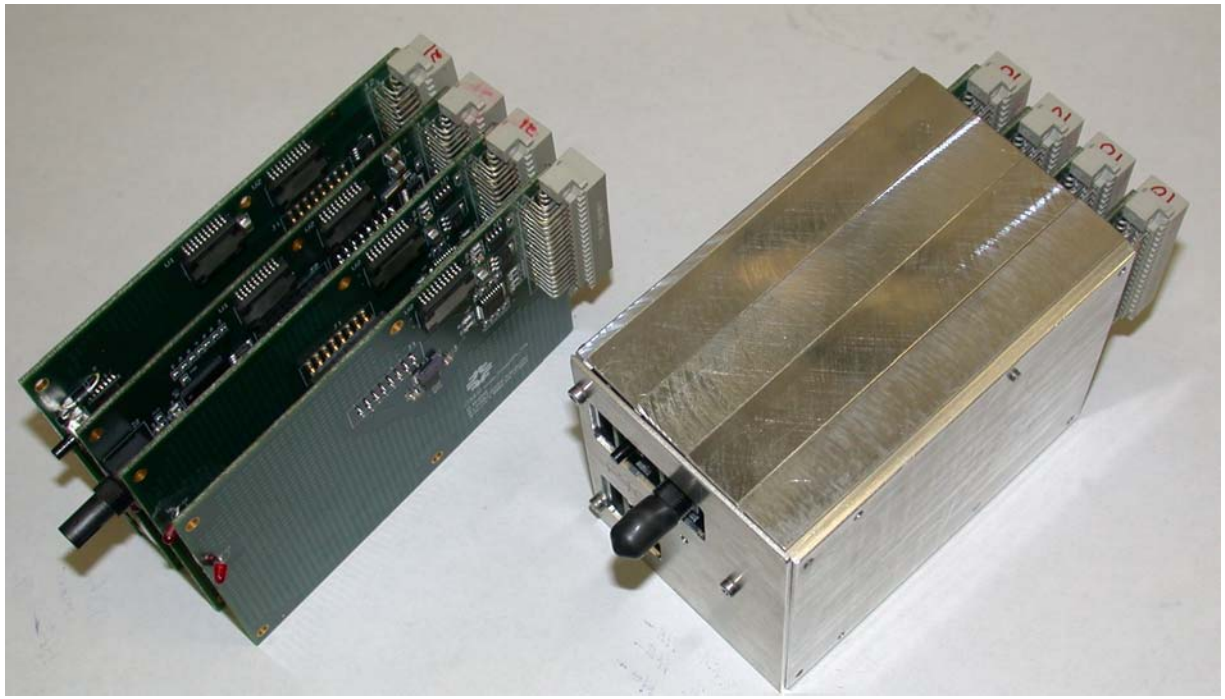
Clock, Control and Monitoring (CCM) Module

Documentation and Communication Protocol



The Clock, Control and Monitoring (CCM) Module is being designed for the Hadron Calorimeter subsystem of the CMS Detector[1]. The CCM is an integral part of the Front End Readout electronics and operates in conjunction with twelve Front End (FE) Readout cards[2] that condition and digitize data from the Calorimeter. The CCM and the FE cards plug into a custom backplane that resides in a custom crate called a Readout Box (RBX). The backplane, besides distributing power, is used to provide a serial communication path between the CCM and four groups of three FE cards.

The CCM module operates in a radiation environment and component selection was a critical factor in the design. Components were chosen that were not susceptible to a Total Ionizing Damage (TID) of less than 330 rads and were also Single Event Effects (SEE) tolerant. Access to the CCM will be minimal and the other FE boards will have limited functionality without a proper operating CCM.

The CCM performs three functions: 1). Receive and distribute a low skew 40 MHz. Clock along with a beam-crossing marker to the FE cards. 2). Monitor the temperature at the FE cards and also the power supply voltages that are present on the backplane. 3). Provide a means for a Main Control system to communicate with the ASICs located on the FE cards and also provide a method to download parameters to those ASICs in a global manner.

The CCM module consists of four printed circuit boards that reside within a custom chassis. The chassis plugs into the middle location on the backplane located within the HCAL Readout Box.

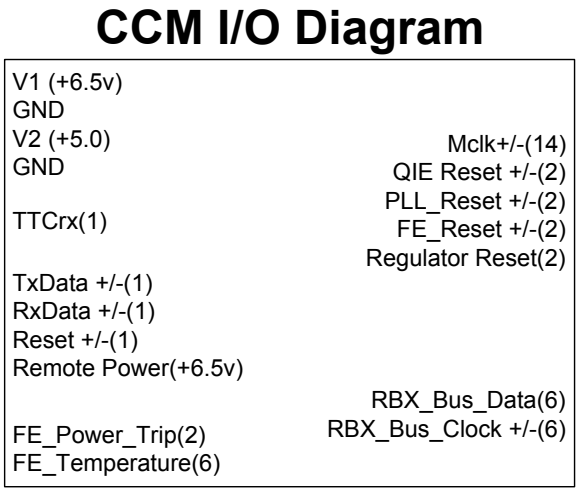


Fig. 1. CCM Module I/O diagram.

