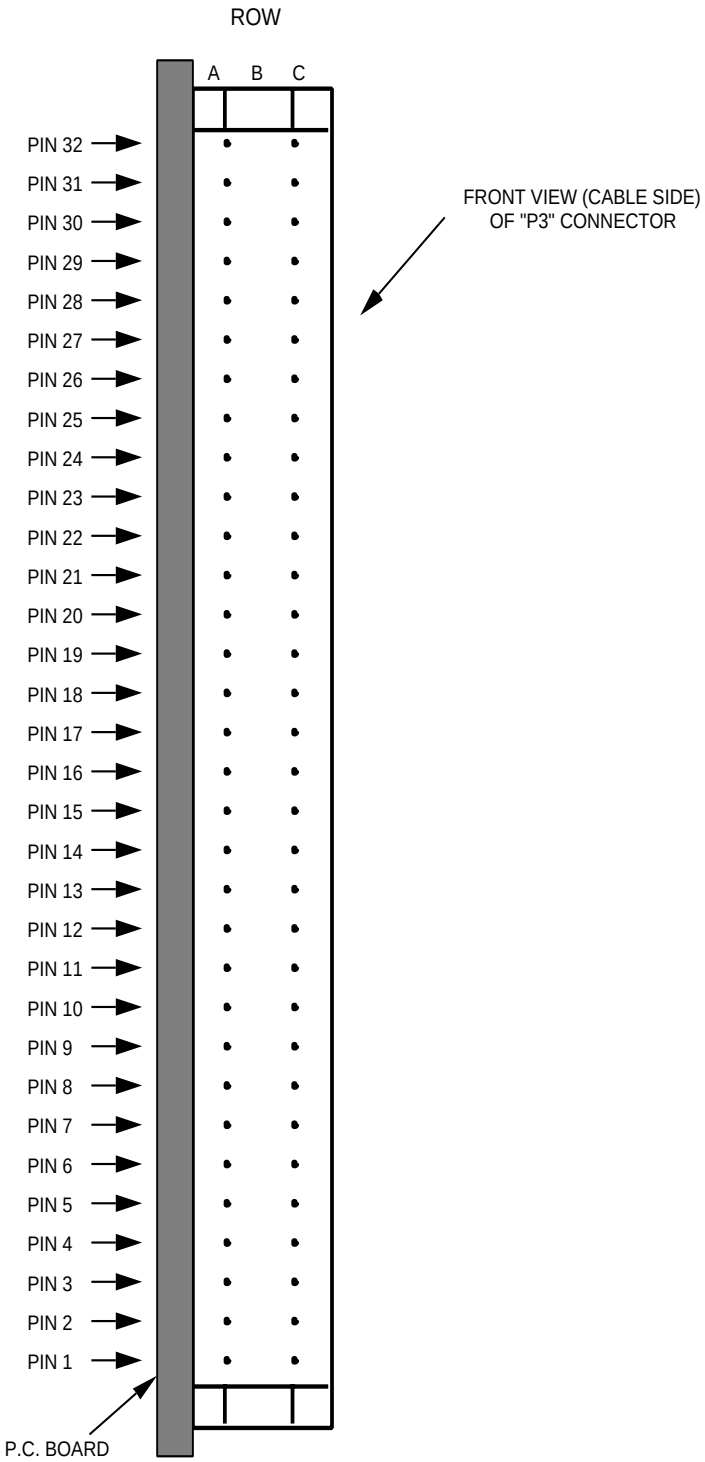


Figure 2-6 P3 Connector Pin Layout

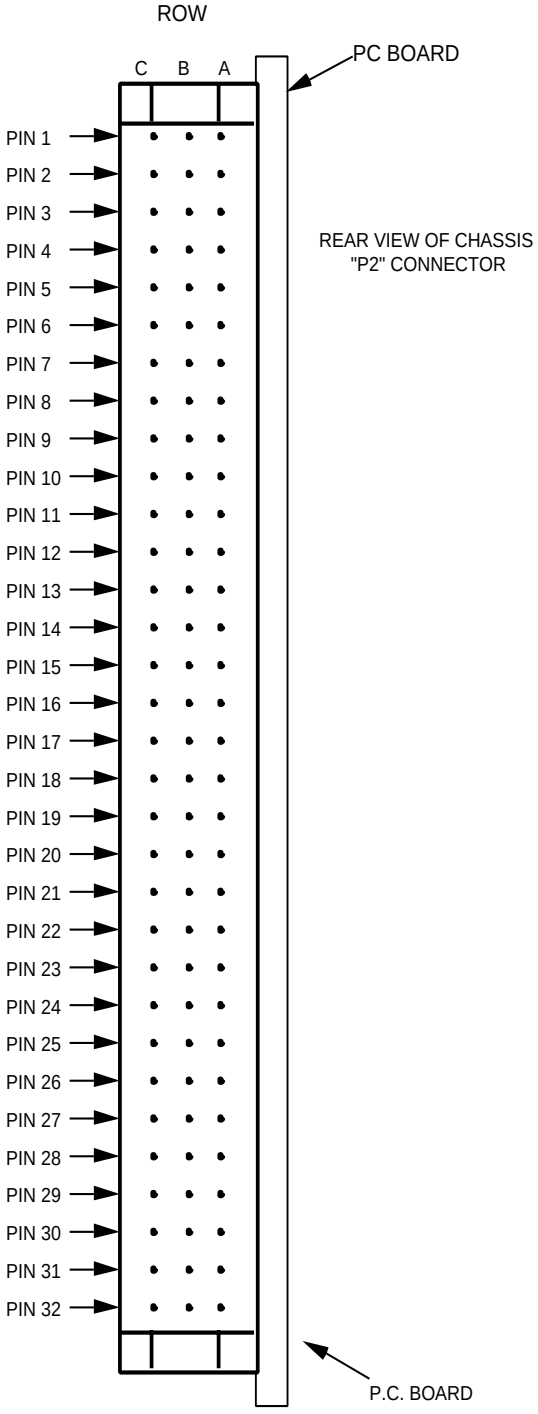


Figure 2-7 P2 Connector Pin Layout

Table 2-1 P2 Connector Pin Assignments

Pin No.	Row A[1]	Row B[2]	Row C
1		+5 V	
2		GND	
3			
4			
5			
6			
7			
8			
9			
10			
11			
12		GND	
13		+5 V	
14			
15			
16			
17			
18			
19			
20			
21			
22		GND	
23			
24			
25			
26			
27			
28			
29			
30			VPOS
31		GND	VNEG
32		+5 V	

NOTES: 1). External reference is supplied by the user.
2). Inputs to the board - not required.

Table 2-2 P3 Pin-Channel Assignment

P3	
ROW A PIN	CHANNEL NO.
32	31
31	30
30	29
29	28
28	27
27	26
26	25
25	24
24	23
23	22
22	21
21	20
20	19
19	18
18	17
17	16

P3	
ROW A PIN	CHANNEL NO.
16	15
15	14
14	13
13	12
12	11
11	10
10	09
09	08
08	07
07	06
06	05
05	04
04	03
03	02
02	01
01	00

CHAPTER 3

Programming

Contents

Introduction

Operational Overview

The VMIVME-1160A is designed to provide 32 channels of high voltage, optically coupled digital inputs with change-of-state detection. Each eight bits (byte) of input may have a unique interrupt vector that is generated upon a change-of-state of any bit in that byte. The VMIVME-1160A also has an Interrupt Enable Register which is used to allow interrupts on change-of-state to be enabled on a byte basis. The channels of input data may be accessed as two 16-bit words or four 8-bit bytes.

A register map is shown in Table 3-1 below. Detailed register bit definitions are shown in Table 3-2 on page 40 through Table 3-8 on page 42. A typical flow diagram is shown in Figure 3-1 on page 43.

