NATIONAL
OPTICAL
ASTRONOMY
OBSERVATORY

MAJOR INSTRUMENTATION GROUP
950 N. Cherry Ave.
P. O. Box 26732
Tucson, Arizona 85726-6732
(520) 318-8000 FAX: (520) 318-8303

MONSOON
Master Control Board
(MCB)

Master Control Board Description

NOAO Document MNSN-AD-08-0001
Revision 4.2

Authored by:
Peter Moore
3/29/2005
Please send comments:
pmoore@noao.edu

Revision History

Version	Date Approved	Sections Affected	Remarks
3.0	8/15/2004	All	Revised to reflect revision 2 hardware and version 4.xx firmware products.
4.0	3/30/2005	All	Major revision and additions.
	3/24/2006	All	Edit and reformat – aro.
4.1	8/31/2006	Appendix	Added Appendix II and III.
4.2	9/4/2007	1.4.26 4.2.3 4.3.10.1 4.3.15 Appendix	Reworded description for Bit 4. Bit 5 to 6 is now Bit 5 to 7. Bit 7 was changed to Bit 8. Corrected errors in Table 16. Updated Table 18. Added new paragraph to include instruction “LRB”. Added Appendix IV and Appendix V.

Table of Contents

List of Figures

List of Tables

Preface

This document describes the hardware implementation of the Master Control Board, Revision 2.0, that boots the pixel and sequencer FPGA firmware revision 4.0 and later. Minor revisions and modifications to the capabilities and functionality of this board can be added as an appendix to this document. Major modifications to the functionality of the board that would require extensive modification to this document must originate a new and separate board description document.

Document Scope

This document provides an overall description, as well as detailed information, on the architecture, configuration, testing and functionality of the MONSOON Master Control Board (MCB). It is intended that this document be read by anyone who needs to consider, build, use, or test a MONSOON system that requires this hardware module.

1.0 Board Functional Description

1.1 Purpose of Board

The Master Control Board (MCB) provides the essential control and communication functions required by the detector head electronics (DHE). Every MONSOON DHE chassis requires just one MCB that must reside in the 'SYSTEM' slot or 'Slot 1' of the PCI backplane. This board provides the interface for communication between the Pixel Acquisition Node (PAN) and the DHE electronics. It controls the unidirectional Sequencer Bus on the backplane and provides a programmable sequencer (SEQUENCER MPU) to control DHE functions independently of the PAN to support detector functions. It also provides for DHE clock distribution and a mechanism to synchronize multiple DHE chassis. All communication, control and sequencing functions are built from firmware contained within two Xilinx Virtex300E series FPGAs. One FPGA (U20) is dedicated to data path handling and is called generically the 'Pixel FPGA'. The other (U22) is configured for PAN command decoding, Sequencer Bus control and Clock distribution It also contains the programmable sequencer (MPU Sequencer). This FPGA is generically called the Sequencer FPGA.

1.2 Interface Description

1.2.1 Systran Interface

The MCB provides a seat for a Systran fiber optic CMC module using connectors J3, J4 and J6. The Systran module provides either a 1 Gbit (SLM100) or a 2.4 Gbit (SLM240) bi-directional fiber optic link to the Pixel Acquisition Node (PAN). All communication with the PAN is through this link. The MCB seat for this module is a parallel Front Panel Data Port (FPDP) according to IEEE proposed standard for CMC (P1386/Draft 2.0).

There are two physical FPDP ports available on the MCB. The port designated as J3A, J4A and J6A is the principle data port used by the Systran interface module. This port connects directly to the corresponding data ports on the two FPGA devices to provide the interface to the communication channel. The second FPDP port designated as J3B, J4B and J6B is an alternative site for the Systran FPDP module when some local computing power is required at the DHE node itself. Under these circumstances, a local computing element (or some other function) can be inserted at the prime FPDP port between the MCB FPGA logic and the PAN communication interface (the Systran module) which now occupies the second FPDP port. Making this computing element look like a PAN to the FPGAs and as a DHE to the PAN will allow a seamless integration into the system. Communication between the two FPDP ports is achieved through connectors J7 and J8. This special configuration will be described in an appendix to this document and information in the following paragraphs assumes the Systran module is occupying the prime FPDP port.

Data received from the PAN by the Systran link is transferred to the DHE through J4A data lines **TAD0** to **TAD31** when the signal **/TA_DVALID** is true on the rising edge of **TASTROBE**. Data being transmitted to the PAN is latched into the Systran CMC module from J6A signals **RAD0** to **RAD31** on the rising edge of **RA_STROBE** when **/RA_DVALID** is true. Data received from the PAN is always 32 bits wide and, with two exceptions, always echoed back to the PAN to acknowledge receipt. The exceptions are:

- After a hard boot or soft reset when an asynchronous status message is always echoed for any data received.
- The start exposure command is not echoed.

Data sent to the PAN can be either a message resulting from a command or pixel data. All message data have a ‘sync’ frame appended to them to identify them as messages to the PAN. This is implemented by taking J6A signal **/RA_SYNC** true for one rising edge of **RA_STROBE** one clock cycle after sending the data with **/RA_DVALID**. The nominal transfer rate for data across this interface is at the system clock rate of 40MHz. For details on the data protocols used on this communication link, see document MNSN-AD-01-0005 R1.0, DHE Command and Hardware Interface Description.

J3A and J3B are dedicated to configuring the Systran board during boot through the static configuration jumpers **JP6** to **JP10.** The mpu configuration option is not used.

An external clock can be supplied to some special order Systran CMC modules. This requires the system clock to be set for the nominal frequency required by the Systran (53.1 MHz for SL100) and supplied to the CMC through the PECL driver (U23) and network (C44, C45, R81, R82, R153, R155 and R157).

The complete PAN to DHE communication subsystem can be reset to enable re-synchronization if needed. This is accomplished by utilizing the two PIO lines (**PIO1_IN**, **PIO2_IN**) from the Systran CMC which, when commanded from the PAN low for > 1.5milliseconds, will reset the communication path in the DHE including the Systran CMC itself.

1.2.2 Clock Distribution

The MCB controls the activity of peripheral boards by enabling or disabling the system clock to them. This allows boards that are not specifically required during some periods to shut down to conserve power or reduce induced noise from their activities.

Control of this interface is by directly writing to the clock enable register of the MCB (at address 0x100). Because of some non-standardization within the cPCI specification, some peripheral boards share common clocks. Tables 1 and 2 map the control word written to the clock enable register to physical peripheral boards.

Writing a 'one' to the respective bit in the clock enable register turns that clock source on.

Table 1 - Clock Distribution Control Bit Assignments for 8-Slot Backplanes

Control Word Bit	Backplane Signal	Peripheral Board Clocks
0		Not used, always active
1	CLK23	Board 2 and 3 active
2	CLK23	Board 2 and 3 active
3	CLK45	Board 4 and 5 active
4	CLK45	Board 4 and 5 active
5	CLK6	Board 6 active
6	CLK7	Board 7 active
7	CLK8	Board 8 active
15	BKPLN_SLCT	Set high for 8-slot backplane mapping

Table 2 - Clock Distribution Control Bit Assignments for 6-Slot Backplanes

Control Word Bit	Backplane Signal	Peripheral Board Clocks
0		Not used, always active
1	CLK2	Board 2 and 3 active
2	CLK3	Board 2 and 3 active
3	CLK4	Board 4 and 5 active
4	CLK5	Board 4 and 5 active
5	CLK6	Board 6 active
15	BKPLN_SLCT	Set low for 6-slot backplane mapping

The appropriate mapping for 8-Slot or 6-Slot backplanes is set by bit 15 of the register: Setting this bit high enables mapping for 8-Slot backplanes. When this bit is set low, the mapping is enabled for 6-Slot backplanes.

1.2.3 Sequencer Bus Interface

The Master Control Board controls all activity on the Sequencer Bus. This bus is unidirectional, away from the MCB, and controls all peripheral DHE boards on the backplane. Bus activity is synchronous to the rising edge of the system clock.

There are four groups of signals that make up the Sequencer Bus: eight bits of board select (**/SEL2 TO /SEL8**), two bits of mode (**SEQ_MODE[1:0]**), six bits of device address (**DEV_ADDR[5:0]**), and 32 bits of data (**SEQ_DATA[31:0]**).

The **SEQ_MODE[1:0]** signals allow three types of bus transactions, reset, write and read. These are detailed in Table 3.

Table 3 - Sequencer Bus Mode Bit Definitions

Mode Bits	Transaction	Mode	Description
00	Reset	Hard or Soft	Reboot FPGA or soft reset the addressed board depending on SEQ_DEVADDR[2:0] bits.
01	Read	32 Bits	Read a word from the addressed board
10	Write	16 Bits	Write a 16-bit word to the addressed board
11	Write	32 Bits	Write a 32-bit word to the addressed board

The board select bits (**/SEL2 TO /SEL8**) are used to activate a board for a bus transaction. It is legal for multiple board select lines to be active for a reset or write transaction but only a single board select can be active for a read transaction.

A reset or write transaction takes one clock cycle and a read transaction takes three clock cycles to complete.

Reset Transactions

Reset mode is similar in timing to a write transaction.

Setting all bits of the **SEQ_DEVADDR[5:0]** signals high results in a reboot of the FPGA on the selected board and all configuration data is set to default values.

Setting the **SEQ_DEVADDR[5:0]** signals low during a reset transaction performs a 'soft reset' of the board, only the functionality is reset (i.e. state machines, etc.) and current configuration data is preserved.

NOTE: At this time Sequencer bus data content has no significance in the reset transaction.

Write Transactions

Only the lowest sixty four memory locations (0x0000 to 0x003F) of a peripheral board can be written to in 32-bit mode. In this mode the **SEQ_DEVADDR[5:0]** signals carry the address information to the peripheral board. Data is carried on signals **SEQ_DATA[31:0]**. The data is written into the peripheral board on the rising edge of the board SYSCLK when **/SELN** is set true.

When writing 16-bit data, the address on the peripheral board is carried in the most significant 16 bits of the sequencer data bus (**SEQ_DATA[31:16]**) signals and data contained in the least significant bits (**SEQ_DATA[15:0]**).

Read Transactions

During a read transaction, a 'write' is performed to the peripheral board with the required board address contained in the least significant 16 bits of the sequencer bus signals (**SEQ_DATA[15:0]**). Two clock cycles are allowed for the peripheral board to decode the address and put the requested data onto the Pixel Data Bus; Then the PIXEL FPGA control line **FPGA_CTL1** is taken true by the PAN command decode logic and data from the pixel data bus is latched into the Pixel Bus FIFO.

An additional set of signals, Peripheral Board Acknowledge Strobes (**/ACK2 TO /ACK8**), is defined for this bus. Currently, these signals are not used.

1.2.4 Pixel Data Bus Interface

The Pixel Data Bus (**PIX_DATA[47:0]**) transfers data from peripheral boards to the MCB and from there to the PAN by way of the Systran link. It is a unidirectional 48-bit bus that connects to a 128-word FIFO (Pixel Bus FIFO) within the pixel FPGA. Data loaded into this FIFO is transmitted directly to the PAN by way of the Systran Interface. Data present on the Pixel Data Bus is loaded into this FIFO by two methods:

- If the data is in response to a decoded read command from the PAN, the signal **FPGA_CTL1** is used to strobe data into the FIFO with timing controlled by the Pan Decode logic.
- If the data is pixel data generated by a peripheral board then data is loaded into the FIFO on the rising edge of the system clock when any of the seven **/PIPE_REQ_N[6:0]** signals **and** signal **/PIPE_REQ_N[6]** are active.

This facility allows a peripheral board (Video Acquisition Board) to burst transfer pixel data to the PAN. There is a priority scheme associated with the use of this feature that is handled by the peripheral boards themselves. More details on the priority scheme can be found in the Board Description Document the CCD 8 Channel Acquisition Board (MNSN-AD-08-0004) and the description document, which is currently in development, for the 36-Channel IR Acquisition Board.

1.2.5 Synchronization Interface

To allow multiple DHE chassis to act in unison, two connectors are provided to enable tight, clock cycle-level synchronization. J9 and J14 (which are wired in parallel) accept externally generated synchronization signals and J18 and J19 (also wired in parallel) drive synchronization signals out to other DHE chassis in the daisy chain.

There are two synchronization signals, **EXTCLK** and **EXTSYNC,** that are driven and received as PECL logic levels. In a multiple DHE configuration, one DHE acts as master and generates the EXTCLK from the system clock source. The driven **EXTCLK** signal from a master DHE can be adjusted in phase by writing to an MCB register (***MCB_SYNC_CNTRL***) to provide phase match of slave DHE chassis to the master MCB system clock over controlled impedance cables.

The **EXTSYNC** signal can be generated and driven to J18 and J19 as a master as well as detected and read by the programmable sequencer on the MCB. Defining an MCB as master or slave is controlled by setting bit 4 in the MCB control register (***MCB_DHE_MASTER***). This provides the basic mechanism to begin control processes run by the MPU Sequencer in synchronization with each other.

To establish synchronous operation on multiple DHE chassis, jumpers JP2 and JP3 must be configured. In addition, the SEQUENCER MPU code should be able to use the sync signal that appears in the sequencer command register (***SEQ_CMD_SYNCFLAG***) and set the master flag in the MCB control register.

A slow and simple daisy-chained serial data interface is available for inter-DHE communication. There is one serial data in and one serial data out port assigned to the respective synchronization connectors. The protocol, and therefore internal logic, to control these has not yet been defined. However, there is a simple one bit interface available through the ***MCB_SYNCLNK*** register which can be used to set the output bit and read the input bit.

1.2.6 JTAG Interface, Reset and Power on Boot.

The two Virtex300E programmable logic devices (Pixel and Sequencer FPGAs) that contain all the logic for the MCB are accessible for programming and limited diagnostics by the JTAG connector J10.

The JTAG order is: J10:TDI to Sequencer boot EEPROM (type 18V02) to Sequencer FPGA (type Virtex300E) to Pixel sequencer boot EEPROM (type 18V02) to Pixel FPGA (type Virtex300E) to J10:TDO.

Each FPGA requires the code that defines its function to be loaded at power up. These codes are stored in separate 18V02 EEPROM devices that are automatically read during the power up cycle. U25 contains the code for the Sequencer FPGA (U22) and U26 contains code destined for the Pixel FPGA (U20). After power has stabilized to the FPGA devices, the time to boot the code and perform power up initialization is approximately 30 milliseconds.

In addition, the reloading of code and initialization cycle can be initiated manually by pressing the front panel reset button (SW1) and by PAN command by writing to the (***MCB_REBOOT***) register.

1.2.7 Front Panel Indicators

There are two front panel LED indicators used for basic diagnostics of the board. D18 is controlled by firmware in the Pixel FPGA and D19 by firmware in the Sequencer FPGA.

During power up, each indicator is off until the firmware for the respective FPGA has been booted and started. After booting their respective code, the indicators will be steadily on until an asynchronous command has been sent from the PAN node to synchronize the communication link. At this time both indicators will go out until programmed to indicate some functionality through their respective control registers (***MCB_PIXLEDCNTL***, ***MCB_SEQLEDCNTL***). For version 4.2 FPGA firmware, Table 4 lists the functionality that is provided.

Table 4 - Front Panel Indicator Functions

MCB_PIXLEDCNTL		**MCB_SEQLEDCNTL**	
BIT	**Function**	**BIT**	**Function**
0	FPDP TM Write Request Active	0	FPDP /**TA_DVALID** Flag True
1	Funct. Cmd. Decoder Active	1	PAN Seq. Bus Access Request
2	Pixel Data Request Active	2	MPU Seq. Bus Access Request

1.2.8 Serial Configuration EEPROM

U30 is an optional serial data EEPROM device that, when installed, can be used to hold local configuration data for the DHE and/or instrument configuration. This device conforms to the I2C interface standard and is read by the Pixel FPGA after power up or when commanded by the corresponding control register (***MCB_SERCFGRD***). Data read from this device is available from the 128 locations starting from ***MCB_SERCONFIG***.

In addition, the I2C interface extends out to the backplane through J5:B17 and J5:B18. This enables an external compatible serial EEPROM (embedded inside an instrument for example) to be read by the PAN. The MCB mounted device is strapped for address 1 and since, after power up or reset of the MCB, the lowest addressed EEPROM device is searched for and read, an external EEPROM should be strapped for address 0.

CAUTION: Some 24LC02 devices do not conform to the external addressing methods.

Data may be written to the serial EEPROM by overwriting the shadow memory space with new data and accessing the serial EEPROM write function at ***MCB_SERCFGWRITE***.

1.2.9 EX Data Port

There are four data lines accessible through the register ***MCB_EXDATA*** that allow writing control bits to a suitable interface connected to the J3 connectors of the FPDP ports. These bits are for future use with a compute element board fitted to the primary FPDP port.

1.2.10 Shutter and Detector Temperature Control Interface

There are provisions to bias and monitor eight temperature sensor channels. These can be either diode or PTC type sensors. These values can be read by the pan through the eight ***MCB_ADCREG*** registers. There are also provisions for two heater control ports and two simple open collector high current drives which are controlled by DACs. These are controlled via the ***MCB_DACREG*** registers. These areas of functionality have not yet been fully implemented in the MCB. In future hardware and firmware revisions these functions may be enabled and described in more detail.

1.3 Data Paths

Data paths, with some exceptions, are 32 bits wide. Figure 1 shows the principle data paths within the Pixel and Sequencer FPGAs of the MCB.

Data received from the PAN via the Systran CMC receiver port is echoed back to the Systran transmitter port to acknowledge reception to the PAN. A sync signal is appended to this function to identify the transmitted word as a message to the PAN. However, if the DHE has undergone a reset from either a programmed event, manual reset, or an unexpected power outage, the word echoed back to the PAN is an asynchronous status message. See the document MNSN-AD-01-0005 R1.0, DHE Command and Hardware Interface Description for more details on this.

The same data that is received from the PAN by the Sequencer FPGA is inserted onto the FPGA Interface FIFO for transmission to the Pixel FPGA. This interface is a 16-bit bi-directional data path controlled by signals **FPGA_CTL3, FPGA_CTL4** and **FPGA_CTL5**. Current firmware versions only employ a unidirectional data path of this interface sending data and commands only from the sequencer FPGA to the Pixel FPGA. Bi-directional functionality is maintained for future development of the MCB firmware.

