# MUV Trigger Firmware Status

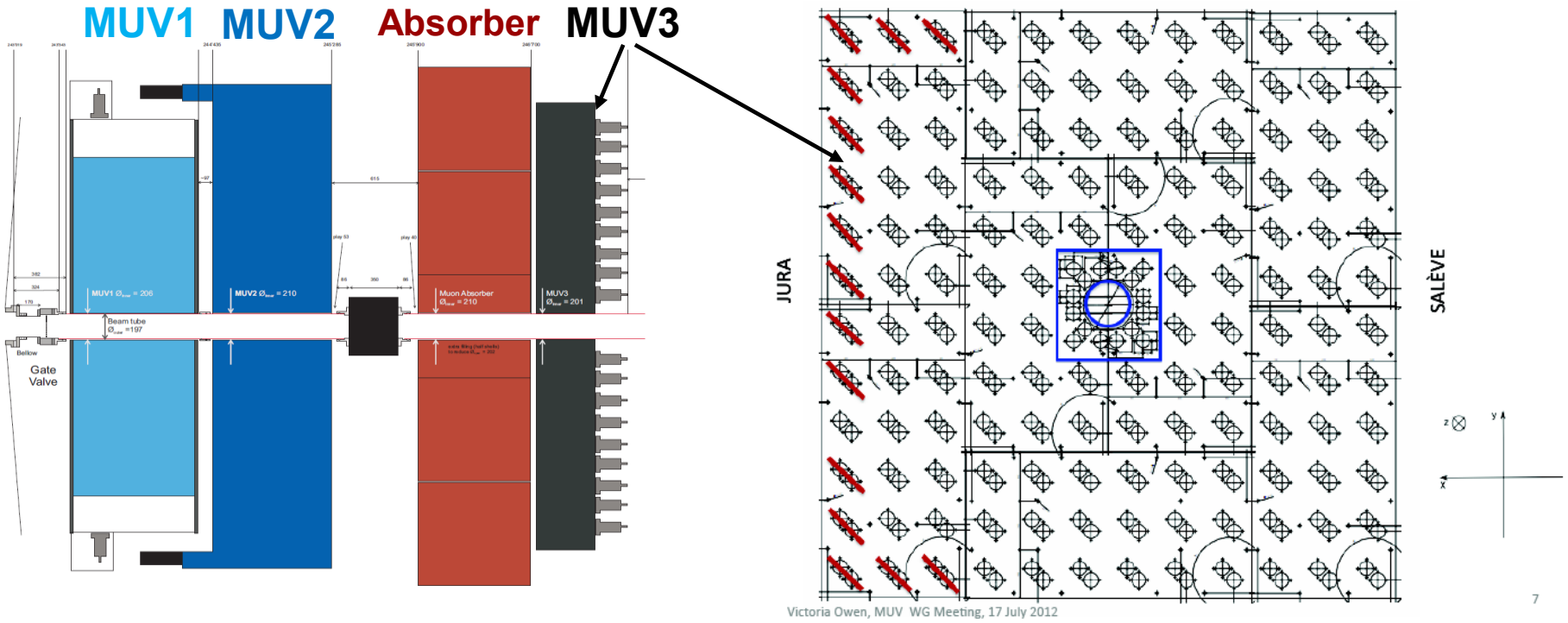
Plamen Petrov
Centre for Cosmology, Particle Physics and Phenomenology
Université catholique de Louvain, Belgium

Chris Parkinson
School of Physics and Astronomy
University of Birmingham

# Outline

- MUV3 trigger firmware
  - PP-FPGA
  - SL-FPGA
- Simulation with Modelsim
- Integration in the TEL62 framework
- Tests with the TEL62
- Status and plans until the start of run

# MUV3

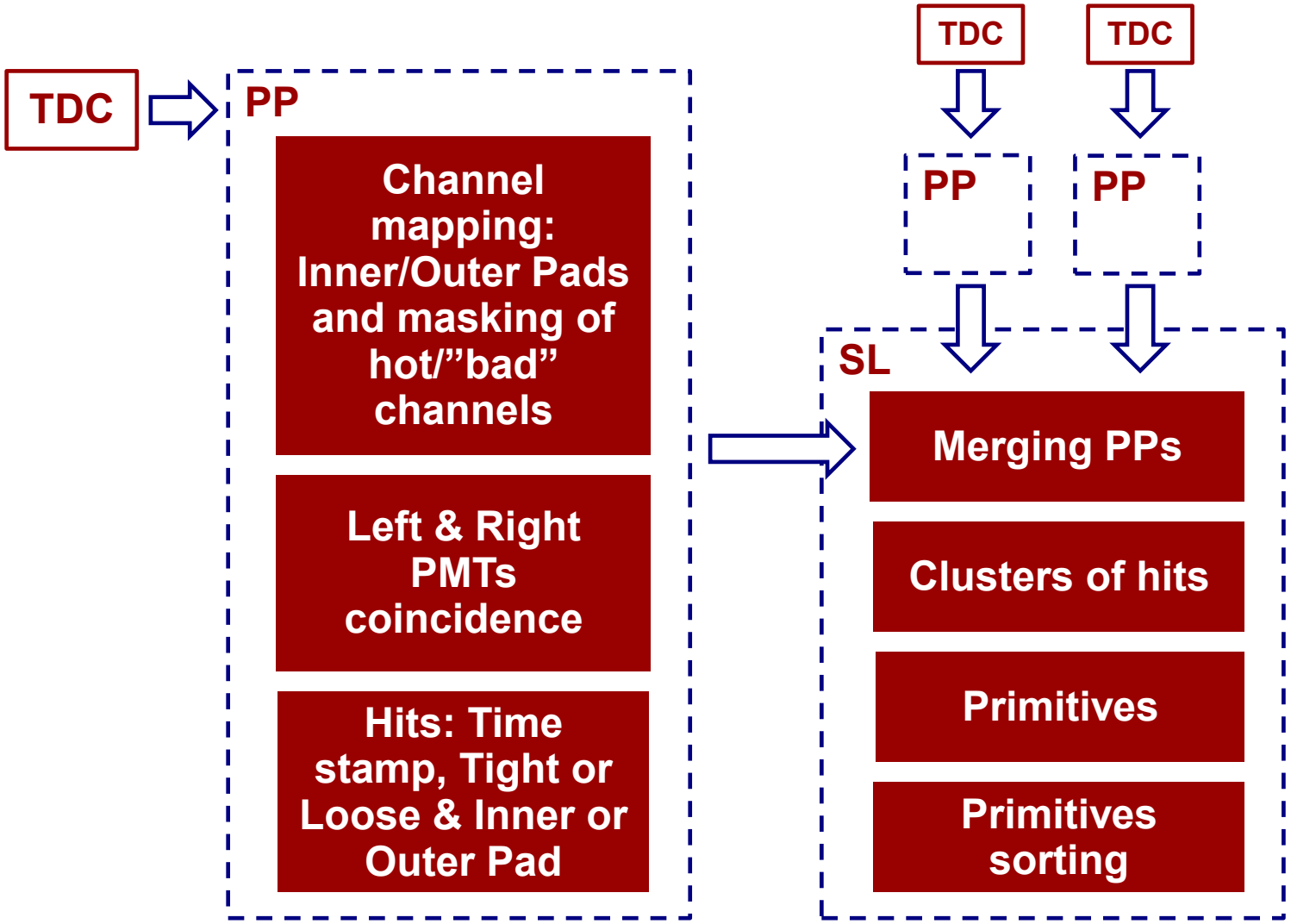



- 140 scintillator tiles, 22 x 22 cm, 5 cm thick, each seen by two PMTs (left and right);
- 8 additional smaller tiles around the beam vacuum pipe (inner tiles)

