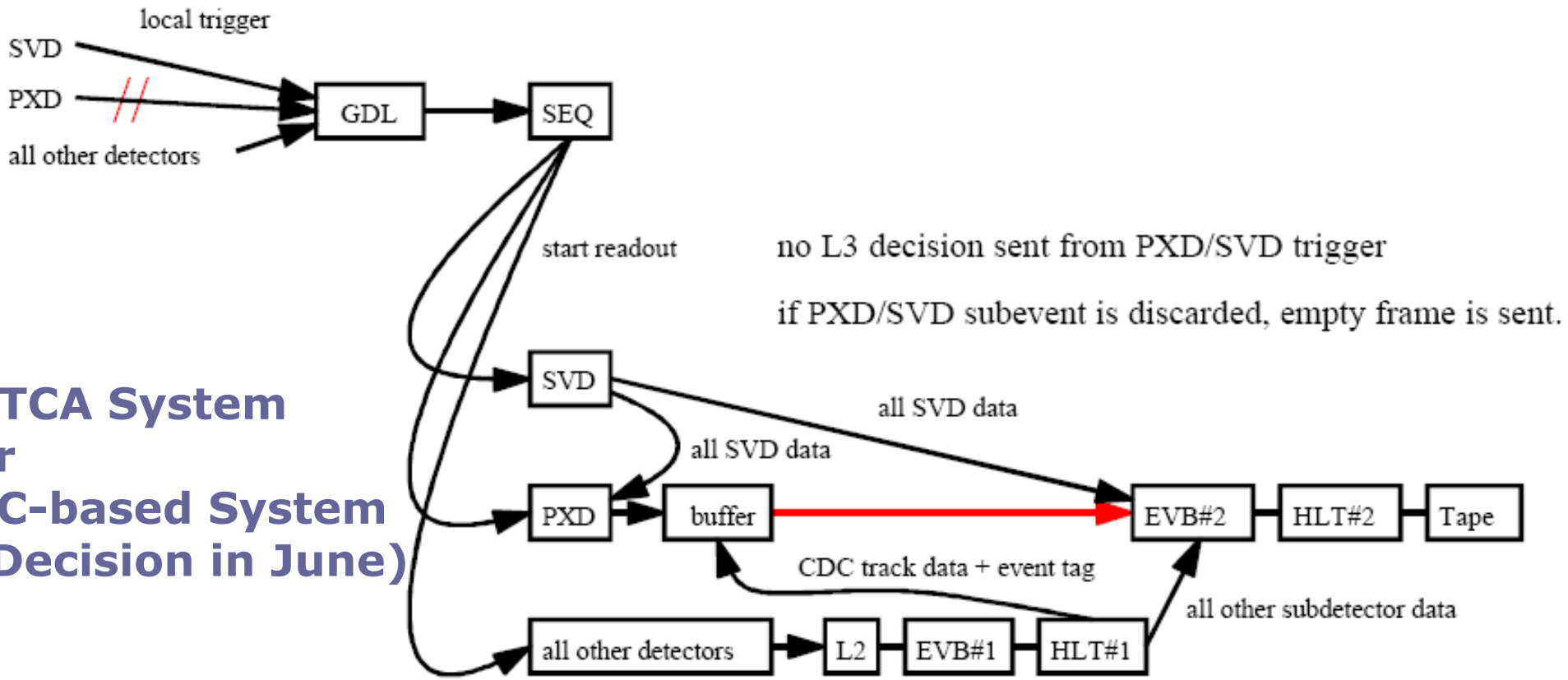
PXD DAQ Algorithms

Sören Lange, David Münchow, Thomas Geßler, Björn Spruck (Univ. Giessen)

Belle-II PXD & HLT Workshop, MPI München, 21.-23.02.2011

ATCA System
or
PC-based System
(Decision in June)



no L3 decision sent from PXD/SVD trigger

if PXD/SVD subevent is discarded, empty frame is sent.

PXD buffer also needs event sorting.

