Pulsar Status Report

PULSAR: PULSer And Recorder

• Pulsar design overview:
  → from L2 teststand tool to a general purpose tool
• Current status:
  → schematics/layout, firmware, board level simulation etc.
• Possible applications for Pulsar
• Board level simulation and prototype testing plan

Note: This talk ended up with ~40 slides, will only show ~20 at the meeting.

Ted Liu
Friday Trigger Meeting, May 10th. 02

Upcoming talks:
  Hotlink mezzanine cards design and prototype (Natalia Kuznetsova)
Pulsar firmware in pulser mode: design and status (Peter Wittich)
Now the Level 2 system is working, things are much less hectic, we can do what we should have done much earlier, which is to get together to talk about the test stand plans.

**this meeting should focus on the functionality requirements (or specifications) for the Level 2 test stand**, once we agree on what we should build, then things can move very fast…

We will have more meetings on test stand issues as needed.

---

**From last meeting (Feb. 22rd)**

Last meeting we focused on functional requirements. At this meeting, I will present Pulsar overall design and current status. A web page is being setup which will contain all the design (hardware and firmware) details for Pulsar, and will be available to everyone soon. Since now we have the actual design, we will have brief status report at every Friday meeting from now on. We will have a Design Review later to discuss all the details.
What has been done since Feb.?

Pulsar design is “optimized and almost finalized”:

• schematics finished early April
• core firmware in place
• intensive board level simulation in progress
• detailed firmware design and VHDL coding in progress
• initial layout work started
• Pulsar web page is almost ready with all design details

Hotlink mezzanine cards (Tx and Rx)

• schematics finished in March
• board level simulation finished in April (with both Tx and Rx)
• prototype boards (Tx and Rx) fully loaded end of April
• prototype debugging/testing in progress

All firmware (VHDL code), compile setting files, simulation waveform, FPGA pin maps are in CVS
Basic hardware requirement: have all hardware interfaces

Pulsar is designed to have all the data interfaces that Level 2 decision crate has. It is a data source for all trigger inputs to Level 2 decision crate, it can be used to record data from upstream as well.

PulsAR: Pulser And Recorder

Main difficulty for Pulsar design:
Each subsystem data path was implemented differently, to design an universal tester board is not all that easy…
The only way is to use mezzanine cards…
Level 2 trigger input data paths were implemented differently

<table>
<thead>
<tr>
<th></th>
<th>SVT</th>
<th>XTRP</th>
<th>L1</th>
<th>CLIST</th>
<th>ISO</th>
<th>Muon</th>
<th>Reces</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Incoming data Clock rate</strong></td>
<td>30Mhz</td>
<td>7.6Mhz</td>
<td>7.6Mhz</td>
<td>20Mhz</td>
<td>12Mhz</td>
<td>30Mhz cdfclk x 4</td>
<td>7.6 Mhz cdfclk</td>
</tr>
<tr>
<td><strong>Interface hardware</strong></td>
<td>SVT cable</td>
<td>SVT cable</td>
<td>L1 cable</td>
<td>Hotlink+fiber</td>
<td>Taxi+fiber</td>
<td>Hotlink+fiber</td>
<td>Taxi+fiber</td>
</tr>
<tr>
<td><strong>data size range</strong></td>
<td>150bits/trk</td>
<td>21 bits/trk</td>
<td>96 bits/evt</td>
<td>46bits/cluster</td>
<td>145bits/cluster</td>
<td>11Kbits/evt</td>
<td>1.5Kb/evt</td>
</tr>
<tr>
<td><strong>Latency range</strong>*</td>
<td>~10-100us</td>
<td>~1us - 10us</td>
<td>~132 ns</td>
<td>~1-20us</td>
<td>~few us</td>
<td>~1-5 us</td>
<td>~ 6 us</td>
</tr>
<tr>
<td><strong>Fixed or variable data length?</strong></td>
<td>variable</td>
<td>variable</td>
<td>fixed</td>
<td>variable</td>
<td>variable</td>
<td>fixed</td>
<td>fixed</td>
</tr>
<tr>
<td><strong>Data with Buffer#?</strong></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td><strong>EOE with data?</strong> (or from separate path?)</td>
<td>yes</td>
<td>yes</td>
<td>-</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>-</td>
</tr>
<tr>
<td><strong>B0 marker?</strong></td>
<td>BC#</td>
<td>BC#</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td><strong>Data gap within one event?</strong></td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td><strong>Flow control ?</strong></td>
<td>Not used</td>
<td>not used</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

* Latency range also depends on L1A history …
9U VME (VME FPGA not shown)

