irfu

cea

saclay

Proposal for the GET System

Description of an acquisition system for TPC within the context of exotic nuclear study

Authors: Shebli Anvar, Frédéric Druillole, CEA Irfu

I. General Considerations

1 General description

1.1 Introduction

The GET project aims to develop new embedded electronics for several advanced projects of time projection chambers (TPC). Four laboratories are involved in GET:

- **C**entre d'**É**tudes **N**ucléaires de **B**ordeaux **G**radignan, with a special interest in the study of di-proton radioactivity,
- **G**rand **A**ccélérateur **N**ational d'**I**ons **L**ourds à Caen, with a special interest in ion beam facilities and associated instrumentation,
- **I**nstitut de **R**echerche sur les lois **F**ondamentales de l'**U**nivers, with a special interest in the study of exotic nuclei in active target experiments.
- **N**ational **S**uperconducting **C**yclotron **L**aboratory at **M**ichigan **S**tate **U**niversity, with a special interest in a new TPC for active target experiments.

The GET development is interesting for several others nuclear science communities such as the R3B/GLAD (Germany) or the TPC project at Riken (Japan). Research and development on TPCs worldwide seems equally interested by the GET acquisition system.

The **GET** system includes a new integrated circuit based on a development made for the TPC of the T2K experiment (Japan) and a set of embedded boards designed to meet the constraints expected within the context of the foreseen experiments, especially in terms of energy dynamic range, signal shape, detector channels multiplicity, elaboration of trigger and high data rates.

AGET consists of a 72-analog channels asic which integrates charge sensitive amplifiers (CSA) with adjustable gain, a shaper ranging from 0.1µs to 2µs and an analogue sampler with 511 capacitors to store analogue values. The asic has an integrated leading edge discriminator with programmable threshold for each channel. Each asic is read by a pipelined ADC at a frequency of 25MHz, the conversion being 12 bits wide.

The **AsAd** board contains 4 AGET asics and provides services for the control and diagnostics of the system.

The **CoBo** board is expected to extract and timestamp the digitized data of 4 AsAd boards. It will also implement some degree of processing on the extracted data (zero suppression, shape analysis…) before sending them to the general acquisition and data processing system. The CoBo boards are also expected to set the front end parameters and the channel calibrations.

Because of the sampling of the analogue channel, GET must address stringent constraints relating to the data acquisition back end system. Some serious data reduction must be applied to the whole chain of acquisition including the AGET boards in order to allow for the processing of **1khit/s** and even more for the 2p-TPC. Depending on the signal type in a TPC, algorithms can be very complex and CPU intensive. As a consequence, a selection mechanism of the most interesting events must be implemented through the use of a full trigger system, probably implemented in two levels: a very fast, hardware-based, level 1 trigger based on simple algorithms, completed by a software level 2 trigger implementing more complex filters over a computer farm. The level 1 trigger is foreseen to be implemented on the **MuTant** board.

In order to ensure a coherent time stamping of the data, an experiment-wide clock system is necessary. The **BEM** board is expected to implement this function.

Using the experience acquired in experiments such as T2K, Antares, MUST2 or MUSETT, this document tries to propose an architecture based on a global analysis of the different constraints and configurations that the GET system is expected to face. The aim of the document is to describe a complete solution that can be used as a basis for future discussions between institutes.

> **Complete information is available on the collaboration web site:**
> **http://groups.nscl.msu.edu/tpc/wiki/doku.php?id=electronics**

1.2 System description

Each AGET continuously produces an analogue signal proportional to the number of triggered channels. This signal is continuously digitized and readout by the CoBo board. A CoBo board must read out 16 asics, distributed on 4 AsAd cards. So a complete data set consists of **4 AsAd × 4 AGET × 72 channels × 511 cells× 12 bits**. A complete event has therefore a size of at least **6.8 Mbit**. The speed of the AsAd ADC is **25 Mword/s** where 1 word = 12 bits. So the complete data stream CoBo must stand is **4.8 Gbit/s** (16 AGET × 2 LVDS links @ 150 MHz). So the minimum time needed to completely readout 4 AsAd boards is 1.46 ms. An important consequence is that:

> **Without data reduction coming directly from the asic, the system cannot handle a data rate higher than 680 hits/s.**

The adopted solution is to integrate in the asic a level 0 trigger on each channel. It is the leading edge discriminator with a programmable threshold. A second point to reduce the data flow is to control and detect rare events by a level 1 trigger system. It is the role of the MuTant and BEM cards to select interesting events and the channels that need to be readout for these in the CoBo board. In other word, **the acquisition in the CoBo cards must be driven by the MuTant system**.

Once an event slice has been acquired by a CoBo board, it still has to undergo some compression before it can be transmitted over the GET network and processed by higher level software processes. In this document, we propose to achieve this data reduction within the

CoBo board itself through the use high performance algorithms such as zero suppression or shape analysis. If enough reduction is applied, data transmission can be achieved using a standard Gigabit Ethernet switched network.