Table 3-1 Register Map

Relative Address	Mnemonic	Name/Function	
$XX00	DR0	Data Register 0	
$XX01	DR1	Data Register 1	
$XX02	DR2	Data Register 2	
$XX03	DR3	Data Register 3	
$XX05	IER	Interrupt Enable Register	
$XX11	CR0	Control Register 0	BIM Registers
$XX13	CR1	Control Register 1	
$XX15	CR2	Control Register 2	
$XX17	CR3	Control Register 3	
$XX19	VR0	Vector Register 0	
$XX1B	VR1	Vector Register 1	
$XX1D	VR2	Vector Register 2	
$XX1F	VR3	Vector Register 3	

Table 3-2 $XX00 Data Register 0 (DR0)

BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
INPUT DATA							
ID31	ID30	ID29	ID28	ID27	ID26	ID25	ID24

Table 3-3 $XX01 Data Register 1 (DR1)

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
INPUT DATA							
ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16

Table 3-4 $XX02 Data Register 2 (DR2)

BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
INPUT DATA							
ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8

Table 3-5 $XX03 Data Register 3 (DR3)

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
INPUT DATA							
ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0

Table 3-6 $XX05 Interupt Enable Register (IER)

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
				Interrupt Enable (see note)			
Reserved				DR0	DR1	DR2	DR3

NOTE: A one (1) in the bit location will enable the interrupts for the stated Data Register.

Table 3-7 Typical BIM Control Register Map

Control Register 0 for DR3 Inputs (Offset $XXXX11)							
BIT 7	**BIT 6**	**BIT 5**	**BIT 4**	**BIT 3**	**BIT 2**	**BIT 1**	**BIT 0**
FLAG	Flag Auto Clear	Vector	INT Enable	INT Auto Clear	Interrupt Level		
F	FAC	0 = Internal 1 = External	IRE	1 = Auto 0 = No	L2	L1	L0

Control Register 1 for DR2 Inputs (Offset $XXXX13)							
BIT 7	**BIT 6**	**BIT 5**	**BIT 4**	**BIT 3**	**BIT 2**	**BIT 1**	**BIT 0**
FLAG	Flag Auto Clear	Vector	INT Enable	INT Auto Clear	Interrupt Level		
F	FAC	0 = Internal 1 = External	IRE	1 = Auto 0 = No	L2	L1	L0

Control Register 2 for DR1 Inputs (Offset $XXXX15)							
BIT 7	**BIT 6**	**BIT 5**	**BIT 4**	**BIT 3**	**BIT 2**	**BIT 1**	**BIT 0**
FLAG	Flag Auto Clear	Vector	INT Enable	INT Auto Clear	Interrupt Level		
F	FAC	0 = Internal 1 = External	IRE	1 = Auto 0 = No	L2	L1	L0

Control Register 3 for DR0 Inputs (Offset $XXXX17)							
BIT 7	**BIT 6**	**BIT 5**	**BIT 4**	**BIT 3**	**BIT 2**	**BIT 1**	**BIT 0**
FLAG	Flag Auto Clear	Vector	INT Enable	INT Auto Clear	Interrupt Level		
F	FAC	0 = Internal 1 = External	IRE	1 = Auto 0 = No	L2	L1	L0

