Using the FADC Module (Moller) (8/14/09)

1.1 Controlling the Module

Communication with the module is by standard VME bus protocols. All registers and memory locations are defined to be 4-byte entities. The VME slave module has three distinct address ranges.

A24 – The base address of this range is set by a 12-element DIP switch on the board. It occupies 4 Kbytes of VME address space, organized in 1 K 32-bit words. Relative to the base address, this space is divided as follows:

000-0FF – Register space to control and monitor the module (64 long words)

100-1FF – ADC FPGA processing registers (64 long words)

200-2FF – MOLLER FPGA processing registers (64 long words)

300-FFF – Reserved (832 long words)

In addition to registers that are directly mapped to a VME address (Primary Address), the module supports Secondary Addressing in the A24 address space. These registers are accessed through an address mapping register (Secondary Address Register). Each secondary address is associated with a primary address. A Primary Address may have up to 64 K secondary addresses associated with it. A VME cycle loads the mapping register with data which is the internal (secondary) address of the target register. A VME cycle with the associated primary address accesses (read/write) the chosen internal register. Important registers are assigned primary addresses, allowing them to be directly accessible in a single VME cycle. Setup tables are assigned secondary addresses. This allows for a large internal address space, while maintaining a small VME footprint.

A32 - The base address of this range is programmed into register ADR32. It occupies 8 Mbytes of VME address space, organized in 2 M 32-bit words. A read of any address in this range will yield the next FADC data word from the module. Even though the module is logically a FIFO, the expanded address range allows the VME master to increment the address during block transfers. This address range can participate in single cycle, 32-bit block, and 64-bit block reads. The only valid write to this address range is the data value 0x80000000 which re-enables the module to generate interrupts (after one has occurred). The address range must be enabled by setting ADR32[0] = 1.

A32 - The lower and upper limits of this address range are programmed into register ADR_MB. This common address range for a set of FADC modules in the crate is used to implement the Multiblock protocol. By means of token passing FADC data may be read out from multiple FADC modules using a single logical block read. The board possessing the token will respond to a read cycle in this address range with the next FADC data word from that module. The token is passed along a private daisy chain line to the next module when it has transferred all data from a programmed number of events (register BLOCK SIZE). The address range must be enabled by setting ADR_MB[0] = 1.

1.3 Module Registers

VERSION – board/firmware revision (0x0)

[7…0] – (R) – firmware revision

[15…8] – (R) – board revision

[31…16] – (R) – board type ("FADC")

CSR – Control/Status (0x4)

0 – (R) – Event Accepted

1 – (R) – Block of Events Accepted

2 – (R) – Block of Events ready for readout

3 – (R) – BERR Status (1 = BERR asserted)

4 – (R) – Token Status (1 = module has token)

5 – (R) – PLL 1 Locked

6 – (R) – PLL 2 Locked

7 – (R) – PLL 3 Locked

8 – (R) – PLL 1 Loss of Lock Occurred

9 – (R) – PLL 2 Loss of Lock Occurred

10 – (R) – PLL 3 Loss of Lock Occurred

11 – (R) – Data FIFO 1 (channels 1 – 8) Empty Flag Asserted

12 – (R) – Data FIFO 1 (channels 1 – 8) Almost Empty Flag Asserted

13 – (R) – Data FIFO 1 (channels 1 – 8) Half Full Flag Asserted

14 – (R) – Data FIFO 1 (channels 1 – 8) Almost Full Flag Asserted

15 – (R) – Data FIFO 1 (channels 1 – 8) Full Flag Asserted

16 – (R) – Data FIFO 2 (channels 9 – 16) Empty Flag Asserted

17 – (R) – Data FIFO 2 (channels 9 – 16) Almost Empty Flag Asserted

18 – (R) – Data FIFO 2 (channels 9 – 16) Half Full Flag Asserted

19 – (R) – Data FIFO 2 (channels 9 – 16) Almost Full Flag Asserted

20 – (R) – Data FIFO 2 (channels 9 – 16) Full Flag Asserted

21 – (R) – MOLLER FIFO Empty Flag Asserted

22 – (R) – MOLLER FIFO Almost Empty Flag Asserted

23 – (R) – MOLLER FIFO Half Full Flag Asserted

24 – (R) – MOLLER FIFO Almost Full Flag Asserted

25 – (R) – MOLLER FIFO Full Flag Asserted

26 – (R) – Local Bus Time Out – target AK or DK timed out (5 us);

