Front End Chip for a L1 Track Trigger

Marvin Johnson, James Hoff, FNAL
Guido Magazzu UCSB (on leave from INFN-Pisa)

Outline

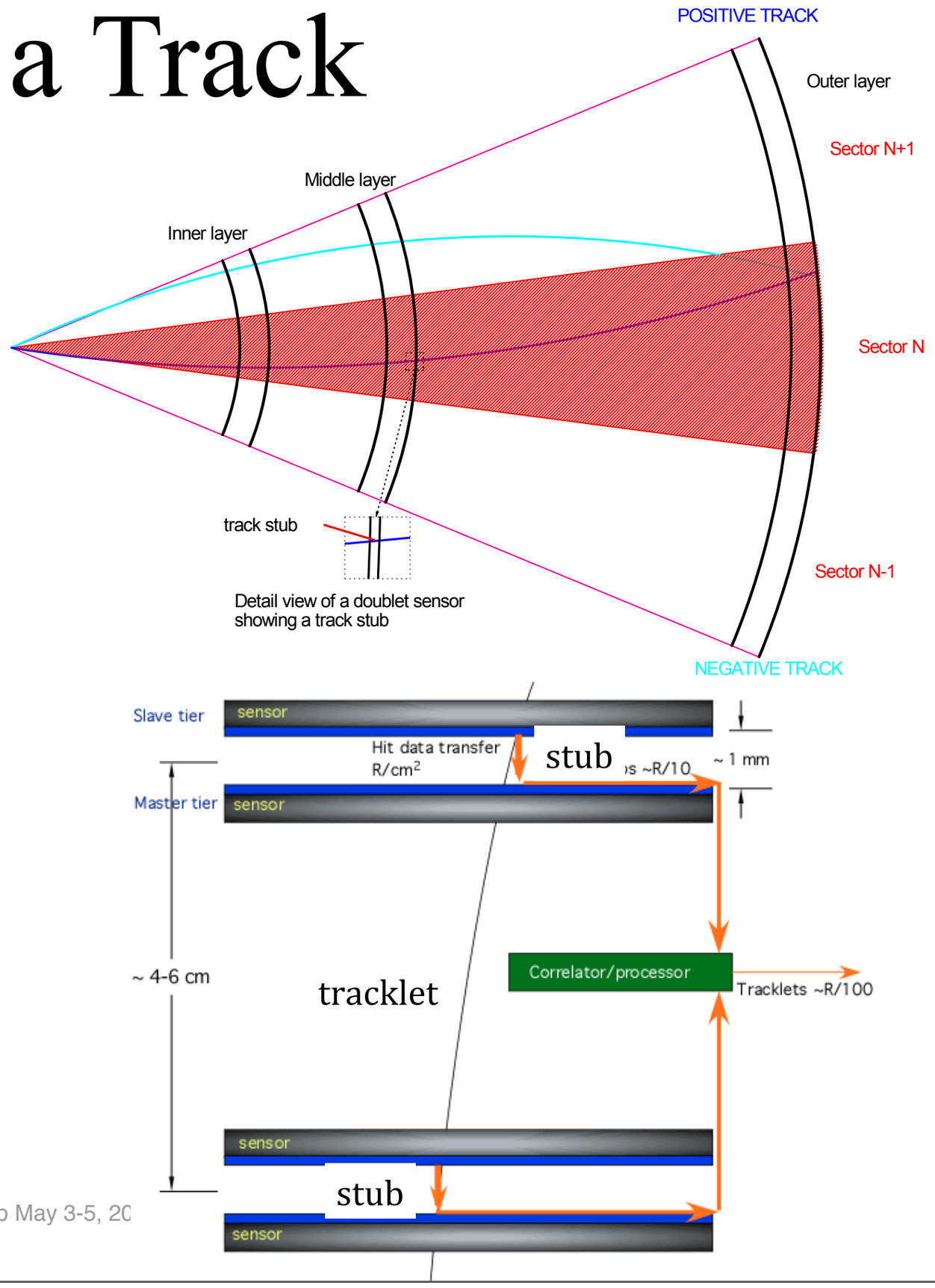
- On-detector read out chip
  - Functions - what it does
  - Methods - how it does it
- Digital chip to test the above methods
- Some results from simulations of the test chip

Tracker Triggers

- At least 2 possible methods for a trigger
  - Gather all the data and search for tracks at the end
    - CDF SVT is a good example
  - Incrementally build the track as the data is being gathered
    - This is the approach we are exploring

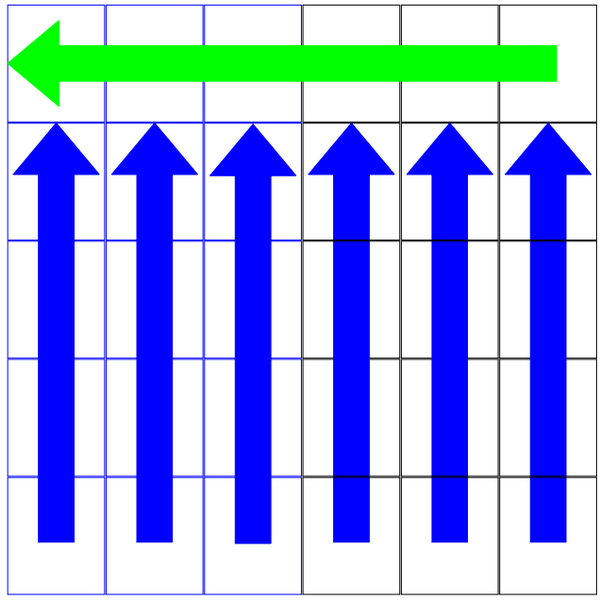
Finding a Track

- Find stubs in all 6 doublet sensors
  - Done in front end chip
  - Pt>2.5 GeV/c
- Combine stubs in each layer to form tracklets
  - Off detector processing
- Project tracklet to other 2 layers
  - Few mm accuracy
  - Match tracklets between layers