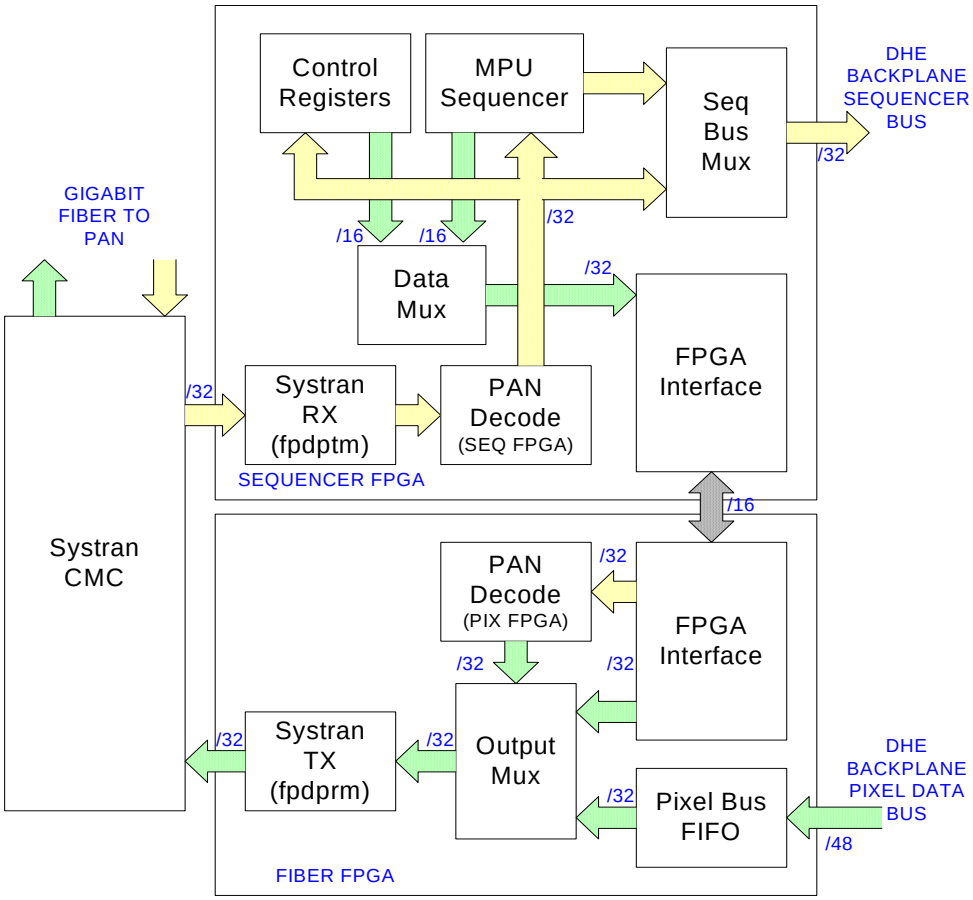
Command decoding functions in the Sequencer FPGA are blocked by the internal reset until an asynchronous command is received from the PAN. This forces synchronization of the PAN ⇔ DHE communication channel.

After synchronization occurs, the data received from the PAN is interpreted by the PAN Command Decode module as a command for execution by the DHE. The data format and protocol used by the PAN to communicate with the DHE is described in document ICD-6.0 "Generic DHE Interface Document". There are just four commands that the PAN can send:

- Asynchronous command
- Write data
- Read data
- Start Exposure

Write data directed to a peripheral board are decoded and sent to the backplane bus via the Sequencer Bus multiplexor.

Write data directed to the MCB are locally decoded and stored internally in the MCB FPGAs without access to the sequencer bus. Access for reading and writing by the PAN is asynchronous to the MPU sequencer operation. Any PAN operation is delayed if the MPU sequencer currently has the sequencer bus control. If the MPU sequencer is enabled, the MPU sequencer program memory space is locked out from access by the PAN.



MCB Principle Data Paths
Figure 1

A read command addressed to a peripheral board is also forwarded through the Sequencer Bus by way of the multiplexor to the peripheral board. Return data from the peripheral board is put onto the Pixel Data bus and strobed directly into the Pixel Data Bus FIFO using control line (**FPGA_CTL1**).

The Pixel Data Bus FIFO can receive either 32-bit data or 48-bit pixel data. The 48-bit pixel data is sent as two 24-bit integers in parallel which allows a maximum burst transfer rate of 80 MPixels / Second across the Pixel Data Bus in current firmware versions.

A normal 32-bit read controlled by the PAN Decode logic takes 3 clock cycles to complete. This allows the peripheral boards enough time to access the data at the requested address and put the data onto the Pixel Data Bus before activating the Pixel Data Bus FIFO write strobe.

Data received from the PAN that contain commands addressed to the MCB are decoded directly within the Sequencer FPGA. Read commands are processed in a slightly different manner since the return data is passed once again through the FPGA Interface to be seen at the output of the Pixel FPGA Interface and latched into the Systran TX module via the Output Data Multiplexer.

Pixel data can be burst to the Systran CMC TX port at a rate of either 80 or 40 MPixels / Second. This depends on the MCB register (***MCB_DBLPIX_XFER***) which, when set true, concatenates two 16-bit pixel values (**PIX_DATA[39:24] + PIX_DATA[15:0]**) into each Systran 32-bit word being sent to the PAN. When false, the pixel data is read from the Pixel Data Bus FIFO as 24 bits and expanded to 32 bits for transfer by the Systran board.

NOTE: The maximum *sustained* data rate of this interface is set by the selection of the Systran product coupled to the data latency of the PAN software / hardware.

Message data sent to the PAN (that requires a sync pulse to be appended) can be streamed to the PAN at 20MWord / Second. Table 3 contains information about the widths and depths of the various FIFO memories on the MCB.

Table 5 - MCB Data Path FIFO Descriptions

Module Name	Input Word size	Output Word Size	FIFO Depth
Sequencer FPGA Systran RX	32	32	64
Sequencer FPGA TX Interface	32	16	8
Pixel FPGA RX Interface	16	32	8
Pixel Data Bus FIFO	32 / 48	32	128 / 128
Pixel FPGA Systran TX	32	32	0

1.4 Control Functions

All control functions that the MCB is capable of are initiated by writing to specific memory locations on the MCB The board address for the MCB is always 1. For details on the format of the data written to the DHE from the PAN, please refer to document MNSN-AD-01-0004 R 1.0 ICD 6.0 Generic DHE Command and Data Stream Interface. A summary of the memory space for the MCB is dispalyed in Table 6 followed by a more detailed description of each function.

Table 6 - MCB Memory Map

Address	Function Name	Function Description
0x0000	SEQ_ITR	Requested Integration time register
0x0001	MCB_TSTCNTR	Synthetic Test pixel counter
0x0100	MCB_CLKENBL	System clock enable register
0x0101	SEQ_ITC	Actual Integration timer counter
0x0102	SEQ_CMD	MPU Sequencer command register
0x0110 to 0x011F	SEQ_LOOPREG	MPU Sequencer loop counters
0x0200 to 0x027E	MCB_SERCFG	127 x 16 bit word non-volatile storage space – possibly to be used to conserve system configuration data.
0x027F	MCB_SERCFGSTAT	SERCFG memory status register
0x0280	MCB_SERCFGRD	Used to read a specific serial EEPROM address

Table 6 - MCB Memory Map (Cont.)

Address	Function Name	Function Description
0x0281	MCB_CLKCTL	System clock phase control for multiple DHE clock phasing
0x0282	MCB_PIXLEDCTL	3 bit led indicator control register
0x0283	MCB_SYNCLINK	1 bit synchronization port in/out port
0x0284	MCB_HSSLINK	Configurable high speed serial link
0x0285	MCB_PIXSTAT	Pixel FPGA status register
0x0286	MCB_EXDATA	4 bit register available to FPDP port
0x0288	MCB_DBG_REG_1	Debug register that is firmware dependent
0x0289	MCB_DBG_REG_2	Debug register that is firmware dependent
0x028A	MCB_DBG_REG_3	Debug register that is firmware dependent
0x028B	MCB_DBG_REG_4	Debug register that is firmware dependent
0x02A0 to 0x02A7	MCB_DACREG	Controls 8 DACs used for heater and shutter control.
0x02C0 to 0x02C7	MCB_ADCREG	Reads 8 temperature sensors.
0x02C8 to 0x02CF	MCB_ADCMODE	Configures the read mode for the 8 temperature sensors
0x02FC	MCB_DBLPIX_XFER	Enables double pixel transfers to Systran
0x1000 to 0x1BFF	SEQ_PGMMEM	MPU Sequencer program code memory space
0x4000 to 0x43FF	SEQ_PATMEM	MPU Sequencer pattern memory space
0xFFF9	MCB_SEQLEDCTL	3 bit register controlling the Seq FPGA indicators
0xFFFA	MCB_SERNUM	Unique 32 bit serial number
0xFFFB	MCB_TEMP	Current board temperature.
0xFFFC	MCB_CTLREG	MCB control register
0xFFFD	MCB_STATREG	MCB status register
0xFFFE	MCB_IDENTREG	MCB board identity code (read only)
0xFFFF	MCB_FIRMVERS	MCB firmware code revision (read only)
0xFFFE	MCB_RESET	MCB Soft reset register (write only)
0xFFFF	MCB_REBOOT	MCB firmware reboot register (write only)
	SEQ_ENBLREG	MPU Sequencer enables register

1.4.1 Integration Time Register (SEQ_ITR)

This is a 32-bit register that holds the absolute time required for an integration. The value is used by the Integration Time Counter to determine the end of integration for an exposure. If necessary, the register can be written to with a new value during the integration phase to reflect a change of integration time. The value written to this register is in units milliseconds. The minimum value is zero and the maximum 4294967295 (i.e. 1193 hours).

1.4.2 Test Counter Register (MCB_TSTCNTR)

A 32-bit value written to this register arms the test counter function. When a START EXPOSURE command is received from the PAN, the written value of pixels will be sent to the PAN at intervals of 1μs. The data for each pixel will be the current 32-bit counter value, which is decremented after each pixel is sent. When the test counter reaches zero, it will become inactive and another value must be written to the register to repeat the test. Its primary purpose is to provide data path verification between the MCB to PAN. It is necessary to prevent the MPU SEQUENCER from responding to the START EXPOSRE command and sending its own data to the PAN by disabling the MPU SEQUENCER. Disabling the MPU SEQUENCER can be done by writing a zero to the MCB Control Register (CTRL) bit 0 (***SEQ_ENABLE***).

1.4.3 Clock Enable Register (MCB_CLKENBL)

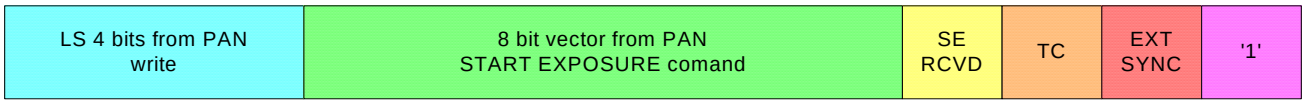
A 16-bit value written to this register enables the system clock to each peripheral board slot on the backplane that has a bit set in the value. The mapping for this function is detailed in Table 1. Although the backplane and peripheral boards are designed to be synchronous, that is, they require a clock to perform any transaction or function, peripheral boards have a limited ability to access their Board Identity Register and Firmware Code revision register without activating the appropriate clock to that backplane slot. This enables a PAN to determine which boards and what functionalities are present in the DHE without activating a clock to each peripheral backplane slot in turn.

1.4.4 Integration Timer Counter (SEQ_ITC)

The integration timer counter is a 32-bit read only register. The counter can be enabled and disabled from the MPU Sequencer Enables register using microcode. When enabled, the value will increment each millisecond until the value stored in the Integration Time Register is reached or surpassed. At this time a flag (***SEQ_CMD_ITC***) is set in the MPU Sequencer Command register. The counter will continue to increment until it is disabled by an MPU Sequencer microcode instruction. In this way, and assuming the microcode is designed for this, the PAN may read the actual integration time from this register after integration is complete. This counter can be ‘paused’ by the PAN by using the appropriate bit (bit 1) in the MCB_CTLREG register (***SEQ_INTPAUSE***).

1.4.5 Sequencer Command Register (SEQ_CMD)

This 16-bit register provides the mechanism for conditional branching within the MPU SEQUENCER microcode. The MPU SEQUENCER instruction set provides one conditional jump command (JCB) that requires a destination for the jump and a bit number to test. The instruction branches if the specified bit in the Sequencer Command register is set. Therefore there are 16 possible branches that can be specified. The lowest 4 bits (3:0) of the command register are set by hardware. The middle 8 bits (11:4) are set from the START EXPOSURE command as the required start vector within the sequencer program code. The last 4 bits (15:12) are written directly from the data value written to this register from the PAN. Figure 2 illustrates the relative bit positions for the branch conditions.



MPU SEQUENCER Command Register Structure
Figure 2

Bit 0 is tied to '1' to provide an unconditional branch instruction (JMP).

Bit 1 reflects the state of the external **SYNC** signal used for multiple DHE synchronization.

Bit 2 is set true whenever the value of the Integration Time Counter is greater than or equal to the Integration Time Register.

Bit 3 is set whenever a START EXPOSURE command is received from the PAN. This indicates that the condition of bits (11:4) has been established. This bit can be reset by a microcode instruction to the SEQ_ENBLREG.

Bit 11 to **4** is set by the START EXPOSURE command vector and is used to evaluate what type of sequence should be carried out by the MPU Sequencer microcode.

Bit 15 to **12** are directly written from the least significant four bits of data during a PAN write command to this register. These bits may be tested and used to control modes of functionality within the MPU Sequencer microcode.

1.4.6 Sequencer Loop Counters (SEQ_LOOPREG)

To provide for looping structures within the MPU Sequencer microcode, sixteen 16-bit registers are available. Each register can be tested and branching controlled by the LRB (Loop Register Begin) and LPE (Loop End) instructions. These registers are static, that is, their value remains unchanged by the effect of microcode instruction execution.

1.4.7 Serial EEPROM Configuration Memory (MCB_SERCFG)

There are 127 locations of 16-bit words reserved as shadow RAM to a small serial EEPROM that is read through an I2C interface. The EEPROM can be physically located on the board or remotely located (embedded into an instrument or camera). The EEPROM is read and copied to this shadow RAM after a power on reset, after rebooting, and under command from the MCB_SERCFGRD register. The contents of this memory space has no functional significance to the MCB but can be read and written to directly from the PAN.

1.4.8 Serial EEPROM Shadow Memory Status (MCB_SERCFGSTAT)

This register supplies the status of the I2C interface and the status of the contents for the shadow RAM. This can be interpreted as follows:

BIT 2 to 0 Device addr: Physical device address of last read or write.

BIT 3 Always zero.

BIT 10 to 4 Shadow RAM addr: Actual address value currently being read or written to during I2C activity or where an error occurred during operation.

BIT 11 Always zero.

BIT 12 Read process active: I2C is reading bits.

BIT 13 Clock process active: I2C is sourcing clock.

BIT 14 Shadow memory data valid: The contents of the shadow memory accurately reflect the contents of the EEPROM.

BIT 15 I2C process is busy: Contents of shadow ram is not valid yet.

In addition, writing to this register with the least significant 3 bits of write data set to identify a physical device address will write the contents of the shadow RAM to the addressed serial EEPROM.

1.4.9 Serial EEPROM Read Command Register (MCB_SERCFGRD)

Writing to this register with the least significant 3 bits of write data set to identify a physical device address will read the addressed EEPROM contents to shadow RAM.

1.4.10 System Clock Phase Control Register (MCB_CLKCTL)

Writing 8 bits of significant data to this register controls the clock phasing for the synchronization port clock recovery and clock distribution. This functionality is used when two or more DHE chassis need to be synchronized. Normally, each DHE sources its own clock to run the logic, however, when multiple DHE chassis are used to control one focal plane it is important to synchronize their operation to avoid impulse noise sources from each system interfering with the others.

When using multiple DHE chassis in synchronized operation, one DHE is nominated as a master and feeds the system clock to the slave DHEs in a daisy chain configuration of controlled impedance cables. Each slave has two clock skew adjustment regimes that can be used together to arrive at a sub-nanosecond phase alignment between all DHE chassis.

The DHE system clock is derived from a nominal 40MHz clock source and passed through a PI6C3991 clock skew buffer to drive the logic. This skew buffer has four channels of skew control of which two are important for synchronized operation. The clock source for a master DHE is the crystal oscillator. The clock source for a slave DHE is the PECL logic receiver which is driven from the EXT_CLKIN and /EXT_CLKIN signals at J14 or J9. Clock selection is achieved with JP3.

This clock source drives all the skew buffers in the PI6C3991 device. From this source, a programmable skew controlled by this register can be applied to channel 1 and 2 to control system clock phase to this local DHE chassis (skew channel 1) and to the system clock sent out to the next DHE in the daisy chain through the PECL driver U36 and connectors J18 and J19 (skew channel 2).

The range of adjustment for each clock phase is approx. ± 4ns with a 40MHz system clock. This implies that the propagation time through the originating PECL driver plus the cable plus the receiving PECL receiver must be less than 8ns or a multiple of 25ns (one 40MHz clock phase) to achieve synchronization.

The PECL devices have worst case propagation times of 2.5ns and nominal propagation times of 1.7ns each. This leaves minimal time for cable propagation (3ns to 5ns or approximately 0.75 meter length) The default value of 0x00 sets the clock phasing to nominal center point. Table 7 indicates the skew adjustment codes.

Table 7 - Programmable Clock Skew Values

Skew Channel 1 - DHE System Clock		**Skew Channel 2 - Sync Port Clock**	
Bits 8 to 4	**Clock Delay**	**Bits 3 to 0**	**Clock Delay**
0x0	0 ns	0x0	0 ns
0x1	+ 0.96 ns	0x1	+ 0.96 ns
0x2	+ 1.92 ns	0x2	+ 1.92 ns
0x3	+ 2.88 ns	0x3	+ 2.88 ns
0x4	+ 3.84 ns	0x4	+ 3.84 ns
0x5	+ 3.84 ns	0x5	+ 3.84 ns
0x6	+ 3.84 ns	0x6	+ 3.84 ns
0x7	+ 3.84 ns	0x7	+ 3.84 ns
0x8	- 3.84 ns	0x8	- 3.84 ns
0x9	- 3.84 ns	0x9	- 3.84 ns
0xA	- 3.84 ns	0xA	- 3.84 ns
0xB	- 3.84 ns	0xB	- 3.84 ns
0xC	- 3.84 ns	0xC	- 3.84 ns
0xD	- 2.88 ns	0xD	- 2.88 ns
0xE	- 1.92 ns	0xE	- 1.92 ns
0xF	- 0.96 ns	0xF	- 0.96 ns

1.4.11 Pixel FPGA LED Indicator Control Register (MCB_PIXLEDCTL)

There are three bits that control the function of the Pixel FPGA indicator LED (D18). After power on or a reboot, the indicator will be on permanently until communication with the PAN has been established. After this, writing to one or more of the control bits will provide a 10ms indicator flash each time the specific function goes true. The functions that can be assigned for this are listed in Table 8.