CDF ctrl

Control

Optical IO

3 Altera APEX 20K400 FPGAs

Mezz card connectors

SRAM 128K x 36

VME and CDF Ctrl interfaces are visible to all three FPGAs

Pulsar: Pulser and Recorder (as Level 2 test stand tool)
Pulsar interface to/from P1 and P2 backplane

**VME interface to all three FPGAs:**

- based on UC VMEchip used on other UC boards
- the interface from UC VMEchip to three main FPGAs:
  - VMEdata(31:0), VMEaddr(23:2), vmeAS, vmeDS and vmeWrite

**CDF control (P2) signals to all three FPGAs:**

- CDFCLK, BC, B0, L1A/R, L2B0/B1, Halt, Recover, Run, L2A/R, L2BD0/BD1, CDF_error, GLIVE, STOP, RL(2:0) … -- this is the current map, can add more.

**Pulsar inter-communication control lines (P2 user defined pins):**

- follows SVT implementation (can communicate with any SVT board)
- A1: Pulsar_init
- A2: Pulsar_error
- A3: Pulsar_freeze
- A4: Pulsar_lostlock
- A5: Pulsar_spare

Any Pulsar board can drive and listen to these 5 lines from P2.
Some details on SVT/XTRP, L1 and TSI interfaces

L1 input and output share the same IO pins, set by one enable bit.
Mezzanine Card design as L2 teststand tool only

Front-panel (double width)

Other connectors (2 L1 outputs, 1 TS) will stay inside the board.
The mezzanine card connectors are used for optical I/O (hotlink and taxi)
FPGA choice: the 356-Pin BGA package only allows us to choose up to EP20K200 (with only 271 user I/O pins), while 652-pin BGA package (~500 user I/O pins) allows us to choose anything above EP20K200…prefer 5 V compatible which leaves EP20K400 the only choice. EP20K400 has 26KB internal RAM capability which should be big enough.
Custom Mezzanine cards (follow CMC standard)

- Hotlink: Tx and Rx (CLIST, Muon data paths)
- Taxi: Tx and Rx (Iso, Reces data paths)

CMC: Common Mezzanine Card standard

Altera EP1K30_144 FPGA

Hotlink or Taxi Tx/Rx chips
- Hotlink Tx/Rx: CY7B923JC/933JC
- Taxi Tx/Rx: AM7968/7969-175JC

Hotlink Optical Tx/Rx: HFBR-1119T/2119T
Taxi Optical Tx/Rx: HFBR-1414T/2416T

Hotlink mezzanine card prototype
CMC Connectors (J1 and J3)
Usually has 4 fiber connectors.

add one LVDS connector for CLIST case: only two fiber connector (left side) will be loaded for one Mezzanine card, and one LVDS connector will be loaded on the right side instead of two fiber connectors.
Hotlink mezzanine cards prototypes (Tx and Rx)

Natalia will talk about the details next week
Pulsar design (cont.): From test stand tool to a general purpose tool

Since there are some spare FPGA IO pins left, decided to use them to enhance Pulsar capability.

A few simple modifications:

1. Add signal traces to P3 connector for SLINK IO, this allows Pulsar to interface directly with a PC via commercially available SLINK to PCI cards
2. Make L1 and SVT/XTRP inputs visible to all 3 FPGAs instead of just one FPGA

Note: the mezzanine card connector is already compatible with SLINK mezzanine cards as both of them follow CMC standards. This allows us to test Pulsar prototype with SLINK test tools as well.

Since the modification is simple enough at hardware level, it doesn’t hurt to add them in, to make the board more general purpose. It provides the interface to a PC (via SLINK to PCI board) which could be very useful as a general purpose diagnostic tool.

To learn more about SLINK, see CERN web page:
http://hsi.web.cern.ch/HSI/s-link/
S-LINK on the web
www.cern.ch/hsi/s-link

General information
- News, Specifications, Introduction

Projects using S-LINK
- Experiments
- Institutions outside HEP

S-LINK products
- FEMBs
- S-LINK Implementations
- ROMBs
- Testing devices
- Models
S-LINK is just like a loooong FIFO

Features not shown:
- Self test mode
- Link down signalling
- Return lines
With minor modifications: see next
Pulsar design (general purpose tool)

9U VME
(VME FPGA not shown)

P1

P2
user ctrl

P3

SLINK signal lines

sV T

T S

Control/Merger

3 Altera APEX 20K400 FPGAs

Data IO

Mezz card connectors

Data IO

With minor modifications
Mezzanine cards

Hotlink/Taxi/S-LINK...

Front-panel (double width)

LVDS connectors

Optional user defined signal connection from P2/P3

Three FPGAs: Atlera APEX20K400

PULSAR design

The mezzanine card connectors can be used either for user I/O or SLINK cards

To/from Pulsar or a PC
The transition module is very simple (just a few SLINK CMC connectors).
It uses P2 type connector for P3. We will only use P3 for SLINK and spare (user defined) signals. (it doesn’t have to be P2 type connector).