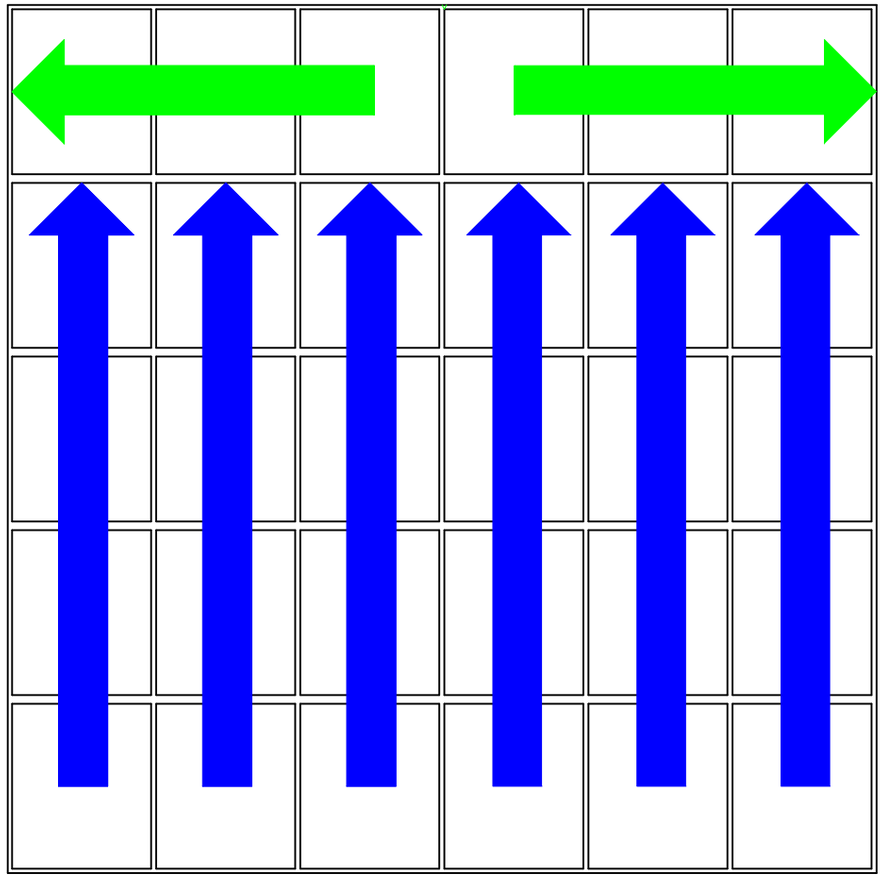
Track Projection

- Need good Z resolution for projection in Z
  - short z strips ~1.25 mm in length
  - Only need short z strips on one layer
- Requires many short z sections and ~30 chips per sensor
- Read out stubs in 2 LHC crossing intervals
  - blue path in first interval and green in second
  - Green path goes to optical driver (GBT)



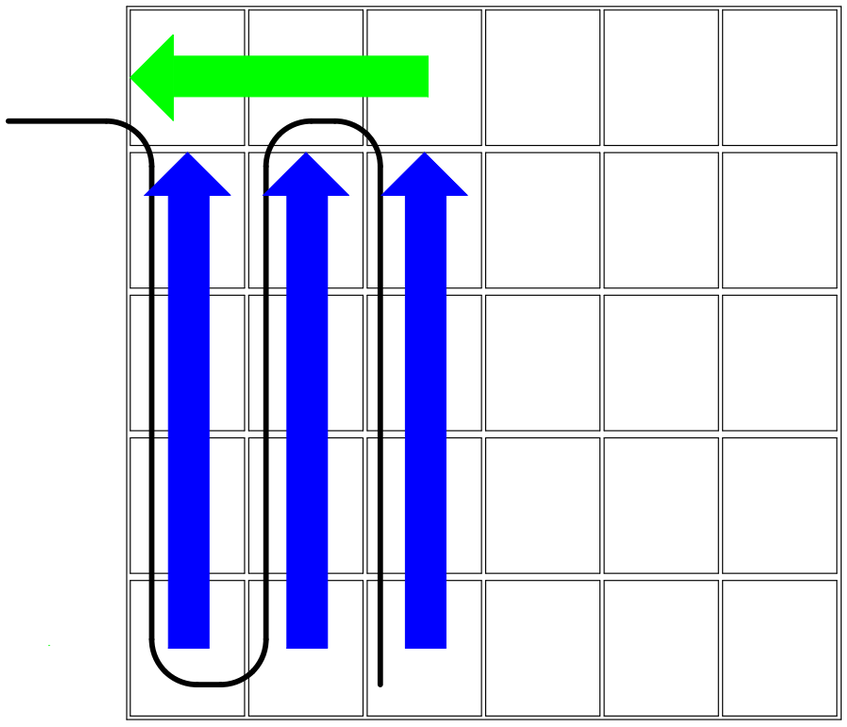
Readout Alternatives

- Inner layer has higher track density
  - Use two readouts per sensor
  - Can easily be extended to 4 readout per sensor
- Outer layers have low track density
  - Read out 2 sensors per GBT
    - Upper an lower sensor at same position on rod



Event Readout

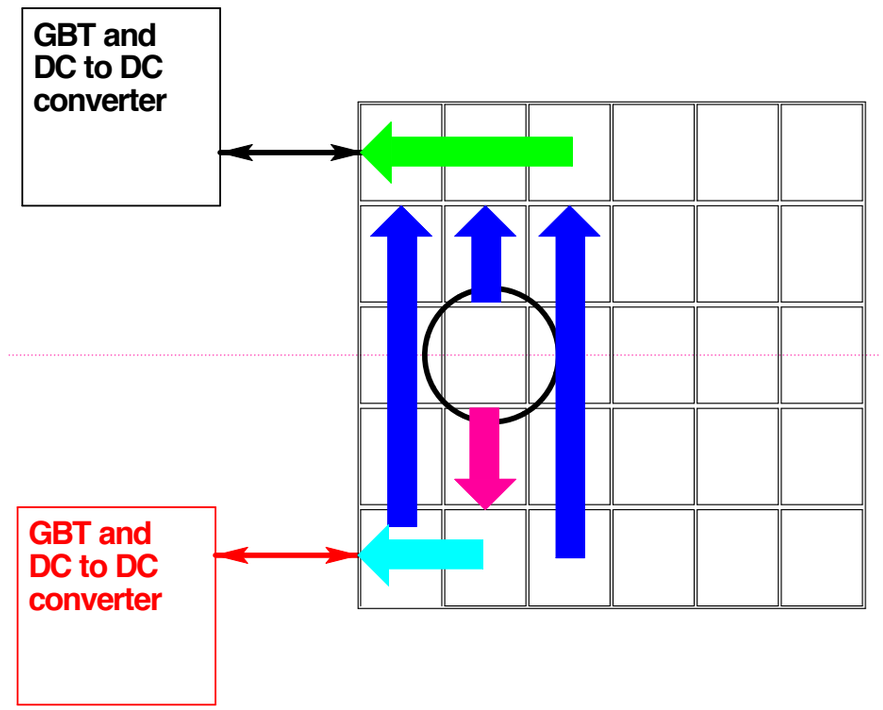
- Use same readout for both trigger and data
- Much lower rate for event than trigger so read out one hit per crossing
  - Append event data to trigger data
  - 100 KHz is 400 crossings
- Use a token to control which chip can read out event data
  - Additional control line to indicate that data was received in GBT



