The APVE emulator to prevent front-end buffer overflows within the CMS Silicon Strip Tracker

G. Iles, W. Cameron, C. Foudas, G. Hall, N. Marinelli

Blackett Laboratory, Imperial College London, Prince Consort Road, London, SW7 2BW
gm.iles@ic.ac.uk

Abstract

A digital circuit board, employing Field Programmable Gate Array (FPGA) technology, has been built to emulate the logic of the pipeline memory of the APV25 readout circuit for the CMS Silicon Strip Tracker. The primary function of the APVE design is to prevent buffer overflows in the APV25. It will also provide information to the Front End Drivers (FEDs) to ensure synchronisation throughout the Silicon Strip Tracker. The purpose and functionality of the APVE is presented along with a prediction of the performance from simulation results.

I. INTRODUCTION

The CMS Silicon Strip Tracker APV25 readout chip [1, 2] is designed to record analogue data at a rate of 40MHz. Data are stored in analogue pipelines on the readout chip. Upon reception of a Level 1 Accept (L1A) signal from the Trigger Control System (TCS) [3], data are transferred ~100m via optical links [4] to Front End Driver (FED) cards [5] located in the CMS electronics room. The FEDs digitise the analogue data and employ fast FPGAs to apply pedestal and noise corrections and to reduce the raw data sample by cluster finding. The clustered data are then transmitted to the CMS DAQ via S-links[3, 6].

The maximum average frequency of L1As will be 100kHz (10μs period) while events can be read out from the APV25 at a rate of one event per 7μs. To allow for Poisson fluctuations of the First Level Trigger (FLT) rate, buffers have been introduced both at the APV25 and the FED level. In the APV25 this has been achieved by simply extending the analogue pipeline already necessary for buffering data until receipt of a L1A. The CMS Trigger design [7, 8] requires that all readout buffers are monitored and that their status classification (Busy, Ready, Warning-Overflow, Error, Out-Of-Sync) is transmitted back to the TCS. The TCS may then inhibit triggers from the FLT if the status is Busy or take other action depending on the information it receives.

Monitoring the APV25 buffers, which are located on the CMS detector, poses a particular challenge because Poisson fluctuations of the FLT rate can produce L1As within time intervals which are shorter than the travel time of the monitoring signals from the CMS detector to the CMS electronics room. Fast monitoring signals coming from the CMS detector cannot therefore provide sufficiently early warning that APV25 buffers are about to overflow. The consequence is that the APV25 buffers would overflow and data would be lost. The APV25 would need to be reset.

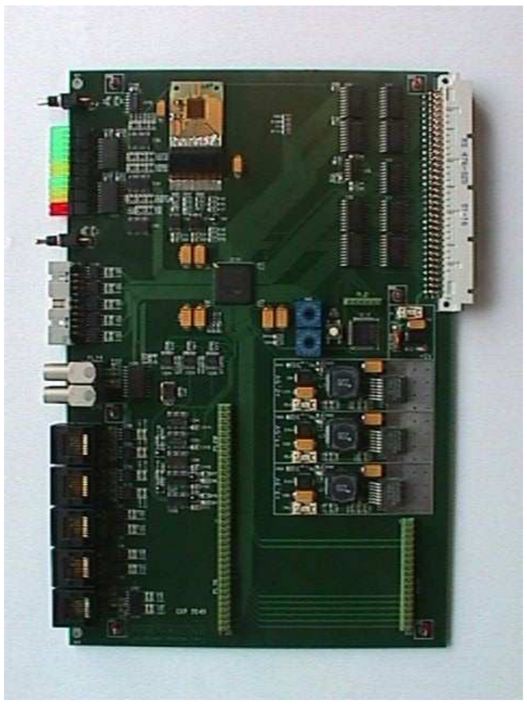


Figure 1: The APVE board

To avoid buffer overflows, an APV25 buffer emulator board (APVE) which emulates exactly the status of the APV25 buffers is under development (fig. 1). The APVE will be installed very close to the TCS crate. Hence, it will be able to report the status of the APV25 buffers to the TCS (Trigger Control System) quickly enough. The APVE uses fast FPGA technology and is capable of determining the APV25 status in ≤ 2 LHC clock cycles, thus ensuring maximum buffer efficiency and preventing APV25 buffers from overflowing.