Table 8 - Pixel LED Indicator Function Codes

MCB_PIXLEDCTL	
BIT	**Function**
0	Flash each time a data word (pixel or message) is written to the Systran FPDP port
1	Flash each time the Pixel FPGA is busy decoding a PAN CMD (functions with address ranges 0x0200 to 0x02FF).
2	Flash each time a pixel data value is written to the Systran FPDP port

1.4.12 Synchronization Link Serial Bit Register (MCB_SYNCLINK)

There is a single bit available to be written and read by the PAN that is carried across the synchronization link ports. When reading this function, the data bit is that which appears on pin 1 of J9 or J14. When writing this bit, the data appears on pin 1 of J18 and J19. This enables a slow and primitive data exchange between DHE chassis belonging to the synchronized group. Currently this port has no function and is reserved for future use.

1.4.13 High Speed Serial Interface Port Register (MCB_HSSLINK)

This function is a place holder for a future expansion to incorporate two high speed serial ports for detector control. Althought these ports have no assignment, the hardware for them is in place.

1.4.14 Pixel FPGA Status Register (MCB_PIXSTAT)

Currently this is a place holder. Future use may be as a general purpose FPGA status register.

1.4.15 External Data Register (MCB_EXDATA)

This function allows writing four bits of data to the J3 configuration connector on the two FPDP ports. The format nor protocol is not defined but my be enabled to convey status / state information to a compute element when one is fitted to the primary FPDP port.

1.4.16 Pixel Data Counter Register (MCB_PIXCNTR)

This 32-bit register contains the count of the number of data writes to the Systran link that did not have a sync signal associated with them. Currently these data are defined as pixels and so a diagnostic check can be made to determine how many pixels were sent to the PAN. Writing anything to this register will reset the count to zero.

1.4.17 Message Counter Register (MCB_MSGCNTR)

This 32-bit register contains the count of the number of data writes to the Systran link that did have a sync signal associated with them. Currently these data are defined as messages and so a diagnostic check can be made to determine how many messages were sent to the PAN. Writing anything to this register will reset the count to zero.

1.4.18 DAC Register Control (MCB_DACREG)

Currently not implemented, these registers are a place holder for the temperature control heater supplies and shutter control functions.

1.4.19 ADC Data Registers (MCB_ADCREG)

Currently not implemented, these registers are a place holder for the temperature measurement functions. To read the current temperature it is necessary to first write to the address of the channel of interest and then read back the result after a delay of at least 50 microseconds.

1.4.20 ADC Mode Registers (MCB_ADCMODE)

Currently not implemented, these registers are a place holder for the temperature measurement mode setting functions.

1.4.21 Sequencer Program Code Memory Space (SEQ_PGMMEM)

The microcode used to control DHE functionality is stored within this 1K memory block. This memory is written and read as 16 bits wide, however, the microcode is based on a four-bit word slice so there is effectively 4K word memory for the MPU SEQUENCER microcode. This memory space is blocked to the PAN when the MPU sequencer is running (bit 0 of the MCB_CTLREG is 1).

1.4.22 Sequencer Pattern Memory Space (SEQ_PATMEM)

The MPU SEQUENCER functionality is controlled by the microcode stored in the Sequencer Program Code Memory space, however, the data that is actually put onto the Sequencer Bus during MPU SEQUENCER control comes from the Sequencer Pattern Memory space. This memory space appears to the PAN as a 4K x 16-bit word block but is used by the sequencer as 2K x 32-bit words. This pattern memory contains data that is to be written to peripheral boards at certain times under control of the microcode. Any memory location on any peripheral board can be written to with data from the pattern memory. Memory locations on the MCB cannot be written to under microcode control with the exception of the SEQ_ENBLREG which can *only* be written to under microcode control. It alsodoes not appear in MCB Memory space visible from the PAN.

1.4.23 MCB Sequencer LED Indicator Control Register (MCB_SEQLEDCTL)

There are three bits which control the function of the Sequencer FPGA indicator LED (D19). After power on or a reboot, the indicator will be on permanently until communication with the PAN has been established. After this, writing to one or more of the control bits will provide a 10ms indicator flash each time the specific function goes true. The functions that can be assigned for this are listed in Table 9.

Table 9 - MCB Sequencer Function Codes

MCB_SEQLEDCTL	
BIT	**Function**
0	Flash each time a data word is received from the PAN
1	Flash each time the PAN command decode logic requests use of the sequencer bus
2	Flash each time the MPU Sequencer logic requests use of the sequencer bus

1.4.24 Silicon Serial Number Register (MCB_SERNUM)

Reading this register returns a 32-bit word that is a unique serial number read from U27.

1.4.25 Board Temperature Register (MCB_TEMP)

Reading this register returns a 10-bit signed number that indicates the current ambient temperature of the board. Each LSB corresponds to 0.25deg. C. The temperature is read by first writing to the register (with whatever data), waiting at least 25 microseconds, and then reading the register.

1.4.26 MCB Control Register (MCB_CTLREG)

The MCB Control register provides overall control of the MPU SEQUENCER. The bit assignments for this register are as follows:

Bit 0 ***SEQ_ENABLE***: Setting this bit to '1' allows the sequencer to begin decoding and executing microcode instructions. Whenever the signal is de-asserted, the MPU enters into the reset state after executing the current instruction. All the board selects and the Sequencer Bus request signals are masked into non-active state. The program counter is reset to zero.

Bit 1 ***SEQ_INTPAUSE***: Setting this bit to '1' will stop the Integration Time Counter (ITC) from incrementing. Resetting this bit allows the ITC to continue counting up.

Bit 3 to **2**: ***SEQ_CLKDIV***: This 2-bit pattern defines the rate at which the MPU SEQUENCER instructions are executed. Note that current firmware runs the MPU SEQUENCER at half the system clock frequency with ***SEQ_CLKDIV*** set to zero . Each microcode instruction takes exactly one sequencer clock cycle to complete.

Table 10 - MPU Sequencer Instruction Cycle Time

Bits		
3	2	**Instruction cycle time**
0	0	50ns (20 MHz)
0	1	100ns (10 MHz)
1	0	200ns (5 MHz)
1	1	400ns (2.5 MHz)

Bit 4 ***DHE_MASTER:*** When set, this bit establishes the DHE as a mster amd enables the output of the external SYNC_OUT signal on J18/J19. In the default state of zero (0), the DHE is considered a slave DHE and the external SYNC_IN signal on J9/J14 is routed to the CMD register SYNC bit. When configured as a master DHE, the SYNC bit in the CMD register reflects the state of the SYNC RS Flip Flop controlled through the EFR.

Bit 5 to**7**: Currently not used.

Bit 8 to **15** ***SEQ_SYNCDELAY***: This 8-bit value sets the delay (units of milliseconds) between the time the MPU SEQUENCER resets the SYNC bit in the EFR until the hardware signal is emitted by the master DHE. This delay allows sufficient time for slave DHEs to be configured with a start exposure command by their relative PANS before looking for the SYNC event.

1.4.27 MCB Status Register (MCB_STATREG)

This read-only 16-bit register contains status information from various functions on the MCB. It is currently not used to any advantage, although the least significant bits of the MPU Sequencer program counter can be seen to provide limited feedback on what the microcode is doing.

1.4.28 MCB Board Identity Code (MCB_IDENTREG) (MCB_RESET)

Reading from this address will return data that will identify this board as a Master Control Board (Code = 0x0190)

Writing to this address will result in the MCB going through a soft reset cycle. This will stop the MPU SEQUENCER, load zeros into the Clock Enable and Test Counter registers, and release the Sequencer Bus. Further information on board identity codes and firmware revision values can be found in Appendix I.

1.4.29 MCB Firmware Code Revision (MCB_FIRMVERS)(MCB_REBOOT)

Reading from this address will return data that will identify the version of firmware running on the MCB. The format of the data is binary that should be divided by 100d to find the correct revision number (XX.nn).

Writing to this address will result in the MCB going through a hard reset cycle. This will set signal **BOOT_N** true and that will result in a reloading of the firmware of the MCB to the FIBER and SEQUENCER FPGAs. This will reset all configuration data to default values and place the communication link into the asynchronous state awaiting an asynchronous command from the PAN.

1.4.30 MPU SEQUENCER Enables Register (SEQ_ENBLREG)

The MPU SEQUENCER Enables register can only be written to from microcode running on the MPU sequencer. It allows control over signals directly related to the MPU function. It cannot be read from. Table 10 outlines the functionality of this register.

Table 11 - MPU Sequencer Enables Register Bit Definitions

Bit	Function	Bit	Function
31		15	Set **SYNC_OUT** signal high.
30		14	Set **SYNC_OUT** signal low
29		13	
28		12	
27		11	
26		10	
25		9	Set LSR Auto-cancel feature off
24		8	Set LSR Auto-cancel feature on
23		7	
22		6	
21		5	
20		4	
19		3	
18		2	Stop Integration Time Counter
17		1	Start Integration Time Counter
16		0	Reset Start Exposure Flag

Bit 0 Reset SE Flag: Resets the Start Exposure flag in the Command register. This allows the microcode to avoid re-triggering on this flag when returning from a sequence.

Bit 1 Start Integration Time Counter: Allows the microcode to elect when to start timing an exposure.

Bit 2 Stop Integration Time Counter: Permits the microcode to elect when to stop the integration timer to accurately register the actual elapsed integration time.

BIT 7 TO **3** NOT USED.

Bit 8 Enable a feature of the MPU SEQUENCER that automatically cancels the board select line one clock cycle after the instruction LSR (Load Board Select Register) has been issued. This eliminates the need to specifically cancel the board select line in the microcode itself.

Bit 9 Disable the automatic canceling feature described above. In this mode, after each LSR instruction, a zero would normally be written out to deselect the board to which the pattern memory data was sent. Disabling this feature can be useful to stream out patterns to a specific board / board address each MPU SEQUENCER clock cycle by using the INC and DEC microcodes to step through sequential pattern memory locations.

BIT 13 TO **10** NOT USED.

Bit 14 Set SYNC signal Low: Allows control over the sync signal used to synchronize multiple DHE chassis.

Bit 15 Set SYNC signal High: Allows control over the sync signal used to synchronize multiple DHE chassis.

Bit 31 to **16** Not Used.

2.0 Board Hardware Description

Refer to drawing MNSN-EL-04-2001

2.1 Power Supplies

The MCB requires three power supplies. The majority of the logic is powered by a 3.3v supply (**+3.3VD**). This supply can be either supplied from the backplane (P1) or generated internally from U38. The jumper JP5 selects the source for this supply. Note that the 'newer' Systran FPDP CMC modules require only a 3.3V supply, however, these models overstress the capabilities of the internal regulator and an external supply must be used. The internal logic cell of the FPGAs requires 1.8v (**+1.8V**) that is generated on board by U39. U38 is supplied from a 5v logic supply (**+5VD**) that must be present on the backplane through P1. U39 is supplied from the 3.3V supply. The nominal current draws on these supplies are shown in Table 11.

Table 12 - Power Supply Current Draws

Supply	Quiescent	Peak
5v (with separate 3.3V supply)	0.6 Amp	1.7 Amp
3.3v (Including Systran power)	1.3 Amp	2.0 Amp
1.8v		

There are provisions to supply a separate ± 12V supply by way of P5 to power the temperature and shutter control circuitry. These hardware functions are currently undergoing testing.

2.2 Logic Section

2.2.1 Clock Generation

All logic on the MCB and the DHE backplane is synchronous to the master clock generated by the MCB. If the MCB is mounted in a master DHE, then this 40MHz clock is generated by Y1. If the MCB is in a slave DHE, then the master clock is fed through the synchronization interface at J9 or J14 and JP11 can be used to disable the output of the slave crystal oscillator by tying pin 2 of JP11 to DGND. This will disable the on board oscillator (Pin 1 is incorrectly tied to +3.3VD and a separate jumper wire must be used). Jumper JP3 selects the clock source depending on master or slave use. U19 provides a means to adjust the clock phase to enable close clock phase matching between master and slave DHE chassis. The MCB internal clock is buffered by U21 and fed to the Pixel and Sequencer FPGAs through their respective global clock IO pins. An external clock is fed out to slave DHE chassis through U36 and J18 or J19. An analog of the DHE clock is available at the connectors P1 or P2 to provide a connection to an oscilloscope for clock synchronization.

Board clocks to the peripheral boards mounted in the backplane are gated versions of the internal clock generated within the Sequencer FPGA and controlled by the MCB_CLKCTL register. The system clock can also supply a drive to the Systran CMC module via P3 to allow fully synchronous operation. To enable this feature, U23 and the PECL network consisting of C44, C45, R81, R82, R153, R155, and R157 require populating. In addition, the master clock frequency should be matched to the requirements of the Systran CMC module (currently 51.3 MHz for the SLM100 CMC).

2.2.2 Reset and FPGA Boot Logic

A power on reset active low signal is generated by the linear supply U39. This signal (**/PROGRAM**) is asserted for 100ms after the core voltage becomes stable. When this signal is released both FPGA devices begin their boot process which basically entails sequencing the signals **/INIT** low then high, asserting **/DONE** false and supplying a clock to the **CCLK** pin. This resets the EEPROM internal address counter, enables the data output through D0 to the **EDAT** signal, and begins serially transferring the function code to the FPGA cores. After the load sequence is completed, the FPGA takes the **DONE** signal high and executes an internal startup sequence that enables the output pins of the FPGA. This process takes approx 30 ms to complete. The Pixel FPGA code is programmed through the one of the FPGA load parameters (Gen. Programming File/Properties/Configuration Options/Configuration Rate) to run slower than the Sequencer FPGA to ensure that the main reset signal asserted by the Sequencer FPGA is asserted before the Pixel FPGA comes alive. The same process as a power on reset can be forced either by a PAN command (writing to the ***MCB_REBOOT*** register) or by way of the front panel switch SW1 which takes the **/PROGRAM** signal low while activated.

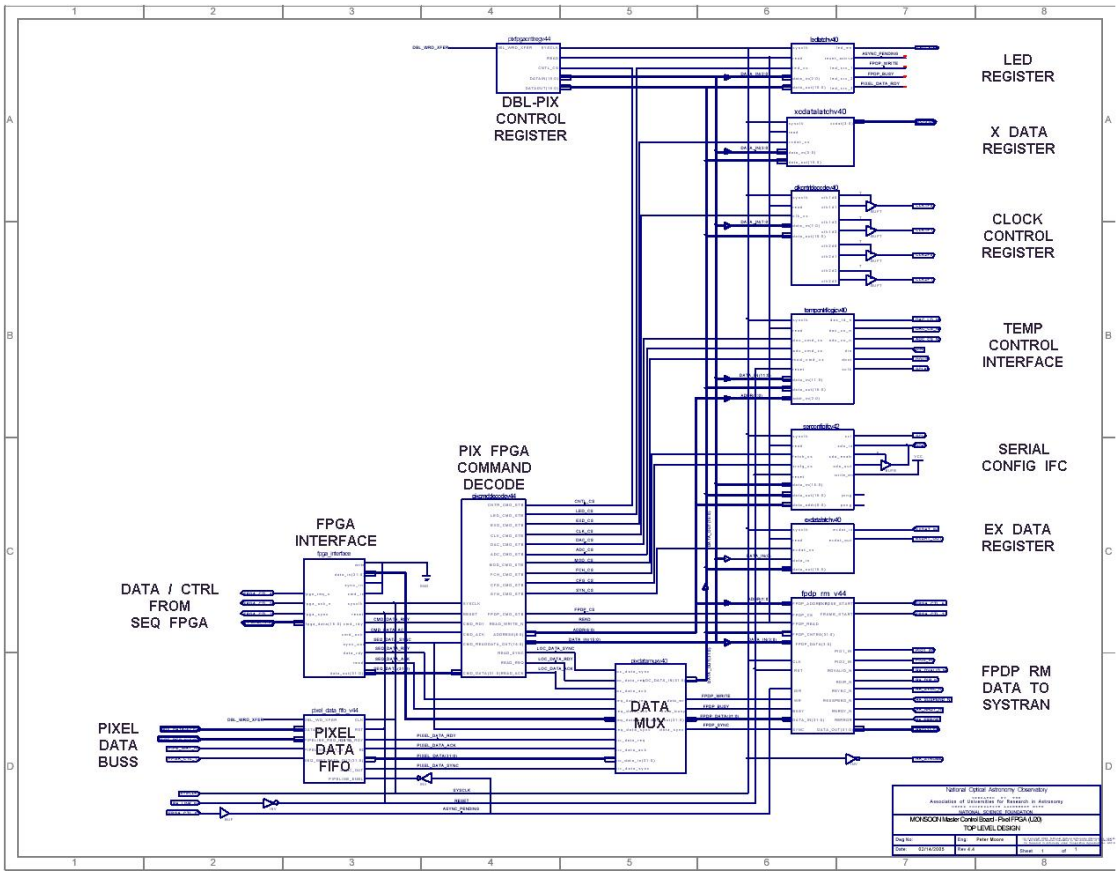
2.2.3 FPDP Data Ports

There are two physical ports available for mounting CMC-type mezzanine cards to the MCB. In most instances, a Systran SL100/240 bi-directional fiber communications module will occupy the first port. This primary port is available through connectors J3A, J4A, and J6A. J3 supplies the static configuration logic required by the Systran module. It also supports the optional external clock for this module. J4A supplies a 32-bit data path for data coming into the DHE from the PAN. Data is strobed out of the Systran module on the rising edge of **TA_STROBE** when **/TA_DVALID** is low. TP7 and TP8 can be used to monitor this data. **TSTROBE** is generated internally to the Systran module and is asynchronous to the DHE system clock. The **/TA_SYNC** signal is not currently used by the FPGA logic. **PIO1_OUT**, **PIO2_OUT**, and **/TA_DIR** are general purpose signals not related to the data flow. **PIO1_OUT** and **PIO2_OUT** are used to initiate a communications reset from the PAN. By taking these two signals low for > 1µs, the Sequencer FPGA initiates an internal reset of all data FIFOs, resets the Systran module, and locks out any PAN command decoding until an asynchronous command has been received. There is a 50ms retrigger delay on these PIO signals. The signals **/TA_DIR** and **TA_ERROR** are not used in the current implementation. **/TA_NRDY** and **/TA_SUSPEND** are data flow control signals that are not currently used.