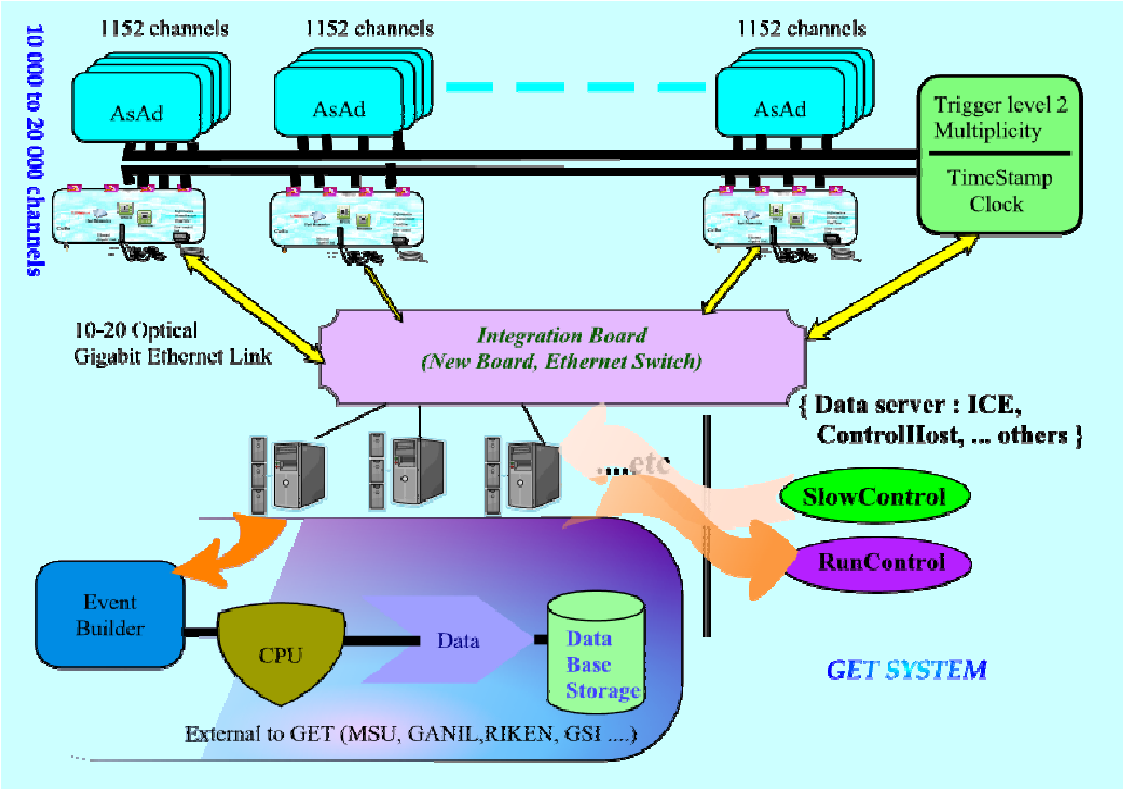


Figure 1: Complete view of the GET system and its different parts. Several modules, consisting of four AsAd board and one CoBo, read out TPC pads and send data to the shore after validation by multiplicity processing system.

A circular buffer within each CoBo using high performance memory will store the data coming from the AsAd boards. This data is organized into time stamped frames, to be reduced by a specialized, probably firmware, CoBo process. This reduced data flow will then be sent over a Gigabit Ethernet link, using a high performance TCP/IP stack running over an embedded RISC processor. The timing information attached to each frame will define a time slice (containing the data relating to a fixed number of events). This time slice will be used as a routing information to send each frame to one of the nodes of a computer farm attached to the switch so that, eventually, only one computer contains all the data pertaining to a single event. In this way, complex level 2 trigger algorithms (using a global snapshot of the 36864 channels of the detector) can be implemented in software over the computer farm in order to ensure an efficient filtering of the data before storage. The filtering algorithms can be based on advanced event analysis using, for instance, track reconstruction and full calibration.

The filtered data is then managed by a data server which stores the data and answers to data requests posted by external systems. Some of these external systems can be processes that consolidate data produced by different detectors, in which case the GET system would appear as a sub-detector within a larger framework. External system are also all the DAQ systems of the different accelerator control rooms.

1.3 DAQ Time Slice principle

The time slice based method of acquisition comes from particle physics experiments which need the online processing of high data flows for purposes of filtering and data reduction. This situation begins to arise with the GET system because 36864 detection channels is a high figure calling for a scalable architecture. Figure 2 illustrates this scheme in which event data are routed to nodes in a computer farm based on the event timestamp.