27 – (R/W) – Local Bus Error – target protocol violation;
(write '1' clears latched bits 26, 27)

28 – (W) – Pulse Soft Sync Reset (if CTRL[11] = 1)

29 – (W) – Pulse Soft Trigger (if CTRL[7] = 1)

30 – (W) – Pulse Soft Reset

31 – (W) – Pulse Hard Reset

CTRL1 – Control 1 (0x8)

[2…0] – (R/W) – Sampling Clock Source Select
0 = P2 connector (also 2,4,6)
1 = Front Panel connector (also 3)
5 = P0 connector
7 = Internal Clock

3 – (R/W) – Enable Internal Clock

[6…4] – (R/W) – Trigger Source Select
0 = Front Panel Connector
1 = Front Panel Connector (synchronized internally)
2 = P0 Connector
3 = P0 Connector (synchronized internally)
4 = P2 Connector
5 = P2 Connector (synchronized internally)
6 = VME (software generated)
7 = Module Internal Logic (synchronized internally)

7 – (R/W) – Enable Soft Trigger

[10…8] – (R/W) – Sync Reset Source Select
0 = Front Panel Connector
1 = Front Panel Connector (synchronized internally)
2 = P0 Connector
3 = P0 Connector (synchronized internally)
4 = P2 Connector
5 = P2 Connector (synchronized internally)
6 = VME (software generated)
7 = no source

11 – (R/W) – Enable Soft Sync Reset

12 – (R/W) – Select Live Internal Trigger to Output

13 – (R/W) – Enable Front Panel Trigger Output

14 – (R/W) – Enable P0 Trigger Output

15 – (R/W) – Enable P2 Trigger Output

16 – (R/W) – Enable Readout of ADC channels 1 – 8

17 – (R/W) – Enable Readout of ADC channels 9 – 16

18 – (R/W) – Enable Event Level Interrupt

19 – (R/W) – Enable Moller Front Panel Interrupt Pulse (Data)

20 – (R/W) – Enable BERR response

21 – (R/W) – Enable Multiblock protocol

22 – (R/W) – FIRST board in Multiblock system

23 – (R/W) – LAST board in Multiblock system

24 – (R/W) – Bypass External RAM

25 – (R/W) – Enable Debug Mode

26 – 27 – (reserved)

28 – (R/W) – Multiblock Token passed on P0

29 – (R/W) – Multiblock Token passed on P2

30 – (reserved)

31 – (R/W) – Use internal helicity cycle generator

CTRL2 – Control 2 (0xC)

0 – (R/W) – Go_Transfer - assertion allows data transfer from external FIFOs to control FPGA input FIFOs.

1 – (R/W) – Go_Moller – assertion enables scalers and module data acquisition (if enabled) on the NEXT trailing edge of the *Helicity_Flip* signal. Negation disables scalers and module data acquisition on the NEXT trailing edge of the *Helicity_Flip* signal.

2 – (R/W) – Force_Pause – assertion immediately disables scalers and module data acquisition. Negation enables scalers and module data acquisition provided Go Moller = 1. (Can be used to end run if *Helicity_Flip* signal fails.)

3 – (R/W) – Enable_Internal_Trigger – assertion allows module data acquisition from internal triggers when GO_Moller = 1.