= reduced PXD data

(by factor 10-100)

HLT = High Level Trigger

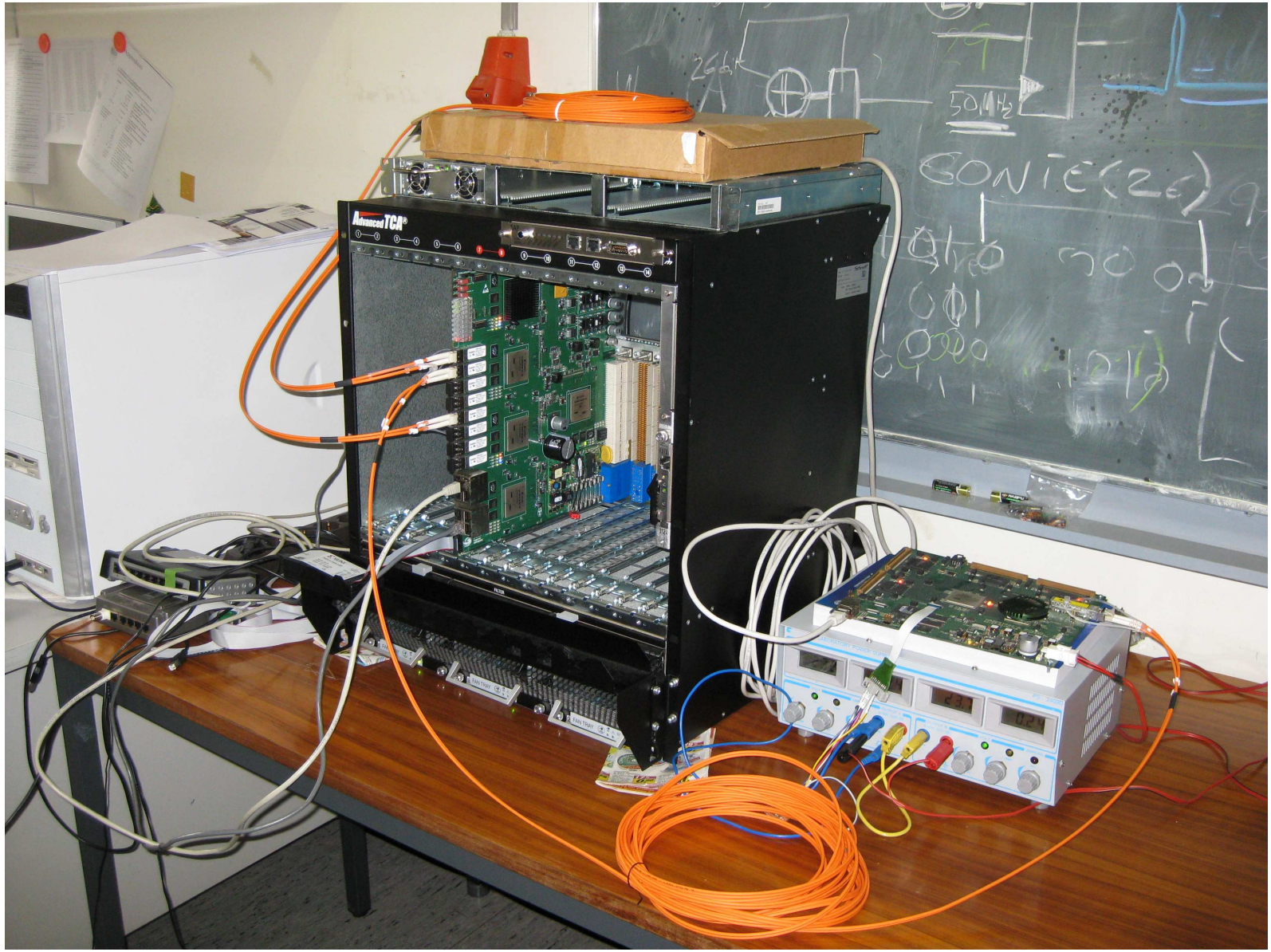
HLT#1 = maybe L3

HLT#2 = maybe L4

HLT#1 runs CDC track finder

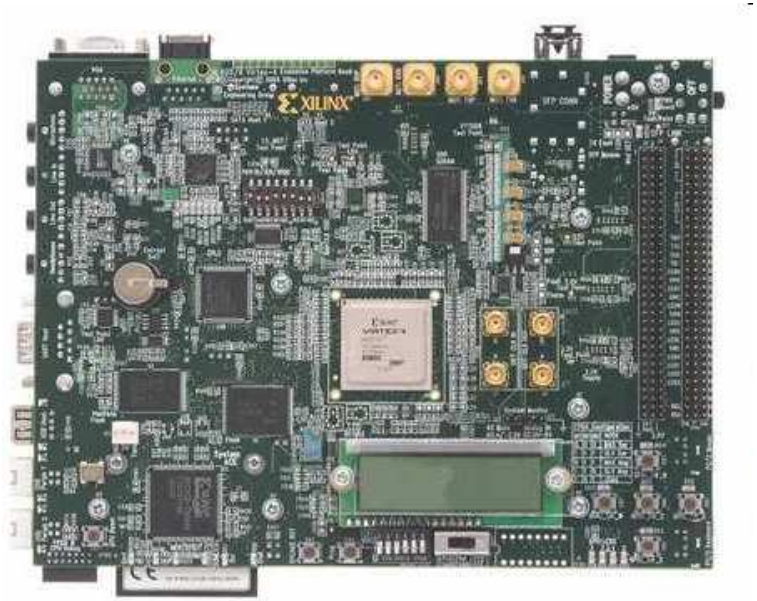