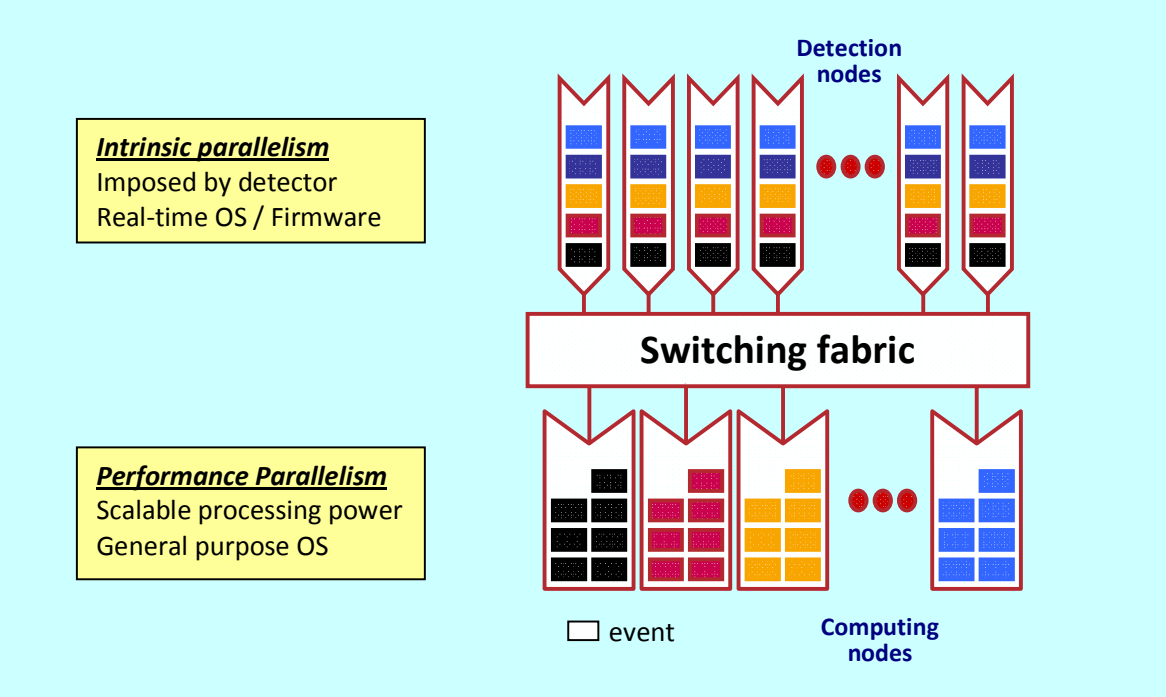


Figure 2: scheme of the difference between intrinsic parallelism and parallelism of performances

2 Scenarios of several trigger modes

As AGET gives continuously the number of triggered channels, several readout modes are possible when a threshold has been reached:

- Readout of the whole set of channels.
- Readout of the register of triggered channels followed by readout of selected channels.
- Readout of triggered channels only.

We need to elaborate different scenarios for data acquisition depending on CoBo and Mutant architecture. The following will describe two main scenarios and chapter 2 will explain one solution meeting the requirement.

2.1 Standalone CoBo

As the first development step is to test AGET and AsAd boards, we need CoBo to be able to run without any need for a MuTant board. This will allow for the standalone test of a complete CoBo+4 AsAd module. The way to do that is to emulate a “MuTant trigger” inside CoBo (See Figure 3).

2.2 MuTant driven CoBo

This scenario calls for a synchronous communication between all CoBo cards and the MuTant/BEM subsystem. The constraint is to minimize the number of connectors and cables to simplify the integration of the GET system. The proposed architecture allows for up to 32 CoBo boards in the system. Each CoBo sends the information concerning the multiplicity of 16 AGET to the MuTant board; then MuTant processes the level 1 trigger. If the outcome of the trigger is positive, MuTant replies with a trigger decision validating the current event (basically a time window). MuTant transmits at the same time the event identification and its associated time. Each CoBo reads out the triggered channels and stores them in its high performance memory. It remains for CoBo to send the event frame over the Gigabit link after some data reduction.

2.3 Acquisition with selective channel readout

A way to validate the data of each AGET channel is to select the channel to be readout. In this case, MuTant must send a list of channels to be read out for each dedicated CoBo. Then CoBo reads out and stores the corresponding data in its memory. The mode is a slower one because of the more complex communications between MuTant and the CoBo boards, involving the definition of a time sequence for each communication (See Figure 3).

Also, we must consider that BEM provides a 100MHz clock to the whole set of cards (CoBo and Mutant) with a precision better than **1ns rms.** Commands are send by BEM to start a run and then MuTant could control the acquisition through CoBo. This means that MuTant is closely attached to the BEM, e.g. a daughter board of BEM. A protocol must be defined with a fast timing to consider trigger policy.

