

Level 2 Test Stand

→ This talk is intended as a starting point to discuss L2 test stand issues within the trigger group

Feb. 22, 02

TL

So far people who have been involved
in developing the test stand tools (part time or full time):

Bill Ashmanskas

Henry Frisch

Natalia Kuznetsova

Peter Wittich

TL

UC engineer: Mircea Bogdan and Harold Sanders

L2 Review committee Recommendation (Dec. 14th, 2001):

“In the longer term, the test-stand system should be aggressively pursued. This will allow completion of the development effort and longer-term maintenance of the full system.” As recommended by the committee, we will schedule a workshop for the L2 group to discuss the specifications for this system...

“The committee thinks that the test station system, combined with the 2nd test crate sounds like the perfect way to provide various types of simulated event data (different luminosity, trigger types, suspected failure modes etc.). Providing that this effort won't impact any activities needed to make the baseline system work, it should be strongly supported, and prototyping and testing work should go ahead at full speed.”

“Hold a design workshop by the rest of the Level 2 groups in order to ensure that what is built is safe to use and is capable of exercising all the important parts of the system in a realistic fashion”.

A few words about this meeting

At Level 2 review, the committee recommended that

“this effort won't impact any activities needed to make the baseline system work”, so we have had to work a bit independently in the past.

Most of what we have done so far was to learn about various tools and to learn (and is still learning) about each subsystems to see how to integrate various interfaces into one board, to fully prepare ourselves to build the test stand.

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.

How you could use expanded teststand capabilities ?

(from this meeting agenda)

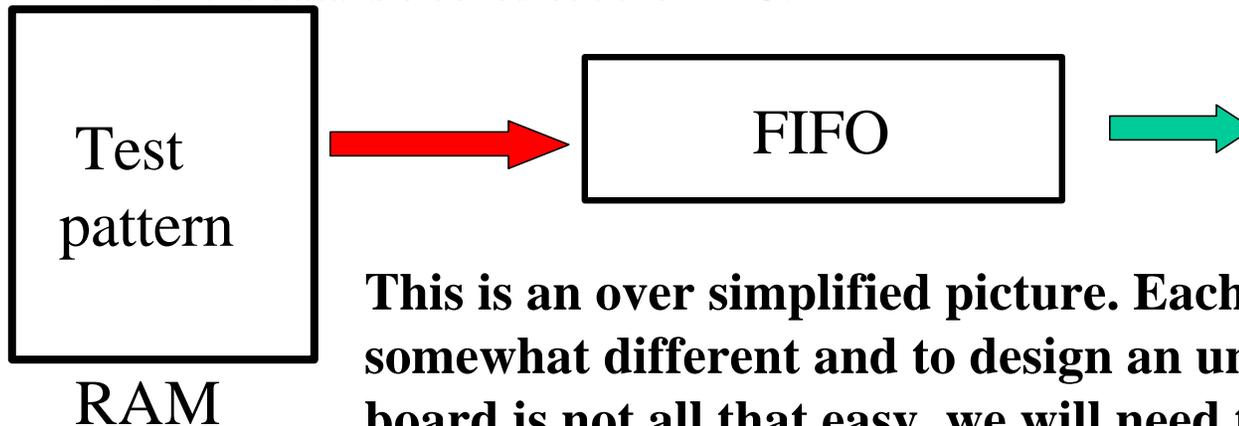
- **if we have a teststand with a general input pulser, how would you want to use it?**
 - **debugging broken/spare boards;**
 - **testing firmware modifications;**
 - **???**
- **what type of tests would you want to run?**
 - **is fixed patterns with fixed timing good enough?**
 - **data from real events?**
 - **test multiple boards and check for interference?**
 - **Randomly timed L1A patterns?**
 - **Random/user controlled latency of input data?**
 - **L1As etc driven in a deterministic way? (TESTCLK)**
 - **L1As etc driven by TS?**
 - **???**
- **what kind of software tools you will need for your testing?**

Basic requirement for a L2 data source board:

L1A for buffer n

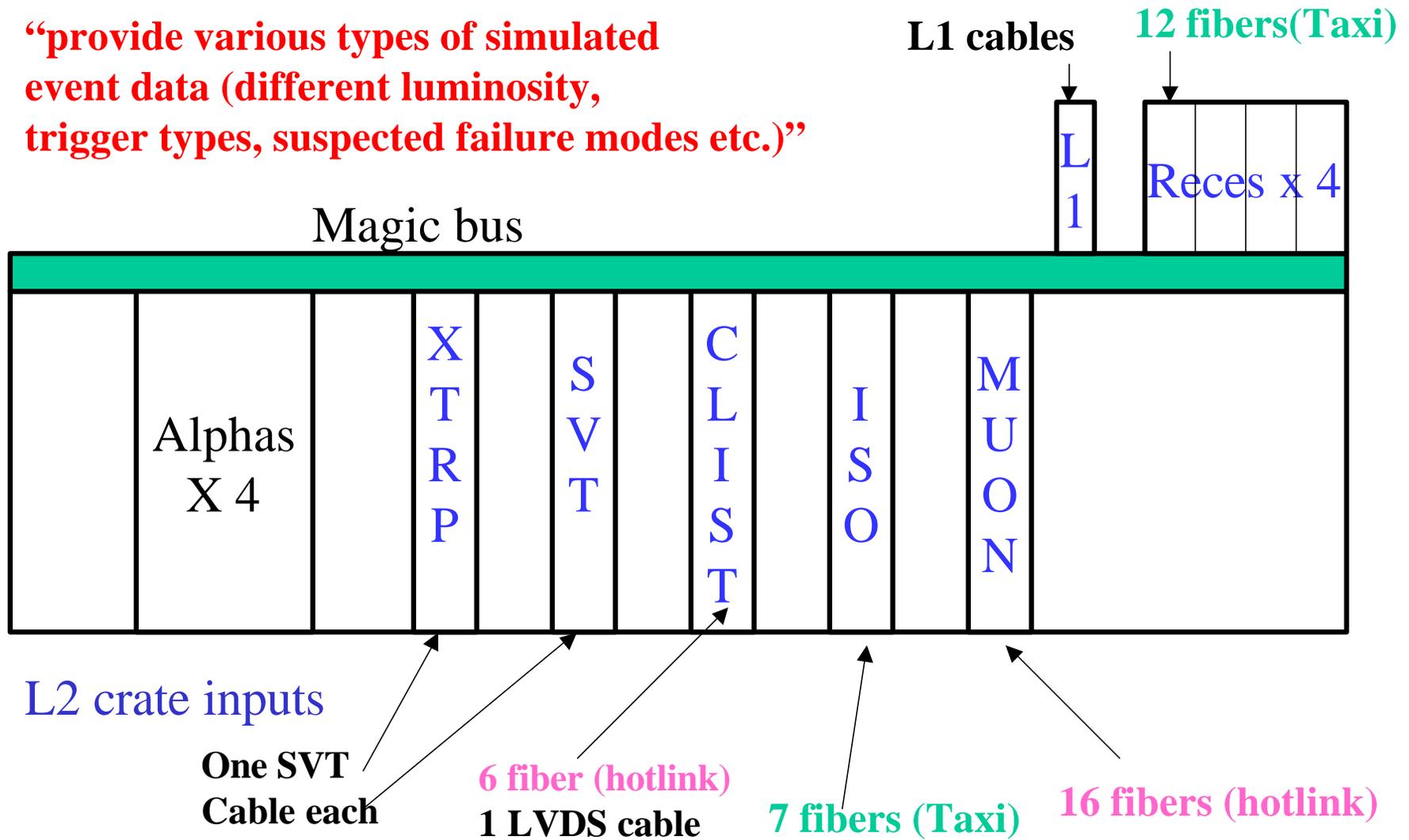


- (1) upon L1A for buffer n, start a counter for buffer n;
- (2) At the same time clock data from RAM into the FIFO,
- (3) once the counter reaches latency threshold, clock the data out from the FIFO at the speed which matches with that of the subsystem... the actual latency is controlled by when the data is clocked out the FIFO.



This is an over simplified picture. Each subsystem is somewhat different and to design an universal test board is not all that easy, we will need the support from the entire group....

“provide various types of simulated event data (different luminosity, trigger types, suspected failure modes etc.)”



can one design an universal L2 test (pulser) board?
-- to enhance the testability of L2

Let's try to fill (or fix) this table

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

* Latency range also depends on L1A history ...