Board Design

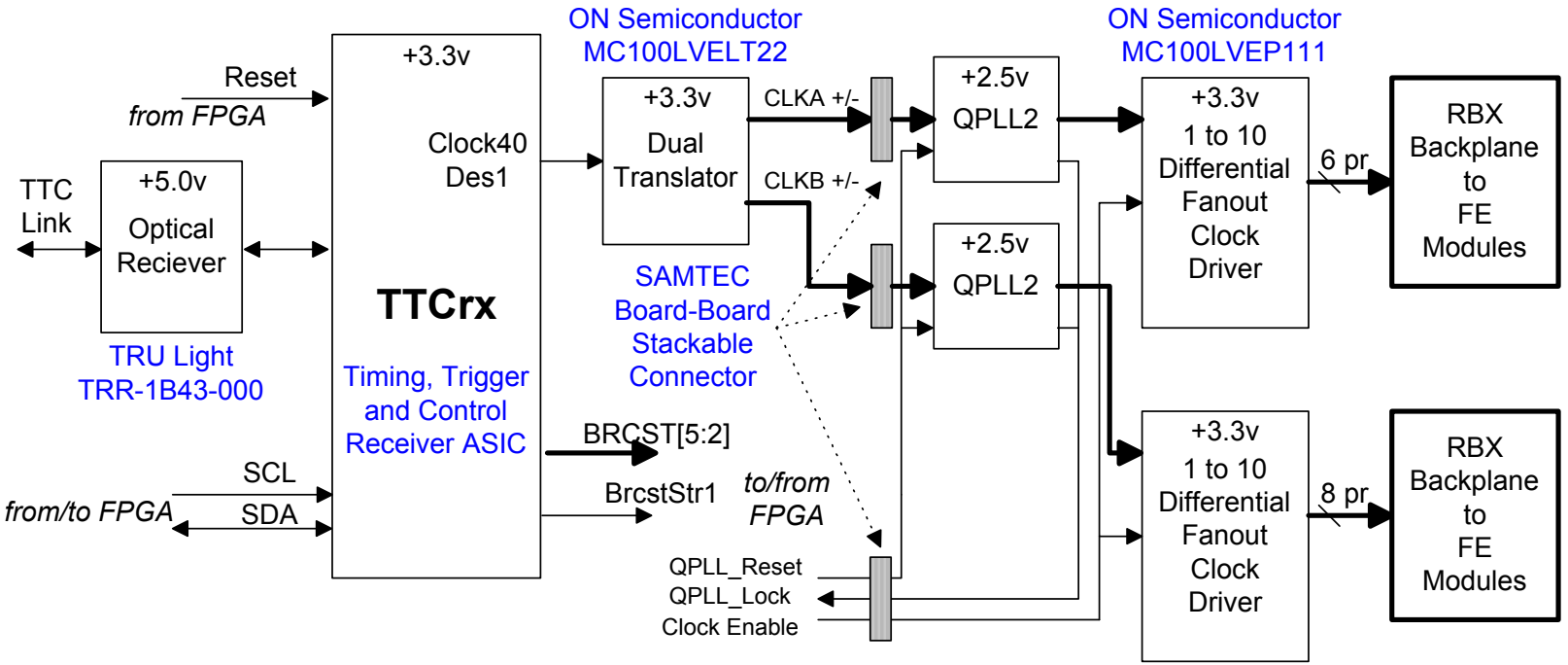
The four - 6 layer printed circuit boards are all based on a design, which allows four signal layers, a split power plane and a ground plane. All four boards will be physically located in a custom Aluminum chassis, which will provide a cooling path to allow heat transfer away from the boards. The boards interconnect with short board-board stackable connectors. The size constraints of the RBX required the four boards to be 8cm tall and 12cm long with a board spacing of 1.6cm.

The CCM module is electrically isolated from both the Main Control system and the TTC clock system using optical isolation devices. Power to the module comes from the RBX backplane in the form of two voltages, +6.5v and +5.0v. Radiation Hardened voltage regulators LHC4913 are used on each of the four boards to reduce the voltages to +5.0v, +3.3v and +2.5v.