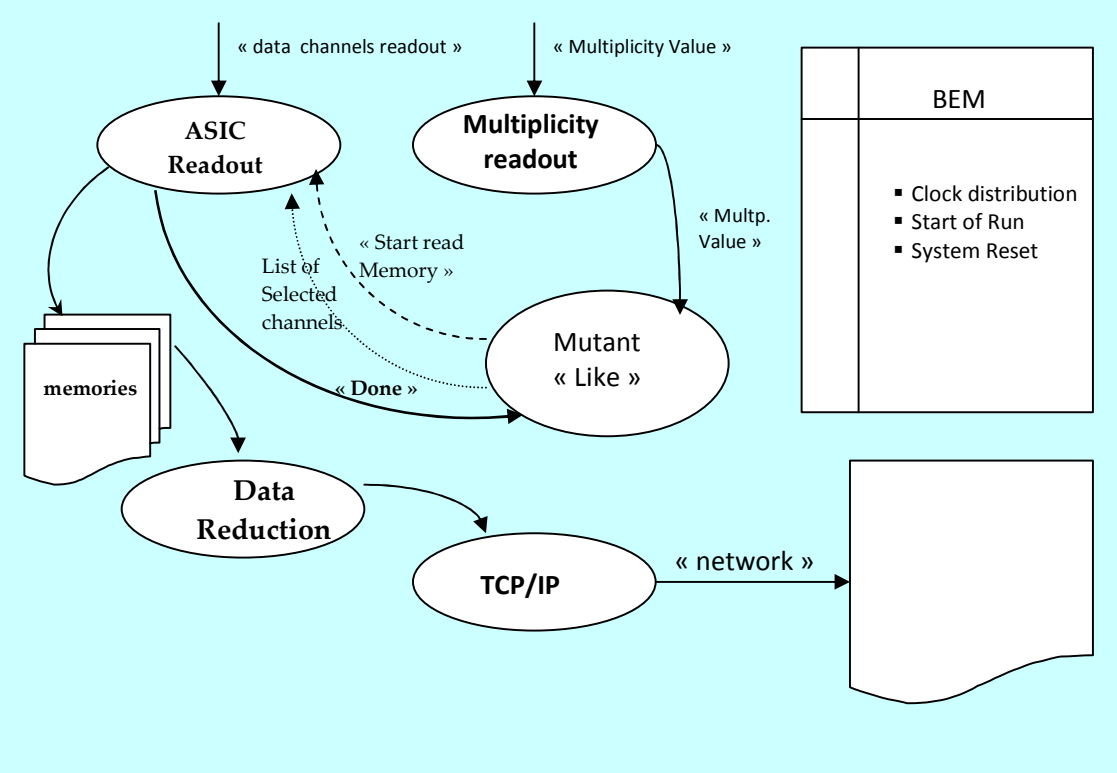


Figure 3: Communication between CoBo, Mutant and BEM are presented in the diagram. A protocol needs to be implemented to do all described scenarios.

2.4 Timing considerations

The frequency of the ADC is fixed at 25 MHz, so 40 ns. It means that CoBo has a view of level 0 trigger every 40ns. If MuTant needs information about triggered channels identification, CoBo will spend 72x40ns = 2.88µs to read the register. The time window for MuTant to take the decision depends on the sampling frequency:

Sampling Time period (ns)	Number of considered cells	Level 1 Trigger time window
10	511	5.11 µs
10	256	2.56 µs
10	128	1.28 µs
20	511	10.22 µs
20	256	5.11 µs
20	128	2.56µs
30	511	15.33µs
30	256	7.698 µs
30	128	3.84 µs
40	511	20.44 µs
40	256	10.22 µs
40	128	5.11 µs
50	511	25.55 µs
50	256	12.775 µs
50	128	6.387 µs
100	511	51.11 µs
100	256	25.6 µs
100	128	12.8 µs
500	511	255.5 µs

Tableau 1: Table of level 1 trigger time window to take the decision depending of sampling period and number of cells the user wants to read.

Nevertheless, taking the worst case, MuTant needs to take the decision to validate the event in less than 5µs (considering, receiving information, processing it and replying to all CoBos). 5µs is a complete analogue memory cycle of the incoming signal.

Considering the 5µs, it allows MuTant to ask for the triggered channels register after having taken the decision to validate the event. A way to gain time is for CoBo to continuously read the triggered channels register and store it in case MuTant asks for it. Therefore, a timing for the trigger decision could be :

0.5 µs to retrieve all multiplicity information, to take the decision and to reply to all CoBo in classical mode. Or 0.26µs only to check the trigger value of each CoBo and wait for a later decision.

2.88 µs to store the triggered channels register in the CoBo

If needed, sending triggered channels identification information and time stamp to MuTant in 320 ns.

Sending the decision signal in 240 ns.

So the trigger decision could be done in less than 3.5 µs.

Considering the readout of the data from each channel, the ADC stream has a rate of 4.8 Gbit/s. So depending of the number of cells and the number of channels, the time to store data into the CoBo memory is:

Number of channels	Number of cells	Time of read
79	511	1.62 ms
72	511	1.472 ms
72	256	0.736 ms
72	128	0.368 ms
10	511	0.21 ms
20	256	0.21 ms
30	128	0.154 ms

Tableau 2: examples of time spend reading a part of the analog memory inside AGet in AsAd board.

