STAR EEMC MAPMT Box

Slow Controls Interface Definitions, Procedures, etc.

7/8/2004

Gerard Visser

Introduction

The STAR MAPMT Box is controlled through a standard STAR HDLC mezzanine board, which may be accessed either via the HDLC link, or for debugging purposes via an RS-232 serial connection[1]. The STAR HDLC mezzanine board is described in STAR note #241; it is based on a 68302 microcontroller with some resident firmware developed at LBL. Slow controls software simply targets this board over the HDLC link, and does not actually execute on the HDLC mezzanine board.

The HDLC mezzanine board communicates to the readout board ("MUX" board) of the MAPMT Box over two paths. The main path is a memory-mapped interface wherein the readout board FPGA is the target of read and/or write accesses in the memory range 0x020000 to 0x03fffe. In actuality, only the lower 16 bits of the address (A16 – A1, that is, A0 is not considered) are decoded on the readout board, together with the "chip-select" signal CS3 which is asserted for STAR-standard address range 0x020000 to 0x0ffffe. Thus the readout board will respond also in the range 0x040000 to 0x0ffffe, six times overlaying the range 0x020000 to 0x03fffe. Software should not make any accesses in these overlaying regions, and should only use the fundamental range 0x020000 to 0x03fffe.

The HDLC mezzanine board (and so the control software) does not access any of the 48 FEE boards directly; rather the readout board mediates in any transaction which ultimately targets the FEE boards, for instance to read event data. All such accesses appear in the memory space described above, which is connected to the readout board FPGA.

The second communication path from the HDLC mezzanine board to the readout board uses the eight I/O bits which are available direct from the microcontroller. These are also controlled over HDLC and/or the RS-232 port, although the protocol is different. Alternatively, for higher speed, the I/O bits can be accessed directly in a register which is internally memory mapped in the 68302 microcontroller, allowing the I/O bits to appear to the MAPMT Box slow controls software as just another memory location. The eight I/O bits are used for the following functions, which are required to work in the absence of a properly configured readout board FPGA:

1. The power trip status bit (indicates if power has tripped off for the main digital logic and the FEE boards, due either to overtemperature or overvoltage).
2. The readout board FPGA configuration status and control bits (~PROGRAM, ~INIT, DONE) which can be used to initiate a re-configuration of the FPGA and to check whether configuration has succeeded.
3. A serial interface to an Atmel AT45DB021B 2-megabit serial flash memory, from which the readout FPGA downloads its configuration data, that of the FEE board FPGA's, and possibly other configuration parameters. The flash memory may be read, written, and cleared through the I/O bit interface from the HDLC mezzanine board, regardless of the state of the readout board FPGA configuration.

This document defines below the protocols and bit and address definitions for controlling and operating the MAPMT FEE Box.

The following stuff is not correct in all detail (had to change hardware connections to HDLC mezzanine board, in order to use HDLC block transfers – I thought this could be made to work with through the built-in I/O bits but that is unfortunately not compatible with the HDLC board firmware). The big picture is the same, but ignore the details in this section for now. The correct details are "documented" in the code smdControl.c and in the schematics.

---

[1] 9600 baud, 8-bit, no parity, 1 stop bit

I/O Bit Definitions

The eight I/O bits (four read bits and four write bits) are defined in the HDLC mezzanine board firmware and described in SN241. They *may* be accessed using a 12-byte HDLC packet with command/function codes 0x09, 0x0a, and 0x0b. According to SN241, the HDLC packet format for these commands is

byte address	*contents*
00	target node ID
02	message ID MSW
04	message ID LSW
06	length count
08	command/function code
0a	bit specification

The bit specification word is not documented in SN241, although it is of course determined by the HDLC mezzanine board firmware. Investigation shows that the bit specification word is must be one of the following eight values:

bit specification	*I/O bit name (ref. SN241)*	*function in MAPMT readout*
8000	OUTBIT4	~PROGRAM (FPGA ~PROGRAM)
4000	OUTBIT3	~CMDIN (CM serial data input)
2000	OUTBIT2	CMCLK (CM serial clock)
1000	OUTBIT1	~CMCS (CM chip select / framing)
0800	INBIT4	DONE (FPGA DONE)
0400	INBIT3	~INIT/~DONE (FPGA ~INIT)
0200	INBIT2	CMDOUT (CM serial data output)
0100	INBIT1	~TRIP (power trip status)

Combinations (such as attempting to set all output bits simultaneously using one HDLC packet with the bit specification word 0xf000) are not allowed.