Clock Section

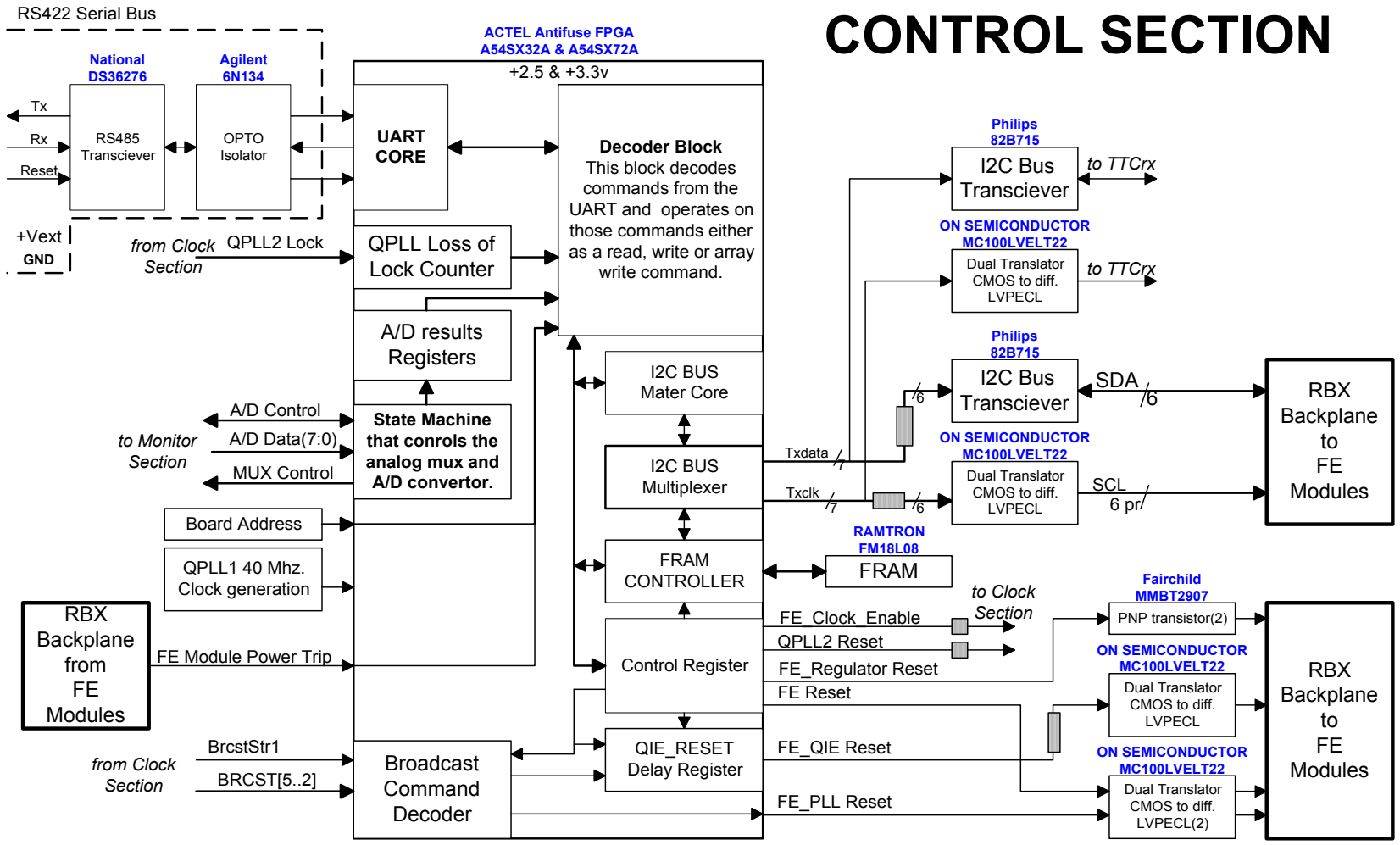
The Master Clock is received from the Timing, Trigger and Control (TTC)[3] system. The TTCrx a custom IC that was developed by the CERN MICROELECTRONICS GROUP receives the timing information. The 40.08 MHz bunch–crossing reference clock signal along with a Broadcast message are extracted from the custom IC. The Master clock signal is distributed onto the backplane to the FE cards using an ON-Semiconductor MC100LVEP111 differential LVPECL clock driver. The jitter of the raw 40.08 MHz clock output of the TTCrx is too high to effectively operate the high-speed serializers on the FE boards. A VCXO/PLL ASIC has been developed by the CERN Microelectronics Group for use as a jitter filter. The QPLL2 is expected to have a peak-peak output jitter of less than 50 ps and preceeds the MC100LVEP111 clock driver chip in the clock fanout chain. The QPLL2 devices will provide a signal that represents the locking of the PLL of the device to the incoming signal from the TTCrx. This lock signal is monitored by the CCM. A counter within the control section will increment when the QPLL2 device locks. The Clock Enable signal allows the clock signal from the QPLL2 to be distributed onto the backplane and thus to the FE Modules. When the Clock Enable Signal is the signal outputted onto the backplane is a fixed differential logic low level. The TTCrx device has internal registers that can be accesed using the I2C interface form the control section.

Clock Section



Control Section

The Control Section performs a number of functions including interfacing with the Main control system, distributing control signals to the FE boards and controlling the Clock Fanout. Two Actel Antifuse FPGAs along with associated drivers and receivers provide the serial communication link between the FE card's ASICs and the CMS HCAL main control system. The FPGAs acts as a slave to the Main Control system and as a master to the FE cards.

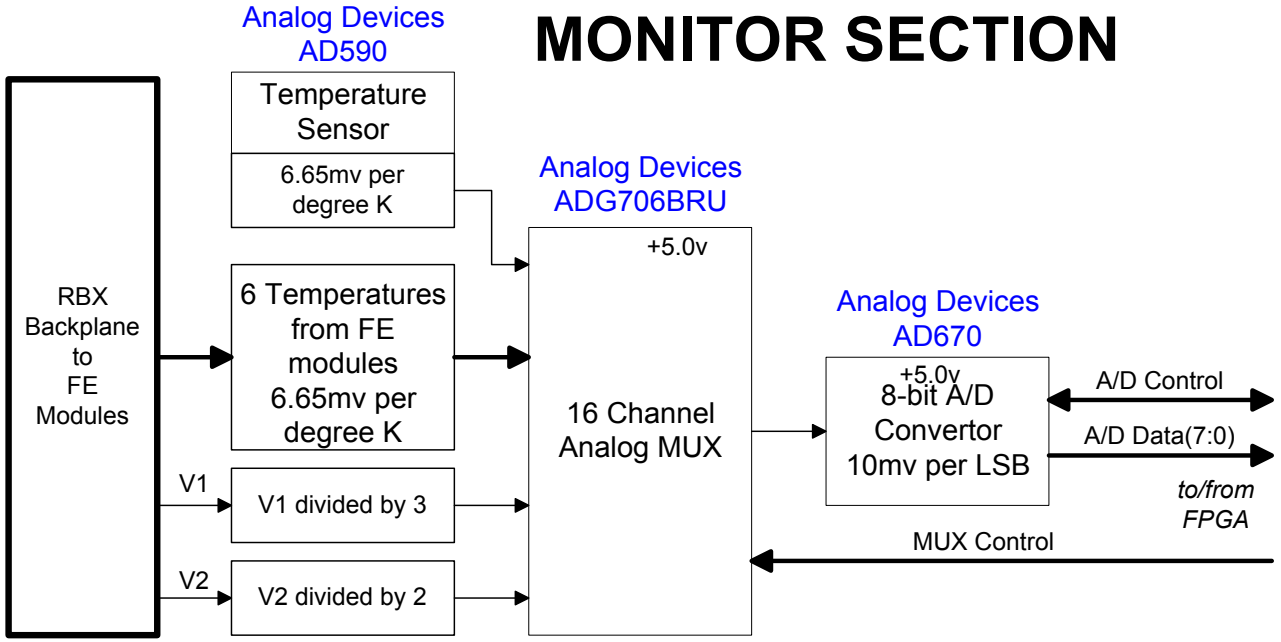


An optically isolated serial path allows communication between a UART core within the FPGA and the Main Control system using National Semiconductor's DS36276 RS485 transceivers and Agilent 6N134 opto-couplers. A four pair Category 5 cable is used for the serial link with a separate transmit and receive pair along with a power and reset pair. The FPGA receives the information from the Main Control system and distributes it onto one of four serial communication paths labeled the RBXbus across the backplane. Each of the four RBXbus consists of a data line and differential LVPECL clock line. Each RBXbus on the backplane connects to three FE boards grouped together to form a readout module. A Ramtron FM18L08 Ferro-Electric RAM is used on the Control board to hold preset parameters that are downloaded on a command to all FE cards using the same four RBXbus on the back plane.

