

## Proposed MFC2 design

### Introduction

This note describes proposed new design of the Muon Fanout Card (MFC) for D0 muon detector system. Key concepts, new features implemented and changes to the original design are discussed. The new MFC design is expected to be fully compatible with the existing muon electronics.

### Basic concept

MFC design is entirely based on the D0 Electronics Certification Board (ECB) specifications: SCL specification and GS specification\*. The basic concept is described in the Muon Electronics Technical Design Report (D0 Note 3299). The main function of the MFC is to interface the muon front-end electronics to the D0 Trigger Framework (TFW). The MFC receives timing and control signals (including trigger decisions) from the TFW via the SCL daughter card and distributes them on a customized J2 backplane in the muon readout crate. These signals are sent to the front-end electronics by the Muon Readout Cards (MRCs) located in that crate. The MFC receives status signals from the Muon Readout Cards and sends them to the TFW via the SCL daughter card. The MFC is a VME interrupter and can generate interrupt requests originating from variety of sources including service requests from the MRCs.

The MFC has an internal trigger pattern generator, which can be used to simulate D0 TFW activities. A list of triggers can be stored in a 32 Kbyte Dual Port RAM. One port of this RAM is accessible via VME to program a trigger sequence. During its operation the trigger pattern generator reads data from the other port of this RAM. In order to facilitate synchronization of several MFCs a Cypress HOTLink™ serial receiver is implemented for use with the muon Trigger Fanout Card (TFC). This feature allows several muon readout crates to be run locally without involving the TFW. The MFC has an internal 8 Kbytes of non-volatile RAM to store status and user information. The MFC interfaces with the VME Buffer Driver (VBD) using a hardware backplane connection. Internal trigger FIFOs and counters are implemented for diagnostic purpose. A simplified block-diagram of the MFC is shown in **Fig. 1**.

The MFC has a standard **A24:D16** slave VME interface. It also has an interrupt controller programmable via VME commands. The MFC generates an encoded timing signal that includes the First Crossing, GAP and Sync GAP signals. The front-end electronics decode this timing signal back into the three original signals. The MFC has three modes of operation: SCL mode, Sequencer mode and TFC mode. In the SCL mode,

---

\* These documents can be found at the following URL: [http://www-d0.fnal.gov/muon\\_electronics/](http://www-d0.fnal.gov/muon_electronics/)

timing and control signals are received from the TFW. In Sequencer mode, the MFC generates timing signals internally synchronized with an on-board 53 MHz quartz oscillator. In the TFC mode, the MFC receives timing and control signals via HOTLink receiver from the TFC. Specific details of the new design are described in the following paragraphs.