4 – (R/W) – Enable Streaming mode (NO event build)

5 – (R/W) – Select data for readout (0 = channels 1 – 8, 1 = channels 9 – 16)
(streaming mode only)

Bits 16 – 31 are functional only in Debug Mode (CTRL1[25] = 1)

16 – (R/W) – Transfer data: input FIFO → build FIFO

17 – (R/W) – Transfer data: build FIFO → output FIFO

30 – (R/W) – Disable external FIFO output (channels 1 – 8)

31 – (R/W) – Disable external FIFO output (channels 9 – 16)

BLOCK SIZE (0x10)

[15…0] - (R/W) – number of events in a BLOCK.
Stored Event Count ≥ BLOCK SIZE → CSR[3] = 1.

[31…16] – (not used)

INTERRUPT (0x14)

[7…0] – (R/W) – Interrupt ID (vector)

[10…8] – (R/W) – Interrupt Level [2..0]. Valid values = 1,..,7.

11 - 15 – (not used)

[20…16] – (R) – Geographic Address (slot number) in VME64x chassis.

21 – 22 – (not used)

23 – (R) – Parity Error in Geographic Address.

24 – 31 – (not used)

ADR32 – Address for data access (0x18)

0 – (R/W) – Enable 32-bit address decoding

1 – 6 – (not used – read as 0)

[15…7] – (R/W) – Base Address for 32-bit addressing mode (8 Mbyte total)

ADR_MB – Multiblock Address for data access (0x1C)

0 – (R/W) – Enable Multiblock address decoding

1 – 6 – (not used – read as 0)

[15…7] – (R/W) – Lower Limit address (ADR_MIN) for Multiblock access

16 – 22 – (not used – read as 0)

[31…23] – (R/W) – Upper Limit address (ADR_MAX) for Multiblock access

The board that has the TOKEN will respond with data when the VME address satisfies the following condition:

$$ADR\_MIN \leq Address < ADR\_MAX.$$

SEC_ADR – Secondary Address (0x20)

[15…0] – (R/W) – Secondary Address for 24-bit addressing mode

16 – (R/W) – Enable auto-increment mode (secondary address increments by 1 after each access of the associated primary address)

DELAY – Trigger/Sync_Reset Delay (0x24)

[21…16] – (R/W) – Sync reset delay

[5…0] – (R/W) – Trigger delay

INTERNAL TRIGGER CONTROL (0x28)

[23…16] – (R/W) – trigger width (4 ns per count)

[7…0] – (R/W) – trigger hold off delay (4 ns per count)

RESET CONTROL (0x2C)

0 – (W) – Hard reset – Control FPGA

1 – (W) – Hard reset – ADC processing FPGA 1 (channels 1 – 8)

2 – (W) – Hard reset – ADC processing FPGA 2 (channels 9 – 16)

3 – (W) – Hard reset – MOLLER FPGA

4 – (W) – Soft reset – Control FPGA

5 – (W) – Soft reset – ADC processing FPGA 1 (channels 1 – 8)

6 – (W) – Soft reset – ADC processing FPGA 2 (channels 9 – 16)

7 – (W) – Soft reset – MOLLER FPGA

8 – (W) – Reset – ADC data FIFO 1 (channels 1 - 8)

9 – (W) – Reset – ADC data FIFO 2 (channels 9 - 16)

10 – (W) – Reset – MOLLER FIFO (external)

11 – (W) – Reset – DAC (all channels)

12 – (W) – Reset – EXTERNAL RAM Read & Write Address Pointers

13 – (W) – Reset – MOLLER TRIGGER DATA FIFO (internal)

14 – (W) – Reset – MOLLER SCALERS

15 – (W) – Reset – MOLLER SCALER FIFO

16 – (W) – force TOKEN to 1st board of multi-board chain

17 – (W) – force load of INTERNAL TRIGGER BURST parameters

18 – 31 – (not used)

TRIGGER COUNT (0x30)

[31…0] – (R) – total trigger count

31 – (W) – reset count

EVENT COUNT (0x34)

[23…0] – (R) – number of events on board (non-zero → CSR[0] = 1).

[31…24] – (not used)

BLOCK COUNT – (0x38)

[31…20] – not used

[19…0] – (R) - number of event BLOCKS on board (non-zero → CSR[2] = 1).