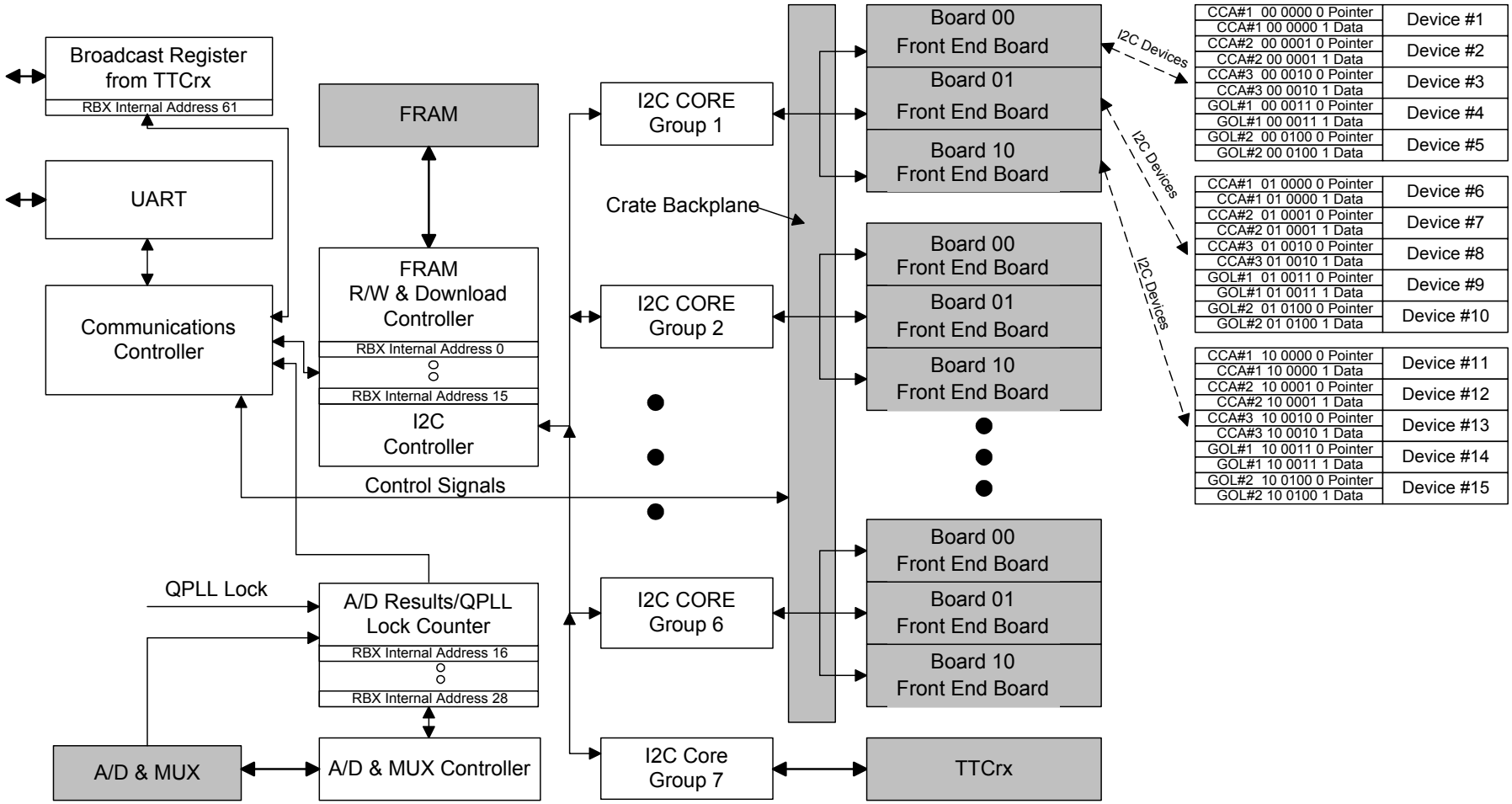
The Main control system is also able to send various controls or reset signals via the FPGA to the FE boards using On-Semiconductor MC100LVELT22 differential LVPECL translators.

Monitor Section

An Analog Devices AD590 Temperature Transducer produces a current representing the temperature of each readout module. The current is converted to a voltage and is sent via the backplane to an ADG706 Analog Mulitplexer. These temperatures along with the power supply voltages are converted to a digital word using the Analog Devices AD670 Bipolar 8-Bit Analog to Digital converter. The temperature and voltage readings can be accessed by the Main Control system through the Control Board of the CCM.



FPGA internal block diagram & external connections



Shaded boxes are outside the FPGA. There is actually one I2C core which is multiplexed to 7 I/O's.

COMMUNICATIONS PROTOCOL V6.1 4/27/04

SEE THE COMMUNICATION PROTOCOL of S. SERGUEEV> 8 bit command and operand> xxyyyyy
http://cmshcal.web.cern.ch/cmshcal/DCS/images/DCS_CMSw_27.09.01_SS.pdf
http://sergueev.home.cern.ch/sergueev/hcal/HCAL_DCS_architecture.doc

The ASICS that are populated in CMS HCAL FE modules contain R/W register. These registers are all implemented with the I2C format. The CCM module can communicate with these registers by using an I2C core within a FPGA device within the CCM. The CCM can communicate with an operator using RS422 protocol. The CCM manipulates the RS422 protocol in a manner which allows the CCM to R/W the registers within the ASICS on the FE modules.