Table 3-8 Typical BIM Vector Register Map

Vector Register (VR0) for DR3 COS (Offset $XX19)							
V7	V6	V5	V4	V3	V2	V1	V0

Vector Register (VR1) for DR2 COS (Offset $XX1B)							
V7	V6	V5	V4	V3	V2	V1	V0

Vector Register (VR2) for DR1 COS (Offset $XX1D)							
V7	V6	V5	V4	V3	V2	V1	V0

Vector Register (VR3) for DR0 COS (Offset $XX1F)							
V7	V6	V5	V4	V3	V2	V1	V0

NOTE: COS = Change-of-State

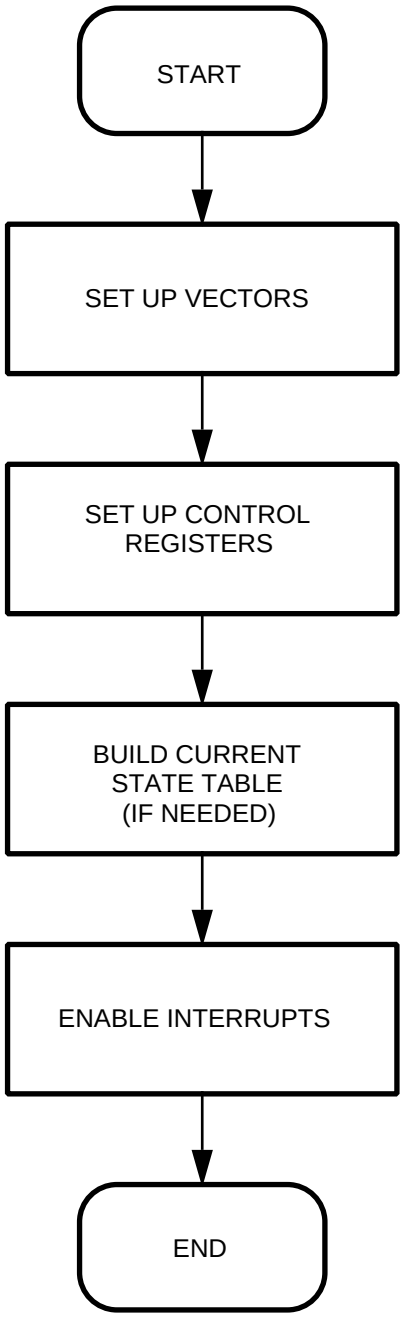


Figure 3-1 VMIVME-1160A Programming Flow Diagram

Typical Programming Example

The following example code enables interrupts, processes interrupts, and displays the current input data when a change-of-state occurs.

```
/*
**  1160.h
**
**  VMIVME-1160 32 BIT OPTICALLY COUPLED DIGITAL INPUT W/COS
*/
struct vmivme_1160_registers {
  unsigned char offset[32];
};
typedef struct vmivme_1160_registers v1160_t;

/*
**  register offset definitions
**
**  note: defining register offsets using the above
**        structure/array and defines facilitates
**        skipping unused addresses in the board map.
*/

#define   dr0       offset[0x00]
#define   dr1       offset[0x01]
#define   dr2       offset[0x02]
#define   dr3       offset[0x03]
#define   ier       offset[0x05]
#define   cr0       offset[0x11]
#define   cr1       offset[0x13]
#define   cr2       offset[0x15]
#define   cr3       offset[0x17]
#define   vr0       offset[0x19]
#define   vr1       offset[0x1b]
#define   vr2       offset[0x1d]
#define   vr3       offset[0x1f]

/*
**  interrupt enable control bits
*/

#define   IE_DR0    0x08
#define   IE_DR1    0x04
#define   IE_DR2    0x02
#define   IE_DR3    0x01
/*
**  BIM control bits
*/

#define   FLAG_BIT            0x80
#define   FLAG_AUTO_CLEAR     0x40
#define   EXTERNAL_VECTOR     0x20      /* don't use ! - see manual */
#define   INTERRUPT_ENABLE    0x10
```

```
#define   INTR_AUTO_CLEAR     0x08
#define   REQUEST_LEVEL_7     0x07
#define   REQUEST_LEVEL_6     0x06
#define   REQUEST_LEVEL_5     0x05
#define   REQUEST_LEVEL_4     0x04
#define   REQUEST_LEVEL_3     0x03
#define   REQUEST_LEVEL_2     0x02
#define   REQUEST_LEVEL_1     0x01
#define   INTERRUPTS_OFF      0x00
```

```
/*
**  1160.c
**
**  VMIVME-1160 32 BIT OPTICALLY COUPLED DIGITAL INPUT W/COS
*/

#include  <stdio.h>
#include  "1160.h"

/*
**  System dependent definitions ( Force Cpu-33 SBC )
*/

#define   SHORTIO   0xfbff0000          /* short io window */

#define   USER_VECTOR( v )  v + 0x54    /* 1st available user vector */

/*
**  Define global pointer to vmivme 1160 board
*/

#define   BASE_ADDR_1160  0x0000    /* see manual section 5 */

v1160_t * board = (( v1160_t * )( SHORTIO + BASE_ADDR_1160 ));

/*
**  Define Global Change of State Flags and Data Storage
**
**  Note: COS flags and data variables are modified in the
**        Interrupt Service Routines (ISR's) and therefore
**        need to be global and/or visable to them.
*/

unsigned  char    cos_flags;

#define   COS_0   0x01
#define   COS_1   0x02
#define   COS_2   0x04
#define   COS_3   0x08

unsigned  char    cos_0_data;
unsigned  char    cos_1_data;
unsigned  char    cos_2_data;
unsigned  char    cos_3_data;
```

```
/*
