

# The AMSRW Board for the Silicon Vertex Tracker upgrade at CDF

J. Adelman, A. Annovi, A. Bardi, M. Bitossi, R. Carosi, M. Dell'Orso, P. Giannetti, P. Giovacchini, J. Lewis, T. Liu, M. Piendibene, M. Pitkanen, B. Reisert, L. Sartori, M. Shochet, B. Simoni, F. Spinella, F. Tang, S. Torre

**Abstract**– The Silicon Vertex Trigger (SVT) processor has been built at CDF for extremely fast (~ 10 msec) and high precision (roughly offline quality) pattern recognition. It is going to be upgraded to have at high luminosity the same crucial role it had in the data collection for the RUN II physics.

A modern version of most of the SVT boards is obtained with a minimum of new hardware. A Pulsar board, which was designed for and is now being used in the CDF level-2 upgrade, has been used for most the SVT functions that needed to be upgraded. The Pulsar combines the power of dedicated hardware with the flexibility of general purpose CPUs. The new Sequencer (AMS) that controls the Associative Memory (AM) operation for pattern recognition, uses a small fraction of a Pulsar board. The same Pulsar is powerful enough to also carry out the Road Warrior function (AMS-RW), to remove redundant track candidates prior to track fitting. We report about the AMS-RW design, tests and performances.

## I. INTRODUCTION

THE AMS-RW card implements and enhances the functions of two boards installed in the current SVT [1]: the Associative Memory Sequencer (AMS) and the Road Warrior (RW) [2]. The AMS is the operating sequencer for the AM++ associative memory board [3] built for the SVT upgrade [4]. The RW function eliminates redundant track candidates (ghost roads) before track fitting. The greatly increased number of stored trajectories provided by the new AM++, the protocol for the new AM chips [5], and the necessity of increasing processing speed require a new AMS-RW board.

Manuscript received February 20, 2006.

A. Bardi, M. Bitossi, M. Dell'Orso, P. Giannetti, M. Piendibene and F. Spinella are with INFN Sezione di Pisa, Largo Pontecorvo 3, 56127 Pisa, Italy (telephone: + 39-050-2214000, e-mail: [firstname.lastname@pi.infn.it](mailto:firstname.lastname@pi.infn.it)).

M. Bitossi, M. Dell'Orso are with Dipartimento di Fisica, Università di Pisa, Largo Pontecorvo 3, 56127 Pisa, Italy (telephone: + 39-06-2214000, e-mail: [firstname.lastname@pi.infn.it](mailto:firstname.lastname@pi.infn.it)).

A. Annovi and S. Torre are with INFN, Laboratori Nazionali di Frascati, via E. Fermi 40, I-00044 Frascati (Roma), Italy (telephone: + 39-06-94031, e-mail: [firstname.lastname@lnf.infn.it](mailto:firstname.lastname@lnf.infn.it)).

P. Giovacchini and B. Simoni are with STMicroelectronics Srl, Via C. Olivetti 2, 20041 Agrate, Milano, Italy (telephone: + 39-0396031, e-mail: [firstname.lastname@st.com](mailto:firstname.lastname@st.com)).

J. Adelman, M. Shochet and F. Tang are with The Enrico Fermi Institute, University of Chicago, 5640 South Ellis Avenue, Chicago, Illinois 60637, USA (telephone: 773-702-7823, e-mail [firstname.lastname@hep.uchicago.edu](mailto:firstname.lastname@hep.uchicago.edu)).

J. Lewis, T. Liu, M. Pitkanen, B. Reisert are with FNAL, PO Box 500, Batavia, Illinois 60510, USA.

To perform this task we use a Pulsar board [6], which is highly reconfigurable and complies with CDF and SVT standards. We implement both the AMS and RW functions in a single pulsar card. Two custom mezzanine boards are plugged into the Pulsar to satisfy the memory needs of the AMS-RW algorithms. The firmware for all the necessary logic is contained in the 3 large FPGAs stuffed on the Pulsar. The AMS-RW and the AM++ cards are connected through a backplane on the P3 connectors (figure 1) and the data are exchanged using a custom bus named AM Bus

## II. ARCHITECTURE

The logic dataflow of the AMS-RW board is shown in figure 1.