To write/read data to a register on a FE board the operator must:

1. Select I2C register in a I2C device(CCA(28 registers), GOL(6 registers)).
2. Select I2C device on a I2C bus(15 I2C devices on a bus, uses 6 address lines)
3. Select I2C bus(core#)(7 I2C buses per CCM – 6 FE groups & 1 TTCrx)
4. Select CCM address(64 addresses – I2C, A/D, RAM, I/O, Group #, Status)
5. Select RBX node(1-9) from Custom HUB.
6. Select Custom RS HUB(1-18) from Industrial HUB.
7. Select COMTROL Rocketport PCI 550 HUB(1-3).
8. Invoke Master PC to communicate with FE board.

RBX – Readout Box
Commercial Hub – RS232 to RS422 fanout – 16 output ports, COMTROL Rocketport PCI 550
Custom RS Hub – RS422 to RS422 - 1 to 9 Fanout box
CCM – Clock and Control module – resides in the RBX.
FE – Front end module – resides in the RBX
CCA - Custom ASIC residing on a FE module, 3 per FE module.
GOL - Custom ASIC residing on a FE module, 2 per FE module.
TTCrx - Custom ASIC residing on a CCM module, 1 per CCM module.
QPLL2 - Custom ASIC residing on a CCM module, 2 per CCM module.
I2C – 2 wire Serial Bus.

How a HUB is selected is setup with the communication ports and software of the Industrial HUB.

Each CCM will have 64 addresses that are used to select a group within the RBX to communicate with.

Address 0-15 are reserved for the I2C Cores and FRAM(reset parameters).
Address 16-25 are reserved for the A/D converter results.(temperature and voltage levels)
Address 26-27 are registers that contain a count of the number of times each of the QPLL2s have lost sync.
Address 28-60 are unused.
Address 61 is Broadcast Enable register.
Address 62 is Delay register.
Address 63 is Status/Control register.

The node number of the CCM is implemented with a one-time programmable shunt located on the board. After every communication the CCM responds with the node number and the contents of the status register.
The Master PC sends commands to a particular CCM in a RBX.
The command XXYYYYYY contains the command code itself(XX) and the operand(YYYYYY).

XX = 00 = is used by the Custom RS HUB. The Custom Hub only responds to 1 command – a two byte write command. The command "00111111" followed by a second byte selects a Hub port and also has the option of sending a reset to that selected port. The four LSB of the second byte in hex format select the ports(1-9), bit 5 if '1' sends a reset to that port.

! Custom hub responds with Node number and Status register – Status register shows which port is selected.

XX = 01 = reads out the content of the register of the CCM address specified by the operand(YYYYYY).

! CCM responds with Node number, Status register and selected register's content.

XX = 10 = writes single byte of information to the CCM address specified by the operand(YYYYYY). A second byte follows with data to be written to the address specified.

! CCM responds with Node number, Status register content.

XX = 11 = write an array of data to the CCM at address specified by the operand(YYYYYY). The second byte contains the size of the array and the following bytes contain the data to be written to the address specified.

! CCM responds with the Node number and Status register.

I.E. to read the data stored in the register with address 39 of the CCM, write address 39 in 6 LSB along with 01 in two MSB. The CCM returns a 3 byte message – Node number, Status register content, register 39's data.

01 100111 ! Selects address 39 for a read.
! CCM responds with Node number, Status and selected address(39) content.

I.E. to write a value of 27 into the register with address 62 of the CCM, write address 62 in 6 LSB along with 10 in two MSB followed by the 27. The CCM returns a 2 byte message – Node number, Status register content.

10 111110 ! Selects address 62 for a write.
00011011 ! value of 27 written into the register at address 62
! CCM responds with Node number and Status.

I2C Device and FRAM Communications
! The CCA can do multiple I2C reads and writes, the GOL can only do one per I2C access.

The communication protocol for I2C devices in the RBX call for access to 2 address spaces in a device. These two address spaces are called the pointer and data register. An I2C device will have a number of registers but only two are accessed by the I2C bus. To write or read an I2C device register first the register address must be put in the pointer register. The I2C Cores of the CCM will be controlled by an I2C Controller. The I2C Core will do the formatting of the command and data that is communicated to the I2C device. Data that is written to the I2C devices will come directly from the RBX master or the FRAM. In either case the CCM Internal Address(0-15) locations are used to hold data while it accessed by the I2C Controller. Data that is read from the I2C devices will be stored in the same CCM Internal Address(0-15) locations while the RBX master is performing the reads. These addresses are defined below.

0 = Control(r/w) and I2C Core/FRAM selector(1-7), MSB =R/W, LSB = 000 0000 each bit enables an I2C core.
1 = Number of transfers.
2 = I2C Device Register Pointer/FRAM lower order address.
3 = I2C Device Data Register/FRAM higher order address.
4-15 = Data registers.

I2C to RBX communication

To **read** an I2C device register via the RBX– perform a write command to the CCM internal address 0, 1, 2 and 3. Write: the Core # and the read command into address 0, the number of reads in address 1, the register pointer address of the particular device into address 2 and the register number into address 3. The I2C Controller and I2C Core selected will perform the read and store the results back into addresses 4-15(depending upon the number of reads). Follow the write command from the master with a read command(XX=01) of data in the register that is being pointed to by the channel selection register(address 4). The status of the communication will be forwarded to the RBX address 63 – status register.

! All writes by the master controller via the RBX to the I2C Controller will be array writes – XX command = 11.

