DCM / VME / ONCS Interface

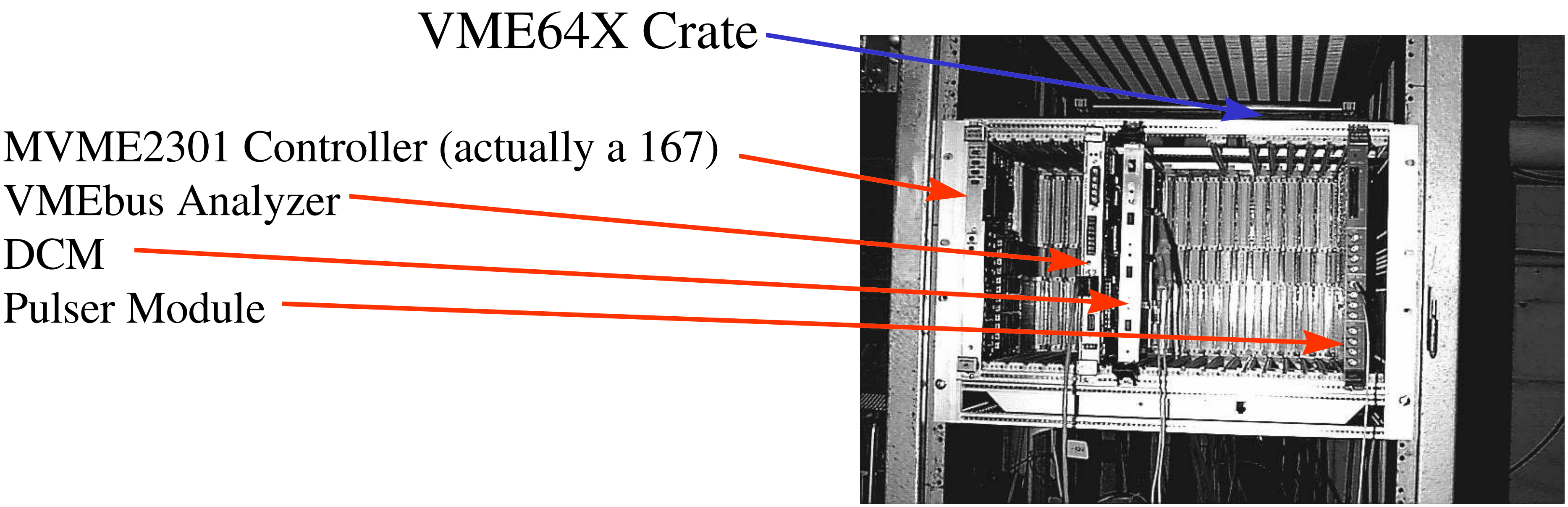
Overview Issues:

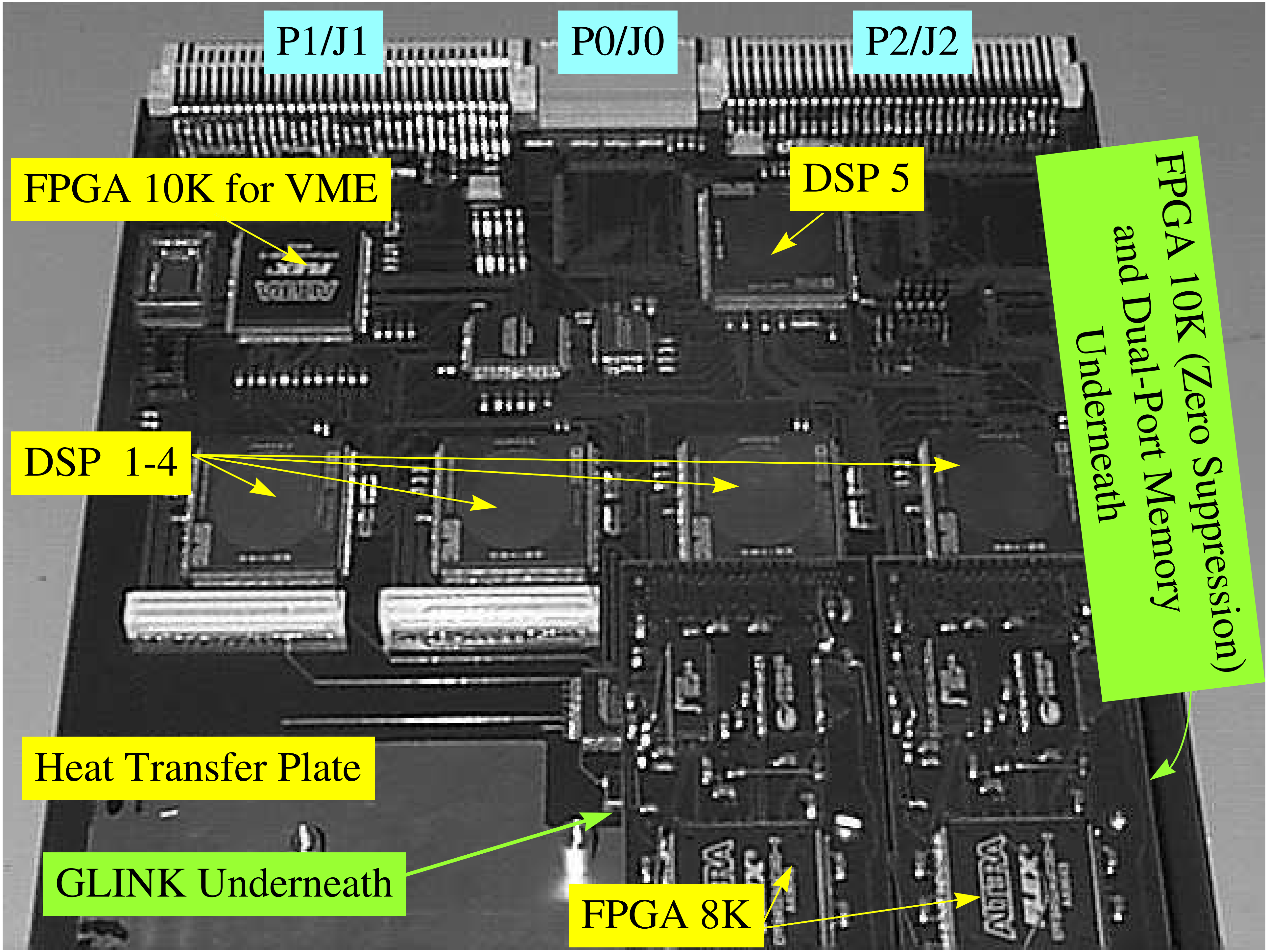
- Mechanical Considerations
- ONCS / DCM Interface
- Communications Protocol to DCM
- DSP Software
- Tests and Progress Report

