ODMB user’s manual

Optical DAQ MotherBoard for the ME1/1 stations of the CMS muon endcap detector

Firmware tag: 3.12

ODMB.V2, ODMD.V3, and ODMB.V4 compatible

Manuel Franco Sevilla, Frank Golf, Guido Magazzù, Tom Danielson, Adam Dishaw, Jack Bradmiller-Feld
UC Santa Barbara
# Table of Contents

**Front panel**  
2

**General**  
3
- Firmware version  
3
- VME access through the board discrete “emergency” logic  
3
- Jumpers and test points  
4

**Device 1: DCFEB JTAG**  
5
- Example: Read DCFEB UserCode  
5

**Device 2: ODMB JTAG**  
6
- Example: Read ODMB UserCode  
6

**Device 3: ODMB/DCFEB control**  
7
- Bit specification DCFEB pulses command “W 3200”  
8
- Information accessible via command “R 3YZC”  
8

**Device 4: Configuration registers**  
10
- Delay diagrams  
10

**Device 5: Test FIFOs**  
11

**Device 6: BPI Interface (PROM)**  
12

**Device 7: ODMB monitoring**  
13
- Translation into temperatures, current, and voltages  
13

**Device 8: Low voltage monitoring**  
14

**Device 9: System tests**  
15

**Firmware block diagram**  
16

**ODMB headers/trailers**  
17
Front panel

**Push buttons**
- **HRST**: Reloads firmware in PROM onto FPGA
- **SRST**: Resets registers/FIFOs in FW. **LEDs 1-12** blink at different speeds for ~3s
- **PB1**: Sends L1A and L1A_MATCH to all DCFEBs. Turns on LED 12

**LEDs set in firmware**
- 1: 4 Hz signal from clock for data → DDU
- 3: 2 Hz signal from clock for data → PC
- 5: 1 Hz signal from internal ODMB clock
- 7: Data taking: ON normal, OFF pedestal
- 9: Triggers: ON external, OFF internal
- 11: Data: ON real, OFF simulated
- 2: Bit 0 of L1A_COUNTER
- 4: Bit 1 of L1A_COUNTER
- 6: Bit 2 of L1A_COUNTER
- 8: Bit 3 of L1A_COUNTER
- 10: Bit 4 of L1A_COUNTER
- 12: Briefly ON when a VME command is received. Also ON when PB1 is pressed

**LEDs set in hardware**
- **DDU**: Signal Detected on DDU RX
- **PC**: Signal Detected on PC RX
- **ETD**: DTACK enable for discrete logic (active low)
- **EJD**: JTAG enable for discrete logic (active low)
- **DON**: DONE signal from FPGA. ON when programmed
- **INIT**: INIT_B signal from FPGA (active low)
- **LOCK**: QPLL is locked
- **ERR**: Error with QPLL
- **Bottom 12**: Voltage monitoring
General

Firmware version

For a given firmware tag VXY-ZK:

❖ Usercode is XYZKdbdb
❖ Firmware version read via “R 4200” is XYZK

VME access through the board discrete “emergency” logic

The FPGA may be accessed via JTAG through the discrete logic as follows

❖ The VME address is 0xFFFC
❖ The bit 0 of the data sent is TMS
❖ The bit 1 of the data sent is TDI

For example, to read the Usercode, starting from JTAG idle (five TMS = 1 & one TMS = 0), the commands are:

W FFFC 1 To Select-DR-Scan
W FFFC 1 To Select-IR-Scan
W FFFC 0 To Capture-IR
W FFFC 0 To Shift-IR
W FFFC 0 Shifting IR (Read UserCode IR = 3C8)
W FFFC 0 Shifting IR
W FFFC 0 Shifting IR
W FFFC 2 Shifting IR
W FFFC 0 Shifting IR
W FFFC 2 Shifting IR
W FFFC 2 Shifting IR
W FFFC 2 Shifting IR
W FFFC 2 Shifting IR
W FFFC 3 Shifting IR and to Exitl-IR
W FFFC 1 To Update-IR
W FFFC 0 To Run_Test/Idle
W FFFC 1 To Select-DR-Scan
W FFFC 0 To Capture-DR
W FFFC 0 Shifting DR
R FFFC 0 Shifting DR (Read bit 0 of UserCode)