BLOCK FIFO COUNT – (0x3C)

[31…6] – not used

[5…0] – (R) - number of entries in BLOCK WORD COUNT FIFO

BLOCK WORD COUNT FIFO – (64 deep FIFO) (0x40)

[31…25] – not used (read as '0')

24 – (R) – count not valid (word count FIFO empty)

[23…20] – not used (read as '0')

[19…0] – (R) - number of words in next event BLOCK

INTERNAL TRIGGER COUNT (0x44)

[31…0] – (R) – internal live trigger count

31 – (W) – reset count

EXTERNAL RAM WORD COUNT (0x48)

[31…22] – not used (read as '0')

21 – (R) – RAM empty

20 – (R) – RAM full (1,048,576 eight byte words)

[19…0] – (R) – data word count (eight byte words)

DATA FLOW STATUS (0x4C) (debug use)

DAC 1_2 – DAC channels 1,2 (0x50)

31 – (W) – load DAC channels 1 – 8

[30…24] – (not used – read as 0)

[23…16] – (R/W) – DAC value channel 1

15 – (W) – load DAC channels 1 – 8

[14…8] – (not used – read as 0)

[7…0] – (R/W) – DAC value channel 2

DAC 3_4 – DAC channels 3,4 (0x54)

31 – (W) – load DAC channels 1 – 8

[30…24] – (not used – read as 0)

[23…16] – (R/W) – DAC value channel 3

15 – (W) – load DAC channels 1 – 8

[14…8] – (not used – read as 0)

[7…0] – (R/W) – DAC value channel 4

DAC 5_6 – DAC channels 5,6 (0x58)

31 – (W) – load DAC channels 1 – 8

[30…24] – (not used – read as 0)

[23…16] – (R/W) – DAC value channel 5

15 – (W) – load DAC channels 1 – 8

[14…8] – (not used – read as 0)

[7…0] – (R/W) – DAC value channel 6

DAC 7_8 – DAC channels 7,8 (0x5C)

31 – (W) – load DAC channels 1 – 8

[30…24] – (not used – read as 0)

[23…16] – (R/W) – DAC value channel 7

15 – (W) – load DAC channels 1 – 8

[14…8] – (not used – read as 0)

[7…0] – (R/W) – DAC value channel 8

DAC 9_10 – DAC channels 9,10 (0x60)

31 – (W) – load DAC channels 9 – 16

[30…24] – (not used – read as 0)

[23…16] – (R/W) – DAC value channel 9

15 – (W) – load DAC channels 9 – 16

[14…8] – (not used – read as 0)

[7…0] – (R/W) – DAC value channel 10

DAC 11_12 – DAC channels 11,12 (0x64)

31 – (W) – load DAC channels 9 – 16

[30…24] – (not used – read as 0)

[23…16] – (R/W) – DAC value channel 11

15 – (W) – load DAC channels 9 – 16

[14…8] – (not used – read as 0)

[7…0] – (R/W) – DAC value channel 12

DAC 13_14 – DAC channels 13,14 (0x68)

31 – (W) – load DAC channels 9 – 16

[30…24] – (not used – read as 0)

[23…16] – (R/W) – DAC value channel 13

15 – (W) – load DAC channels 9 – 16

[14…8] – (not used – read as 0)

[7…0] – (R/W) – DAC value channel 14

DAC 15_16 – DAC channels 15,16 (0x6C)

31 – (W) – load DAC channels 9 – 16

[30…24] – (not used – read as 0)

[23…16] – (R/W) – DAC value channel 15

15 – (W) – load DAC channels 9 – 16

[14…8] – (not used – read as 0)

[7…0] – (R/W) – DAC value channel 16

STATUS 1 – Input Buffer Status (0x70)

31 – (R) – data buffer (channel 1 – 8) ready for input

30 – (R) – data buffer (channel 1 – 8) input paused

29 – (R) – not used (read as '0')