Mechanical Considerations

The DCM's reside in a 21 slot, 280 mm deep VME64X crate (which includes P0, P1, and P2 connectors). The crate controller (MVME2301) uses a 120 mm extender card for connection.

The large insertion force due to three connectors with 415 pins is dealt with using Rittal injector/ejector (IEEE-1102). Note that the Motorola PowerPC controller comes with these injectors.

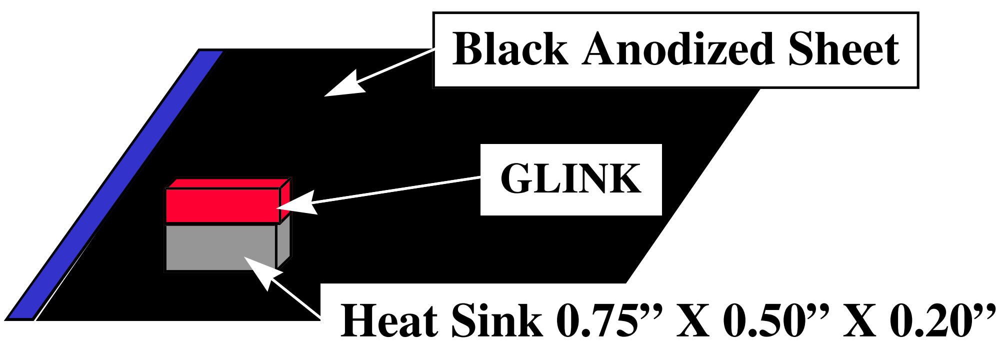


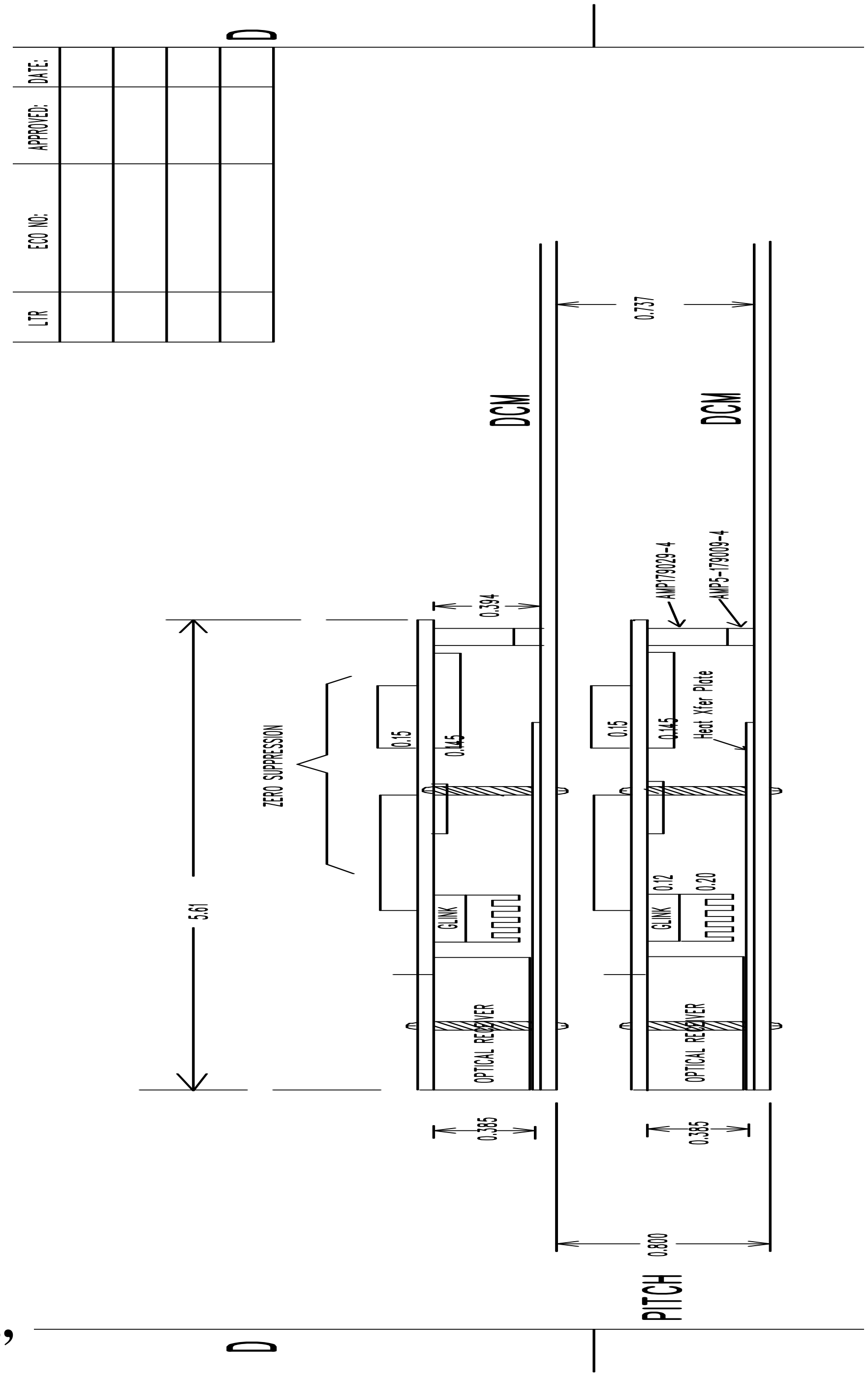

P1/J1
P0/J0
P2/J2
FPGA 10K for VME
DSP 5
FPGA 10K (Zero Suppression) and Dual-Port Memory Underneath