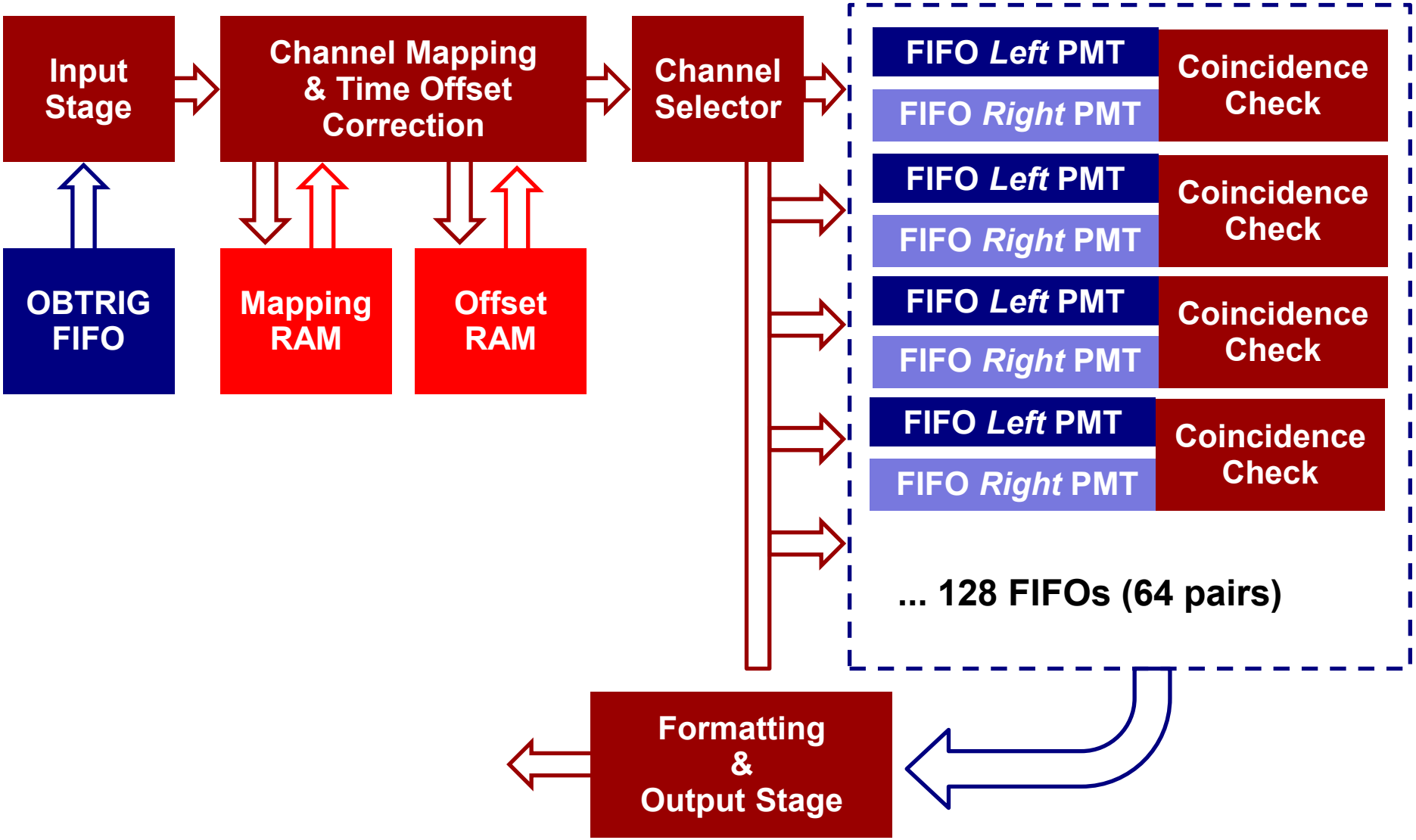
*In total, 296 PMTs.*

- The PMT signals go to 20 x CFD boards
- Two CFD boards are back-to-back with 2x16 inputs and 1x32 LVDS outputs to the TDC
- The 296 PMT signals are distributed among 3 x TDC boards (320 free channels)

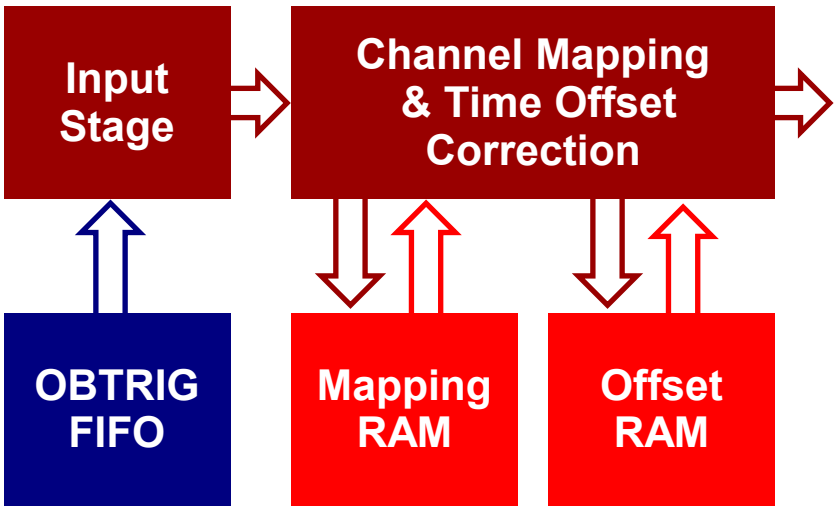
# MUV3 Firmware Layout



# PP Firmware


Input Stage
Channel Mapping & Time Offset Correction
Channel Selector
OBTRIG FIFO
Mapping RAM
Offset RAM
FIFO Left PMT
FIFO Right PMT
Coincidence Check
FIFO Left PMT
FIFO Right PMT
Coincidence Check
FIFO Left PMT
FIFO Right PMT
Coincidence Check
FIFO Left PMT
FIFO Right PMT
Coincidence Check
... 128 FIFOs (64 pairs)
Formatting & Output Stage