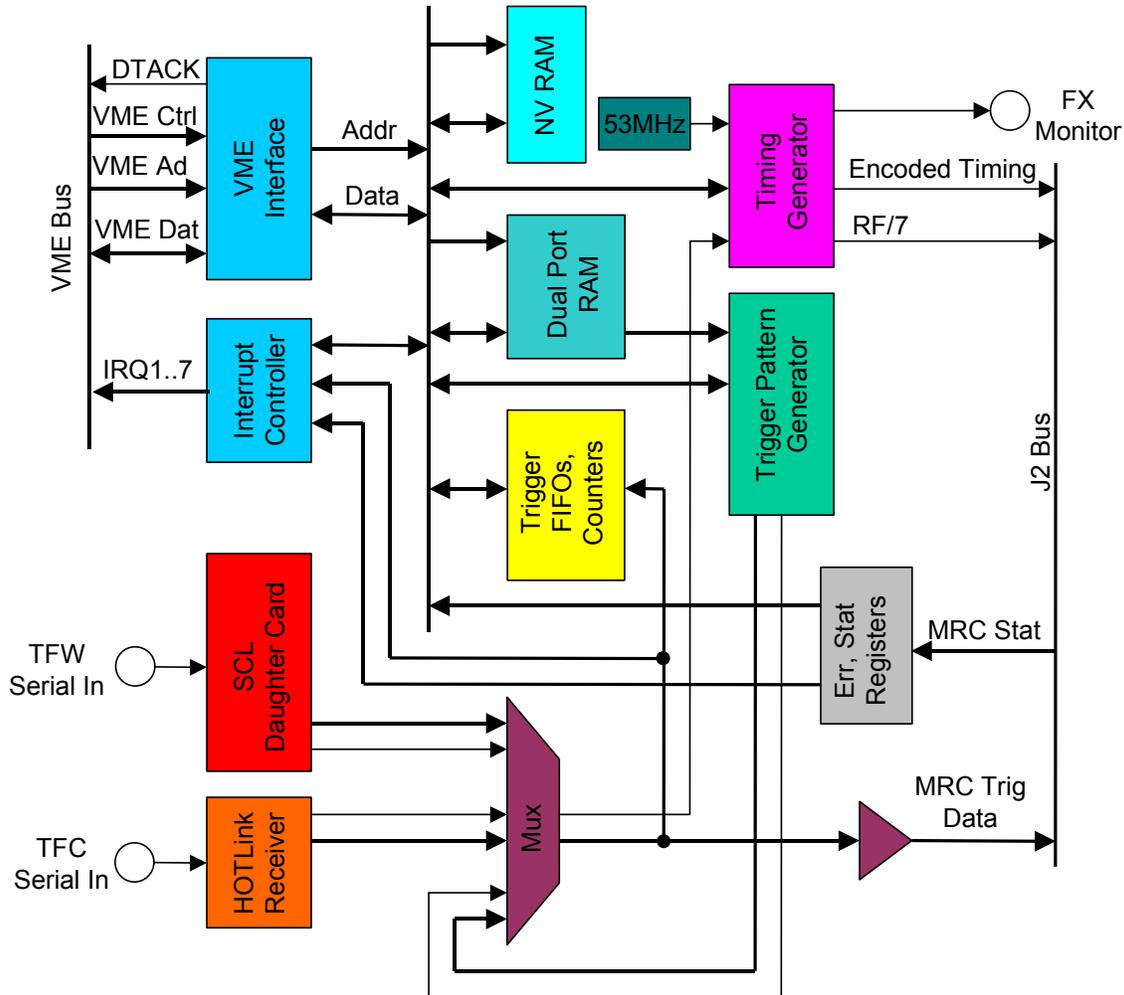


Fig. 1 Simplified block-diagram of the MFC

### Clock generation

Clock generator is simplified in the new design ( Fig. 2). A 53 MHz voltage controlled oscillator is synchronized to the RF/7 signal received from the SCL or TFC. This eliminates possible phase ambiguities between incoming timing signals and the generated encoded timing signal. The internal frequency multiplier of the ALTERA® EP1K30 FPGA chip is used instead of an external one. This excludes any 106 MHz signals from the board and eliminates possible crosstalk and distortion of these signals running on the PCB. A 5/7RF clock signal of the HOTLink receiver is also used as the master clock signal for the entire module and is generated by a new IDT5V995 3.3V clock driver chip.

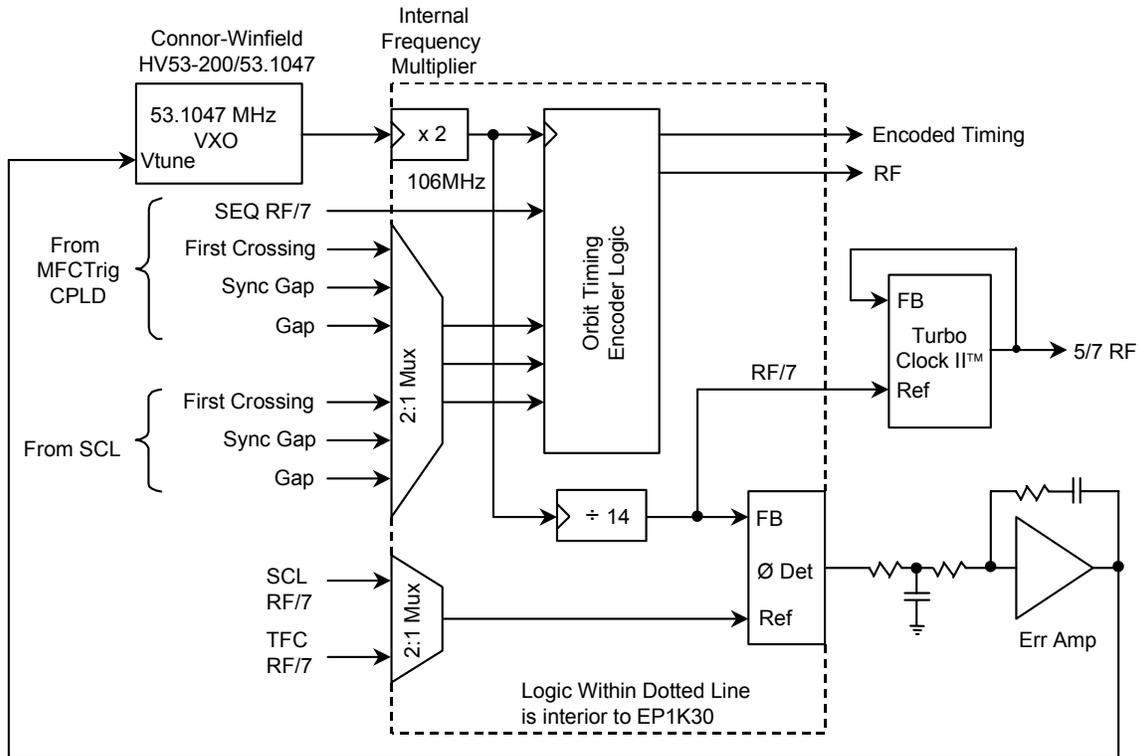


Fig. 2 Details of the clock generator and timing signals encoder

## FIFO memories and trigger counters

FIFO memories are implemented using internal memory blocks of the ALTERA FPGA. The L1 FIFO and Sim (simulated) L2 FIFO are used to check consistency of the TFW decisions. This is done by comparing the current L2 decision information in the L2 FIFO with the output of the Sim L2 FIFO. Every L1 Accept issued by the TFW must be sequentially followed by the L2 decision with identical crossing and turn numbers. It could be either an L2 Accept or L2 Reject. Since the trigger number stored in the L1 FIFO is automatically transferred to the Sim L2 FIFO by L2 Accept, it should match the current L2 trigger number stored in the L2 FIFO. In case of L2 Reject, the content of the L1 FIFO is read by the L2 decision signal, but not transferred to the Sim L2 FIFO. All three FIFO memories are 16 bits wide and *require two VME reads* to retrieve both crossing and turn number of the appropriate decision. The event transfer number is stored in the XferNum FIFO and has to be attached to the current event by the VME processor. Finally, the MFC history FIFO stores all the decisions for diagnostic purpose. This FIFO memory is a circular buffer, which stores the most recent 128 trigger decisions. The data formats of the FIFOs are described in later sections of this note. For diagnostic purposes the FIFOs can be written to and read from by VME.