DSP 1-4
Heat Transfer Plate
GLINK Underneath
FPGA 8K

Cooling in the lab is achieved with six rack-mounted fans for the crate.

The most significant on-board cooling issue is with the GLINK chip (~ 3 Watts) located on the underside of the daughter card.

We have attached using a heat conducting pad a black anodized aluminum block which then connects to a black anodized sheet on the The sheet connects to the front panel.



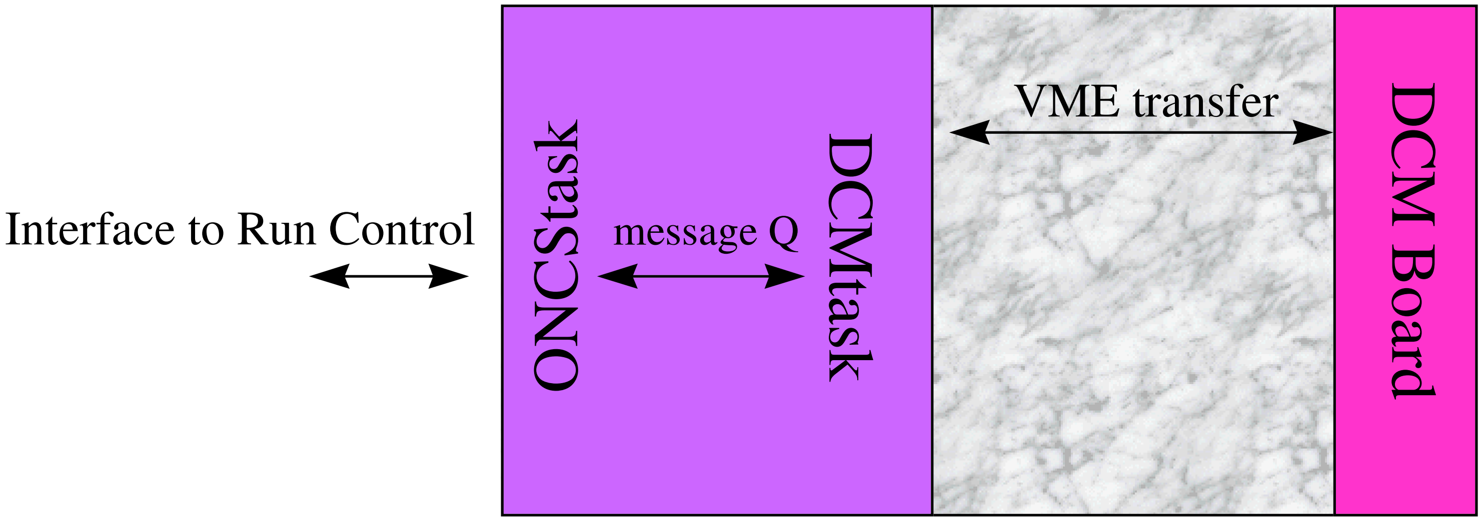


DCM to ONCS Interface

Not really a topic for this review… However...