28 – (R) – data buffer (channel 1 – 8) empty

27 – (R) – data buffer (channel 1 – 8) full

[26…16] – (R) – data buffer (channel 1 – 8) word count

15 – (R) – data buffer (channel 9 – 16) ready for input

14 – (R) – data buffer (channel 9 – 16) input paused

13 – (R) – not used (read as ‘0’)

12 – (R) – data buffer (channel 9 – 16) empty

11 – (R) – data buffer (channel 9 – 16) full

[10…0] – (R) – data buffer (channel 9 – 16) word count

STATUS 2 – Build Buffer Status (0x74)

[31…29] – not used (read as ‘0’)

28 – (R) – data buffer ‘A’ empty

27 – (R) – data buffer ‘A’ full

[26…16] – (R) – data buffer ‘A’ word count

[15…13] – not used (read as ‘0’)

12 – (R) – data buffer ‘B’ empty

11 – (R) – data buffer ‘B’ full

[10…0] – (R) – data buffer ‘B’ word count

STATUS 3 – Output Buffer Status (0x78)

[31…30] – not used (read as ‘0’)

29 – (R) – data buffer ‘A’ empty

28 – (R) – data buffer ‘A’ full

[27…16] – (R) – data buffer ‘A’ word count

[15…14] – not used (read as ‘0’)

13 – (R) – data buffer ‘B’ empty

12 – (R) – data buffer ‘B’ full

[11…0] – (R) – data buffer ‘B’ word count

STATUS 4 – (spare) (0x7C)

[31…0] – reserved

AUXILIARY 1 – (spare) (0x80)

[31…0] – reserved

AUXILIARY 2 – (spare) (0x84)

[31…0] – reserved

AUXILIARY 3 – (spare) (0x88)

[31…0] – reserved

AUXILIARY 4 – (spare) (0x8C)

[31…0] – reserved

RAM – Read Address (0x90)

[31…24] – not used

[23…21] – reserved (read as 0)

20 – increment address after data read

[19…0] – read address

RAM – Write Address (0x94)

[31…24] – not used

[23…21] – reserved (read as 0)

20 – increment address after data write
[19…0] – write address

RAM 1 – Data Register (0x98)

[31…0] – data word (bytes 0-3)

RAM 2 – Data Register (0x9C)

[31…0] – data word (bytes 4-7)

BERR Module Count (0xA0)

[31…0] – BERR count (driven by module to terminate data transmission)

BERR Total Count (0xA4)

[31…0] – BERR count (as detected on bus)

Auxiliary Scaler 1 (0xA8)

[31…0] – Total word count FPGA 1 (channel 1-8)

Auxiliary Scaler 2 (0xAC)

[31…0] – Total word count FPGA 2 (channel 9-16)

Auxiliary Scaler 3 (0xB0)

[31…0] – Event header word count FPGA 1 (channel 1-8)

Auxiliary Scaler 4 (0xB4)

[31…0] – Event header word count FPGA 2 (channel 9-16)

Auxiliary Scaler 5 (0xB8)

[31…0] – Event trailer word count FPGA 1 (channel 1-8)

Auxiliary Scaler 6 (0xBC)

[31…0] – Event trailer word count FPGA 2 (channel 9-16)

Module Busy Level (0xC0)

[31] – Force module busy

[30…20] – reserved

[19…0] – Busy level (eight byte words)
(External RAM word count > Busy level → module busy = 1)

SCALER DATA STATUS (0xC4)

[31…16] – not used

15 – (R) – scaler data set available

14 – (R) – scaler read in progress

[13…10] – not used

9 – (R) – scaler data FIFO empty

8 – (R) – scaler data FIFO full

[7…0] – (R) – scaler data FIFO word count

SCALER DATA FIFO – (256 deep FIFO) (0xC8)

While triggers are enabled (CTRL2[1] = 1) each helicity change initiates a read of scaler registers (10) located in the Moller FPGA. To tag the scaler data set, a helicity interval count is written into the FIFO. Five 32-bit scaler quantities are assembled and then written to the FIFO. Six data words comprise the scaler data set:

Word 1: 31 – (R) – helicity state for scaler data set

[30…0] – (R) – helicity interval count (increment by 1 each change)

Word 2: [31 – 0] – (R) – CL*CR Scaler value

Word 3: [31 – 0] – (R) – CL*SL Scaler value
Word 4: [31 – 0] – (R) – CR*SR Scaler value

Word 5: [31 – 0] – (R) – CL*CR*SL*SR Scaler value

Word 6: [31 – 0] – (R) – CL*CR*(SL*SR)delay Scaler value

TRIGGER DATA STATUS (internal FIFO) (0xCC)

[31…16] – not used – (read as 0)

15 – (R) – trigger data internal FIFO empty

[14…11] – not used – (read as 0)

10– (R) – trigger data internal FIFO full

[9…0] – (R) – trigger data internal FIFO word count

INTERNAL TRIGGER BURST CONTROL 1 (0xD0)

[31…7] – reserved

[6…0] – (R/W) – maximum number of triggers allowed in BURST WINDOW. When exceeded, triggers are held off for BUSY PERIOD.

INTERNAL TRIGGER BURST CONTROL 2 (0xD4)

[31…16] – (R/W) – BUSY PERIOD (4 ns/count)

[15…10] – reserved

[9…0] – (R/W) – BURST WINDOW (4 ns/count)

(Note: a write that includes the least significant byte of this register initiates a load of the BURST parameters. The default values may be loaded by writing a '1' to bit 17 of the RESET register.)

Reserved (10 registers) (0xD8 – 0xFC)

The following are ADC processing FPGA registers. See *ADC FPGA Version 1* by Hai Dong for details of these registers.

ADC STATUS 0 – (0x100)

[31…16] – (R) – Status 0, FPGA 1 (channels 1 - 8)

[15…0] – (R) – Status 0, FPGA 2 (channels 9 - 16)

ADC STATUS 1 – (0x104)

[31…16] – (R) – Status 1, FPGA 1 (channels 1 - 8)

[15…0] – (R) – Status 1, FPGA 2 (channels 9 - 16)

ADC CONFIGURATION – (0x108)

[31…16] – (R/W) – Configuration, FPGA 1 (channels 1 - 8)

[15…0] – (R/W) – Configuration, FPGA 2 (channels 9 - 16)

PWT – (0x10C)

[31…16] – (R/W) – PWT, FPGA 1 (channels 1 - 8)

[15…0] – (R/W) – PWT, FPGA 2 (channels 9 - 16)

PL – (0x110)

[31…16] – (R/W) – PL, FPGA 1 (channels 1 - 8)

[15…0] – (R/W) – PL, FPGA 2 (channels 9 - 16)

NSB – (0x114)

[31…16] – (R/W) – NSB, FPGA 1 (channels 1 - 8)

[15…0] – (R/W) – NSB, FPGA 2 (channels 9 - 16)

NSA – (0x118)

[31…16] – (R/W) – NSA, FPGA 1 (channels 1 - 8)

[15…0] – (R/W) – NSA, FPGA 2 (channels 9 - 16)

Spare – (0x11C)

THRESHOLD 1_2 – (0x120)

[31…16] – (R/W) – Channel 1 threshold

[15…0] – (R/W) – Channel 2 threshold

THRESHOLD 3_4 – (0x124)

[31…16] – (R/W) – Channel 3 threshold

[15…0] – (R/W) – Channel 4 threshold

THRESHOLD 5_6 – (0x128)

[31…16] – (R/W) – Channel 5 threshold

[15…0] – (R/W) – Channel 6 threshold

THRESHOLD 7_8 – (0x12C)

[31…16] – (R/W) – Channel 7 threshold

[15…0] – (R/W) – Channel 8 threshold

THRESHOLD 9_10 – (0x130)

[31…16] – (R/W) – Channel 9 threshold

[15…0] – (R/W) – Channel 10 threshold

THRESHOLD 11_12 – (0x134)

[31…16] – (R/W) – Channel 11 threshold

[15…0] – (R/W) – Channel 12 threshold

THRESHOLD 13_14 – (0x138)