The J6A connector provides a 32-bit data path and control for data leaving the DHE for transport to the PAN. Similar to the J4 description, data is strobed into the Systran CMC module on the positive edge of **RA_STROBE** when **/RA_DVALID** is asserted low. **RA_STROBE** is currently the DHE system clock. The **/RA_SYNC** signal is used to differentiate pixel data from message data. All message data is also accompanied by a **/RA_SYNC** true signal one clock after the **/RA_DVALID** signal whereas pixel data is sent without a sync pulse. **/RA_DIR** signal is used to echo the reset state of the DHE communications link to the PAN directly. This signal is set true after a reboot or reset and remains set until receipt of an asynchronous command from the PAN. **PIO1_IN** echoes the local state of the transmitter busy signal which goes true if the link to the PAN becomes congested or overruns the receive FIFO. **PIO2_IN** echoes the local Systran receiver error bit.

The second FPDP port is identical to the first with the exception that the J4B and J6B connectors are bussed to identical pins on the connectors J7 and J8 respectively. Physically the J7 and J8 connectors are under the footprint of the first FPDP CMC site. This allows the insertion of an auxiliary mezzanine board into the first FPDP footprint to intercept the data transmission both to and from the DHE. This site is provided to allow the inclusion of some computing power local to the DHE itself. This has the effect of eliminating the latency associated with a PAN running a non-real time operating system (i.e. Linux). Additionally, there are also 4 bits of data that can be written to the ***MCB_EXDATA*** register and are available to the P3A and P3B connector to pass state and/or status information to the mezzanine card. The auxiliary front panel connectors P1, P2, J9, J13, J14, J15, J18, and J19 are duplicated to allow alternate space on the front panel for the Systran module if this option is being used.

2.2.4 Pixel FPGA Logic



Top Level PIXEL FPGA Firmware Components
Figure 3

The Pixel FPGA logic is mainly designed to handle the volume of pixel data that occurs during detector readout. Toaccomplish this it is connected to the 48-bit pixel bus and to the FPDP RM port (Systran data input port). The pixel data bus is designed to synchronously transport 2 pixels of 24-bit data simultaneously across the bus to provide 80 Mpixels/sec throughput. The 24-bit pixel data retains the dynamic range required if non-normalized digital averaging is employed and allows a simple mapping for 18-bit resolution ADCs. It also allows for remapping to 3 x 16-bit pixels / clock cycle to give a 120 Mpixel/sec bus throughput if necessary.

There are nine signals used to strobe data into the Pixel FPGA Pixel Data Bus FIFO. The signal **/FPGA_CTL_1** is controlled by the Sequencer FPGA PAN command decoder and is used to capture data on the pixel bus that is valid as a result of a PAN read request to a DHE peripheral board. The seven signals **/PIPE_REQ_N(6:0)** are used by DHE peripheral acquisition boards to stream pixel data to the pixel FIFO. The pixel FIFO will accept data whenever any of these signals and **/PIPE_REQ_N[7]** is true on a rising edge of the system clock. The seven signals are priority rated by words written to each acquisition board that is using this facility so the **/PIPEREQ N** signal goes true in time priority order (Priority 0 goes true, then priority 1 four clock cycles later, and so forth). Once a board's priority status has been established, the board uses **/PIPE_REQ_N[7]** to control write strobes to the Pixel Bus FIFO. This method is called pipelined writes and forms the basis of extracting data from the different acquisition boards and transmitting these data to the PAN without intervention of the MPU sequencer.

In addition to pixel bus data, the Pixel FPGA supports a unidirectional 16-bit interface (FPGA Interface) from the Sequencer FPGA to receive the results of PAN commands and to locally decode and execute instructions specific to the Pixel FPGA logic (Pix FPGA Command Decode). The signals associated with the FPGA interface are **FPGAD0 => FPGAD15** and **FPGA_CTL3 => FPGA_CTL5**. This interface transports a 32-bit word in two transfers across the **FPGAD0 => FPGAD15** signals. **FPGA_CTL4** and **FPGA_CTL5** act as a requesting / acknowledge handshake pair while **FPGA_CTL3** transports the data sync bit in the first transfer and the local command strobe in the second transfer. Data received by the Pixel FPGA through this interface that is not tagged as a local command is sent to the FPDP port for transmission to the PAN. Data that is tagged as a local command (Function address range 0x0200 to 0x02FF) is decoded and executed.

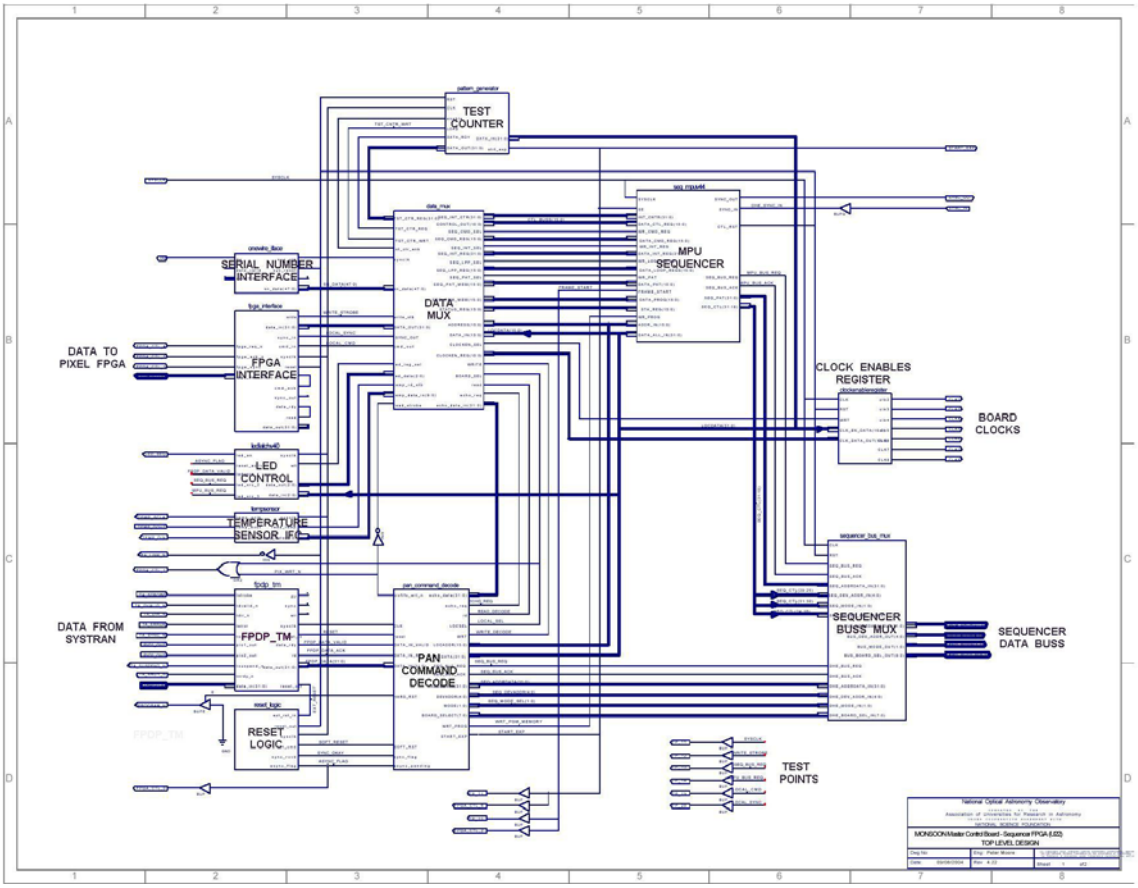
There are three data feeds to the FPDP RM interface.They are:

- Data originating from the SEQ FPGA and arriving via the FPGA Interface
- Data coming directly from the Pixel Data Buss via the Pixel Data Buss FIFO
- Data originating from the PIX FPGA itself as the result of a PAN command decoded locally.

These three streams are merged in a simple asynchronous data multiplexor that adjudicates requests from the three sources on a linear priority basis. Pixel data always has priority, followed by data coming from the SEQ FPGA, and finally local data.

Additionally, there are various local registers dedicated to specific hardware functions on the MCB board. Theseare simple, addressable data latches that control hardware external to the FPGA. The exception to this are the state machines developed to drive the serial configuration EEPROM interface (SERIAL CONFIG IFC) and the state machines used to control the temperature control DACs and ADCs.

2.2.5 Sequencer FPGA Logic



Top Level SEQUENCER FPGA Firmware Components
Figure 4

The SEQUENCER FPGA Firmware is primarily designed to handle the communications and control functions required of the MCB. Thirty-two-bit data from the PAN is captured at the FPDP-TM interface on the rising edge of signal **TA_STROBE** when **/TA_DVALID** is true and buffered in a 32-bit FIFO. The PAN COMMAND DECODE LOGIC interprets this data and performs a board level decode of the address. When signal **LOC_SELECT** is true, these commands are further decoded into local control signals and on-board address decoding takes place in the DATA MUX logic. Commands going to peripheral boards are translated into the Sequencer Bus protocol and presented to the SEQ BUSS MUX. Peripheral and local commands are sequenced and timed by the PAN COMMAND DECODE LOGIC directly.

Current firmware sets the MCB address range of 0x0200 to 0x02FF as being local command / address space for the PIX FPGA. When the signal **LOC_SELECT** is true and the address points to that segment, the original command is sent via the FPGA INTERFACE to the PIX FPGA for further decoding in that device. In this case, a distinction is made between message data and a command destined for the PIX FPGA by the signal **LOCAL_CMD** which is passed through the FPGA INTERFACE and interpreted in the PIX FPGA.

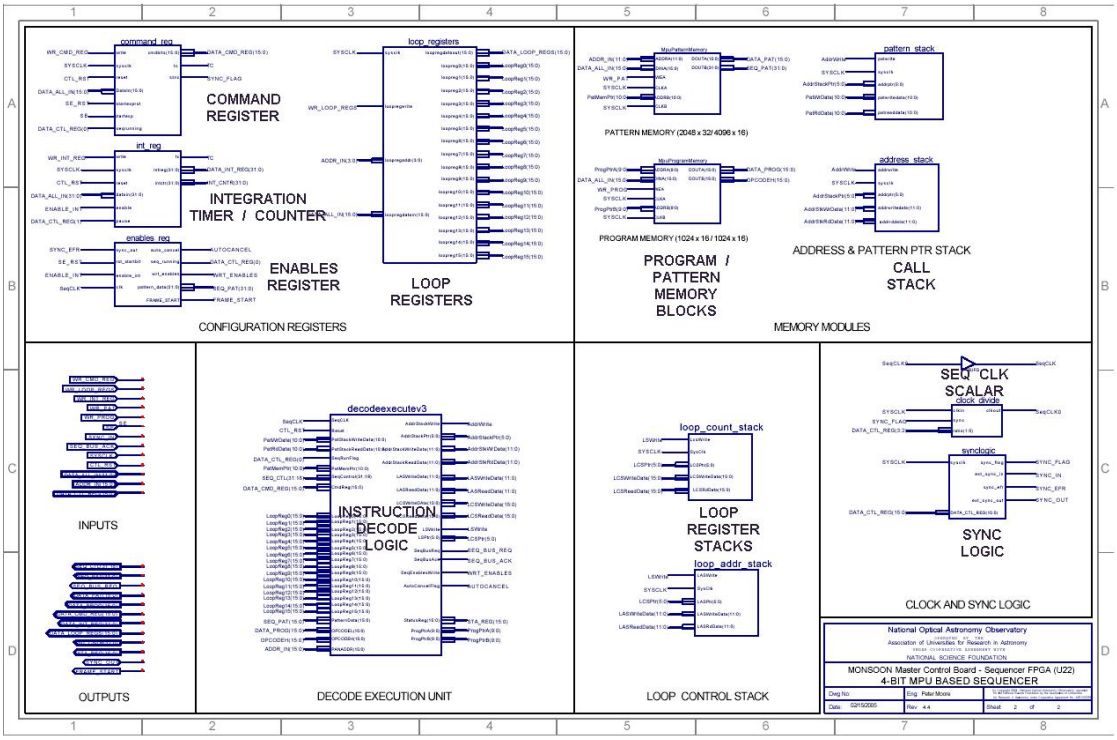
The SEQ BUSS MUX adjudicates on a last served, highest priority, synchronous manner to place the command onto the SEQUENCER BUSS. The alternative data source for this mux is the MPU SEQUENCER logic.

Board clocks for peripheral boards are generated by the CLOCK ENABLES REGISTER that takes the source 40 MHz. master clock and inverts it onto the relevant individual board clock line across the cPCi backplane after appropriate decoding for the backplane type.

NOTE: Since the cPCi backplane is developed for 66 MHz. clocks, the actual phase margin for data setup (originally 25 ns) when using an inverted clock phase is reduced to approx. 15ns.

The RESET LOGIC function provides for power on boot / reboot detection and reset synchronization between the PIX FPGA and the SEQ FPGA. This is achieved through signal **FPGA_CTL_0**. This logic also times the respective reset periods to provide a 15ms. reset period for a boot event and a 1µs. reset event for a soft reset. The 15ms reboot period is set to allow both the PIX FPGA and the SEQ FPGA time to complete the asynchronous boot process (The PIX FPGA boot is takes longer to ensure that it comes alive last) so that the reset release takes place synchronously across the two FPGAs.

2.2.6 MPU Sequencer Logic.



Top Level MPU Sequencer Firmware Components
Figure 5

Unlike more conventional sequencer operation, this sequencer is decoupled from the direct control of clock and control ports on the peripheral boards by the sequencer bus (Shared with the PAN COMMAND DECODE logic block. See Figure 4). While classic sequencers emit clock and control waveforms directly to a hardware port, this sequencer performs transactions on the sequencer bus that in turn control registers located on the individual peripheral boards in the DHE. In this way, the sequencer could be called a 'macro' sequencer since it essentially stores and sequences bus transactions that could be performed equally well (except for timing) from the PAN. The advantage of this approach is in the access to the complete DHE range of functions that can be performed by the sequencer. Currently however, with one exception (the EFR register), the MPU SEQUENCER cannot access registers within the address space of the MCB itself. It can only access those functions corresponding to address spaces on peripheral boards (DHE slots 2 to 8).

All functions of the MPU SEQUENCER logic are carried out synchronous with the MCB **SYSCLK** signal by way of a frequency divider that can be set accommodate the requirements of the detector (bits 3 to 5 of MCB control register -- **SEQ_CLKDIV**).

The INSTRUCTION DECODE LOGIC controls sequencer operation. This module controls the fetch of 4-bit fixed length instructions and operands from the PROGRAM MEMORY BLOCK, then decodes and executes them sequentially. From the PAN perspective, this appears as a 16-bit wide, 1024-word memory block. This memory is blocked from PAN read and write operations whenever the sequencer is enabled by way of the sequencer enable bit in the MCB control register (bit 0 -- **SEQ_ENABLE**). Instructions are fetched through a short pipeline process to assure single cycle execution. Sixteen instructions are available to build code sequences to control detector operations. These instructions are grouped into three categories:

- Control instructions that allow branching, looping and calls to subroutines
- Delay instructions to implement fixed unitary delays
- Output instructions that are used to control the sequencer buss.

For more detail on the MPU SEQUENCER instruction set and coding examples, please rfer to the Sequencer MPU Description document (MNSN-AD-0-000X).

Data that is to be used as operands in MPU SEQUENCER bus transactions are stored in a 32-bit wide memory block called the PATTERN MEMORY BLOCK. The PAN sees this memory area as little-endian, 16-bit wide 4094-word memory block. These data contain the actual values that are to be sequenced by way of the sequencer bus to various peripheral board registers to perform the detector control required. To accomplish this, there are four special registers to control the sequencer bus:

- The 2-bit sequencer bus MODE register which is loaded via the LMR instruction and controls the **SEQ_MODE[1:0]** bus signals.
- The 6-bit sequencer bus DEVICE ADDRESS register which is loaded by way of the LDA instruction and controls the **DEV_ADDR[5:0]** bus signals.
- The 8-bit sequencer bus BOARD SELECT register which is loaded by way of the LSR instruction and controls the **/SEL_1** TO **8** board select signals.
- The 11-bit pattern memory pointer that is loaded by way of the LPP instruction and points to an address in the pattern memory block where data is to be output onto the **SEQ_DATA[31:0]** signals.

Whenever the LSR instruction is executed, the INSTRUCTION DECODE LOGIC also emits a sequencer bus request via signal **MPU_BUSREQ** to the SEQUENCER BUS MUX that will be honored on the next clock cycle. At that time, the data plus control signals are established onto the sequencer bus and the transaction processed by the peripheral boards on the subsequent clock cycle.

Control of the sequencer is achieved through the specific control instructions of the MPU.

Unconditional and conditional jumps use the 16 bits of the sequencer command register (**SEQ_CMDS**); Bit 0 of this register is tied true to effect the unconditional jump condition by testing this bit.

Repeating code sections (looping) can be implemented using the LPB, LRB, and LPE instructions to test either one of the sixteen 16-bit loop registers or loop on the value of a pattern memory location. Loops can be nested to 64 levels.

Calls to subroutines are available to a depth of 64 calls. The program memory counter and pattern memory pointer are saved on the call and restored on the return instruction.

Internal functioning of the sequencer is controlled by a set of RS-style flip flops that correspond to configuration and events involving the sequencer itself. These flip flops are accessed by writing a bit pattern to the EFR register in 32-bit mode and with a board address of 1 (MCB).

2.2.7 JTAG Interface

The front panel JTAG connector (J10) provides access to the two boot EEPROM devices and the two FPGAs. The chain order is SEQ FPGA BOOT (18V03 EEPROM), SEQ FPGA (Virtex E 300), PIX FPGA BOOT (18V02), PIX FPGA (Virtex E 300). The front panel connector pin assignment is designed to mate on a one to one basis with most JTAG pod devices. These pin assignments are shown in Table 12.

Table 13- JTAG Pin Assignments

Pin	Function
1	+3.3VD
2	DGND
3	KEY
4	TCLK
5	N.C.
6	TDO
7	TDI
8	TMS

2.3 Options for Building the MCB

Currently there are no build time options for configuring the Master Control Board.

2.3 Jumper Descriptions

Table 14 - Configuration Jumper Descriptions

Jumper Number	Label name	Default Condition	Default Function
JP1	SER CFG WRT EN	2 to 3	Enable write functions to serial configuration store
JP2	INT/ EXT SYNC SRC	OPEN	Selects local SYNC Source (MASTER)
JP3	INT / EXT CLK SRC	1 to 2	Internal (local oscillator) clock source (MASTER)
JP4	FREQ RANGE SLCT	OPEN	MID frequency select (25 to 50 MHz operation)
JP5	3.3VD PWR SLCT	2 to 3	Selects EXT (backplane) +3.3VD power supply.
JP6	IGNORE FC	2 to 3	Do not ignore flow control (Systran)
JP7	CLK CONFIG[0]	2 to 3	Full clock speed option (Systran)
JP8	CONVERT SYNC	2 to 3	Do not convert sync frame (Systran)
JP9	ENABLE CRC	1 to 2	Enable CRC function (Systran)
JP10	CLK CONFIG[1]	2 to 3	Full clock speed option (Systran)
JP11	OSC ENABLE	OPEN	Tie pin 2 of this jumper to DGND to disable the master clock. Pin 1 is incorrectly wired to +3.3VD.