We can clearly see that if we want to have enough time to send the data to the PC through the Ethernet switch, we need to choose as a classic case to read 72 channels with 128 cells. With that, CoBo stores the corresponding data in less than 400µs and we have 550µs to send the data over the network.

3 Conclusion

The GET system must have a dedicated fast protocol between CoBo and MuTant-BEM, minimizing the number of cable and connector. To reach the required data rate, we must require a trigger decision taken in less than 3.5 µs and a readout data from AGET less than 400 µs.

II. System Architecture

1 Front end communication

1.1 Hardware description

To minimize the number of cables and also to define a high speed protocol especially for trigger decision, we considered that a solution based on a combination of parallel and daisy-chain architecture could be a good compromise between speed and hardware integration. Figure 4 shows an implementation of links between CoBo Boards and MuTant-BEM. We consider 8 "CoBo" connectors consisting of 13 differential pairs of cables (you can find cables with 15 differential pairs on the market). 8 CoBos are directly controlled by the MuTant: These are the CoBos of rank 0 (first rank). Then each first rank CoBo sends the control stream coming from MuTant to the next rank. The delay will be only one or two clock cycle per hop over a rank which is negligible and could be calibrated by slow control.

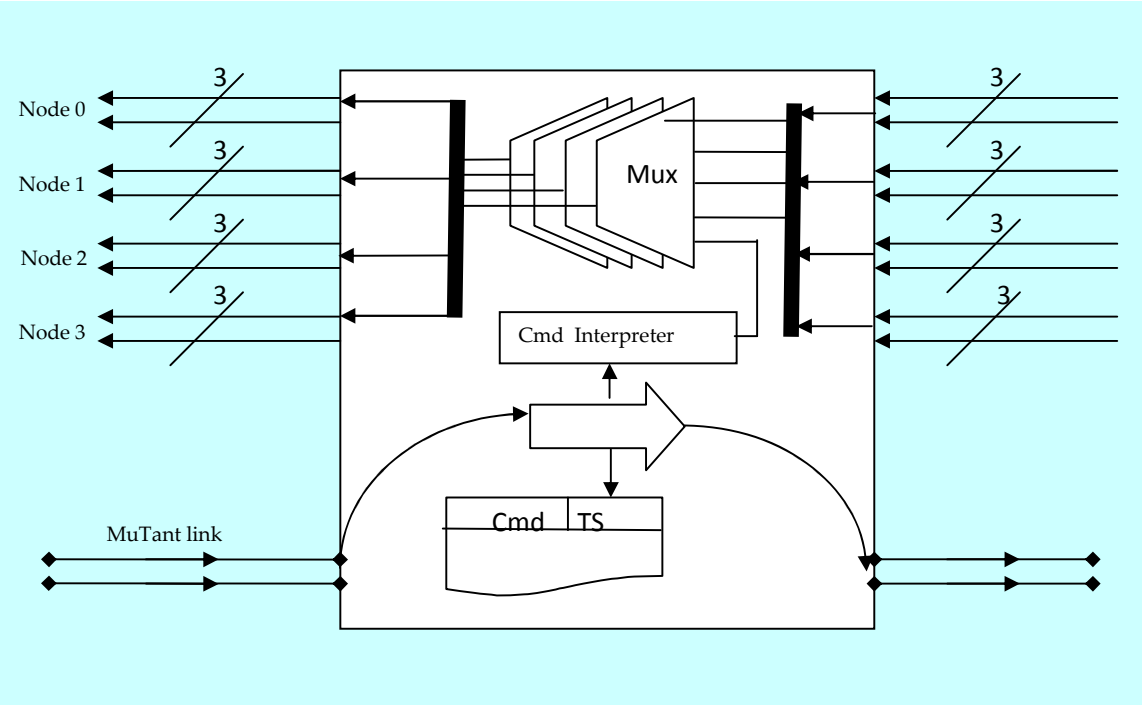


Figure 4: scheme of the daisy-chain architecture of the CoBo Board.

In figures 4 and 5, we consider how to communicate between CoBo cards. Each CoBo has one differential pair going from MuTant to CoBo and 12 differential pairs going from CoBo to MuTant. Each set of three pairs, called CoBo node, belongs to one CoBo rank. Therefore, up to 4 ranks can be addressed. One node helps CoBo to communicate with MuTant, the other nodes of the same CoBo board are just pass-through links for the higher rank CoBos. As all CoBo boards must be interchangeable (they are identical), a set of multiplexers must be used to select the function of each node. The command sent by the MuTant specifies a rank so that every CoBo on a command daisy chain can be specifically addressed. All commands are systematically transferred on the MuTant link output to the CoBo board of the next rank.