# PP Firmware: Input Stage and Mapping



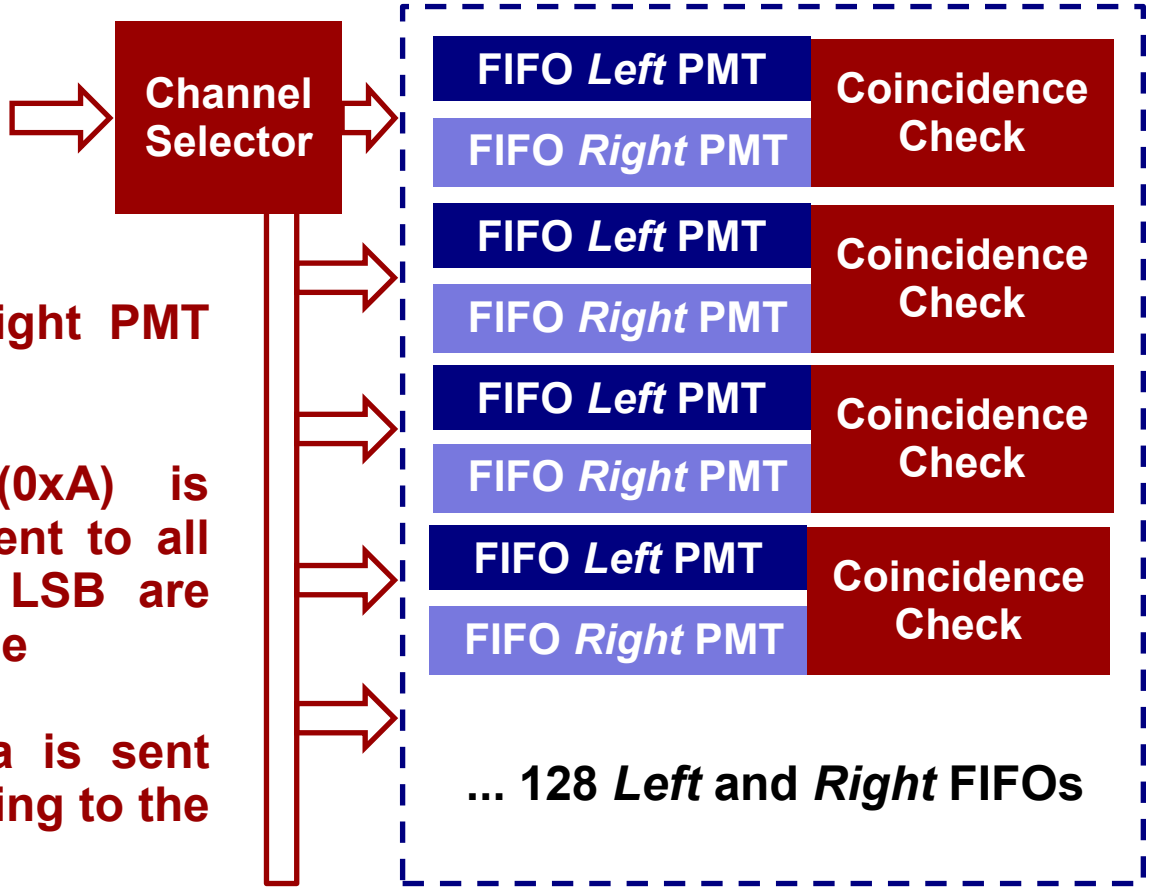

- If OBTRIG FIFO is *not empty* 32 bit data is taken and passed to next module

| | | | |
|---|---|---|---|
| 31 "**0xA**" 28 | 27 **Frame Time Stamp (400 ns LSB)** 0 | | |
| 31 "**0xB**" 28 | 27 **End of Frame** 0 | | |
| 31 "**0x4**" 28 | 27 **#TDC** 26 | 25 **Channel** 19 | 18 **Fine Time** |
| 31 "**0x5**" 28 | 27 **#TDC** 26 | 25 **Channel** 19 | 18 **Fine Time** |

- ***0x4 – leading time***: channel mapping and time offset corrections are made. Request with TDC number and channel ID is sent to RAM blocks.
- Remap channel number so that 2 PMTs from the same pad have adjacent numbers in the firmware
- Fine Time adjustment (100 ps precision)
- A signal is driven high if the channel is from an inner pad
- If a masking bit is low the data is sent to the next stage

# PP Firmware: Channel FIFOs and Coincidence

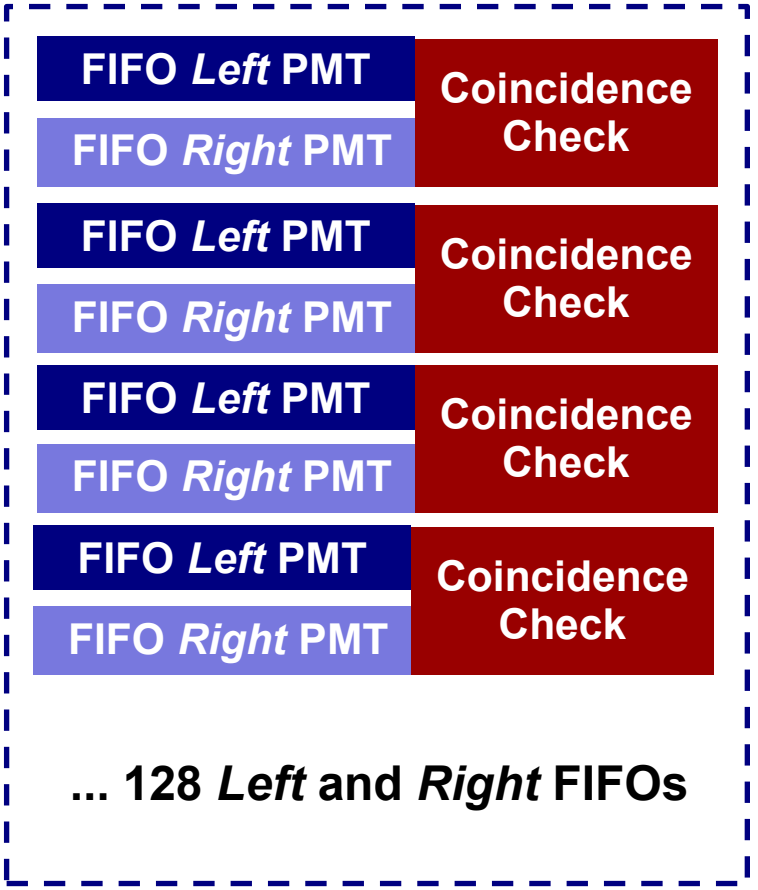
- Channel Selector (FSM) receives the 32 bit 0xA and 0x4 data and distributes it to the proper channel FIFOs
- 128 x FIFOs for left and right PMT channels (16 word x 22 bit)
- If Frame Time Stamp (0xA) is received the 20 MSB are sent to all channel FIFOs and the 7 LSB are cached internally for later use
- If Fine Time (0x4) the data is sent only to the FIFO corresponding to the remapped channel number