PXD runs PXD+SVD track finder

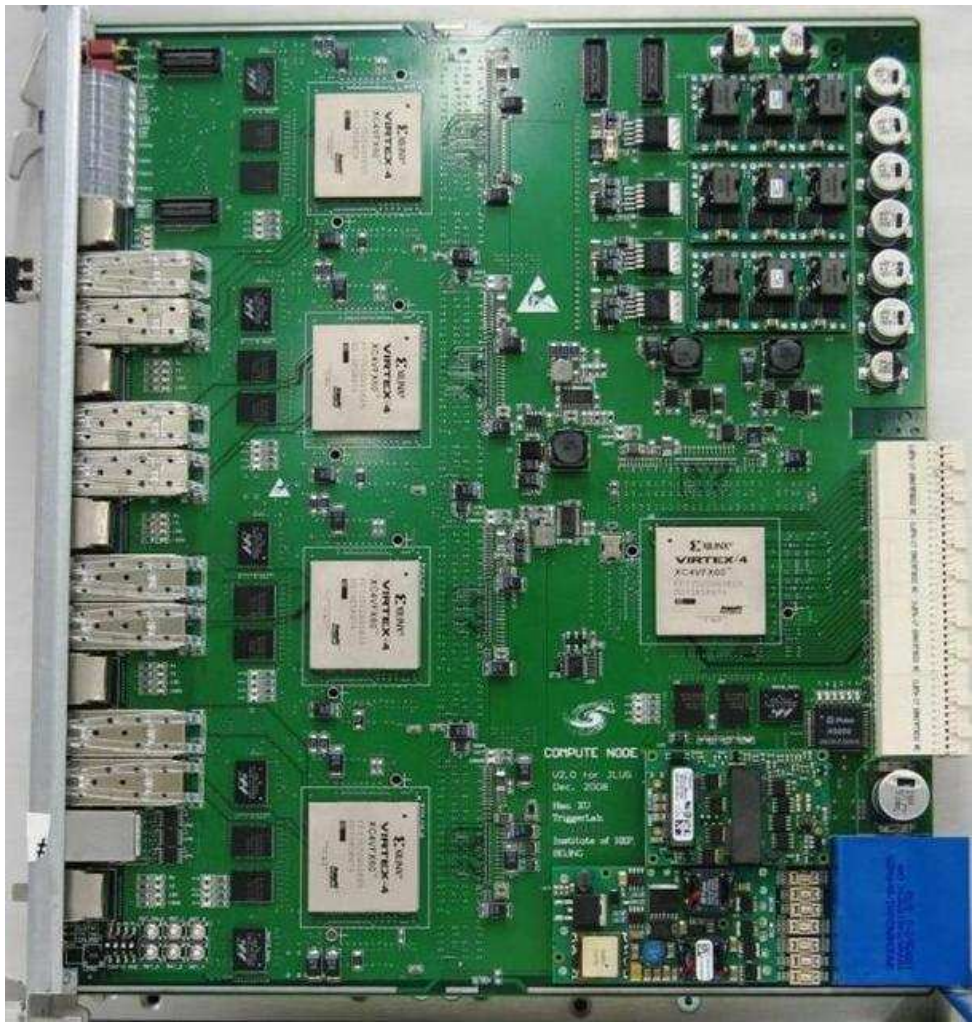
ATCA System



3 platforms supported

ML403 (Virtex-4 FX12)
ML405 (Virtex-4 FX20)
CN (Virtex-4 FX60)