I.E. to read the I2C register of core 2(device 5, register#12), write address(0) in 6 LSB along with 11 in two MSB. The following bytes are stored in CCM internal address 0, 1, 2 and 3. The CCM returns a 2 byte message after the write – Node Number and Status. The I2C Controller will use the MSB of register 0 to determine a read or write operation and will take the 7 LSBs to enable the appropriate I2C Core. The I2C Core will perform a read of device 5 register 12. The returned data will be stored at CCM internal address 4. The I2C Controller will inform the CCM status register if an error occurred during transmission. CCM internal address 4 will be read out with the next command(read=01) along with the RBX node number and status. The RBX will return a 3 byte message – Node Number and Status followed by I2C data.

11 000000 ! Select CCM address 0 to do the array write to.
0000 0100 ! Write 4 as the size of the array to write.(The byte following an array write command always contains the numbers of bytes to write). The serial communications software does this automatically.
1000 0010 ! Write I2C Core #2 enable and MSB as a means to define R/W operation(Address 0).
0000 0001 ! Write number of reads or writes(Address 1).
x 00 0100 0 ! Write pointer address of I2C device(Address 2).
0000 1100 ! This value(12) eventually gets stored in the I2C device pointer register(Address 3).
! CCM responds with Node Status.
! The above number of commands will be the same for 1-12 reads.
! The I2C controller will get back a value from the selected I2C device which it puts in CCM address 4.
01 000100 ! Read command(01) of Address 4.
! RBX responds with Node Number, Status and selected address(4) content(I2C device 5, register 12's data).
! If multiple reads were needed this last command would need to be executed once for every read.

! The I2C Core and Controller can perform an array read, the RBX can only perform one read with each command(01).
! The number of reads the I2C Controller and Core should perform would be stored in Address 1.

To **write** an I2C device register – perform a write to the CCM address 0,1,2,3… write the I2C Core# and the write command into address 0, number of writes into address 1, the register pointer address into address 2, the register number into address 3 followed by the data into addresses 4-15.

I.E. to write the I2C register of core 4(device 2, register#19, value 3Bh)
11 000000 ! Array write command(11) to CCM address 0.
0000 0101 ! Write 5 as the size of the array to write. The serial communications software does this automatically.
0000 1000 ! Write I2C Core #4 enable and MSB as a means to define R/W operation(Address 0).
0000 0001 ! Write number of reads or writes(Address 1).
0 00 0001 0 ! Write pointer address of device(Address 2).
0001 0011 ! This register number(19) eventually gets stored in the I2C device pointer register(Address 3).
0011 1011 ! This data(3Bh) eventually gets stored in register 19 of I2C device #2, group 4(Address 4).
! RBX responds with Node number and Status.
! The number of commands will increase one for each number of I2C registers that are written. The number in address 1 would need to be set to the correct size and the number of data bytes sent would also increase.

FRAM to RBX communication

The FRAM will be loaded with values that are quickly downloaded to the registers within the ASICs(CCA, GOL) on the FE modules using the I2C format – this elimates the master from having to go in and write every register using the RS422 protocol everytime. The FRAM will be loaded with values by the operator via the RS422 communication protocol. To write the FRAM a 10 bit address will be needed to allow for an address space for all 28 registers of all CCA's and 6 registers of all the GOL's. The TTCrx will need 20 address spaces. There are 3 CCA(84) and 2 GOL(12) or 96 registers per FE board. There are 18 FE boards or 1728 registers plus 20 TTCrx registers or 1748 registers per RBX that will need to be downloaded when initiated by the operator.
! All registers will need to be downloaded, it looks like maybe 6 per CCA, 4 per GOL, and 4 for the TTCrx.

The FRAM is R/W device which is manipulated similar to the I2C devices. The maximum block size that can be R/W is 12 continuous locations(limited by the number of Data Registers(4-15) of the CCM).

To **read** the FRAM – perform a write command to the CCM internal address 0, 1, 2 and 3. Write: the FRAM enable(7 LSB's = 0, no core's are selected) and the read command(bit 8 = 1) into address 0, the number of reads in address 1, the lower order address of the FRAM into address 2 and the higher order address of the FRAM into address 3. The FRAM Controller will perform the read and store the results back into addresses 4-15(depending upon the number of reads). Follow the write command from the master with a read command(XX=01) of data in the register that is being pointed to by the channel selection register(address 4). The status of the communication will be forwarded to the RBX address 63 – status register.
! All writes by the master controller via the RBX to the FRAM will be array writes – XX command = 11.

I.E. to read the FRAM(address 750), write address(0) in 6 LSB along with 11 in two MSB. The following bytes are stored in CCM internal address 0, 1, 2 and 3. The RBX node selected returns a 2 byte message after the write – Node number and Status. The FRAM Controller will use the MSB of register 0 to determine a read or write operation. The returned data will be stored at CCM internal address 4. The FRAM Controller will inform the CCM status register if an error occurred during transmission. CCM internal address 4 will be read out with the next command(read=01). The RBX will return a 3 byte message – Node number, Status register and then followed by FRAM data.

11 000000 ! Array write command(11) to CCM address 0.
0000 0100 ! Write 4 as the size of the array to write.
1000 0000 ! Write FRAM enable and MSB as a means to define R/W operation(Address 0).
0000 0001 ! Write number of reads or writes(Address 1).
1110 1110 ! Write lower order address of FRAM(Address 2).
0000 0010 ! Write higher order address of FRAM(Address 3).
! CCM responds with Node number and Status.
! The above number of commands will be the same for 1-12 reads.
! The FRAM Controller will get back a value from the selected FRAM which it puts in CCM address 4.
01 000100 ! Read command(01) of Address 4.
! CCM responds with Node Number, Status and selected FRAM address(750) content).
! If multiple reads were needed this last command would need to be executed once for every read.
! The FRAM Controller can perform an array read(12), the RBX can only perform one read with each command(01).
! The number of reads the FRAM Controller should perform would be stored in Address 1.

To **write** to the FRAM, write address 0 in 6 LSB along with 11 in two MSB. Write the number of transfers followed by the FRAM enable(6 LSB=0) and the write command into address 0, the FRAM lower order address, the FRAM higher order address and the data. The RBX node selected returns a 2 byte message – Node number and Status. The FRAM Controller will perform a write based on the information in the CCM Internal addresses.

