The CMS DT Muon DDU:

a PMC based interface between frontend and data-acquisition.

Benotto, F. Bertolino, F. Cirio,R. Dellacasa,G.

INFN, Via Giuria 1 I-10125 Torino, Italy
cirio@to.infn.it

Abstract

CMS will use gas Drift Tubes (DT) as active part of the barrel muon sub-detector. In total 200.000 wires will be readout by TDCs and signals will be sent to Data Acquisition System (DAS). The entrance door to the standard CMS DAS will be a board (Detector Dependent Unit - DDU) that will be specific to each sub-detector. We have built a PCI Mezzanine Card (PMC) based prototype of the DT muon DDU that features two input channels with Optolink, data check and reconstruction by FPGA and PCI slave output through a FIFO. A description of the board and the FPGA schematics will be given together with the lab setup used to test and debug it.

I. Introduction.

The recognition and measurement of muons in the barrel part of the CMS detector is performed combining drift chambers with resistive plate chambers [1]. The drift chambers detector (DT) consists of a set of 48 concentric layers of drift tubes arranged in 4 stations, each with 12 wire planes. This part of detector allows first level triggering and measurement of the momentum of the incoming muons. The resistive plate chambers, arranged in four layers as for the DTs, are used as an independent trigger [2].

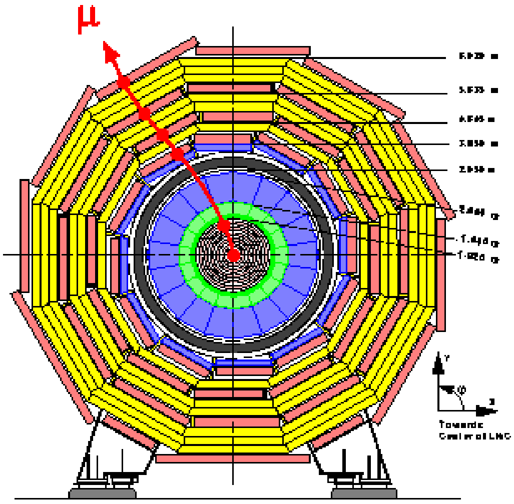

Figure 1: Transverse view of the CMS detector

With this system it will be possible to measure the momentum of a muon with 10 GeV < p_T < 1 TeV, $|\eta|<1$ with a relative error of 0.5 – 7%.

The DT part of the detector features a total of 200.000 wires, each connected to a VLSI TDC located at the chamber's end. The TDC provides a measurement of the time of arrival of the charges on the wire with a resolution of 12 bits. Once data are digitised, a set of boards mounted on the chambers' ends gathers them [3]; data are first assembled at the chamber level (each chamber has 12 layers of wires) by the Readout Boards (ROBs), then at the sector level (CMS is divided in 12 ϕ and 5 R sectors, adding up to 60 of them) by the Readout Server (ROS). Figure 1 shows the transverse segmentation of CMS; pink rectangles are the 4 layers of barrel muon chambers.

Only data that pass the first level trigger are sent from the detector frontend buffers to the DAS. The CMS DAS [4] is organized in 3 layers, as it can be seen in Figure 2: a Readout System that receives data from the frontend, an Event Builder that reconstructs the data fragments and a Filter System where all high level trigger algorithms are run. From the Readout System onwards, all the hardware is no longer custom for the various sub-detectors. The last custom part is an interface board, located at the very entrance of the Readout System, that allows reception of data from the frontends and handshaking with the standard DAS: such a board is called Detector Dependent Unit (DDU).

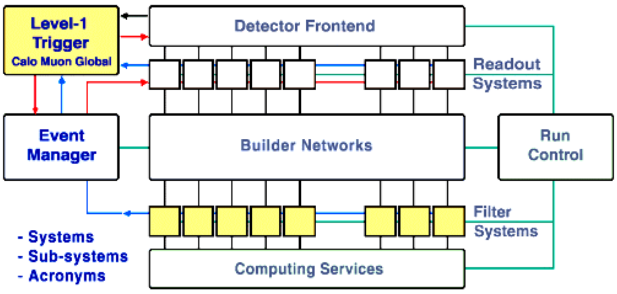

Figure 2: Block diagram of the CMS data acquisition system

II. Drift Tubes DDU functionalities.

The Detector Dependent Unit is the last custom module that will be used in the CMS data acquisition system. It has to ensure the correct data flow between frontend and event builder and it has to offer the possibility of reading out data in an independent way;

this last feature will be useful during the setup phase of the detector and, once the whole detector will be run, during periods when the general DAS will be not available. To summarize, the following are the services that the DDU has to implement:

- receive data from frontend and check consistency;
- store data in a memory readable from DAS;
- store data in a memory readable from a local CPU;
- look for errors and eventually take appropriate actions.