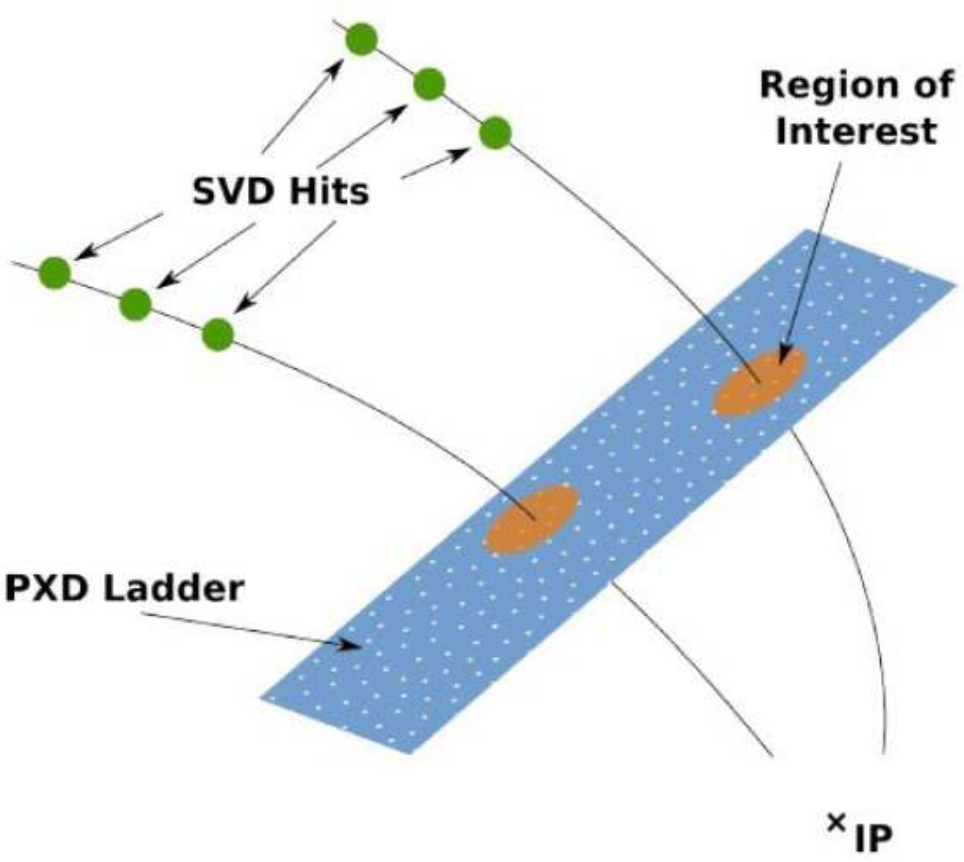
Outline

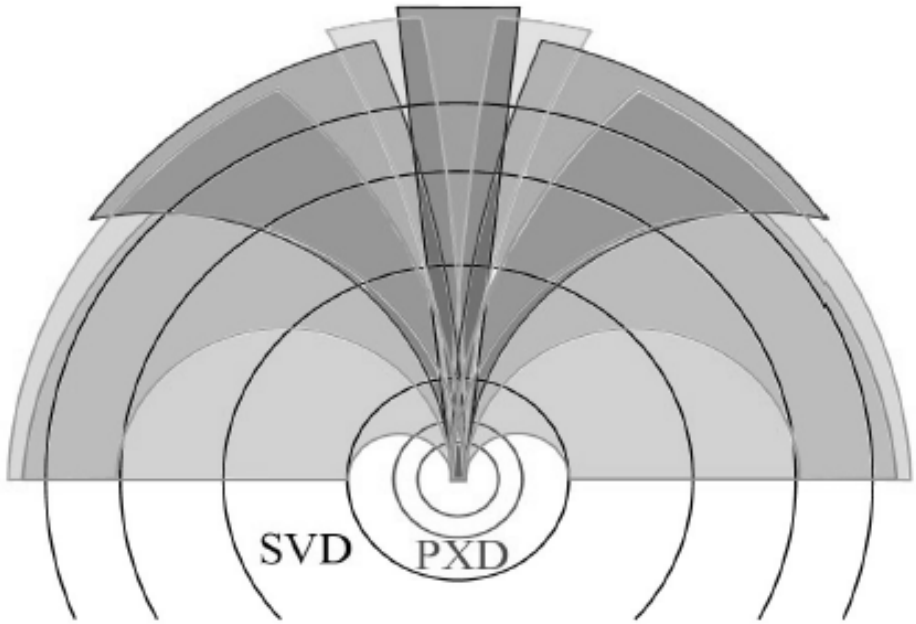
Most important: ROI (region-of-interest)

1. ROI finder by SVD data
   4 hits per tracklet
   - Hough transform
   - maybe with pre-step conformal map
   - ADAPTIVE, fast Hough transform (Claudio Heller)
     because avoids peak finder and speed-up factor >=100
2. RAM access („FPGA shared memory“)
   test bench for receiving ROI from HLT
3. ROI selector, data filter
   (i.e. cut on data inside ROI)