CERN sent us two transition modules.

Can simply use CDF CAL backplane.

Loaded with SLINK Mezzanine cards.
Examples of SLINK products

LSC (Link Source Card), LDC (LINK Destination Card)

Mezzanine card which can plug onto motherboard via CMC(Common Mezzanine Card) Connector (just like PMC).

Proven technology, has been used by a few experiments to take hundreds of TB data in the past few years
New 32-bit SLINK to 64 bit PCI interface card: S32PCI64

- highly autonomous data reception
- 32-bit SLINK, 64-bit PCI bus
- 33MHz and 66 MHz PCI clock speed
- up to 260MByte/s bandwidth

High-speed follow up of the Simple SLINK to PCI interface card
SLINK format example: ATLAS SLINK data format

<table>
<thead>
<tr>
<th>Beginnin of Block control word</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start of Header Marker</td>
</tr>
<tr>
<td>Header Size</td>
</tr>
<tr>
<td>Format Version No.</td>
</tr>
<tr>
<td>Source Identifier</td>
</tr>
<tr>
<td>Level 1 ID</td>
</tr>
<tr>
<td>Bunch Crossing ID</td>
</tr>
<tr>
<td>Level 1 Trigger Type</td>
</tr>
<tr>
<td>Detector Event Type</td>
</tr>
<tr>
<td>Data or Status elements</td>
</tr>
<tr>
<td>Status or Data elements</td>
</tr>
<tr>
<td>Number of status elements</td>
</tr>
<tr>
<td>Number of data elements</td>
</tr>
<tr>
<td>Data/Status First Flag</td>
</tr>
<tr>
<td>End of Block control word</td>
</tr>
</tbody>
</table>
Pulsar Top Level schematics

IO FPGA

Control FPGA

Pulsar Top Level schematics
MANY IO pins are assigned by hand to make the routing easier. Critical signals (clocks etc) are assigned by compiler with core working VHDL code.
Pulsar layout
Pulsar in pulser mode: hotlink examples:

Muon case (only one mezzanine card shown)

The latency is controlled by when the data is clocked out the FIFO
load test pattern memory:
use 36 bits data width, 32 will be for 4 fiber output (4 x 8), the highest 4 bits will be used as control bits to mark the content of data. For each event worth data, the first one will be the header, and the 32 bits data will contain the latency (&number of words etc) for this particular event and this particular path. The last one is the trailer, which can contain other info if needed (such as what L2 decision should be etc (either use internal RAM or use 128K x 36 external SRAM, CY7C1350):

<table>
<thead>
<tr>
<th>4 Ctrl bits</th>
<th>8 bits data</th>
<th>8 bits data</th>
<th>8 bits data</th>
<th>8 bits data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Buffer0 data</td>
<td>Buffer1 data</td>
<td>Buffer2 data</td>
<td>Buffer3 data</td>
<td></td>
</tr>
</tbody>
</table>

The highest two address bits will be controlled by buffer number to divide automatically the memory for 4 buffers.
How does it work:
(1) after L1A, read the first word (header) and get the latency, at the same time start a counter;
(2) continue to readout the rest of the data words from the memory and clock them into a FIFO, until the trailer is reached (can get the L2 decision information there)
(3) once the counter reaches latency threshold, clock the data out from the FIFO at the speed which matches with the subsystem.
this way the latency for each event and each data path can be individually controlled by user.

<table>
<thead>
<tr>
<th>header</th>
<th>Latency for this event, and other info</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>data</td>
</tr>
<tr>
<td>1st event</td>
<td>data</td>
</tr>
<tr>
<td>trailer</td>
<td>Other information (what L2 decision should be etc)</td>
</tr>
</tbody>
</table>

Buffer 0 data memory

One could have more control by inserting gaps in between data words...etc using the 4 control bits, to better mimic the real situation for certain data paths.

This approach seem to be quite flexible

- Ctrl bit 35: header
- Ctrl bit 34: trailer
- Ctrl bit 33: gap
- Ctrl bit 32: reserved
Initial thoughts on tester firmware design

- Latch L1A+buf#
- Read 1st word from RAM
- Save latency & compare with counter
- Continue reading data from RAM to FIFOs
- Until last word
- Once counter counts up to latency, enable FIFO output and ctrl signals for Tx chips
- Ready for next L1A

Buffer 0 data
Buffer 1 data
Buffer 2 data
Buffer 3 data
Possible implementation A:

Comments:
Pro: simple
Con: not so elegant, as the state machine has to finish sending all data out of the FIFO before able to process next one. Maybe ok if run at higher clock rate. Will be some intrinsic delay between events.