There are three diagnostic counters and two sets of registers implemented in the MFC. One counter is used for each L1 Accept, L2 Accept and L2 Reject decisions. The counters maintain current trigger decision counts. The appropriate counter is incremented every time a decision is received. The other two sets are registered copies of the current counts latched by the Error 1 or Error 2 signals respectively. The content of these

registers does not change after the corresponding error signal is set. Details of the data formats of the counters are presented later.

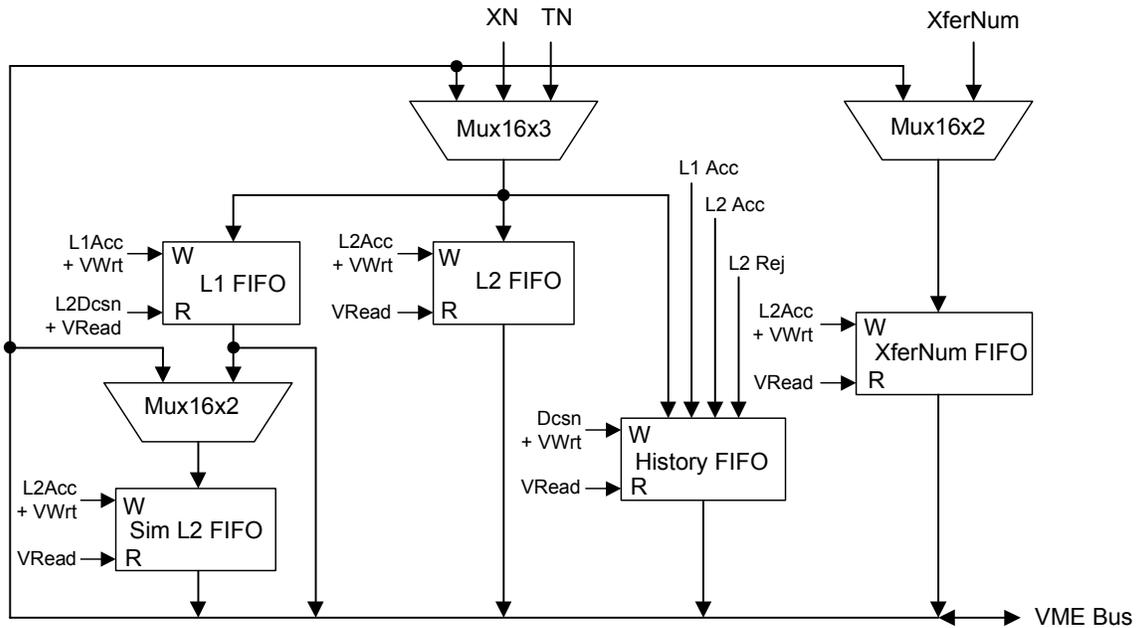


Fig. 3 Details of internal FIFO memories

### Interrupt controller

The interrupt controller is implemented completely in an ALTERA FPGA chip as opposed to the separate chip implementation of the previous design. This provides greater flexibility and convenience for the user. The interrupt controller is designed according to the VME specification and is fully programmable via VME. A selection of the type of interrupt sensitivity (Edge/Level) is fixed for certain signals, but most of the interrupts have this feature as a selectable option. There are a total of fourteen interrupts with pre-set priority levels. The priority level of a particular interrupt can be changed, if necessary, by re-compiling interrupt controller logic.

One channel of the interrupt controller input logic is shown in Fig. 4.