- 7 overlapping bits between FT and TS used to check for possible overflow of the Fine Time after the offset correction in the previous stage

# PP Firmware: Channel FIFOs and Coincidence

- Every FIFO is monitored by a FSM. When both TS and FT are available the absolute time is calculated (40 bits) and a channel ready flag is set high
- When a channel FIFO is full the hit is lost. Error log is sent
- When both channels are ready the time difference is computed
- If less than a 'low' threshold - indicative of a muon hit in the MUV3 pad. Average time of the two channels is outputted.
- If greater than 'low' but less than 'high' threshold – muon hitting pad and passing through one of the PMTs producing Cherenkov radiation in the tube. The later time is outputted.
- If time diff is greater than 'high' – no coincidence. The earlier time is send as a single PMT event



# PP Firmware: Output Stage

- **A flag ‘01’ identifies the first of two words to be send to the output buffer (towards SL)**

| 31 “100” 29 | 28 Hit type | 27 isDouble | 23 Error | 22 Block Number 16 | 7 Time Stamp MSB 0 |
|---|---|---|---|---|---|
| 31 | Time Stamp (LSB = 100 ps) | | | | 0 |

- **One bit for inner (high) or outer (low) MUV3 pad**
- **One bit for single (high) or double-PMT (low)**
- **In the case of a double-PMT datum, one bit is used to identify if the low (bit high) or high (bit low) coincidence threshold was satisfied. In the case of a single-PMT datum, the same bit is used to identify which PMT the datum corresponds to (left or right)**
- **One bit to identify an error: channel FIFO full.**
- **Six bits for the pad number (the internal to the PP going from 0 to 63)**
- **Eight MS bits of the 40 bit time go in the first word and the remaining 32 in the second**
- **A flag ‘11’ identifies an EoF. The remaining 30 bits are used to pass to the SL the number of hits in the frame that has been send (for consistency check)**

| 31 “11” 30 | Frame number |
|---|---|

# SL Firmware: Primitives

- In the SL firmware the hits from the PPs are combined into clusters with the coincidence time interval set via a 8 bit register (up to ±12.5 ns)

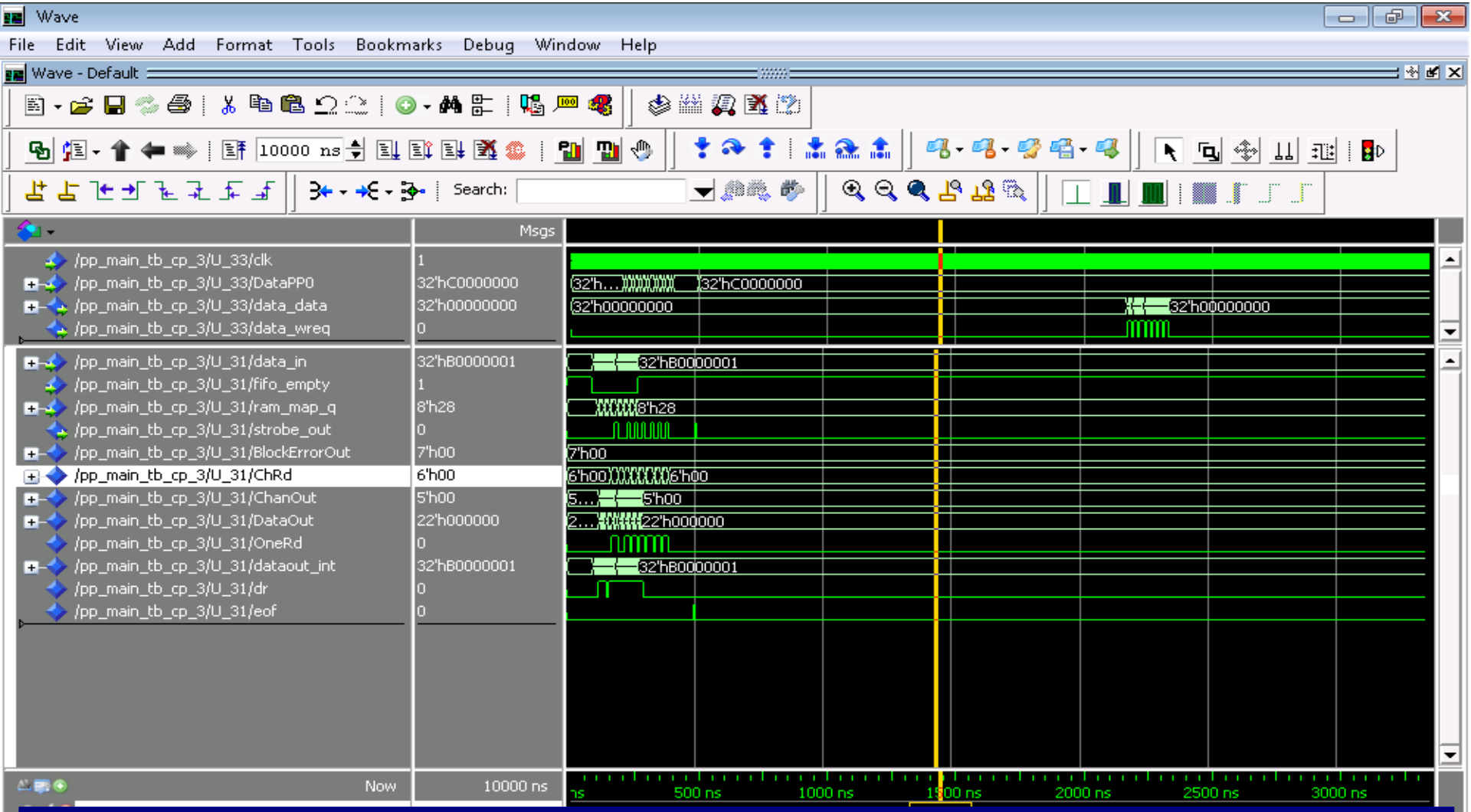
- Each cluster keeps four registers which are used to make the L0 primitives:

  - SI – number of single-PMT hits in an inner pad
  - DI – number of double-PMT hits in an inner pad
  - SO– number of single-PMT hits in an outer pad
  - DO – number of double-PMT hits in an outer pad

| Primitive | Primitive ID |
|---|---|
| SI + SO + DI + DO > 0 | 0x0001 |
| DI + DO > 0 | 0x0002 |
| SO + DO > 0 | 0x0004 |
| DO > 0 | 0x0008 |
| SO + DO > 1 | 0x0010 |
| DO > 1 | 0x0020 |