"CM" refers to the Atmel AT45DB021B confirguration memory, "FPGA" refers to the Xilinx XC2S100 FPGA. The function of the FPGA ~PROGRAM and ~INIT signals is not described here – see the XC2S.. data sheet from Xilinx.

Alternative to the above, the eight I/O bits may be directly at the memory location 0x700822, which is the 68302 "Port A Data Register".[2] This procedure is preferred for the MAPMT Box slow controls software, because of the large amount of data that has to be sent through the I/O bits during a download of new configuration to the configuration memory.[3] The most significant byte of this register corresponds to the eight I/O bits as follows:

15	*14*	*13*	*12*	*11*	*10*	*9*	*8*
~PROGRAM	~CMDIN	CMCLK	~CMCS	DONE	~INIT/~DONE	CMDOUT	~TRIP

The least significant byte corresponds to other functions on the HDLC mezzanine board and should be considered "reserved" from the point of view of the MAPMT Box slow controls software. Their value upon a read is not defined in this specification. *A write to the Port A Data Register shall always have 0xff in the least significant byte.*

The values written to the input bits (bits 11 – 8) are ignored by the 68302 and may have any value. Upon a read, the values of the output bits will reflect the last written value, and the values of the input bits will be determined by the state of the MAPMT Box hardware.

Other addresses internal to the 68302 processor appear in the range 0x700000 to 0x700fff, and care must be taken in the slow controls software not to write to any such locations.

---

[2] See the MC68302 User's Manual, page 2-17; base address for the internal registers is set by the HDLC mezzanine board firmware at 0x700000

[3] Memory-mapped accesses by the slow controls software can send up to 1012 words per packet and can broadcast to all HDLC target nodes, in contrast to the defined I/O bit control accesses which can change only a single bit per packet and cannot broadcast.

Downloading to the Configuration Memory

Dowloading code to the configuration memory is required for maintenance (and of course an initial download is required prior to testing the MAPMT Box). Code download is not used in normal operation, and if this function is included in the MAPMT Box slow controls software it should be on an "experts only" window. It is perhaps better to have the code download functions as a stand alone program. This need not have any GUI interface, and in fact can simply be invoked from the VxWorks shell (so it can simply be a C function).

The code download function needs to be passed the start address (19 bits), and a filename. The filename will contain the code, in Intel (MCS) hex file format. This file format is conveniently generated by the Xilinx development system. The code download function will read data from the file and write it in "pages" of 264 bytes to the configuration memory using the "main memory page program through buffer 1" and "status register read" commands (opcodes 0x82 and 0x57) of the AT45DB021B.

This document will not describe the AT45DB021B operation in detail, instead please refer to the data sheet. The connections between the I/O bits above and the ~CS, SCK, SI, and SO pins of the AT45DB021B on the readout board are as follows:

The ~PROGRAM bit must be set to 0 to connect to the AT45DB021B. After doing this, confirm that the ~INIT/~DONE bit reads 0. If not, there is a hardware error. Once these preparatory steps have been carried out:

- The ~CMDIN bit is sent to the AT45DB021B pin SI through an inverter.
- The CMCLK bit is sent to the AT45DB021B pin SCK.
- The ~CMCS bit is sent to the AT45DB021B pin ~CS.
- The AT45DB021B pin SO is sent to the CMDOUT bit.

At this point, the procedures defined in the AT45DB021B data sheet should be followed to download the code and/or perform a readback.

After the download (and/or readback) is complete, the ~PROGRAM bit should be set to 1. This will cause the readout board FPGA to download its configuration from the configuration memory starting from address 0x000000. If successful, the DONE bit will read 1 and the ~INIT/~DONE bit will read 0 within a small time. If that fails there is either a hardware error or the configuration memory contents are incorrect.

Power-on of the MAPMT Box

At power-on of the MAPMT Box, its HDLC mezzanine board will boot, and at the same time its readout board FPGA will configure itself from the configuration memory and then configure the FEE board FPGA's from (a different starting address in) the configuration memory. No action is needed from the slow controls software for these tasks.

Start-up of Slow Controls Software

Ignore for now – subject to further revision...