Since the Usercode register is 32 bits, the last two commands should be repeated 31 more times.
Jumpers and test points

Place the **jumpers** marked in **red** in the diagram (master mode). The signals sent to the **test points** marked are:

<table>
<thead>
<tr>
<th>TP13</th>
<th>RAW_LCT(1)</th>
<th>TP14</th>
<th>L1A_MATCH(1)</th>
<th>TP31</th>
<th>Defined by TP_SEL</th>
<th>TP32</th>
</tr>
</thead>
<tbody>
<tr>
<td>TP15</td>
<td>RAW_LCT(2)</td>
<td>TP16</td>
<td>L1A_MATCH(2)</td>
<td>TP33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP17</td>
<td>RAW_LCT(3)</td>
<td>TP18</td>
<td>L1A_MATCH(3)</td>
<td>TP35</td>
<td>Defined by TP_SEL</td>
<td>TP36</td>
</tr>
<tr>
<td>TP19</td>
<td>RAW_LCT(4)</td>
<td>TP20</td>
<td>L1A_MATCH(4)</td>
<td>TP37</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP21</td>
<td>RAW_LCT(5)</td>
<td>TP22</td>
<td>L1A_MATCH(5)</td>
<td>TP39</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP23</td>
<td>RAW_LCT(6)</td>
<td>TP24</td>
<td>L1A_MATCH(6)</td>
<td>TP41</td>
<td>Defined by TP_SEL</td>
<td>TP42</td>
</tr>
<tr>
<td>TP25</td>
<td>RAW_LCT(7)</td>
<td>TP26</td>
<td>L1A_MATCH(7)</td>
<td>TP43</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TP27</td>
<td>L1A</td>
<td>TP28</td>
<td>DDU_DATA_VALID</td>
<td>TP45</td>
<td>Defined by TP_SEL</td>
<td>TP46</td>
</tr>
<tr>
<td>TP29</td>
<td>OTMBDAV</td>
<td>TP30</td>
<td>ALCTDAV</td>
<td>TP47</td>
<td>DCFEB_TDI</td>
<td>TP48</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>TP49</td>
<td>DCFEB_TMS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2.5V</td>
<td></td>
</tr>
</tbody>
</table>

**Add ST15 for slave mode**

Set M2 to P2V5 for slave mode

Firmware tag: 3.12
Device 1: DCFEB JTAG

“Y” refers to the number of bits to be shifted

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W 1Y00</td>
<td>Shift Data; no TMS header; no TMS tailer</td>
</tr>
<tr>
<td>W 1Y04</td>
<td>Shift Data with TMS header only</td>
</tr>
<tr>
<td>W 1Y08</td>
<td>Shift Data with TMS tailer only</td>
</tr>
<tr>
<td>W 1Y0C</td>
<td>Shift Data with TMS header &amp; TMS tailer</td>
</tr>
<tr>
<td>R 1014</td>
<td>Read TDO register</td>
</tr>
<tr>
<td>W 1018</td>
<td>Resets JTAG protocol to IDLE state (data sent with this command is disregarded)</td>
</tr>
<tr>
<td>W 1Y1C</td>
<td>Shift Instruction register</td>
</tr>
<tr>
<td>W 1020</td>
<td>Select DCFEB, one bit per DCFEB</td>
</tr>
<tr>
<td>R 1024</td>
<td>Read which DCFEB is selected</td>
</tr>
</tbody>
</table>

Example: Read DCFEB UserCode

DCFEB registers are set and read via JTAG. The following procedure reads the 32-bit USERID of DCFEB 3:

W 1020 4 Select DCFEB 3 (one bit per DCFEB)
W 191c 3C8 Set instruction register to 3C8 (read UserCode)
W 1F04 0 Shift 16 lower bits
R 1014 0 Read last 16 shifted bits (DBDB)
W 1F08 0 Shift 16 upper bits
R 1014 0 Read last 16 shifted bits (XY2K)