[31…16] – (R/W) – Channel 13 threshold

[15…0] – (R/W) – Channel 14 threshold

THRESHOLD 15,16 – (0x13C)

[31…16] – (R/W) – Channel 15 threshold

[15…0] – (R/W) – Channel 16 threshold

PWT Last Address – (0x140)

[31…16] – (R) – PWT last address, FPGA 1 (channels 1 - 8)

[15…0] – (R) – PWT last address, FPGA 2 (channels 9 - 16)

PWT Max Buf – (0x144)

[31…16] – (R) – PWT max buf, FPGA 1 (channels 1 - 8)

[15…0] – (R) – PWT max buf, FPGA 2 (channels 9 - 16)

The following are MOLLER FPGA registers. See *Hall A Moller DAQ Firmware* by Hai Dong for details of these registers.

MOLLER STATUS 0 – (0x200)

[31…16] – not used

[15…8] – (R) – '0x08'

[7…0] – (R) – version number

MOLLER STATUS 1 – (0x204)

[31…16] – not used

[15…0] – (R) – '0x0000'

SCALER LOW CL*CR – (0x208)

[31…16] – not used

[15…0] – (R) – low word of CL*CR scaler

SCALER HIGH CL*CR – (0x20C)

[31…16] – not used

[15…0] – (R) – high word of CL*CR scaler

SCALER LOW CL*SL – (0x210)

[31…16] – not used

[15…0] – (R) – low word of CL*SL scaler

SCALER HIGH CL*SL – (0x214)

[31…16] – not used

[15…0] – (R) – high word of CL*SL scaler

SCALER LOW CR*SR – (0x218)

[31…16] – not used

[15…0] – (R) – low word of CR*SR scaler

SCALER HIGH CR*SR – (0x21C)

[31…16] – not used

[15…0] – (R) – high word of CR*SR scaler

SCALER LOW CL*CR*SL*SR – (0x220)

[31…16] – not used

[15…0] – (R) – low word of CL*CR*SL*SR scaler

SCALER HIGH CL*CR*SL*SR – (0x224)

[31…16] – not used

[15…0] – (R) – high word of CL*CR*SL*SR scaler

SCALER LOW CL*CR*(SL*SR)delay – (0x228)

[31…16] – not used

[15…0] – (R) – low word of CL*CR*(SL*SR)delay scaler

SCALER HIGH CL*CR*(SL*SR)delay – (0x22C)

[31…16] – not used

[15…0] – (R) – high word of CL*CR*(SL*SR)delay scaler

CL ADD LENGTH – (0x230)

[31…6] – not used

[5…0] – (R/W) – number of CL samples to add (min = 3)

CL SUM THRESHOLD LOW – (0x234)

[31…16] – not used

[15…0] – (R/W) – low word of 20-bit CL SUM threshold

CL SUM THRESHOLD HIGH – (0x238)

[31…4] – not used

[3…0] – (R/W) – bits [19..16] of CL SUM threshold

CR ADD LENGTH – (0x23C)

[31…6] – not used

[5…0] – (R/W) – number of CR samples to add (min = 3)

CR SUM THRESHOLD LOW – (0x240)

[31…16] – not used

[15…0] – (R/W) – low word of 20-bit CR SUM threshold

CR SUM THRESHOLD HIGH – (0x244)

[31…4] – not used

[3…0] – (R/W) – bits 19-16 of CR SUM threshold

CL PULSE WIDTH – (0x248)

[31…16] – not used

[15…0] – (R/W) – width of logic pulse generated when CL SUM crosses threshold (width[ns] = 4 * (1 + value))

CR PULSE WIDTH – (0x24C)

[31…16] – not used

[15…0] – (R/W) – width of logic pulse generated when CR SUM crosses threshold (width[ns] = 4 * (1 + value))

SL1 PULSE WIDTH – (0x250)

[31…16] – not used

[15…0] – (R/W) – width of SL1 pulse used in coincidence logic (width[ns] = 4 * (1 + value))