Algorithm on PXDDAQ #1:
ROI SVD-only

Claudio Heller Algorithm for SVD-only ROI Selection





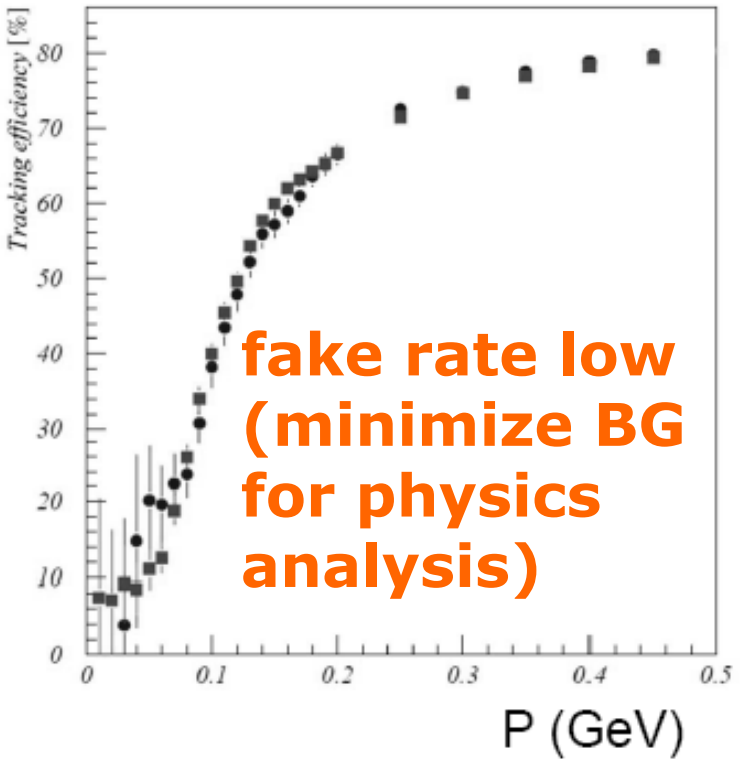
Total # of
sectors and „wedges“
for Hough transform
520
(parallisable!)

- 40 straight (for high pT)
- 8 x 60 curved (for low pT)