# Simulation with Modelsim

- Tried different sequences of input to the PPs
- Output from SL as expected



# Integration in TEL62 Framework

- **The MUV3 firmware was integrated in the TEL62 framework (*mid July*)**

**Top-level Entity Name : pp_fpga**
Device : EP3SL200F1152C4
Logic utilization : 52 %
Combinational ALUTs : 50,515 / 159,120 ( 32 % )
Memory ALUTs : 4,738 / 79,560 ( 6 % )
Dedicated logic registers : 46,923 / 159,120 ( 29 % )
Total registers : 47580
Total pins : 581 / 744 ( 78 % )
Total block memory bits : 3,274,322 / 9,621,504 ( 34 % )
DSP block 18-bit elements : 0 / 576 ( 0 % )
Total PLLs : 3 / 8 ( 38 % )
Total DLLs : 1 / 4 ( 25 % )

**Top-level Entity Name : sl_fpga**
Logic utilization : 53 %
Combinational ALUTs : 42,605 / 159,120 ( 27 % )
Memory ALUTs : 40 / 79,560 ( < 1 % )
Dedicated logic registers : 47,638 / 159,120 ( 30 % )
Total registers : 47777
Total pins : 728 / 744 ( 98 % )
Total block memory bits : 1,668,954 / 9,621,504 ( 17 % )
DSP block 18-bit elements : 0 / 576 ( 0 % )
Total PLLs : 2 / 8 ( 25 % )
Total DLLs : 0 / 4 ( 0 % )

# Lab Test

- First test with the TEL62 board at the end of July
- Used the bench setup at Birmingham
- Problem with DDR Error

```
[IDLE]TDSpy>ppstatus
---------- PP-FPGA    PP-FPGA #0    PP-FPGA #1    PP-FPGA #2    PP-FPGA #3
Mode:                    IDLE          IDLE          IDLE          IDLE
Status:                0x00000001    0x00000001    0x00000007    0x00000007
Error:                 0x02000400    0x00000400    0x00000000    0x00000000
                      DDRINIT       DDRINIT
```

- Problem is still not understood but afterwards seen by others (RICH/CHOD firmware)
- Thanks to Cristiano for helping with this issue

# Status and Plans

- Firmware is written and simulated successfully
- MUV3 firmware documentation is written and will be made available shortly
- Tests with the board will continue in Birmingham until 15th September
- After that - move to CERN
  - MUV3 dedicated PC will be installed in the control room
  - New TEL62 board in the MUV3 rack

# BACK UP

# MUV3 layout

- LEMO cables run between the MUV3 patch panel and the Constant Fraction Discriminators (CFD)
- There are 20 x CFD boards:
  - Two boards are back-to-back with 2 x 16 inputs and 1 x 32 LVDS outputs to the TDC
- The TDC board has 4 x CERN HPTDC with 32 channels each
- Eight consecutive HPTDC channels go to a 256-word buffer which is read out every 6.4 us (1 frame)
- If this buffer is filled in less than 6.4 us any additional signal is lost with no warning
- One PMT signal consists of two TDC words (times of leading and trailing edge)
  - max. 128 PMT signals in 6.4 us (20 MHz)
- If both PMTs of the same tile share the same 256-word buffer the maximum number of tile hits is 64 / 6.4 ms (considered in the cable mapping). See talk by Chris: MUV and New CHOD WG Meeting, 02.04.2014)
- The 296 PMT signals are distributed among 3 x TDC boards (320 free channels)

# Rates and trigger primitives

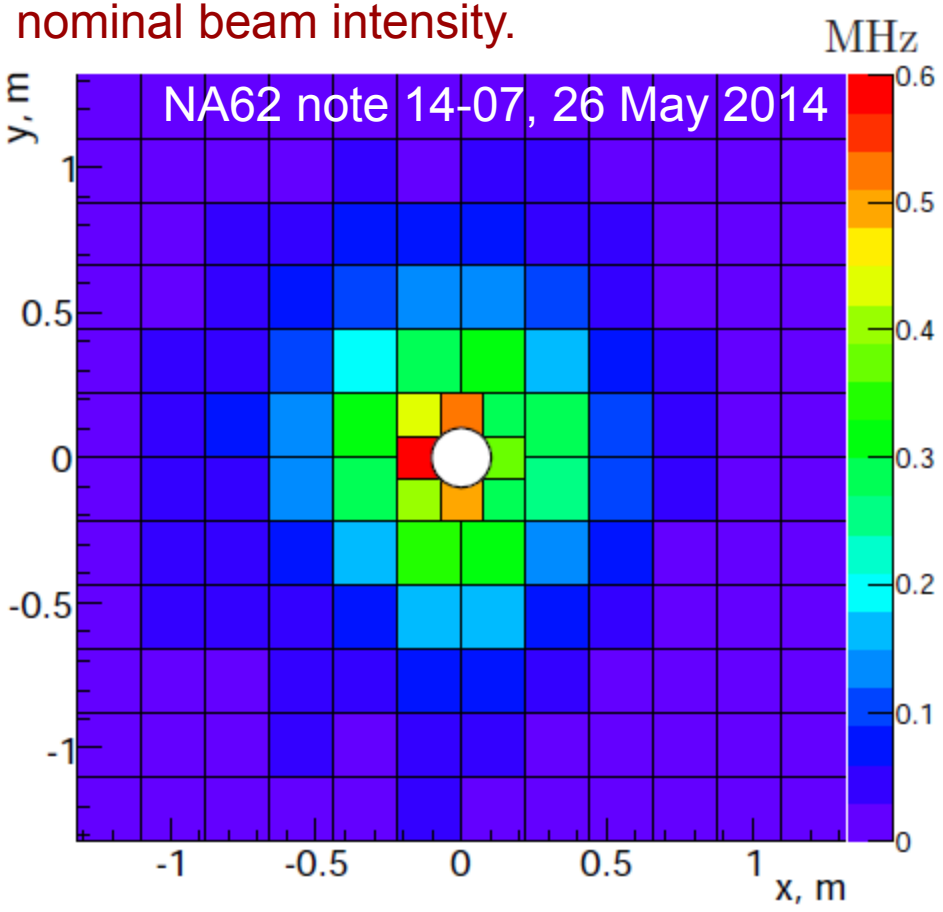
**Signal from a tile is:**

- ***Loose:*** **signal from only 1 PMT**
- ***Tight:*** **coincidence of the 2 PMTs watching the tile**

**The currently proposed trigger primitives are based on *loose* and *tight* signal multiplicities:**

- **Signal in at least one pad (*MUV3-L1, MUV3-T1*)**
  - **Expected rate: ~13 MHz**
- **Signal in at least one outer pad (*MUV3-LO1, MUV3-TO1*)**
  - **~ 5 MHz**
- **Coincidence of signals in at least two outer pads (*MUV3-LO2, MUV3-TO2*)**
  - **~ 0.1 MHz (for 5 ns coincidence window)**