I.E. to write the FRAM address location 300 with data 73h.
11 000000 ! Array write command(11) to CCM address 0.
0000 0101 ! Write 5 as the size of the array to write.
0000 0000 ! Write FRAM enable(all 0's) and MSB as a means to define R/W operation(Address 0).
0000 0001 ! Write number of reads or writes(Address 1).
0011 0000 ! Write lower order address of FRAM(Address 2).
0000 0001 ! Write higher order address of FRAM (Address 3).
0111 0011 ! This data(73h) eventually gets stored in FRAM address 300(Address 4).
! CCM responds with Node Number and Status.
! The number of commands will increase one for each number of FRAM locations that are written. The number in address 1 would need to be set to the correct size and the number of data bytes sent would also increase.

The memory map for the FRAM devices is in the EXCEL file: CCM_FRAM_MEMORY_MAP.xls

FRAM Download Controller

On a CCM command the FRAM Download Controller will need to access the FRAM and deliver it's contents to the I2C devices. The FAST Download Controller will operate in a manner such that it downloads data to an I2C device one at a time. The FAST Download Controller will need to generate the device address, pointer address and data in a systematic method(State Machine). The state machine should be able to increment through the Cores, Boards, Devices and registers. The FRAM data is read out and put in the Internal Address Registers(0-15) and then sent to the appropriate I2C device.

A/D & MUX Controller

The A/D controller will continuously run by sequencing through the MUX channels and performing A/D conversions when the I2C registers or FRAM is not being accessed. It will stop while the A/D results are being read via the CCM. The conversions will be stored in CCM addresses 16-25. The conversion factor for the Temperature Sensor is 1.5037 Degrees Kelvin per LSB. i.e. if the reading from the A/D is C4h or 196dec, then the temperature is (196 * 1.5037 = 294.72 Kelvin), 294.72 – 273.15 = 21.6 degrees celcius. CCh=33 Celicius

The conversion for the voltage reading for V1 is 30mv per LSB and 25mv per LSB for V2. It appears that the reading are slightly less than what is expected.
! The RM# does not correspond to the CORE #. HB and HO use Core 1-5, HE uses Core 1-2, 4-6.
Address 16 = Temperature 1 Core1
Address 17 = Temperature 2 Core2
Address 18 = Temperature 3 Core3
Address 19 = Temperature 4 Core4
Address 20 = Temperature 5 Core5
Address 21 = Temperature 6 Core6
Address 22 = Temperature CCM
Address 23 = Voltage v1 – 30mv/LSB typ D8
Address 24 = Voltage v2 – 25mv/LSB typ C8
Address 25 = Voltage v2 – 25mv/LSB typ C8

QPLL loss of sync REGISTERS

Registers are 8 bit counts of the number of times the QPLL has lost sync.
A CCM_Reset or QPLL_Reset command to the Control register will reset the counters to 0.
Address 26 = QPLL2-A on clock board 3
Address 27 = QPLL2-B on clock board 4

STATUS and Control Registers

Address 61 = Broadcast register
Writable register that is used to enable broadcast commands from the TTCrx system. The broadcast bits from the TTCrx are labeled BRCST[5..2]. The TTCrx broadcast commands are user defineable, CMS HCAL is defining BRCST[2] as a QIE Reset and BRCST[4] as a FE PLL reset. If bit 0 of the Broadcast register is a "1" and the BRCST[2] from the TTCrx is a "1" a 25ns pulse will be generated and sent through a delay and then onto the backplane. Bit 2 of the Broadcast register is used to enable BRCST[4], this signal represents the FE_PLL_Reset signal. This signal is on as long as the bits are enabled. Bit 3 and 5 are not defined at this time.

Address 62 = Delay register
Writable register that is used to delay the QIE Reset command from the TTCrx system. This is an 8 bit register with each LSB representing a 25ns delay. Register value of 0 to 255(0 to 6.375us).

Address 63 = Status/Control register
The Status register contains bits that report the operation of the RBX/CCM. Any communication with the CCM will cause the CCM to respond with the contents of the status register. Writing to the Control register does not control the bits that are read back. The description of the 8 bits are shown below.

Status register readback
Bit 0: CCM Busy – FE board being written to or read.
Bit 1: unused
Bit 2: Clock to RBX backplane(FE boards) enabled
Bit 3: Error Reported in CCM communication to FE board.
Bit 4: RBX FE board power tripped – Side 1 of RBX
Bit 5: RBX FE board power tripped – Side 2 of RBX
Bit 6: RS422 communication Overflow error
Bit 7: RS422 communication Parity error

After every communication with the CCM the Status Register contents are sent back to the PC and the contents of the register are reset to 0.

Control register write

Bit 0: FE_Regulator RESET
Bit 1: FRAM Download
Bit 2: FE_QIE RESET
Bit 3: FE_Clock Enable
Bit 4: QPLL RESET
Bit 5: CCM RESET
Bit 6: FE_PLL RESET
Bit 7: FE RESET

What do these bits do:

Bit 0 – FE_Regulator RESET: This is a high going pulse that is 800us long. This signal should be sent when a FE power trip has been received(Bit 4 or 5 of the Status Register readback). This signal will reset the power regulators on the FE boards. If the signal is sent and the regulators are not in a tripped condition the reset is ignored. Bit is self clearing.

Bit 1 – FRAM Download: Writing a one to this bit will cause the data stored in the FRAM to be downloaded to the FE board registers. All FE board registers in the RBX will be overwritten with the data that is in the FRAM. The FRAM address and the FE board Register number can be found in the memory map on page XX. The FPGA sequentially retrieves data from the FRAM and writes that data to the appropriate FE board register. The function of this bit along with the FRAM is to allow for a download of stored register values to all FE board registers with one command. This could be done on a particular RBX or multiple RBX's at the same time. Bit is self clearing.