Two tasks will run simultaneously in the crate controller (DCMtask and ONCStask). The DCMtask will have exclusive rights to VME transfers (and the crate controller is the only VME master in the crate).

The two tasks communicate using the VxWorks message queue passing specified information for various downloading and uploading procedures.



DCM OPERATION CODES (OpCode)

A24/D32

24 bit address				**32 bit data field**	
23	22-18	17-14	13-0	31-16	15-0
Broadcast	DCM address	Operation Code (OP)	Don’t Care	L1 table / test data	VME-DCM command / status

Broadcast for write operations

Geographic Addressing (5 bits)

Defined Operations Codes (4 bits)

Data Transferred

Overview of Operation Codes

Opcode (1): Allows for clear and reset of on-board devices (for example DSP, link port, level-1 port).

Opcode (2) and (3): Specifies which DSP link port is addressed in following VME write or read operations.

Opcode (4) and (9): Write and read operations to DSP link port. Each operations is a 16-bit data transfer, thus for 32-bit words two transfers are necessary and three transfers for 48-bit words.

Opcode (8): Checks the status of the five DSP link ports. ACK and LCK bits specify whether there is room to write to the link port buffer and data to read form the buffer, respectively.

Operation Code Definitions

Opcode	Definition	* note
1	Write DCM Configuration Word	Define DCM state
2	Define LINK Input DSP #, Word size	Define which DSP VME talk to and word
3	Define LINK out DSP #, Word size	Define which DSP VME take data from ar
4	Write data to DCM Link Port	Send data/program word to DSP
5	Write to L1 Primitive Look-up table	VME D 16-31 as table address
6	Write test data to L1 input FIFO	VME D 16-31 as test data input
7	Interrupt DSP	Signal DSP
8	Read DCM-VME link port status	
9	Read data from DSP link port	receive data / program word from DSP
10	Read DCM status	
11	Read Compressor port status	

DCM to VME Communication

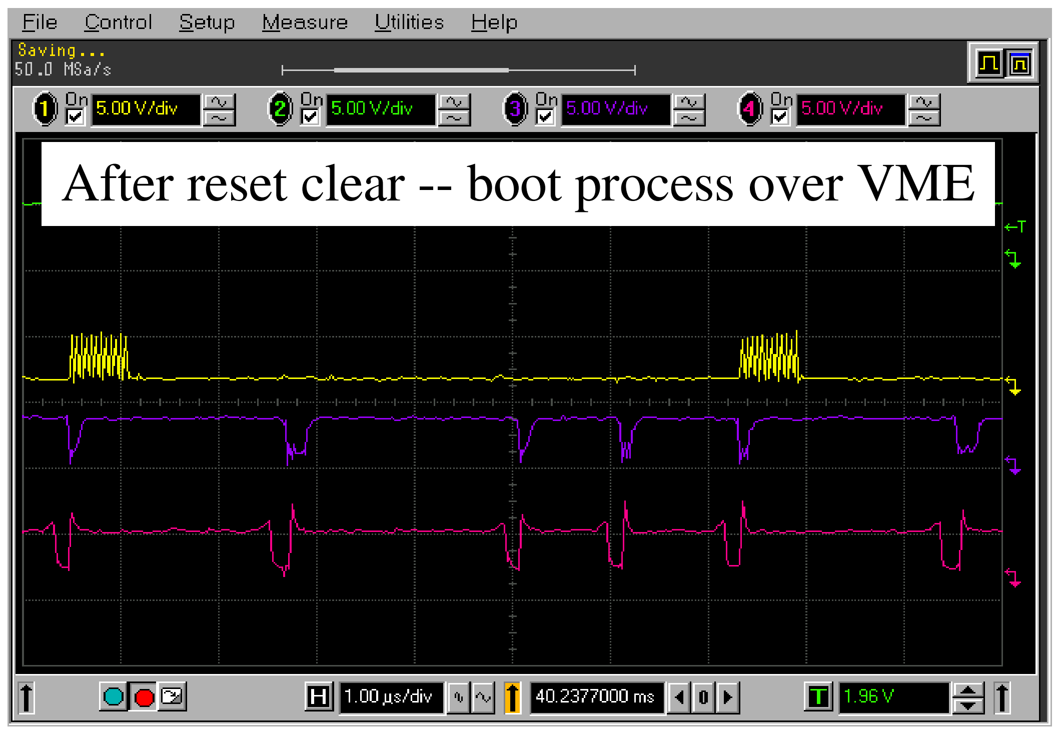
Document (DCM to VME Interface) defines a set of operation codes for communication. The transfers are non-standard A24/D32 using a user-defined address modifier (AM=0x18). We have successfully tested the memory map for MV162, MV167 and MVME2301. Geographic addressing of DCM has been tested.

Note: VME write 10^6 Hz
VME read 4 x 10^5 Hz

DATA CLK to DSP Link Port

VME DTACK*

VME DS0*



DSP Software Issues

We are using Analog Devices software…

asm21k (Analog Devices assembler)
g21k (GNU CPP version 3.2 < 2.3.3 snapshot)
ldr21k (Analog Devices 21000 Loader 3.22)
--we modified boot kernel for VME transfer booting

Known Limitations…

Windows feature tested only with 3.1
Beta version for Windows 95 and NT available in March
Known bug in binary format output (not observed)…

Investigated 3rd Party Software Support -- does not look promising

Extensive tests using assembler and C code, but without optimization features.

