

DFEC2

CFT/CPS Axial Track Finder
Run IIB Upgrade

Design Specification
[DØ Note 4675](#)

Engineering Note 2004-02-18a

Jamieson Olsen
Fermi National Accelerator Laboratory

revised 6 January 2005

1	OVERVIEW	4
1.1	IMPROVEMENTS	4
1.1.1	<i>High Bandwidth Communication</i>	4
1.1.2	<i>Global Timing Distribution</i>	4
1.2	DFEC2 BLOCK DIAGRAM	5
2	BOARD MECHANICAL	5
2.1	BOARD DIMENSIONS	5
2.2	FRONT PANEL	6
2.2.1	<i>Status LEDs</i>	6
2.2.2	<i>Reset Button</i>	6
2.2.3	<i>Parallel Port Connector</i>	6
2.2.4	<i>JTAG Connector</i>	6
2.2.5	<i>LEMO Connectors</i>	6
2.2.6	<i>Ethernet Fiber Connection</i>	6
2.2.7	<i>SCL Connection</i>	6
2.2.8	<i>DIP Switch Settings</i>	7
2.2.9	<i>Handles</i>	7
2.2.10	<i>Shielding</i>	7
3	POWER DISTRIBUTION	7
4	GIGABIT ETHERNET INTERFACE	8
4.1	FIBER OPTIC CONNECTION	8
4.2	DFEC2 COMMANDS	9
4.2.1	<i>A DFEC2 “download” example</i>	9
4.2.2	<i>DFEC2 Control Registers</i>	10
4.3	ETHERNET FRAME FORMAT	11
4.4	DFEC2 FRAME FORMATS	12
4.5	ETHERNET HANDSHAKING	13
4.5.1	<i>DFEC2 Status Word</i>	13
4.6	NETWORK TOPOLOGIES	14
4.7	SOFTWARE	15
4.7.1	<i>Software Interfaces</i>	15
5	DFE BACKPLANE READ/WRITE BUS	16
5.1	ADDRESS SPACE	16
5.1.1	<i>Slot Address</i>	16
5.2	BACKPLANE TIMING	17
5.2.1	<i>Write Cycle</i>	17
5.2.2	<i>Read Cycle</i>	18
5.3	SIGNAL TERMINATIONS	18
6	TIMING AND CLOCKS	19
6.1	CLOCK AND CONTROL BIT DISTRIBUTION	19
6.2	SIMULATED CONTROL BITS	19
6.3	BPCLK ALIGNMENT	19

7 BACKWARDS COMPATIBILITY..... 20

8 REFERENCES..... 20

9 REVISION HISTORY 21

APPENDIX A: LED MAP 22

1 Overview

The original DFE crate controller (DFEC) used the MIL-1553B bus to communicate with the outside world. The purpose of the DFEC was to:

- Manage FPGA configuration files on a CompactFlash card.
- Transfer files over the backplane to FPGAs on the DFE boards.
- Perform read and write operations over the DFE backplane.
- Collect status information from the DFE boards.

The original DFEC [1] was setup as a list processor – a user would fill its command buffer and tell it to go. The DFEC would process the commands, and put the results in a shared memory buffer that the online computers could access after the DFEC was done. However, the relatively low throughput of the 1553 bus, when coupled with the speed of the micro controller and CompactFlash card made file downloads particularly slow.

1.1 Improvements

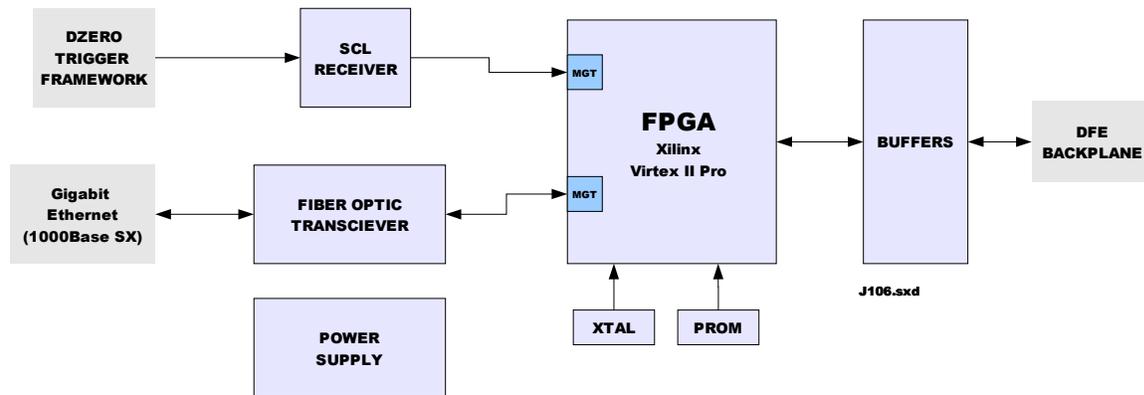
1.1.1 High Bandwidth Communication

Gigabit Ethernet (GbE) will be used instead of the 1553 bus, a 1000X speed improvement. With a faster connection there is no need to store and manage files locally on a flash card – the files can be kept on an online computer and downloaded in a minute or so. This reduces the DFEC2 complexity significantly and makes file management much easier.

