

# MIPP Serial Readout Controller Module

Technical description  
(Prototype firmware v.1)

## 1. Introduction

MIPP Serial Readout Controller module (MSRC) is designed to provide a serial interface for Front-End modules using MIPP Data Cable Protocol<sup>[1]</sup>. There are two Front-End modules developed for the MIPP experiment: an 8-channel Scintillation Counter module and a 32-channel EMCal Wire Chamber Front End module<sup>[2]</sup>. Both modules implemented as 6Ux220 mm VME size boards with 48-pin power connector and require a custom europack (VME) 21-slot crate.

## 2. Main features

The Serial Readout Controller Module has the following front panel features:

- External timing input RJ-45 connector compatible with the MIPP Trigger Module<sup>[3]</sup>
- Eight RJ-45 connectors for front-end data chains with incorporated yellow and green LEDs; The green LED indicates presence of a front-end on the chain and yellow LED indicates an error in communication with the front-end
- LED indicators for the absence of the External RF (red) and presence of the VME cycle (green)

The following timing modes of operation are implemented:

- External timing mode - external clock with embedded timing signals from external timing input distributed via eight RJ-45 ports; Additional LVDS signals INIT, AADR and GRST are processed, encoded and distributed by the MSRC via same RJ-45 ports
- Internal timing mode – main clock and timing signals are generated internally in sync with a free running on-board 53.1047 MHz quartz oscillator; Embedded timing signals can be generated by a set of VME commands

The following functionality is implemented in the on-board FPGA firmware:

- VME A24D16 slave interface with D32 BLT and 64-bit MBLT data transfers from internal FIFO memory in each channel
- VME D08(O) programmable interrupter
- 53.1 MHz VCOX controlled by PLL circuit

### 3. MSRC Functional blocks

The MSRC has several independent functional blocks along with a common VME A24D16 slave interface:

- D08(O) interrupter module
- MSRC status and control logic
- Eight data chain interface modules

The following sections describe each module separately.