The APVE will also be used to transmit the address of the memory cell in the APV25 that was used to buffer the L1A event. This address is included in the header of the APV25 data frames sent to the FEDs. The "golden" pipeline address received from the APVE will be compared with the actual ones coming from the tracker, thus providing an important verification that synchronisation has been maintained.

II. BUFFER OVERFLOWS IN THE APV25

When the APVE detects that a buffer overflow is about to occur it will send the status signal Busy to the TCS. The TCS will respond by vetoing further L1As.

L1As may be issued at maximum rate of 1 every 3 bunch crossings. Consequently, if we want to make maximum use of the APV25 buffers by only vetoing L1As when all the buffers are full we need the control loop from TCS L1A inhibit gate to APVE and back again to take less than 3 bunch crossings. If this were not the case a further L1A may be allowed through by the TCS before the APVE has had time to report that the APV25 buffers are full. The additional L1A would cause the full buffer to overflow and all APV25s would need to be reset. Control loops larger than 3 bunch crossings must assert Busy before the buffers are full to provide space for additional L1As that may pass through the TCS before the L1A inhibit is applied. The number of buffers required for additional L1As after Busy has been asserted (i.e. determined by the magnitude of the control loop) is set by a VME write.

III. CONTROL STRUCTURE

To form a small TCS-APVE control loop the following control structure has been adopted (fig. 2). The APVE receives the LHC clock and following control signals; L1A; Level 1 Reset (L1Reset); Bunch Crossing 0 (BC0); Event Counter Reset (ECR); Orbit Counter Reset (OCR) from both a Central TCS (CTCS) & Local TCS (LTCS). The Local TCS allows the Tracker to operate when the main Trigger system is down for maintenance. CTCS or LTCS can be selected under VME control.

The APVE sends back the Tracker status signal which is either Busy, Warning-Overflow, Out-Of-Sync, Ready or Error to the active TCS. Error is asserted to the unused TCS. The Tracker status is derived by combining the APVE status signal with the equivalent ones coming from the FEDs.

The APVE clock and control signal input from each TCS, the APVE status output to each TCS and merged FED status input are transmitted as LVDS logic on standard Ethernet type cable and connectors (i.e. 4 bit, twisted pair cables of approx 100 ohm impedance with RJ-45 connectors). The control signals are encoded using 3bits, with the remaining bit used for the LHC clock. The APVE status is encoded with all 4 bits.

The "golden" APV25 pipeline address is transmitted to the FEDs via the TTCci B channel [9]. APV25s in the Tracker receive L1As and control information via the FEC and CCU ring control system. They send data frames, which include the pipeline address of each event, to the FEDs where the pipeline address is checked.

The Tracker can be operated in 4 separate partitions and thus 4 APVE modules are required.

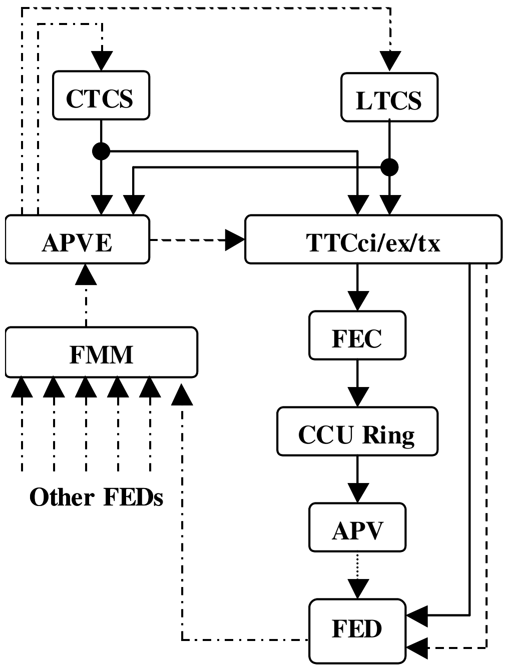


Figure 2: The CMS Tracker control system. Control signals (solid line) are generated by the Central or Local TCS. Data is sent from the APV25s to the FEDs (dotted line). The APVE and FED status are reported back to the Central and Local TCS (dot-dash line). The "golden" pipeline address (dashed line) is transmitted to the FEDs via the TTCci B channel

IV. IMPLEMENTATION

The APVE is a standard 6U VME board with A24/D16 addressing. All the logic, with the exception of that used to provide a back up clock is implemented in a single Xilinx Virtex-II FPGA (XC2V1000) [10].