1.1.2 Global Timing Distribution

The original DFEC did not provide a master clock or control signals to the DFE boards in the crate. Thus the DFE boards relied on upstream hardware for clocks and control bits. The DFEC2 will get a master 53MHz clock and control bits from the Serial Command Link (SCL) and distribute this information over the backplane to all DFE boards. This scheme is more robust as it synchronizes all DFE boards to a common clock, and it's not dependant on upstream hardware. This last point is important because the upstream hardware (AFE boards) will likely be out of operation while the new DFE hardware is being commissioned.

1.2 DFEC2 Block Diagram



Both the Gigabit Ethernet and the Serial Command Link interfaces will use the Virtex II Pro's Multi-Gigabit Transceivers (MGTs) to handle the high speed serializing/deserializing, framing and encoding/decoding.

2 Board Mechanical

2.1 Board Dimensions

The DFEC2 board dimensions are 6U (233.5mm) x 320mm deep. While the backplane connectors are custom, the rest of the board specifications (spacing, max. component heights, etc.) should follow VME mechanical guidelines [2]. Refer to the DFE Backplane Specification [3] for the pinout of the backplane connectors. Also included on that website are mechanical drawings of the DFE board showing connector locations.

2.2 Front Panel

2.2.1 Status LEDs

LED indicators will be provided for the following:

- All board voltages (+1.5V, +2.5V, and +3.3V)
- FPGA Configured (FPGA DONE=1)
- A bank of six general purpose LEDs which are controlled by the firmware.

The bank of general purpose LEDs can display up a “page” of six status bits at a time. There are six “pages” selected by quickly tapping the reset button. After tapping the reset button the current page number will be displayed as a six-bit one hot value. Then the LEDs will turn off and the current status page will fade in.

All status bits are stretched with a monostable: off means logic zero, on means logic 1, blinking means the bit is toggling. Status page 0 is the power on default. See Appendix A for the LED map.

2.2.2 Reset Button

Holding down the reset button for at least 2 seconds will generate a hard reset. Quickly tap the reset button to cycle through the LED status pages.

2.2.3 Parallel Port Connector

This connector was intended for use as diagnostic test port. Not used for production firmware.

2.2.4 JTAG Connector

This 20-pin 2mm connector matches the Xilinx Parallel Cable IV [5].

2.2.5 LEMO Connectors

These connectors can be wired as buffered inputs or outputs. Currently they’re used as diagnostic outputs, i.e. for running “probes” into the FPGA.

2.2.6 Ethernet Fiber Connection

The DFEC2 board uses an LC style fiber optic connector. An industry-standard small form factor (SFF) transceiver is used for this connection. Multi-mode (850nm) fiber with 50 or 62.5µm core diameter should be used with this board.

2.2.7 SCL Connection

The SCL connects to the DFEC2 with a coaxial copper line. The DFEC2 board uses a right angle SMB type male connector here.

2.2.8 DIP Switch Settings

```

+-----+
| 1 2 3 4 5 6 7 8 | UP   position
| 0                 |
| N (front view)   | DOWN position
+-----+

```

SW1: UP = GBE internal loopback enabled; DOWN = normal operation
 SW2: UP = enable DTACK checking; DOWN = ignore DTACK
 SW3: UP = delay reply transmission; DOWN = send reply immediately
 SW4: reserved
 SW5: mac bit 3 (UP=1, DOWN=0)
 SW6: mac bit 2
 SW7: mac bit 1
 SW8: mac bit 0

SW5	SW6	SW7	SW8	DFEC2 MAC ADDRESS
↓	↓	↓	↑	00-80-55-D0-10-01
↓	↓	↑	↓	00-80-55-D0-10-02
↓	↓	↑	↑	00-80-55-D0-10-03
↓	↑	↓	↓	00-80-55-D0-10-04
↓	↑	↓	↑	00-80-55-D0-10-05
↓	↑	↑	↓	00-80-55-D0-10-06
↓	↑	↑	↑	00-80-55-D0-10-07
↑	↓	↓	↓	00-80-55-D0-10-08
↑	↓	↓	↓	00-80-55-D0-10-09
↑	↓	↑	↓	00-80-55-D0-10-0A

2.2.9 Handles

To make insertion and removal easier, type HL (high leverage, IEEE 1101.10 or equivalent) handles should be used. Refer to Schroff documentation [4] for a drawing of the handles and front panel dimensions.