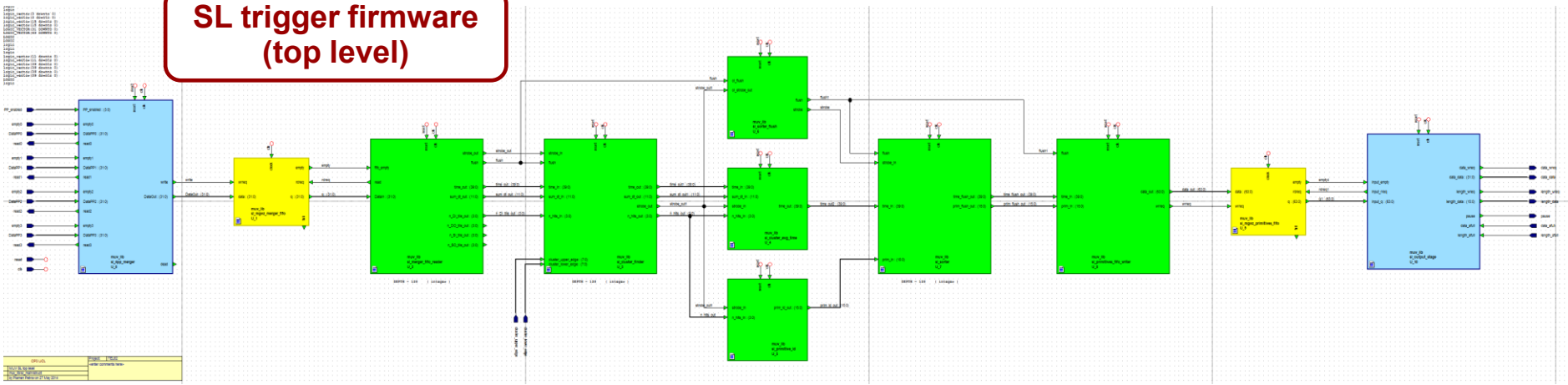
MUV3 channel profile and rate visualization, at nominal beam intensity.



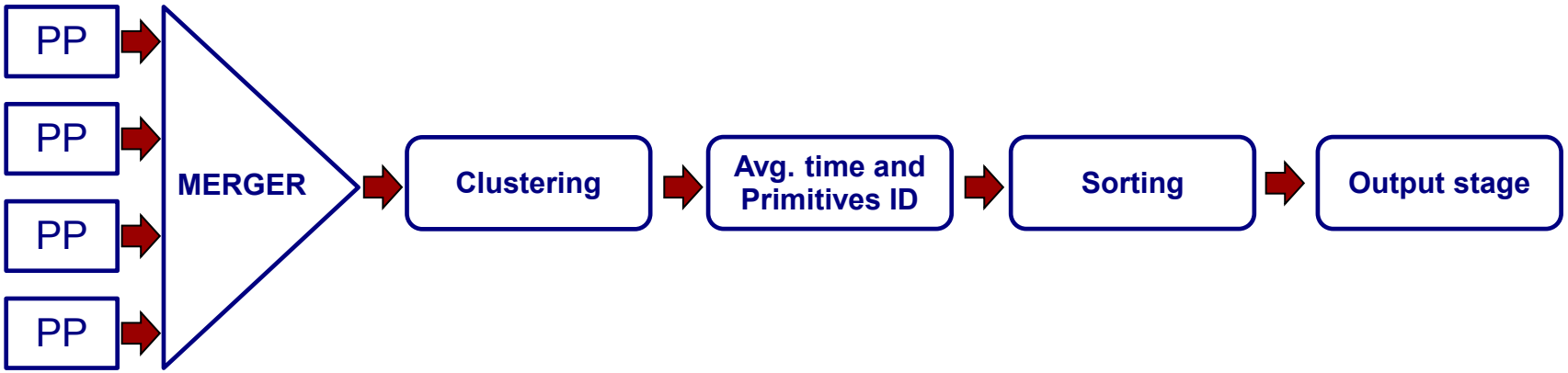

***Hot spot*** **rate of ~ 3.2 MHz**
**Max. total rate of ~ 13.75 MHz**
- **~90 hits in 6.4 us time frame**

# SL Firmware

**SL trigger firmware (top level)**



- **Data merging:** receive hits from 4 x PP-FPGA and merge them into a single data stream
- **Clustering:** form clusters of hits within a given time interval
- **Primitives:** tag clusters with a primitive ID
- **Sorting:** order the primitives in time
- **Output stage:** Multiple Trigger Packet (MTP) generation



# SL Firmware: data merging

- **PP to SL output format**
  - **1 Hit = 2 x 32 bit words**

| *31* “10” *30* | *29* Hit type *28* | **Not specified yet!** | *7* Fine Time (100 ps) *0* |
|---|---|---|---|
| *31* | Time Stamp | | *0* |

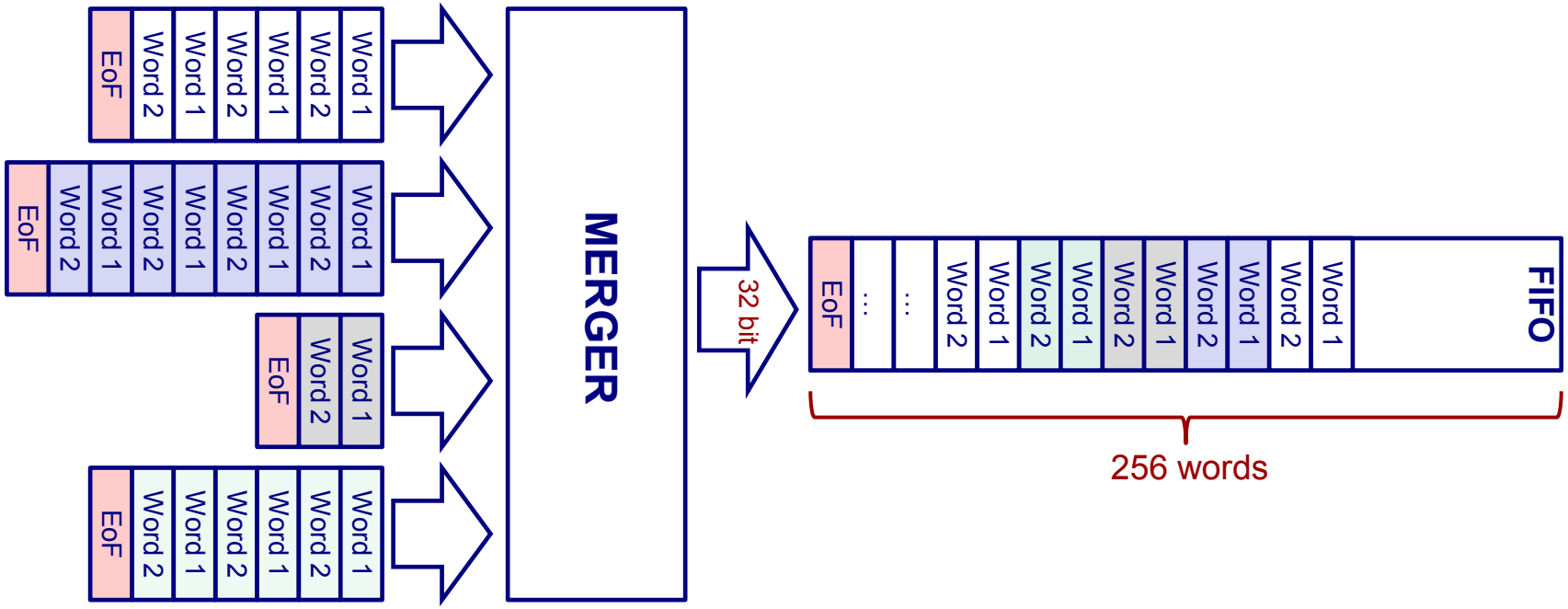
| *31* “11” *30* | **End of Frame** |
|---|---|