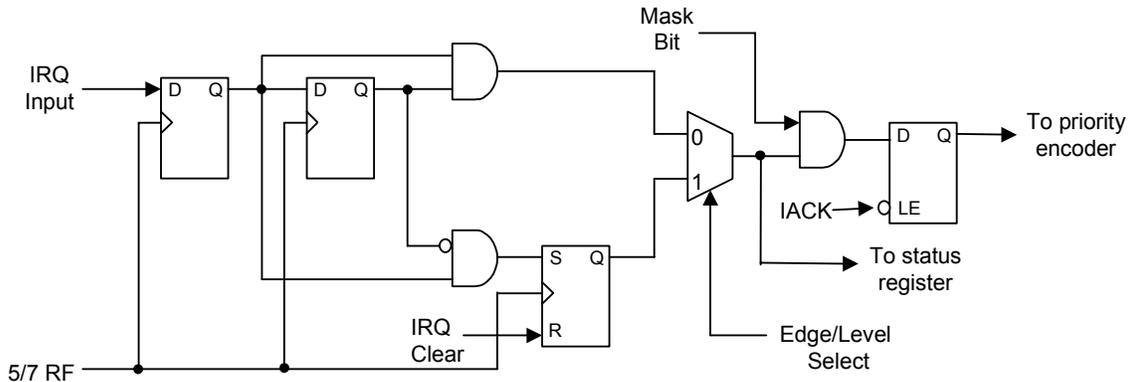
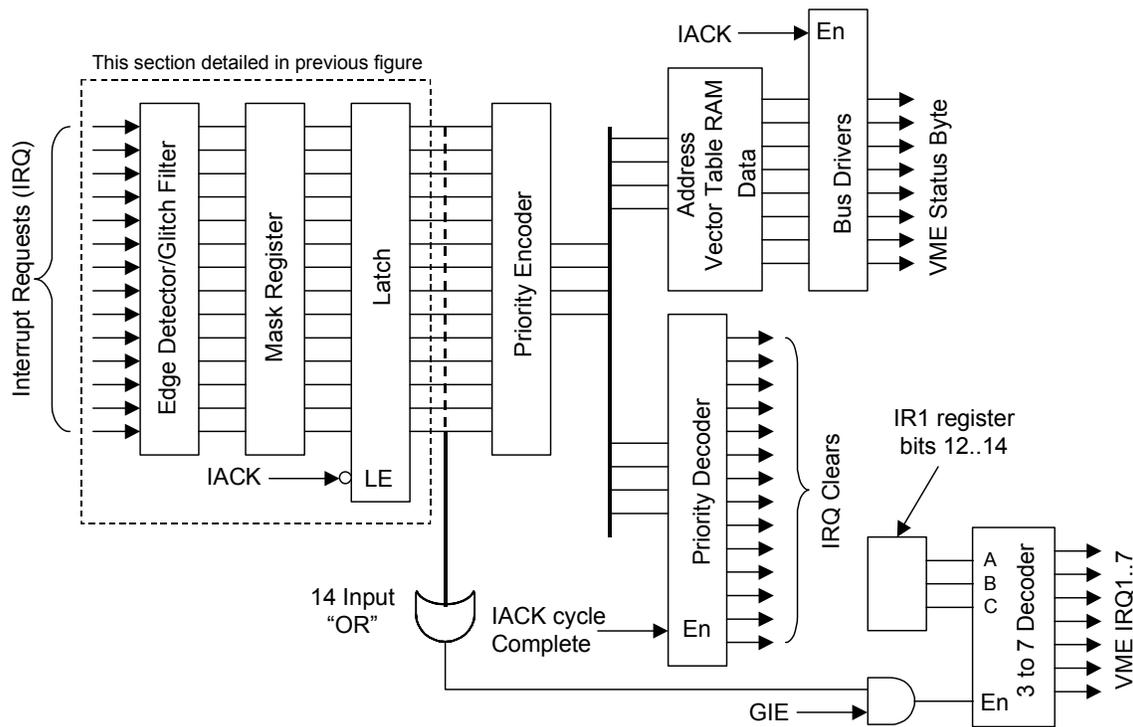


Fig. 4 One channel of the interrupt controller logic

An input interrupt signal is processed by two D-type flip-flops to eliminate glitches and detect positive edge. After two clock cycles the output of the AND gate is stable for the duration of the input level and can be used as a level interrupt. When there is a positive transition at the IRQ input, the output of the second gate will generate a short (one clock cycle wide) pulse that will set the RS flip-flop. The output of this flip-flop can be used as an edge interrupt. The Edge/Level select bit determines which signal is used for the interrupt. All interrupts can be masked by individual bit controls or global interrupt enable bit (GIE). When the VME processor generates an interrupt acknowledge cycle, the interrupt levels are latched to prevent corruption of the interrupt vector. This may happen if another higher level interrupt is generated during the interrupt acknowledge cycle. A full block diagram of the interrupt controller is shown in **Fig. 5**.



**Fig. 5** Block diagram of the interrupt controller

The highest priority interrupt signal is encoded into a binary vector and sent to the address lines of the Vector Table RAM. Any 8-bit vector value can be assigned to the particular interrupt. A VME interrupt line IRQ1..IRQ7 is user selectable via the interrupt controller register IR1 (see **MFC registers**). At the completion of the interrupt acknowledge cycle, the global interrupt enable (**GIE**) bit is cleared and must be re-enabled by the user interrupt routine. If a higher level interrupt occurs before the interrupt handler begins its acknowledge cycle, a new interrupt vector value will be used instead of the one caused the interrupt cycle. The original lower level interrupt will be processed after the higher level interrupt is complete. Pending interrupts can be monitored and individually cleared from VME by accessing the interrupt status register **IR3**. The content of this register is not affected by the mask register settings. Details of the interrupt controller registers can be found in later sections of this note.

## Summary of new features

- Bit assignment and register addressing is re-organized for better clarity and simplification of use (see **Table I**)
- New interrupt controller is implemented using ALTERA FPGA chip (see **Interrupt controller**)
- Arbitrary interrupt vector for each interrupt is implemented using look-up-table RAM
- List of interrupts and priorities is modified (see **MFC registers**)
- Module ID register is implemented at the address \$C00E for software compatibility (see **MFC registers**)
- VBD DONE signal processing is modified: leading edge of this signal resets SLAVE READY bit
- INIT signal processing is modified: INIT signal issued to the front-ends only after INIT enable bit is set by the VME processor
- Added VBDN, L1A, L2A, L2R, INIT, BSY1, BSY2 and RVS Light Emitting Diodes on the front panel (see **Mechanical Specification**)

## Trigger pattern memory data format

The pattern memory is accessible via standard VME commands as a 32 Kbyte memory block. The first 16-bit word (at the even memory address) defines the crossing number, the second defines the turn number of the pointer. The pointer has to be set *one crossing earlier* than the time at which a decision is to be generated. The third and fourth 16-bit words define the trigger number and trigger bits of the decision. An additional control bit is added to the third word to simplify memory data decoding. The end of the valid data is marked by a terminator word with bit 11 set to one. There are no checks implemented in the hardware to prevent invalid patterns. More than one trigger bit set simultaneously (L1A, L2A and L2R) or invalid crossing number (more than 159) creates an invalid pattern. This validation of the data must be performed by the software when downloading pattern memory.