2.2.10 Shielding

RF shield gaskets should be used on the side edges of the front panel. Refer to Schroff documentation [4] for part numbers and drawings.

3 Power Distribution

The DFEC2 will be powered by a bulk 48VDC supply. An isolated DC-DC converter will generate +3.3V from the +48V mains; the FPGA core voltage (+1.5V) will be generated from the +3.3V using a small non-isolated DC-DC converter. Additional supply voltages (+2.5V and +1.8V) will be generated from the +3.3V using low noise linear LDO regulators.

If the new DFEC2 is plugged into an old DFE backplane it will use the +3.3V supply instead of the isolated +3.3V DC-DC converter; all other board voltages are derived from the +3.3V like normal. The DFEC2 will be fused and protected against over-voltage according to the Fermilab/D0 Safety Guidelines [8].

4 Gigabit Ethernet Interface

Gigabit Ethernet is used to provide a high speed link between an online PC and the DFEC2. Simple commands are wrapped in an Ethernet frame and sent to the DFEC2 for processing. The DFEC2 will absorb the incoming frame, check it for errors, and then process the requested operation and reply back to the host computer. The reply frame will include a status word followed by the requested data, if applicable.

The DFEC2 will send and receive raw GbE frames over a fiber optic connection (1000Base SX). GbE has been selected because of the availability of “off the shelf” hardware such as network interface cards and switches. Higher level protocols such as TCP/IP will not be used. Raw Ethernet frames are simple to implement and they pass through (L2) switches without any problems. It’s important to note that the DFEC2s will always be located on a closed, private network – that is, they will either be connected directly to a PC, or they will go through a single switch or hub and then arrive at a PC. There will be no other network traffic on these Ethernet connections.

4.1 Fiber Optic Connection

The DFEC2’s optical transceiver will connect directly to a Xilinx Virtex II Pro FPGA, which features several MGT’s which handle the SERDES, 8B/10B encoder/decoder, and framing operations. Very simple MAC functionality will be implemented in the FPGA logic, and this design is based loosely on a VME crate controller board being developed at Ohio State University [6].

4.2 DFEC2 Commands

The original DFEC was fairly complex due to the fact that it needed to handle all of the file management on the CompactFlash card. As a result it needed to support some high level commands such as “Initialize device X in slot Y with file number Z” or “Delete file X” and so on.

The DFEC2 will not have a CompactFlash card, so all of the file management functionality will be moved “off board”. This simplifies the DFEC2 tremendously. Now, to the outside world a DFE crate will appear as a simple flat memory space.

All of the old DFEC commands disappear and are replaced with simple memory read and write commands.

Every incoming Ethernet frame will contain one read or write command. Each read/write command can transfer between 1 and 744 16-bit words from/to a single memory address on the DFE backplane.

This document does not cover the definitions of specific backplane addresses, that is addressed in the DFE board documentation. The DFEC2 simply handles read and write requests to backplane addresses; it has no knowledge of what backplane address/register does what.

4.2.1 A DFEC2 “download” example

Since there is no more file management on the new DFEC2 boards, this process is much simpler. Unlike the previous example there is no longer a “download” phase – files are simply transferred from the online system computers directly to registers on the DFE boards as follows:

1. Write and read a few registers on the target DFE board. This will prepare the DFE board for configuration data.
2. Take the disk file and break it up into (1 to 744 word) chunks. Write each chunk of data to backplane address X. Repeat until the file has been transferred.
3. Read a few backplane registers to verify that the target board initialized properly.

4.2.2 DFEC2 Control Registers

Addresses 0x0000 through 0x07FF map to an internal buffer in the DFEC2. Writes to addresses will still go out over the backplane, but the data will also get written into the internal buffer and dtack will be ignored. Likewise, reading from this area will cause a backplane read transaction to occur, but dtack will be ignored. The data values returned to the host will come from the internal buffer in the DFEC2. For example:

