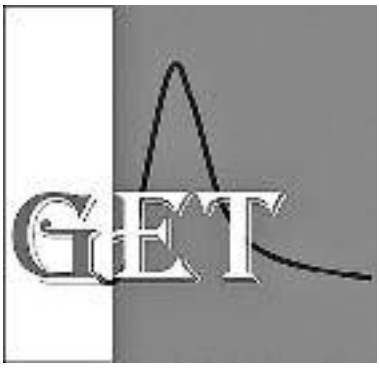

CoBo Module

Specifications

Draft 0.1

Nathan USHER

1. Introduction

This document specifies the design of the CoBo module of GET. The primary task of the CoBo is to readout the ASICs on the AsAd, compress the data, and forward it to the data acquisition system. This must be done with minimum dead time to allow the high event rates required by GET. Therefore, the CoBo firmware must be designed such that the data readout time is limited by the speed of the SCA/ADC sample clock, not the CoBo.

Other tasks that must be performed by the CoBo include:

- Clock distribution from MUTANT to AsAds
- Multiplicity from AsAd to MUTANT
- AsAd SCA read / write control
- Add time stamp to data
- Slow control pass-through to AsAds
- Monitor supply voltages and board temperature

Particular attention must be paid to the clock distribution paths and as a poor design could cause synchronization errors between data packets from different CoBos.

The CoBo is proposed to be a single-wide NIM module based around a Xilinx Virtex-5 FXT FPGA with an embedded PowerPC core (XC5VFX70T). If, during testing of the first prototype it is determined that more processing power is required, the design can be updated to use an FPGA with two PowerPC cores and more logic (SC5VFX100T). The two FPGAs are pin-compatible, so no board changes would be required to switch from one two the other.

The CoBo will also contain a separate FPGA, which will run an embedded Linux distribution on its embedded processor. Using a separate FPGA for slow control tasks will allow reuse of firmware and code from other detectors while simplifying debugging by dividing the high-speed and slow tasks into separate FPGAs.

2: Specifications

2.1 Standards

The CoBo module will be designed as a single-wide NIM module. The hardware will be identical, regardless of where each CoBo is placed within the GET system. Since the MUTANT is a double-wide NIM module, a maximum of 10 CoBos and 1 MUTANT will be placed in each NIM crate.

2.2 Multiplicity

The multiplicity from AsAd is transmitted via the same high-speed ADC used for data transmission. Therefore, the multiplicity is 12 bits per ASIC and received every 40 ns (25 MHz sampling rate). The sum of the multiplicity values from all 16 ASICs is kept added to a circular buffer which will serve as the sliding window requested by the trigger group (buffer size will be configurable by slow control). The sum of all elements in the sliding window will be scaled to 11 bits and transmitted to the MUTANT. The CoBo will add a 12th bit for parity as proposed in the MUTANT Specifications draft. As the connection to MUTANT only contains four bits for multiplicity, the multiplicity will be split and transmitted over three clock cycles.

2.3 SCA Readout

When the MUTANT triggers, it will send the "STOP_WRITE_SCA" signal. When this is received, it is forwarded to the ASICs, and the CoBo begins SCA, along with a "START_READ_SCA" signal, which initiates readout of the switched capacitor arrays. The CoBo also sends a "DEAD_TIME" signal back to the MUTANT until readout is completed and the CoBo is ready to begin sending multiplicity information again.

The CoBo firmware will be pipelined so that zero suppression and additional compression takes place simultaneously with SCA readout. As a result, the dead time during readout is limited by the maximum rate which the SCAs can be readout. The prototype firmware will implement zero suppression; additional compression algorithms will be added in future versions.

At the end of the compression pipeline, the data will be stored in a DDR2 SDRAM double buffer. Each half of the double buffer will contain two (Micron MT47H16M16BG-3) modules totaling 512 Mb. These modules support a maximum clock frequency of 333 MHz (21 Gb/s), and the recommended speed of the Xilinx memory interface is 266 MHz (17 Gb/s). In the worst case of full readout with no data being suppressed, this is enough to buffer 16 events. (This assumes 32-bit data: 9 bits for time bucket, 2 bits for AsAd, 2 bits for ASIC, 7 bits for pad, 12-bit SCA value.) Switching to a 14-bit ADC on AsAd as has been discussed would expand the data beyond 32-bits, which will require the addition of an additional SDRAM module to each half of the buffer to accommodate the larger data word size.

An embedded processor running Linux performs data transmission to InBo (Gbit Ethernet). Since a double buffer is used, there is no particular timing requirement between reading of the current event and transmission of previous events to from the CoBo. When the Linux processor has completed sending all buffered events in its memory and the compression pipeline finishes its current event, the buffers are swapped.

2.4 Slow Control

Slow control is handled by a separate FPGA than the data (Appendix 2). Breaking the design up this way allows easier debug of the individual firmware sections and should allow reuse of slow control firmware from another project. This FPGA must initialize the rest of the CoBo at startup and initialize all connected AsAds. It will also be set up for debugging as well as normal run control.