Data from the frontend are sent to the DDU via optical serial links; each of the 60 sectors in which CMS is divided leads one link. It is foreseen to have an occupancy of 1 track per sector per event. Each track has 48 space points for a total of 144 bytes plus some 22 overhead (header/trailer,…) bytes. The CMS DAS, at the DDU level, asks for a 2 kBytes event: this implies that the maximum number of channels per DDU for the drift tubes detector will be 2k/166≈12. If one builds boards with 12 channels each, 5 boards will be needed for the whole DT. The number of channels per board will be decided once the exact components will be available, allowing a correct estimate of the space occupied; cost matters will also be considered. The actual guess is to use VME standard; not yet decided is the dimension 9U or 6U. With any of the choices, the whole setup needed for DTs will be housed in a single crate.

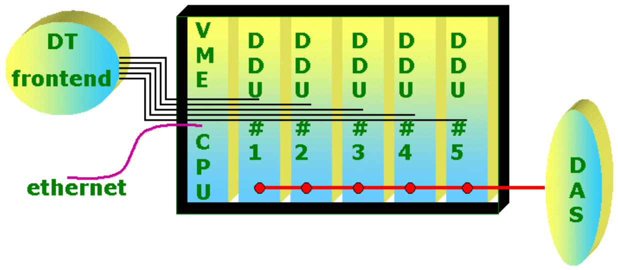

Figure 3: The muon drift tubes DDU crate

The DDUs will be responsible for giving the chance of having an acquisitions system independent from the standard CMS DAS. The DDU crate will house a CPU that, using the VME bus, will allow readout of data in parallel with CMS DAS. This feature will be used for monitoring during standard operation (spying) or as unique readout when the CMS DAS will be unavailable or not yet mounted (during detector assembly in the ground hall, for example). Data read by the CPU can be locally treated and results will be available to any concerned user via a link (say Ethernet).

The DT frontend gathers data from the various chambers within a sector and sends them to the DDU as a sequence of 9-bit words. DDUs will receive them via a serial optical link. The first DDU duty is to check that transmission and data consistency are correct. This is done using the error code of the link and checking that the header-data-trailer sequence is correct. The event is then re-formatted: 9-bit words are assembled into 32-bit words, new headers/trailers are substituted to the previous ones. At this stage it can be eventually performed a reordering of data words, to speed up high level triggers decision times; this has to be very carefully evaluated because purely hardware operations can be very difficult to be implemented in FPGAs.

Formatted data are stored in a memory that can be accessed both by DAS and by local readout. The memory has to be 32-bit wide and should have a depth that allows the storage of some hundred events. The memory should be readout in a FIFO-like mode using the fastest speed available by the output transfer bus. This reflects, for the actual prototype, in using a FIFO that can be readout with DMA in PCI.

The DDU has to be ready to react in case any error condition is reached both within the DDU itself and/or in the frontend electronics. Error messages are contained in the data received from the frontend; they are decoded and checked in the first actions taken by DDU. These errors include buffer-full, readout-error, chamber-malfunctioning. The locally generated errors include buffer-full, transmission-error, timeout-in-data-reception. The actions that DDU takes are twofold:

- include error code in data for further *slow* action;
- signal error with an interrupt for immediate *fast* action.

All the design of the DDU has to take into account the speed of data arrival: it is foreseen that on average each sector will send its data (166 Bytes) every 10μs, speed of the Level_1_accept (trigger). This adds up to a data throughput of 64 Mbits/s as DDU input. The output speed (essentially the speed needed for the output memory) has to be compatible with the data transfer rate allowed by data bus. The actual rate is 66 MHz obtainable with PCI.