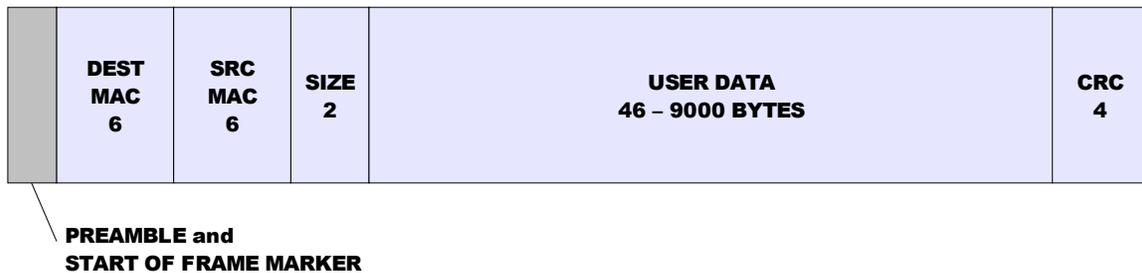
```
write 0x1234, 0x5678, 0xabcd to addr 0x0000
read 2 words from 0x0000: 0x1234, 0x5678
read 1 word from 0x0012: 0x1234

write 0x1111 0x2222 0x3333 0x4444 to addr 0x0002
read 1 word from 0x0000: 0x1111
read 1 word from 0x0001: 0x1111
read 1 word from 0x0002: 0x1111
read 2 words from 0x0000: 0x1111 0x2222
```

NOTE: reading or writing to address 0x0123 will KICK the SCL receiver logic and force it to attempt to relock onto the SCL data stream.

4.3 Ethernet Frame Format

The Ethernet frame is defined by IEEE 802.3 as having a 6 byte (octet) source and destination fields, which are the unique MAC addresses. A size field provides the size of the user data area, which can be between 46 and **1500** bytes. Finally, a 4 byte CRC field is used to verify the integrity of the entire frame. *The DFEC2 currently does not support “jumbo” 9000 byte frames because not all Ethernet hardware supports them.*



All DFEC2 boards will be assigned a unique MAC Address (00-80-55-D0-10-0x) and they will not respond to broadcast messages. The size field will never indicate anything other than the size of the user data area (this field will never be used to indicate special packet types). Likewise, the user data area will be used solely for DFEC2 commands; additional “encapsulation” protocols such as IP or TCP/UDP are not allowed to put anything into this user data area. Furthermore, care must be taken that VLAN routers and switches do not add any new fields to this frame format.

The user data area will contain a single DFEC2 read or write command. No read or write command is allowed to span multiple frames.

4.4 DFEC2 Frame Formats

Write Request Frame:

DFEC2 MAC Address (6 bytes)
Host MAC Address (6 bytes)
User Data size in bytes (2 bytes)
User Data (46 to 1500 bytes) which contains:
 Start Marker (4 bytes: 0x12, 0x34, 0xAB, 0xCD)
 Write Command (2 bytes: 0101)
 DFE Backplane Address (2 bytes)
 Number of words to write (2 bytes)
 Data words to write (1-744 words)
 End Marker (2 bytes: 0xAA, 0x55)
 Padding (if necessary)
Frame CRC (4 bytes)

Read Request Frame:

DFEC2 MAC Address (6 bytes)
Host MAC Address (6 bytes)
User Data size in bytes (2 bytes: 0x00, 0x2E)
User Data (46 bytes) which contains:
 Start Marker (4 bytes: 0x12, 0x34, 0xAB, 0xCD)
 Read Command (2 bytes: 0202)
 DFE Backplane Address (2 bytes)
 Number of words to read (2 bytes)
 End Marker (2 bytes: 0xAA, 0x55)
 Padding (36 bytes, all 0x00)
Frame CRC (4 bytes)

DFEC2 Reply Frame:

Host MAC Address (6 bytes)
DFEC2 MAC Address (6 bytes)
User Data size in bytes (2 bytes)
User Data (46 to 9000 bytes) which contains:
 Start Marker (2 bytes: 0xDF, 0xEC)
 Status Word (2 bytes)
 DFE Backplane Address (2 bytes)
 Number of data words (2 bytes)
 Requested Data (length varies)
 End Marker (2 bytes: 0xAA, 0x55)
 Padding (if necessary)
Frame CRC (4 bytes)

4.5 Ethernet Handshaking

If the DFEC2 is idle it will monitor the input and check all incoming Ethernet frames. If the DFEC2 is busy it will ignore all Ethernet traffic. When an Ethernet frame is absorbed by the DFEC2 it will check the following:

- The frame CRC bits are correct.
- The destination MAC address matches the DFEC2 MAC address.
- The Length of the User Data section is correct.
- The User Data section contains a valid start and end markers.
- The Command Word is valid.
- The DFE Backplane Address is valid.
- The number of data words requested is less than the maximum.