Good starting point, allow us to simulate the board soon.

Would be better to separate the RAM to FIFO part from the actual data sending part.

Peter Wittich will talk about pulser mode firmware design detail and status soon.
Possible implementation B:

Comments:
Pro: more elegant
Con: somewhat more involved.

Implement this later for the real thing.

We decided to go for implementation A first.
Pulsar in recorder mode (*readout via VME*)

hotlink examples:

(only one mezzanine card shown)

Configure the (S)RAM as a circular buffer for recording (for each L1A) and can be stopped and read out via VME.

Each Optical IO FPGA looks at 8 fiber channels, SRAM has 32+4 bits. So need ping-ponging for recording (recording is at twice the incoming data rate, 60MHz)

Natalia is working on this part of firmware
Pulsar in **general purpose** recorder mode (directly into a PC)

Firmware: this mode exercises the whole board. The core firmware has been written in this mode and we have been simulating the whole board this way since April. Will talk about the details later. The firmware in this mode can be used initially to test the prototype board by using SLINK test tools.
Pulsar in recorder mode (into a PC): firmware design is similar to all three FPGAs

- L1A Buffer(2)
- L1A(x4) queue
- FIFO
- (pulls one event worth of data at a time)
  *Checks data consistence
  *merges and stamp data
- SVT/XTRP data
- L1 trigger bits
- FIFO
- FIFO
- FIFO
- SLINK Formatter
  32-bit@40MHz
Board Level simulation example
Pulsar firmware in CVS
• common: library for lower level VHDL code
• fpgapinmap: IO pin assignment files, compile setting files, VHDL templates etc
• Rx: VHDL code in various receiving modes
• Tx: VHDL code in various pulser modes

crucial for team work and long term maintenance

Pulsar is designed for long term maintenance for Level 2
Possible applications for Pulsar

• As Pulser for L2 decision crate:
  can source any data path as well as multiple paths at the same time

• As Recorder for L2:
  can record data from upstream for any data path (or multiple paths)

• As Pulser for Hit-Finder boards in SVT system:
  with a G-LINK mezzanine card, Pulsar can pulse SVT system
  by sending fiber data into Hit Finder boards:
  * can receive SVT cable input from Hit Finder output
    (for on-the-fly loop test)

• Can sink/source G-link/TAXI/HotLink/LVDS data for generic DAQ/trigger diagnostics
• ....

Other possible/potential applications:
• As a simple standalone DAQ system (in a test beam environment with custom mezz cards)
  → can receive external trigger signals (NIM etc) via AUX card in the back
• maybe possible to use it as a general diagnostics tool for beams division…?
• Future expansion is as cheap/fast/easy as designing a mezzanine card ……
  the rest are all commercially available …

The flexible design (“lego style”) makes it possible to use Pulsar as a general purpose tool
within or outside CDF …
Flexible design: “Lego style” philosophy

One possible configuration of multiple Pulsars

The formation of Pulsar cluster
Board Level simulation and prototype testing plan

Pulsar Mezzanine cards: hotlink Tx and Rx mezzanine cards have been simulated together, has a teststand setup at UC for prototype testing.

Pulsar motherboard:

Board Level simulation:

• intensive board level simulation in progress
• prepare for multi-board simulation: Pulsar + mezzanine cards
• schedule depends on board level simulation work…
• the goal is to validate the design with INTENSIVE board level simulation by the end of June. Send out the prototype early July, and continue board level simulation…
• a new visiting student (Sakari Pitkanen) from Finland just joined this effort

Prototype testing plan:

• can use SLINK test tools first
• then test with custom mezzanine cards
Mezzanine cards board level simulation

(1) 4 fiber case first
(2) 2 fiber + LVDS case

Natalia will talk about details later

This setup can test everything except CMC connectors
Mezzanine card Prototype/production test plan I (use the working teststand setup at UC):

1. Use PG + LA;
2. Use FPGA internal RAM + LA
3. Use BIST + LA (hotlink) (run for long time and set limit on bit-error-rate to test for robustness)

This setup can test everything except CMC connectors
Pulsar + 4 hotlink mezzanine cards: prepare for multi-board simulation
Initial test plan for Pulsar board prototype
Pulsar board prototype can be first tested with SLINK test tools,
then can be tested with custom mezzanine cards

Use SLINK source card to send data
Use SLINK data sink to check data
SLIN data source test board

SLINK data Drain test board

(ideal for initial Pulsar prototype testing)
Prototype test plan II: use one Pulsar prototype board
this would allow full tests (including the CMC connectors)

Mezzanine cards production can start **ONLY AFTER** the prototypes are tested with Pulsar prototype