Design issues for an universal test board:

Hardware requirement is clear:

- have all hardware interfaces for all data paths;

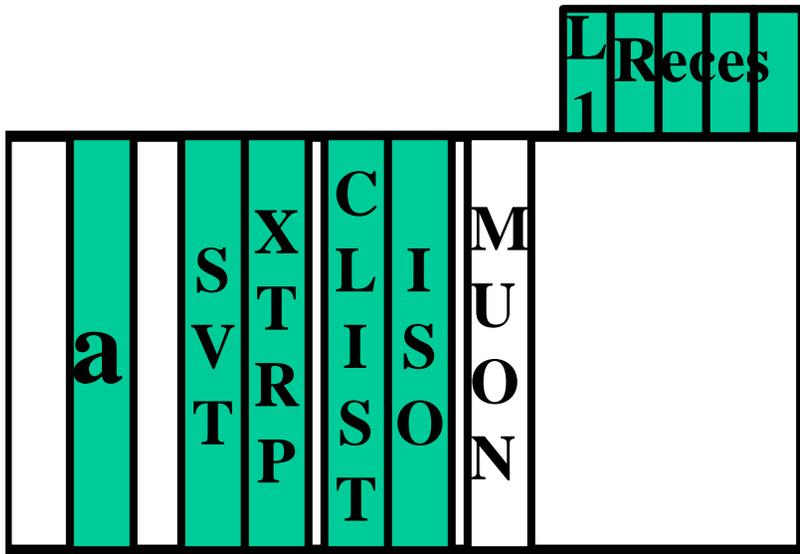
Firmware requirements need more thinking:

- variable data size for some subsystem;
- variable latency (from event to event);
- correct buffer number in the data for a given L1A;
- gaps for certain data paths;
- record real data and reproduce in test stand;
- response to HRR etc
-???

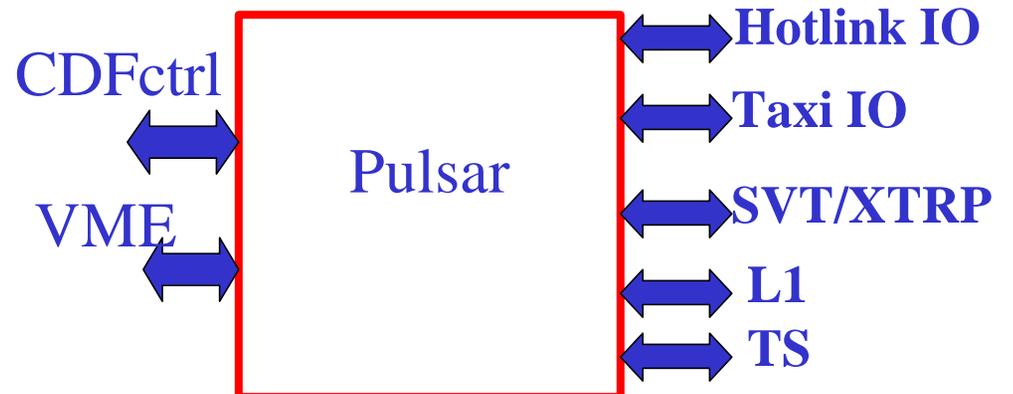
Basic hardware requirement: have all hardware interfaces