Bit 2 – FE_QIE RESET: When this bit is written to a 25ns high going pulse is sent to all FE Boards. Bit is self clearing. QIE_Reset pulse can also be generated using the TTCrx system. If broadcast register bit 0 is enabled and a specific broadcast command(BRCST2 is on) is received through the TTCrx the 25ns pulse is sent to the FE boards. The QIE_reset pulse can be delayed the value of clock ticks found in the Delay Register.

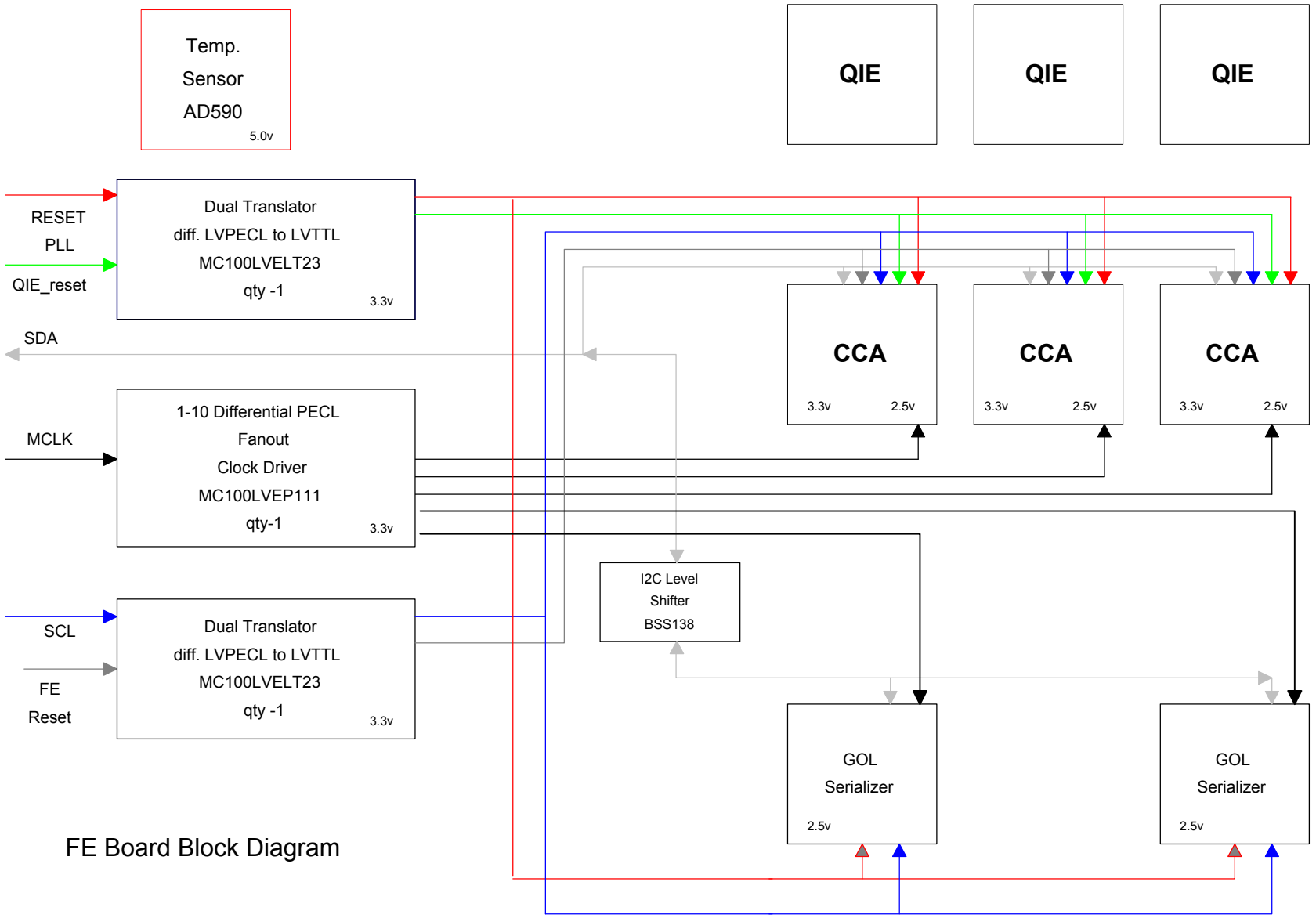
Bit 3 – FE_Clock Enable: By writing a one to bit 3 the Clock to RBX backplane(Fe boards) is turned on. The clock should be disabled during power up or when the FE Regulators are reset. This bit in not self clearing and will remain on until the Control register is written to again with bit 3 at zero. When writing to other bits of the Control register, Bit 3 should also be written to if you want the clock to remain on.

Bit 4 – QPLL RESET: Pulse to reset the QPLL asics. This will cause them to initiate a Frequecny calibration cycle and a new lock acquisition. This will also cause the QPLL Loss of Lock counters to be reset to zero(Register address 26 and 27). Bit is self clearing.

Bit 5- CCM RESET: Writing a one to this bit will reset all operations of the CCM including the UART(There will not be a response sent back to the PC). Bit is self clearing.

Bit 6 – FE_PLL RESET: When this bit is written a 10-18us low going pulse is sent to all FE Boards. This pulse connects to the FE boards GOL "Reset_b" pin and the CCA's "Reset_DLL" pin. Bit is self clearing. The PLL_Reset pulse can also be generated using the TTCrx system. If broadcast register bit 2 is enabled and a specific broadcast command(BRCST4 is on) is received through the TTCrx the PLL_Reset pulse is sent to the FE boards.

Bit 7 – FE RESET: When this bit is written a 10-18us low going pulse is sent to all FE Boards. This pulse connects to the FE boards CCA's "Reset_CCA" pin. Bit is self clearing.



FE Board Block Diagram