SL2 PULSE WIDTH – (0x254)

[31…16] – not used

[15…0] – (R/W) – width of SL2 pulse used in coincidence logic (width[ns] = 4 * (1 + value))

SL3 PULSE WIDTH – (0x258)

[31…16] – not used

[15…0] – (R/W) – width of SL3 pulse used in coincidence logic (width[ns] = 4 * (1 + value))

SL4 PULSE WIDTH – (0x25C)

[31…16] – not used

[15…0] – (R/W) – width of SL4 pulse used in coincidence logic (width[ns] = 4 * (1 + value))

SR1 PULSE WIDTH – (0x260)

[31…16] – not used

[15…0] – (R/W) – width of SR1 pulse used in coincidence logic

(width[ns] = 4 * (1 + value))

SR2 PULSE WIDTH – (0x264)

[31…16] – not used

[15…0] – (R/W) – width of SR2 pulse used in coincidence logic
(width[ns] = 4 * (1 + value))

SR3 PULSE WIDTH – (0x268)

[31…16] – not used

[15…0] – (R/W) – width of SR3 pulse used in coincidence logic
(width[ns] = 4 * (1 + value))

SR4 PULSE WIDTH – (0x26C)

[31…16] – not used

[15…0] – (R/W) – width of SR4 pulse used in coincidence logic
(width[ns] = 4 * (1 + value))

CL DELAY – (0x270)

[31…8] – not used

[7…0] – (R/W) – additional delay for CL logic pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

CR DELAY – (0x274)

[31…8] – not used

[7…0] – (R/W) – additional delay for CR logic pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

SL1 DELAY – (0x278)

[31…8] – not used

[7…0] – (R/W) – additional delay for SL1 pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

SL2 DELAY – (0x27C)

[31…8] – not used

[7…0] – (R/W) – additional delay for SL2 pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

SL3 DELAY – (0x280)

[31…8] – not used

[7…0] – (R/W) – additional delay for SL3 pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

SL4 DELAY – (0x284)

[31…8] – not used

[7…0] – (R/W) – additional delay for SL4 pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

SR1 DELAY – (0x288)

[31…8] – not used

[7…0] – (R/W) – additional delay for SR1 pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

SR2 DELAY – (0x28C)

[31…8] – not used

[7…0] – (R/W) – additional delay for SR2 pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

SR3 DELAY – (0x290)

[31…8] – not used

[7…0] – (R/W) – additional delay for SR3 pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

SR4 DELAY – (0x294)

[31…8] – not used

[7…0] – (R/W) – additional delay for SR4 pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

SL*SR DELAY – (0x298)

[31…8] – not used

[7…0] – (R/W) – delay for SL*SR(delay) pulse used in coincidence logic
(delay[ns] = 4 * (1 + value))

CR TRIGGER PRESCALE FACTOR – (0x29C)

[31…11] – not used

[10…0] – (R/W) – prescale factor for CR trigger (1 = default, 0 not valid)

CL TRIGGER PRESCALE FACTOR – (0x2A0)

[31…11] – not used

[10…0] – (R/W) – prescale factor for CL trigger (1 = default, 0 not valid)

CL*CR TRIGGER PRESCALE FACTOR – (0x2A4)

[31…11] – not used

[10…0] – (R/W) – prescale factor for CL*CR trigger (1 = default, 0 not valid)

MOLLER FIFO – (0x2A8)

[31…16] – not used

[15…4] – (R) – low 12 bits of accepted trigger count

3 – (R) – helicity state of beam

2 – (R) – CL*CR trigger type

1 – (R) – CR trigger type

0 – (R) – CL trigger type

HIT WINDOW WIDTH – (0x2AC)

[31…8] – not used

[7…0] – (R/W) – width of coincidence window for CL, CR, and CL*CR triggers. The leading edge of any trigger starts the window. Any trigger that arrives while the window is active is latched. The latched data pattern (including the trigger initiating the window) is stored in the MOLLER FIFO. (width[ns] = 4 * (1 + value))