The slow control FPGA will monitor the power supply voltages and temperature of the CoBo board.

2.5 Links to Other GET Modules

This section describes the connections to AsAd and MUTANT modules.

2.6 AsAd Connection

The connections to AsAds are proposed to use two 3M-2x68 MDR connectors, and will connect to each AsAd via a 3-meter shielded twisted pair cable. The following are the proposed connections to each AsAd. These may change slightly or have other signals added as the AsAd design is finalized (pulser?, power?, AsAd ID?, temperature?, "Spy" mode?, etc.).

Differential pairs:
- ADC bit clock, frame clock, data (in, 10 pairs, LVDS)
- Multiplicity / ADC sample clock (out, LVPECL)
- SCA write clock (out, LVPECL)

Single-ended:
- ASIC and ADC slow control (11 wires, SPI)
- AsAd powerdown (out, CMOS 3.3V)
- SCA write enable (out, CMOS 3.3V)
- SCA read enable (out, CMOS 3.3V)

2.7 MUTANT Connection

The connection to MUTANT will use the stacked 2x15 points MDSM connectors also used on the MUTANT end of the connection (ITT-Cannon MDSM-30PE-Z10-VR22). This will connect via two shielded twisted pair cables. Each cable carries seven twisted pairs, and one single-ended signal.

The first cable is used for clock reception and transmission of multiplicity. The connections are as follows (all are differential pairs):

- MASTER_CLK (in)
- MULT_DATA[3:0] (out)
- MULT_FRAME (out)
- STOP_WRITE_SCA (in)

The other cable is used primarily for serial information reception. These connections also use differential pairs:

- DEAD_TIME (out)
- SERIAL_CLOCK (in)
- SERIAL_FRAME (in)
- SERIAL_DATA (in)

2.8 Ethernet

(This section will probably change depending on decisions about InBo / Ethernet switch.)

There are two designs being considered for the Ethernet link. The first design (Figure 1) has a single Gbit Ethernet connection to the slow-control FPGA. In this design, event data is transmitted from the Virtex-5 FPGA into the slow-control FPGA, which then transmits the data over the Ethernet link.