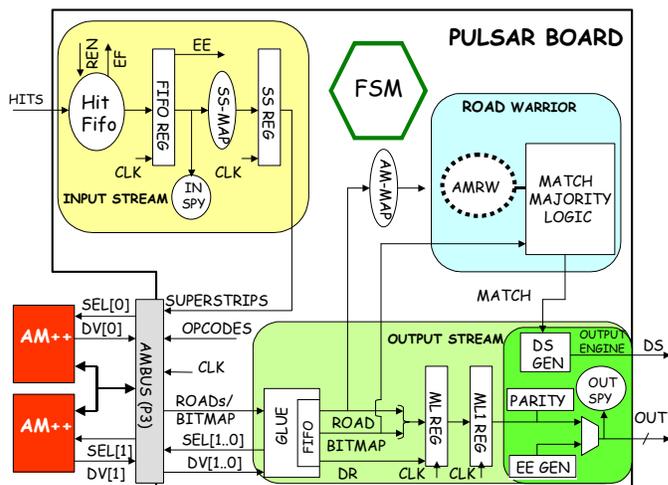


Figure 1: AMS-RW dataflow

Two data streams are defined: the input and the output streams.

### A. Input stream

In the input stream each data packet is called a “hit”. Each word contains a hit coordinate (12 bits), a layer number (3 bits) and an operation code (4 bits). Hits may be one word long or more depending on which kind of detector the hit is coming from (the silicon vertex detector or the drift gas chamber). Under normal circumstances (Run Mode) hits flow from the input stream to the AM bus (P3 connector) and reaches the AM++ cards. Incoming hits are stored in a FIFO, and popped when the sequencer is ready to process them (the

previous event has been fully processed). The hits are then sorted into a number of classes according to their coordinate values. These classes are called Super Strips (SS). The mapping between each hit and the corresponding Super Strip is obtained through a 128k x 16 bit lookup table (SSMAP). The SSMAP is stored in a static RAM, downloaded through the VME bus at startup. The Super Strips are then sent to the AM++ boards. The Input Stream is controlled by a flexible finite state machine (FSM). The FSM generates also instructions, called operation codes (opcode) for the AM++ boards [3]:

1. Init (opcode= 5): the AM++ boards are reset each time a new event has been processed.
2. Input (opcode= 4): SSs are sent to the P3 connector.
3. End-of-Hit (opcode= 1): to communicate to the AM++ boards that hits from the current event are finished. It is generated when the input stream End-Event word has been popped from the input FIFO.

### B. Output stream

The AM++ boards compare the Super Strips on the fly with the associative memory content and keeps track of matches: whenever a stored candidate track is matched in the requisite number of layers, this is signaled by the Data Valid (DV\_) to the AMS-RW by the corresponding AM++. The Glue (see figure 1) is the logic inside the AMS-RW that controls up to 2 AM++ boards. The Glue enables the data transfer from the AM++ activating the Select signal, Sel. The enabled AM++ sends out the address of the road and also the list of the matched superstrips inside the road, the “bitmap”. This is a 6 bit word, one bit per layer, which contains information about the fired layers: a bit equal 1 (0) means that the corresponding layer was (was not) fired. Each of these road addresses is complemented with a bit field that identifies the sending AM++ to form a “road” word. The road is sent in parallel to the Road Warrior section and to a pipeline where it waits the for the Road Warrior decision. All roads passing the Road Warrior selection (Match= 0 in figure 1) are sent to the output as two (road address and bitmap) or single (road address only) word packets. A VME control register is available to set the desired option.

The output stream for a particular event can be interrupted in two different cases:

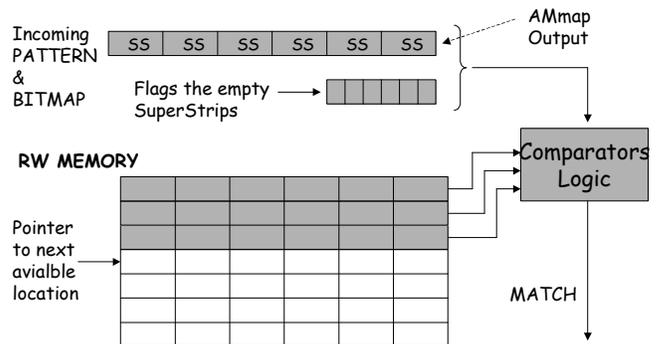
- The event belongs to a class (identified by trigger logic before SVT) that should not be analyzed by SVT. No mechanism prevents hits to arrive to the AM++ boards, but in the Hit\_End\_Event word a bit called KILL on Bit 18 (KILL18) is used by the previous logic to signal that the event should not go ahead inside SVT. In this case no roads are sent to the AMS-RW output, even if the AM++ boards have found them.
- The AMS-RW has the possibility to set a road limit used to reduce the probability of very large processing time fluctuations, due to very crowded events. If the number of roads sent to the output exceeds the road limit, a warning

flag is set in the End Event word (Truncated Output, TO) and the flux of roads is suddenly stopped to go ahead with the next event.