DSP Memory Architecture

Two banks each having one Megabit SRAM (**32k** 32-bit words) for the SHARC 21062

1) Program Memory
2) Data Memory

In this example:

heap == 16k
data == 4k

In the 5th DSP can the memory contain an entire un-zero suppressed event….

TEC = 20 words x 64 channels x 4 FEM/DCM fiber (Day-1) x 4 DCM/DCBoard = **20k**

```
.SYSTEM       YER_BASIC_062_C_ACH;

.PROCESSOR = ADSP21062;

!bank 0...
!4k rth
.SEGMENT/RAM/BEGIN = 0x00020000 /END = 0x00020fff /PM seg_rth;

!4k initialization
.SEGMENT/RAM/BEGIN = 0x00021000 /END = 0x00021fff /PM
seg_init;

!12k code
.SEGMENT/RAM/BEGIN = 0x00022000 /END = 0x00024fff /PM
        seg_pmco;

!bank 1...
!4k pm data
.SEGMENT/RAM/BEGIN = 0x00028000 /END = 0x00028fff /PM
/WIDTH=32 seg_pmda;

!4k dm data
.SEGMENT/RAM/BEGIN = 0x00029000 /END = 0x00029fff /DM
/WIDTH=32 seg_dmda;

!16k heap
.SEGMENT/RAM/BEGIN = 0x0002a000 /END = 0x0002dfff /DM
/WIDTH=32 seg_heap;

!8k stack
.SEGMENT/RAM/BEGIN = 0x0002e000 /END = 0x0002ffff /DM
/WIDTH=32 seg_stak;

! external area for external port fifo transfers
.SEGMENT/RAM/BEGIN = 0x00410000 /END = 0x0041FFFF /DM/uninit
SEG_DUM;

.ENDSYS;
```

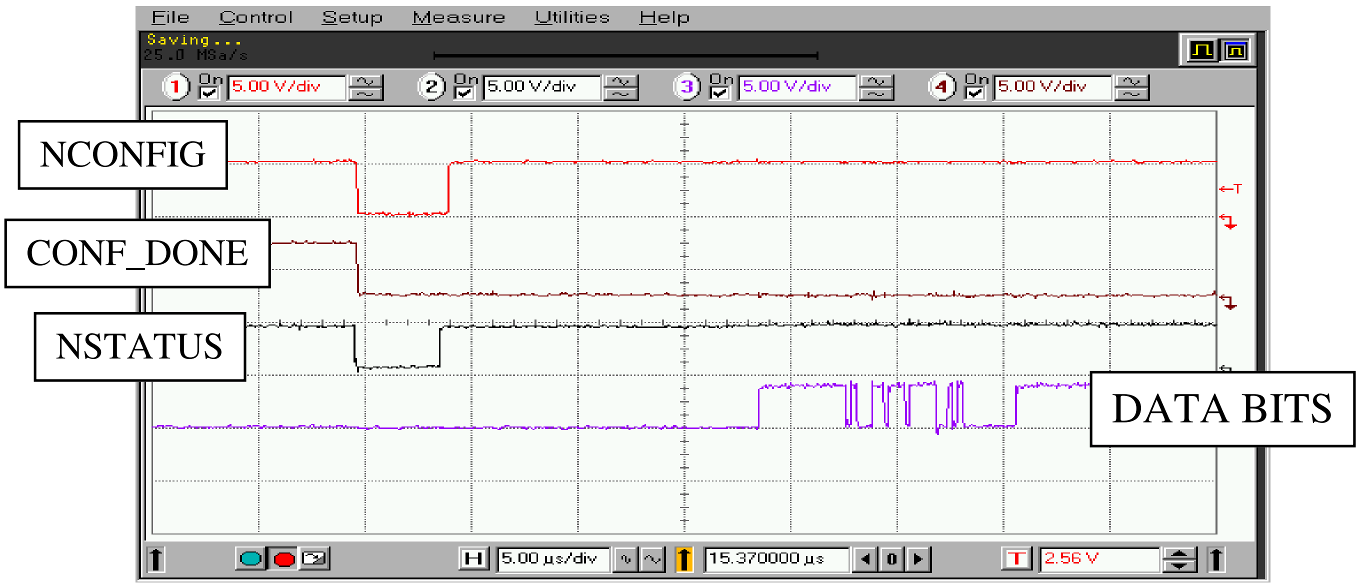
Boot Procedure

• At power up the FPGA 10K (handling VME transfers) automatically boots from on-board EPROM.
• VME Operation Code (1) clears the DSP's and associated link ports and upon reset forces the DSP's to boot from on-board EPROM. This initial boot program simply gives the appropriate hand-shaking and then resets the DSP's to wait for their program. This booting is mediated by the FPGA 10K into the DSP link ports.
• The main program is transferred to the DSP's via Operation Code (4) transfers. The DSP's send back a control status word to verify successful booting.
• Booting DSP's over VME allows easy changing of programs and individual programs for each DSP if desired.
• The boot time is approximately 12 milliseconds.