The slow controls software may be restarted at any time, as far as the MAPMT Box is concerned at least. Upon starting, the first thing it should do is verify the status of the MAPMT Box hardware:

1. The I/O bits register (most significant byte) should be 111110X1. This confirms that the power is ok, the readout board FPGA is configured, and of course the HDLC board is running.
2. The FEE board FPGA configuration should be checked by reading ...

Voltage, Current, and Temperature Monitors

Various temperatures and voltages in the MAPMT Box can be monitored with a 12-bit ADC. The channel assignments are:

channel	*read address*	*monitor function*	*nominal scale (per ADC unit)*
0	0x023010	+3.3 regulated supply voltage	-4.008016 mV
1	0x023012	+3.3 regulator base drive voltage	-4.008016 mV
2	0x023014	center plate temperature – 0 ºC	-0.1078156 K
3	0x023016	RDO board temperature	-0.2004008 K
4	0x023018	FEE board monitor A (selected FEE temperature)	-0.2004008 K
5	0x02301a	FEE board monitor B (selected FEE temperature)	-0.2004008 K
6	0x02301c	spare	—
7	0x02301e	spare	—
8	0x023100	ground (0V)	-4.008016 mV
9	0x023120	“+6” input current	-2.004008 mA
10	0x023140	“-6” input current	-2.004008 mA
11	0x023160	“+6” input voltage	-4.008016 mV
12	0x023180	+5 regulated supply voltage	-4.008016 mV
13	0x0231a0	“-6” input voltage	-4.008016 mV
14	0x0231c0	“+4” input voltage	-4.008016 mV
15	0x0231e0	“+4” input current	-4.008016 mA

Data is read at the above addresses as a 12-bit word sign-extended to 16 bits. In other words, negative values run from 0xf800 to 0xffff, positive values from 0x0001 to 0x07ff, and zero is of course 0x0000. The ADC chip scale is 1 mV per ADC unit. Either of the spare channels may be read to determine the offset value which can be subtracted then from all the data for greater accuracy.

For each the FEE board temperature monitors A & B, one and only one of the FEE boards in the range 0 – 23 for “A” and 24 – 47 for “B” needs to be enabled to drive their temperature data on the monitor busses “A” and “B” respectively. Note only a few of the FEE boards installed will actually have a temperature sensor. If no FEE board is selected, or if the selected FEE board has no temperature sensor, the temperature will read about 0 K. If multiple FEE boards are selected at the same time, their temperatures (in K) will add together, producing an absurdly high value. No damage can occur to the hardware from such contention on the monitor busses. To enable or disable a particular FEE board for driving the temperature on the monitor bus, a bit needs to be set or cleared in its auxiliary control register (see the section “FEE board registers” below).

Readout Board Control Registers

The registers associated to programming and readback of the configuration memory are described in an earlier section. In this section we describe registers that are control the normal operation of the box (as opposed to the special configuration-memory-programming operations).

The voltage, current, and temperature monitors have already been described above.

The box name register (Table 1) provides storage for a 32-bit “name” for the box. It is intended that the name is a four character ASCII string such as ‘ 3S2’ or ‘11P1’ which is equal to the standard EEMC system name of the particular box on the poletip, used in most EEMC system documentation. At the moment the box name register is volatile and read/write, but a planned upgrade for the FPGA design will make the box name nonvolatile and read-only, with contents programmed through the configuration memory interface.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x022000	*BOX_NAME[31:16]*															
0x022002	*BOX_NAME[15:0]*															

Table 1. Box name register (read/write while under construction – will be read only).

The control and status register (Table 2). The bit assignments are not yet written here. Some functions: Manual LED pattern (to verify physical box you're talking to on the poletip). FEE powerdown control (may go in a new register). Local oscillator vs. TCD mode. PLL lock status bit. Unrecognized TCD command status bit.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x022004	*?*	*?*	*?*	*?*	*?*	*?*	*?*	*?*	*?*	*?*	*?*	*?*	*?*	*?*	*?*	*?*

Table 2. Readout board control & status register.

There are three analog time delay adjustments:

1. The overall box delay which offsets the phase of the RHIC strobe as actually used by the box, with respect to the input from the TCD cable.
2. The reset phase which controls the phase of the reset pulse with respect to the ADC clock.
3. The width of the reset pulse.