Track Efficiencies

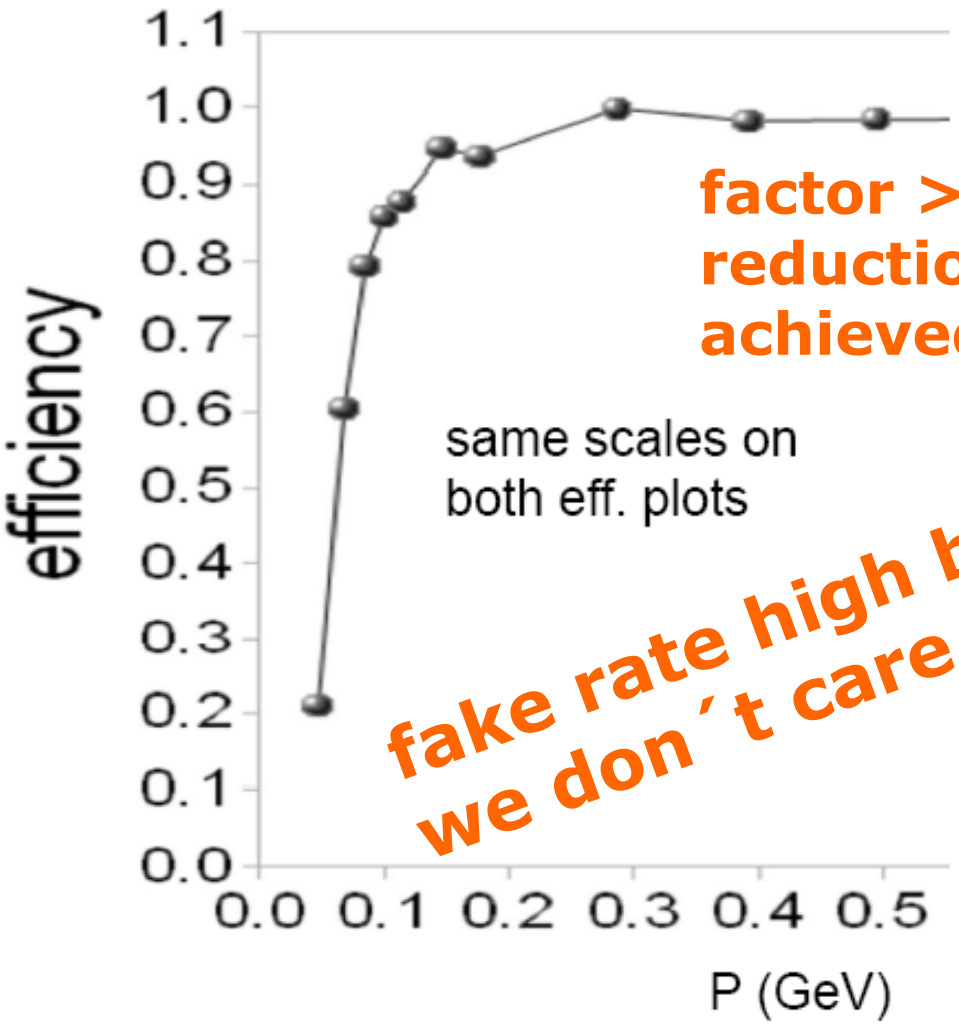
single muons

Full reconstruction, Belle
SVD + CDC



fake rate low (minimize BG for physics analysis)

Hough with SVD, Belle-II



factor >10 reduction achieved

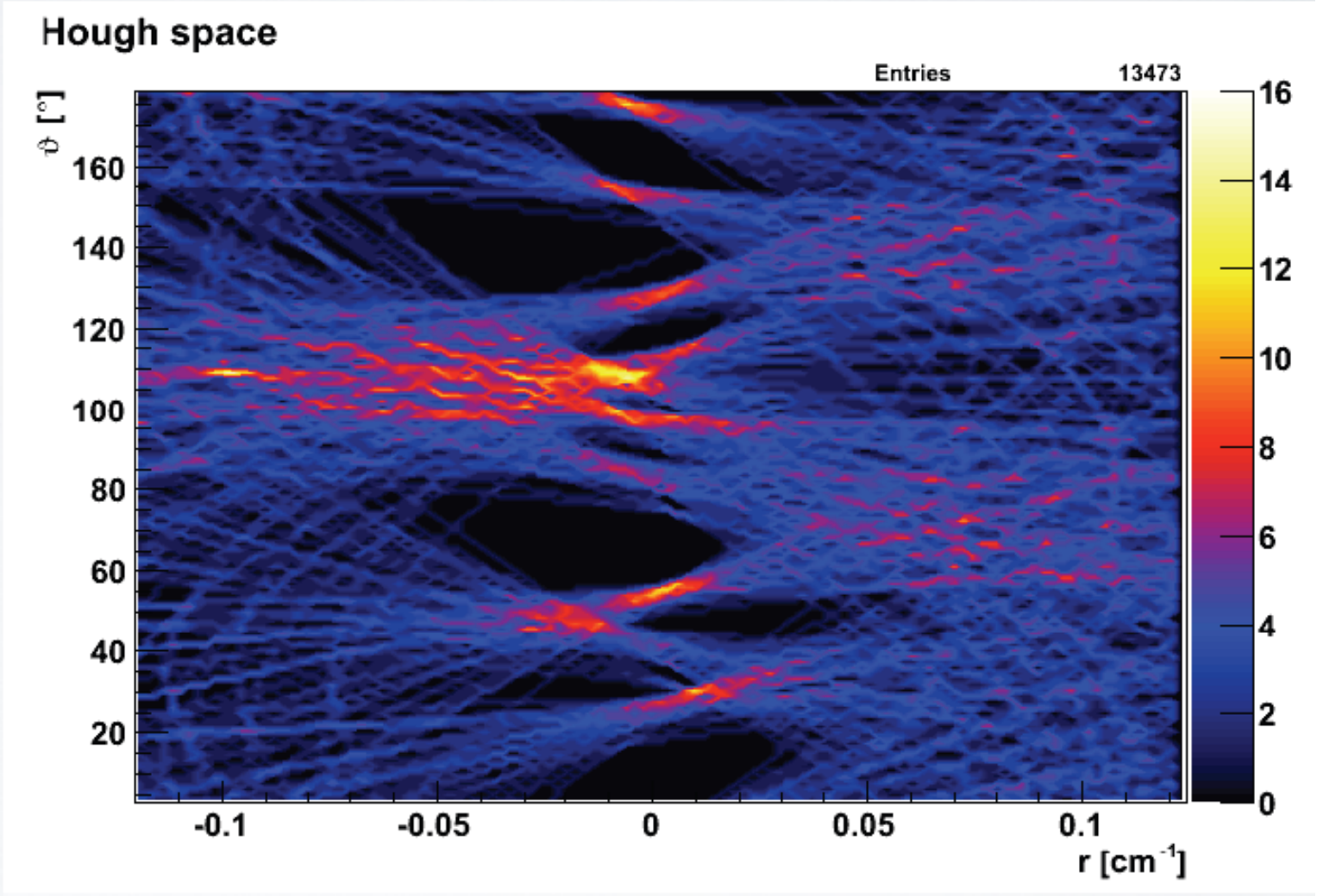
same scales on
both eff. plots

fake rate high but we don´t care !