3.0 Board Specifications

Table 15 – Specifications

Power Requirements	+5.0V digital @ 0.5 amp average, 1.0 Amp peak (200ms) +3.3V digital @ 0.3 amp average, 0.9 amp peak (200ms) These values do not include power required by the Systran interface, which is: SL100 -- +3.3V digital @ 1.2 amp average, 1.4 amp peak. SL240 -- +3.3V digital @ 1.4 amp average, 1.7 amp peak.
Power Consumption	7.5 Watts
PAN Command Execution Time	120ns
Sequence Type and Memory Depth	Application-specific MPU in FPGA, 4K code store, 1.5K pattern store
Sequencer Clock Resolution	50ns
Integration Timer Resolution and Capacity	Configurable 100ns, 1µs, 1ms - 32-bit count up register
Pixel Data Rate	Maximum 50 Mpixel/se with SLM100, 80 Mpixel/sec with SLM 240
Diagnostic Channels	Temperature, serial number, synthetic pixel generator, firmware revision
Auxiliary Functions	Master/Slave DHE sync logic, HS Serial link, 8 x temperature monitor
Physical Specifications	Bare board: Height - 233 mm (6U) Depth - 160 mm Thickness - 1.6 mm Fully populated: Height - 233 mm (6U) Depth - 208 mm Thickness - 20 mm
Mating Backplane	Per MNSN-AD-01-0006 (Linked Here)

4.0 Appendix

4.1 Appendix I - Board Identity and Firmware Revision Codes.

THE BOARD IDENTIFIER IS CODED AS A 16-BIT UNSIGNED INTEGER (I.E. CODES 0 TO 65535 ARE AVAILABLE).

WHEN READ AS AN ATTRIBUTE, THE INTEGER DECIMAL VALUE OF THE ENCODED VALUE SHOULD REPRESENT THE PRODUCT CODE OF THE PHYSICAL HARDWARE.

CURRENT NOAO IMPLEMENTATIONS OF THIS CODE ARE RESERVED AND SHOWN BELOW:

- 100 = 36 CHANNEL IR ACQUISITION MOTHER BOARD REVISION A
- 200 = 18 CHANNEL IR ACQUISITION DAUGHTER BOARD REVISION A
- 300 = CLOCK AND BIAS BOARD REVISION A
- 400 = MASTER CONTROL BOARD REVISION A
- 500 = CCD ACQUISITION BOARD - PROTOTYPE

THE FIRMWARE REVISION IDENTIFIER CONTAINS A UNIQUE PATTERN TO IDENTIFY THE FPGA CODE VERSION CURRENTLY LOADED ONTO THE HARDWARE.

CURRENT IMPLEMENTATION USES THE SAME 16-BIT UNSIGNED INTEGER CODING SCHEME TO DESCRIBE A THREE DIGIT MAJOR + TWO DIGIT MINOR VERSION SET.

WHEN READ AS AN ATTRIBUTE, THE DECIMAL EQUIVELENT OF THE ENCODED VALUE SHOULD BE DIVIDED BY 100 TO SEPARATE THE MAJOR AND MINOR REVISION NUMBERS.

TWO SPECIAL CASES ARE RESERVED FOR CODE DEVELOPMENT AND TESTING.

VERSION 000.XX IS CONSIDERED UNDER DEVELOPMENT OR UNASSIGNED.

VERSION 9XX.XX IS CONSIDERED AS DEBUG CODE WHERE THE TWO REMAINING MAJOR AND TWO MINOR CODES REPRESENT THE SOURCE VERSION FOR THIS DEBUG CODE.

IT IS SUGGESTED THAT MAJOR CODES ARE RESERVED FOR CHANGES TO FUNCTIONS AND PROTOCOLS THAT AFFECT THE INTEROPERABILITY OF BOARDS WITHIN A DHE.

MINOR CODES ARE USED TO DIFFERENTIATE ON-BOARD FUNCTIONALITY CHANGES THAT DO NOT AFFECT OTHER BOARDS WITHIN A DHE.

4.2 Appendix II - Using the MONSOON SYNC Function

4.2.1 Introduction

The use of the synchronized logic contained in the hardware / firmware of the MONSOON image acquisition system allows multiple Detector Head Electronics (DHE) chassis to operate in close proximity to each other. Multiple DHE systems are typically required for large focal planes where the number of video channels exceeds the capabilities of a single DHE. Without synchronization, each DHE will generate electrical fields that can potentially interfere with an adjacent DHE chassis leading to an increase in system noise.

To obtain synchronization two criteria must be met:

- Each DHE needs to have an internal system clock signal that is locked in phased to a master clock.
- b). Any activity that is sensitive to induced noise, such as reading out of a detector, must be 'lock stepped' to ensure that interference sources are seen as common mode noise sources by all DHE chassis.

4.2.2 MONSOON Implementation

The requirements for synchronous operation are met in MONSOON by the software configuration of several PAN attributes, two front panel connectors on the Master Control Board (MCB), and the use of a synchronization bit in the sequencer code running on the MCB.

The software attributes control the designation of each DHE as a master or slave and establish an appropriate delay that allows each DHE enough time to have received the 'Start Exposure' command from its local PAN. In addition, software attributes control the system clock phase control (by way of U19) to align the different DHE clock phases. These attributes are described further in Table 1.

The connectors are designated as "Sync In" (J9) and "Sync Out" (J18). The normal topology for the Master / Slave cabling is a daisy chain where a cable is plugged into the designated Master DHE 'Sync Out' connector and the first Slave DHE 'Sync In' connector. Successive Slave DHE chassis are then linked in the same way. Figure 1 shows this scheme. The requirement for the synchronization signals is that the combined delay associated to the cable length and line receivers / transmitters will not exceed 50ns total. If this condition is not met by the simple passive daisy chain topology, a 'Star' topology can be implemented using a powered repeater.

Routing of the hardware signals in each MCB is achieved through jumpers JP2, JP3, and JP11. Table 2 describes the appropriate jumper positions for each mode. Figure 2 shows a schematic of the signal routing on the MCB.

There is a dedicated bit in the testable Sequencer control register that acts as the synchronization flag for sequencer code running the readout and other sequencer programs. This bit (bit 1 of the SEQ_CMDS register at MCB address 0x0102) is controllable through the EFR register of the sequencer when the mode is set Master. Delaying sequencer program flow until this bit changes state allows both the master and slave DHE sequencers to initiate a program in 'lock step' with all other DHEs. Paragraph 4.2.4 details the basic mechanism of this.

4.2.3 Limitations of Revision 'A' MCB boards.

The connectors and cable generally used to connect Masters to Slaves is a standard FireWire connector. Unfortunately, these cables come with TX / RX pair crossovers as standard. This results in the clock signal pair from the Master appearing as the Sync signal pair in the Slave. If non-crossover cables can be found, these can be purchased and used off the shelf. If non-crossover cables cannot be found, you have the option of either constructing custom cables using standard FireWire connectors or hardware strapping the MCB to cross over the signals between the jumpers JP2 and JP3.

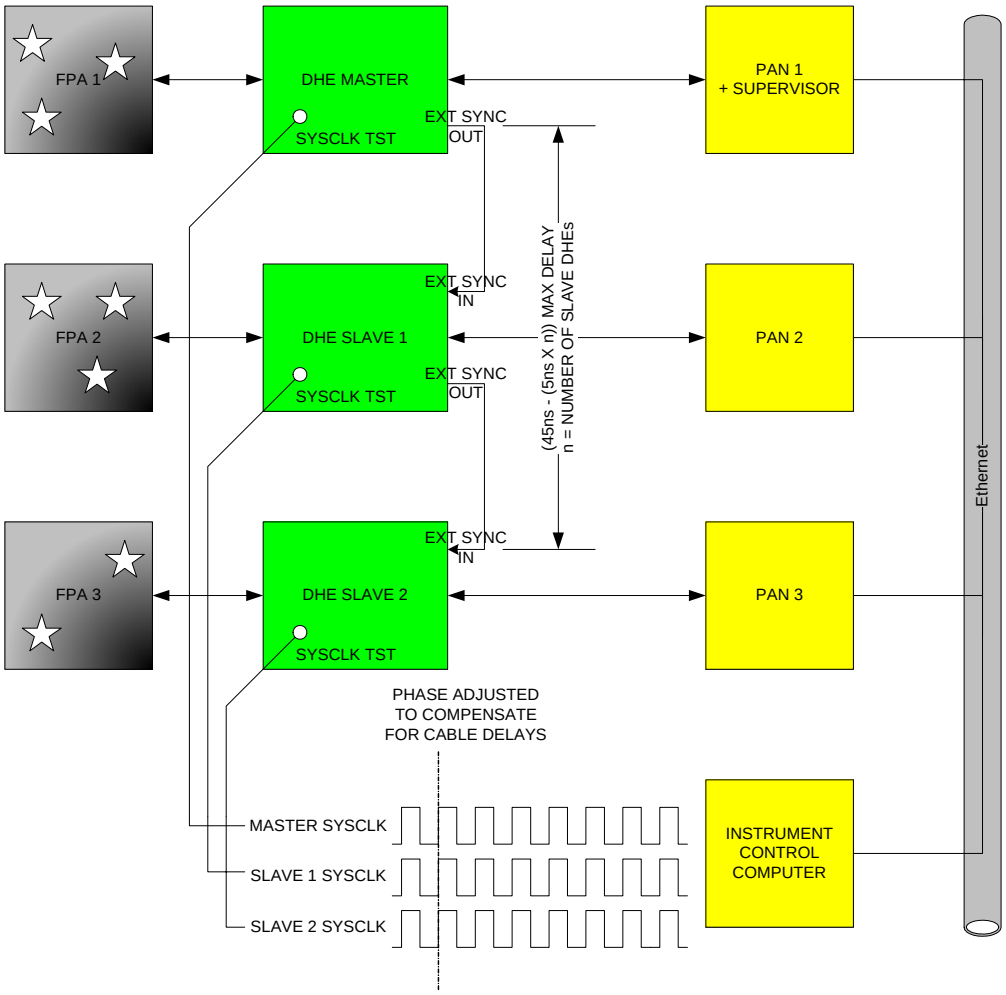
JP11 pin 1 is strapped to +3.3VD in error. This pin should have been routed to DGND. To fully disable the internal MCB oscillator, it is necessary to take JP11 pin 2 to DGND via a convenient wire jumper.

Table 16 - Software Attributes Used for Sync Function

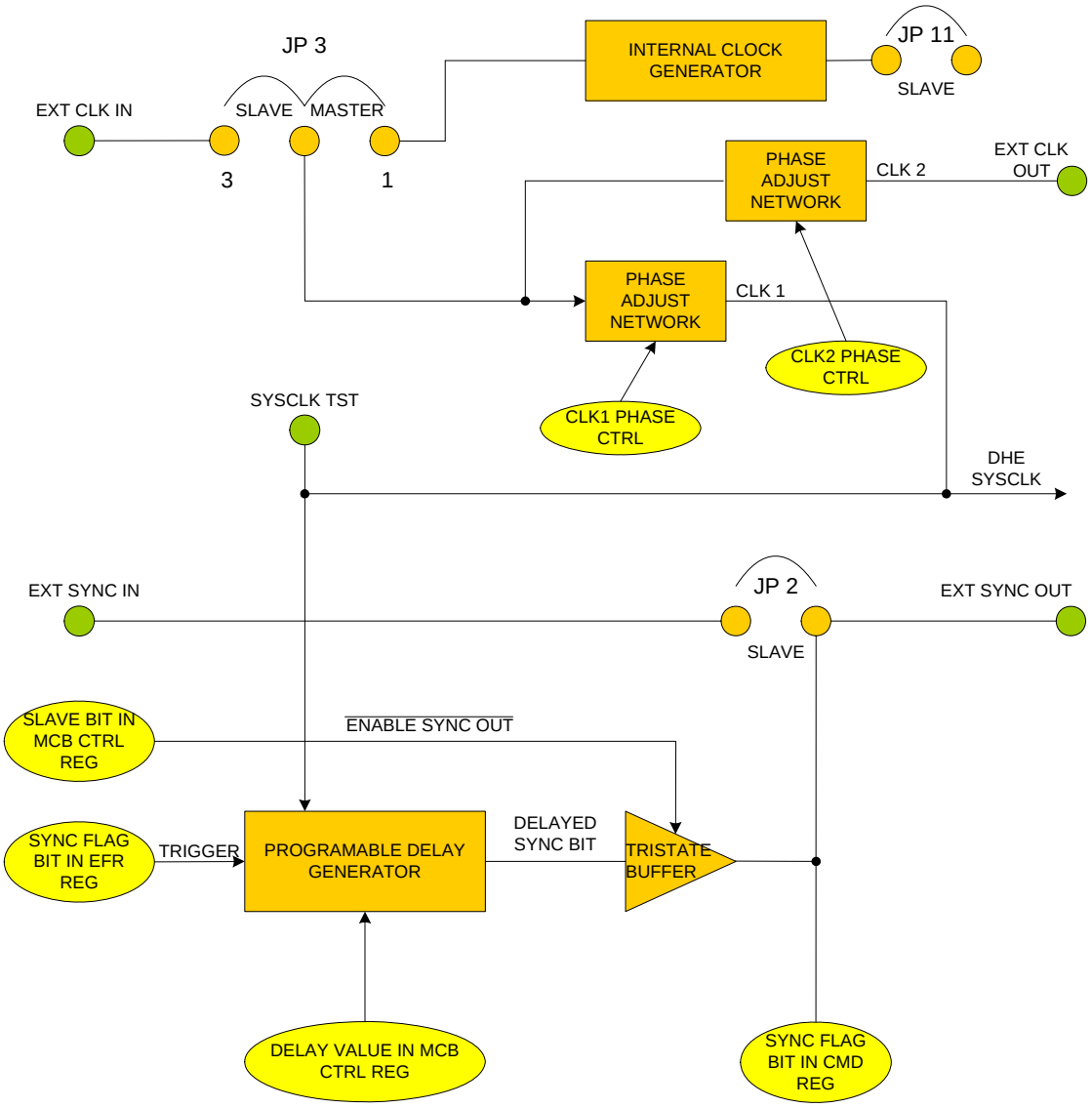
Attribute Name	Function Name	MCB Address	Description
McbClkInCtl	MCB_CLKINCTL	0x0281[7:4]	Provides ± 4ns adjustment of clock source to internal System clock
McbClkOutCtl	MCB_CLKOUTCTL	0x0281[3:0]	Provides ± 4ns adjustment of clock source to external System clock
DheMstr	DHE_MASTER	0xFFFC[4]	When set true, configures DHE as Master.
MstrSyncDly	SEQ_SYNCDELAY	0xFFFC[15:8]	0 to 128 millisecond delay time between when the EFR sync flag is set low by the Master Sequencer code until the hardware sync signal goes low. Setting the EFR register sync flag true sets the hardware signal high immediately.

Table 17 - Hardware Jumper Positions for Each Mode

Jumper	Master Mode	Slave Mode	Function
JP2	OPEN	SHORT	Selects the onboard clock oscillator or the external clock source
JP3	1:2	2:3	Enables the external Sync signal to the sequencer control register
JP11	OPEN	SHORT	Disables the MCB Clock Oscillator. – See limitation note above.



