Gains from timing

A 4D stub based tracking approach

Marco Petruzzo, Paolo Gandini, Nicola Neri

INFN Milano

3rd Workshop on LHCb Upgrade II

Annecy, France

March, 21-23 2018

TIMESPOT R&D INFN project

- **TIME & SPace real-time Operating Tracker:**
  - 1 M€ **three years** project (from 2018), funded by **INFN**
  - Development of a silicon and diamond **3D tracker with timing facilities**
  - **Construction of a demonstrator** integrating sensors, electronics, real-time processors

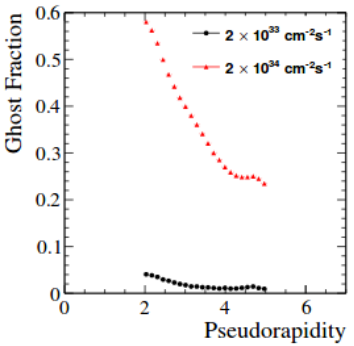
  - **Work Packages** and coordinators:
    - **3D silicon sensors: development and characterization** – G.F. Dalla Betta
    - **3D diamond sensors: development and characterization** – S. Sciortino
    - **Design and test of pixel front-end** – V. Liberali
    - **Design and implementation of fast tracking devices** – N. Neri
    - **Design and implementation of high speed readout boards** – A. Gabrielli
    - **System integrations and tests** – A. Cardini

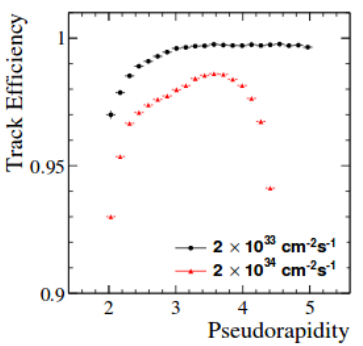
Timing in LHCb Upgrade II

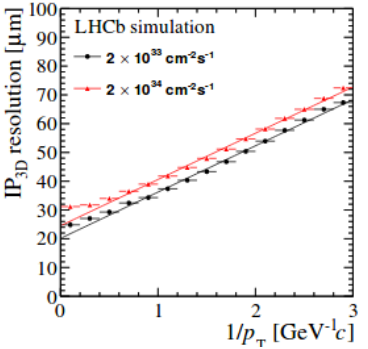
- Upgrade II luminosity:
  - **2*10^{34} cm^{-2} s^{-1}**
  - 10x w.r.t. Upgrade I

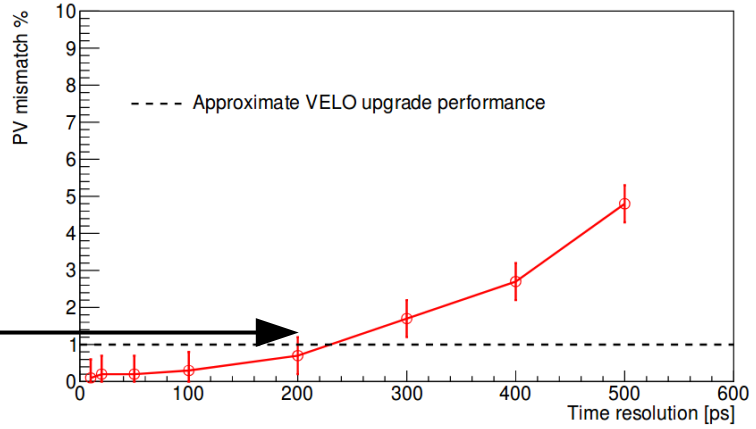
- Phase-I detector can not sustain the increased luminosity:
  - ~40% ghost rate
  - PV merging, ~13% mis-association

- Inclusion of **timing to recover performance:**
  - in PV association ~200ps resolution needed
  - **in tracking**, ~30ps needed





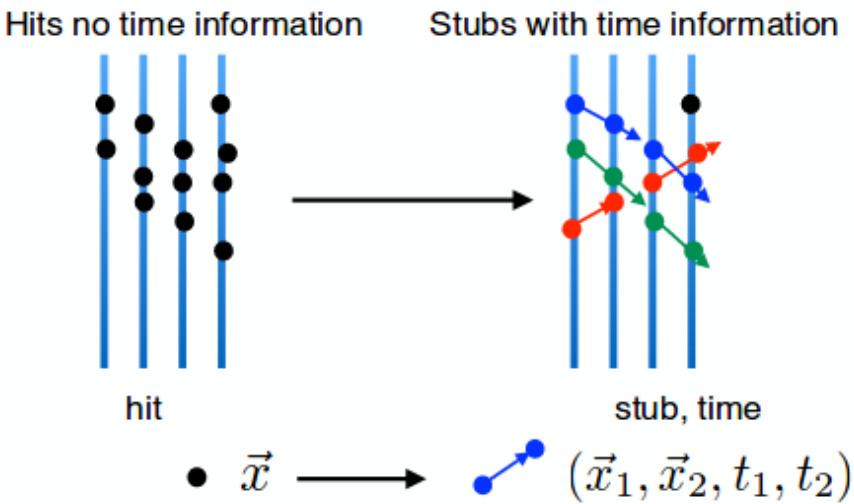




[CERN-LHCC-2017-003]

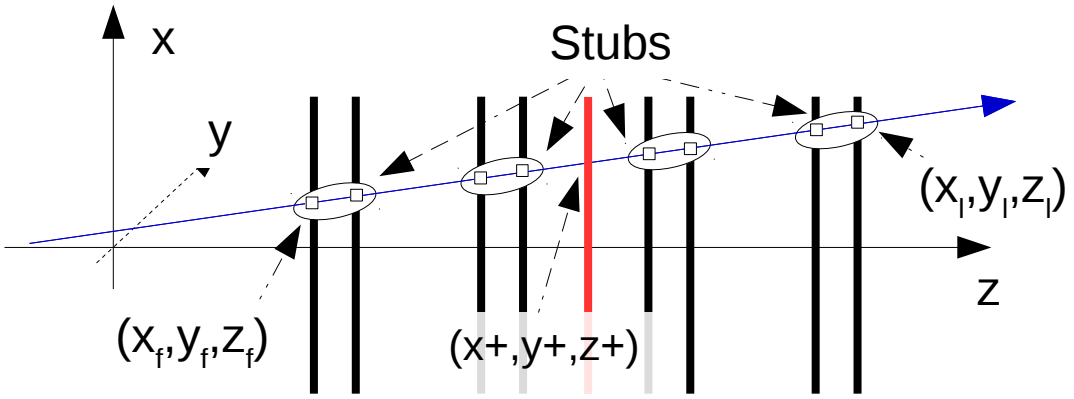
4D stub based tracking

- **“Stub” approach:**
  - A **couple of hits** in adjacent planes forms a stub **[N. Neri et al., JINST 11 (2016) no.11, C11040]**
  - Stubs provide **“track hints”**
  - **Geometrical cuts** are applied to filter stubs not compatible with tracks from the luminous region
  - Tracks are formed by multiple stubs with similar parameters