Compressor Port Boot

•The compressor port (FE2_DCM) FPGA 8K boots itself on power up from an on-board EPROM.

•The FPGA 10K (used for reading out dual-port memory and handling zero suppression) is booting through the serial port of the associated DSP, mediated by the FPGA 8K.



•The external list memory, external pedestal memory and threshold values (stored in the FPGA 10K memory) are loaded and read back via the serial port of the associated DSP.

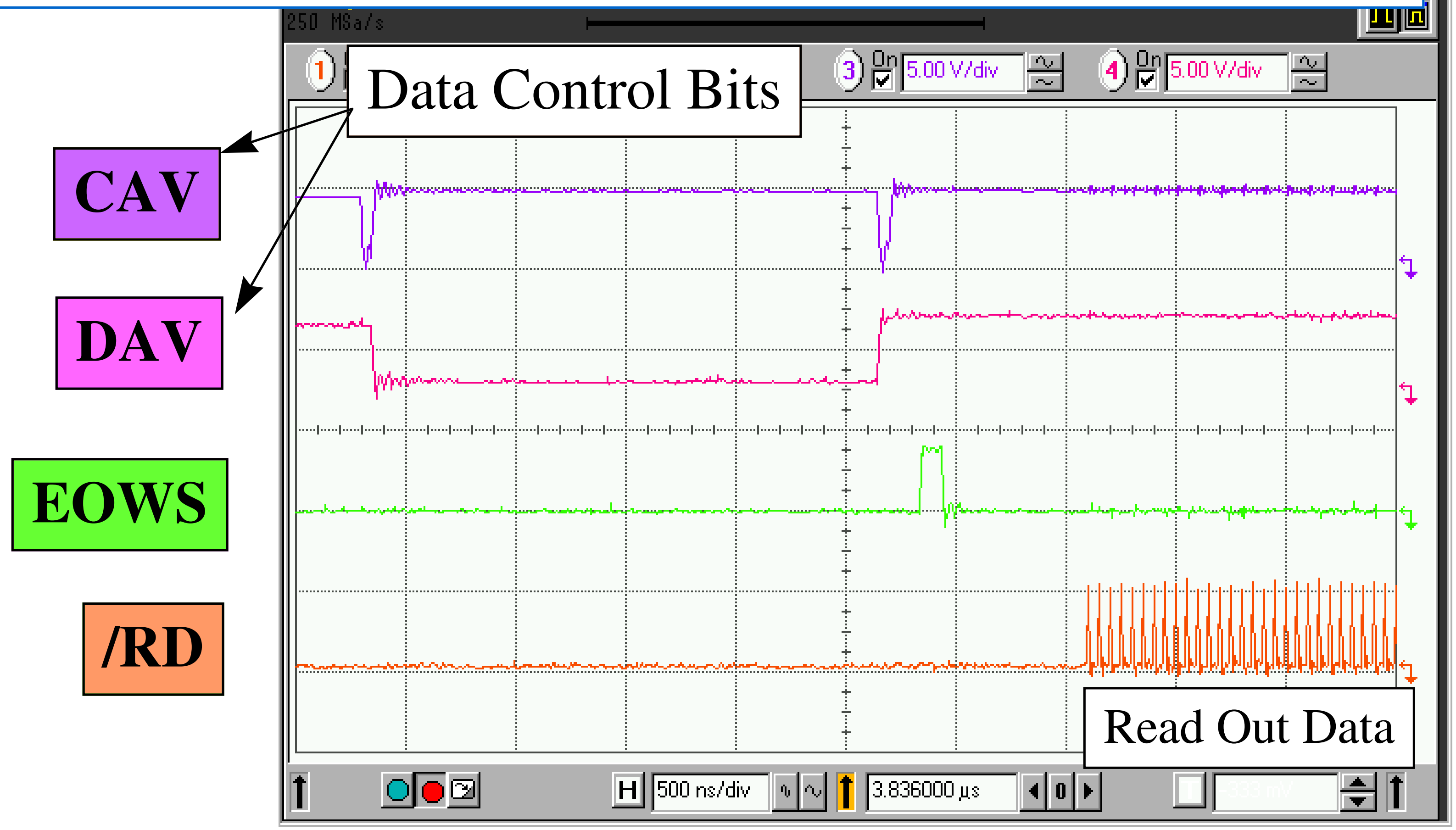
Loading Serial Data to Compressor Port

We have successfully loaded serial data to list memory, pedestal memory, threshold memory and fake dual-port data. We have also read back and verified all of the above memory values.


```
switch(cmdcode) {

case RESET:
    reset(module,&status);
break;

case LOAD_DSP_PROGRAM:
    loadDspProgram(module,unit,filename,&status);
break;

case LOAD_FPGA:
        loadFPGA(module,unit,filename,&status);
break;

case GET_LIST_MEMORY:
        getListMemory(module,unit,start_address,number_reads,filename,&status);
break;

case LOAD_LIST_MEMORY:
        loadListMemory(module,unit,start_address,number_writes,filename,&status);
break;

case GET_PED_MEMORY:
        getPedMemory(module,unit,start_address,number_reads,filename,&status);