- *Pointer crossing number* (relative address = 0)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	X7	X6	X5	X4	X3	X2	X1	X0

X7..X0

- Pointer crossing number

- *Pointer turn number* (relative address = 2)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

T15..T0 - Pointer turn number

- *Event crossing number* (relative address = 4)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	L2R	L2A	L1A	0	0	0	0	X7	X6	X5	X4	X3	X2	X1	X0

XP7..XP0 - Event crossing number

L1A - L1 Accept bit (1 – true)

L2A - L2 Accept bit (1 – true)

L2R - L2 Reject bit (1 - true)

- *Event turn number* (relative address = 6)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

T15..T0 - Event turn number

- *Terminator word* (even relative address)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	TM	0	0	0	0	0	0	0	0	0	0	0

TM - Terminator bit (1 – true)

## MFC registers

The MFC requires initialization from VME. The following is a list of the internal registers accessible via standard **A24:D16** VME commands (the relative hex address shown). By default, all registers support reads and writes unless noted otherwise.

- *MFC control register 0, CR0 (\$A000)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	NC	NC	INE	FCR	GIR	BRS	SRD	SC2	SC1	IAC	ACK	GIE	MD1	MD0	NVE

NVE - NVRAM Write Enable (1 – enable)

MD0..MD1 - MFC mode select, 01 – SCL, 02 – sequencer, 03 – TFC (default – sequencer mode)

GIE - Interrupt controller enable (1 – enable)

ACK - SCL Acknowledge (0 – off, 1 – on)

IAC - INIT Acknowledge (0 – off, 1 – on)

SC1 - SCL spare 1 control bit (0 – off, 1 – on)

SC2 - SCL spare 2 control bit (0 – off, 1 – on)

SRD<sup>1)</sup> - Set VBD Slave Ready (write 1)

BRS - Internal MFC Reset (write 1)

- GIR - Reset pending interrupts (write 1)  
 FCR - Reset FIFO and trigger counters (write 1)  
 INE<sup>ii)</sup> - INIT enable bit (0 – off, 1 – on)

<sup>i)</sup> VBD Slave Ready is reset by the leading edge of the VBD DONE signal

<sup>ii)</sup> INE is set to zero by the trailing edge of INIT signal in SCL and TFC modes

▪ *MFC control/status register 1, CSR1 (\$A002)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E2E	E1E	B2E	B1E	E2T	E1T	B2T	B1T	E2C	E1C	B2C	B1C	L2E	L1E	L2B	L1B

- L1B - J2 Level 1 Busy status (read-only)  
 L2B - J2 Level 2 Busy status (read-only)  
 L1E - J2 Level 1 Error status (read-only)  
 L2E - J2 Level 2 Error status (read-only)  
 B1C - Level 1 Busy control bit (1 – set active)  
 B2C - Level 2 Busy control bit (1 – set active)  
 E1C - Level 1 Error control bit (1 – set active)  
 E2C - Level 2 Error control bit (1 – set active)  
 B1T - Level 1 Busy status sent to TFW (read-only)  
 B2T - Level 2 Busy status sent to TFW (read-only)  
 E1T - Level 1 Error status sent to TFW (read-only)  
 E2T - Level 2 Error status sent to TFW (read-only)  
 B1E - Busy 1 enable bit (1 – enable)  
 B2E - Busy 2 enable bit (1 – enable)  
 E1E - Error 1 enable bit (1 – enable)  
 E2E - Error 2 enable bit (1 – enable)

▪ *MFC status register 2, SR2 (\$A004)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	NC	NC	ERR	NC	NC	FIN	INIT	NC	NC	SP2	SP1	SL	DER	SER	RDY

- RDY - SCL Ready (read-only)  
 SER - SCL Sync Error (read-only)  
 DER - SCL Data Error (read-only)  
 SL - SCL Sync Lost (read-only)  
 SP1 - SCL spare 1 status bit (read-only)  
 SP2 - SCL spare 2 status bit (read-only)  
 INIT - INIT signal (read-only)  
 FIN - Front-end INIT (read-only)

ERR - Internal MFC error (read-only)

▪ *MFC control/status register 3, CSR3 (\$A006)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	NC	NC	NC	NC	NC	RSQ	INIT	NC	NC	NC	NC	NC	NC	SST	SEN

SEN - Enable internal sequencer (1 – enable)

SST - Start one pass trigger sequence (write 1)

INIT - Set INIT signal level (0 – low, 1 – high)

RSQ - Reset sequencer (write 1)

▪ *MFC control/status register 4, CSR4 (\$A008)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	NC	BST	BSS	NC	NC	BON	BSE	RVS	NC	RFT	RFS	NC	NC	HLR	REN

REN - Start HOTLink reframe cycle (write 1)

HLR - Reset HOTLink error bits (write 1)

RFS - HOTLink re-framed OK (read-only)

RFT - Reframe timeout (read-only)

RVS - HOTLink received violation symbol (read-only)

BSE - Start HOTLink BIST test cycle (write 1)

BON - Continuous BIST test (0 – off, 1 – on)

BSS - BIST cycle finished OK (read-only)

BST - BIST timeout (read-only)

▪ *MFC status register 5, SR5 (\$A00A)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	NC	NC	NC	L2QF	L2DF	L2SF	L1AF	NC	NC	NC	HSFE	L2QE	L2DE	L2SE	L1AE