The delay control registers are shown in Table 3. These registers are write only – a read will generate a bus error to the 68302. The analog delays are controlled by 10-bit DAC's. (Need here to insert explanation of the delay in terms of DAC setting – will do this later...)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x022010	X						*BOX_DELAY*									
0x022012	X						*RESET_DELAY*									
0x022014	X						*RESET_WIDTH*									

Table 3. Delay control registers (write only).

The pulser to (readout) trigger latency register (Table 4) controls the delay in cycles after receiving a trigger command 8 (and thus sending a test pulse command to the FEE boards), before issuing a readout trigger (as if trigger command 4 were just received when the delay times out). The test pulser operation is described in more detail in a later section of this document. The value of *PT_LATENCY* may be in the range 0 – 127.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x022006	X									*PT_LATENCY*						

Table 4. Pulser to trigger latency register.

The LED to (readout) trigger latency register (Table 5) controls the delay in cycles after receiving a trigger command 9 before issuing a readout trigger (as if trigger command 4 were just received when the delay times out). The value of *LT_LATENCY* may be in the range 0 – 127.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x022008	X									*LT_LATENCY*						

Table 5. LED to trigger latency register.

FEE Board Registers

From the viewpoint of the slow controls interface MAPMT box consists of a readout board and 48 FEE boards (numbered 0 – 47). FEE board *n* provides channels 4*n*, 4*n*+1, 4*n*+2, and 4*n*+3 for the box.

The readout board address space (described above) is 0x022000 – 0x03fffe. The FEE board address space is 0x020000 – 0x021ffe, with 0x000080 per FEE board. In other words the base address of FEE board *n* is 0x020000+*n*×0x000080. All accesses to the FEE boards use **only the lower byte** of the 16-bit word. The upper byte is ignored on a write, and undefined on a read.

The address map for one FEE board is as follows:

address offset	*size*	*Function*
0x000000	59	readout buffer (2nd port – for diagnostics only)
0x000076	1	trigger latency register
0x000078	1	pulser control register MSB
0x00007a	1	pulser control register LSB
0x00007c	1	control and status register
0x00007e	1	ID register

The readout buffer 2nd port is not implemented yet. (This is a 2nd port accessing the readout buffer, which could be used to write test data which will be read out during subsequent readouts. Note that a CSR bit needs to be set to disable normal writing of FEE data into the readout buffer if this test mode is to be utilized.) The readout buffer can also be used as a scratchpad memory for instance to test the data communications from the CPU to the Radstone/HDLC board, through the network, to the HDLC daughterboard, through the readout board, through the MAPMT box data busses and interface boards, to the FEE boards.

The trigger latency register (Table 6) controls how many cycles back in time that the FEE board pulls data from, in response to a trigger input to the MAPMT box. There will be a constant offset *C* (value TBD). In other words, the readout data will correspond to the signals arriving on the optical fibers into the box at *(C + (trig latency register value))* cycles prior to the arrival of the trigger command at the box. The trigger latency register is 7 bits wide. However, the allowed values are 0 through 85, inclusive. The hardware will **not** enforce this limit. Values in the range 86 – 255 may be written (and read back from) the trigger latency register but will cause garbled data readout. The trigger latency register occupies the lower 7 bits of the byte, itself embedded (as usual for FEE board registers) as the lower byte of the 16-bit word.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000076	X								X	*trigger latency (0 – 85)*						

Table 6. FEE board trigger latency register.

The pulser control register (Table 7) is a 16-bit register controlling the test pulser precharge level and channel enables. The use of the test pulser is described in a separate section, below. See also the sweeping pulser control bit in the FEE board CSR, which must be set to 0 for normal operation.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000078	X								*E3*	*E2*	*E1*	*E0*	*Q[11:8]*			
0x00007a	X								*Q[7:0]*							

Table 7. FEE board pulser control register.