- **Stubs with time:**
  - In highly occupied detectors **fake stubs can survive the geometrical cuts**
  - Use of timing allows a **combinatoric suppression**
  - **Particle velocity** is required to be compatible with the **speed of light**

Track and stub definition



$(x_f, y_f, z_f), (x_l, y_l, z_l)$: intersections of the track with first and last tracking plane

$$x_\pm = (x_f \pm x_l)/2$$
$$y_\pm = (y_f \pm y_l)/2$$
$$z_\pm = (z_f \pm z_l)/2$$

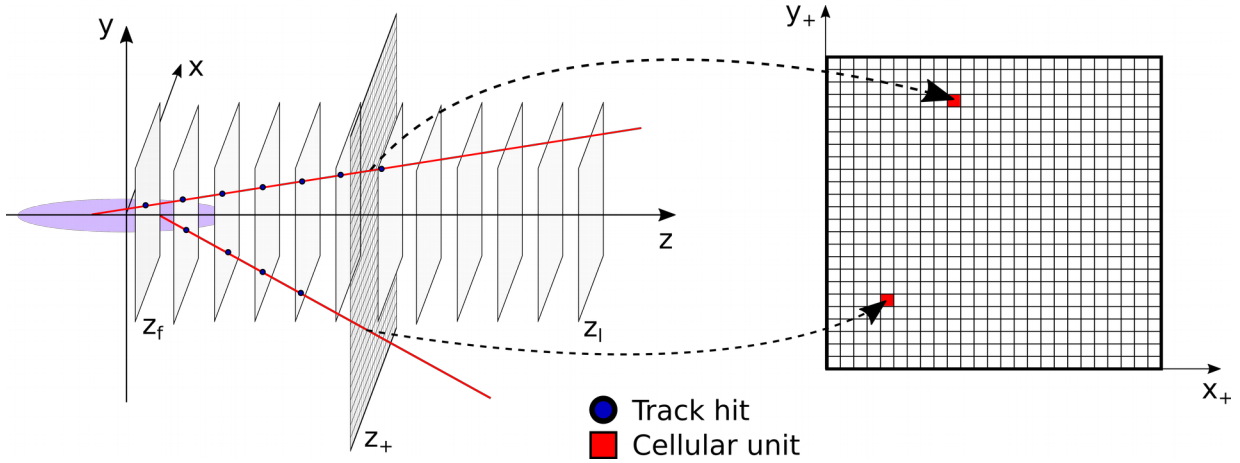
: (x+,y+) intersections with a tracking plane placed at z+; (x-,y-) define the track slopes

- Tracks and stubs are defined by **5 parameters:**
  - Four **3D track parameters**
  - **time of the track** at the origin

$$\begin{pmatrix} x_- \\ x_+ \\ y_- \\ y_+ \\ t \end{pmatrix}_{stub} = \begin{pmatrix} \frac{x_1 z_- - x_2 z_-}{z_1 - z_2} \\ \frac{x_1(z_+ - z_2) - x_2(z_+ - z_1)}{z_1 - z_2} \\ \frac{y_1 z_- - y_2 z_-}{z_1 - z_2} \\ \frac{y_1(z_+ - z_2) - y_2(z_+ - z_1)}{z_1 - z_2} \\ \frac{t_1 + t_2}{2} - \frac{z_1 + z_2}{2\,c\,\sqrt{1 + (x_-/z_-)^2 + (y_-/z_-)^2}} \end{pmatrix}$$

- The track velocity is assumed to be $v = c$

4D tracking algorithm



- A grid of ***engines*** is distributed in a reference plane (x+,y+) at z=z+.
- Each engine evaluates its **distance from the stub projection** to the reference plane
- A **Gaussian response** is evaluated based on the stub-engine distance

$$s^2_{ijk} = (x_{k+} - x_{i+})^2 + (y_{k+} - y_{j+})^2 \quad \Rightarrow \quad W_{ijk} = \begin{cases} \exp\left(-\frac{s^2_{ijk}}{2\sigma^2}\right) & \text{if } s_{ijk} < 2\sigma \\ 0 & \text{otherwise} \end{cases}$$

[N. Neri et al., JINST 11 (2016) no.11, C11040]

4D tracking algorithm (2)

- The **total response** is evaluated as: $W_{ij} = \frac{1}{N_{ij}} \sum_{k}^{N_{ij}} W_{ijk}$

- **(x-,y-,t)** information **not used in the pattern recognition**
- Each **local maximum** identifies a candidate track

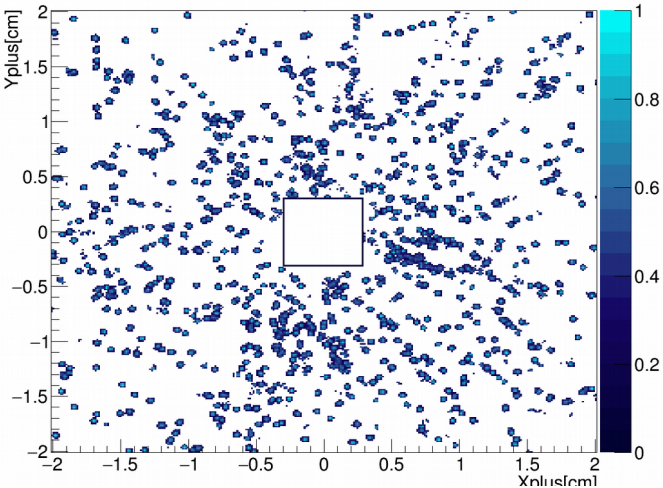
- The **(x+,y+) track parameters** are recovered by **interpolation** of the response function