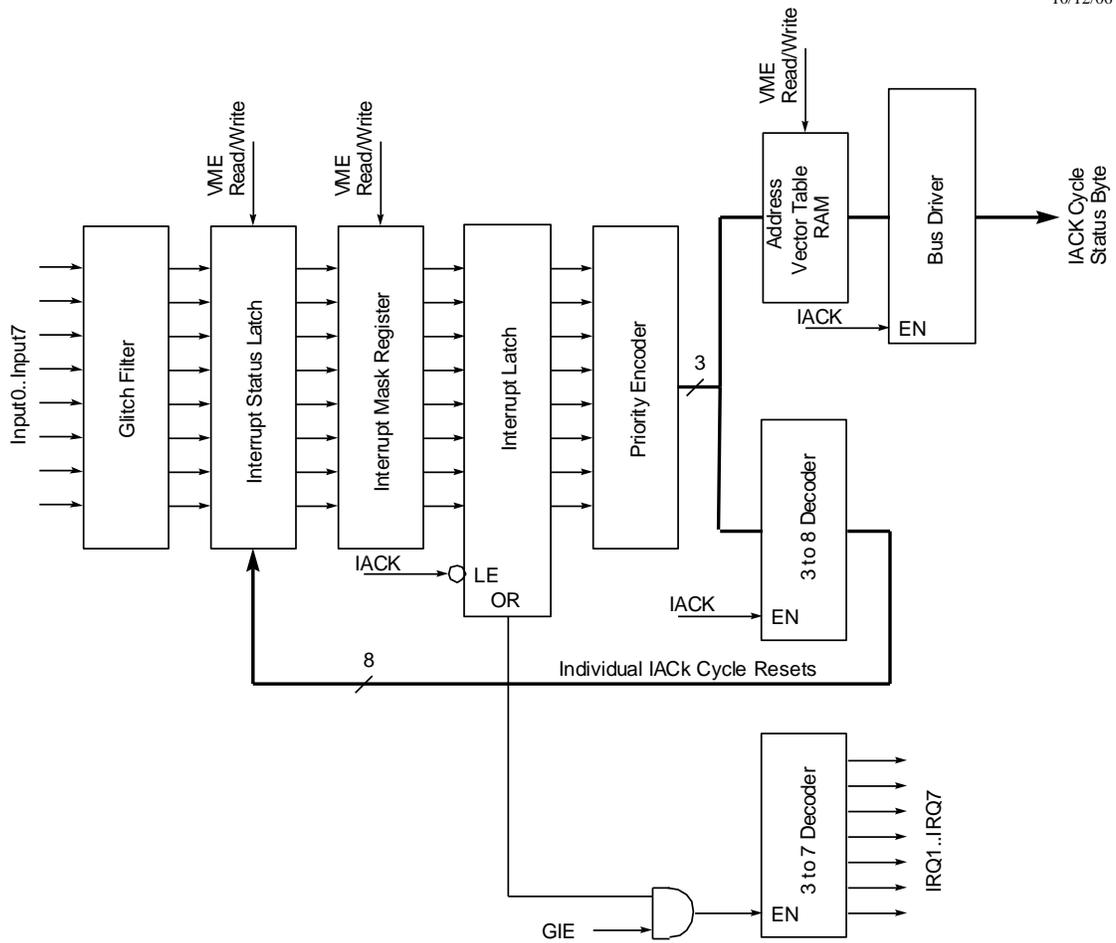
#### ▪ Interrupter module

The interrupter module has eight inputs assigned in the following order:

1. Input 0 - Channel 1 Event Received signal
2. Input 1 - Channel 2 Event Received signal
3. Input 2 - Channel 3 Event Received signal
4. Input 3 - Channel 4 Event Received signal
5. Input 4 - Channel 5 Event Received signal
6. Input 5 - Channel 6 Event Received signal
7. Input 6 - Channel 7 Event Received signal
8. Input 7 - Channel 8 Event Received signal

Input 0 has the highest priority and Input 7 has the lowest. If there are two or more interrupts pending, they are processed in the order of priority. In order to use VME interrupts, the VME master has to have an interrupt handler.

A block diagram of the MSRC Interrupter Logic is shown in Figure 1. On power up, interrupts are disabled, interrupt level is set to five and interrupt vector table is loaded with default values (see table below). When GIE bit is set to one, interrupts are enabled. If any of the interrupt mask register bits is set to one, the corresponding interrupt input is enabled. When input signal sets an interrupt status bit to one, the IRQ line corresponding to the selected interrupt level is driven to zero. If the corresponding software driver is active, an interrupt acknowledge cycle (IACK cycle) is triggered. During IACK cycle the interrupter logic



**Figure 1: Block Diagram of the MSRC Interrupter Logic**

compares three bit vector on address lines set by the interrupt handler with its programmed interrupt level. If there is a match, the interrupter puts an eight bit status byte (status ID) on the data bus and clears corresponding interrupt status latch. After that the GIE bit is set to zero and has to be re-enabled by the software. All interrupter features are programmable via VME bus using the following registers:

### Interrupt mask register IM, 0xF000

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	M7	M6	M5	M4	M3	M2	M1	M0

Note: M0...M7 are individual mask bits corresponding Input 0...7 signals (1 - enable)

### Interrupt status register IS, 0xF010

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	I7	I6	I5	I4	I3	I2	I1	I0

Note: I0...I7 are pending interrupts corresponding Input 0...7 signals (1 - active);  
Writing a one to any of I0...I7 bits will clear corresponding pending interrupt

### Clear pending interrupts register CP, 0xF020

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

Note: Writing **0x81** to this register will clear all pending interrupts

### Interrupt configuration register IC, 0xF040

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	GIE	0	0	0	0	IL2	IL1	IL0

Note: IL2 - IL0 is 3-bit VME interrupt level (1...7 - valid, default - 5),  
GIE - global interrupt enable (1 - enable, 0 - disable, default - 0)

### Status ID vector table VT, Base address 0xF800

Input #	Memory Address	Default Status ID Vector
0	Base + 0x0010	0x08
1	Base + 0x0012	0x09
2	Base + 0x0014	0x0A
3	Base + 0x0016	0x0B
4	Base + 0x0018	0x0C
5	Base + 0x001A	0x0D
6	Base + 0x001C	0x0E
7	Base + 0x001E	0x0F

Note: Value of the status ID vector can be anything from 1 to 255, 0 – invalid

- **MSRC status and control logic**

The status and control logic includes several registers:

**Status/Control register ST, 0xC000**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RF	X	X	X	X	X	X	X	X	X	X	X	X	X	X	TM

Note: TM – timing mode (1 – internal, 0 - external)

RF – RF LED test (1 – ON)

**TADC data register TD, 0xC010**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	X	X	X	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Note: D11..D0 - AD7466 ADC data bits (X - Don't care)

**Start TADC register SS, 0xC020**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	X	1	0	0	0	0	0	0	1

Note: Writing 0x81 starts serial TADC readout (X - Don't care)

**Generate Global Reset register TP, 0xC0E0**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	X	1	1	1	0	0	0	0	0

Note: Writing 0xE0 generates GSR (X - Don't care)

▪ **Data chain interface module**

Each pair of data channels implemented in one common FPGA chip. The following registers control serial interfaces for each channel independently. The channels within the FPGA have relative numbers 0 and 1. Actual channel number can be calculated by adding upper four bits N of the VME address to the relative channel number (where N = 0, 2, 4 or 6) :