III. PROTOTYPES AND LAB TEST SETUP.

The development of the DDU and its related hardware and software is being done on a several years span. The frontend is developed and built in the laboratories of Padova and Madrid, while the DAS (in particular the RUI) is being built at CERN. To allow debugging and test of the system in a standalone mode, we have built and assembled a number of boards that add up to a complete readout. The heart of the lab test setup is a VME CPU.

The data coming from the frontend are simulated with a PMC board nicknamed PTT. Two versions of the DDU have been built so far: a VME board housing a single input channel and a PMC board with two input channels. When using the PMC version of the DDU, the PCI bus of the CPU is expanded with Motorola PMC span boards. Figure 4 shows the lab setup that is used with the PMC version of the DDU.

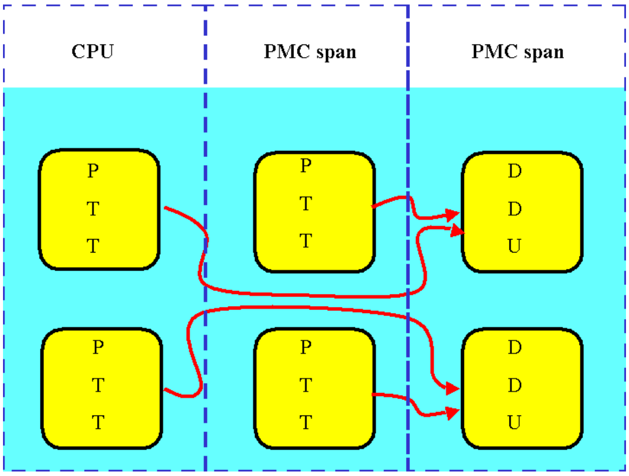

Figure 4: The laboratory test setup block diagram

A. *Frontend electronics simulator (PTT).*

Time of arrival is digitised with TDCs located at wire ends. Data are then grouped at the chamber and at the sector level to be sent to the DDUs. As data are sent via 8-bit optical serial links, the format is structured in corresponding way. The data format for an event is listed in Tables 1a and 1b.

Table 1a: Frontend data format for an empty event

Without data	**N.of bytes**	**Type**
Start ROS	1	Control
Event number TTC	3	Data
End	1	Control
Data (0hits)	**0**	
Overhead	**5**	
Total	**5**	

Table 1b: Frontend data format for a typical event

With data	**N.of bytes**	**Type**
Start ROS	1	Control
Event number TTC	3	Data
Format=Data 3 –byte	1	Data
Start ROB	1	Control
ROB address	1	Data
Datum 1 [23:16]	1	Data
Datum 1 [15:8]	1	Data
Datum 1 [7:0]	1	Data
…		
Datum n	3	Data
Start ROB	1	Control
ROB address	1	Data
Datum 1	3	Data
…		
Datum n	3	Data
Start ROB	1	Control
ROB address	1	Data
Datum 1	3	Data
…		
Datum n	3	Data
Start ROB	1	Control
ROB address	1	Data
Datum 1	3	Data
…		
Datum n	1	Data
Start ROB	1	Control
ROB address	1	Data
Datum 1	3	Data
…		
Datum n	1	Data
End	1	Control
Status	1	Data
End Data	1	Control
Data (48 hits)	**144**	
Overhead	**22**	
Total	**166**	

The PTT is a simple PMC board that houses a PCI bridge PLX 9050, a 2k*9-bit FIFO IDT72231-15-J and a serial link Cypress Hotlink CY7B923-JC run at 20 MHz. The FIFO can be written and read from PCI. This operation allows checking the correct behaviour of the board. Once data are being loaded on the FIFO, it can be started an asynchronous transfer toward the DDU via the serial link with a PCI command. The PPT has been used to test both the VME and the PMC versions of the DDU.