Firmware tag: 3.12
Device 2: ODMB JTAG

“Y” refers to the number of bits to be shifted

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W 2Y00</td>
<td>Shift Data; no TMS header; no TMS tailer</td>
</tr>
<tr>
<td>W 2Y04</td>
<td>Shift Data with TMS header only</td>
</tr>
<tr>
<td>W 2Y08</td>
<td>Shift Data with TMS tailer only</td>
</tr>
<tr>
<td>W 2Y0C</td>
<td>Shift Data with TMS header &amp; TMS tailer</td>
</tr>
<tr>
<td>R 2014</td>
<td>Read TDO register</td>
</tr>
<tr>
<td>W 2018</td>
<td>Resets JTAG protocol to IDLE state (data sent with this command is disregarded)</td>
</tr>
<tr>
<td>W 2Y1C</td>
<td>Shift Instruction register</td>
</tr>
<tr>
<td>W 2020</td>
<td>Change polarity of V6_JTAG_SEL</td>
</tr>
</tbody>
</table>

Example: Read ODMB UserCode

Read FPGA UserCode:

W 291C 3C8  Set instruction register to 3C8 (read UserCode)
W 2F04 0  Shift 16 lower bits
R 2014 0  Read last 16 shifted bits (DBDB)
W 2F08 0  Shift 16 upper bits
R 2014 0  Read last 16 shifted bits (XYZK)
## Device 3: ODMB/DCFEB control

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W/R 3000</td>
<td>0 → nominal mode, 1 → calibration mode (ODMB generates L1A with every pulse)</td>
</tr>
<tr>
<td>W 3004</td>
<td>ODMB soft reset</td>
</tr>
<tr>
<td>W 3008</td>
<td>ODMB optical reset</td>
</tr>
<tr>
<td>W 3010</td>
<td>Reprograms all DCFEBs</td>
</tr>
<tr>
<td>W 3014</td>
<td>L1A reset and DCFEB RESYNC</td>
</tr>
<tr>
<td>W/R 3020</td>
<td>TP_SEL register (selects which signals are sent to TP31, TP35, TP41, TP45)</td>
</tr>
<tr>
<td>W/R 3024</td>
<td>Number of words in DCFEB packet before autokill. 1024 by default and after each reset.</td>
</tr>
<tr>
<td>W/R 3100</td>
<td>LOOPBACK: 0 → no loopback, 1 or 2 → internal loopback</td>
</tr>
<tr>
<td>R 3110</td>
<td>DIFFCTRL (TX voltage swing): 0 → minimum ~100 mV, F → maximum ~1100mV</td>
</tr>
<tr>
<td>R 3120</td>
<td>Read DONE bits from DCFEBs (7 bits)</td>
</tr>
<tr>
<td>R 3124</td>
<td>Read if QPLL is locked</td>
</tr>
<tr>
<td>W 3200</td>
<td>Sends pulses to DCFEBs (see below)</td>
</tr>
<tr>
<td>W/R 3300</td>
<td>Data multiplexer: 0 → real data, 1 → dummy data</td>
</tr>
<tr>
<td>W/R 3304</td>
<td>Trigger multiplexer: 0 → external triggers, 1 → internal triggers</td>
</tr>
<tr>
<td>W/R 3308</td>
<td>LVMB multiplexer: 0 → real LVMB, 1 → dummy LVMB</td>
</tr>
<tr>
<td>W/R 3400</td>
<td>0 → normal, 1 → pedestal (L1A_MATCHes sent to DCFEBs for each L1A).</td>
</tr>
<tr>
<td>W/R 3404</td>
<td>0 → normal, 1 → OTMB data requested for each L1A (requires special OTMB FW)</td>
</tr>
<tr>
<td>W/R 3408</td>
<td>Bit 0 → kills L1A. Bits 1-7 → kills L1A_MATCHes</td>
</tr>
<tr>
<td>W/R 340C</td>
<td>MASK_PLS: 0 → normal, 1 → no EXTPLS/INJPLS (for non-pulsed pedestals from CCB)</td>
</tr>
<tr>
<td>R 3YZC</td>
<td>Read ODMB_DATA corresponding to selection YZ (see below)</td>
</tr>
</tbody>
</table>
Bit specification DCFEB pulses command “W 3200”