**Status register for channel 0 SR0, 0xN010**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A1	A0	CH	RD	NA	NA	NA	RO	CP	EP	OV	DL	CS	TO	FF	EF

Note: EF - FIFO empty flag, EP - Event data parity error  
 FF - FIFO full flag, CP - Command data parity error  
 TO - Command timeout error, RO - Read event timeout error  
 CS - Data checksum error, RD - Event read flag  
 DL - Data length error, CH - Channel number (0)  
 OV - FIFO overflow error, A1,A0 - FPGA address (0..3)

**Status register for channel 1 SR1, 0xN020**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A1	A0	CH	RD	NA	NA	NA	RO	CP	EP	OV	DL	CS	TO	FF	EF

Note: EF - FIFO empty flag, EP - Event data parity error  
 FF - FIFO full flag, CP - Command data parity error  
 TO - Command timeout error, RO - Read event timeout error  
 CS - Data checksum error, RD - Event read flag  
 DL - Data length error, CH - Channel number (1)  
 OV - FIFO overflow error, A1,A0 - FPGA address (0..3)

**Read Event register (write) for channel 0 RD0, 0xN200**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE address								Trigger number							

Note: Trigger number is a copy of the lower eight bits of a full trigger number.

**Read Event register (read) for channel 0 RD0, 0xN200**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Total event length															

Note: Total event length is calculated for one event that is readout.

**Read Event register (write) for channel 1 RD1, 0xN300**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE address								Trigger number							

Note: Trigger number is a copy of the lower eight bits of a full trigger number.

**Read Event register (read) for channel 1 RD1, 0xN300**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Total event length															

Note: Total event length is calculated for one event that is readout.

**Initial AA address for channel 0 register IA0, 0xN400**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Last assigned FE address								Initial AA address							

Note: Only lower eight bits are writable, upper eight bits are updated by the hardware.

**Initial AA address for channel 1 register IA1, 0xN500**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Last assigned FE address								Initial AA address							

Note: Only lower eight bits are writable, upper eight bits are updated by the hardware.

### Front-End address for channel 0 register SA0, 0xN800

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE address								Internal FE address							

Note: The address is stored in MSRC register for later use.

### Write control data to channel 0 register WC0, 0xN802

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Note: The data is written to the Front-End using the address stored in MSRC register.

### Read control data from channel 0 register RC0, 0xN804

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Note: The data is read from the Front-End using the address stored in MSRC register.

### Front-End address for channel 1 register SA1, 0xN900

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FE address								Internal FE address							

Note: The address is stored in MSRC register for later use.

### Write control data to channel 1 register WC1, 0xN902

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Note: The data is written to the Front-End using the address stored in MSRC register.

**Read control data from channel 1 register RC1, 0xN904**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Note: The data is read from the Front-End using the address stored in MSRC register.

**Start channel 0 address assignment register AA0, 0xN806**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	0	0	0	0	A7	A6	A5	A4	A3	A2	A1	A0

Note: A7...A0 - initial address of the first Front-End in the chain

**Start channel 1 address assignment register AA1, 0xN906**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	0	0	0	0	A7	A6	A5	A4	A3	A2	A1	A0

Note: A7...A0 - initial address of the first Front-End in the chain

**Channel 0, 1 data FIFO memory map**

VME Data bits															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DL (0)								DL (1)							
Byte (0)								Byte (1)							
...								...							
Byte (n-1)								Byte (n)							
CHK (0)								CHK (1)							

Note: DL (0), DL (1) – data length, Byte (0) – first byte of the message, Byte (1) – second byte of the message, CHK (0), CHK (1) - checksum.

**Channel 0 data FIFO memory D32 read, 0xNA00**

VME Data bits			
32..24	23..17	16..8	7..0
DL (0)	DL (1)	Byte (0)	Byte (1)

**Channel 1 data FIFO memory D32 read, 0xNB00**

VME Data bits			
31..24	23..16	15..8	7..0
DL (0)	DL (1)	Byte (0)	Byte (1)

**Channel 0 data FIFO memory D64 read, 0xNA00**

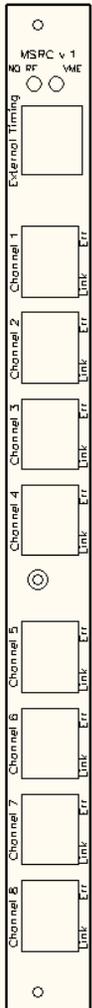
VME Data bits							
63..56	55..48	47..40	39..32	31..24	23..16	15..8	7..0
DL (0)	DL (1)	Byte (0)	Byte (1)	Byte (2)	Byte (3)	Byte (4)	Byte (5)