### C. The Road Warrior

The AMS can request the Associative Memory board to output first roads in which all 5 silicon layers are hit (5/5 mode), second all roads with a missing layer (4/5). Allowing for a missing hit increases the tracking efficiency but also the SVT processing time is increased. A large part of this timing increase is due to “ghost” roads. When a track produces a hit on each of the five silicon layers, it generates many output roads: one contains all five fired superstrips (5/5), all the others (4/5) contains different combinations of 4 hit superstrips. Without the Road Warrior function, duplicate tracks go down in the SVT pipeline requiring extra processing time into each board they cross, with no real advantage. Duplicate tracks are eliminated at the SVT end, after the fitting function. With the Road Warrior, the duplicate tracks are eliminated before the roads are analyzed at full resolution, significantly reducing the fitting time. The Road Warrior was recently installed [2]. Now we anticipate further the Road Warrior function, moving it inside the AMS Pulsar and reducing further the processing time [4].

#### ROAD WARRIOR principles



IF MATCH = 0 then STORE Incoming PATTERN to next available location in memory and send Road to the output  
ELSE DISCARD Incoming PATTERN and Road

Figure 3: Road Warrior working principles..

The Road Warrior algorithm employs the associative memory idea, widely used throughout SVT. It stores the SS combinations (patterns) of found roads in a small associative memory (AMRW) built on the fly when roads are sent to the output. Empty SS's are identified and flagged using the bitmap. Each road is allowed to proceed to the output only if it differs in at least one non-empty SS from all roads previously sent and stored in the AMRW. Whenever the road number is received, it is used as an address to a lookup memory, the Associative Memory Map (AMMAP), which stores for each layer the superstrip associated with the road. The AMMAP is implemented as 1Mx36 bit RAM. Each pattern is associated with 6 superstrips (one for each layer), each one 12 bits wide,

so each pattern (72 bits) needs 2 locations in the AMMAP. The RW logic (figure 3) includes the AMRW, a 72 bit wide associative memory, 64 locations deep, which easily fits in one of the Pulsar board FPGAs. The output of the AMMAP, after receiving a road and applying the bitmap to identify empty layers, is compared to all the stored roads in the AMRW. If there is a match, the newly received road is a duplicate and is discarded. If there is no match, the AMMAP output is copied into the next available location in the AMRW, and the road is sent to the output engine. At the end of the event, the AMRW is cleared.

The output engine also calculates the parity of the output stream and sends the end-of-event word, which contains the Event Number and diagnostic information.

#### D. The AMS-RW Finite State Machine

The dataflow in the AMS-RW is implemented as a multi-step pipeline, supervised by a finite state machine, FSM, shown in figure 4.