- DCFEB_PULSE[0] - Sends INJPLS signal to all DCFEBs.
- DCFEB_PULSE[1] - Sends EXTPLS signal to all DCFEBs.
- DCFEB_PULSE[2] - Sends test L1A and L1A_MATCH to non-killed DCFEBs.
- DCFEB_PULSE[3] - Sends LCT request to OTMB.
- DCFEB_PULSE[4] - Sends external trigger request to OTMB.
- DCFEB_PULSE[5] - Sends BC0 to all DCFEBs.

Information accessible via command “R 3YZC”

Trigger and packet counters

- YZ = 3F: Least significant 16 bits of L1A_COUNTER (also MSP in 33AC and LSP in 33BC)
- YZ = 5F: Least significant 16 bits of L1A_COUNTER (only reset by hard resets, no RESYNCS)
- YZ = 71-77: Number of LCTs for given DCFEB
- YZ = 78: Number of OTMBDAVs (available OTMB packets)
- YZ = 79: Number of ALCTDAVs (available ALCT packets)
- YZ = 21-29: Number of L1A_MATCHes for given DCFEB, OTMB, ALCT
- YZ = 41-49: Number of packets received for given DCFEB, TMB, or ALCT
- YZ = 4A: Number of packets sent to the DDU
- YZ = 4B: Number of packets sent to the PC
- YZ = 51-59: Number of packets shipped to DDU and PC for given DCFEB, TMB, or ALCT
- YZ = 61-67: Number of data packets received with good CRC for given DCFEB

Timing

- YZ = 31-37: Gap (in number of bunch crossings) between the last LCT and L1A for given DCFEB
- YZ = 38: Gap (in number of bunch crossings) between the last L1A and OTMBDAV
- YZ = 39: Gap (in number of bunch crossings) between the last L1A and ALCTDAV

Monitoring of QPLL, RX, TX

- YZ = 4F: Read number of times the QPLL lock has been lost
- YZ = A1-A7: Number of bad CRCs for given DCFEB
- YZ = B1-B7: Number of times there are fiber errors for given DCFEB in last 63 cc (includes errors on IDLE)
- YZ = A8: Times the PLL for the DDU TX lost its lock
- YZ = A9: Times the DDU RX has an error
- YZ = AA: Number of bit errors in the DDU RX
- YZ = AB: Times the PC RX has an error
- YZ = AC: Number of bit errors in the PC RX
- YZ = B8: DCFEBs auto-killed due to fiber errors or too-long packets (last 7 bits)
- YZ = B9: DCFEBs auto-killed due to fiber errors only (last 7 bits)
Production tests

- YZ = 5A: Read last CCB_CMD[5:0] + EVTRST + BXRST strobed
- YZ = 5B: Read last CCB_DATA[7:0] strobed
- YZ = 5C: Read toggled CCB_CAL[2:0] + CCB_BX0 + CCB_BXRST + CCB_L1ARST + CCB_L1A + CCB_CLKEN + CCB_EVTRST + CCB_CMD_STROBE + CCB_DATA_STROBE
- YZ = 5D: Read toggled CCB_RSV signals
Device 4: Configuration registers