If any of the above checks fail, the DFEC2 will immediately send back a reply frame to the host and return to idle mode. Otherwise, the DFEC2 will begin to process the read or write request.

While the DFEC2 is reading or writing to the DFE Backplane it may encounter a DTACK error, meaning that the DFE board did not respond with a DTACK signal at the appropriate time during the read/write cycle. If a DTACK error occurs, the DFEC2 stops processing the command and sends a reply frame to the host. The bad address is contained in this reply frame. For read requests the DFEC2 will send back the all of the requested data up to the point where the DTACK error occurred. For example, the host requests 20 words from address 0x5001. On the 17th read cycle the DFEC2 detects a DTACK error. The DFEC2's reply frame will contain the first good 16 data words.

4.5.1 DFEC2 Status Word

Each DFEC2 reply frame will include a 16-bit status word, defined as:

- 15. Write command processed OK
- 14. Read command processed OK, requested data is in this frame.
- 13. Last incoming frame had BAD CRC
- 12. Bad User Data length
- 11. User Data area garbled, could not find start/end markers
- 10. Unrecognized command
- 09. Bad DFE backplane address
- 08. Requested too much data (overflow)

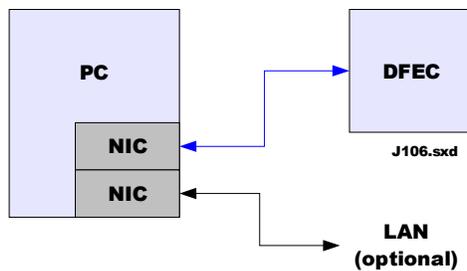
- 07. DTACK timeout accessing the backplane
- 06: SCL receiver is properly locked to the Trigger Framework
- 05: SCL signal detect OK
- 04: SCL MGT/DCM locked OK
- 03: SCL parity error
- 02: SCL framing error
- 01: reserved, zero
- 00: reserved, zero

There are a couple of points worth noting here:

- The PC host software (driver) *must* wait for the reply frame to come back before sending another read/write request frame.
- If the total size of the requested read data exceeds the maximum size of the reply frame, the DFEC2 will not process the command frame and it will reply with the appropriate error bits in the status word.

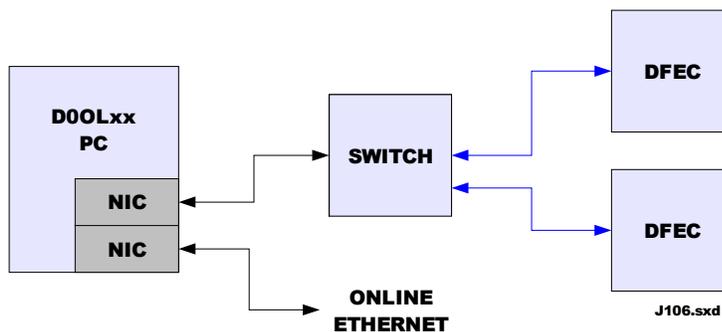
4.6 Network Topologies

For firmware development, the local configuration will be the most straightforward:



In this configuration a separate GbE network interface card (NIC) is installed in a Linux box. This NIC must have optical outputs, or it can use an external commercial CAT5 to fiber converter box.

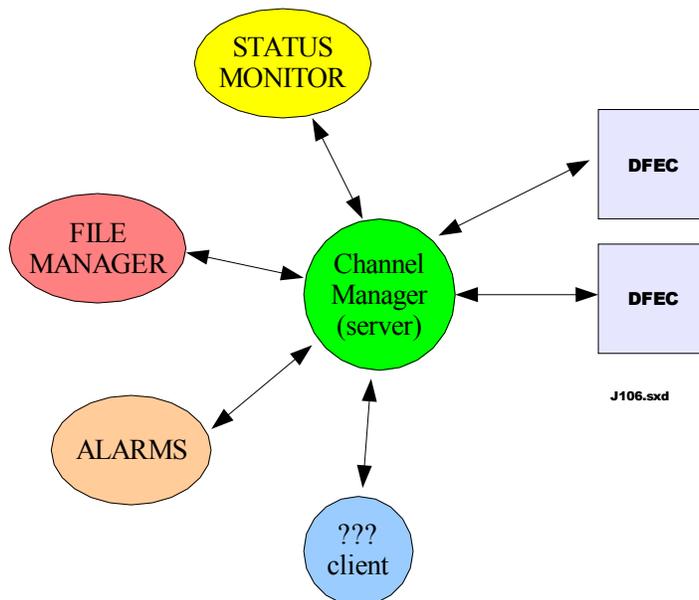
The final configuration will likely use a fiber optic GbE switch to fan out to all DFEC2s:



4.7 Software

Linux C libraries allow a user level process to send and receive raw Ethernet frames over a NIC card. Some example code (untested) has been found [7], and at first glance it doesn't appear to be too difficult to implement. No kernel hacking appears to be necessary.