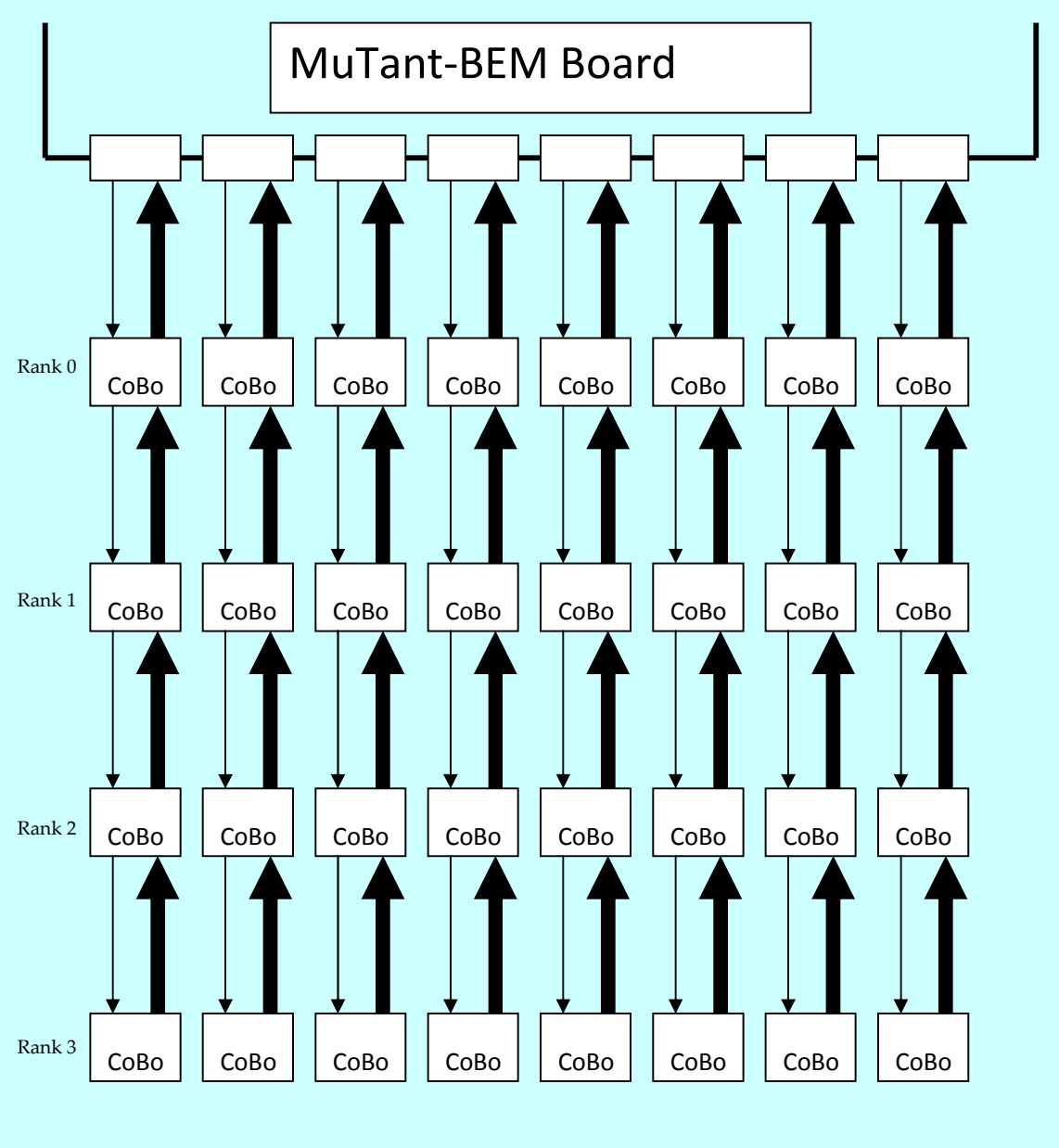


Figure 5: MuTant and CoBo boards connection for fast communication. Up to 32 CoBo boards can be connected to the system. Each CoBo is driven by MuTant commands.

We considered using a node of 3 differential pairs in the communication from CoBo boards to MuTant to reduce communication latency, because the latency budget is tight, being determined by the front end circular buffers. With a single link, the number of transmitted bits would make it impossible to keep the timing constraint.

1.2 Protocol

We need to transmit an order in the same time as the global clock. Also, for precision purposes, we need to calibrate clock delays for the whole system. So, why not multiplex clock and data on the same link? We can reduce the number of cables and measure the delay between the incoming clock from MuTant to CoBo and the retransmitted clock from CoBo to MuTant. The

main clock is 100MHz to obtain a 10ns time precision. The communication stream is showed in Figure 6. Concerning the command, we need a start bit at one to be able to retrieve the command from the bit stream.