To monitor the buffer status of the APV25 two options have been implemented. The first and simplest method uses a real APV25 chip and a counter to keep track of the number of events waiting to be read out of the APV25. The counter is incremented every time a L1A arrives and decremented every time an event is read out from the APV25.

The second method uses an FPGA emulation of the APV25 in real time. This provides complete knowledge about the status of the internal APV25 buffer logic and thus the maximum buffer efficiency is achieved. This method places stringent demands on timing within the FPGA yet early indications are that it will work.

In addition to the buffer status logic the FPGA contains a VME bus to Wishbone bus [11] bridge that allows extra FPGA components to be easily added. The components currently attached to the Wishbone bus are; the APVE control and status registers; an I2C interface for communication with the real APV25; a 64bit wide by 4k deep status memory. The latter records any change in status of the APV25 emulator, FED input, or Central and Local TCS feedback signals. The 64bits are composed of the status signals (4x4bits), bunch crossing number (12bits), orbit number (32bits) and reserved area (4bits).

V. Tracker Efficiency

The Tracker induced dead time versus the length of the control loop has been studied. The APV25 can be operated in two modes. In peak mode 31 L1A buffers are available whereas in deconvolution mode only 10 L1A buffers, each containing three pipeline samples, are available. Deconvolution is therefore the most demanding mode of operation.

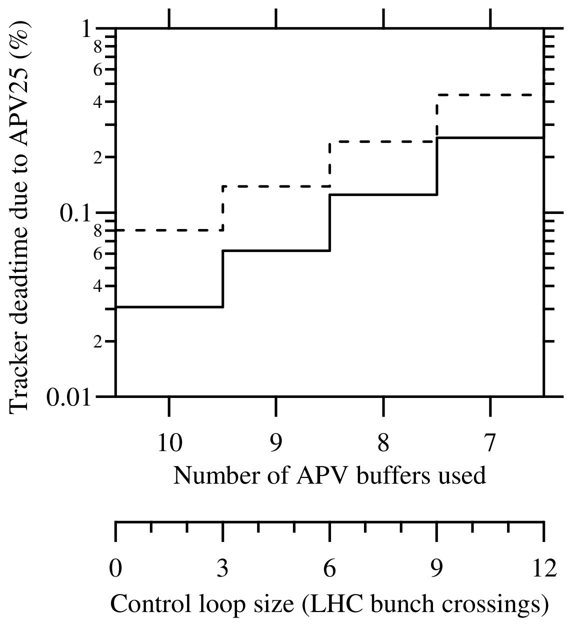


Figure 3: Tracker deadtime versus the magnitude of the control loop and the maximum buffer size allowed by such a control loop. The APV25 is operating in deconvolution mode with a maximum buffer size of 10 and a raw L1A rate of 100kHz. Buffer size calculated and thus deadtime calculated using both a real APV25 (dashed line) and a simulation of the APV25 (solid line). The better performance achieved with simulation is due to a greater knowledge of the state of internal APV25 buffer logic.

The Tracker deadtime has been plotted (fig. 3) versus the number of buffers used for a 100kHz raw L1A rate with the APV25 in deconvolution mode. L1As which would have produced a L1A separation of less than 3 bunch crossings were vetoed, but any other possible trigger rules were ignored. The deadtime was calculated from the ratio of L1As that were vetoed because the APVE was asserting Busy and the raw L1A rate. The buffers were monitored using a real APV25 (dashed line) and a C code simulation of the APV25 internal buffer logic (solid line) derived from previous work [12].

The C code simulation performs better than the real APV25 because we have more precise knowledge of the buffer states within the APV25. The C code has been translated into the hardware description language VHDL, which will be downloaded into the FPGA.

The current control structure (fig. 2) should produce a control loop of just a few clock cycles. The number of buffers used in deconvolution mode should therefore be 8, 9 or all 10. This corresponds to a deadtime of <0.25% if a real APV25 is used or <0.13% if a hardware emulation of the APV25 is used.