  - **If the 1st word identifier is “10”**
  - **Take the 2nd word to be the Time Stamp**
  - **When EoF word is received from all enabled PPs the MERGER produces a global EoF word (x”FFFFFFFF”)**

- ***Hit type* as evaluated in the PP**
  - **Inner or Outer pad: set bit 29 to 0 or 1**
  - **Tight or Loose hit: set bit 28 to 0 or 1**

# SL Firmware: data merging

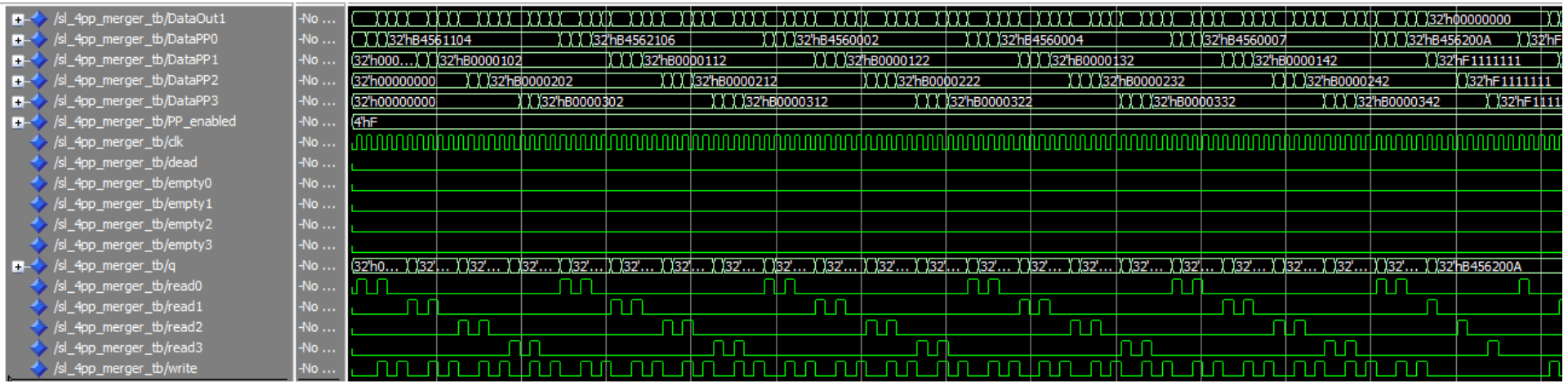
- *MERGER module* (4 x PP FIFOs to 1 x SL)
  - FSM monitoring the input FIFOs of the enabled PPs (3 x PPs for MUV3)
  - Identifies Data and EoF by the bits 32 and 31
  - If more than 1 FIFO 'not empty' *MERGER* reads them out cyclically
  - Stores the single data stream into a FIFO
  - Issues a Global EoF word when all PPs have sent an EoF




- FIFO:
  - 32 bit wide, 256 words deep
  - 128 hits per 6.4 us (20 MHz hit rate)