DHE Daisy Chain Synchronization Scheme
Figure 6



MCB System Clock and Sync Bit Routing
Figure 7

4.2.4 Sample Assembler Code for Synchronized DHE Function.

```
*******************************************************
** PROJECT    : MONSOON ENGINEERING
** TITLE      : EXEC PORTION OF TEST ALADDIN CODE MODIFIED TO ACHIEVE SYNCRONIZATION BETWEEN
**            : TWO OR MORE DHE MODULES.
** VERSION    : 002
** DATE       : 08/30/2006
** AUTHOR     : PETER MOORE
** HISTORY    : 12/09/2003 PCM ORIGINAL VERSION SCULPTED FROM EXEC ON ALADDIN.
**            : 08/30/2006 PCM ADOPTED CODE TO REFLECT CCD USE OF ASM4. UPDATED JUMPER
**            : DESIGNATIONS FOR MCB REVISION A HARDWARE.
**
** DESCRIPTION :
**          : DUMMY SEQUENCER EXEC PORTION TO EXERCISE AND DEMONSTRATE THE SYNC
**          : FUNCTIONALITY.
**          :
**          : NORMAL READOUT CODE IS REPLACED BY A DUMMY STUB THAT JUST RETURNS.
**          : FOR ACTUAL USE YOU WOULD INSERT YOUR CODE AT THE STUBS
**          :
**          : TO USE THIS CODE THE MASTER AND SLAVE DHES NEED TO BE LINKED WITH THE SYNC
**          : CABLE. MASTER SYNC OUT => SLAVE SYNC IN. THE SLAVE DHE SHOULD BE SET TO
**          : USE THE EXTERNAL CLOCK (JP3 2=>3) AND ADJUSTED IN PHASE (VIA THE MCB_CLKINCTL
**          : AND MCB_CLKOUTCTL FUNCTIONS) TO BE COINCIDENT WITH THE MASTER SYSTEM CLOCK.
**          : THE SYNC CABLE CONNECTING THE MASTER AND SLAVE DHES MUST HAVE LESS THAN
**          : 45ns OF DELAY IN ITS ENTIRE LENGTH (approximately 7 meters physical length).
**          :
**          : JP2 NEEDS TO BE JUMPERED FOR EXTERNAL SYNC IN THE SLAVE DHE AND REMOVED
**          : IN THE MASTER DHE.
**          :
**          : THE PAN SETS THE MASTER AND SLAVE MODE IN THE RELEVENT DHES VIA
**          : THE MCB CONTROL REGISTER BIT 15. BIT SET INDICATES MASTER MODE.
**          :
**          : THE PAN THEN ISSUES A NORMAL "START EXPOSURE COMMAND WITH A VECTOR (1 IN THIS
**          : EXAMPLE). ALL SEQUENCERS JUMP OUT OF THEIR IDLE MODE AND AWAIT THE ARRIVAL OF SYNC.
**          : SYNC IS DELAYED BY A FIRMWARE DELAY GENERATOR BY A VALUE THAT EXCEEDS THE
**          : ESTIMATED MAXIMUM LATENCY OF THE START EXPOSURE COMMAND REACHING EACH
**          : INDIVIDUAL PAN NODE. THE SYNC IS EMMITED BY THE MASTER TO ALL SLAVES AFTER THE
**          : DELAY. THE MASTER RECEIVES ITS OWN SYNC SIGNAL IN THE SAME FASHION (AND AT THE SAME
**          : TIME) AS THE SLAVES. WHEN SYNC IS RECEIVED AND RECOGNIZED BY THE SEQUENCER BRANCH
**          : INSTRUCTION, ALL SEQUENCERS RUN THE SAME CODE ON THE SAME CLOCK EDGE.
**          :
**
** RESOURCES :
**
*****************************************************************
** JUMP FLAG DEFINITIONS - BIT POSITION IN COMMAND REGISTER **
*****************************************************************
DEF #SYNC                1          * Jump on sync bit high
DEF #ITC                 2          * Jump on Integration time expired
DEF #START               3          * Jump on start exposure flag set
DEF #CCDREADOUT          4          * STRVCTR =  1: DO COMPLETE CCD CLEAR - INTEGRATE - READ
DEF #CCDCLEAR            5          * STRVCTR =  2: DO CCD CLEAR ONLY
DEF #CCDREAD             6          * STRVCTR =  4: DO CCD READ ONLY
DEF #VCTR_4              7          * STRVCTR =  8:
DEF #VCTR_5              8          * STRVCTR = 16:
DEF #VCTR_6              9          * STRVCTR = 32:
DEF #VCTR_7              10           * STRVCTR = 64:
DEF #VCTR_8              11           * STRVCTR =128:
DEF #CONT_RUN            12           * USER BIT 1: LOOP INDEFINATELY ON TESTS
DEF #USR_2               13           * Jump on user mode bit value 2
DEF #USR_3               14           * Jump on user mode bit value 4
DEF #CONT_CLR            15           * Jump on user mode bit value 8
**
```

```
****************************
** EFR REGISTER CONSTANTS **
****************************
DEF #KILL_EXPFLG          x00000001     * Cancel start exposure flag using the EFR register
DEF #STRT_INTCNTR         x00000002     * Start the integration time counter using the EFR register
DEF #STOP_INTCNTR         x00000004     * Stop the integration time counter using the EFR register
DEF #LSR_AUTO_KILL        x00000100     * Enables automatic board select cancel after an LSR instruction
DEF #LSR_MAN_RESET        x00000200     * Disables automatic board select cancel. LSR will remain active
                                         *until an LSR #DSLCT is issued
DEF #SET_SYNC_LOW         x00004000     * If enabled as master DHE, set sync bit low after sync delay
DEF #SET_SYNC_HIGH        x00008000     * If enabled as master DHE, set sync signal high immediately
DEF #KILLANDSYNC          x00004001     * Set sync low and kill start exposure flag
**
****************************
** MODE REGISTER CONSTANTS **
****************************
DEF #RESET                0             * Mode register constant for board reset operation
DEF #READ                 1             * Mode register constant for read operation
DEF #WRITE16              2             * Mode register constant for 16 bit write operation
DEF #WRITE32              3             * Mode register constant for 32 bit write operation
**
**********************************
** LSR BOARD SELECT ASSIGNMENTS **
**********************************
DEF #DSLCT                0
DEF #MCB                  1
*******************************
** PATTERN MEMORY DEFINITIONS **
*******************************
** PATTERN MEMORY SEGMENTATION MAP
DEF #PATSEG_00            x0000          * Pattern memory origin - Initialization stuff
**
**  ADDITIONAL DEFINITIONS USUALLY GO HERE
**
**
****************************************
** INSTANTIATE PATTERN MEMORY VALUES **
****************************************
ORG PAT #PATSEG_00              * ********* GENERAL CONSTANTS KEPT IN PAT MEM *********
PAT #KILL_EXPFLG          #KILL_EXPFLG     * We can use the same label name here because
                                           *they are in different contexts
PAT #STRT_INTCNTR         #STRT_INTCNTR    * The definition earlier points to a data value. The label here points to
PAT #STOP_INTCNTR         #STOP_INTCNTR    * a pattern memory address
PAT #SET_SYNC_LOW         #SET_SYNC_LOW    * CODE VALUE TO WRITE TO EFR TO SET SYNC PIN LOW
PAT #SET_SYNC_HIGH        #SET_SYNC_HIGH   * CODE VALUE TO WRITE TO EFR TO SET SYNC PIN HIGH


*************************************
* PROGRAM CODE SEGMENT STARTS HERE **
*************************************
** EXECUTIVE ROUTINE

#INIT     CAL #INIT                 * SET DETECTOR UP FOR INITIAL CONDITIONS
          LMR #WRITE32
          LPP #SET_SYNC_HIGH
          LSR #MCB                  * SET SYNC BIT HIGH AS INITIAL CONDITION
#EXEC     JCB #GOBABY #START        * IF START EXPOSURE FLAG SET
          JMP #EXEC                  * ELSE JUMP BACK TO EXEC

#GOBABY LPP #KILL_EXPFLG            * KILL THE START EXP FLAG PATTERN = 0x00000001
          LSR #MCB                  * WRITE TO THE EFR
          LPP #SET_SYNC_LOW
          LSR #MCB                  * TRIGGER THE SYNC SIGNAL. NOTHING HAPPENS HERE
                                    * UNTIL THE DELAY HAS EXPIRED.
```

```
#WAIT    JCB #WAIT #SYNC            * SYNC GOES FALSE SYNCHRONOUSLY ON ALL DHES AFTER DELAY
         LPP #SET_SYNC_HIGH
         LSR #MCB                   * RESET SYNC BIT HIGH
         JCB #JRDOUT #CCDREADOUT        * DO CLEAR - INTEGRATE - READOUT OPERATION
         JCB #JCLEAR #CCDCLEAR          * DO CLEAR ONLY
         JCB #JREAD #CCDREAD            * DO READ ONLY
         JMP #EXEC                      * NO FLAGS SET, GO HOME.

#JRDOUT CAL #CLEAR
         LPP #STRT_INTCNTR
         LSR #MCB                   * START INTEGRATION TIMER RUNNING
         NOP                        * A bug in the MPU .. needs time to reset the flag
         NOP
#RD1     JCB #RD2 #ITC               * WAIT FOR INTEGRATION TIME
         JMP #RD1
#RD2     LPP #STOP_INTCNTR
         LSR #MCB                   * STOP INTEGRATION TIMER
         CAL #ARRAY_READ
         JCB #JRDOUT #CONT_RUN * DO IT ALL AGAIN IF TOLD TO DO SO
         JMP #EXEC                  * CALL IT A DAY

#JCLEARCAL #CLEAR
         JMP #EXEC

#JREAD   CAL #READ
         JMP #EXEC
```

4.3 Appendix III – Master Control Board Sequencer MPU Description

4.3.1 Introduction

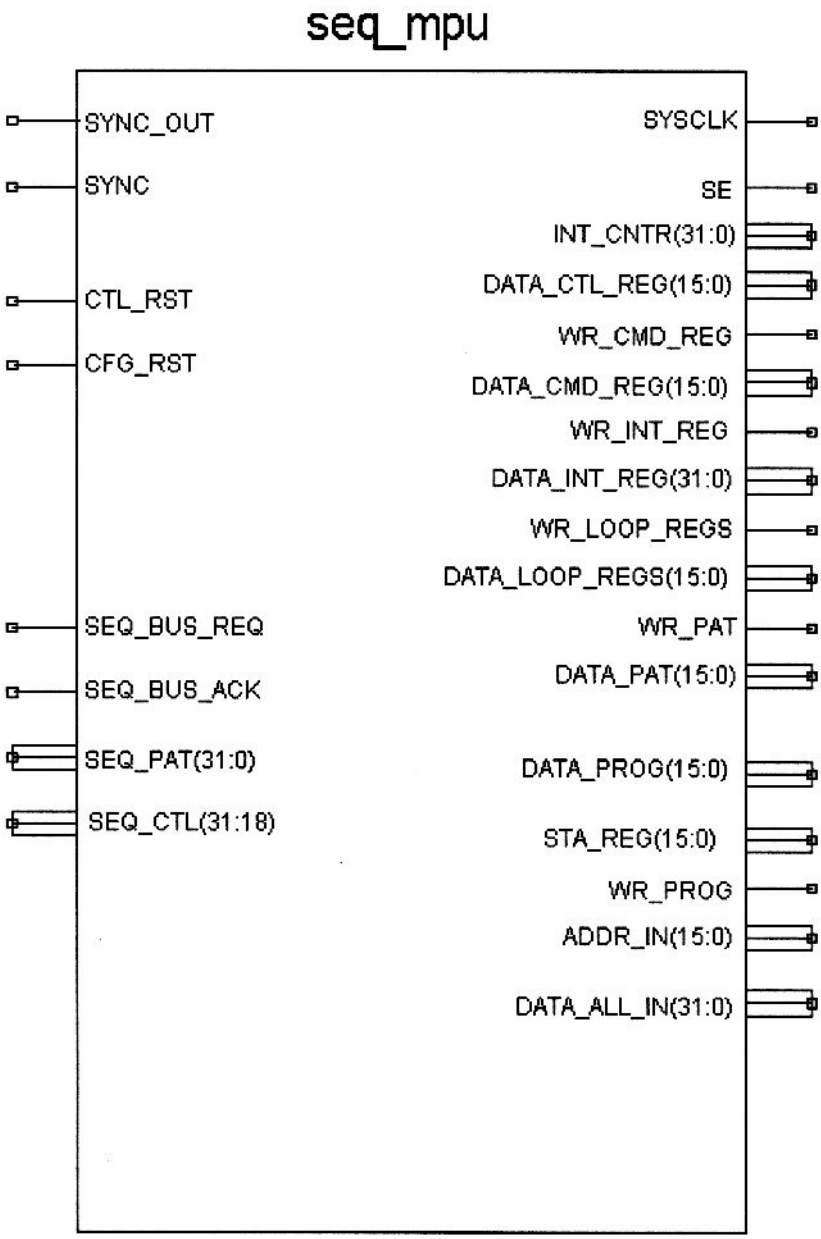
The sequencer for the Master Control Board (MCB) is implemented as a completely imbedded 4-bit (16 instructions) micro-controller. A micro-code program loaded into the sequencer program memory controls the logic flow of the sequence. A fixed instruction decode and execution cycle is of great value, particularly for timing critical waveform generation. Therefore, the sequencer is designed with a uniform one-clock-cycle execution time for all 16 instructions. The assembly micro-code for the sequencer can be easily customized to change functionality and adjust the required timing.

4.3.2 Sequencer Performance

The current sequencer design implementation performs satisfactorily at 20 MIPS on the Virtex XCV 300 PQ 240 and meets all worst-case timing requirements including clock skew analysis with a positive slack of more than 1 ns. The current performance of 20 MIPS is very modest but meets the objective of single-clock-cycle execution without any branching latency penalties. It should be noted that the sequencer currently operates at the maximum frequency of 20 MHz = ½ system clock frequency of 40 MHz.

4.3.3 Sequencer Interface

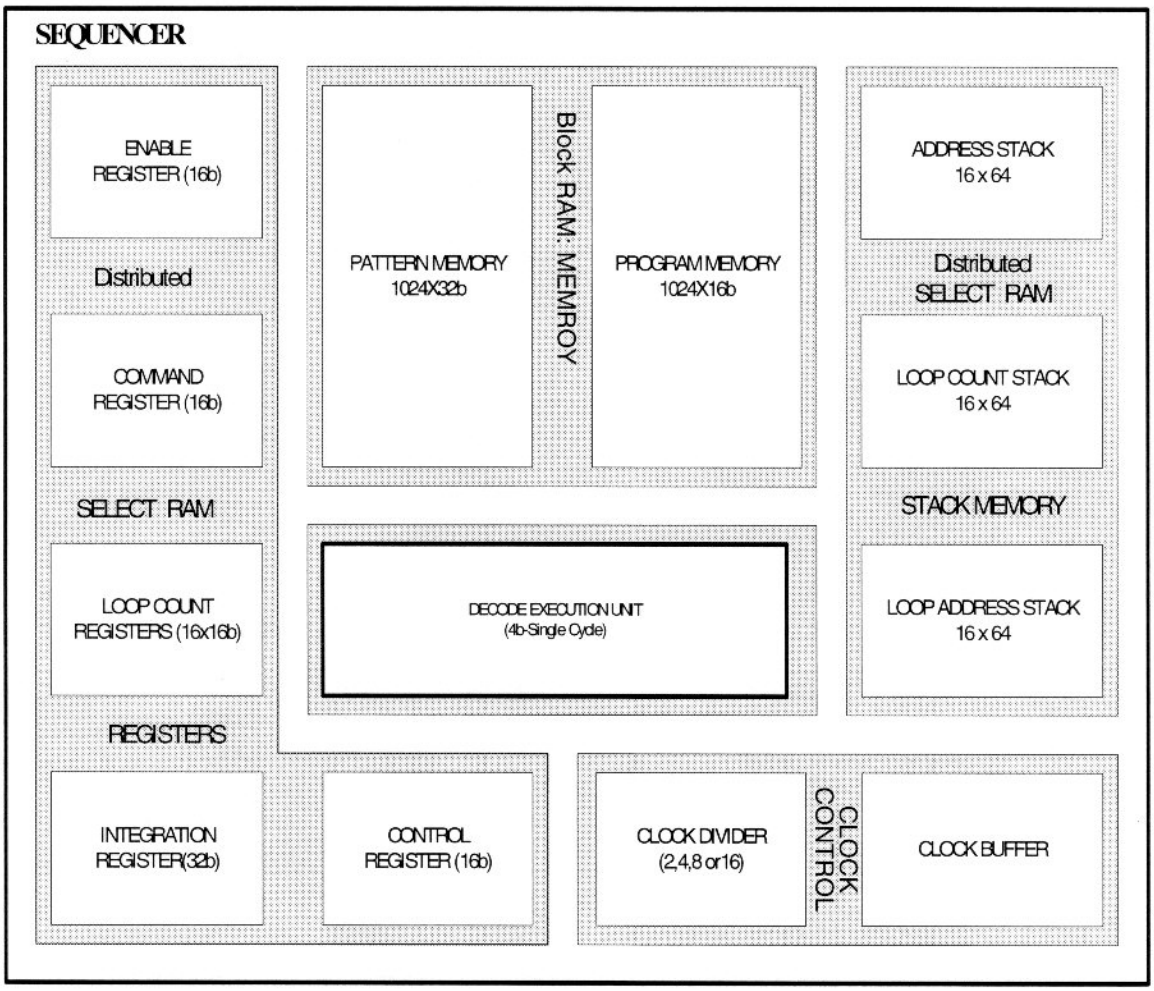
The sequencer interface symbol is shown in Figure 1. The sequencer derives its clock internally from the ***sysclk***. Only the decode execution unit of the sequencer operates on a ***seqclk*** derived from the global system clock ***sysclk***. All internal registers and memory elements except for the stacks are accessible through a common data write port and individual read port which is multiplexed external to the sequencer. The address decode of all components of the sequencer is also implemented external to the sequencer and individual input write strobes are asserted when the corresponding components are addressed for writes. The data in all sequencer registers based on Select-RAM is always available for read during the same clock cycle in which they are addressed. However, the Block RAM-based program memory and pattern memory elements incur one-clock-cycle latency before the data corresponding to their address is available on their output ports. The sequencer accesses the Sequencer Bus by asserting a bus request signal and waiting for an acknowledge signal. This wait blocks the sequencer execution and is critical for timing sensitive delay instructions. However, if the acknowledge signal is asserted before the next sequencer rising edge, no additional delay is incurred.



Sequencer Interface Symbol
Figure 8

4.3.4 Architecture Overview

The sequencer is based on Harvard architecture with separate program memory and pattern memory implemented using the Block RAM resources. Apart from the conventional branching instructions, dedicated looping instructions are also included in the sequencer instruction set, therefore the sequencer also includes an address stack, loop address stack and loop count stack memories to support the nested looping and branching functionality. Five types of special function registers, Enable, Command, Loop Count, Integration and Control, are also included for their specific functions. A central Decode Execution Unit (DEU) processes the sequencer micro-code to implement the programmed control flow and timing. A clock control unit, which includes a sequencer clock divider and clock buffer, is included for programmable clock cycle and low skew operations. The detailed functionality and implementation of each of these sequencer components is described later in this section.



Sequencer Components
Figure 9

4.3.5 Program Memory (BRAM)

The program memory is implemented using the Block RAM resources. The program memory is dual port, 16 bits wide and 1K (1024 words deep, read-write RAM). Both ports are clocked with the same system clock (***SYSCLK 40 MHZ***). Conventional pipeline-based architecture results in pipeline refill latencies while executing branching instructions that are unacceptable for single clock execution. To avoid such latencies, one port of the program memory is dedicated to substitute for pipelining operation. This is achieved by performing a *Program counter + 1* read at the second port of the program memory. Access to this port is limited to the DEU only. Access to the other port of the program memory is time multiplexed between the PAN and the sequencer DEU, where the PAN is given the higher priority. The result is that the DEU's access to program memory is suspended when the PAN is reading or writing to the sequencer memory. This inserts a blocking delay in sequencer execution so the timing information encoded into the sequencer micro-code gets altered. Because of this, program memory read and writes should be avoided during timing-critical normal mode of sequencer operation.

4.3.6 Pattern Memory (BRAM)

The pattern memory is dual port 32K bits of read-write RAM. The two ports of pattern memory include independent read-write PAN port and read only DEU port but differing in width and depth. From the PAN port the program memory is 16 bits wide and 2K words deep whereas from the DEU port it is 32 bits wide and 1K patterns deep. The even and odd addressed 16-bit words correspond to the least and most significant 16 bits of a pattern respectively. The dual port access to the pattern memory is not time multiplexed so both PAN and DEU can randomly access the program memory independently. The pattern memory is clocked using the same system clock (***SYSCLK 40 MHZ***). The pattern memory is implemented using the Block RAM resources. It should be noted that patterns in the program memory are not cleared on reset and require explicit writes (with 0 data) from the PAN to clear the contents of the Pattern Memory.

4.3.7 Stack RAMs (Select-RAM)

Program Address Stack RAM and Loop begin address Stack RAM provides 12 bits wide and 64 words deep buffer spaces each to implement the Program Pointer Stack and Loop Begin Address Stack respectively, whereas the loop count Stack RAM is 16 bit wide and also 64 words deep. The Program Address is pushed into the Program Address Stack RAM whenever a sub-routine is called and is popped back at the end of sub-routine execution. Similarly, to support the nested looping instructions at the beginning of the loop, the next program memory address and previous contents of the loop counter are pushed into their respective stacks, whereas the loop counter is loaded with the new count value. After execution of iteration, if the loop counter is greater than zero, it is decremented and program pointer is re-loaded with the loop beginning address stored at the top of the stack. Otherwise the program pointer is incremented to point to the next instruction and loop count and loop starting address are both popped out from the sequencer stack. All three Stack RAMs are clocked from the system clock (***SYSCLK 40 MHZ***). The Stack RAMs are not transparent to the PAN and are controlled by the DEU only. Stack RAMs are implemented using Distributed Select RAM resources.