**  Declare external functions ( ISR's )
**
**  Note: The interrupt service routines are written
**        in assembler and linked to the main C program.
**        They modify the globally defined variables
**        above that are visable to main and the ISR's.
*/

void  cos0isr( void );
void  cos1isr( void );
void  cos2isr( void );
void  cos3isr( void );

/*
**  INITIALIZE 1160 BOARD AND DISPLAY CHANGE OF STATE INPUTS
*/

main()
{

  /*
  **  Install ISR Addresses into CPU-33 Vector Table
  **
  **  Note: Vector installation is system dependent. Our
  **        method is through a libarary call that installs
  **        the ISR addresses according to the vector used.
  */

  setvect( USER_VECTOR( 0 ), &cos0isr );
  setvect( USER_VECTOR( 1 ), &cos1isr );
  setvect( USER_VECTOR( 2 ), &cos2isr );
  setvect( USER_VECTOR( 3 ), &cos3isr );

  /*
  **  Initialize Vector Registers
  **
  **  Note: Vectors available to the user are system dependent.
  */

  board->vr0 = USER_VECTOR( 0 );
  board->vr1 = USER_VECTOR( 1 );
  board->vr2 = USER_VECTOR( 2 );
  board->vr3 = USER_VECTOR( 3 );

  /*
  **  Initialize Interrupt Control Registers
  **
  **  The change of state interrupts are enabled with request
  **  levels 3 through 6 where 6 is the highest priority. This
  **  is an arbitrary level selection that could have been a mix
  **  or all the same level depending on the application ...
  */
```

```
    board->cr0 = ( INTERRUPT_ENABLE | INTR_AUTO_CLEAR | REQUEST_LEVEL_3 );
    board->cr1 = ( INTERRUPT_ENABLE | INTR_AUTO_CLEAR | REQUEST_LEVEL_4 );
    board->cr2 = ( INTERRUPT_ENABLE | INTR_AUTO_CLEAR | REQUEST_LEVEL_5 );
    board->cr3 = ( INTERRUPT_ENABLE | INTR_AUTO_CLEAR | REQUEST_LEVEL_6 );

    /*
    **  Initialize Interrupt Enable Register
    **
    **  Writing this register is necessary to enable COS information
    **  to cause interrupts. Each port has an associated interrupt
    **  enable control bit so that COS interrupts may be enabled on
    **  each bytewide port. This simple application enables interrupts
    **  and leaves them enabled. These enables could be used to switch
    **  interrupts on and off to capture or ignore data according to
    **  the application.
    */

    board->ier = ( IE_DR3 | IE_DR2 | IE_DR1 | IE_DR0 );

    /*
    **  Monitor Change of States and Print Input Data
    */

    for( ;; ) {       /* loop forever */

      if( cos_flags &  COS_3 ) {
        printf("\r\nDR3 COS data = %.2X", cos_3_data );
        cos_flags &= ~COS_3;
      }

      if( cos_flags &  COS_2 ) {
        printf("\r\nDR2 COS data = %.2X", cos_2_data );
        cos_flags &= ~COS_2;
      }

      if( cos_flags &  COS_1 ) {
        printf("\r\nDR1 COS data = %.2X", cos_1_data );
        cos_flags &= ~COS_1;
      }

      if( cos_flags &  COS_0 ) {
        printf("\r\nDR0 COS data = %.2X", cos_0_data );
        cos_flags &= ~COS_0;
      }
    }
}
```

```
**  1160isr.sa
**
**  VMIVME-1160 32 BIT OPTICALLY COUPLED DIGITAL INPUT W/COS
**

*    external references and declarations
          xref      _board
          xref      _cos_flags
          xref      _cos_3_data
          xref      _cos_2_data