L1AE - L1ACC FIFO Empty flag

L2SE - L2SIM FIFO Empty flag

L2DE - L2DIR FIFO Empty flag

L2QE - L2DAQ FIFO Empty flag

HSFE - TFWHS FIFO Empty flag

L1AF - L1ACC FIFO Full flag

L2SF - L2SIM FIFO Full flag

L2DF - L2DIR FIFO Full flag

L2QF - L2DAQ FIFO Full flag

- *LIACC\_RX1 register (\$A010)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0

XN0..XN7 - most recent L1 Accept crossing number before Error1 occurred

- *LIACC\_RT1 register (\$A012)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

T0..T15 - most recent L1 Accept turn number before Error1 occurred

- *LIACC\_RX2 register (\$A014)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0

XN0..XN7 - most recent L1 Accept crossing number before Error2 occurred

- *LIACC\_RT2 register (\$A016)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

T0..T15 - most recent L1 Accept turn number before Error2 occurred

- *L2ACC\_RX1 register (\$A018)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0

XN0..XN7 - most recent L2 Accept crossing number before Error1 occurred

- *L2ACC\_RT1 register (\$A01A)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

T0..T15 - most recent L2 Accept turn number before Error1 occurred

- *L2ACC\_RX2 register (\$A01C)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0

XN0..XN7 - most recent L2 Accept crossing number before Error2 occurred

- *L2ACC\_RT2 register (\$A01E)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

T0..T15 - most recent L2 Accept turn number before Error2 occurred

- *L2REJ\_RX1 register (\$A020)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0

XN0..XN7 - most recent L2 Reject crossing number before Error1 occurred

- *L2REJ\_RT1 register (\$A022)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

T0..T15 - most recent L2 Reject turn number before Error1 occurred

- *L2REJ\_RX2 register (\$A024)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0

XN0..XN7 - most recent L2 Reject crossing number before Error2 occurred

- *L2REJ\_RT2 register (\$A026)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

T0..T15 - most recent L2 Reject turn number before Error2 occurred

- *Interrupt mask register 0, ICR0 (\$B000)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	FEF	FFF	L2B	L1B	L2E	L1E	SLT	ITR	ILE	SRQ	L2R	L2A	L1A	VBD

VBD - Enable VBD DONE leading edge interrupt (IR0, 1 – enable)

L1A - Enable L1ACC leading edge interrupt (IR1, 1 – enable)

L2A<sup>ii)</sup> - Enable L2ACC level interrupt (IR2, 1 – enable)

L2R - Enable L2REJ leading edge interrupt (IR3, 1 – enable)

SRQ<sup>i)</sup> - Enable SRQ interrupt (IR4, 1 – enable)

ILE - Enable INIT leading edge interrupt (IR5, 1 – enable)

ITR - Enable INIT trailing edge interrupt (IR6, 1 – enable)

- SLT<sup>i)</sup> - Enable SCL error interrupt (IR7, 1 – enable)
- L1E<sup>i)</sup> - Enable L1 Error interrupt (IR8, 1 – enable)
- L2E<sup>i)</sup> - Enable L2 Error interrupt (IR9, 1 – enable)
- L1B<sup>i)</sup> - Enable L1 Busy interrupt (IR10, 1 – enable)
- L2B<sup>i)</sup> - Enable L2 Busy interrupt (IR11, 1 – enable)
- FFF<sup>i)</sup> - Enable FIFO Full Flag interrupt (IR12, 1 – enable)
- FEF<sup>i)</sup> - Enable FIFO Empty Flag interrupt (IR13, 1 – enable)

<sup>i)</sup> Both edge and level selections are available if not specified

<sup>ii)</sup> L2ACC FIFO empty flag generates IR2 interrupt

- *Interrupt control register 1, ICR1 (\$B002)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	IL2	IL1	IL0	NC	NC	NC	NC	IC7	IC6	IC5	IC4	IC3	IC2	IC1	IC0

- IC0 - IR4 Level/Edge sensitivity select bit (0 – level, 1 – edge)
- IC1 - IR7 Level/Edge sensitivity select bit (0 – level, 1 – edge)
- IC2 - IR8 Level/Edge sensitivity select bit (0 – level, 1 – edge)
- IC3 - IR9 Level/Edge sensitivity select bit (0 – level, 1 – edge)
- IC4 - IR10 Level/Edge sensitivity select bit (0 – level, 1 – edge)
- IC5 - IR11 Level/Edge sensitivity select bit (0 – level, 1 – edge)
- IC6 - IR12 Level/Edge sensitivity select bit (0 – level, 1 – edge)
- IC7 - IR13 Level/Edge sensitivity select bit (0 – level, 1 – edge)
- IL0..IL2 - VME interrupt level (1..7 - valid, 0 - invalid)

- *Interrupt status register 2, ICR2 (\$B004)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IR15	IR14	IR13	IR12	IR11	IR10	IR9	IR8	IR7	IR6	IR5	IR4	IR3	IR2	IR1	IR0

- IR0..IR15 - Pending interrupt requests status (read only)
- IR0..IR15 - Clear selected interrupt request (write 1)

- *Interrupt control register 3, ICR3 (\$B010)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0							

- IV0..IV7 - Interrupt vector 0

- *Interrupt control register 4, ICR4 (\$B012)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0							

IV0..IV7 - Interrupt vector 1

- *Interrupt control register 5, ICR5 (\$B014)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0							