**Channel 1 data FIFO memory D64 read, 0xNB00**

VME Data bits							
63..56	55..48	47..40	39..32	31..24	23..16	15..8	7..0
DL (0)	DL (1)	Byte (0)	Byte (1)	Byte (2)	Byte (3)	Byte (4)	Byte (5)

## 4. VME module implementation

The current version of the serial readout controller module is implemented as a standard 6Ux160 mm **A24D16** VME slave module. It has the following front panel connectors and indicators (see Fig. 4):



- A single RJ-45 input connector labeled External Timing
- Eight RJ-45 connectors labeled Channel 1 through Channel 8 with incorporated yellow and green LEDs
- Yellow LED indicates an error in communications with the front-end
- Green LED indicates that there is a front-end present on the serial data bus

There is a bi-level LED on the front panel:

- “VME” (green) – VME command decoded
- “NO RF” (red) – missing external RF input signal

**Fig. 4 CRIM front panel**

## 5. Specification

- Power consumption - +5V/0.6A
- Main clock frequency - 53.1047 MHz ( $\pm 200$  ppm)
- External timing inputs - LVDS (see <sup>[3]</sup>)

- MSRC timing outputs - LVDS (see <sup>[1]</sup>)
- VME interrupter type - D08(O)
- VME address modifiers - 38 trough 3f
- VME address/data mode - A24D16, D32 BLT and MBLT
- Channel FIFO size - 512x16 bit
- Data chain cable type - CAT5e
- Maximum cable length - 60 ft.
- Maximum chain data rate - 20 Mbit/s

An eight bit on-board switch selects 64K address space for the module. A general VME memory map is shown in Table 1 and data chain channel map is shown in

Table 2.

**Table 1. MIPP Serial Readout Interface module VME address map**

<b>Starting Address</b>	<b>Size (bytes)</b>	<b>Read/Write</b>	<b>Comment</b>
Base + 000000	6K	R/W	Channel 0,1 address space (N = 0)
Base + 002000	6K	R/W	Channel 2,3 address space (N = 2)
Base + 004000	6K	R/W	Channel 4,5 address space (N = 4)
Base + 006000	6K	R/W	Channel 6,7 address space (N = 6)
Base + 00C000	2	R/W	Status/Control register, ST
Base + 00C010	2	R	Temperature ADC data register, TD
Base + 00C020	2	W	Start TADC register, SD
Base + 00C0E0	2	W	Generate Global Reset register
Base + 00F000	2	R/W	Interrupt mask register, IM
Base + 00F010	2	R/W	Interrupt status register, IS
Base + 00F020	2	W	Clear pending interrupts register, CP
Base + 00F040	2	R/W	Interrupt configuration register, IC
Base + 00F800	16	R/W	Status ID vector table memory, VT

**Table 2. Serial readout interface data chain channel VME address map**

Starting Address	Size (bytes)	Read/Write	Comment
Base + 00N010	2	R	Status register for channel 0, SR0
Base + 00N020	2	R	Status register for channel 0, SR1
Base + 00N200	2	R/W	Read Event register for channel 0, RD0
Base + 00N300	2	R/W	Read Event register for channel 1, RD1
Base + 00N400	2	R/W	Initial AA address for channel 0 register, IA0
Base + 00N500	2	R/W	Initial AA address for channel 1 register, IA1
Base + 00N800	2	W	Front-End address for channel 0 register, SA0
Base + 00N802	2	W	Write control data to channel 0 register, WC0
Base + 00N804	2	R	Read control data from channel 0 register, RC0
Base + 00N806	2	W	Start channel 0 address assignment register, AA0
Base + 00N900	2	W	Front-End address for channel 1 register, SA1
Base + 00N902	2	W	Write control data to channel 1 register, WC1
Base + 00N904	2	R	Read control data from channel 1 register, RC1
Base + 00N906	2	W	Start channel 1 address assignment register, AA1
Base + 00NA00	2	R	Channel 0 FIFO data register (D32 and D64)
Base + 00NB00	2	R	Channel 1 FIFO data register (D32 and D64)

## 6. References

- [1] B.Baldin et al. "MIPP Data Cable Protocol," MIPP Document Database #764-v2, <http://mipp-docdb.fnal.gov>, 17 December 2008.
- [2] B.Baldin and Lou DalMonte. "Scintillation Counter and Wire Chamber Front End Modules for High Energy Physics Experiments," FERMILAB-TM-2483-E, January 2011.
- [3] B.Baldin and Holger Meyer. "Trigger electronics for MIPP upgrade," MIPP Document Database #1034-v1, <http://mipp-docdb.fnal.gov>, 05 August 2010.