4.3.8 Special Function Registers

As mentioned earlier, the current implementation of the sequencer includes the following five different types of special function registers:

- Command Register
- Control Register
- Loop Count Register
- Clock Control
- Decode Execution Unit

4.3.8.1 Command Register Definition

This register is a combination of hardwired and software settable flags. The intended purpose is to provide a conditional jump capability within the sequencer micro code by using the sequencer instruction “Jump if Control Bit Set” (JCB). This instruction allows any of the 16 flags to be tested for an active condition.

4.3.8.2 Bit 0: Unconditional Jump.

Always tied to '1' to facilitate the implementation of unconditional jump.
Default 1 Active 1.

4.3.8.3 Bit 1: SYNC_IN

This bit reflects the condition of the ***EXT_SYNC*** signal when MCB jumper JP17 is correctly configured for slave mode. In Master mode this bit reflects the status of the ***SYNC_OUT RS*** flip flop controlled by the Enable Function Register (EFR).
Default 0 Active 1.

4.3.8.4 Bit 2: Integration Timer Terminal Count

This bit is set when the Integration Time counter (ITC) reaches the value stored in the Required Integration Time (RIT) register (at MCB address 0x0000). The flag will remain set until either the RIT register is loaded with a larger value than the ITC contents or asserting the enable bit via the EFR activates the integration timer. Default condition (after reset) is active since the ITC and the RIT are reset to zero. If this bit is to be used to determine if the integration timer is active, the PAUSE bit in the control register should also be checked.
Default 1 Active 1.

4.3.8.5 Bit 3: Start Exposure Active High.

This bit is set when the DHE receives a “Start Exposure” command from the PAN. It remains set until a reset start exposure flag bit is set via the EFR. The purpose of this bit is to act as a qualifier for bits 4 to 11 of this register.
Default 0 Active 1.

4.3.8.6 Bit 4 to Bit 11: Start Exposure vector.

These bits are latched from the address portion of the start exposure command at the same time as bit 3. They are meant to enable a variety of sequence actions to be selected based upon the bits set and the use of the JCB instruction in the sequencer micro code.
Default 0 Active 1.

4.3.8.7 Bit 12 to Bit 15: User Defined Bit Pattern.

These bits are written directly to the command register from the PAN by writing to the register address. They are transposed from bits 0 to 3 in the data written to the register.
Default 0 Active 1.

4.3.9 Control Register Definition

This register controls the general behavior of the Master Control Board. The principal functions are listed below.

4.3.9.1 Bit 0: MPU Run:

Asserted by writing to the MCB at address 0xFFFC. Asserting this bit allows the sequencer to begin decoding and executing micro code. Whenever the signal is de-asserted, the MPU enters into the Hold State after executing the current instruction. All the board selects and the request signal are masked during the non-active hold state. The program counter is reset to zero.
Default 0 Active 1.

4.3.9.2 Bit 1: PAUSE ITC:

Setting this bit to 1 will stop the Integration Time Counter (ITC) from incrementing. Resetting this bit will allow the ITC to continue counting up.
Default 0 Active 1.

4.3.9.3 Bit 3 & bit 2: Sequencer Clock Pre-Scalar

This 2-bit pattern defines the base sequencer clock frequency. Note that the current maximum frequency is half the system clock frequency. Each micro code instruction takes one sequencer clock cycle to complete.

- 00 SYSCLK / 1 20 MHz
- 01 SYSCLK / 2 10 MHz
- 10 SYSCLK / 4 5 MHz
- 11 SYSCLK / 8 2.5 MHz

Default 00.

4.3.9.4 Bits 4-14: Currently Not Used

4.3.9.5 Bits 15: Slave DHE.

When set this bit disables the output of the ***SYNC_OUT*** signal and enables the reception of ***SYNC_IN*** signals to the CMD register. When this bit is not set the SYNC bit in the CMD register reflects the state of the ***SYNC RS*** Flip Flop controlled through the EFR.
Default 0 Active 1.

4.3.10 Loop Count Registers

There are 16 Loop Counters. These 16-bit registers can be loaded directly from the PAN and are intended for use where repetitive cycles of clock sequences are needed. The sequencer instruction "Loop Register Begin" (LRB) identifies the first instruction to be repeated and one of the 16 loop registers. The "Loop End" (LPE) instruction identifies the end of the repeated block of micro code and will repeat the sequence until the count stored in the loop register has been completed.

4.3.10.1 Enable Function Register (EFR)

The EFR is accessible only through the micro code. This register controls strobes that are set and reset by executing a "Load Select Register" (LSR) instruction with the bit 0 set in the operand and the MODE Register set to a 32-bit write code, that is, writing to the MCB board itself with a WRT32 mode. The contents of the current pattern pointer are written to this register with the following effects:

Table 18 – Enable Function Register

Bit	Function	Bit	Function
31		15	Set SYNC_OUT signal high
30		14	Set SYNC_OUT signal low
29		13	
28		12	
27		11	
26		10	
25		9	Set LSR Auto-cancel feature off
24		8	Set LSR Auto-cancel feature on
23		7	
22		6	
21		5	
20		4	
19		3	
18		2	Stop Integration Time Counter
17		1	Start Integration Time Counter
16		0	Reset Start Exposure Flag

4.3.10.2 Integration Time Register and Control

The Required Integration Time Register is 32 bits wide and thus can be programmed by PAN with integration time of minimum of 1 millisecond up to a maximum of approximately 50 days. The integration period delay is implemented using an internal up counter. When the start bit of the Enable Function Register is asserted, the internal counter is reset to zero and it starts counting up every millisecond until a terminal count equal to the value loaded into the RIT register is exceeded. The TC flag in the command register is negated (0) until the ITC is less than RIT value. The ITC is paused by the pause ITC bit in the control register is asserted.

4.3.11 Clock Control

The sequencer clock is programmable by configuring the control register appropriately. Refer to the description of the Control Register. Thus for debugging purposes the sequencer can by slowed by a factor of 2, 4 or 8. However, the Virtex FPGA doesn't provide BUFGMUX modules to multiplex multiple clocks. To minimize the clock skew across combinational logic, the sequencer clock divider and multiplexer are implanted by using the clock enable based implementation. To route the sequencer clock on high speed fast global resources of the FPGA, the output of the clock divider is buffered through on-chip clock buffers. Note that the delays implemented are currently with respect to the fastest sequencer clock frequency of 20 MHz so if the sequencer clock is divided by a factor, all delays specified will get multiplied by the same factor.

4.3.12 Decode Execution Unit (DEU)

The decode execution unit is the central control logic of the sequencer. Based on the functionality performed by various instructions, the control logic internal to the Decode execution unit is grouped into various control and data path elements, which collectively perform various functions as briefly explained in sequel.

4.3.12.1 Program Address Multiplexer

As mentioned earlier, the DEU's access to the program memory Multiplexed Port is time-shared with the PAN Local address. The Program Address Multiplexer implements this priority multiplexed port.

4.3.12.2 Program Counter Control Logic

Increments the program counter by the size of the instruction (nibble) executed while performing sequential (non-branching) instructions. For branching and sub routine call instructions the counter is loaded with the address specified by the operands. While executing subroutine return and looping instructions, the program counter is loaded with address on the top of the program pointer address stack and loop begin address stack respectively. Apart from this program counter output, another output equal to "Program Counter + 1" is also provided to address the program memory Pipeline Port.

4.3.12.3 Instruction Multiplexer

The current instruction set implementation supports a variable instruction size from 1 to a maximum of 5 nibbles or 20 bits. So the maximum size instruction consists of 1 nibble opcode and 4 nibbles operand. The 32-bit data read per clock cycle from the two ports of the Program memory includes 16 bit data at the current memory location addressed by the program counter and the next 16 bits located at the next memory location. The opcode can be the first, second, third or fourth significant nibble of the 16-bit multiplexed port program word if the value of the least significant 2-bits of the program counter is b'00, b'01, b'10 or b'11 respectively. The operand for the maximum size instruction is the following 4 nibbles. Therefore, from the 32 bits program memory data, the **Instruction Multiplexer** selects the 20 bits valid instruction based on the least significant 2 bits of the program counter at the beginning of every execution cycle.

4.3.12.4 Pattern Pointer Control Logic

Logic initializes the pattern pointer to 0 when Reset is asserted. Decodes and executes LPP, IPP and DPP instructions when the sequencer run flag is asserted.

4.3.12.5 Output Registers and Control Logic

Apart from the pattern pointer sequencer include three other static registers, namely Select Register, Device Address Register and Output Register to control the sequencer bus. The control logic decodes and executes the LSR, LMR and LDA instructions to load the operand data into Select registers, Mode register and Device Address register on respective execution.

4.3.12.6 Program Counter Stack Control Logic

Decodes sub-routine Call and Return instruction and performs Stack Memory push (write) and pop (read) operations respectively.

4.3.12.7 Loop Control Logic

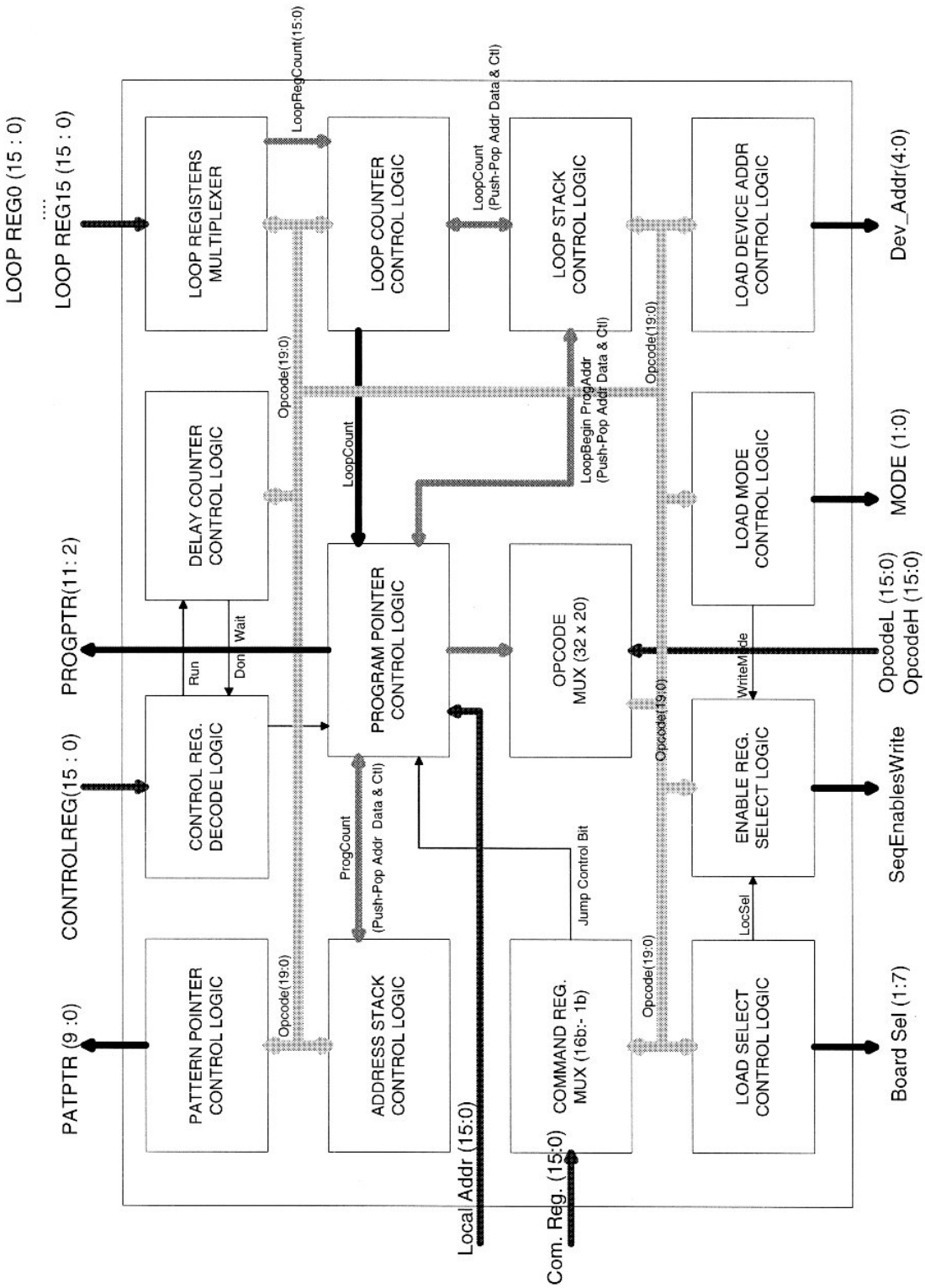
Controls the Loop Counter and selects the addressed Loop Register from the bank of 16 loop registers using an internal Loop register Multiplexer while executing looping instructions. Also controls the push and pop operations on Loop Count Stack and Loop Begin Address Stack for implementing nested loop functionality.

4.3.12.8 EFR Write Decode

Asserts a write to the enables register when local select bit in the select register is asserted and the pattern of the mode register bits correspond to the write mode.

4.3.12.9 Delay Control Logic

Three delay counters, clock-counter, microsecond counter and millisecond counter are designed to insert the explicitly specified blocking timing delays apart from the non-blocking integration time controller. The delay control logic decodes the delay instruction and loads the corresponding resolution down counter with specified value in the operands. Once the counters are loaded, they start decrementing to implement the specified delay and block the sequencer from executing the next instruction until all counters are reduced to inactive zero state. On reset, all three counters are initialized to zero value.



Decode Execution Unit
Figure 10

4.3.13 Instruction Set for the Sequencer

Instead of controlling the 32 sequencer signals (max) on the Sequencer address and data bus individually, the Sequencer Data Bus is grouped and is controlled as a 32-bits wide binary vector including the address and data bus. All the 32-sequencer signals are controlled simultaneously by changing the 32-bit word, defined as a Pattern, instead of switching one signal individually. Any arbitrary desired waveform can be expressed as a sequence of pattern with appropriate delay for each pattern. The output of the Sequencer can be divided into such elementary control-waveforms as Row-Shift Column-Shift Waveform. The Sequencer architecture includes a 32-bit wide and 1K deep memory called the Pattern Memory implemented using FPGA Block RAM. All valid 32-bit patterns required for generating the desired waveforms are loaded into the Pattern Memory and are accessed by Pattern Memory address. By sequentially changing the address of the Pattern Memory after the required amount of delays desired, waveform can be generated on the Sequencer Address Data-Bus.

One can observe that for any defined waveform, the Pattern Memory access will be generally sequential except for some exceptions. The Sequencer CPU indirectly addresses the Pattern Memory by a Pattern Pointer (similar to the Data Pointer in 8051). All instructions have a unique binary Opcode 1-nibble wide and 0-4 nibble wide operand based on the function executed by the instruction. Thus an instruction set of maximum 16 instructions with the longest instruction being 5 nibble or 20 bits wide and the shortest instruction being 1 nibble or 4 bits wide. However, all instructions will be executed in a single ***SEQCLK*** Based on the requirements and functionality executed, the instructions are classified as:

A. Output Instructions

B. Control Instructions

C. Delay Instructions

4.3.14 Output Instructions

Six different instructions are included to control all the output signals to the back-plane Sequence Bus. The first 3 instructions are designed to mange the pattern pointer efficiently with goals to optimize the program size and at the same time provide enough flexibility to the programmer to control the complete Sequence Bus. The next 4 instructions provide the programmer with necessary instructions to control the Sequence-Mode bits, Device Address bits and Board Select signals.

4.3.14.1 LPP: Load Pattern Pointer: 4 Nibble (16 bit)

This instruction loads the Pattern Pointer of the Sequencer CPU with the immediate 10-bit address specified by the next three nibbles in the program memory. Note that since the address space of the Pattern Memory for the current design is limited to 1K, the most significant 2 bits of Nib1 are don't care and the other two bits specify the most significant bits of the pattern address PA9 PA0.

LPP: *"Pattern Address"*

Nib 00	***Nib 01***	***Nib 02***	***Nib 03***
0001b	**2-bit X**	**10-bit Pattern Address. PA9 - PA0.**	

4.3.14.2 IPP: Increment Pattern Pointer: 1 Nibble (4-bit)

On execution, the Pattern Pointer is incremented by one. It is useful when waveform is running through a sequence of consecutive patterns in ascending order.

IPP:

Nib 00
0010b

4.3.14.3 DPP: Decrement Pattern Pointer: 1 Nibble (4-bit)

Decrements the Pattern Pointer by one and is useful when waveform is running through a sequence of consecutive patterns in descending order.

DPP:

Nib 00
0011b

4.3.14.4 LDA: Load Device Address (12-bit)

It loads the 5-bit Sequence Device Address Register, which is the output register for the ***DEV_ADDR*** [4:0] signals and the D3, D2 & D1 bits of the first nibble are don't care.

LDA: ***"Device Address"***

Nib 00	*Nib 01*		*Nib 02*
0100b	***D3-D2:x***	**D0: Dev_Addr (4)**	**D3-D0 : Dev_Addr (3-0)**

4.3.14.5 LMR: Load Mode Register 3 Nibble (12-bit)

It loads the 2-bit Sequence Mode register, which is the output register for the ***SEQ_MODE*** [1:0] signals and the D3 & D2 bits of the first nibble are don't care.

LMR: ***"Mode Value"***

Nib 00	*Nib 01*		
0101b	***D3: x***	**D2: x**	**D1-D0: *SQM1-SQM0***

4.3.14.6 LSR: Load Select Register 3 Nibble (12-bit)

It loads the 8-bit board Select Register, which is the output register for the board select signals Sel#8:1. Note that the least significant select SEL1 is a sequencer internal select signal.

LSR: "Select Value"

Nib 00	*Nib 01*		*Nib 02*
0110b	*D3x*	D2-D0 Sel6-Sel4	*D3-D0 Sel3-Sel0*

4.3.15 Control Instructions

4.3.15.1 CAL: Call Subroutine: 4 Nibble (16 bit)

This is the classic CALL instruction similar to 8085; the address pointing to the sub-routine is an operand, which for the current implementation is (12-bit). The next address is pushed to the address stack and control is transferred to the specified address.

CAL: "Subroutine *Address*"

Nib 00	*Nib 01*	*Nib 02*	*Nib 03*
1000b	12-bit Subroutine Address. A11 - A0		

4.3.15.2 RET: Return 1 Nibble (4-bit)

This is the classic RETURN instruction similar to 8085; all sub-routines called must end with a return instruction. On execution of RET, return address is popped from the top of the stack and control is transferred to that address.