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

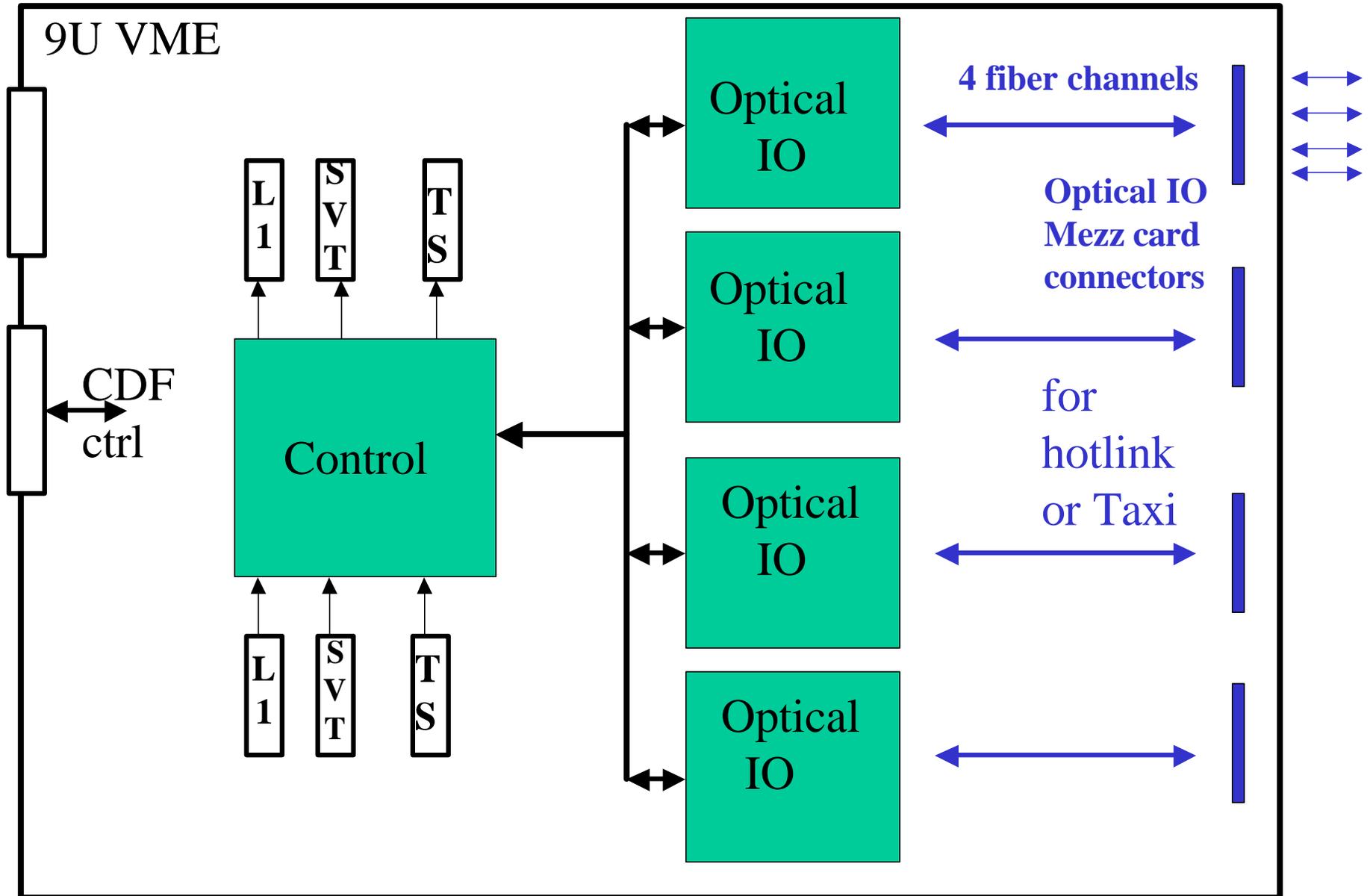
PulsAR: Pulser And Recorder



L2 decision crate



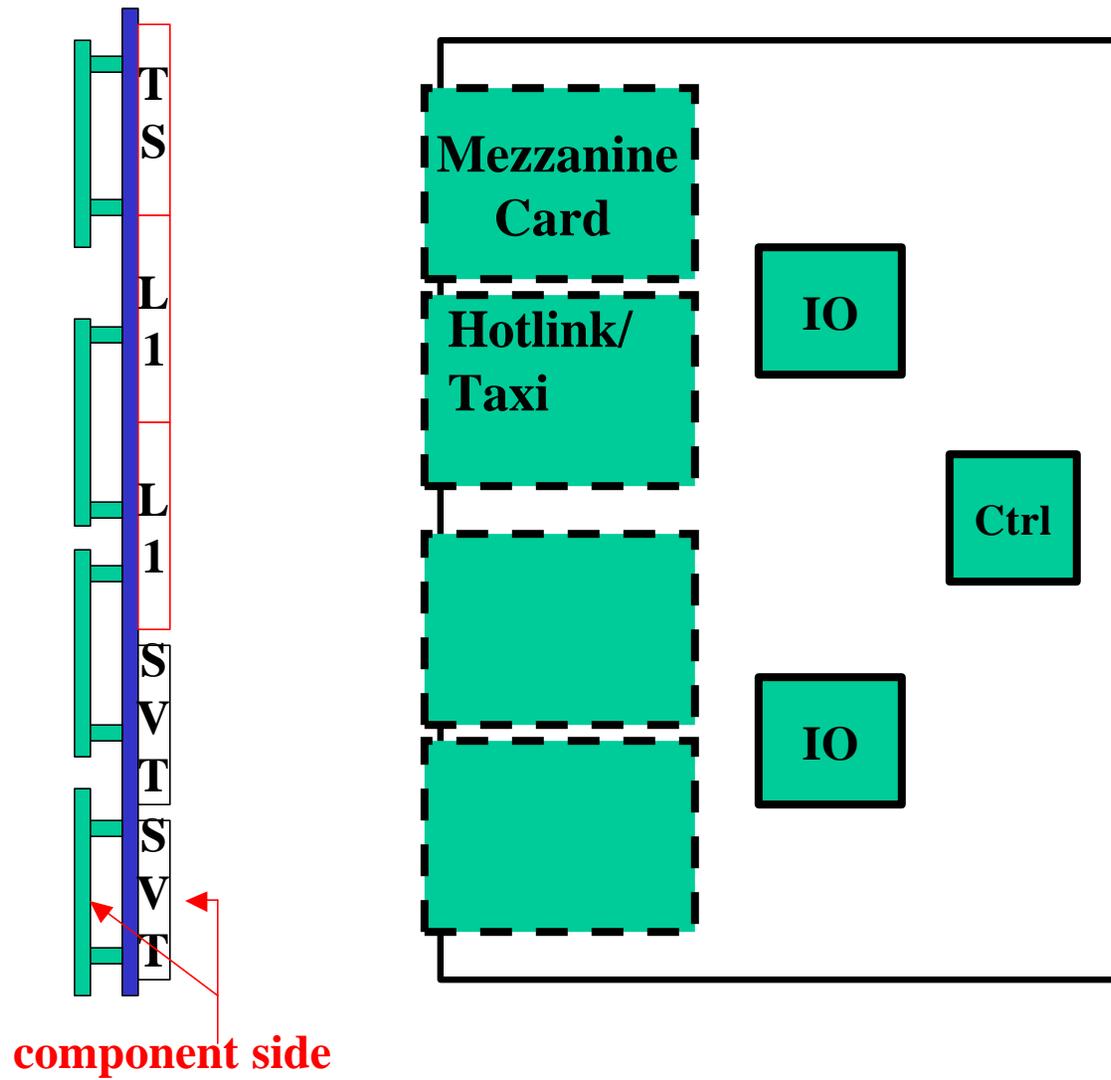
Level2_Pulsar Functional Block



Can source data, also can record data from upstream

Front-panel
(double width)

PULSAR (baseline design)



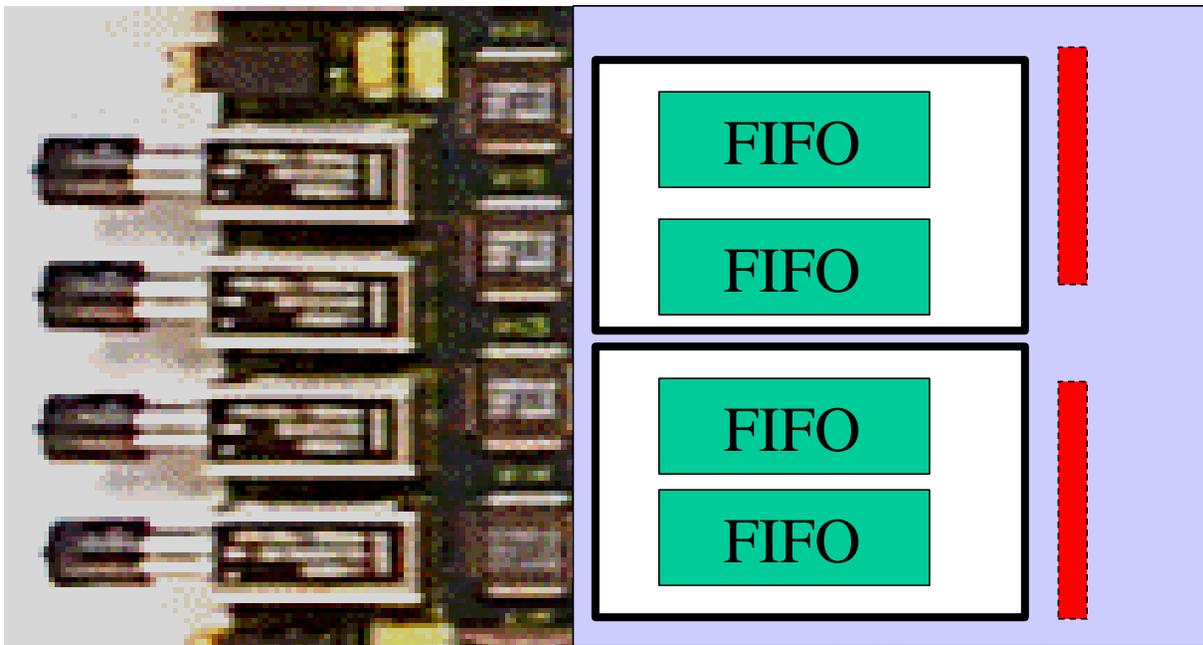
component side

Other connectors (1 L1, 1 TS) will stay inside the board.

The mezzanine card connectors are used for optical I/O (hotlink and taxi)

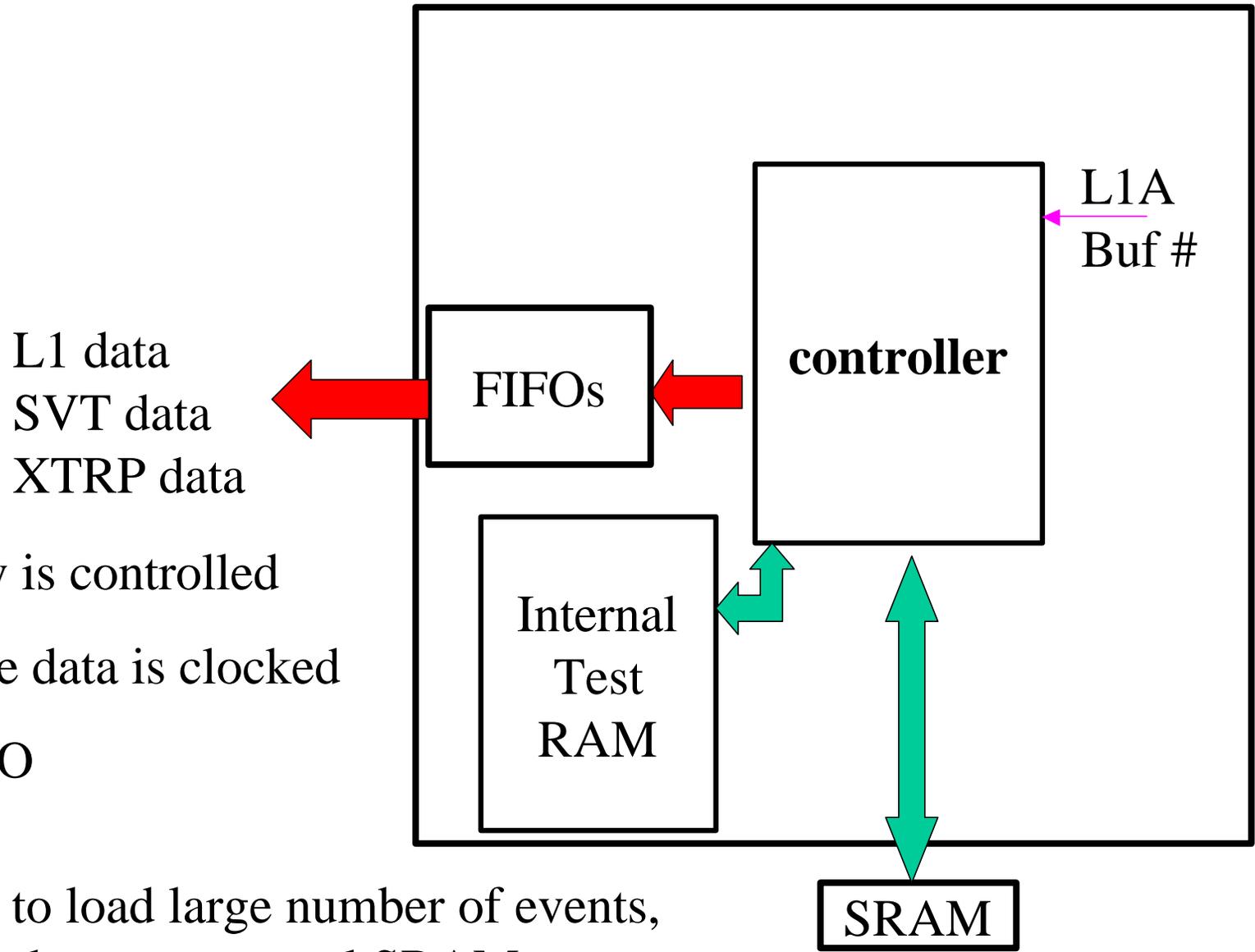
Mezzanine cards

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



**work on hotlink mezzanine cards is well underway ...(Natalia Kuznetsova).
The details are being documented and will be made available to everyone later.**