break;

case LOAD_PED_MEMORY:
        loadPedMemory(module,unit,start_address,number_writes,filename,&status);
break;

case GET_THRESHOLD_MEMORY:
        getThresholdMemory(module,unit,start_address,number_reads,filename,&status);
break;

case LOAD_THRESHOLD_MEMORY:
        loadThresholdMemory(module,unit,start_address,number_writes,filename,&status);
break;

case LOAD_FAKE_DATA:
        loadFakeData(module,unit,start_address,number_writes,filename,&status);
break;

case GLINK_ONLINE:
        GLINK_online(module,unit,&status);
break;

case GET_DATA_EVENT:
        getDataEvent(module,unit,start_address,number_reads,filename,&status);
break;
};
```

Data Transfer from Fiber Optic

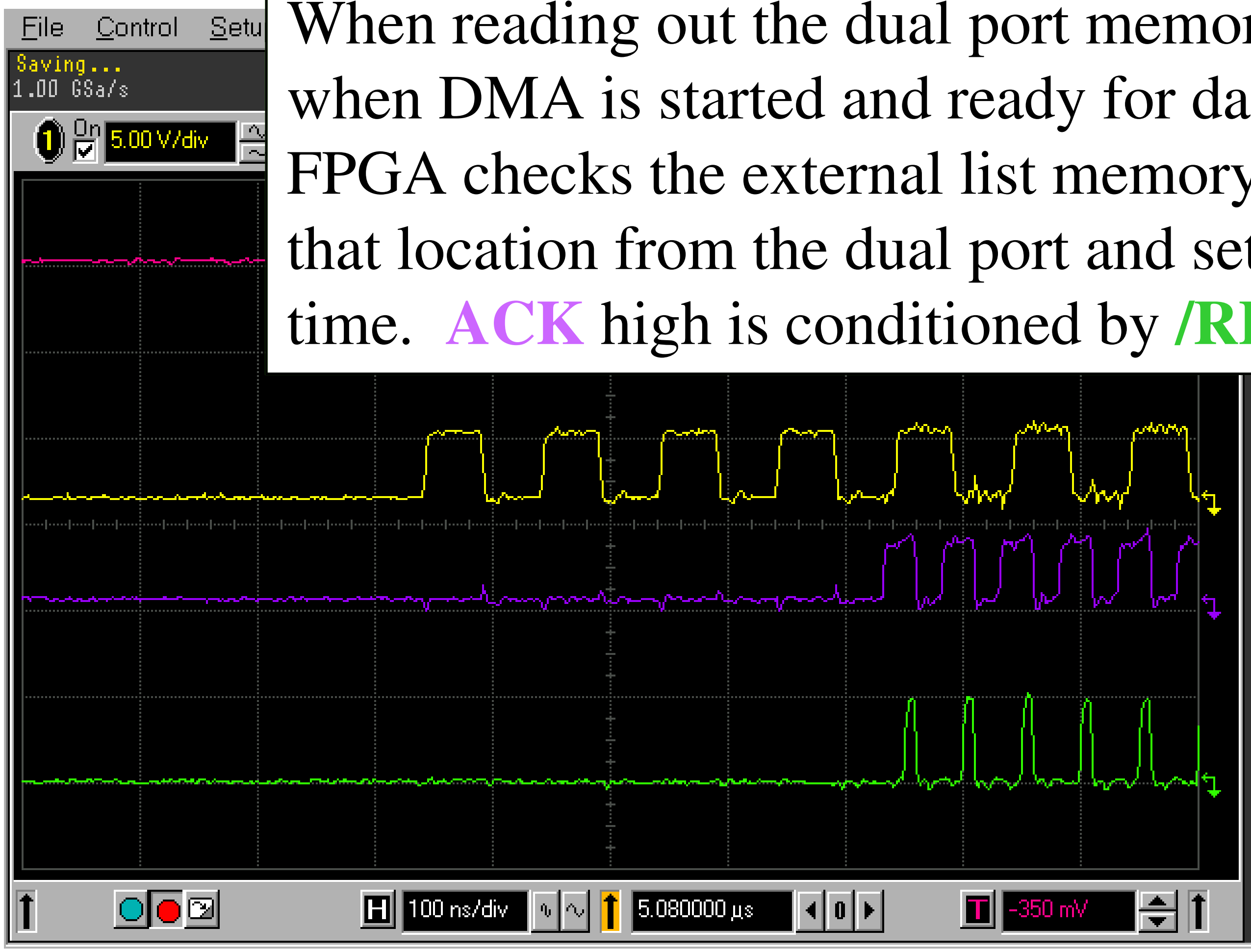
We use an HP82000 connected to a test board with a GLINK and transceiver to send data over fiber into the DCM.   We have checked heck on control bits (CAV and DAV) and the timing relative to end of write (short format) signal and /RD.



Compressor Port: Reading Data Out of Dual Port Memory

We have tested sending data **over fiber** into the compressor port, through the GLINK, into dual-port memory, through the zero suppression FPGA, into the DSP, and out through the DSP link port over VME.