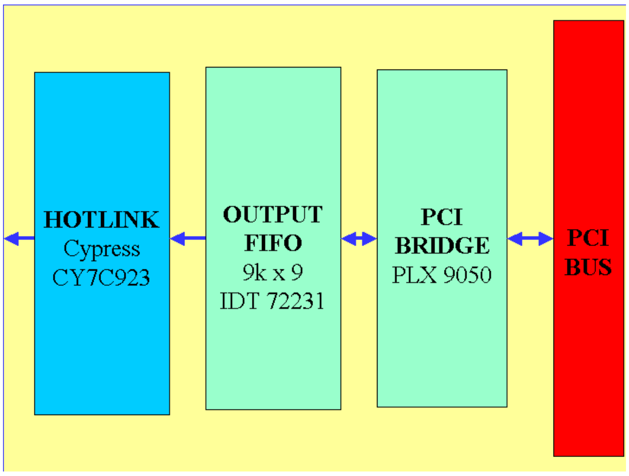

Figure 5: Block diagram of the PTT PMC board

A description of the board can be found in [5].

B. VME version of the DDU prototype.

The first version of the drift tubes DDU has been realized implementing one input channel on a 6U VME board. The reason for such a choice was uniquely practical and related to the need of learning the FPGA programming and the PCI bridges use on two separate boards: it has been chosen to learn the former using the well known VME bus. The data are input on a Cypress Hotlink CY7B933, stored on a 2k*9-bit IDT72231 input FIFO. The Xilinx XC4005EPG156 is used to transform the 9-bit words in 32-bit output words. Data are then stored on two 256*18-bit CY7C4205 output FIFOs that can be readout in slave mode via VME. Both input and output FIFOs can be written by VME for debugging. The frequency of the Xilinx clock is 4 MHz and the frequency of the Hotlink clock is 20 MHz. The board has been successfully tested in a lab test bench that included a ROB prototype built in CIEMAT Madrid. A description of the board can be found in [6].

C. PMC version of the DDU prototype.

During 2000 it has been built a PMC prototype of the DDU. It has been decided to have two channels on the same board, to allow complete testing of the input protocol that the FPGA has to handle. Data can be received on an optical fibre by a Hewlett Packard HFBR5208 or on the electric input of the Cypress Hotlink CY7B933-SC, used at the frequency of 20 MHz. Then each channel stores them on a 2k*9-bit FIFO. At this stage data are checked for transmission errors (parity, interrupted link) and for the correct sequence *header-trailer*. As transmission is asynchronous, it is implemented a 200 μs timeout protocol that forbids DDU from being in an infinite waiting loop. Data are then merged in the two FPGAs Xilinx XCS40XL-4PQ240c. The appropriate header is written, error conditions from the frontend are checked and then, if no error is present, after having written all data from the first input-FIFO that showed data, the reading procedure passes to the second input-FIFO. If no errors are found, the event is closed with the appropriate trailer and the DDU is set again in a waiting state. If errors are found, the error code is written in the status word of the event and, if it is enabled, an interrupt is sent to the control CPU. Table 2 contains the format of the data as they are written on the output memory.

Table 2: Format of data at the DDU output

HEADER
Event Number
Data 1 - ROS 1
Data 2 - ROS 1
……………
Data n - ROS 2
STATUS
Word count
TRAILER

The DDU has a control register and a status register that allow the selection and the verification of the several features of the board. Table 3 shows the detail of the bits of the control register.

Table 3: Detail of the control register bits.

Bit name	Action taken	…if…	Available in mode :
Bit 0	State SETUP/RUN		Always W
Bit 1	Reset STRG		Always W
Bit 2	Reset ITRG		Always W
Bit 3	Enable Interrupt 0	In Fifo A Almost Full	W in Setup State
Bit 4	Enable Interrupt 1	In Fifo A Full	“
Bit 5	Enable Interrupt 2	In Fifo B Almost Full	“
Bit 6	Enable Interrupt 3	In Fifo B Full	“
Bit 7	Enable Interrupt 4	Event Number Mismatch	“
Bit 8	Enable Interrupt 5	Out Fifo Almost Full	“
Bit 9	Enable Interrupt 6	Out Fifo Full	“
Bit 10	Enable Interrupt 7	Timeout 1 RosA	“
Bit 11	Enable Interrupt 8	Timeout 1 RosB	“
Bit 12	Enable Interrupt 9	Timeout 2 RosA	“
Bit 13	Enable Interrupt 10	Timeout 2 RosB	“
Bit 14	Enable Interrupt 11	Interrupted link in ROS A	“
Bit 15	Enable Interurpt 12	Parity error Link A	“
Bit 16	Enable Interrupt 13	Interrupted link in ROS B	“
Bit 17	Enable Interrupt 14	Parity error Link B	“

When an error is found, if the control register is set in such a way to allow it, an interrupt is issued. When the control CPU will look at this interrupt, it should read the Interrupt Register in such a way that the appropriate action is taken for the various cases that can cause an interrupt to be issued In Table 4 the bits of this register are shown.