For much of the initial development period the relationship between the client process and the DFEC2 will one-to-one, and no channel arbitration will be needed. However, as the system design progresses, server software must be developed that will allow multiple clients to talk to multiple DFECs. It is thought that EPICs might be a suitable software package for this application (?).



If a user at a remote location needs to talk to the DFEC2s located at DZERO, it is thought that he or she will have to log into a kerberized online Linux PC and then run their programs on that computer. The DFECs will certainly not be bridged to a network directly accessible from outside the laboratory.

4.7.1 Software Interfaces

The most fundamental primitives will enable a user level program to read and write from the DFE backplane. In C pseudo-code, such function calls might look something like this:

```
int dfe_read(MAC_addr, slot_number, register, number_of_words, data_array[]);
int dfe_write(MAC_addr, slot_number, register, number_of_words, data_array[]);
```

With these two functions, any application can be built up in C/C++ or Python or whatever. For example, the firmware downloader is nothing more than a little program that reads in Xilinx BIN or BIT file from disk and breaks the file up into many write commands and sends them out as Ethernet frames. Downloading an entire crate may be as easy as FTPing the files into a directory, then running a shell script to call the downloader program for each file.

5 DFE Backplane Read/Write Bus

Communication with DFE boards in slots 2 through 21 occurs over the DFE backplane bus. The DFE backplane bus is much simpler than VME or PCI busses. It's a simple, *synchronous* A16 D16 read/write bus with the following LVTTTL signals:

- The clock (SYSCLK, or BPCLK) frequency is $RF/7 = \sim 7.5\text{MHz}$.
- DATA[15..0]
- ADDR[15..0]
- AS active high “address strobe”
- WRITE – active high write enable
- DTACK – active low data acknowledge, open collector.

It was originally thought that the backplane clock frequency could be higher, possibly $RF/4$, but there is concern that the calorimeter may be sensitive to this frequency. $RF/7$ is the beam crossing frequency, and the calorimeter is not sensitive to EMI generated by hardware clocked at this frequency.

5.1 Address Space

The upper six bits of the address specifies the slot number and the lower ten bits specify the register or memory address on the DFE board:



2004-04-30a

Thus each slot in the backplane may have up to 1024 registers, although in reality only a few registers be used. Broadcast writes are supported by setting all of the ADDR[15..10] bits. (DFEA boards should not respond with DTACK when broadcast write operations occur.)

5.1.1 Slot Address

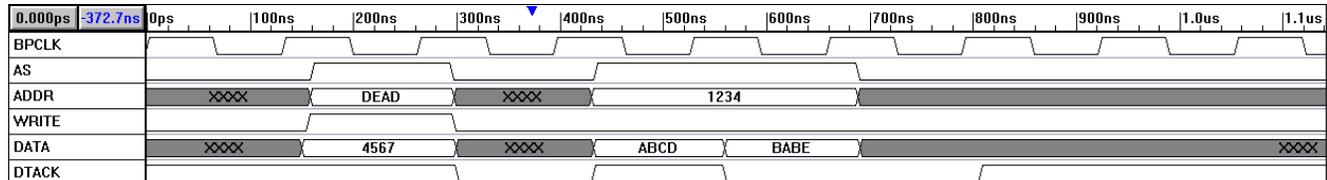
The DFE backplane specification [4] shows six slot address pins named SLOT[5..0]. These pins encode a unique slot address (00010 - 10101) on the lower five bits. The most value of the most significant bit is determined by a jumper on the backplane. On the backplane these pins are either tied to GND or left unconnected, thus **pull-up resistors are required** on the DFEA boards.

5.2 Backplane Timing

The DFE backplane bus allows for reads and writes. DMA type mechanisms are not supported. All backplane signals are synchronous to the rising edge of BPCLK.

5.2.1 Write Cycle

Write cycles are designed for maximum throughput on the backplane bus – this is needed to quickly transfer large configuration files to the FPGAs located on the DFE boards. In the example below three words are written to a DFE board.



Data is written to the DFE boards on the rising edge of BPCLK when AS and WRITE are high. On the next rising edge of BPCLK the DFEC2 will check the DTACK line to make sure the write operation succeeded. If DTACK signal is not low at this time the DFEC2 will stop and set the appropriate error bits. No retry will be attempted.