Black line is token path

Chip Failures

- Want to minimize impact of a chip failure
- Simple idea is to reverse direction
- Requires additional GBT and DC-DC but provides nearly complete redundancy
  - Second GBT is unpowered until needed
- Still under development

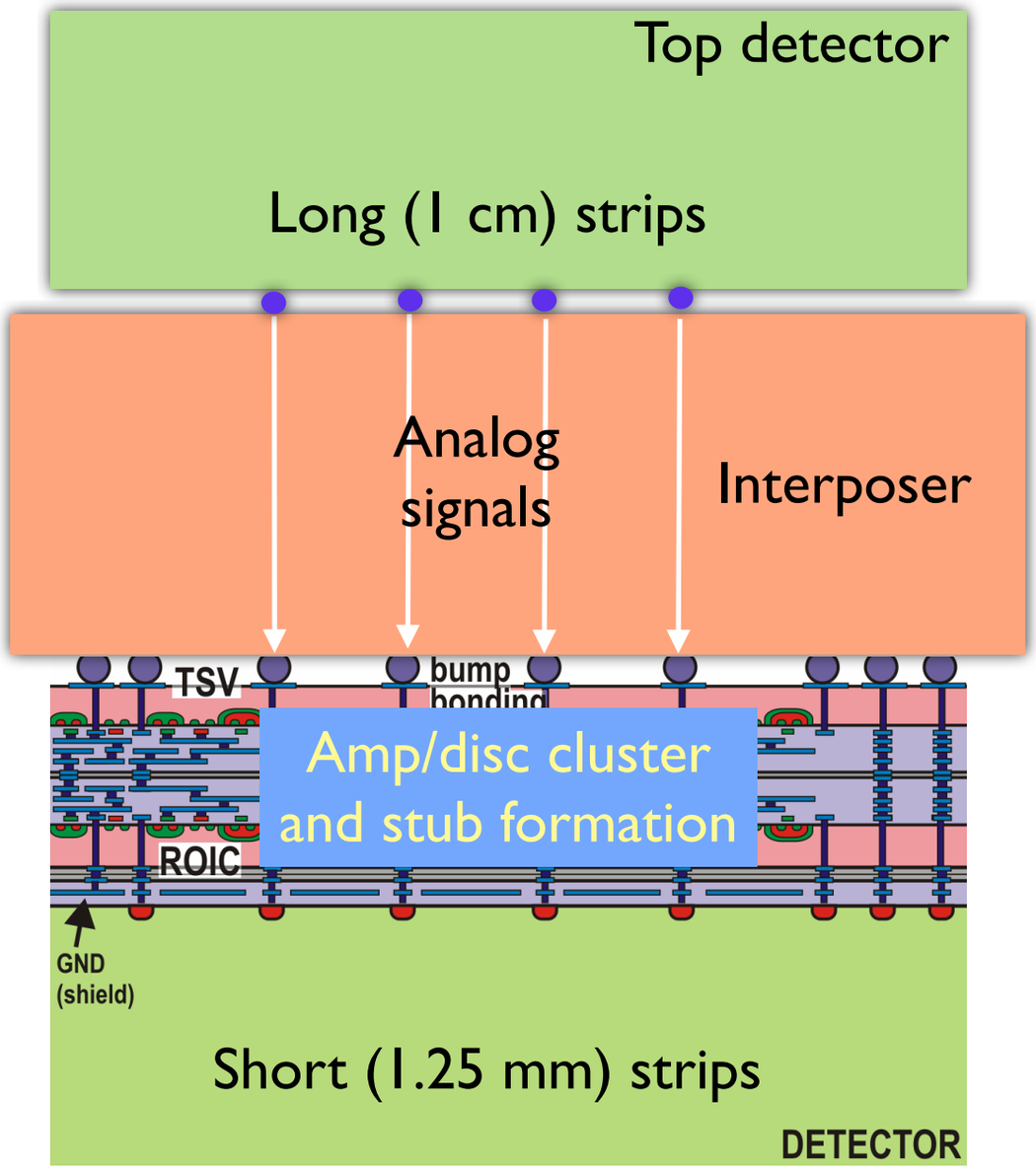


Tentative Trigger Data

- Stub address in short z strip (7 bits)
- 3 bits of Pt
- Short Z address (4 bits)
- Chip address (5 bits)
- Data/Trigger flag (1 bit)
- 20 bits total to match GBT

Chip Design Issues

- Need limited 3 D chip technology
  - Sensor inputs from top and bottom of chip
- Need low electrical noise
  - Chip is located between 2 sensors
- Need low power consumption
- Must send stub list every 25 ns



Design Goals

- Minimize the number of synchronous operations
- Only use the LHC crossing clock
- Break up calculation into 25 ns segments
- All data transfer (inter and intra chip) is done with asynchronous pipelines

Some Terminology

- Pipeline step
  - Operations performed in a 25 ns LHC crossing interval
- Micropipeline
  - Event driven (i.e., asynchronous) logic structure used to transfer data from one location to another

Asynchronous Design

- Advantages
  - Low Power
    - Circuits are inactive if there is no data
  - Low noise
    - No high speed clocks
    - LHC crossing clock is used to synchronize chip to LHC
- Disadvantages
  - Possible data corruption due to glitches and race condition
    - Much active research on designs to minimize these issues