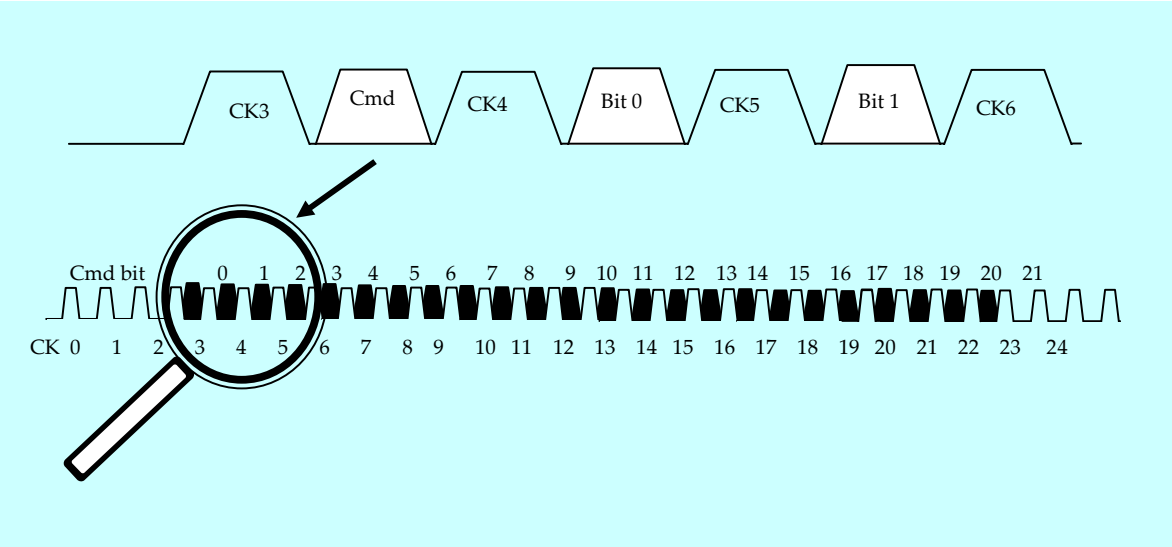


Figure 6: Stream of the communication from MuTant to CoBo. The internal clock frequency is 100MHz and a start bit is necessary to analyze the command syntax. A typical command will take 24 bits.

The implementation of such a protocol is easily done on a Xilinx Virtex FPGA because of the Dual data rate function in ILOGIC blocks. (see note below)

Input DDR Overview (IDDR)

Virtex-5 devices have dedicated registers in the ILOGIC to implement input double-data rate

(DDR) registers. This feature is used by instantiating the IDDR primitive.

There is only one clock input to the IDDR primitive. Falling edge data is clocked by a

locally inverted version of the input clock. All clocks feeding into the I/O tile are fully

multiplexed, i.e., there is no clock sharing between ILOGIC or OLOGIC blocks. The IDDR

primitive supports the following modes of operation:

• OPPOSITE_EDGE mode

• SAME_EDGE mode

• SAME_EDGE_PIPELINED mode

The SAME_EDGE and SAME_EDGE_PIPELINED modes are new for the Virtex-5

architecture. These new modes allow designers to transfer falling edge data to the rising

edge domain within the ILOGIC block, saving CLB and clock resources, and increasing

performance. These modes are implemented using the DDR_CLK_EDGE attribute.