VI. Pipeline Address Transmission

The APVE "golden" pipeline address is transmitted to the FEDs by the Trigger, Timing and Control (TTC) system [13]. The TTC distributes to the experiment, via a single optical fibre, a clock, and 2 data channels. Channel A is used for L1A transmission whereas channel B is used for control commands. The TCS initiates B channel commands and the TTCci formats them. Channels A & B are then sent to a TTCex for encoding and transmission on optical fibre.

The "golden" pipeline address will be sent asynchronously on the B channel so that it does not interfere with synchronous commands being initiated by the TCS. However, the pipeline address must not be blocked by transmission of synchronous B channel commands for a long time period if the pipeline address is to reach the FED before the FED is ready to check it. This requires guaranteed asynchronous bandwidth on the B channel of $\geq$ 42clks (pipeline address transmission length) [14] every 280clks (APV25 data frame length). This requires the TCS, that initiates the B commands, and TTCci, that implements the B commands, to follow certain rules.

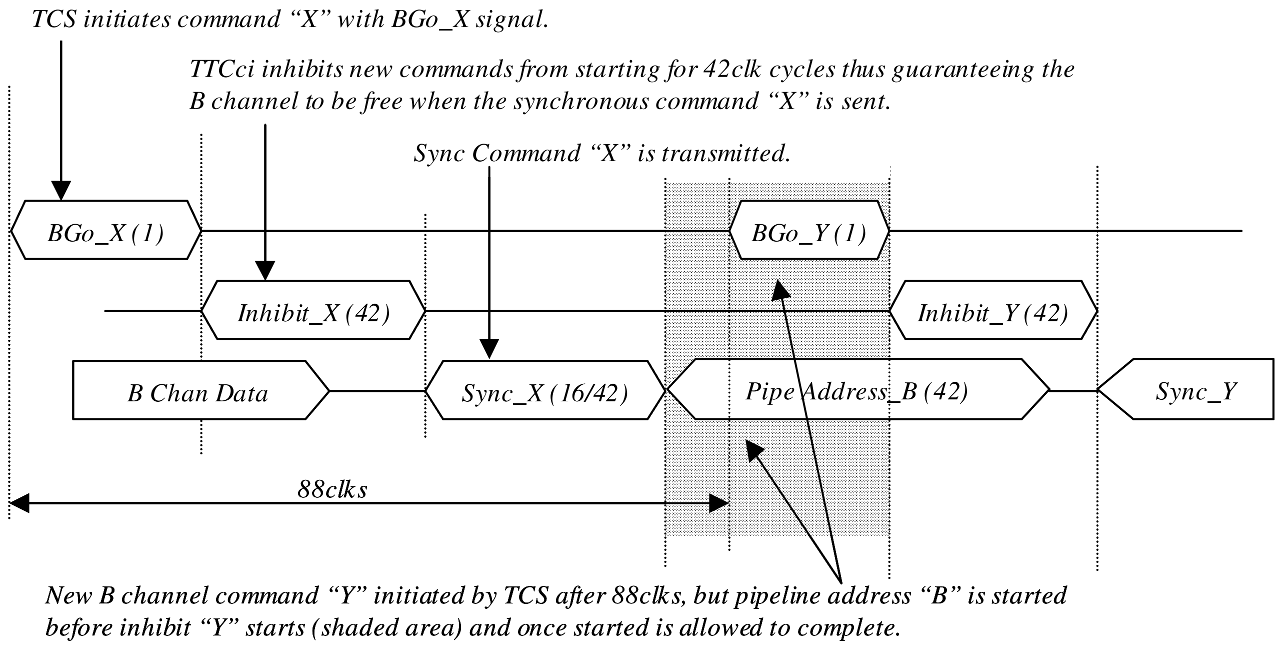
The TCS (Central & Local) must separate synchronous B channel commands by $\geq$ 88clk cycles (fig. 4). This ensures that a pipeline address may be transmitted between successive synchronous B channel commands. The latter should take no longer than 84clk cycles to complete if an inhibit period of 42clk cycles is chosen on the TTCci. The pipeline address will take longer than the 4clk cycle gap to be sent, but once started it is allowed to complete during the inhibit period that exists prior to a synchronous command being sent (16 or 42clk cycles). The purpose of the inhibit period is to free the B channel for a synchronous command by allowing any currently executing command to finish, but by blocking any new command from starting.