- The **(x-,y-,t) track parameters** are obtained by average of the stub values forming the track

$$x_{-ij} = \frac{1}{N_{ij}} \sum_{k}^{N_{ij}} x_{-ijk}$$

$$y_{-ij} = \frac{1}{N_{ij}} \sum_{k}^{N_{ij}} y_{-ijk}$$

$$t_{ij} = \frac{1}{N_{ij}} \sum_{k}^{N_{ij}} t_{ijk}$$

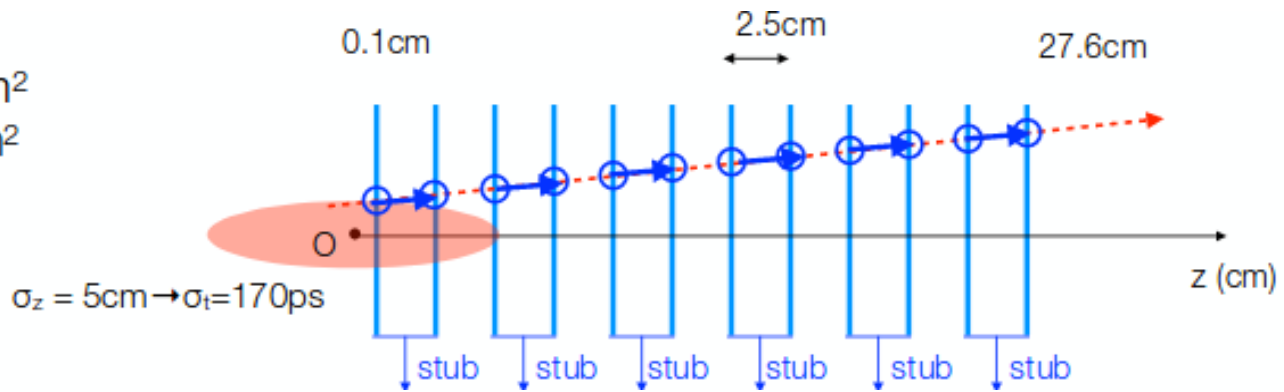


Typical response of the grid of engines to an event with multiple tracks. Each local maximum identifies a candidate track

Simulation conditions

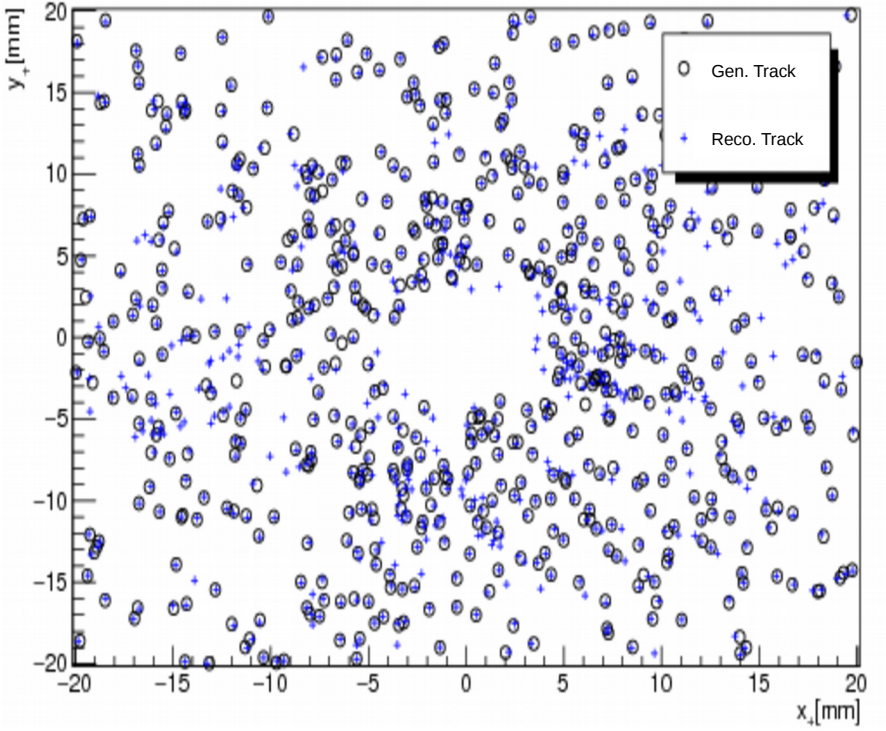
- **VELO-like tracking device:**
  - 12 planes of silicon pixel detectors in the forward region.
  - Pile-up ~**40**
  - ~**1200** tracks/event
  - Luminous region Gaussian distributed: σ_z=5cm, σ_t=167ps

Sensor area = 6x6cm²
pixel size = 55x55μm²
thickness = 200 μm
time res σ_t=30 ps



Simulated response of the algorithm

- Standalone software simulation
  - **90’000 engines** (~5000eng/FPGA)
  - Uniform distribution in the (x+,y+) space
  - Not optimized, [-2,2]x[-2,2]cm^2 square
- Response simulated **with** and **without** using the **time information** of the stubs
- Tracking **resolution, efficiency and purity** evaluated and compared

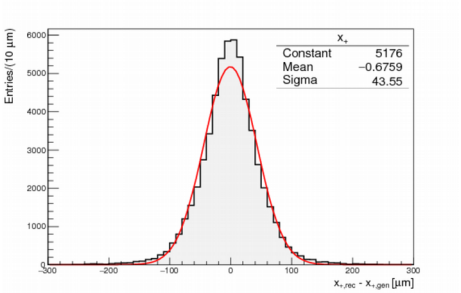


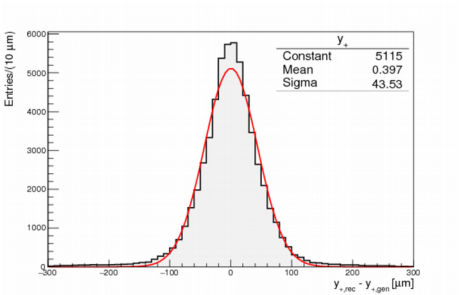
Tracking performance

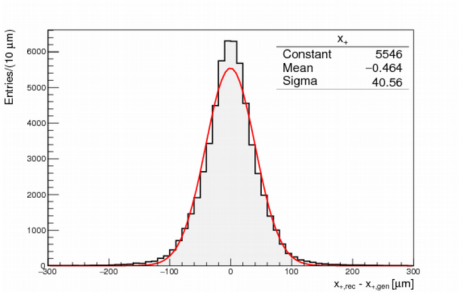
$\sigma_{x-/y-}$ = 95.0μm
$\sigma_{x+/y+}$ = 43.5μm

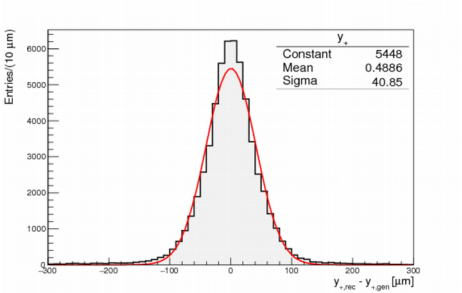
Time

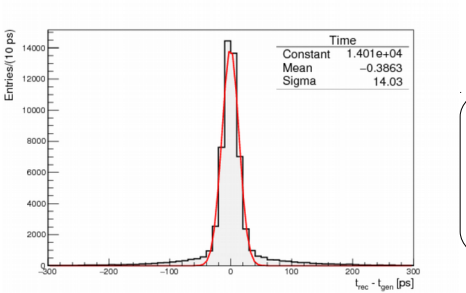
$\sigma_{x-/y-}$ = 70.1μm
$\sigma_{x+/y+}$ = 40.6μm
σ_{t} = 14.3ps











First row: resolution on track parameters (x+,y+) **without** using the time information of the stubs