Control unit

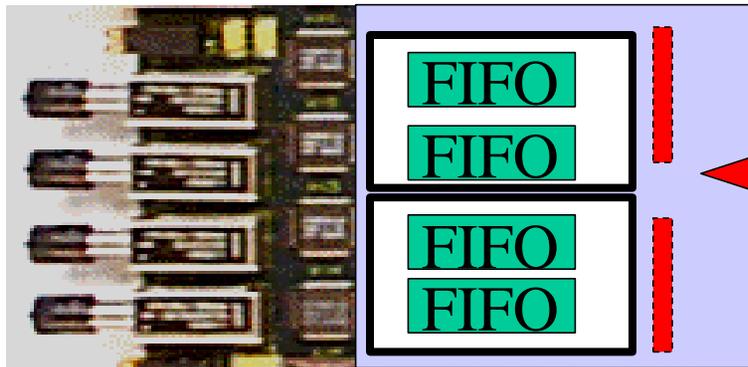


The latency is controlled
by when the data is clocked
out the FIFO

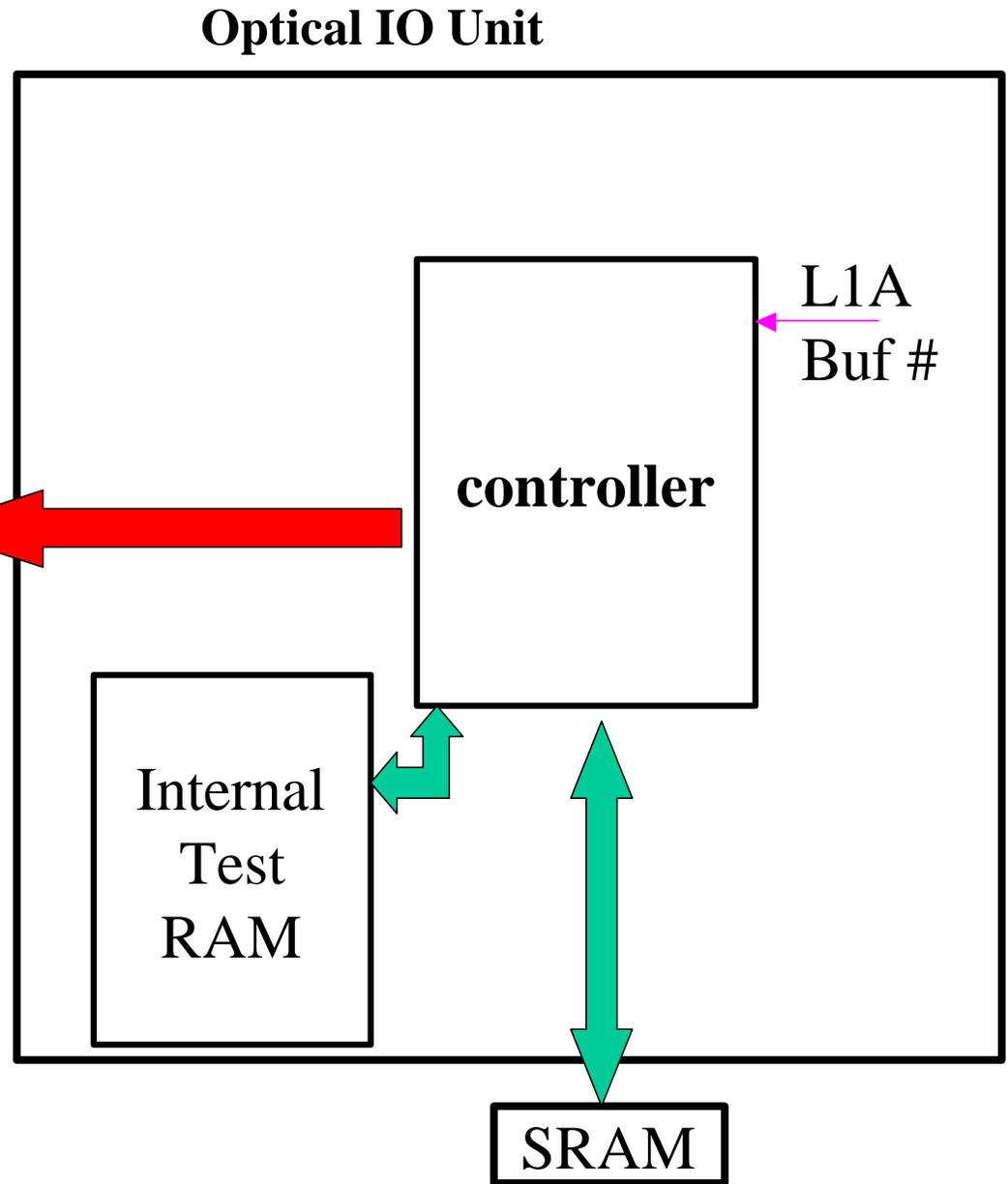
If we want to load large number of events,
we will need to use external SRAM.

hotlink examples:

Muon case
(4 mezzanine cards)