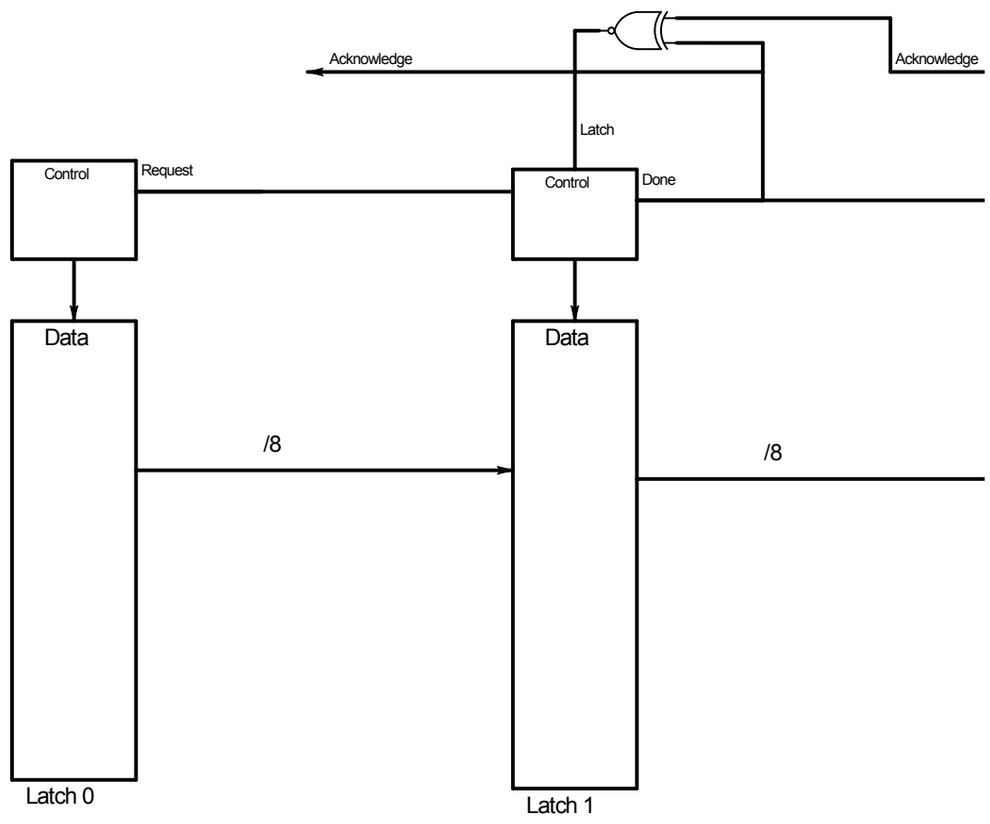
Micropipelines

- Invented by I. Sutherland
  - Received 1988 Turing award from the ACM
- Used in the SVX 2 silicon chip to compress the digital data after threshold discrimination
  - Design had some problems with race conditions so it was run at less than maximum speed
  - Still faster than the standard logic design in the SVX 4 even though the SVX 4 uses > 4 times smaller feature size

- Lots of improvements in the last 20 years
  - Simpler logic that can be implemented with standard cells
  - Methods to reduce or eliminate race conditions and errors due to glitches
- We are using the MOUSETRAP micropipeline (Columbia EE dept)
  - M.Singh and S. M. Nowick. MOUSETRAP: Ultra---high---speed transition---signaling asynchronous pipelines. In Proc. International Conf. Computer Design (ICCD), pages 9–17, Nov. 2001.
  - High speed - ~2 GHz for on chip micropipelines
  - Very simple logic
  - can be implemented with standard cell libraries

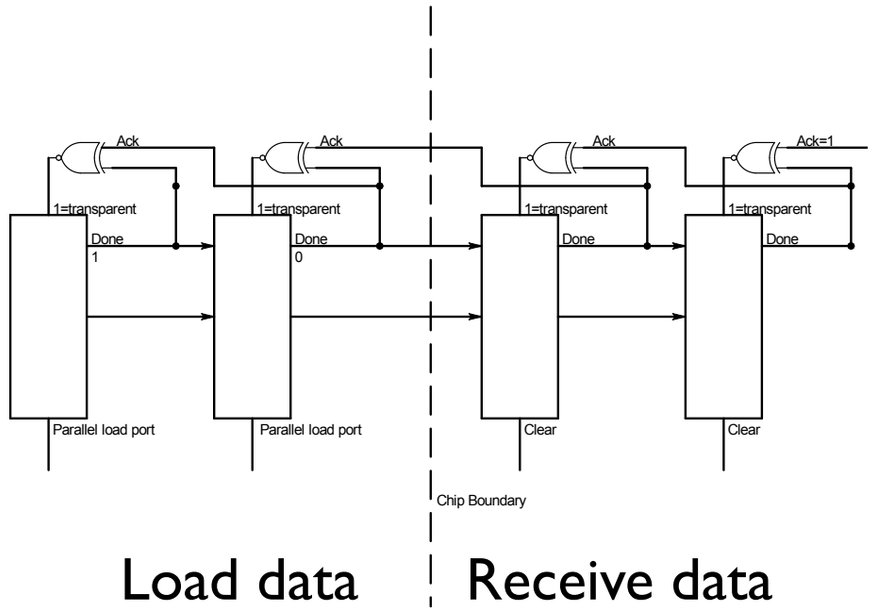
MOUSETRAP

- Latches initially in transparent mode (latch signal =0)
- Data from latch 0 arrives at latch 1
  - Sent control bit is logic 1
- Done =>1 at latch 1 and Acknowledge is 0 so Exclusive NOR => 1 and holds data
- When latch 2 Done => 1, Latch 1 Acknowledge => 1 and Latch 1 goes to transparent
- New cycle starts on Latch 1Done => 0, i.e. no return to 0 required



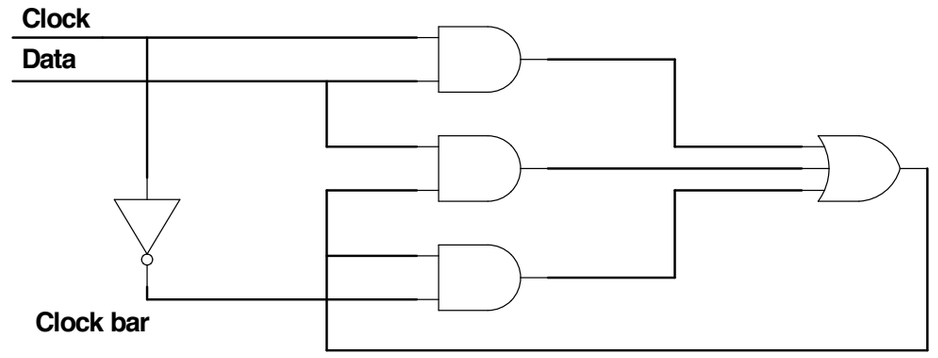
Data Loading

- Data is loaded into the pipeline in the left chip at an LHC crossing
  - Need a latch design that supports both data loading and the micropipeline
- Select the Earle latch