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W/R 4000</td>
<td>LCT_L1A_DLY[5:0] → Set to LCT/L1A gap - 100</td>
</tr>
<tr>
<td>W/R 4004</td>
<td>OTMB_DLY[5:0] → Set to L1A/OTMBDAV gap read with “R 338C”</td>
</tr>
<tr>
<td>W/R 4008</td>
<td>CABLE_DLY[0:0] → Delays sending L1A[_MATCH], RESYNC, BCO by 25 ns</td>
</tr>
<tr>
<td>W/R 400C</td>
<td>ALCT_DLY[5:0] → Set to L1A/ALCTDAV gap read with “R 339C”</td>
</tr>
<tr>
<td>W/R 4010</td>
<td>INJ_DLY[4:0] - Delay: 12.5*INJ_DLY [ns]</td>
</tr>
<tr>
<td>W/R 4014</td>
<td>EXT_DLY[4:0] - Delay: 12.5*EXT_DLY [ns]</td>
</tr>
<tr>
<td>W/R 4018</td>
<td>CALLCT_DLY[3:0] - Delay: 25*CALLCT_DLY [ns]</td>
</tr>
<tr>
<td>W/R 401C</td>
<td>KILL[9:1] (ALCT + TMB + 7 DCFEBs)</td>
</tr>
<tr>
<td>W/R 4020</td>
<td>CRATEID[6:0]</td>
</tr>
<tr>
<td>W/R 4028</td>
<td>Number of words generated by dummy DCFEBs, OTMB, and ALCT</td>
</tr>
<tr>
<td>R 4100</td>
<td>Read ODMB unique ID (if not set request UCSB to write it)</td>
</tr>
<tr>
<td>R 4200</td>
<td>Read firmware version</td>
</tr>
<tr>
<td>R 4300</td>
<td>Read firmware build</td>
</tr>
<tr>
<td>R 4400</td>
<td>Read month/day firmware was synthesized</td>
</tr>
<tr>
<td>R 4500</td>
<td>Read year firmware was synthesized</td>
</tr>
</tbody>
</table>

Delay diagrams

1. **LCT_L1A_DLY**, **OTMB_DLY**, and **ALCT_DLY** match preLCT, OTMBDAV, and ALCTDAV to L1A, respectively

2. **EXT_DLY**/**INJ_DLY** set the distance between the CCB signals and the pulses. **CALLCT_DLY** sets the distance between the pulses and the L1A/L1A_MATCHes

Firmware tag: 3.12
Device 5: Test FIFOs

Z refers to FIFO:  1 → PC TX,  2 → PC RX,  3 → DDU TX,  4 → DDU RX,  5 → OTMB,  6 → ALCT

### Instruction

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R 5000</td>
<td>Read one word of selected DCFEB FIFO</td>
</tr>
<tr>
<td>R 500C</td>
<td>Read numbers of words stored in selected DCFEB FIFO</td>
</tr>
<tr>
<td>W/R 5010</td>
<td>Select DCFEB FIFO</td>
</tr>
<tr>
<td>W 5020</td>
<td>Reset DCFEB FIFOs (7 bits, one per FIFO, which are auto-reset)</td>
</tr>
<tr>
<td>R 5Z00</td>
<td>Read one word of FIFO</td>
</tr>
<tr>
<td>R 5Z0C</td>
<td>Read numbers of words stored in FIFO</td>
</tr>
<tr>
<td>W 5Z20</td>
<td>Reset FIFO</td>
</tr>
</tbody>
</table>

### Notes

1. All these FIFOs except PC/DDU TX can hold a maximum of 2,000 18-bit words (36 kb).
   1. PC and DDU TX are 4 times larger.
2. The **OTMB, ALCT, and 7 DCFEB FIFOs** store the data as it arrives in parallel to the standard data path
   - They can hold a maximum of 3 OTMB, 4 ALCT, and 2 DCFEB data packets
3. The **DDU TX FIFO** stores DDU packets just before being transmitted
   - They include the DDU header (4 words starting with 9, 4 starting with A), ALCT data, TMB data, DCFEB data, and trailer (4 words starting with F, 4 starting with E)
4. The **PC TX FIFO** stores DDU packets wrapped in ethernet frames just before being transmitted
   - They include the ethernet header (4 words) and trailer (4 words) and fillers.
   - They need to be at least 32 words long
5. The **DDU and PC RX FIFOs** can be used for loopback tests
Device 6: BPI Interface (PROM)

Important: Instruction 6000 takes ~1 second, during which Device 4 and 6 write commands are ignored