Second row: resolution on track parameters (x+,y+, t) **using the time information** of the stubs

Efficiency = 99%

Purity = 64%

Time

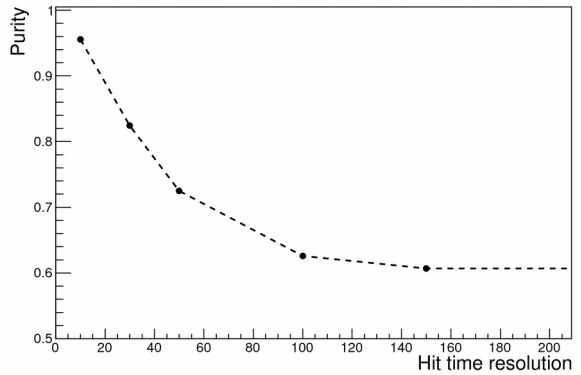
Efficiency= 99%

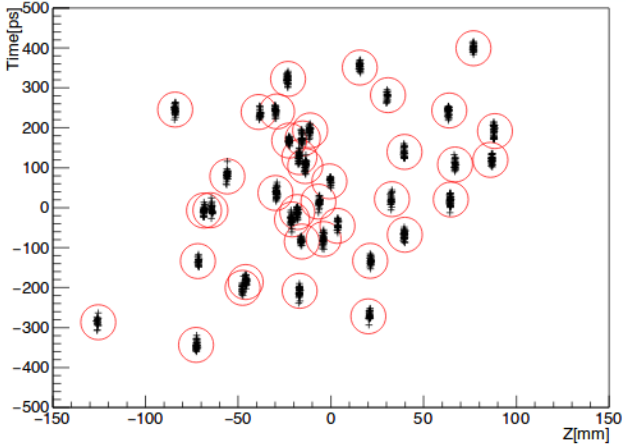
Purity = 85%

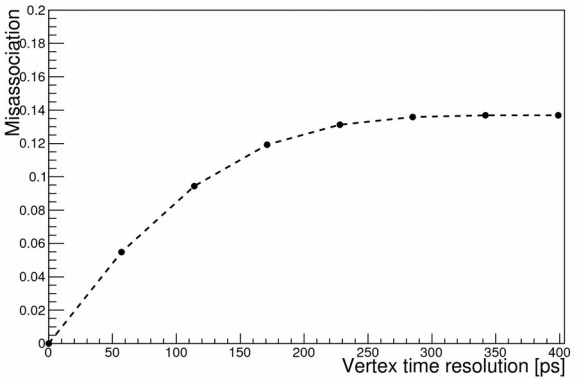
[N.Neri, M.Petruzzo, paper in preparation]

- The **track parameters resolution improves** when including the time information
- The reconstruction **efficiency is stable**
- The tracks **purity improves**

Benefits of timing on track purity and PV association





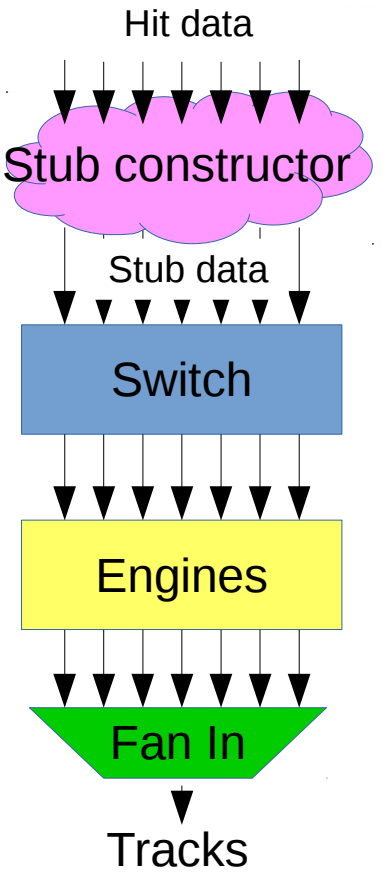


- The track **purity improves with improved time** resolution, as expected
- The track timing allows to **"clusterize" the tracks** in a 2D (z,t) space
- The **PV mis-association reduces**, with improved time resolution

Hardware architecture

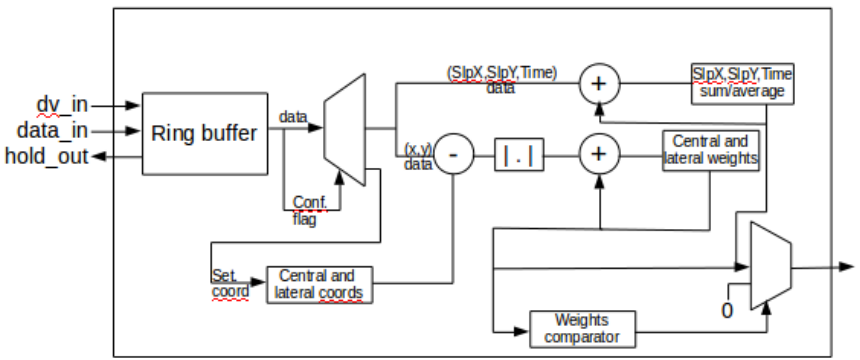
- **Main blocks**:
  - **Stub constructor**, to be designed in FPGA, with Associative Memories, or integrated in front-end.
  - **Switch**, implemented as a full-mesh network, delivers the stubs to the engines
  - **Engines**, organized in “regions”, receive the stubs and calculate the track parameters.
  - **Fan In**, collects and outputs the reconstructed tracks

- The system provides already **reconstructed tracks as output,** to be used for further processing, i.e. in HLT

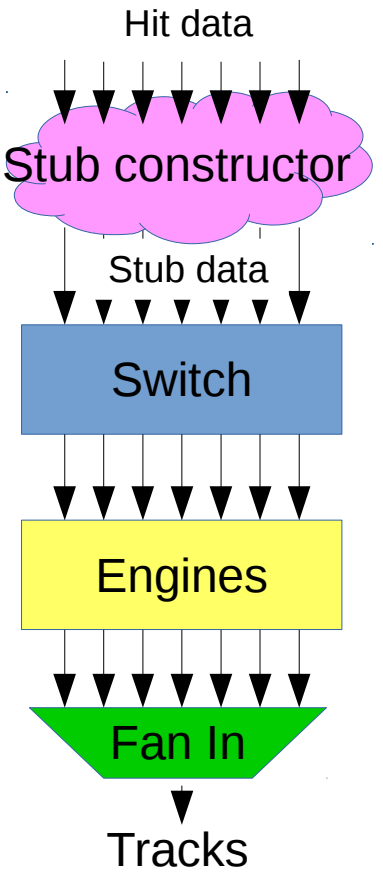


Hardware architecture – Key points

- **Highly parallelized** architecture
- **Pipelined** architecture
- Optimized for **low latency**
  - **<1µs** total latency
- **Hold logic** implemented to manage the data flow when multiple data try to access the same output (in the Switch)