IV0..IV7 - Interrupt vector 2

- *Interrupt control register 6, ICR6 (\$B016)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0							

IV0..IV7 - Interrupt vector 3

- *Interrupt control register 7, ICR7 (\$B018)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0							

IV0..IV7 - Interrupt vector 4

- *Interrupt control register 8, ICR8 (\$B01A)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0							

IV0..IV7 - Interrupt vector 5

- *Interrupt control register 9, ICR9 (\$B01C)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0							

IV0..IV7 - Interrupt vector 6

- *Interrupt control register 10, ICR10 (\$B01E)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0							

IV0..IV7 - Interrupt vector 7

- *Interrupt control register 11, ICR11 (\$B020)*

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0							

IV0..IV7 - Interrupt vector 8

- *Interrupt control register 12, ICR12 (\$B022)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0

IV0..IV7 - Interrupt vector 9

- *Interrupt control register 13, ICR13 (\$B024)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0

IV0..IV7 - Interrupt vector 10

- *Interrupt control register 14, ICR14 (\$B026)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0

IV0..IV7 - Interrupt vector 11

- *Interrupt control register 15, ICR15 (\$B028)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0

IV0..IV7 - Interrupt vector 12

- *Interrupt control register 16, ICR16 (\$B02A)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0

IV0..IV7 - Interrupt vector 13

- *Module ID Register, ID0 (\$C00E)*

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	SN7	SN6	SN5	SN4	SN3	SN2	SN1	SN0

SN0..SN7 - Module serial number

ID0..ID7 - Module version number (\$22)

## MFC diagnostic FIFO memories

- *L1ACC\_XN FIFO* (\$C800)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0

XN0..XN7 - L1 Accept crossing number

- *L2SIM\_XN FIFO* (\$C802)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0

XN0..XN7 - L2 Simulated crossing number

- *L1ACC\_TN FIFO* (\$C804)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

T0..T15 - L1 Accept turn number

- *L2SIM\_TN FIFO* (\$C806)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

T0..T15 - L2 Simulated turn number

- *L2DIR\_XN FIFO* (\$C808)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0

XN0..XN7 - direct from the TFW L2 Accept crossing number

- *L2DIR\_TN FIFO* (\$C80A)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	NC	NC	NC	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0

T0..T15 - direct from the TFW L2 Accept turn number

- *L2DIR\_AQ FIFO* (\$C80C)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
R15	R14	R13	R12	R11	R10	R9	R8	R7	R6	R5	R4	R3	R2	R1	R0

R0..R15 - L2 Accept event transfer number

- *TFWHS\_XT FIFO* (\$C80E)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
NC	NC	NC	NC	NC	L2R	L2A	L1A	XN7	XN6	XN5	XN4	XN3	XN2	XN1	XN0
T15	T14	T13	T12	T11	T10	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

XN0..XN7 - L1/L2 Accept or L2 Reject crossing number

L1A,L2A,L2R - trigger decision bits

T0..T15 - L1/L2 Accept or L2 Reject turn number

Note: Both crossing and turn number have to be read out from the FIFO memory to function properly

### MFC diagnostic counters

- *L1ACC\_ER1 counter* (\$C810)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
N15	N14	N13	N12	N11	N10	N9	N8	N7	N6	N5	N4	N3	N2	N1	N0

N0..N15 - L1 Accept counter value latched by Error1

- *L1ACC\_ER2 counter* (\$C812)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
N15	N14	N13	N12	N11	N10	N9	N8	N7	N6	N5	N4	N3	N2	N1	N0

N0..N15 - L1 Accept counter value latched by Error2

- *L2ACC\_ER1 counter* (\$C814)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
N15	N14	N13	N12	N11	N10	N9	N8	N7	N6	N5	N4	N3	N2	N1	N0

N0..N15 - L2 Accept counter value latched by Error1

- *L2ACC\_ER2 counter* (\$C816)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
N15	N14	N13	N12	N11	N10	N9	N8	N7	N6	N5	N4	N3	N2	N1	N0

N0..N15 - L2 Accept counter value latched by Error2

- *L2REJ\_ER1 counter* (\$C818)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
N15	N14	N13	N12	N11	N10	N9	N8	N7	N6	N5	N4	N3	N2	N1	N0

N0..N15 - L2 Reject counter value latched by Error

- *L2REJ\_ER2 counter* (\$C81A)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
N15	N14	N13	N12	N11	N10	N9	N8	N7	N6	N5	N4	N3	N2	N1	N0

N0..N15 - L2 Reject counter value latched by Error2

- *L1ACC\_GL counter* (\$C81C)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
N15	N14	N13	N12	N11	N10	N9	N8	N7	N6	N5	N4	N3	N2	N1	N0

N0..N15 - current L1 Accept counter value

- *L2ACC\_GL counter* (\$C81E)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
N15	N14	N13	N12	N11	N10	N9	N8	N7	N6	N5	N4	N3	N2	N1	N0

N0..N15 - current L2 Accept counter value

- *L2REJ\_GL counter* (\$C820)

<b>15</b>	<b>14</b>	<b>13</b>	<b>12</b>	<b>11</b>	<b>10</b>	<b>9</b>	<b>8</b>	<b>7</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
N15	N14	N13	N12	N11	N10	N9	N8	N7	N6	N5	N4	N3	N2	N1	N0

N0..N15 - current L2 Reject counter value

## Mechanical Specification

The MFC is a single width VME 9U × 280 mm card. The front panel of the module is shown in **Fig. 6**. The front panel has the following elements:

- Eighteen LEDs indicating the following conditions:
  - ✓ SCL (green) - SCL timing mode
  - ✓ SEQ (green) - SEQ timing mode
  - ✓ TFC (green) - TFC timing mode
  - ✓ SRDY (green) - SCL ready
  - ✓ SERR (red) - SCL error
  - ✓ INIT (green) - INIT signal
  - ✓ SRQ (green) - MRC service request
  - ✓ VBDN (green) - VBD DONE

- ✓ L1A (green) - L1 Accept
- ✓ L2A (green) - L2 Accept



**Fig. 6 MFC front panel**

- ✓ L2R (green) - L2 Reject

- ✓ BSY1 (green) - Level 1 Busy
  - ✓ BSY2 (green) - Level 2 Busy
  - ✓ ERR1 (red) - Level 1 Error
  - ✓ ERR2 (red) - Level 2 Error
  - ✓ RVS (red) - HOTLink RVS error
  - ✓ VME (green) - VME command decoded
  - ✓ +5 (yellow) - +5 V power
- Serial Control Link's RF and multiple pin connector
  - Two LEMO connectors for the following signals:
    - ✓ Current First Crossing (TTL output with 50 ohm series termination)
    - ✓ TFC timing (input, AC terminated with 75 ohm)

### **VME addressing**

There are several address segments in the VME address space that are associated with MFC dual port memory, NVRAM and CSR registers and diagnostic memories. The starting addresses for these segments are listed in **Table I**. The MFC base address (A23..A16) is determined by the DIP switch on the board. The default MFC base address is \$210000. Writes to the NVRAM are disabled on power up and have to be enabled using NVE bit of CR0 when necessary.

**Table I. Muon Fanout Card VME Address Map**

Starting Address	Size (bytes)	Comment
Card Base Address + 000000	32K	Dual Port Memory
Card Base Address + 008000	8K	Non-volatile RAM
Card Base Address + 00A000	2	MFC control register 0
Card Base Address + 00A002	2	MFC control/status register 1
Card Base Address + 00A004	2	MFC status register 2
Card Base Address + 00A006	2	MFC control/status register 3
Card Base Address + 00A008	2	MFC control/status register 4
Card Base Address + 00A00A	2	MFC status register 5
Card Base Address + 00A010	2	L1ACC_RX1 register
Card Base Address + 00A012	2	L1ACC_RT1 register
Card Base Address + 00A014	2	L1ACC_RX2 register
Card Base Address + 00A016	2	L1ACC_RT2 register
Card Base Address + 00A018	2	L2ACC_RX1 register
Card Base Address + 00A01A	2	L2ACC_RT1 register
Card Base Address + 00A01C	2	L2ACC_RX2 register
Card Base Address + 00A01E	2	L2ACC_RT2 register
Card Base Address + 00A020	2	L2REJ_RX1 register
Card Base Address + 00A022	2	L2REJ_RT1 register
Card Base Address + 00A024	2	L2REJ_RX2 register
Card Base Address + 00A026	2	L2REJ_RT2 register

**Table 1. Continued**

Starting Address	Size (bytes)	Comment
Card Base Address + 00B000	2	Interrupt mask register 0
Card Base Address + 00B002	2	Interrupt control register 1
Card Base Address + 00B004	2	Interrupt status register 2
Card Base Address + 00B010	2	Interrupt control register 3
Card Base Address + 00B012	2	Interrupt control register 4
Card Base Address + 00B014	2	Interrupt control register 5
Card Base Address + 00B016	2	Interrupt control register 6
Card Base Address + 00B018	2	Interrupt control register 7
Card Base Address + 00B01A	2	Interrupt control register 8
Card Base Address + 00B01C	2	Interrupt control register 9
Card Base Address + 00B01E	2	Interrupt control register 10
Card Base Address + 00B020	2	Interrupt control register 11
Card Base Address + 00B022	2	Interrupt control register 12
Card Base Address + 00B024	2	Interrupt control register 13
Card Base Address + 00B026	2	Interrupt control register 14
Card Base Address + 00B028	2	Interrupt control register 15
Card Base Address + 00B02A	2	Interrupt control register 16

**Table 1. Continued**

Starting Address	Size (bytes)	Comment
Card Base Address + 00C00E	2	Module ID Register
Card Base Address + 00C800	2	L1ACC_XN FIFO
Card Base Address + 00C802	2	L2SIM_XN FIFO
Card Base Address + 00C804	2	L1ACC_TN FIFO
Card Base Address + 00C806	2	L2SIM_TN FIFO
Card Base Address + 00C808	2	L2DIR_XN FIFO
Card Base Address + 00C80A	2	L2DIR_TN FIFO
Card Base Address + 00C80C	2	L2DIR_AQ FIFO
Card Base Address + 00C80E	2	TFWHS_XT FIFO
Card Base Address + 00C810	2	L1ACC_ER1 counter
Card Base Address + 00C812	2	L1ACC_ER2 counter
Card Base Address + 00C814	2	L2ACC_ER1 counter
Card Base Address + 00C816	2	L2ACC_ER2 counter
Card Base Address + 00C818	2	L2REJ_ER1 counter
Card Base Address + 00C81A	2	L2REJ_ER2 counter
Card Base Address + 00C81C	2	L1ACC_GL counter
Card Base Address + 00C81E	2	L2ACC_GL counter
Card Base Address + 00C820	2	L2REJ_GL counter