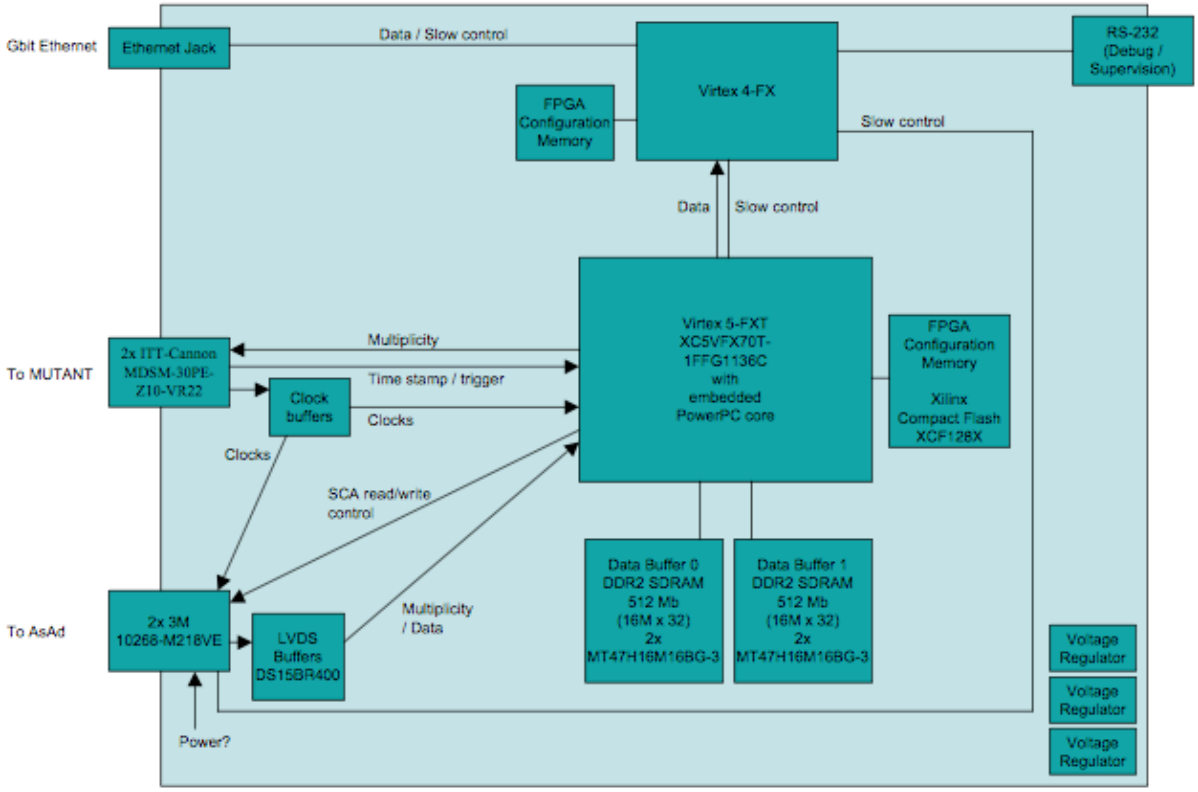

Figure 1: Data and slow control are both sent through a Virtex-4 FX FPGA

The other design (Figure 2) has two Ethernet links on the back of the module. The first link is the Gbit link for event data and is connected to the Virtex-5 FPGA. The other link is a 100 Mbit link that connects to the slow-control FPGA and is used only

for slow-control. Downgrading this link to 100 Mbit/s means that the slow-control processing could be done on a Xilinx Spartan-3 FPGA, instead of the Virtex-4 required by the other design, which would cut cost considerably. Spartan-3 FPGAs currently cost approximately $25 versus approximately $1,000 for a Virtex-4 FX.

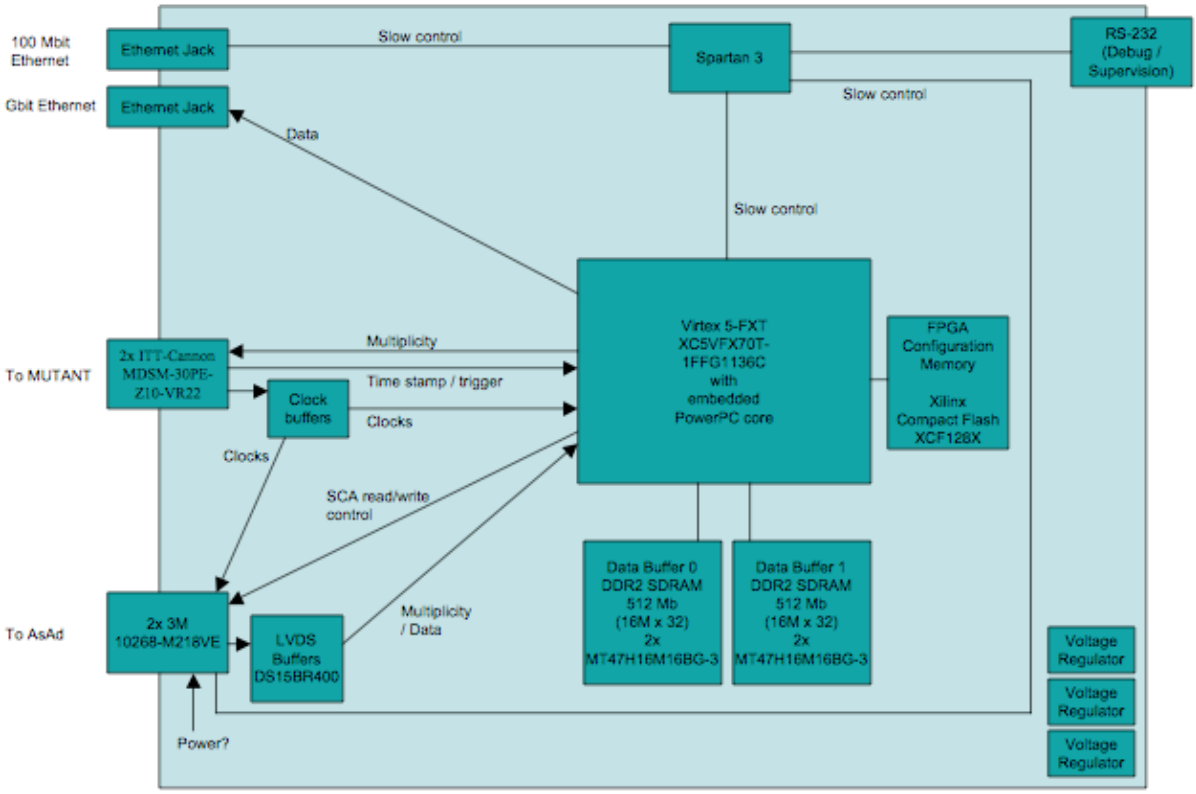

Figure 2: Data is transmitted directly from the main FPGA, allowing a much less expensive FPGA to be used for slow control

3: I/O Signals

The pinouts of the connectors to other GET components will be detailed here as they are finalized.

3.1 AsAd Inputs

To be defined …

3.2 AsAd Outputs

To be defined …

3.3 MUTANT Inputs

To be defined …

3.4 MUTANT Outputs

To be defined …

3.5 InBo / Ethernet

To be defined …

4: CoBo Registers

The registers will be detailed here as they are finalized.

4.1 General Configuration Registers (Slow Control)

4.2 Inspection / Diagnostic Registers