Logic scheme of the Engine



Hardware prototype

- The 4D fast tracking algorithm is being implemented in FPGA on a custom board:
  - **Two Xilinx Virtex Ultrascale FPGAs**
  - High-speed optical transceivers → **up to 1Tbps input data rate per FPGA**
  - One Xilinx Zynq FPGA

	XCVU095
System Logic Cells (K)	1,176
DSP Slices	768
Memory (Mb)	60.8
GTH 16.3 Gb/s Transceivers	32
GTY 30.5 Gb/s Transceivers	32
I/O Pins	832

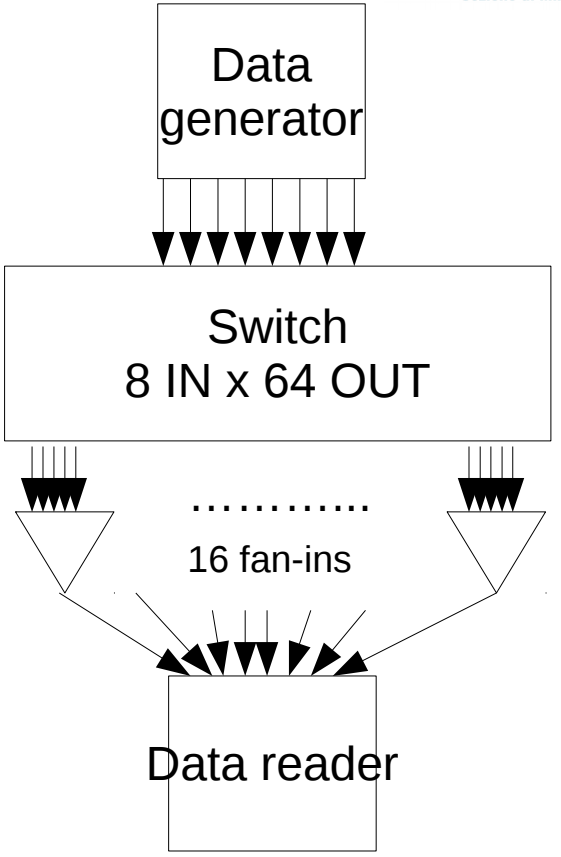
- All the modules described have been implemented using VHDL
  - Synthesizable and behaviorally tested
  - Tests on hardware are ongoing
  - A first test of the Switch module has been performed

Switch test on prototype board

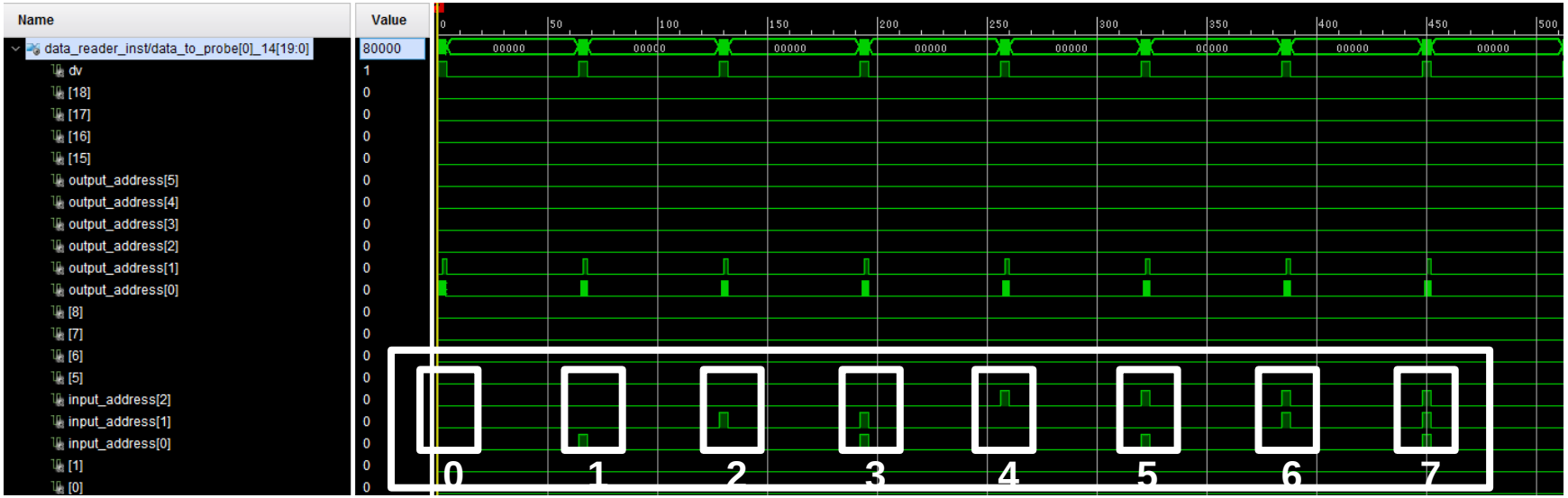
- **Test of the data distribution** through the Switch

- Architecture implemented in a **single FPGA:**
  - Stub data generator
  - 8 inputs / 64 outputs Switch
  - 16 Fan-in modules + data reader

- **Test logic:**
  - One data generated at each clock cycle
  - Loop over the switch inputs
  - For each input data with all the possible “addresses” are generated (the address determines the data path in the switch)



Simple test results



All the data from different inputs and with same address reach the same outptut

The switch has been tested running at 480 MHz clock speed and proved to work.

Conclusions

- A **4D stub based fast tracking algorithm**, is proposed within the TIMESPOT R&D project.
  - Based on FPGA, to reconstruct 4D VELO tracks with with <1μs latency.
  - Tracks as input for the HLT, relieving the workload of the CPUs.

- The algorithm has been simulated, applied to a VELO-like detector **with timing**:
  - Track quality Improvement, Ghost rate reduction
  - Track-vertex association improvement → 2D (Time + Z) track clusterization

- Most of the algorithm has been **implemented in FPGA**:
  - Tests ongoing on a prototype board with Xilinx Virtex Ultrascale FPGAs and high input data rate.
  - A preliminary test of the Switch has been performed in hardware with good results.