The latency is controlled by when the data is clocked out the FIFO



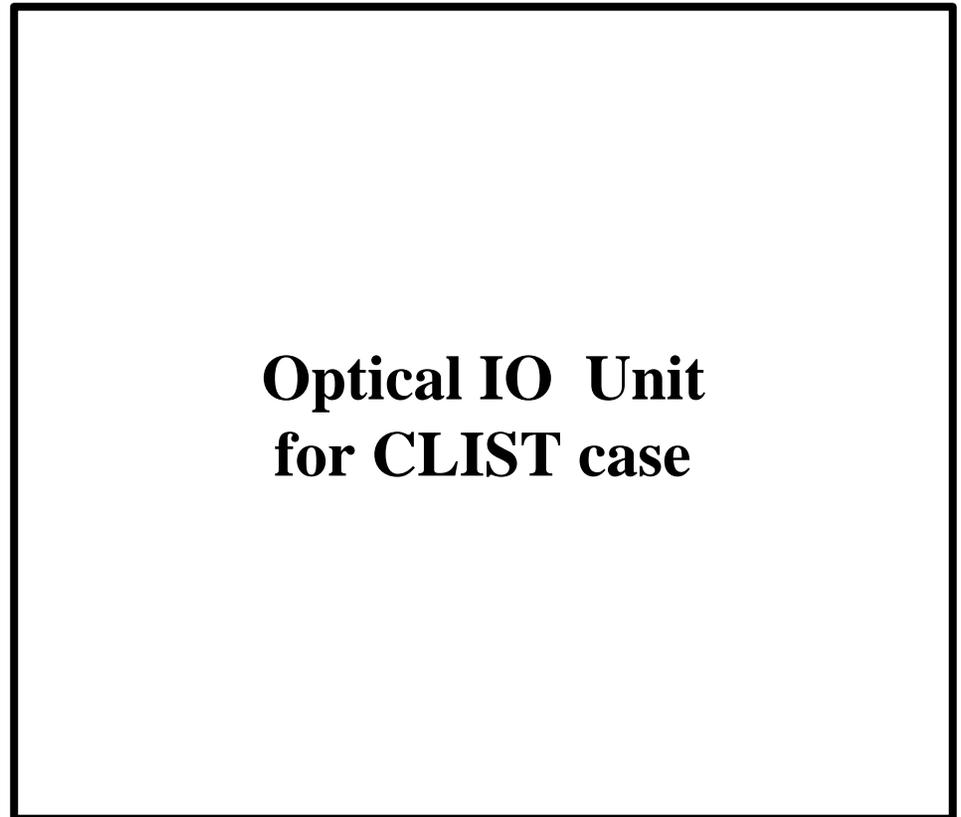
Another hotlink example

outputs: 6 fibers + LVDS



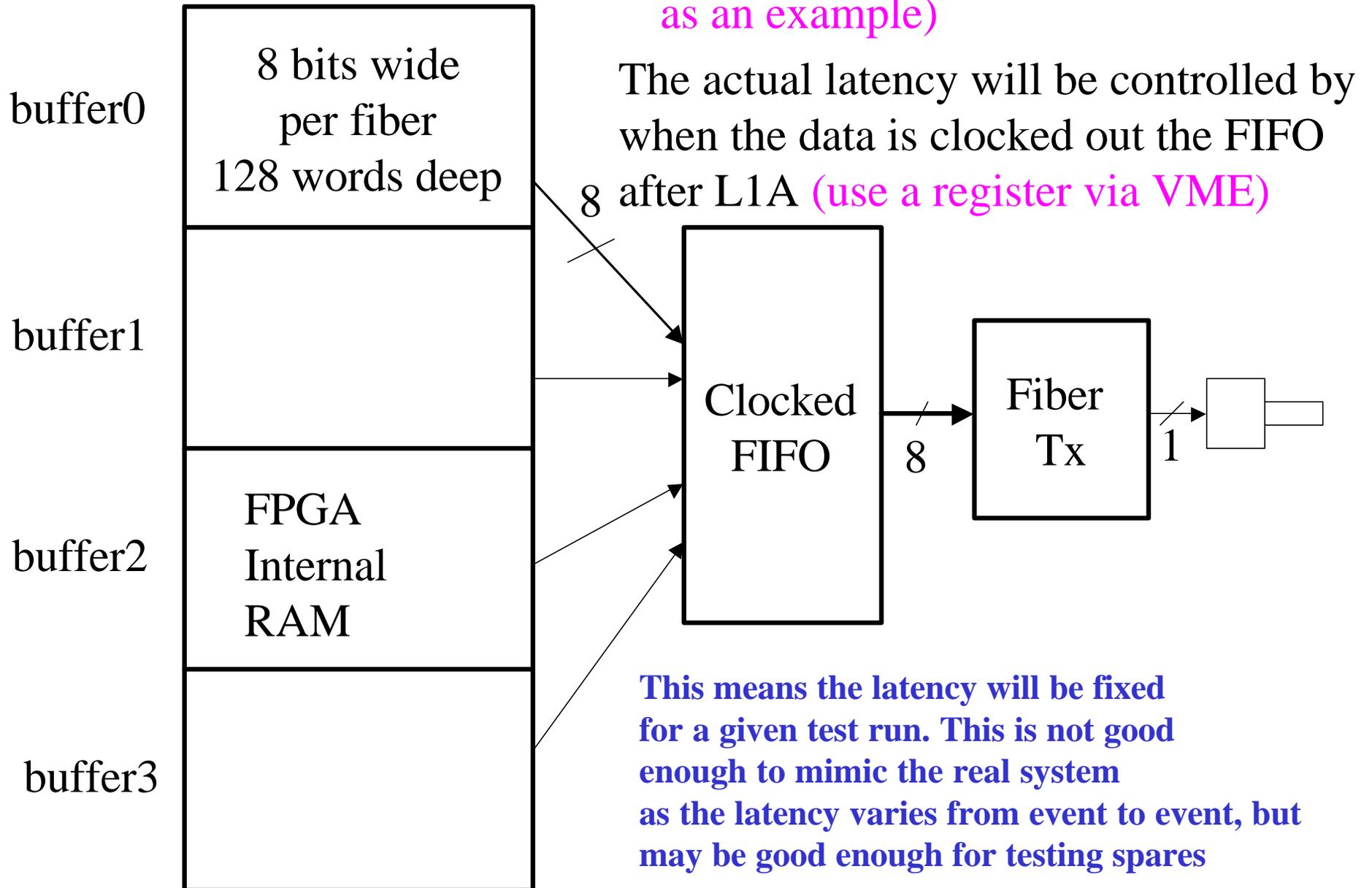
F E D C B A

8 bits each @50ns, one cluster is encoded
in 6 8-bit words in all 6 fibers



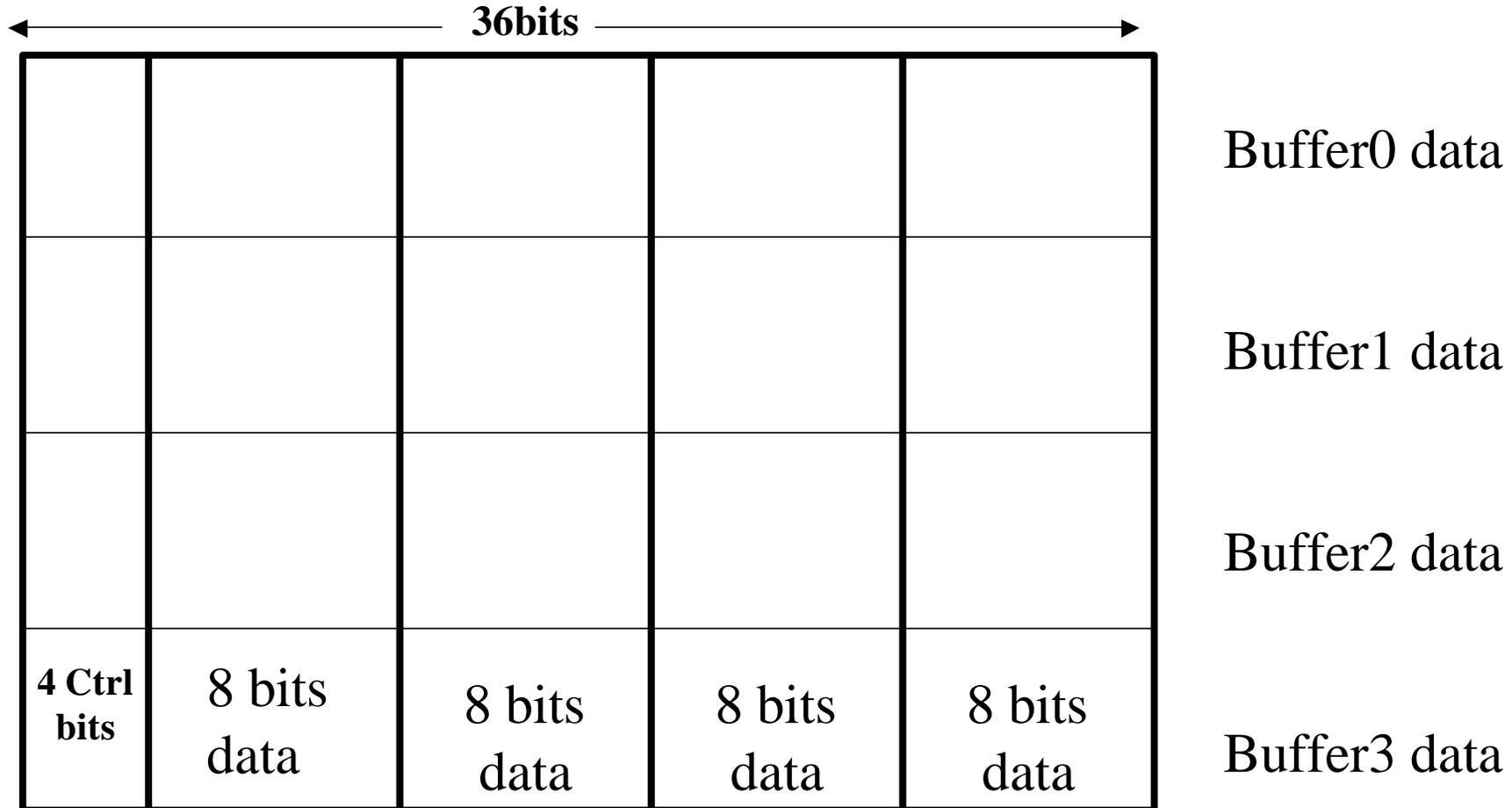
8 bits data streams will be pushed into the FIFOs in the mezzanine card after L1A,
later on they will be clocked out onto fibers. The end of event marker comes out via LVDS connector.

Simple way to load test patterns and send them out (optical paths as an example)



another way to 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 16 bit address 36 bit data external SRAM):**



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.

header	Latency for this event, and other info			
	data 1 st event	data	data	data
trailer	Other information (what L2 decision should be etc)			

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.