Table 4: Detail of the Interrupt Register

Bit 0	In Fifo A Almost Full	Interrupt 0
Bit 1	In Fifo A Full	Interrupt 1
Bit 2	In Fifo B Almost Full	Interrupt 2
Bit 3	In Fifo B Full	Interrupt 3
Bit 4	Event Number Mismatch	Interrupt 4
Bit 5	Out Fifo Almost Full	Interrupt 5
Bit 6	Out Fifo Full	Interrupt 6
Bit 7	Timeout 1 Link A	Interrupt 7
Bit 8	Timeout 2 Link A	Interrupt 8
Bit 9	Timeout 1 Link B	Interrupt 9
Bit 10	Timeout 2 Link B	Interrupt 10
Bit 11	Link A interrupted	Interrupt 11
Bit 12	Parity error Link A	Interrupt 12
Bit 13	Link B interrupted	Interrupt 13
Bit 14	Parity error Link B	Interrupt 14

The PLX 9080 PCI bridge allows DMA transfers, 32-bits and 33 MHz clock; it is foreseen to use the board as a PCI slave. Figure 6 shows the block diagram of the board. It can be seen the parallel structure of the two channels up to the control and merge FPGAs. After the Xilinx, data proceed on an unique row. In a board with more than 2 input channels, the part to be modified would be the one ahead of the Xilinx.

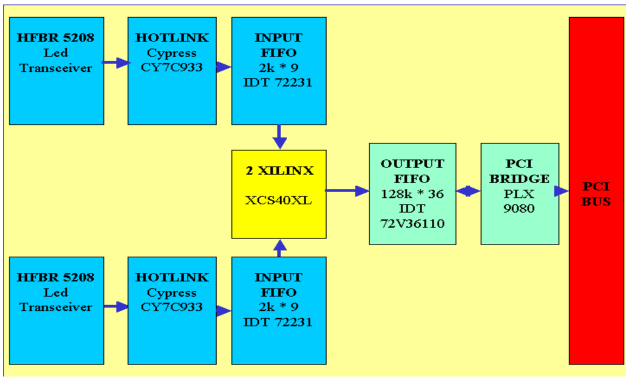

Figure 6: PMC version of the muon DT DDU prototype

Details of the board can be found in reference [7].

D. Control CPU and operating system.

The lab test setup uses a Motorola MVME2301 VME CPU, with a PPC 603 processor. It can house up to two PMCs, thus allowing test and debug of two boards at one time without need of an extension. To be able to use 6 boards, the Motorola PMC-span VME boards are used. The operating system is VxWorks with the Tornado 2 development environment. All programs are written in C and C++.

IV. CONCLUSIONS.

The entrance door for frontend data of the CMS barrel muon drift tubes detector will be made using a board, named Detector Dependent Unit in the CMS DAS jargon, that will allow merging of data, re-formatting, error check, data monitoring and spying. Two prototypes have been realized, in VME and PMC form factors; to test them in the lab other boards have been built. The prototypes have been and are being used with the prototypes of frontend boards realized in CIEMAT Madrid. It is foreseen to build the final DT DDUs in VME, either 6 or 9U.

V. REFERENCES

1. The Compact Muon Solenoid technical proposal CERN/LHCC 94-38, 15 December 1994
2. The muon project technical design report CERN/LHCC 97-32, 15 December 1997
3. Properties and performances of a frontend ASIC prototype for the readout of the CMS Barrel Muon Drift Tubes Chambers – CMS Note 1999/033
4. http://cmsdoc.cern.ch/cms/TRIDAS/html/tridas.html
5. http://www.to.infn.it/esperimenti/CMS/electronics/PTT
6. http://www.to.infn.it/esperimenti/CMS/electronics/pDDU
7. http://www.to.infn.it/esperimenti/CMS/electronics/DDU2000