Future plans

- Implementation of the **Stub Constructor**:
  - Implemented in FPGA as part of the algorithm
  - Or implemented with dedicated hardware, i.e. Associative Memories
- **Implementation of the full algorithm** on hardware prototype
  - Test of a fast tracking system prototype representative of a sector of the VELO detector
  - Test/evaluation of the maximum data/event rate that can be handled by the system (up to the expected 40MHz LHC bunch crossing rate)
- Possible **extension** of the algorithm for the **VELO+UT tracking:**
  - Addition of an extra track parameter to consider the track curvature in the magnetic field
- **Profit of new architectures** as Intel CPUs with embedded FPGA
  - Growing interest of industry in FPGA as “computing accelerators”
  - CERN Openlab started exploiting hybrid architectures

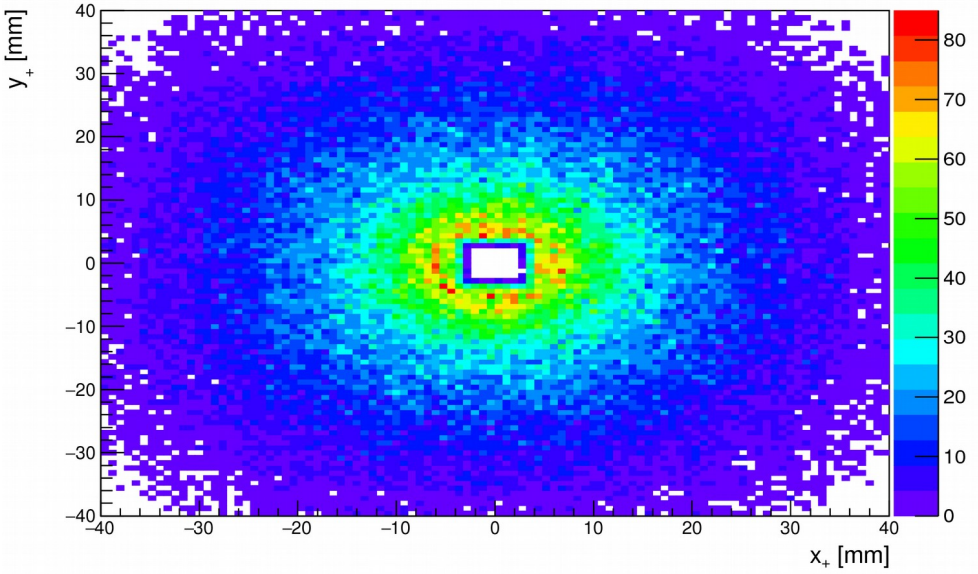
Christian Färber talk at VELO Upgrade Retreat https://goo.gl/ap6J6R

Thanks for your attention

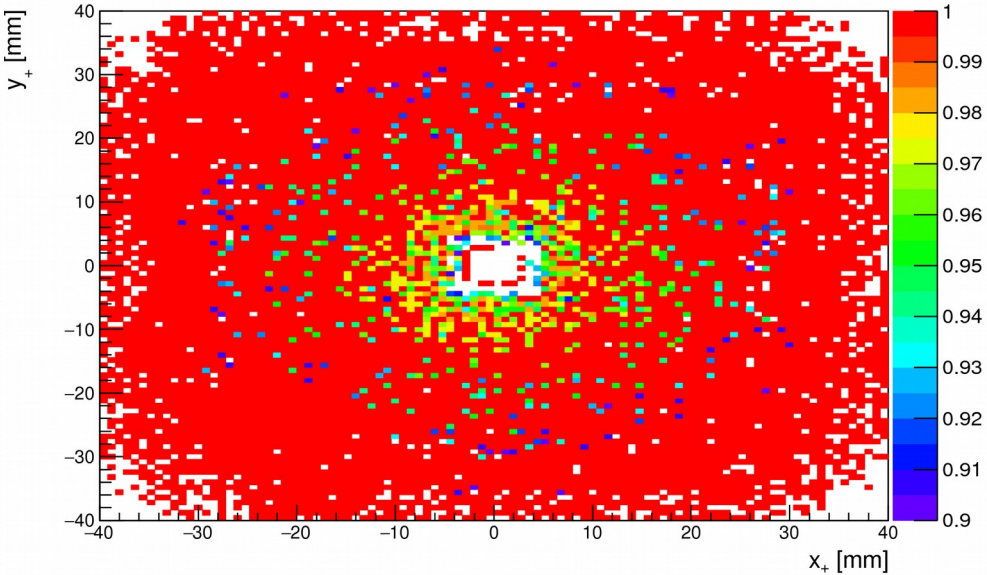
Backup

Simulation results – Tracks distribution and efficiency

- 1200 tracks/event. 30ps time resolution



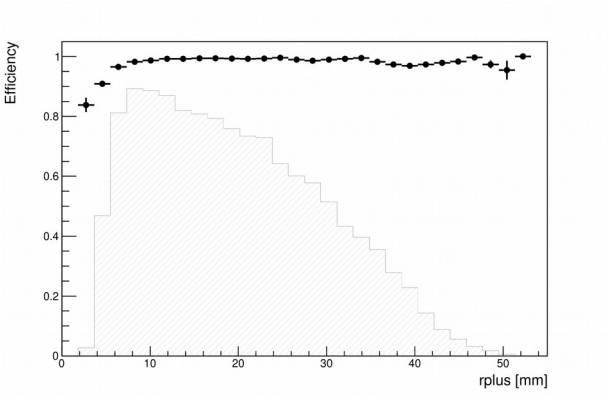
Distribution of the track parameters in the (x+,y+) reference plane at z=z+

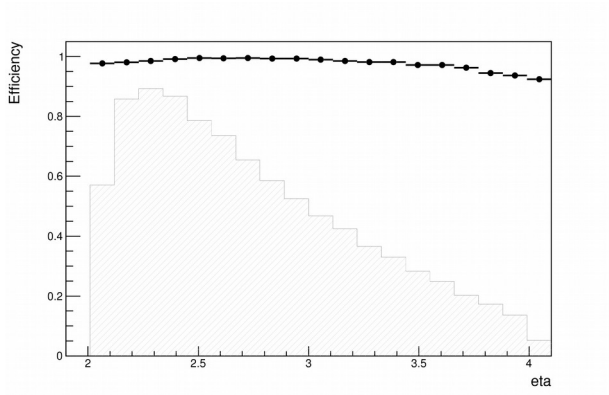


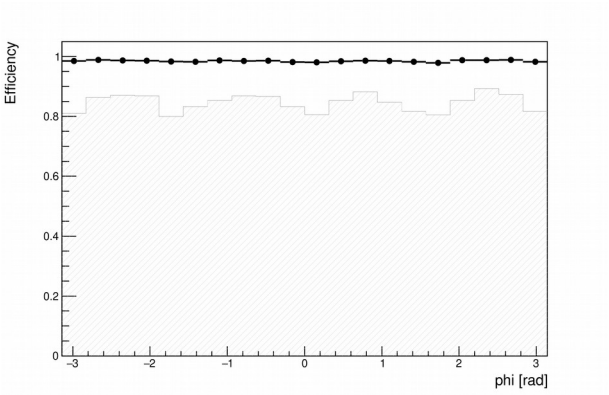
Reconstruction efficiency as a function of the main track parameters (x+,y+)