          xref      _cos_1_data
          xref      _cos_0_data

          xdef      _cos0isr
          xdef      _cos1isr
          xdef      _cos2isr
          xdef      _cos3isr

*    offsets to board data registers

dr0       equ       $00
dr1       equ       $01
dr2       equ       $02
dr3       equ       $03

*    offsets to board intr ctrl registers

cr0       equ       $11
cr1       equ       $13
cr2       equ       $15
cr3       equ       $17

          section   code

*    service dr3 / cos3 interrupt

_cos3isr  movem.l   a0,-(a7)            preserve registers
          move.l    _board,a0           get pointer to board
          move.b    (dr3,a0),_cos_3_data read COS data
          bset      #4,(cr3,a0)         re-enable interrupt
          bset      #3,_cos_flags       set COS intr flag
          movem.l   (a7)+,a0            restore registers
          rte                           return from exception

*    service dr2 / cos2 interrupt

_cos2isr  movem.l   a0,-(a7)            preserve registers
          move.l    _board,a0           get pointer to board
          move.b    (dr2,a0),_cos_2_data read COS data
          bset      #4,(cr2,a0)         re-enable interrupt
          bset      #2,_cos_flags       set COS intr flag
          movem.l   (a7)+,a0            restore registers
          rte                           return from exception
```

```
*    service dr1 / cos1 interrupt

_cos1isr  movem.l   a0,-(a7)           preserve registers
          move.l    _board,a0          get pointer to board
          move.b    (dr1,a0),_cos_1_data read COS data
          bset      #4,(cr1,a0)        re-enable interrupt
          bset      #1,_cos_flags      set COS intr flag
          movem.l   (a7)+,a0           restore registers
          rte                          return from exception

*    service dr0 / cos0 interrupt

_cos0isr  movem.l   a0,-(a7)           preserve registers
          move.l    _board,a0          get pointer to board
          move.b    (dr0,a0),_cos_0_data read COS data
          bset      #4,(cr0,a0)        re-enable interrupt
          bset      #0,_cos_flags      set COS intr flag
          movem.l   (a7)+,a0           restore registers
          rte                          return from exception
```

Maintenance

Maintenance

This section provides information relative to the care and maintenance of VMIC's products. If the product malfunctions, verify the following:

- Software
- System configuration
- Electrical connections
- Jumper or configuration options
- Boards are fully inserted into their proper connector location
- Connector pins are clean and free from contamination
- No components of adjacent boards are disturbed when inserting or removing the board from the chassis
- Quality of cables and I/O connections

If the product must be returned, contact VMIC for a Return Material Authorization (RMA) Number. **This RMA Number must be obtained prior to any return**.

Contact VMIC customer Service at 1-800-240-7782, or
E-mail: customer.service@vmic.com.

Maintenance Prints

User level repairs are not recommended. The drawings and tables in this manual are for reference purposes only.