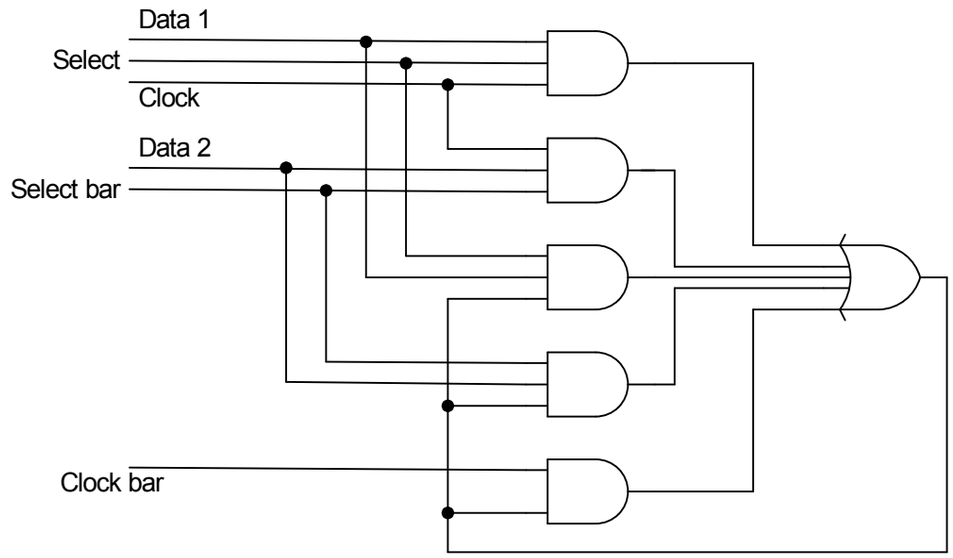


Earle Latch

- Multiplexor and a latch combined.
- Constant propagation delay independent of input
  - More important for 3 input gate described later
- Data 2 is used for loading the pipeline and data 1 for propagation
- Used for both control and data



One input



Two input

Glitches and Race

- Signal speed
  - Logic 1 may be slower than logic 0 so control may be out of synch with some data lines
  - May be significant buss skew on lines between chips
- Glitches either from noise or particles could cause false latch
- Most concerned about control logic
  - Data error rate on the same order as sensor error rate is acceptable
    - 6 independent stub measurements

Dual Rail Protocol

- Uses 2 signal lines for every logic line
  - 01 defines logic 0
  - 10 defines logic 1
  - 00 and 11 are illegal
- Eliminates the problem of different delays between 0 and 1
- Simple to implement with EXCLUSIVE OR
- Used only for control lines (pad count limit)

Other Solutions

- Most common solution is the Mueller C element
  - Used in original Sutherland design
- Simple state machine may also be possible
- Still looking at possible options

Chip Calculations

- Next few slides will describe how the stub calculations are done
- Some of the logic descriptions have been simplified for the presentation

Chip Calculations

- Form Clusters of 3 or fewer strips on upper and lower sensors
- Compare cluster centroids in upper and lower sensors to find stub Pt > 2.5 GeV/c
- Encode up to 4 high Pt stubs for each set of Z strips (strip address, Pt and sign)
  - Monte Carlo simulations indicate that 4 stubs/sensor is adequate
  - Divide sensor in two for inner layer
- Select max of 4 stubs/chip for off chip transmission

Clusters and Stubs

- Simple combinatorial logic
- Both require transmission of strip hits between chips (phi and Z direction)
  - Transfer is by a micropipeline (described below)

Cluster logic

$$\overline{n-1} \bullet n \bullet \overline{n+1}$$

$$\overline{n-1} \bullet n \bullet n+1 \bullet \overline{n+2}$$

$$\overline{n-2} \bullet n-1 \bullet n \bullet n+1 \bullet \overline{n+2}$$

$$\overline{n-2} \bullet n-1 \bullet n \bullet n+1 \bullet n+2 \bullet \overline{n+3}$$

$$\overline{n-3} \bullet n-2 \bullet n-1 \bullet n \bullet n+1 \bullet n+2 \bullet \overline{n+3}$$

Stub logic

$$sz_n \bullet lz_{n+3}$$
$$sz_n \bullet lz_{n+3}$$
$$sz_n \bullet lz_{n+2}$$
$$sz_n \bullet lz_{n+1}$$
$$sz_n \bullet lz_n$$

$$sz_n \bullet lz_{n-1}$$
$$sz_n \bullet lz_{n-2}$$
$$sz_n \bullet lz_{n-3}$$

Encoding

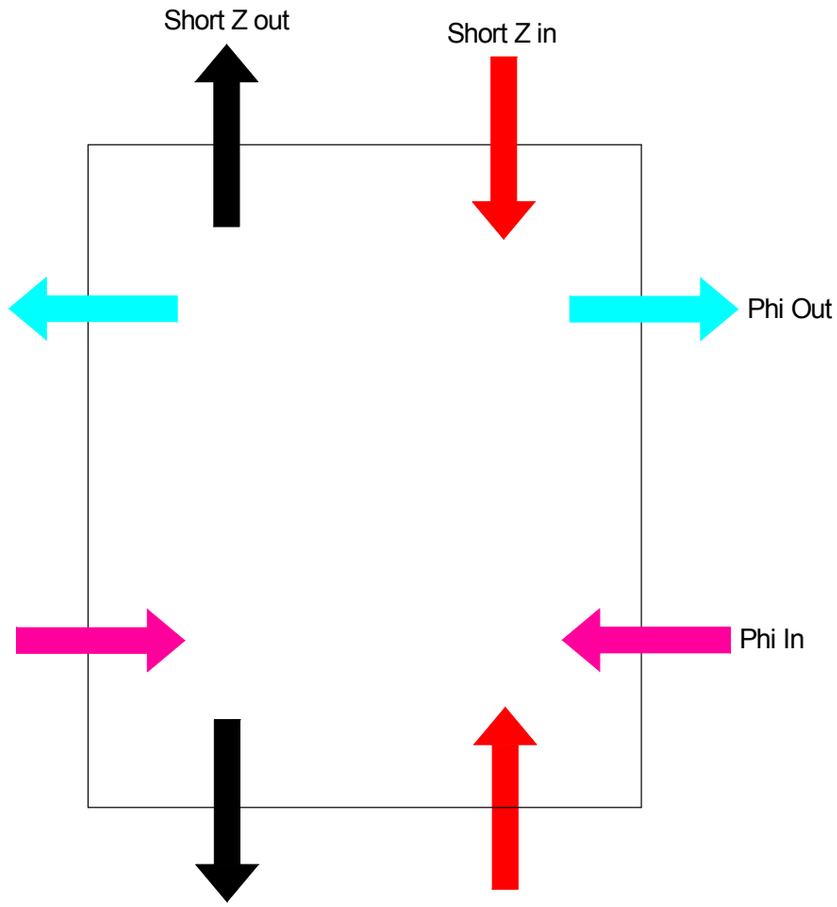
- Need to select stub addresses and Pt value from 128 channel blocks
- Not enough time for logic loops
- Adopt “Mephisto” style priority encoder
  - P. Fischer et al, NIM, A 431(1999) 134 and P. Fischer, NIM,A 461 (2001) 499-504
  - Priority encoder with feedback to select items associated with the encoded hit (Pt in this case)