C. Kiesling, 2nd PXD-DAQ-Meeting, Grünberg, Sep 25-26, 2010

11

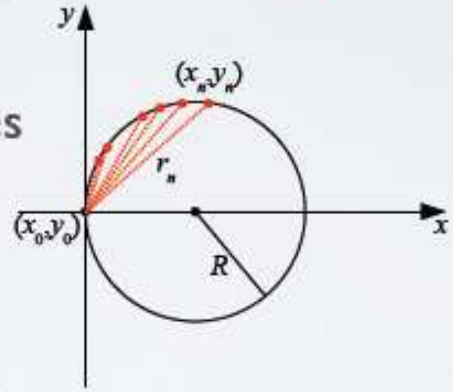
David Münchow (Ph. D. student, Giessen)
Implementation of Panda track finder algorithm conformal map + Hough transform on Virtex-4 FX12 (ML403 board)

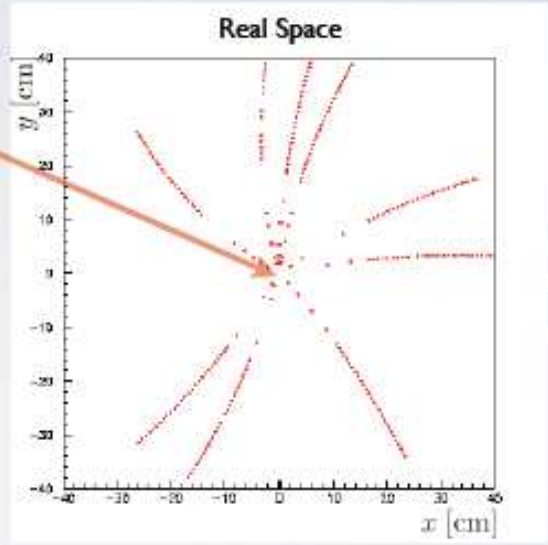


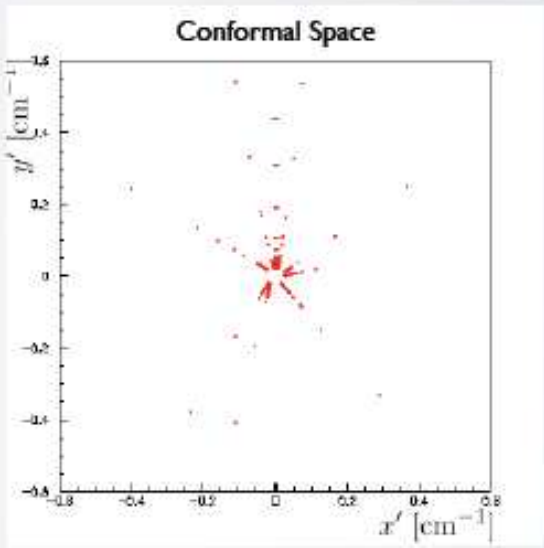
Conformal Transformation

- used for projection perpendicular to the beam direction
- transform circles to straight lines

$$x' = \frac{x - x_0}{r^2}$$
$$y' = \frac{y - y_0}{r^2}$$
$$r^2 = (x - x_0)^2 + (y - y_0)^2$$





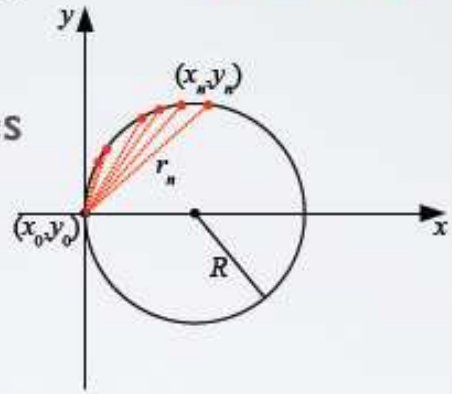


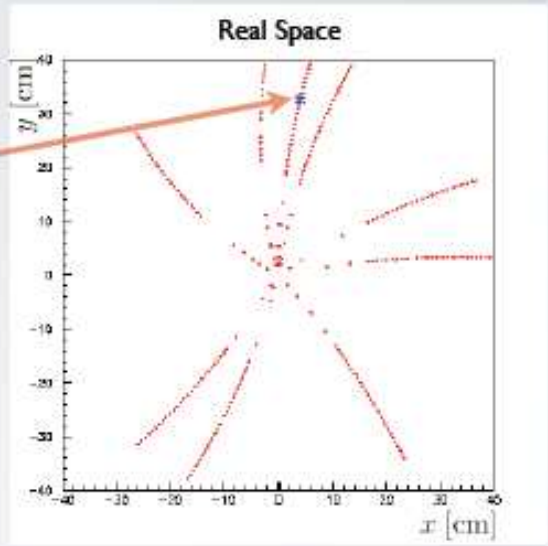
- finding straight lines is less complex than finding circles
- used transformation is vertex (x_0, y_0) constrained