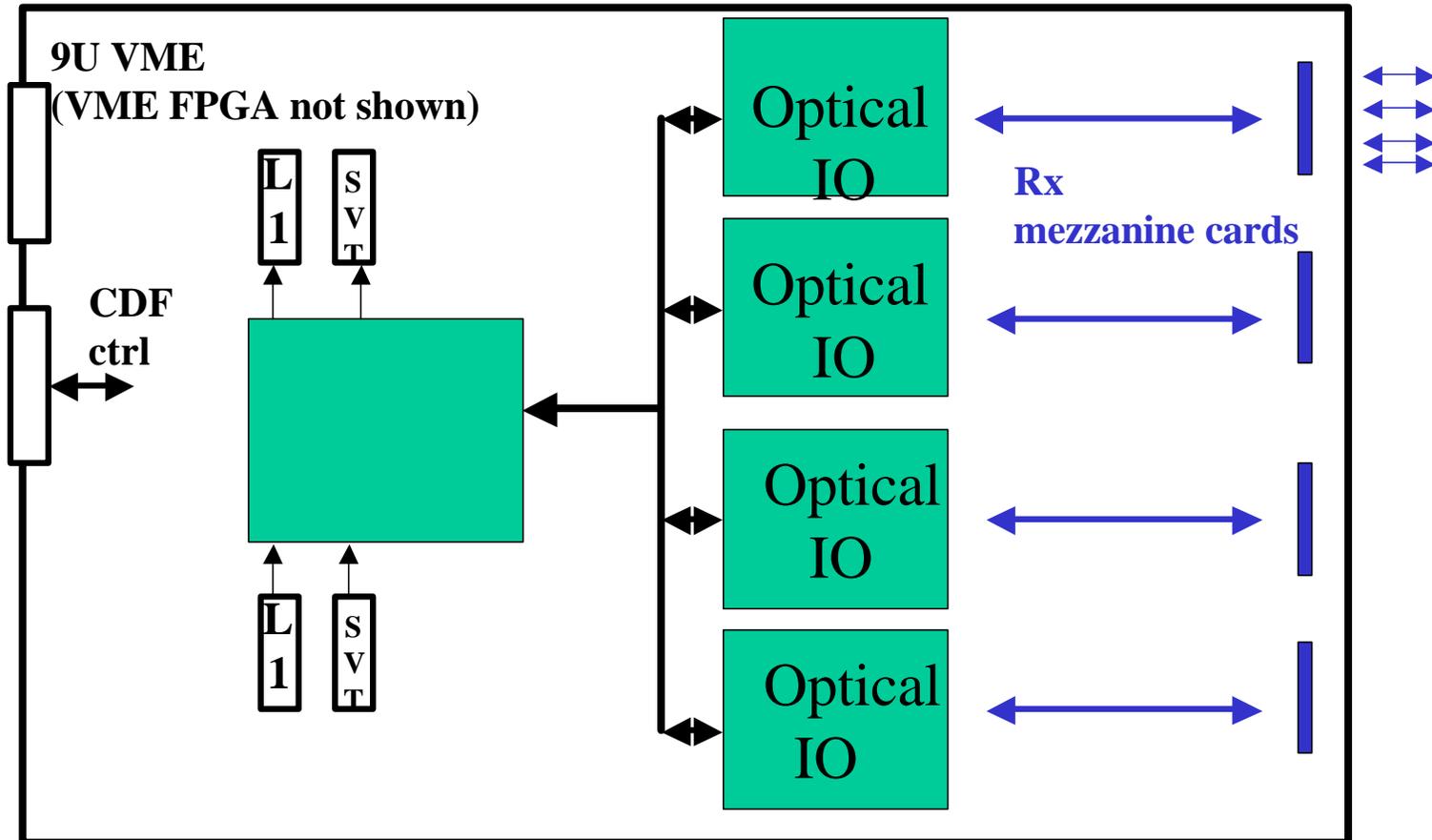
This approach seem to be quite flexible

Buffer 0 data memory

A few comments:

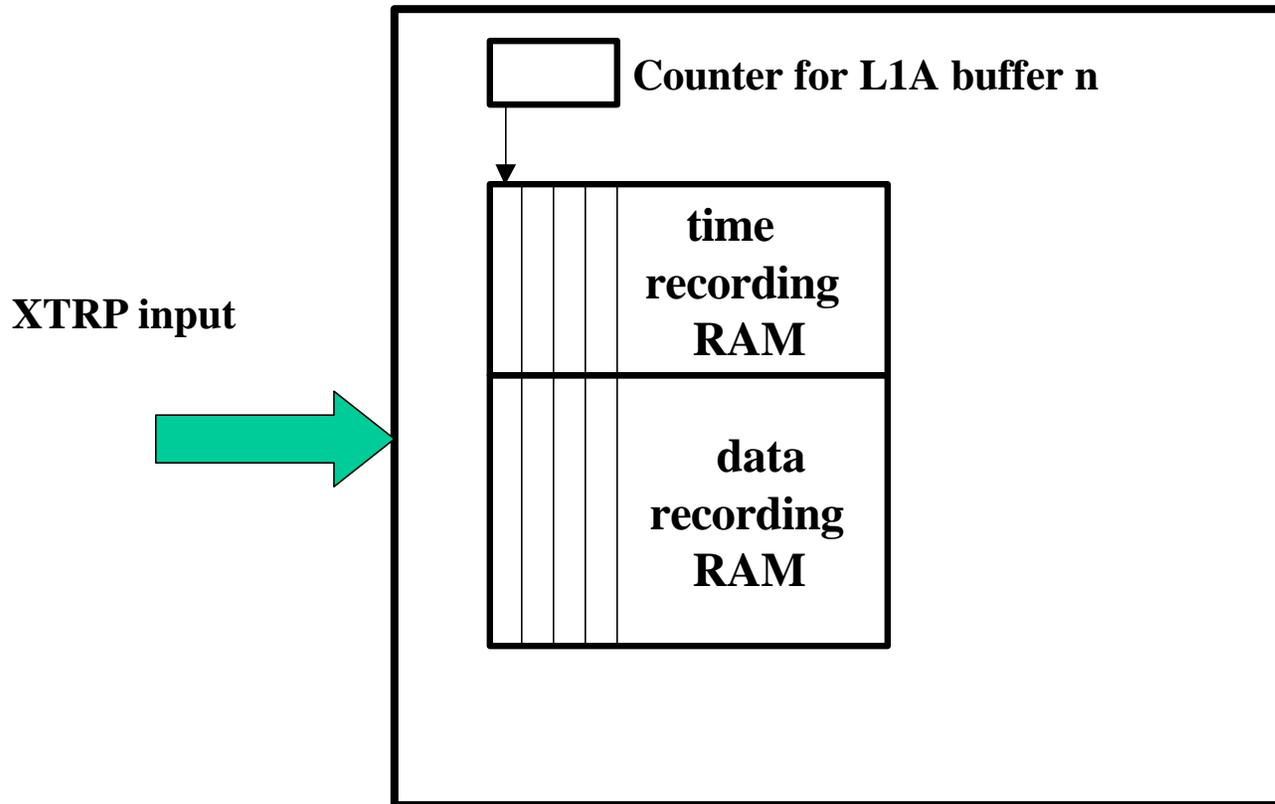
- (1) On average, the maximum data size is from muon. Each fiber can send up to $30 \times 4 = 120$ 8-bit words per event (with 16 fibers total) . If we use a 36 data bits 16 address bits SRAM, we can load up to a few hundreds of different events for muon for a given test run. Can load much more for other subsystems.
- (2) The latency for each event and for each data path (arrival time into L2 decision crate after L1A) can be controlled by user to better mimic the real system;
- (3) The long data gaps or long delays for some subsystems can be simulated this way;
- (4) **latency for individual events?**
 - * estimate based on data size (i.e. more clusters -> longer latency etc);
 - * it may be possible to record the real data with Pulsar in recorder mode and time stamp the incoming data during recording. Then save them and can later be used to reproduce the real data with real timing.
 - * note: actual latency also depends on the history of L1As.