RET:

Nib 00
1001b

4.3.15.3 JCB: Jump if Control Bit Set. 5 Nibble (20-bit)

For conditional branching, a conditional jump instruction is provided.

Unlike all other instructions, the JCB includes two types of operands and control is transferred to the specified address if the corresponding control register (16-bit wide) flag is set.

JCB: "Bit Number"; "*Address*"

Nib 00	*Nib 01*	*Nib 02*	*Nib 03*	*Nib 04*
1010b	*BN3-BN0*	12-bit Address. A11 - A0		

4.3.15.4 LRB: Loop Begin 2 Nibble (8-bit)

The next program counter address and the loop counter value pointed to by the loop register index in nibble 01 are pushed onto their respective stacks. The stack implementation permits the use of nested loops of degree equivalent to stack depth.

LRB: "Index"

Nib 00	*Nib 01*
1010b	4-bit Loop Register Index 0:15

4.3.15.5 LPB: Loop Begin 1 Nibble (4-bit)

The next program counter address and the value of the pattern memory location pointed to by the pattern pointer are pushed onto their respective stacks. The stack implementation permits the use of nested loops of degree equivalent to stack depth.

LPB:

Nib 00
1011b

4.3.15.6 LPE: Loop End 1 Nibble (4-bit)

The loop count on the top of the stack, if not zero, is decremented and address stored on the top of the program counter address stack is loaded into the address pointer. If the loop count is zero, the count and program counter address are popped off the stack and program counter address is incremented.

LPE:

Nib 00
1100b

4.3.16 Delay Instructions

With a combination of the following instructions, any arbitrary delay (less than a second) can be implemented. A corresponding down counter is loaded with the count specified by the operand. The delay values for every instruction are currently limited to 255*least count. This is based on the assumption that most frequently required delays will be within this range. However, if the delay values required are in the complete range of 0-999 *Least count, either the size of the operand or the least count can be scaled to provide the complete range accordingly.

4.3.16.1 DMS: Delay Milliseconds 3 Nibble (12-bit)

A delay with resolution (least count) of a millisecond and of magnitude equivalent to the value specified by the operand "Delay mili seconds count" (8-bit) is inserted. None of the output registers are altered while this instruction is executed. The program counter is halted until the delay counter is reset.

DMS: "Delay ms Count"

Nib 00	*Nib 01*	*Nib 01*
1101b	*DmsC [7:0]*	

4.3.16.2 DμS: Delay Microseconds 3 Nibble (12-bit)

A delay with resolution of a microsecond and of magnitude equivalent to counter value specified by the DμsC Delay micro seconds count (8-bit) is inserted. None of the output registers are altered when this instruction is executed. The program counter is halted until the delay counter is reset.

DUS: "Delay μs Count"

Nib 00	*Nib 01*	*Nib 01*
1110b	***DμsC [7:0]***	

4.3.16.3 DSC: Delay System Clock 3 Nibble (12-bit)

A delay with resolution of ***SEQCLK*** period and of magnitude equivalent to counter value specified by the DSCC Delay system clock count (8-bit) is inserted. None of the output registers are altered when this instruction is executed. The program counter is halted until the delay counter is reset.

DSC: "Delay System Clock Count"

Nib 00	*Nib 01*	*Nib 01*
1111b	***DSCC [7:0]***	

4.3.16.4 NOP: No Operation 1 Nibble (4-bit)

A delay with resolution of ***SYSCLK*** period and of unit magnitude is inserted. None of the output registers are altered when this instruction is executed.

NOP:

Nib 00
000b

4.4 Appendix IV – Synthetic Pixel Generator

Revision 4.63 of the Pixel FPGA firmware (McbPixFpgaV463.mcs) contains a comprehensive pixel data generator that allows testing of the MONSOON data paths and software. The data generator produces synthetic pixel data in an amount and cadence that represents data produced by physical acquisition boards. This can be used to test and optimize the pixel data path without having to use the intrument or detectors. Also, the data generator can be programmed to inject synthetic data into an existing DHE data path to replace one or ore disabled ormissing acquisition baords. The pixel generator suppoorts both visible and infrared readout schemes. Synthetic pixel dta contains a channel number tag and an 8-bit sythetic data field. The data field can be programmed as a sequential ramp or as pseudo noise values. The generator is controlled by the attributes described in Table 19.

Table 19 - Pixel Generator Memory Map

Address	Function Name	Function Description
0x02FC (15:13)	SIM_MODE	3-bit field of the pixel control register. Controls the pixel generator operational mode.
0x02D0	SIM_PIXROWS	Number of pixel rows to synthesize.
0x02D1	SIM_PIXCOLS	Number of pixel columns to synthesize.
0x02D2	SIM_PIXTIME	Period between pixel bursts.
0x02D3	SIM_CHANNELS	Number of pixels to generate each burst.
0x02D4	SIM_DELAY	Time to delay before initial burst is generated.
0x02D5	SIM_INTEGRATE	Integration time for IR mode 2.
0x02D6	SIM_FOWLER	Number of Fowler samples to take in IR mode 2.
0x02D7	SIM_PIPE	Pipeline priority time slots to generate in mode 3.

4.4.1 Simulation Mode Register (SIM_MODE)

This is a 3-bit field in the pixel control register. The value in this register controls the moe o the pixel generator. Table 20 shows the availabel mode values.

Table 20 – Pixel Generator Mode Description

SIM_MODE	Function
0	Pixel generator disabled. No synthetic data generated. (default value).
1	Generate pixels in visible mode. (emulate CCD) - sequential data.
2	Generate pixels in IR mode - sequential data.
3	Generate replacement pixels on time slot - sequential data.
4	Pixel generator disabled. No synthetic data generated.
5	Generate pixels in visible mode (CCD) - noise data.
6	Generate pixels in IR mode - noise data.
7	Generate replacement pixels on time slot - noise data.

Generate pixels in visible mode – The pixel genrator will be triggered on the Start Exposure command sent to the MCB. There will be (SIM_PIXROWS x SIM_PIXCOLS) bursts of data separated by (SIM_PIXTIME x 100ns) time. For each burst, SIM_CHANNELS worth of pixels will be genreated. The data will contain an 8-bit channel ID field (repeated each burst) and either an 8-bit incremental pixel value (SIM_MODE bit 2 = 0) or an 8-bit pseudo noise value. There will be a dealy of (SIM_DELAY x 1ms) before the first burst of pixels is sent to the PAN. Total number of pixels sent to the PAN will be (SIM_PIXROWS x SIM_PIXCOLS x SIM_CHANNELS).

Generate pixels in IR mode – The pixel generator will be triggered on the Start Exposure command sent to the MCB. There will be an initial delay of (SIM_ELAY x ims) before the data begins to arrive at the PAN. For the value of SIM_FOWLER, there will be (SIM_PIXROWS x SIM_PIXCOLS) bursts of data separated by (SIM_PIXTIME x 100ns) time sent to the PAN. Before each Fowler smaple, there will be a delay of (SIM_DELAY x 1ms). After the completion of these data frames, the integration timer will waitr (SIM_INTEGRATE x 1ms) before starting the second readout. The second readout will follow exactly the same process as pre-integration. Total number of pixels sent to the PAN will be (2 x SIM_PIXROWS x SIM_PIXCOLS x SIM_CHANNELS x SIM_FOWLER). The data will contain an 8-bit channel ID field (repeated each burst) and either an 8-bit incremental pixel value (SIM_MODE bit 2 = 0) or an 8-bit pseudo noise value.

Generate replacement pixels on time slot – this mode value is used to inject synthetic pixel data into the normal acquisition data stream generated by a set of MONSOON acquisition boards. In this mode, the 8-bit SIM_PIPE attribute is set up to indicate into which sequence time slots the data is to be placed. Table 21 shows the releveance of this register's bit positions.

Table 21 – Pixel Generator Pipeline Priority Register

Pipeline Priority Value	Bit Position
0	Not available
1	Bit 1 – value 0b00000010
2	Bit 2 – value 0b00000100
3	Bit 3 – value 0b00001000
4	Bit 4 – value 0b00010000
5	Bit 5 – value 0b00100000
6	Bit 6 – value 0b01000000

In this mode it is essential that the pipeline prioity time slot being emulated by the pixel generator be preceded by either a real acquisition board pixel transfer time slot or a previous pixel generator time slot. This is because the pixel generator uses the edges of the pipeline prioity signal to "know" when the correct time slot is available so it can begin the data transfer. Examples follow.

Example: Acquisition boards in DHE slots 3, 4, 5 and 6 – The acquisition board in slot 5 has been disabled)pipeline enable = 0). Pipeline priority values are programmed as 0, 2, 4 and 6 on the respective boards. Setting the SIM_PIPE register to 3 will produce the reuqired result since the pixel generator data will be triggered at the end of the data transfer from the acquisition board in slot 4 at pipeline priority time slot 2. Setting the SIM_PIPE register to 4 will not produce data since there will be no end of transfer activity on priority 3 to trigger the pixel generator.

For simlar reasons pertaining to the above, it is not possible to command the pixel generator to initiate pipeline transfers with a pipeline priority value of 0.

In this mode, for each pipeline priority bit set in the SIM_PIPE register, the number of pixels gerneated will be equal to the value of SIM_CHANNELS.

4.4.2 Simulation Pixel Rows Register (SIM_PIXROWS)

This is a 16-bit register that controls the number of synthetic pixel data that is generated. The product of SIM_PIXROWS x SIM_PIX COLS is equal to the number of bursts that will be generated for each data frame. This register is not used in simulation modes 3 or 7.

4.4.3 Simulation Pixel Columns Register (SIM_PIX COLS)

This is a 16-bit register that controls the number of synthetic pixel data that is generated. The product of SIM_PIXROWS x SIM_PIX COLS is equal to the number of bursts that will be generated for each data frame. This register is not used in simulation modes 3 or 7.

4.4.4 Simulation Pixel Time Register (SIM_PIXTIME)

This 8-bit register controls the time between pixel bursts in units of 100ns. For example, to emulate a pixel data rate of 250Kpix/sec, set this register to a value of 40. This register is not used in simulation modes 3 or 7.

4.4.5 Simulation Channels Register (SIM_CHANNELS)

This 8-bit register controls the number of synthetic pixels generated for each burst. For example, to simulate four CCD acquisition boards, each with eight channels, set this register to 32. This register is used in all active simulation modes.

4.4.6 Simulation Delay Register (SIM_DELAY)

This 8-bit register controls two dealys. In modes 1, 2, 5 and 6, it allows specifying an initial delay from the arrival of the Start Exposure command to when the first pixel burst is generated and sent to the PAN computer.This delay is specified in this register in lilliseconds. In mode 2, the register also controls the timebetween Fowler sample data frames. This register is not used in simulation modes 3 or 7.

4.4.7 Simulation Integration Register (SIM_INTEGRATE)

In mode 2 and 6 (IR mode), this 16-bit register sets the simulated integration time between the pre-integration readout and post-integration readout data streams. Set this register to the required integartion time in milliseconds. This register is not used in modes 1, 3, 5 or 7.

4.4.8 Simulation Fowler Register (SIM_FOWLER)

In simulation modes 2 and 6, this 8-bit register determines the number of data frames sent ot the PAN in the pre-integration and post-integration simulation. Each data frame wil transmit (SIM_PIXROWS x SIM_PIXCOLS x SIM_CHANNELS) pixels to the PAN. This register is not used in modes 1, 3, 5 or 7.

4.4.9 Simulation Pipe Register (SIM_PIPE)

This 8-bit register is only used in modes 3 and 7. It specifies the pipeline prioity time slots to be used by the pixel generator to send data to the PAN. The significance of this register is hsown in Table 21.

4.4.10 Limitations of the Pixel Generator

There are several combinatins of mode and operating configuration that cannot be simulated by the pixel generator. These combinatins ar elisted by their respective SIM_MODE values.

- Modes 1 and 5 - SIM_CHANNEL = 1 and (SIM_PIXROWS x SIM_PIXCOLS) = odd number.
- Modes 2 and 6 - SIM_CHANNEL = 1 and (SIM_PIXROWS x SIM_PIXCOLS) = odd number.
- Modes 3 and 7 - SIM_CHANNEL = 1; Set the SIM_CHANNEL value to 2. It will operate correctly.

4.4.11 Sample Configuration File Setup

The following ten lines indicate a usable configuration file structure to include in the PAN.cxv file.

```
SimPipe,SIM_PIPE,0x00102D7,1,0x0E000000,SIMPLE,SIMPLE,FLOAT,UCHAR,1
.0,0.0,LINEAR,0.0,128.0,Timeslot,PIPELINE RPIORITY TIMESLOT MASK. I
= GENERATE DATA, 0 = DON'T GENERATE

SimIntegrate,SIM_INTEGRATE,0x00102D6,1,0x0E000000,SIMPLE,SIMPLE,FLO
ATUINT,1.0,0.0,LINEAR,0.0,65535.0,millisec,SIMULATED INTEGRATION
TIME IN MILLISECONDS

SimDelay,SIM_DELAY,0x00102D4,1,0x0E000000,SIMPLE,SIMPLE,FLOAT,UCHAR
,1.0,0.0,LINEAR,0.0,255.0,millisec,INITIAL DELAY BEFORE FIRST PIXEL
BURST

SimTime,SIM_TIME,0x00102D2,1,0x0E000000,SIMPLE,SIMPLE,FLOAT,UCHAR,1
.0,0.0,LINEAR,10.0,255.0,Time,PERIOD BETWEEN PIXEL BURSTS IN UNITS
100ns

SimFowler,SIM_FOWLER,0x00102D5,1,0x0E000000,SIMPLE,SIMPLE,FLOAT,UCH
AR,1.0,0.0,LINEAR,0.0,255.0,Samples,NUMBER OF FOWLER SAMPLES PER
READOUT IN IR MODE

SimChannels,SIM_CHANNELS,0x00102D3,1,0x0E000000,SIMPLE,SIMPLE,FLOAT
,UCHAR,1.0,0.0,LINEAR,0.0,255.0,Pixels,NUMBER OF PIXELS TO GENERATE
FOR EACH BURST

SimCols,SIM_COLS,0x00102D5,1,0x0E000000,SIMPLE,SIMPLE,FLOAT,UINT,1.
0,0.0,LINEAR,0.0,65535.0,Pixels,NUMBER OF SIMULATED PIXEL COLUMNS

SimRows,SIM_ROWS,0x00102D0,1,0x0E000000,SIMPLE,SIMPLE,FLOAT,UINT,1.
0,0.0,LINEAR,0.0,65535.0,Pixels,NUMBER OF SIMULATED PIXEL ROWS
```

```
SimData,SIM_DATA,0x00102FC,1,0x0E000000,RDMSKWRT,RDMSKWRT,FLOAT,UIN
T,32768.0,0.0,LINEAR,0.0,1.0,Value,0 = INCREMENTAL DATA, 1 = NOISE

SimMode,SIM_MODE,0x00102FC,1,0x0E000000,RDMSKWRT,RDMSKWRT,FLOAT,UIN
T,8192.0,0.0,LINEAR,0.0,3.0,Value,0 = DISABLED,1 = CCD MODE,2 = IR
MODE,3 = TIMESLOT REPLACEMENT MODE
```

4.5 Appendix V – Firmware Product History for Rev A and B Hardware

Table 22 - Sequencer FPGA Code Versions

Release Version	Checksum / User Code	Comments.
McbSeqFpgaV40	UC=04051801 CS=	First cut for Rev -A- hardware. Derived from McbPixFpgaV34 code.
McbSeqFpgaV41		Modified to support old 8 slot Schroff backplane clock assignments.
McbSeqFpgaV42		Added support for temperature sensor and silicon serial number device.
McbSeqFpgaV421	UC=04092001 CS=	Added support for switch-able back plane support.
McbSeqFpgaV43	UC=04120401 CS=	Corrected problem with sequencer initialization. Corrected bug that produced double write strobes to the EFR register.
McbSeqFpgaV44	UC=05021501 CS=	Corrected MCB control register conflict. Added support for double word transfers. Added debug test lines.
McbSeqFpgaV45	UC=05100401 CS=	Added support for the event register and debug code.
McbSeqFpgaV461	UC=06041901 CS=014CE218	Added support for synthetic pixel generator. FPGA_CTL_2 wired to 'Start Exposure' signal.
McbSeqFpgaV462	UC=06081801 CS=	Adapted and built for use with S-LINK CMC.
McbSeqFpgaV463	UC=06092301 CS=014D1CF7	Added PAN command "echo disable" bit to disable the command echo.
McbSeqFpgaV464	UC=07062601 CS=014E11C2	Added support for switch mode power supply synchronization. Status is experimental.
McbSeqFpgaV465	UC=07080701 CS=014D1930	Changed the use of ExData signals to enhance synchronization logic.
McbSeqFpgaV491	UC=06101101 CS=014D6BED	Built to support the unique requirements of the QUOTA instrument. Sequencer command register structure is not backwards compatible. Morphed from McbFpgaV461.
McbSeqFpgaV492	UC=07061101 CS=014EDD91	Added a hardware "peripheral wait" signal to allow the sequencer to test the readiness of complex functions being performed on peripheral boards. QUOTA Only.

Table 23 - Pixel FPGA Code Versions

Release Version	Checksum / User Code	Comments.
McbPixFpgaV40	UC=04051701 CS=	First code cut for Rev -A- hardware. Derived from McbPixFpgaV34 code.
McbPixFpgaV42	UC=04101301 CS=	Incorporated tally counters for data transmitted through the FPDP.
McbPixFpgaV44	UC=05021502 CS=	Added double pixel transfers through FPDP and extra debug registers for busy timing on FPDP.
McbPixFpgaV45	UC=05091501 CS=	Re-ordered odd / even double pixel transfer word to coincide with PAN expectations
McbPixFpgaV451	UC=05091501(!) CS=	Increased pixel bus FIFO to 128 pixels deep. Added circular buffer to capture FPDP traffic.
McbPixFpgaV46	UC=05102501 CS=015821A4	Corrected problem with FPDP transmit port where occasionally the SYNC bit was deterministically cleared before pixel data was sent. Added pixel firmware revision value at address 0x02ff.
McbPixFpgaV461	UC=06020903 CS=01525CA7	Added true pixel FIFO flush on reset to address random 80808080 data being sent after MCB reset. Added first version of synthetic pixel generator.
McbPixFpgaV462	UC=06020801 CS=	Adapted and built for use with S-LINK CMC.
McbPixFpgaV463	UC=07060603 CS=0151C511	Added IR exposure simulation and time slot replacement modes to synthetic pixel generator.
McbPixFpgaV464	UC=07080701 CS=0159E9F9	Changed the use of ExData signals to enhance synchronization logic.