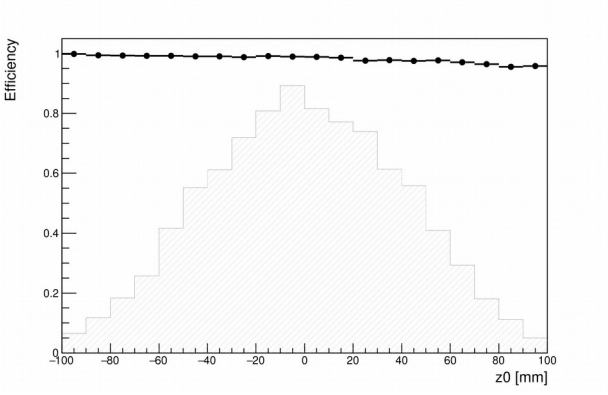
Simulation results – Tracks distribution and efficiency(2)v

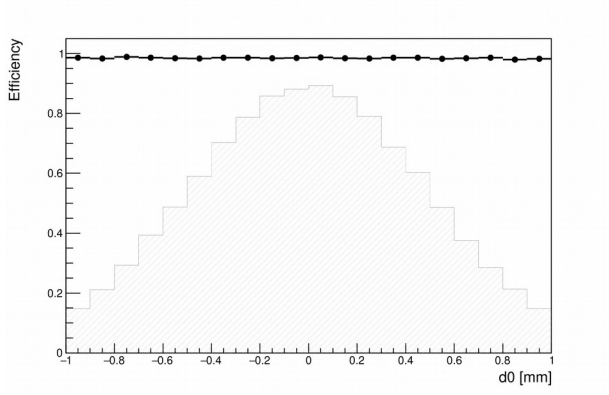
- 1200 tracks/event. 30ps time resolution









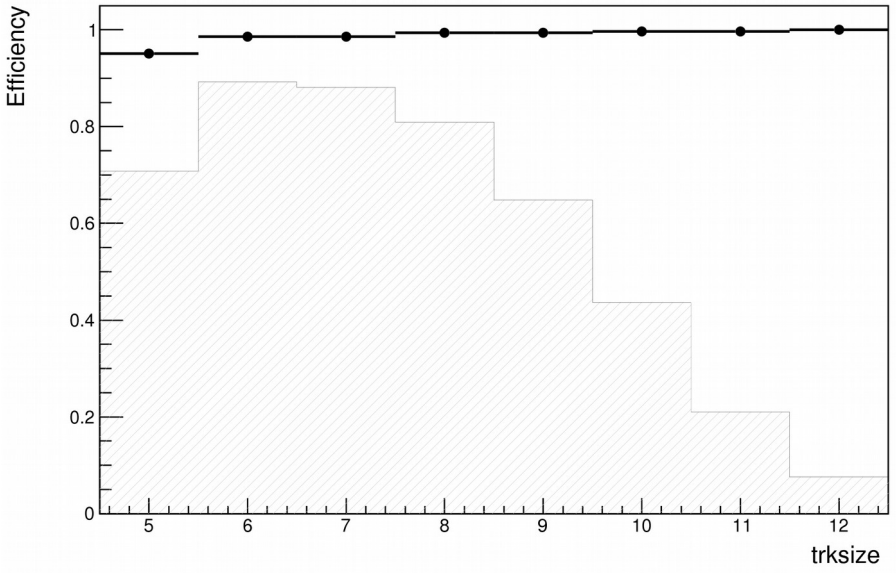


From left to right

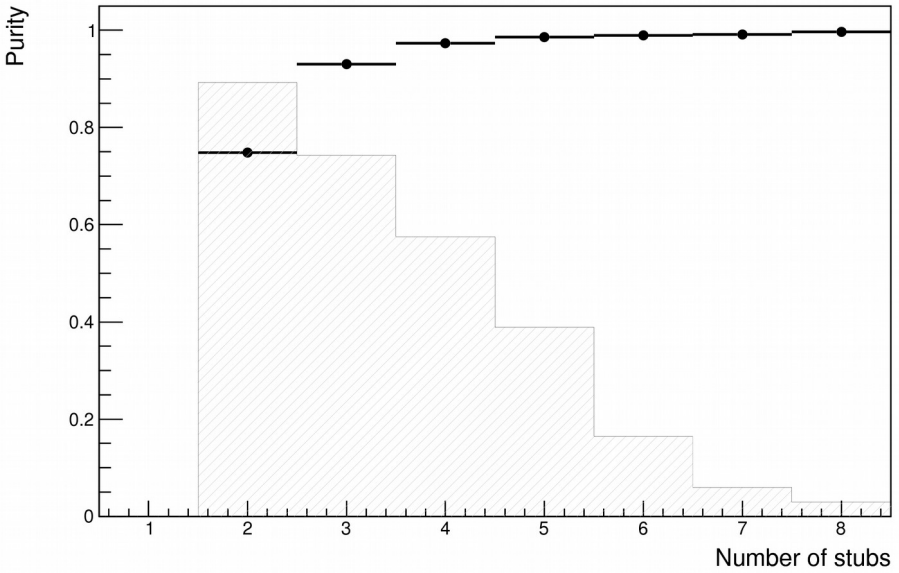
- Efficiency vs. **r+**
- Efficiency vs. **eta**
- Efficiency vs. **phi**
- Efficiency vs. **z0**
- Efficiency vs. **d0**

Efficiency/purity vs track size

- 1200 tracks/event. 30ps time resolution



Efficiency vs #hits/track



Purity vs minimum number #stubs/track

Switch details

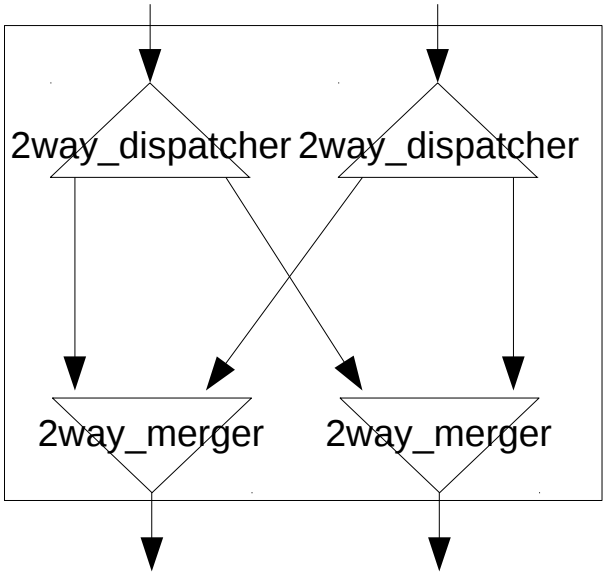
The Switch is based on a network of smaller modules.