Can use Pulsar in recording mode



**It is possible to record real data with short test runs.
If fiber splitters are used, then it is possible to even
spy on the data and to record real events, may also possible to
record problem events... shall we look into fiber splitter?**

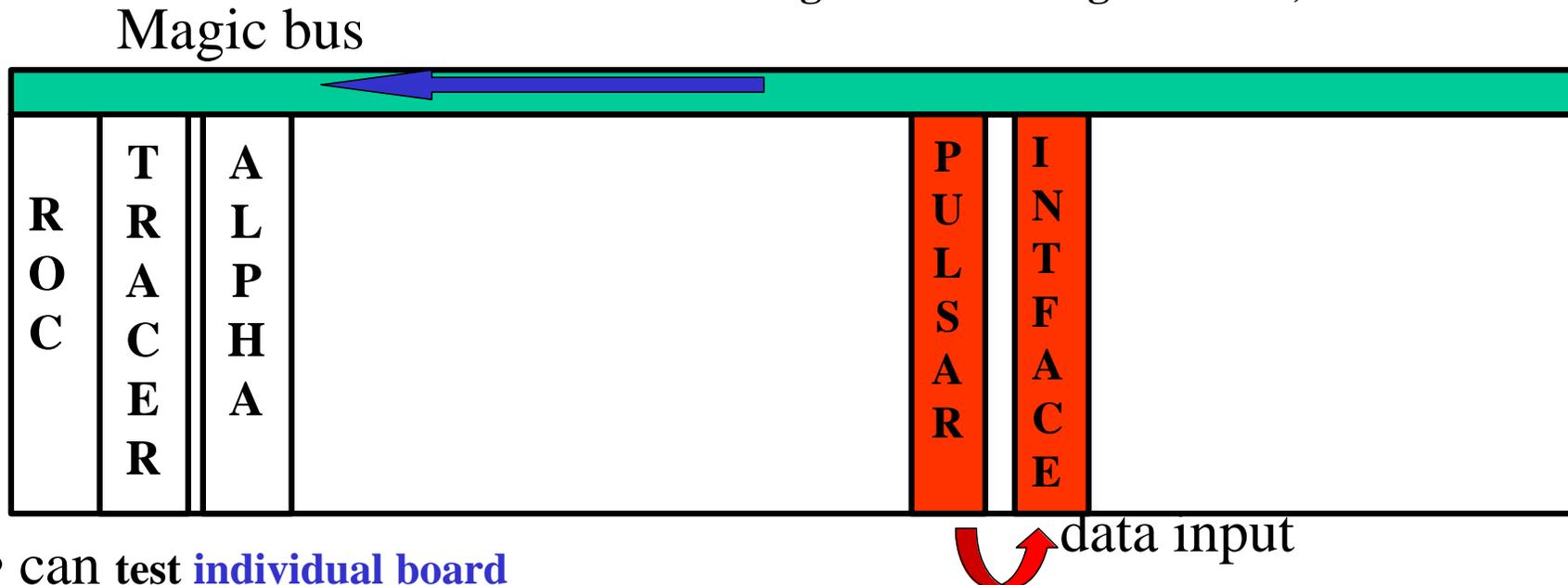
for example, it may be possible to record XTRP data with timing information: would this be useful?



Both data and arrival time can be recorded with the data strobe from XTRP. The data latency AND “gap” information can be recorded this way, and can be reproduce in test stand mode. Since Pulsar has both XTRP input and output connectors, spying on the data is possible. For fiber connections, need to use fiber splitters to spy on data, or take short test runs.

Ideal test stand setup: Alpha + Pulsar + interface board(s)

(this setup has been done already at VME speed by Steve, Matt and Peter with SVT Merger board acting as Pulsar)



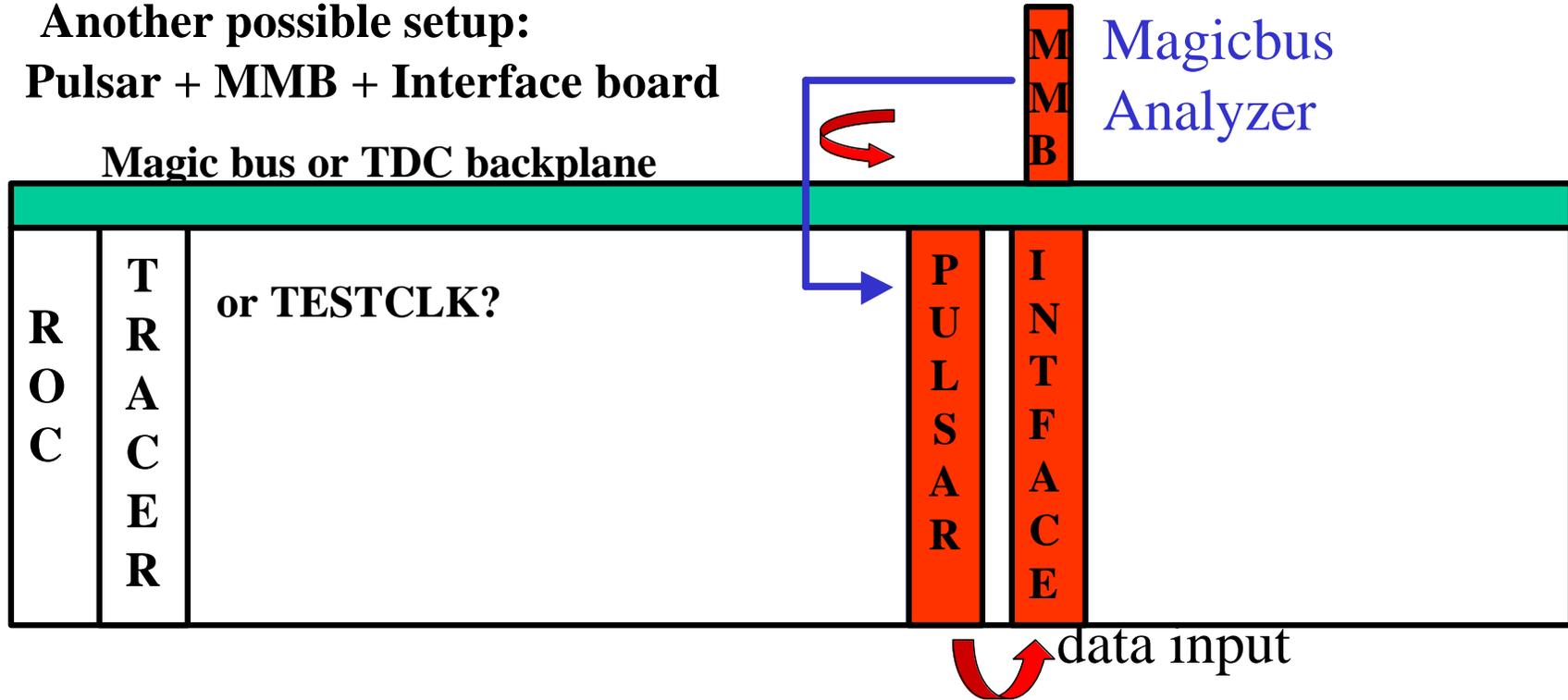
Data source: Level2_Pulsar
Data sink: Alpha

Possible data patterns:

- can test **individual board**
- can test the full data path;
- can also test **multiple boards** and check for interference
- note that with only 2 Pulsar boards one can source data at the same time for L1, XTRP, SVT, CLIST, Iso and muon. need one extra Pulsar to drive one Reces,
- what else?

- (1) hand made
- (2) derived from MC
- (3) **derived from data bank**
- (4) **recorded from upstream, catch errors and reproduce them**

**Another possible setup:
Pulsar + MMB + Interface board**



**in case Alpha is not available,
it is possible to use MMB
to sink the data and convert
into SVT data format then
send the data into Pulsar or
a GhostBuster board.**

**Data source: Level2_Pulsar
Data sink: MMB+Pulsar/GB**

details see Bill's talk.

Summary

- what's shown in this talk is what (we think) is possible to implement, and what we see this test stand can possibly do
- some of what we talked will be easy, and some of them could take more time to implement.

We need to set priority and get the basic part implemented. The important thing for this meeting is for our L2 group to come up with a realistic test stand specifications which

“allow completion of the development effort and longer-term maintenance of the full system” and

“is capable of exercising all the important parts of the system in a realistic fashion”.

Will need help from subsystem experts to understand each subsystem data format, timing requirement etc. (L1, XTRP and SVT are documented)

Have been talking to many subsystem experts, and will talk to you more in the coming month.

will show a few examples of the data format we learned from CLIST and Muon,

CLIST cluster information from one LOCOS

6 8-bits words per cluster on one fiber input, arriving 50ns apart

train no.	I	II	III	IV	V	VI
sig_0	1	em(5)	1	had(5)	1	crate_sel
sig_1	L1AB(0)	em(6)	passbit(0)	had(6)	phi(0)	ntow(0)
sig_2	L1AB(1)	em(7)	passbit(1)	had(7)	phi(1)	ntow(1)
sig_3	em(0)	em(8)	had(0)	had(8)	eta(0)	ntow(2)
sig_4	em(1)	em(9)	had(1)	had(9)	eta(1)	ntow(3)
sig_5	em(2)	em(10)	had(2)	had(10)	eta(2)	ntow(4)
sig_6	em(3)	em(11)	had(3)	had(11)	eta(3)	ntow(5)
sig_7	em(4)	em(12)	had(4)	had(12)	eta(4)	ntow(6)

Data format from Monica.