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W 6000</td>
<td>Write configuration registers to PROM</td>
</tr>
<tr>
<td>W 6004</td>
<td>Set configuration registers to retrieved values from PROM</td>
</tr>
<tr>
<td>W 6020</td>
<td>Reset BPI interface state machines</td>
</tr>
<tr>
<td>W 6024</td>
<td>Disable parsing commands in command FIFO while filling FIFO with commands (no data)</td>
</tr>
<tr>
<td>W 6028</td>
<td>Enable parsing commands in the command FIFO (no data)</td>
</tr>
<tr>
<td>W 602C</td>
<td>Write one word to command FIFO</td>
</tr>
<tr>
<td>R 6030</td>
<td>Read one word from read-back FIFO</td>
</tr>
<tr>
<td>R 6034</td>
<td>Read number of words in read-back FIFO</td>
</tr>
<tr>
<td>R 6038</td>
<td>Read BPI Interface Status Register</td>
</tr>
<tr>
<td>R 603C</td>
<td>Read Timer (16 LSBs)</td>
</tr>
<tr>
<td>R 6040</td>
<td>Read Timer (16 MSBs)</td>
</tr>
</tbody>
</table>

\[
T_{\text{FPGA}} = \frac{R_{600} \times 503.975}{4096} - 273.15 \ [^\circ \text{C}]
\]

\[
I_{\text{PPIB}} = \frac{R_{12} \times 5000}{4096} - 10 \ [\text{mA}]
\]

\[
V_{YZ} = \frac{R_{YZ}}{2048} \times V_{YZ,\text{Nom}} \ [\text{V}]
\]
Device 7: ODMB monitoring

Reads output of the ADC inside the FPGA

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R 7000</td>
<td>FPGA temperature</td>
</tr>
<tr>
<td>R 7100</td>
<td>LV_P3V3: input to FPGA regulators</td>
</tr>
<tr>
<td>R 7110</td>
<td>P5V: input to PPIB regulator and level for 5V chips</td>
</tr>
<tr>
<td>R 7120</td>
<td>IPPIB: current going to PPIB (on V2s and V3s, board temperature THERM2)</td>
</tr>
<tr>
<td>R 7130</td>
<td>P3V6_PP: voltage level for PPIB</td>
</tr>
<tr>
<td>R 7140</td>
<td>P2V5: voltage level for FPGA and 2.5V chips</td>
</tr>
<tr>
<td>R 7150</td>
<td>THERM1: board temperature close to the regulators</td>
</tr>
<tr>
<td>R 7160</td>
<td>P1V0: voltage level for FPGA</td>
</tr>
<tr>
<td>R 7170</td>
<td>P5V_LVMB: voltage level for LVMB</td>
</tr>
</tbody>
</table>

Translation into temperatures, current, and voltages

The output of the 7YZ0 commands is a 12-bit number that we call Ryz. The measurement is:

- The FPGA temperature is

- The PPIB current is

- The temperature of the thermistors THERM1, THERM2 is given by

<table>
<thead>
<tr>
<th>Rxy</th>
<th>377</th>
<th>455</th>
<th>55A</th>
<th>687</th>
<th>7DD</th>
<th>959</th>
<th>AF8</th>
<th>CB5</th>
<th>E87</th>
<th>FFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>T[^°C]</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>30</td>
<td>35</td>
<td>40</td>
<td>45</td>
<td>50</td>
<td>55</td>
<td>60</td>
</tr>
</tbody>
</table>

- The voltage levels are \( V_{\text{yz, nom}} \) for that register. That is, \( V_{10, \text{nom}} = 3.3V, V_{13, \text{nom}} = 3.6V, V_{11, \text{nom}} = V_{17, \text{nom}} = 5V, V_{14, \text{nom}} = 2.5V, \) and \( V_{16, \text{nom}} = 1V. \)
Device 8: Low voltage monitoring