When reading out the dual port memory, the DSP indicates when DMA is started and ready for data (**/RD**). The 10K FPGA checks the external list memory and then reads out that location from the dual port and sets **ACK** at transfer time. **ACK** high is conditioned by **/RD** low.

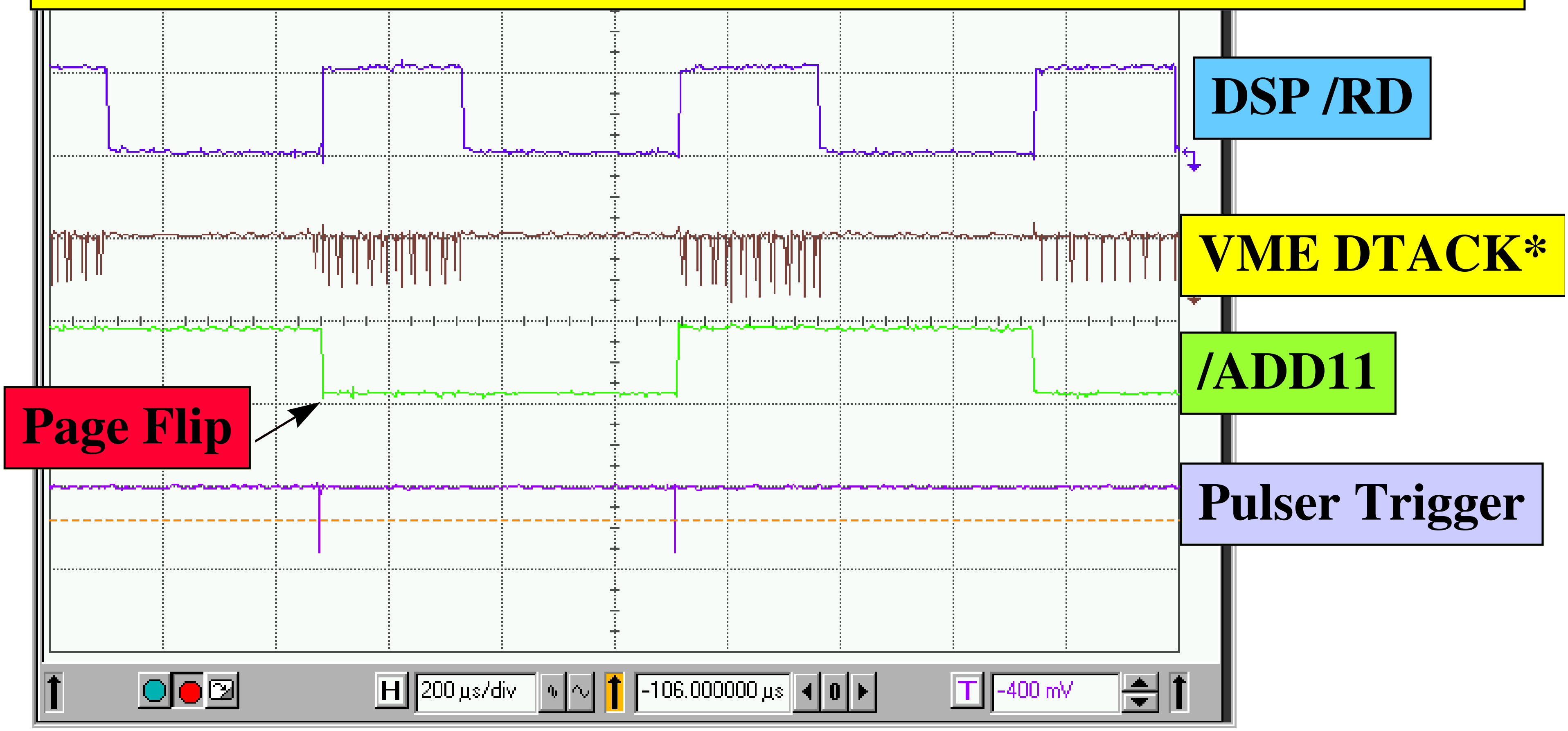


List Memory D0

10K FPGA ACK

DSP /RD

Trigger sent to HP programmable unit which then sends 50 data words over fiber optic. DSP **/RD** is low when ready for data and goes high after full DMA transfer. Then data sent out of DSP link port over VME (**DTACK**). Transfer of 50 words takes approximately 250 microseconds. Each repeated write and read to dual port memory flips the page as seen by FPGA 8K **/ADD11** line.



Speed Limits

•DSP Clock is 40 MHz. Link ports can write or read 4 bits per clock cycle, thus the 32-bit word transfer rate between DSP's is **5 MHz**.

•VME write operations (Opcode 4) run at **1 MHz** for a 16-bit word. VME read operations (Opcode 9) run at **400 kHz** for a 16-bit word. These are non-block transfers.

•DSP serial port access to memory on the compressor port (eg. Pedestal, Threshold, List memory) runs at **4 MHz** per bit.

•For example, since pedestal values are 8-bits data, loading of pedestal data occurs at **500 kHz**. There is additional overhead for transferring this data with VME write operations. There are 128k pedestal channels thus requiring 250 milliseconds.