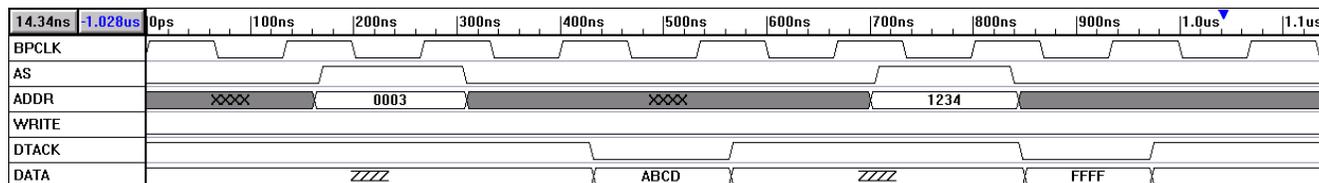
Consecutive write operations to a single address may be packed closely together so that a word is written on each rising edge of BPCLK – thus the data transfer rate will approach 15MB/sec. (Assuming 225MB of configuration data per crate, initialization will take 15 seconds, ignoring the overhead associated with Ethernet, the online computer software, etc.)

For broadcast write cycles, the DFEC2 will not check the status of the DTACK line.

From the DFE board perspective the AS, ADDR, and WRITE signals will have $t_{SU} = \sim 120\text{ns}$ and $t_H = \sim 12\text{ns}$ with respect to BPCLK. The DFEC2 requires a $t_{SU} > \sim 10\text{ns}$ and $t_H > \sim 10\text{ns}$ on the DTACK signal.

5.2.2 Read Cycle

Reading from the DFE boards will take longer than a write cycle, since oftentimes the requested data will not be readily available in the DFE board's backplane interface FPGA. Thus the DFEC2 should allow for an extra clock cycle between the read request and when the DFE board drives the data lines. In the example below two read cycles occur:



In the first cycle the DFEC2 reads 0xABCD from address 0x0003. The DFE board sees AS and ADDR asserted on the rising edge of BPCLK (275ns). On the next rising edge of BPCLK the DFEC2 checks the DTACK line – it is high, so the DFEC2 assumes that the DFE is busy gathering the requested data; the DFEC2 will wait one more cycle and check again. On the next rising edge of BPCLK the DFEC2 checks the DTACK line: it is low, and the DFEC2 samples the DATA bus at this time. Had DTACK been high at this time the DFEC2 would consider this to be a read error, and it stop and set the appropriate error bits.

The next read cycle in the above example shows a faster response – this time the DFE board is ready with the requested data by the next rising edge of BPCLK (925ns).

Read cycles will never be as fast as write cycles. The DFEC2 will insert a wait cycle after each read cycle. From the DFE board perspective the AS, ADDR, and WRITE signals will have $t_{SU} = \sim 120\text{ns}$ and $t_H = \sim 12\text{ns}$ with respect to BPCLK. The DFEC2 requires a $t_{SU} > \sim 10\text{ns}$ and $t_H > \sim 10\text{ns}$ on the DATA bus and DTACK.

5.3 Signal Terminations

The DTACK line is pulled up on the DFEC2 board, so the DFEA boards should only pull this line low with an open collector driver. BPCLK is terminated on the backplane.

6 Timing and Clocks

The original DFE boards received their clocks and control bits directly from the input data stream (*embedded control bits*). Since each DFE board would get multiple input links, typically one link was selected as the master link and that controlled the entire board.

The DFEC2 will receive a master clock and control bits via a Serial Command Link (SCL) receiver daughterboard. These signals will be buffered on the DFEC2 and distributed to each DFE board over the backplane.

6.1 Clock and Control Bit Distribution

A 53MHz master clock is delivered to each DFE board differentially using LVDS. The DFEC2 will control delays so that the master clock skew between any boards in the crate will be less than a couple of nanoseconds. Likewise, the control bits are multiplexed over a single LVDS pair and sent to each board in the crate. The control bits are synchronous to the clock and will have a setup time of approximately $\sim 10\text{ns}$ and a hold time of $\sim 8\text{ns}$. (In the DFE backplane specification [4] the clock is referred to as MCLK_n/MCLK_n* and the master control bits are referred to as NRZ_n/NRZ_n*.)

There are seven RF clock ticks per 132ns crossing. Thus up to seven bits can be packed into a control bit “frame”, shown below:

bit	description
1	start (always set)
2	FX First Crossing
3	SG Sync Gap
4	L1 Level-1 Accept
5	SCL init
6	reserved
7	odd parity

2004-02-11a

These control bit frames are sent continuously with no gaps in between. It is the responsibility of each DFE board to “lock” onto this control bit frame and extract the control bits.