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>W 8000</strong></td>
<td>Send control byte to ADC</td>
</tr>
<tr>
<td><strong>R 8004</strong></td>
<td>Read ADC</td>
</tr>
<tr>
<td><strong>W 8010</strong></td>
<td>Select DCFEBs/ALCT to be powered on (8 bits, ALCT + 7 DCFEBs)</td>
</tr>
<tr>
<td><strong>R 8014</strong></td>
<td>Read selected DCFEBs/ALCT to be powered on (see notes)</td>
</tr>
<tr>
<td><strong>R 8018</strong></td>
<td>Read which DCFEBs/ALCT are actually powered on</td>
</tr>
<tr>
<td><strong>W 8020</strong></td>
<td>Select ADC to be read, 0 to 6</td>
</tr>
<tr>
<td><strong>R 8024</strong></td>
<td>Read which ADC is to be read</td>
</tr>
</tbody>
</table>

**Notes**

The ODMB has an internal 8-bit register that selects with DCFEBs/ALCT to turn on when a LOAD signal is issued. Command **W 8010 XX** both changes the register to **XX** and issues the LOAD signal. **R 8014** reads the internal register, while **R 8018** reads the actual state of the boards on the crate.

The mapping of the 8 bits to DCFEBs/ALCT is non-trivial, and different for forward and backward chambers.

### Table 1. Control-Byte Format

<table>
<thead>
<tr>
<th>BIT 7 (MSB)</th>
<th>BIT 6</th>
<th>BIT 5</th>
<th>BIT 4</th>
<th>BIT 3</th>
<th>BIT 2</th>
<th>BIT 1</th>
<th>BIT 0 (LSB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>SEL2</td>
<td>SEL1</td>
<td>SEL0</td>
<td>RNG</td>
<td>BIP</td>
<td>PD1</td>
<td>PD0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PD1</th>
<th>PD0</th>
<th>MODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Normal operation (always on), internal clock mode.</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Normal operation (always on), external clock mode.</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Standby power-down mode (STBYPD), clock mode unaffected.</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Full power-down mode (FULLPD), clock mode unaffected.</td>
</tr>
</tbody>
</table>

### Table 4. Power-Down and Clock Selection

<table>
<thead>
<tr>
<th>INPUT RANGE</th>
<th>RNG</th>
<th>BIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 to +5V</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0 to +10V</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>±5V</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>±10V</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
### Device 9: System tests

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>W 9000</strong></td>
<td>Test the DDU TX/RX with a given number of PRBS $2^7-1$ sequences</td>
</tr>
<tr>
<td><strong>R 900C</strong></td>
<td>Read number of errors during last DDU PRBS test</td>
</tr>
<tr>
<td><strong>W 9100</strong></td>
<td>Test the PC TX/RX with a given number of PRBS $2^7-1$ sequences</td>
</tr>
<tr>
<td><strong>R 910C</strong></td>
<td>Read number of errors during last PC PRBS test</td>
</tr>
<tr>
<td><strong>W 9200</strong></td>
<td>Check $N*10000$ bits from the PRBS pattern sent by the DCFEB</td>
</tr>
<tr>
<td><strong>W/R 9204</strong></td>
<td>Select DCFEB fiber to perform PRBS test</td>
</tr>
<tr>
<td><strong>R 9208</strong></td>
<td>Read number of error edges during last DCFEB PRBS test</td>
</tr>
<tr>
<td><strong>R 920C</strong></td>
<td>Read number of bit errors during last DCFEB PRBS test</td>
</tr>
<tr>
<td><strong>W/R 9300</strong></td>
<td>Set PRBS type for DCFEB: 1 → PRBS-7, 2 → PRBS-15, 3 → PRBS-23, 4 → PRBS-31</td>
</tr>
<tr>
<td><strong>W 9400</strong></td>
<td>Check $N*10000$ bits from the PRBS pattern sent by the OTMB</td>
</tr>
<tr>
<td><strong>R 9404</strong></td>
<td>Read number of enables sent by the OTMB</td>
</tr>
<tr>
<td><strong>R 9408</strong></td>
<td>Read number of good 10000 bits sent by the OTMB</td>
</tr>
<tr>
<td><strong>R 940C</strong></td>
<td>Read number of bit errors during last OTMB PRBS test</td>
</tr>
<tr>
<td><strong>W 9410</strong></td>
<td>Reset number of errors in OTMB counter</td>
</tr>
</tbody>
</table>
The firmware can be downloaded from [http://github.com/odmb/odmb_ucsb_v2](http://github.com/odmb/odmb_ucsb_v2)
**ODMB headers/trailers**