Encoding II

- Use 4 sets of encoders for each short z section
  - find highest and lowest address in 128 strip section
  - Use feedback to gate off the highest and lowest address and send the reduced list to second set of 2 encoders
  - Final result is the highest, lowest, second highest and second lowest address
- Identical method is used to select 4 stubs from the collection of short z stubs

Inter Chip Data Transfer

- Phi Hits - 2 strips for short z and 5 for long z
  - Long z includes strips for Pt calculation
  - 30 bits for 10 short z and 2 long z strips per chip (6 bits wide by 5 registers deep)
  - One transfer in each direction on right and left edges
- Z hits
  - Short z clusters at chip boundary
- All transfers are done with micropipelines

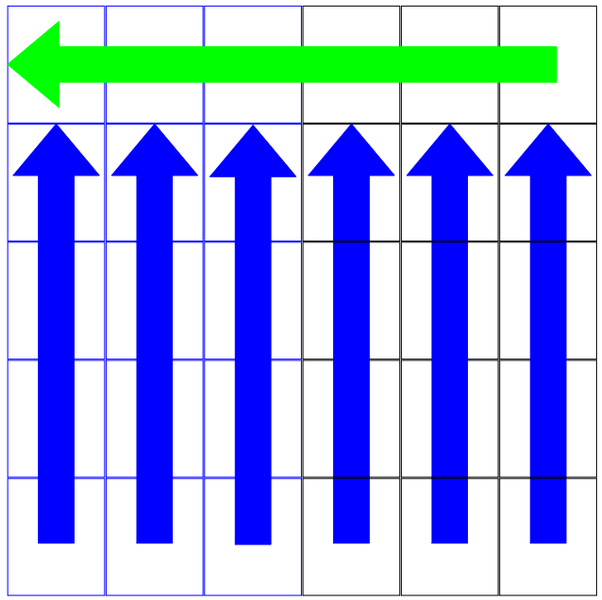


Micropipeline Operation

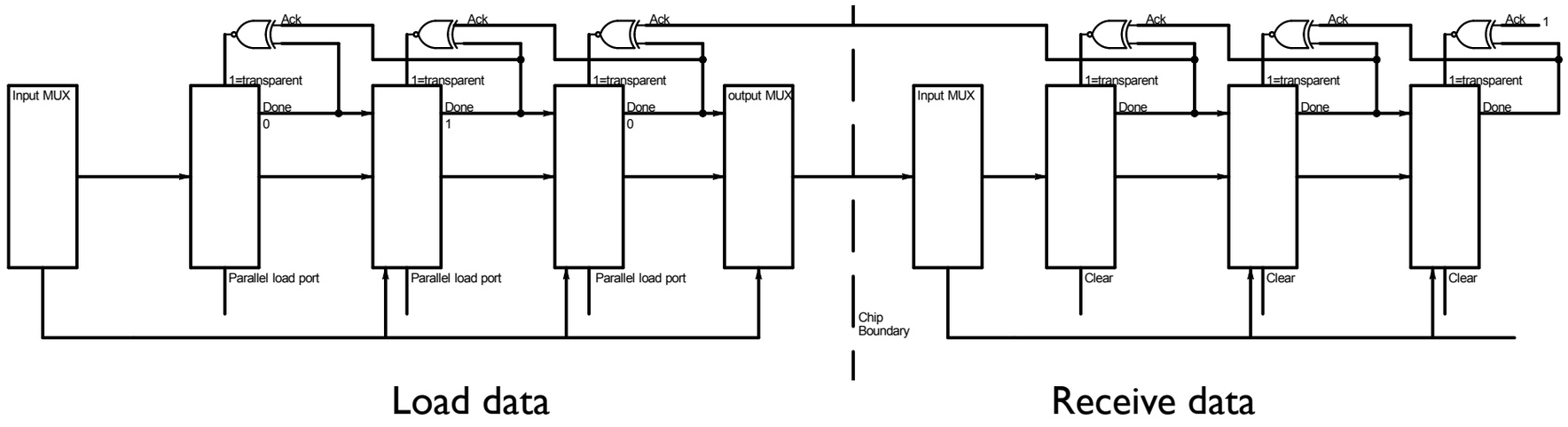
- Each transfer takes 2 operations
  - Micropipeline load
  - Micropipeline transfer
  - Data is saved in the destination chip on the next crossing clock
- Two possible timing methods
  - Asynchronous - add an extra load bit and start transfer when AND of all load bits equals one
  - Employ 2 micropipelines (but one set of chip pads) with one micropipeline being loaded while the other one is transmitting
- Current design uses the second method

Data Readout

- Data readout is more complicated than phi or Z transfer
  - Maximum number of registers in the micropipeline is 16 but we keep only 4
  - Number of stubs for a column is unknown at the start of transfer
  - Not enough bandwidth in either the pipeline or the GBT to send zero data
- Design a variable length micropipeline
  - Only occupied registers appear in the pipeline