6.2 Simulated Control Bits

If an SCL control link is not present, the DFEC2 will automatically switch to a mode where it will produce a 53MHz clock and generate simulated SG and FX control signals. When in this simulated SCL mode the crate will be free running from the 53.1047MHz oscillator on the DFEC2 board.

6.3 BPCLK alignment

The rising edge of BPCLK will always coincide with the rising edge of the NRZ start bit. It is strongly recommended that BPCLK (not the start bit) is used for synchronizing backplane transfers. Both the NRZ bits and BPCLK are synchronous to the master 53MHz clock.

7 Backwards Compatibility

The DFEC2 will operate if plugged into an old DFE backplane, with the following caveats:

- No 1553 access – status monitoring software must be rewritten to use the Ethernet interface.
- The read/write bus data width is 8-bits, not 16.
- Clocks and control bits will not be distributed to DFE boards.

Plugging an old DFEC into a new backplane will not work, but it should cause no damage to the DFEC.

8 References

1. Original DFE Crate Controller Engineering Notes available at:
<http://d0server1.fnal.gov/users/jamieson/www/notes/index.html>
2. VMEbus Mechanical Specifications (Ch. 7)
Revision C.1 (1985). <http://www.ieee.org>
3. DFE Upgrade Hardware Specifications
<http://www-d0.fnal.gov/~jamieson/run2b>
4. Schroff Front Panel Express Design Guide (includes VME/eurocard specs)
<http://www.schroffus.com>
5. Xilinx JTAG connector (Parallel Cable IV)
<http://www.xilinx.com/bvdocs/publications/ds097.pdf>
6. Ohio State University GbE controller overview
<http://www.physics.ohio-state.edu/~bylsma/talks/>
7. Code examples: raw Ethernet frames on Linux
http://www.landshut.org/bnla01/members/Faustus/fh/linux/udp_vs_raw/index.html
8. Fermilab/DZERO Electrical Safety Guidelines
<http://www-d0.fnal.gov/~hance/tools.htm>

9 Revision History

18 Feb 2004: Original Draft.

24 Feb 2004: Removed L3 sender functionality, added backplane timing and control bits.

04 May 2004: update block diagram, limit one read/write command per Ethernet frame, add handshaking and status word bit definitions. Added DIP Switch.

10 Sep 2004: update front panel features, add status word bits 7..0, add internal buffer description, add kick_scl information, update DIP switch definition, add Appendix A (LEDs). Remove references to jumbo frames. Add section 6.3 BPCLK alignment. Change to "DFEC2" everywhere.

06 Jan 2005: make it a d0 note, check to make sure names are consistant (DFEA2, DFEB2, DFEC2, etc.)

Appendix A: LED map

The bold status bits are most important, the others are useful for firmware development or debugging.

Page 0 (power on default)

- 0.5 **GBE write packet received**
- 0.4 **GBE read packet received**
- 0.3 **GBE dropped packet**
- 0.2 **GBE bad packet (trapped)**
- 0.1 **GBE busy sending reply packet**
- 0.0 **GBE transmission error**

Page 1

- 1.5 GBE rx_crc_err, incoming packet failed CRC check
- 1.4 GBE rx_buff_err, buffer over/underflow condition
- 1.3 GBE rx_char_err, incoming byte was not properly 8B/10B encoded
- 1.2 GBE TX FSM waiting for RX FSM to finish
- 1.1 GBE TX FSM in dumpdata state
- 1.0 GBE TX FSM in padding state

Page 2

- 2.5 **GBE optical signal detect**
- 2.4 **GBE 62.5MHz DCM locked**
- 2.3 **GBE MGT locked**
- 2.2 **GBE loopback enabled**
- 2.1 reserved, 0
- 2.0 reserved, 0

Page 3

- 3.5 **backplane busy**
- 3.4 **backplane dtack error**
- 3.3 backplane FSM waiting for RX FSM to finish
- 3.2 **SCL parity error**
- 3.1 **SCL frame bad period**
- 3.0 **SCL bad header format**

Page 4

- 4.5 **SCL signal detect**
- 4.4 **SCL DCL locked**
- 4.3 **SCL MGT locked**
- 4.2 **SCL MGT attempting realignment**
- 4.1 **SCL fatal error, can't find any clock!**
- 4.0 SCL I have been kicked!

Page 5

- 5.5 **SCL is driving FAKE bits**
- 5.4 **SCL is driving REAL control bits**
- 5.3 **SCL FirstXing**
- 5.2 **SCL SyncGap**
- 5.1 **SCL Llaccept**
- 5.0 **SCL SCLinit**