**Structure of ODMB header**

Four **0x9000** words and four **0xA000** words

<table>
<thead>
<tr>
<th>Header Word</th>
<th>Highest 4 bits</th>
<th>DDU Code</th>
<th>Lowest 12 bits [11:0]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a</td>
<td>1001</td>
<td>9</td>
<td>DMB_L1A[11:0]</td>
</tr>
<tr>
<td>1b</td>
<td>1001</td>
<td>9</td>
<td>DMB_L1A[23:12]</td>
</tr>
<tr>
<td>1c</td>
<td>1001</td>
<td>9</td>
<td>ALC T_DAV(1) + TMB_DAV(1) + Fmt_Vers(1:0) + CLCT_DAV-Mismatch(1) + CFEB_CLCT_SENT(7:1)</td>
</tr>
<tr>
<td>1d</td>
<td>1001</td>
<td>9</td>
<td>DMB_BXN[1:0]</td>
</tr>
<tr>
<td>2a</td>
<td>1010</td>
<td>A</td>
<td>ALC T_DAV(1) + TMB_DAV(1) + Fmt_Vers(1:0) + CLCT_DAV-Mismatch(1) + CFEB_DAV(7:1)</td>
</tr>
<tr>
<td>2b</td>
<td>1010</td>
<td>A</td>
<td>DMB_CRATE(8) + DMB_ID(4)</td>
</tr>
<tr>
<td>2c</td>
<td>1010</td>
<td>A</td>
<td>ALC T_DAV(1) + TMB_DAV(1) + CFEB_MOVLP(5:1) + DMB_BXN[4:0]</td>
</tr>
<tr>
<td>2d</td>
<td>1010</td>
<td>A</td>
<td>DMB-CFEB-Sync[3:0] + Fmt_Vers(1:0) + CLCT_DAV-Mismatch(1) + DMB_L1A[4:0]</td>
</tr>
</tbody>
</table>

**Structure of ODMB trailer**

Four **0xF000** words and four **0xE000** words

<table>
<thead>
<tr>
<th>Trailer Word</th>
<th>Highest 4 bits</th>
<th>DDU Code</th>
<th>Lowest 12 bits [11:0]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1a</td>
<td>1111</td>
<td>F</td>
<td>ALC T_End_Timcout(1) + DMB_BXN[4:0] + DMB_L1A[5:0]</td>
</tr>
<tr>
<td>1b</td>
<td>1111</td>
<td>F</td>
<td>CFEB_MOVLP(5:1) + CFEB_End_Timcout(7:1)</td>
</tr>
<tr>
<td>1c</td>
<td>1111</td>
<td>F</td>
<td>CFEB_FULL(3:1) + TMB_Start_Timcout(1) + DMB_L1PIPE(8)</td>
</tr>
<tr>
<td>1d</td>
<td>1111</td>
<td>F</td>
<td>ALC T_Start_Timcout(1) + CFEB_Start_Timcout(7:1) + CFEB_FULL(7:4)</td>
</tr>
<tr>
<td>2a</td>
<td>1110</td>
<td>E</td>
<td>ALC T_FULL(1) + TMB_FULL(1) + ALC T_HALF(1) + TMB_HALF(7:1)</td>
</tr>
<tr>
<td>2b</td>
<td>1110</td>
<td>E</td>
<td>Duplicate Header 2b (DMB Crate &amp; ID)</td>
</tr>
<tr>
<td>2c</td>
<td>1110</td>
<td>E</td>
<td>DMB_CRC_LowParity(1) + DMB_CRC[10:0]</td>
</tr>
<tr>
<td>2d</td>
<td>1110</td>
<td>E</td>
<td>DMB_CRC_HighParity(1) + DMB_CRC[21:11]</td>
</tr>
</tbody>
</table>

Firmware tag: 3.12