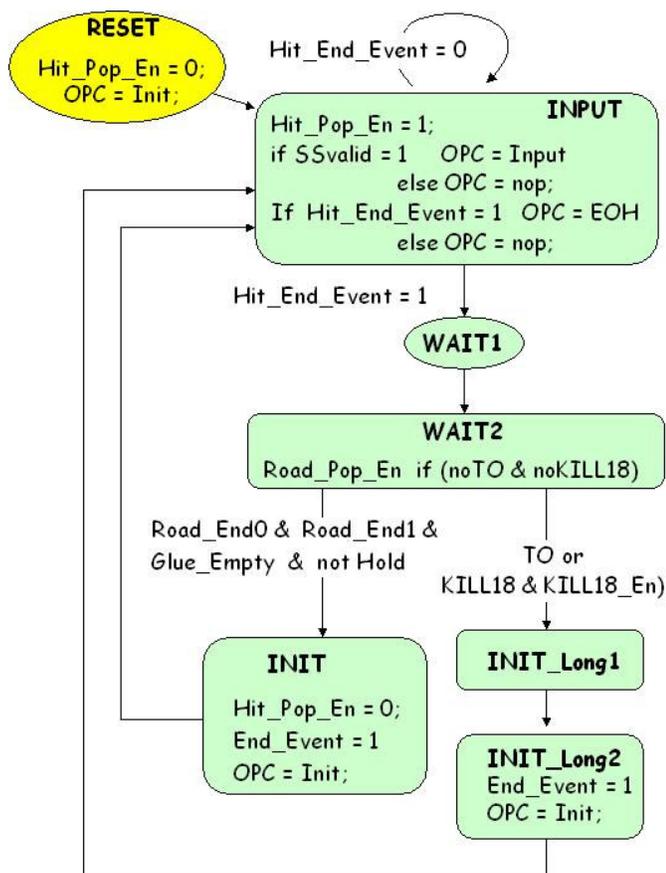


Figure 4: the Finite State Machine, FSM

In the RESET state the machine sends the opcode (OPC) that initializes the AM chips (Init). The Hit Pop Enable signal is inactive: the AMSRW is not ready to process the hits of the next event. The FSM goes automatically in the INPUT state, where the Hit Pop Enable signal is activated and the AMSRW

starts to pop hits from the input FIFO as soon as they are available, see Figure 1. The FSM receives the SSvalid signal that validates each SS to be sent to the AM++ s together with the Input opcode. Data flagged by an inactive SSvalid signal are sent with the no operation (nop) opcode. If a validated Hit\_End\_Event word is extracted from the FIFO (no more hits are in the FIFO for this event), the End-of-Hit (EOH) opcode is sent to the AM++ and the FSM jumps to the WAIT1 and immediately after to the WAIT2 state and starts to pop roads from the GLUE-FIFO if the Kill on Bit 18 option was not enabled or was not active in the received Hit\_End\_Event word. If the Kill on Bit 18 option was enabled and set for the event under processing or the road limit was reached (Truncated Output, TO) the Road\_Pop\_En is suspended and the FSM jumps to the INIT\_long1 state. The FSM stays a programmable number of cycles into the two INIT states. The AMSRW and the AM++ boards are initialized to clean up the whole road pipeline before starting with a new event. If neither Kill on Bit 18 or TO signals are active, the GLUE-FIFO is popped until both the AM++s set the Road\_End signal (Road\_End0 and Road\_End1 in figure 4). Road\_End active means that the related AM++ has finished to send its roads. Moreover the FSM waits for an active Glue-Empty signal meaning that all the event roads have been processed and for the Hold signal from the downstream board being inactive (the downstream board can actually receives roads). When no more roads are inside the AM++ and the AMSRW the FSM can jump to the INIT state to send downstream the End\_Event word and close the event. In the INIT state the Init opcode is sent to the AM++ to be ready for a new event. Going back to the INPUT state the FSM starts to process the new event.

### III. DIAGNOSTICS

The AMSRW board handles a huge amount of data and is continuously running. For this reason several diagnostic tools have been included in the firmware to verify in real time and without halting the dataflow the correct function of the board.

The main tools are:

- Spy Buffers
- L2 buffers

#### A. Spy Buffers

The board, like all devices in SVT [1], is equipped with two Spy Buffers, the Input Spy and the Output Spy. These are circular memories, which continuously spy the data flowing into the Input Connector (Hits) and going out of the output connector (Roads). If an error comes out, the Spy Buffers are automatically “frozen” by a dedicated board of the SVT crate (Spy Control) and then accessed through the VME interface to read the content. In this way many typologies of problems (such as a damaged cable or chip) can be easily identified. Note that these operations can be done during the run mode, without any interference with the normal dataflow. The input Spy Buffer is located in a SRAM chip 128k words deep, while the output Spy Buffer is implemented inside one of the FPGAs

and it is a small RAM 1k words deep. This asymmetry between the input and output spy buffers is justified by the fact that events are characterized by a number of hits much larger than the number of found roads.

### B. DAQ Buffers

Another level of buffers, The DAQ buffers, can be used for monitoring and diagnostics. Every end of event, the incoming hits and roads can be copied in memories to be readout by the CDF DAQ through the VME interface and to be written on tape. These data streams can be enabled to be part of the CDF event, which allows to make periodical offline studies to verify the good shape of the trigger. In the DAQ buffers are also stored AMS-RW relevant timing information, like the processing time per event or the arrival time of the hits.