Tracker TTCci must maintain this command separation. All B channel commands must be initiated by the TCS and have the same inhibit duration of 42clk cycles, and have the same inhibit delay. TTCci "doubles" are not allowed.

VII. Conclusions

A VME board has been designed to prevent buffer overflows in the CMS Tracker. The design of the board and its integration into the CMS control structure ensure that the Tracker deadtime is kept to a minimum.

The deadtime is determined by the magnitude of the control loop formed between the APVE and the TCS and the type of buffer monitoring chosen. The latter can be



TCS initiates command "X" with BGo_X signal.
TTCci inhibits new commands from starting for 42clk cycles thus guaranteeing the B channel to be free when the synchronous command "X" is sent.
Sync Command "X" is transmitted.
BGo_X (1)
BGo_Y (1)
Inhibit_X (42)
Inhibit_Y (42)
B Chan Data
Sync_X (16/42)
Pipe Address_B (42)
Sync_Y
88clks
New B channel command "Y" initiated by TCS after 88clks, but pipeline address "B" is started before inhibit "Y" starts (shaded area) and once started is allowed to complete.

Figure 4: Diagram showing the "golden" pipeline address being transmitted on the B channel. The top line shows the TCS initiating two different synchronous commands. The middle line shows the inhibit generated for each synchronous command. The bottom line shows B channel usage. The time taken in clock cycles for each event to complete is shown in brackets after the event description.

achieved with either a real APV25 chip or a hardware emulation of the chip in an FPGA. If we assume a TCS-APVE control loop of less than 9 clock cycles these yield a deadtime of less than 0.25% and less than 0.13% respectively.

VIII. ACKNOWLEDGEMENTS

We would like to thank the following; Sarah Greenwood for her layout of the design; John Coughlan for his comments on pipeline address transmission on the B channel; Rob Halsall for his help with FPGA design software; Joao Varela and Anton Taurok for discussions about integrating the APVE into the CMS control structure.

IX. REFERENCES

[1] M. Raymond et. al., "The CMS Tracker APV25 0.25µm CMOS readout chip", Sixth Workshop on Electronics for LHC Experiments, CERN/LHCC/2000-041.

[2] The Tracker Project Technical Design Report, CERN/LHCC/98-6.

[3] A. Racz et. al., "Trigger Throttling System for CMS DAQ", Sixth Workshop on Electronics for LHC Experiments, CERN/LHCC/2000-041.

[4] F. Vasey et. al., "Project status of the CMS optical links", Sixth Workshop on Electronics for LHC Experiments, CERN/LHCC/2000-041.

[5] J. Coughlan et. al., "The Front-End Driver Card for the CMS Silicon Strip Tracker Readout", These proceedings.

[6] A. Racz et. al., "CMS Front-End/DAQ Interfacing", Sixth Workshop on Electronics for LHC Experiments, CERN/LHCC/2000-041.

[7] The Trigger and Data Acquisition (TriDAS) Project Technical Design Report, Volume 1, The Level-1 Trigger, CERN/LHCC/2000-038.

[8] J. Varela, "Timing and Synchronization in the LHC Experiments", Sixth Workshop on Electronics for LHC Experiments, CERN/LHCC/2000-041

[9] J. Varela, "CMS L1 Trigger Control System", Draft CMS Note 2002

[10] Xilinx, "Virtex-II Platform FPGA Handbook", www.xilinx.com.

[11] WISHBONE System-On-Chip (SoC) Interconnection Architecture for Portable IP Cores, Revision: B.1, Silicore Corporation, 6310 Butterworth Lane, Corcoran, MN 55340

[12] N. Marinelli, "APV Logic Simulations", CMS Note 1999/028

[13] B. Taylor, "Timing Distribution at the LHC", These proceedings.

[14] J. Christiansen, A. Marchioro, P. Moreira and T. Toifl, "TTCrx Reference Manual, February 2001, Version 3.2", RD12 Working Document, CERN.