The simplest switch has **2 inputs, 2 outputs**, built using:

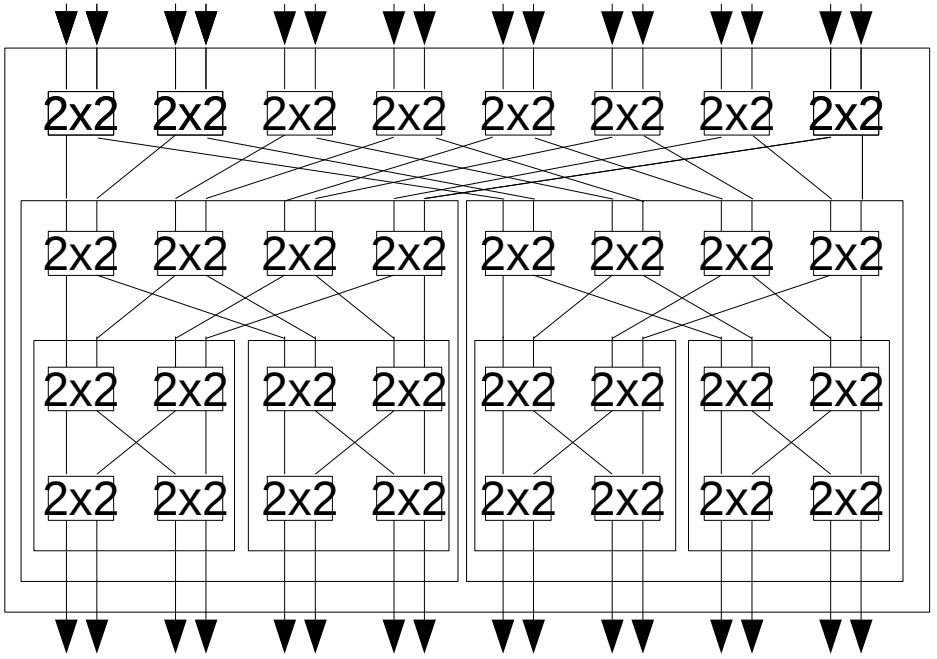
- **Two 2way_dispatchers**
- **Two 2way_mergers**

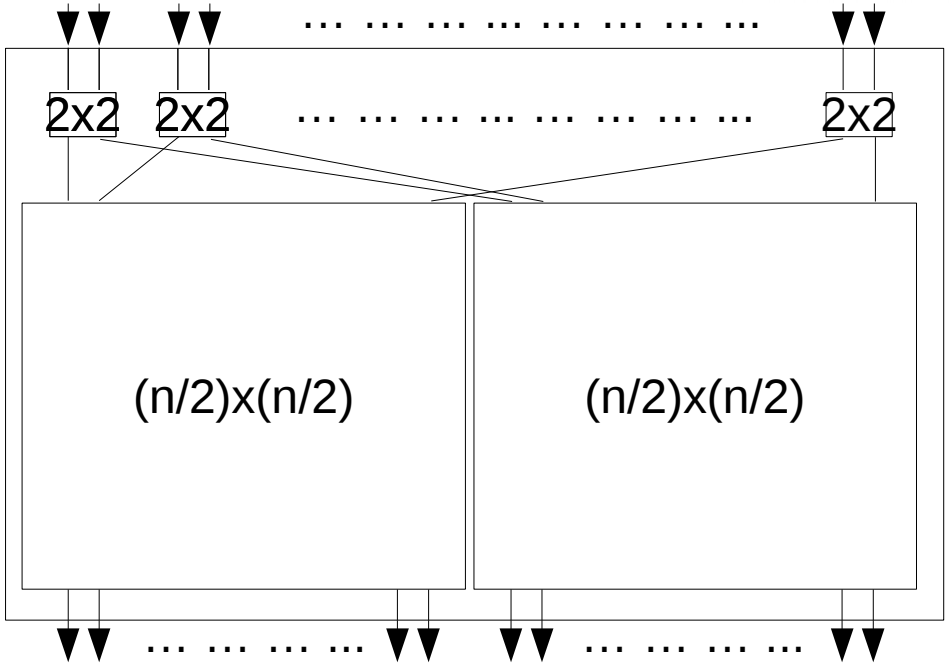
The 2way_dispatchers read the input data and delivers to one or the other output port according to 1 address bit.

The 2way_mergers receives data from two inputs and manage the data flow to the only output port.



Switch details (2)



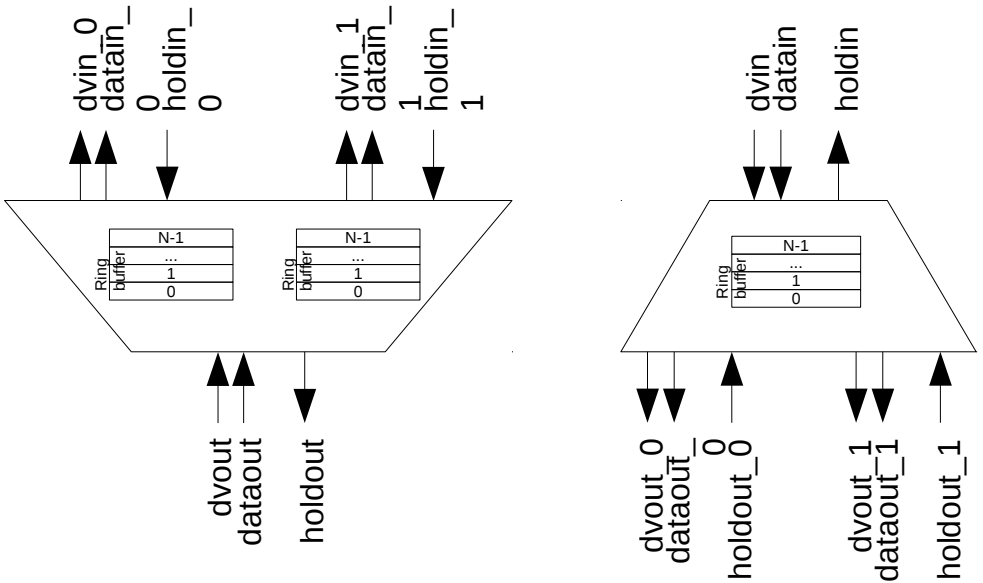


A generic $2^m x 2^m$ switch is built using:

- a layer of $\mathbf{2^{m-1}}$ **(2x2)_switches**
- a layer of **2** **($\mathbf{2^{m-1}x2^{m-1}}$)_switches**

→ Total : $n/2 * \log_2(n)$ (2x2)_switches

Hold logic in the Switch



The **data flow through the switch** is managed by its basic components:

- Dispatchers and mergers have **a ring buffer for each input**
- If the **buffer is full** an **“hold signal”** is back propagated to the previous component

Engine details

- The logic scheme is described in the picture
- **No lateral communications** between the engines
- **The engine evaluates its response to the stubs** and the response of four lateral cells
- **Tracks information are retrieved** by interpolating the response and by averaging the hit values (for SlpX, SlpY, Time)