# SL Firmware: data merging

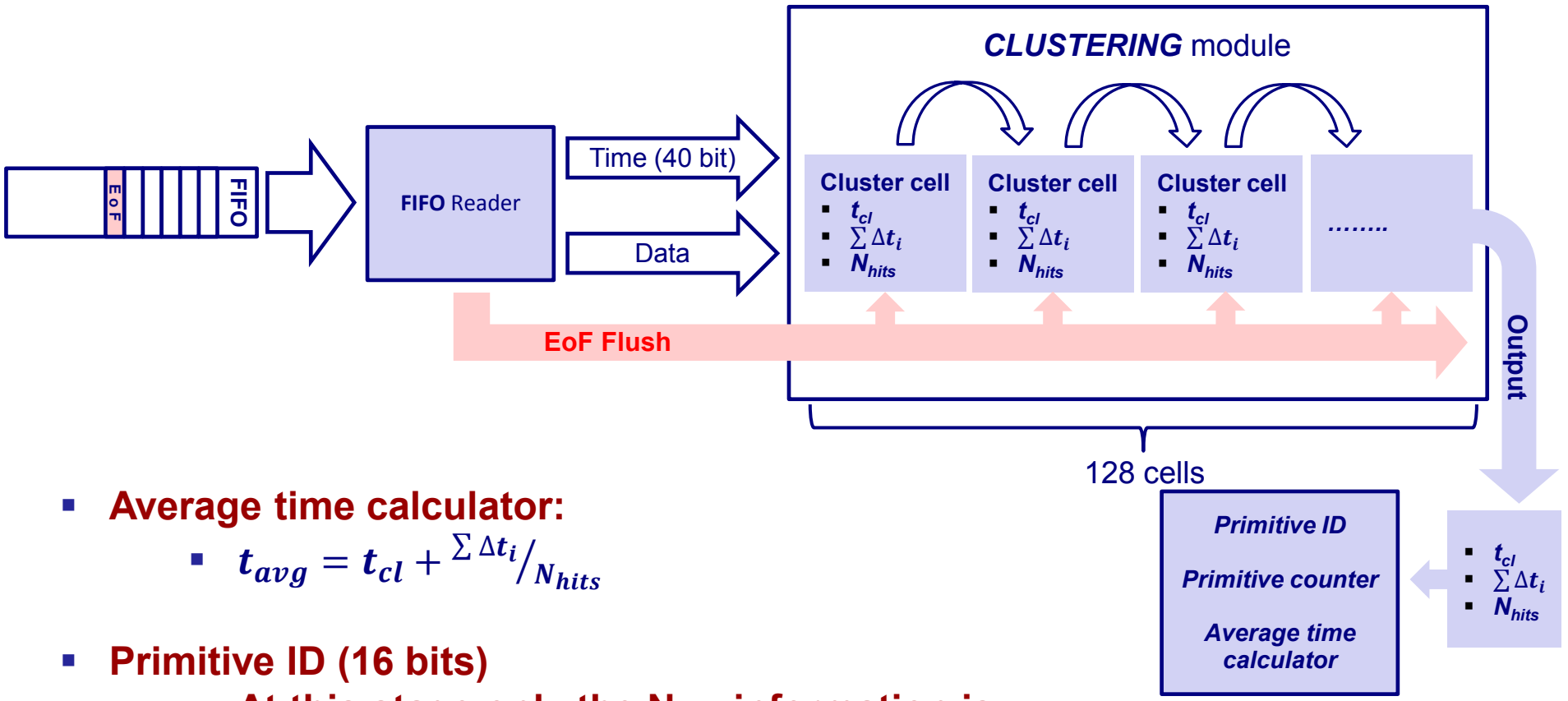
- *MERGER* module in ModelSim



# SL Firmware: making hit clusters

- A dedicated module (FSM) reads out the FIFO and passes the data to the *CLUSTERING* module
  - Reads out the hits from the FIFO in blocks of two consecutive words
  - Outputs the hit time as a 40 bit number
  - Outputs all information needed to make the primitives (tight, loose, inner, outer)
  - All other (unused) information is discarded at this point
  - Signals the *CLUSTERING* module when global EoF is received
- *CLUSTERING* module: Composed of 128 *cluster* cells
- First received hit goes in the first cell and sets the cluster cell time ($t_{cl}$)
- If successive hit times ($t$) match the cell time they are included in the cluster
  - the matching interval is programmable through a 8 bit register (LSB = 100 ps ➔ max ±12.5 ns)
  - $N_{hits}$ is incremented
  - The difference with the cell time is added up: $\sum \Delta t_i = \sum (t_i - t_{cl})$
- If successive hit time doesn't match the cell time it goes to the next cell
- At EoF the clustering module waits for 128 cycles and flushes out the cluster cells one-by-one (as a shift register)

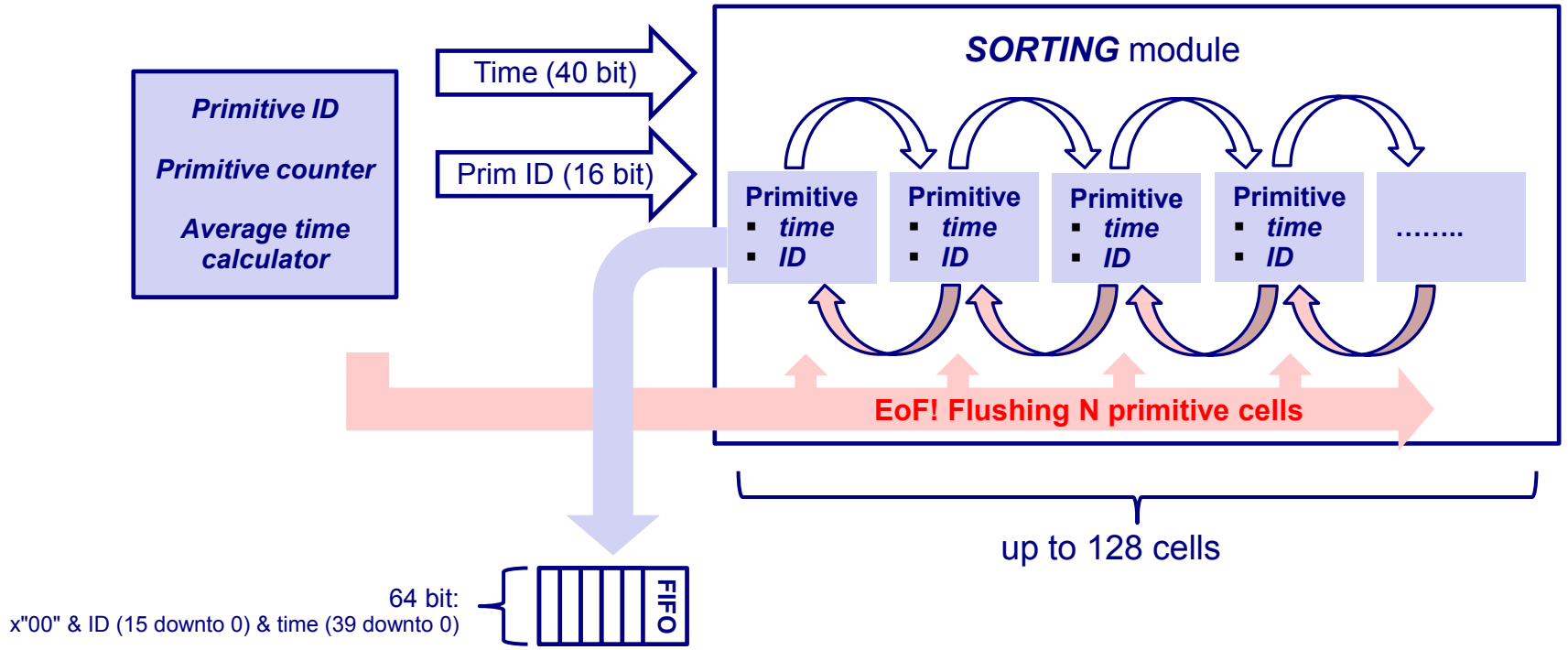
# SL Firmware: making hit clusters




- **Average time calculator:**
  - $t_{avg} = t_{cl} + {}^{\sum \Delta t_i}/_{N_{hits}}$
- **Primitive ID (16 bits)**
  - **At this stage only the $N_{hits}$ information is used**

# SL Firmware: time ordering the primitives

- *SORTING* module similar to the *CLUSTERING*
  - Array of 128 sorting cells
- Receives time (40 bit) and primitive ID (16 bit) and stores them in a sorting cell
  - Successive times are compared to the time stored in the cell. The smaller time stays in the cell and the greater is passed to the next (together with the corresponding primitive ID)
- When the last (Nth) primitive is send to the sorting module the module waits for N cycles (sorting latency) and starts flushing out the sorted primitives one-by-one as a shift register starting from the smaller times



# SL Firmware: Output stage

- **Output module (FSM)**
  - **Reads the primitives FIFO when *not empty***
  - **Counts the received primitives and tags each one with a consecutive number**
  - **Outputs the primitives in the specified format (3 x 32 bit words) to form the MTPs**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| *31* | Primitive Number | | | | | | *0* |
| *31* | Primitive ID | *16* | *15* | Reserved | *8* | *7* Fine Time | *0* |
| *31* | Time Stamp | | | | | | *0* |

# Tests with the

- Test performed in July in Birmingham
  - TEL62 – v2 (fpga 200sll,smkd)
- D
- F
- F
- F
- f

# What next…

- Integration of PP and SL trigger firmware in the TEL62 Firmware
- Error logging
- Testing the full PP and SL firmware with Modelsim
- Hardware tests