The control and status register is shown in Table 8. The *TE* bit enables (if *TE*=1) the FEE board temperature output. At most one board at a time on each of the two halves of the MAPMT box should be enabled for temperature output. Enabling multiple boards for temperature output will result in the sum of temperatures (measured in K) being read. See above under "voltage, current, and temperature monitors". The *RD* bit disables (if *RD*=1) the readout of ADC data from the acquisition buffer to the readout buffer. *[Actually, this function is not yet implemented.]* This is useful for testing, to force a fixed data pattern that has been written into the readout buffer to be presented at the MAPMT box data output in response to a trigger. The *SE* bit indicates an internal clock sync error (between the FEE board and the readout board). *[Not implemented yet.]* The *PP* bit enables (if *PP*=1) the pre-/post-cycle readout mode. *[Not implemented yet.]* The *RP* bit enables (if *RP*=1) the special ramping-pulser test mode (useful for ADC binning and bit tests) – for normal operation it should be 0.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00007c	X								X	X	X	*RP*	*PP*	*SE*	*RE*	*TE*

Table 8. FEE board control and status register.

The ID register **must** be programmed with the FEE board number. This is required for proper operation of the chained block transfer used during the data readout. If, on any of the FEE boards, the ID register value is not equal to the FEE board number, the data readout from the box will be garbled. There is no risk of hardware damage, however.[4] The lower two bits of the ID register are hardwired to the "board ID" bits as set on the associated interface board. The upper two bits of the ID register are hardwired to 00. Only the middle four bits of the ID register will be changed upon a write.

Built-in Charge-injection Test Pulser

On the FEE board there is for each channel a charge-injection test pulser consisting of a 5.6 pF ± 5 % capacitor and a MOSFET switch to dump the capacitor's charge into the FEE input. A single DAC with nominal 0 to 5 V output precharges the test pulse capacitors; the amplitude is common to all four channels of the FEE board. Operation of the test pulser is controlled by the pulser control register (described above) and by a pulser timing signal received from the readout board, which sends pulser and readout trigger signals to the FEE boards in response to a pulser trigger received from the TCD.

In the control register, the enable bits *En* enable the test pulser to channel *n* (1=enable, 0=disable). The 12-bit *Q* value sets the pulser amplitude. When all channel enable bits are cleared, the precharge DAC is also disabled to prevent noise injection into the signals. It is therefore important to keep all enable bits cleared during normal operation, and not merely to rely on a zero amplitude setting or to rely on not issuing the pulser trigger through the TCD input.

Timing of the firing of the test pulser is as follows: The firing time of the test pulser within the cycle is fixed by design, and is set to have the pulse optimally placed (within constraints) with respect to the integration gate. Which cycle the pulser fires on, and which cycle is read out, is controlled as follows:

Upon receiving trigger command code 0x8, the pulser fires "immediately." At the same time, a pulser-readout delay timer is loaded and starts timing. On the cycle on which that times out, a readout is initiated *exactly as if a physics trigger (command code 0x4) were received on that cycle.* The pulser-readout delay is controlled by the pulser-readout latency register on the readout board. It should, of course, be set so that the readout will be for the proper cycle. If the same constant is added or subtracted from both the pulser-readout latency register and the trigger latency registers (on the FEE boards), the operation will be the same (but the overall time required will of course be that much longer or shorter).

If a non-null trigger command is received while the system is busy handling a built-in test pulser trigger command, it will be ignored. (Similar statement applies for all other trigger commands too – must note this elsewhere!)

Note that the L0 trigger busy timer (in the TCD) should be set long enough to cover the pulser-readout delay, in addition to the FEE event readout and the data collector event data store.

External LED Pulser Operation

LED (or other diagnostic) pulsers for the EEMC MAPMT channels are externally timed with respect to a trigger command 0x9 on the TCD bus. (In practice, the LED drivers sit in the tower FEE crates, so we rely on the proper timing relation between the ESMD TCD and the ETOW TCD, e.g., if they are triggered together by the TCU. It will not work to naively trigger only the one or other TCD from the front panel input in local oscillator mode.)

The MAPMT FEE system will decode trigger command 0x9 and handle it exactly as the internal test pulse trigger 0x8, except without firing the internal pulsers, and except that an entirely separate pulser – readout latency register is used. (To conveniently allow for the fact that the proper value to use will be different, in general).

In summary, the philosophy is that the readout-trigger latency register (and the clock phase as controlled both by the ESMD TCD and each box's "box delay" register) is set to put the integration & readout in time with the physics data, and *then* the pulser-trigger latency register, the LED-trigger latency register, the external LED firing delay with respect to trigger command 9, are set to get the charge injection and LED pulser data in time with where the physics data would be.

---

[4] The bus driver chips will still be properly sequenced, as they are under the direct control of the readout board FPGA.