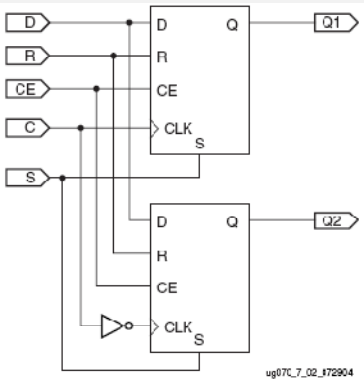
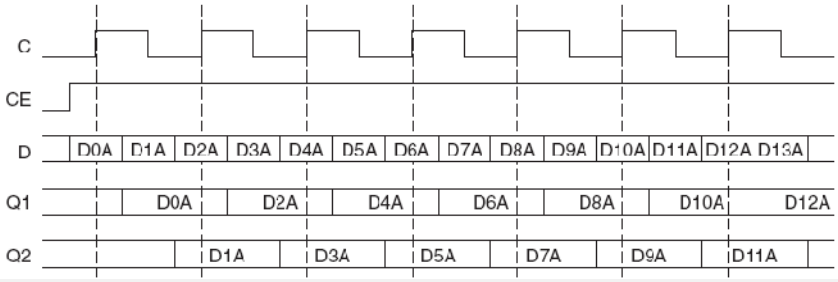
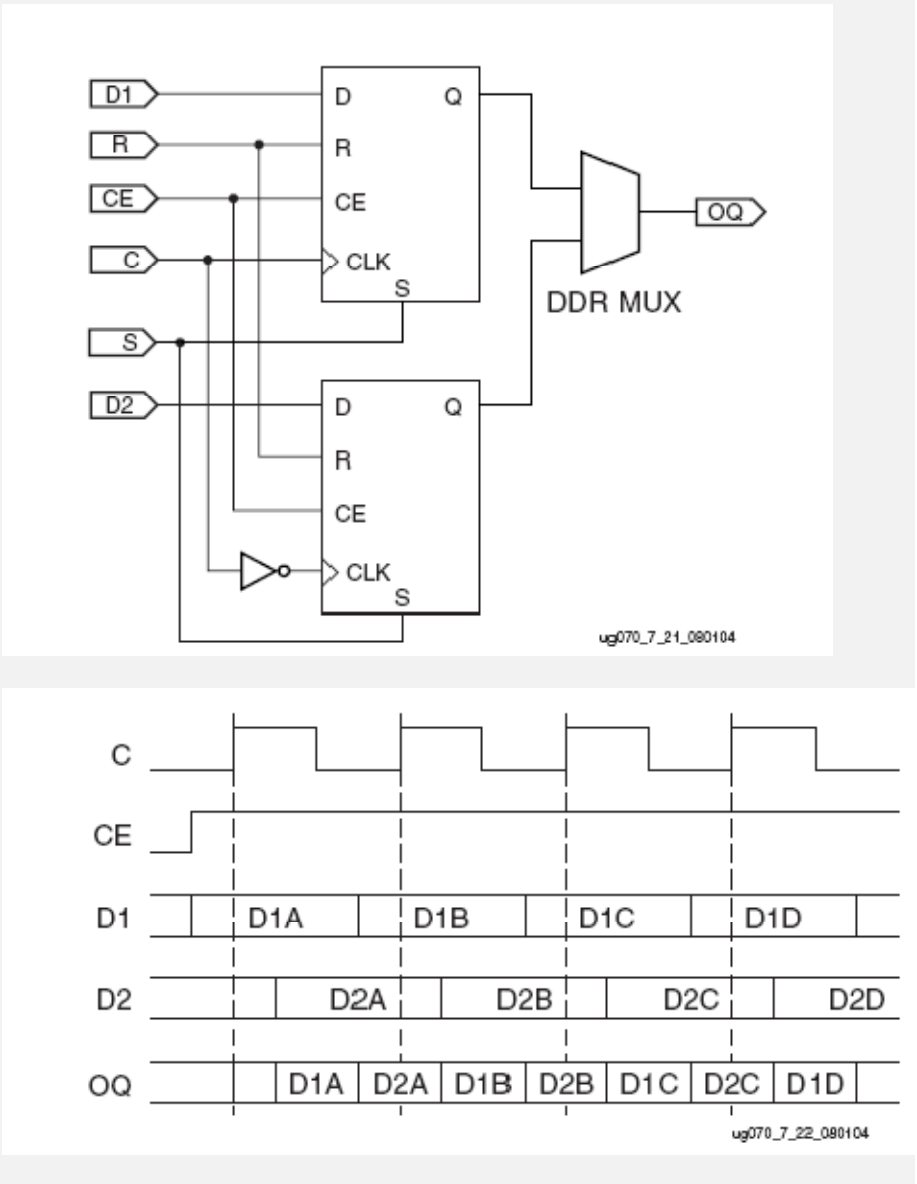


Figure 7-2: **Input DDR in OPPOSITE_EDGE Mode**



The same opposite function for output mode could be implemented in the virtex using ODDR.



Each command will be as long as 24 bits, consisting of a command value on 8 bits and a parameter register on 16 bits. To reduce lantency, the transmission from CoBo to MuTant will be done over 3 links at 100MHz. So the command will be divided in three. Like this, we can check the multiplicity value in 80 ns instead of 240ns in one link.

1.3 Command specification

Each CoBo copies the command on its output MuTant link to send information on the next rank.

Start of Run Command : it allows starting at the same time the writing of the analog memory of all asics of the GET system.

Command value = 0x81

Parameter : Start 0x0001 Stop 0x0000

Time Stamp Command : MuTant sends a value on 12 bit at 10 ns on each CoBo. CoBo has an internal counter synchronous of the global system to add the 30 higher bits of the MuTant time stamp value. Therefore, a 42 bit Time stamp value is created for a valid event, which allows a run time of 12 hours.

Command value = 0x82

Parameter : Time Stamp 0x0A1

Selected channel Command : MuTant could select what channel on what asic on what CoBo it want to get. We use special parameter which could be use for others command with the same type of action.

Command value = 0x83

Parameter : CoBo Rank : 5 bits

Asic number : 4 asic

Channel number: 7 bits

Multiplicity value Command : Each CoBo must give the multiplicity value coming from the 16 asics to MuTant board. Two solution exists. One Cobo send the addition of all the multiplicity value on 16 bits parameters. So Mutant gets the information each 80 ns but looses the information about the individual asic multiplicity value. Another step is to code on 4 bits asic number and then gives the multiplicity value of the cited asic. The time to read all multiplicity value will be 80 ns by 16 = 1.28µs.

Command value = 0x84

Parameter : Multiplicity value : 0xFFF0 (max)

OR asic number 0x1 Multiplicity value : 0xaaa

2 CoBo Features

2.1 Hardware description

2.2 Data Transfer

3 Time slice implementation