Variable Length Micropipeline



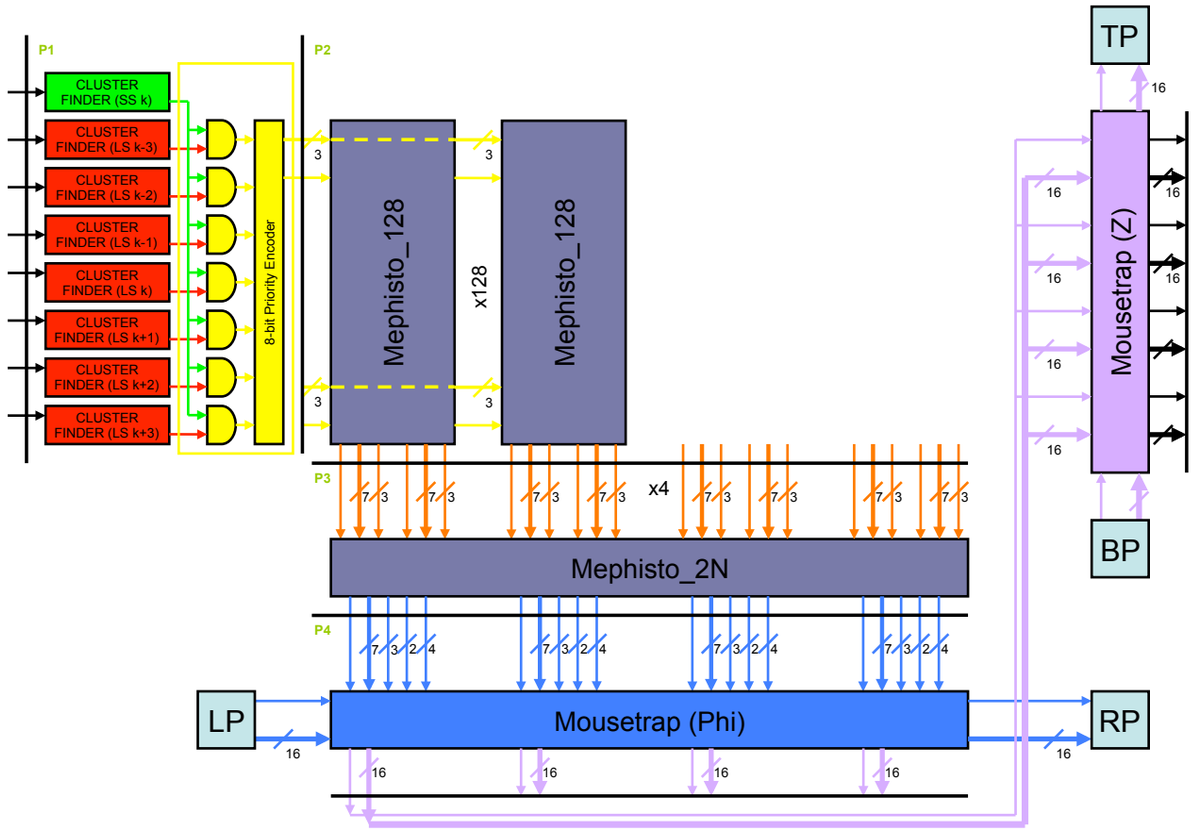
- Only valid data loaded into left chip reg.
- Input MUX routes data from previous chip to reg. with the last valid data or directly to output MUX if chip has no data
  - Requires 3 input Earle gate
    - load data
    - data from previous micropipeline register
    - data from input multiplexor

Micropipeline

- Receiver register only takes 4 stubs
  - Monte Carlo indicates a max of 4 stubs/sensor
- Number of stubs transferred is found from the control bits
- Second pipeline step then transfers the column results to the GBT for transmission off the detector

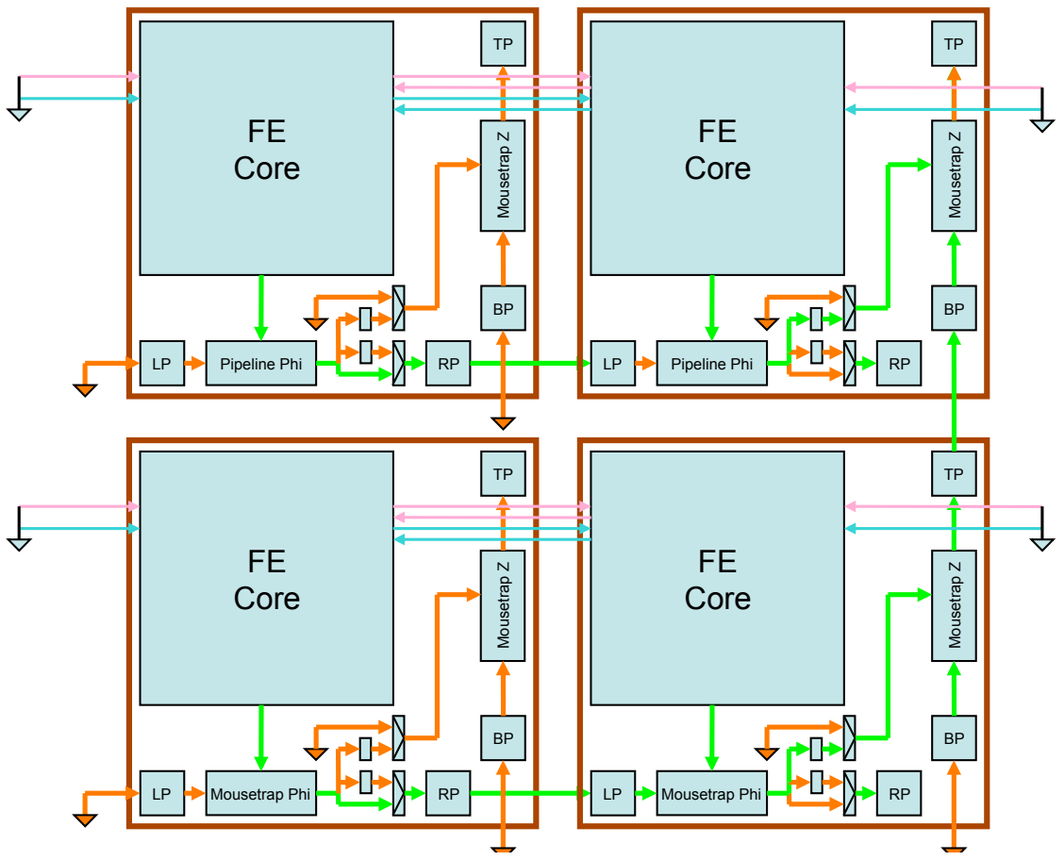
Simulations

- Verilog model for complete test chip
  - Has all required functions but not as many channels as final design
- Assemble 4 Verilog models into a sensor array to test micropipeline