1 CLIQUE connection (via 10 pin twisted ribbon cable)

The LVDS signals are driven by a 16.7 nsec clock which is a divided-by-8 copy of the 132 nsec CDF clock:

- pin 1 BUF_DONE(0)+
- pin 2 BUF_DONE(0)-
- pin 3 BUF_DONE(1)+
- pin 4 BUF_DONE(1)-
- pin 5 CRSUM_SEND+ (not received by CLIST)
- pin 6 CRSUM_SEND- (not received by CLIST)
- pin 7 EVENT_DONE*+
- pin 8 EVENT_DONE*-
- pin 9 unused
- pin 10 unused

The time of EVENT_DONE* with respect to the last cluster found in the event is fixed.



8-bits wide cluster data:

Em(4:0), buff(1:0), 1

Em(12 : 5)

Had(4:0), pass(1:0), 1

Had(12 : 5)

Eta(4:0), phi(1:0), 1

Ntow(6:0), crate_sel

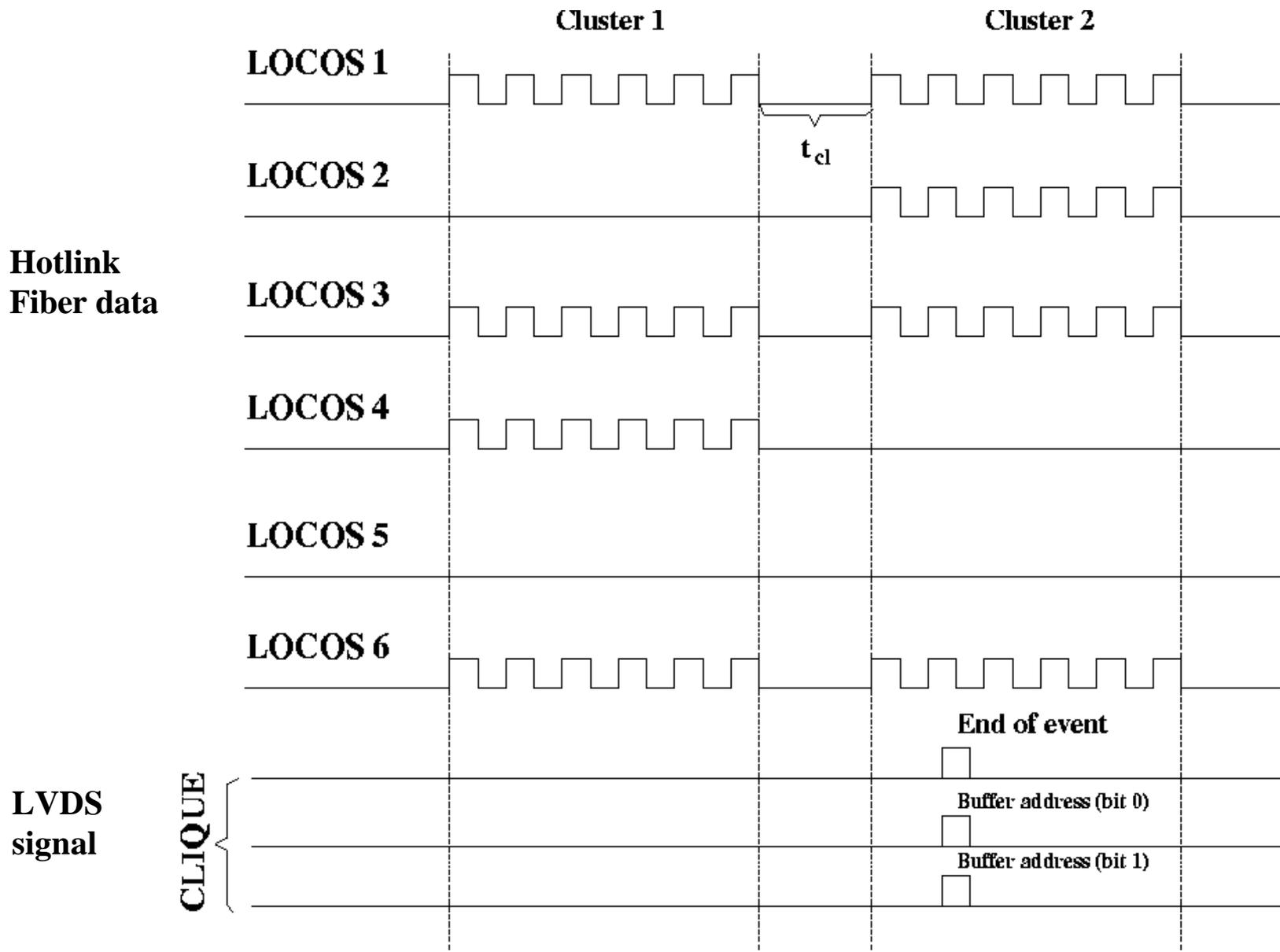
(assume this is the last cluster
for the event →)



LVDS output

Evt_done, buff(1:0)

CLIQUE control word



Information from Eric James about muon input:

Each matchbox card sends up to 32 24-bit words for each fiber. The transfer time for each word is 132 ns. Each 24-bit word is encoded into 32-bit transfer over hotlink and come in as four groups of 8-bit words.

Transfer A (1st 33ns)	Transfer B (2nd 33ns)	Transfer C (3rd 33ns)	Transfer D (4th 33ns)
-----	-----	-----	-----
bit0 - data(0)	bit0 - data(7)	bit0 - data(14)	bit0 - data(21)
bit1 - data(1)	bit1 - data(8)	bit1 - data(15)	bit1 - data(22)
bit2 - data(2)	bit2 - data(9)	bit2 - data(16)	bit2 - data(23)
bit3 - data(3)	bit3 - data(10)	bit3 - data(17)	bit3 - GND
bit4 - data(4)	bit4 - data(11)	bit4 - data(18)	bit4 - GND
bit5 - data(5)	bit5 - data(12)	bit5 - data(19)	bit5 - GND
bit6 - data(6)	bit6 - data(13)	bit6 - data(20)	bit6 - L2 Endmark
bit7 - VCC	bit7 - VCC	bit7 - Bunch Zero Marker	bit7 - GND

There is a register on the Matchbox card which gives one the ability to send zero, ten, twenty, or thirty words to L2. This feature was included in case we needed to complete our data transfer within a given time window to make the system work. The central trigger primitives are sent in the first ten words, the forward trigger primitives are sent in the next ten words, and L1 trigger decision data is sent in the last ten words. If one looks at the table in section 29.5.1 of CDF4152, the words which get sent to L2 begin with the High Pt CMU East bits (P0+3) and end with the IMB Diagnostic bits (P0+32). The output ordering of the words is the same as that shown in the table. The pre-match connections work in exactly the same way. There are only 16 24-bit words output to L2 from each pre-match card. From the table in section 29.5.2 of CDF4152, the first word which gets sent is the CMP primitives for stacks 00-23 (P0+2). The last word sent is CMP/CSP west matches for stacks 72-95 (P0+17). The ordering is the same as in the table. There are also register bits on the Pre-Match to control the number of words being sent. For this card one would transfer either zero, eight, or sixteen words.

Displacement	Data Description
P0	Header Word
P0+1	FPGA Version Numbers
P0+2	Matchbox Output Bits
P0+3	CMU East - High Pt Bits
P0+4	CMU East - Low Pt Bits
P0+5	CMU West - High Pt Bits
P0+6	CMU West - Low Pt Bits
P0+7	CMX East - High Pt Bits
P0+8	CMX East - Low Pt Bits
P0+9	CMX West - High Pt Bits
P0+10	CMX West - Low Pt Bits
P0+11	CSX and HAD East Bits
P0+12	CSX and HAD West Bits
P0+13	BMU East - High Pt Bits
P0+14	BMU East - Low Pt Bits
P0+15	BMU West - High Pt Bits
P0+16	BMU West - Low Pt Bits
P0+17	BSU East Bits
P0+18	BSU West Bits
P0+19	TSU and HAD East Bits
P0+20	TSU and HAD West Bits
P0+21	Spare Word A
P0+22	Spare Word B
P0+23	TOF Bits
P0+24	CMU East Diagnostic Bits
P0+25	CMU West Diagnostic Bits
P0+26	Eta Gap Diagnostic Bits
P0+27	CMX East Diagnostic Bits
P0+28	CMX West Diagnostic Bits
P0+29	IMA East Diagnostic Bits
P0+30	IMA West Diagnostic Bits
P0+31	IMB East Diagnostic Bits
P0+32	IMB West Diagnostic Bits
P0+33	XTRP CMU Map Bits
P0+34	XTRP CMX Map Bits
P0+35	XTRP BMU Map Bits

Muon data as an example.

Each word is 24 bits (sent as 4 8-bit words over hotlink).