## IV. HARDWARE IMPLEMENTATION

The AMS-RW board is implemented using a Pulsar board, which is a general-purpose 9U VME interface board for HEP applications [6] developed by a group of physicist and engineers at the University of Chicago. The Pulsar is high customizable and flexible thanks to three large FPGA and the possibility to extend its functionality with mezzanine cards. Up to four mezzanine (PMC format cards) can be plugged on the Pulsar bottom side. We use two mezzanine cards, developed by the University of Chicago, to implement the AMMAP and SSMAP memories.

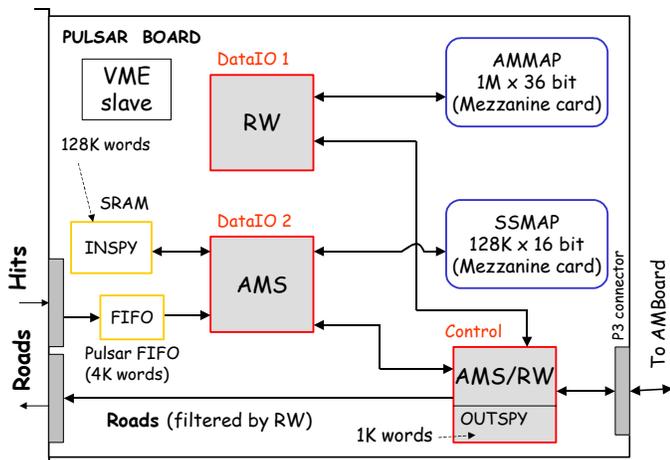


Figure 5: Implementation inside the Pulsar

Figure 5 shows the 9U VME board structure. The firmware is distributed in the 3 large FPGAs (gray boxes): the Road Warrior section is located in the DataIO1 FPGA connected to the AMMAP mezzanine, while the AMS fits in the other two. The DataIO2 FPGA controls the hit input stream and is connected to the SSMAP mezzanine, while the Control FPGA handles the AMSRW output to the downstream board and the communication with the AM++ boards through the P3 connector.

The AMS-RW board is equipped with a slave VME interface, which permits the data exchanging with the crate

CPU. A special working mode, named “test mode”, which stops the data flow, is used to load and check the lookup tables (SSMAP and AMMAP) and all the configuration registers allowing the option selections.

The working frequency, suitable for the CDF needs, has been fixed to 40 MHz. The same clock signal, generated by the AMS-RW is also distributed through the backplane to the AM++ boards.

## V. TESTS

The Pulsar board, programmed as AMS-RW, has been deeply tested together with the AM++ boards and then installed to be part of the SVT system in the Summer 2005. Thanks to the previously made intensive tests, the substitution of the old AMS and RW cards with the new AMS-RW has been extremely smooth (see [3] for a more detailed description of the test procedures before installation). Up to now the system is running without any known problem. The movement of the RW in the AMS-RW, which implements a slightly different algorithm respect to the previous card, is reducing significantly the SVT processing time at high luminosity [4].

## VI. CONCLUSIONS

We have implemented the Sequencer and Road Warrior SVT functions in a single Pulsar Board. The Pulsar board has been proven to be enough powerful to combine the function of two existing boards in a single one.

The new implementation is able to handle a much larger associative memory system and the integrated Road Warrior functionality allows eliminating the duplicate tracks early in the SVT pipeline, reducing the SVT processing time. The installation has been successful.

## VII. REFERENCES

- [1] W. Ashmanskas et al., “The CDF Silicon Vertex Tracker”, Nucl. Instr. and Meth., vol. A518, 2004 pp. 532-536.
- [2] J. Adelman et al., “The Road Warrior for the CDF Online Silicon Vertex Tracker”, Presented at the IEEE NSS Conference, Rome, Oregon, USA, October 22, 2003.
- [3] A. Annovi et al. “The AM++ Board for the Silicon Vertex Tracker upgrade at CDF”, submitted to this conference
- [4] A. Annovi et al. “First Step of the Silicon Vertex Tracker upgrade at CDF”, submitted to this conference
- [5] A. Annovi et al. “A VLSI Processor for Fast Track Finding Based on Content Addressable Memories”, submitted to this conference.
- [6] T. Liu et al., “Pulsar Design and Testing Methodology for CDF Level 2 Trigger Upgrade”, Presented at the IEEE NSS Conference, Portland, Oregon, USA, October 22, 2003. More information about Pulsar project can be found at <http://hep.uchicago.edu/thliu/projects/Pulsar/>.