4 Chip Array

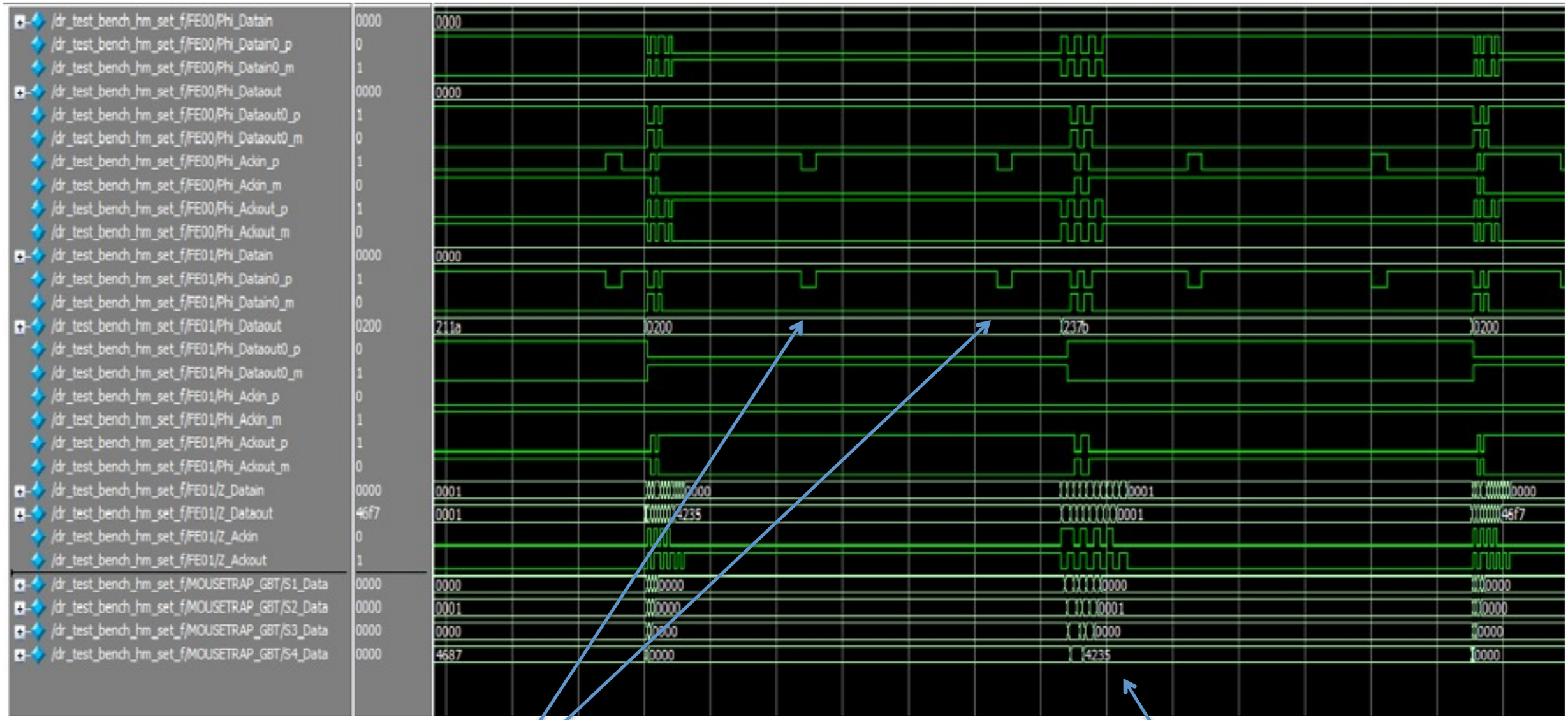
- Mousetrap micropipeline for hits between right and left chips
- Mousetrap for data from left to right and then vertical
  - Data ends in upper right chip



Functional Simulations 1

- Test Bench #1 => No Hit Mousetrap pipelines for hit data transfers between adjacent ASICs => some hits are lost
- Test Bench #2 => Hit Mousetrap pipelines included => missing hits are recovered
- Test Bench #3 => Glitch generators on “Acknowledge” and “Request” lines between adjacent ASICs (left – right Pipelines only) => some hits are lost or arrive in the wrong clock cycle
- Test Bench #4 => Dual Rail signals are used for the Acknowledge and Request lines and Dual Rail logic cells are used in the Control Modules of the Phi Pipeline => no hits are lost or arrive in the wrong cycle

Functional Simulations 2



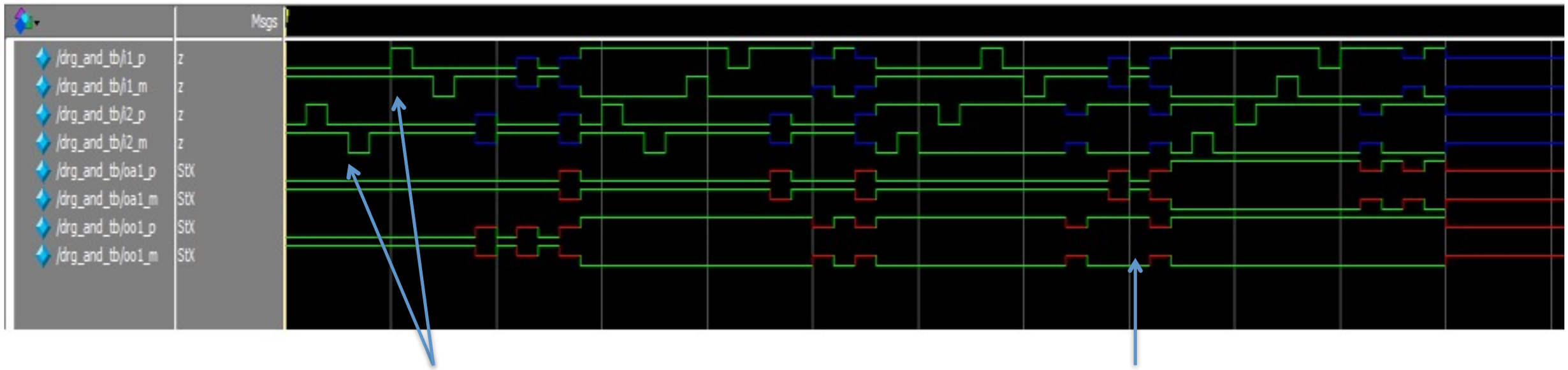
Glitch generated on “Acknowledge” and “Request” lines (Gaussian distribution of both period and pulse width)

Stub data at the GBT-TX input

Dual Rail Library

-) The protection of the critical control signals in the Mousetrap Pipelines ("Acknowledge" and "Request") against Single Event Transients (SETs) due to radiation or noise and glitches in the ASIC to ASIC links will be guaranteed by the use of Dual Rail signals and Dual Rail combinatorial logic => glitches in the input signals are not propagated to the outputs

-) A library of Dual Rail cells (based on the AND2 and the OR2 cells) has been developed and validated through functional simulations



Glitch on Dual Rail input signals => not transmitted to Dual Rail outputs of the AND2 and OR2 gates

Undefined outputs only when both inputs are undefined or when one of the inputs is undefined and the other is 1 (AND2) or 0 (OR2)

Summary

- It looks quite feasible to design a front end chip using only the LHC crossing clock
- We are planning on a MOSIS submission of an all digital test chip this summer
- We are planning on beam tests to look for radiation induced errors in the micropipelines