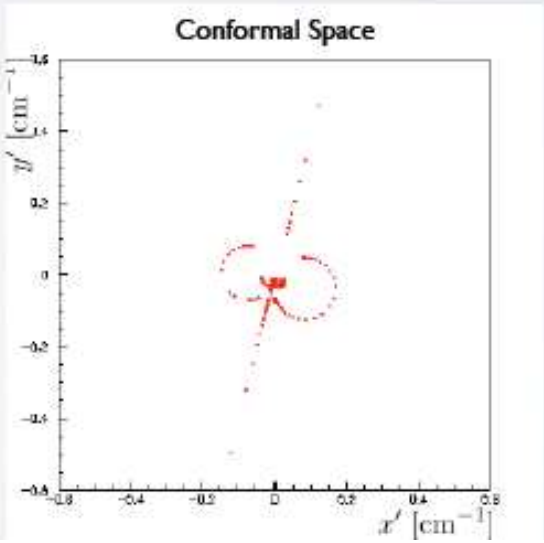
Conformal Transformation

- used for projection perpendicular to the beam direction
- transform circles to straight lines

$$
\begin{aligned}
x' &= \frac{x-x_0}{r^2} \\
y' &= \frac{y-y_0}{r^2} \\
r^2 &= (x-x_0)^2 + (y-y_0)^2
\end{aligned}
$$



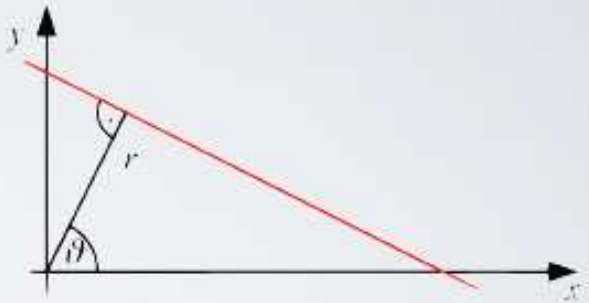




- finding straight lines is less complex than finding circles
- used transformation is vertex (x_0, y_0) constrained

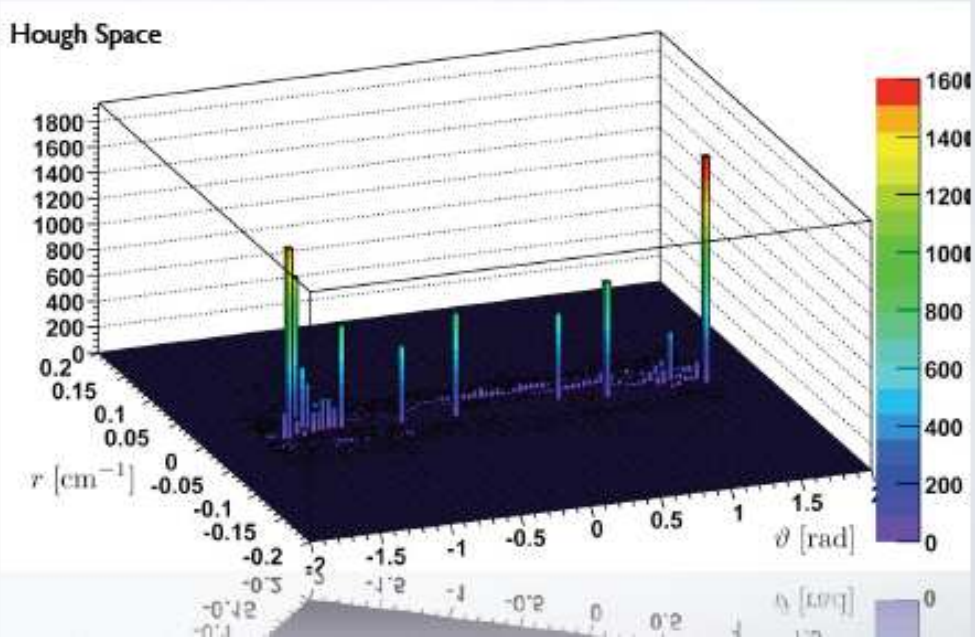
Hough Transformation

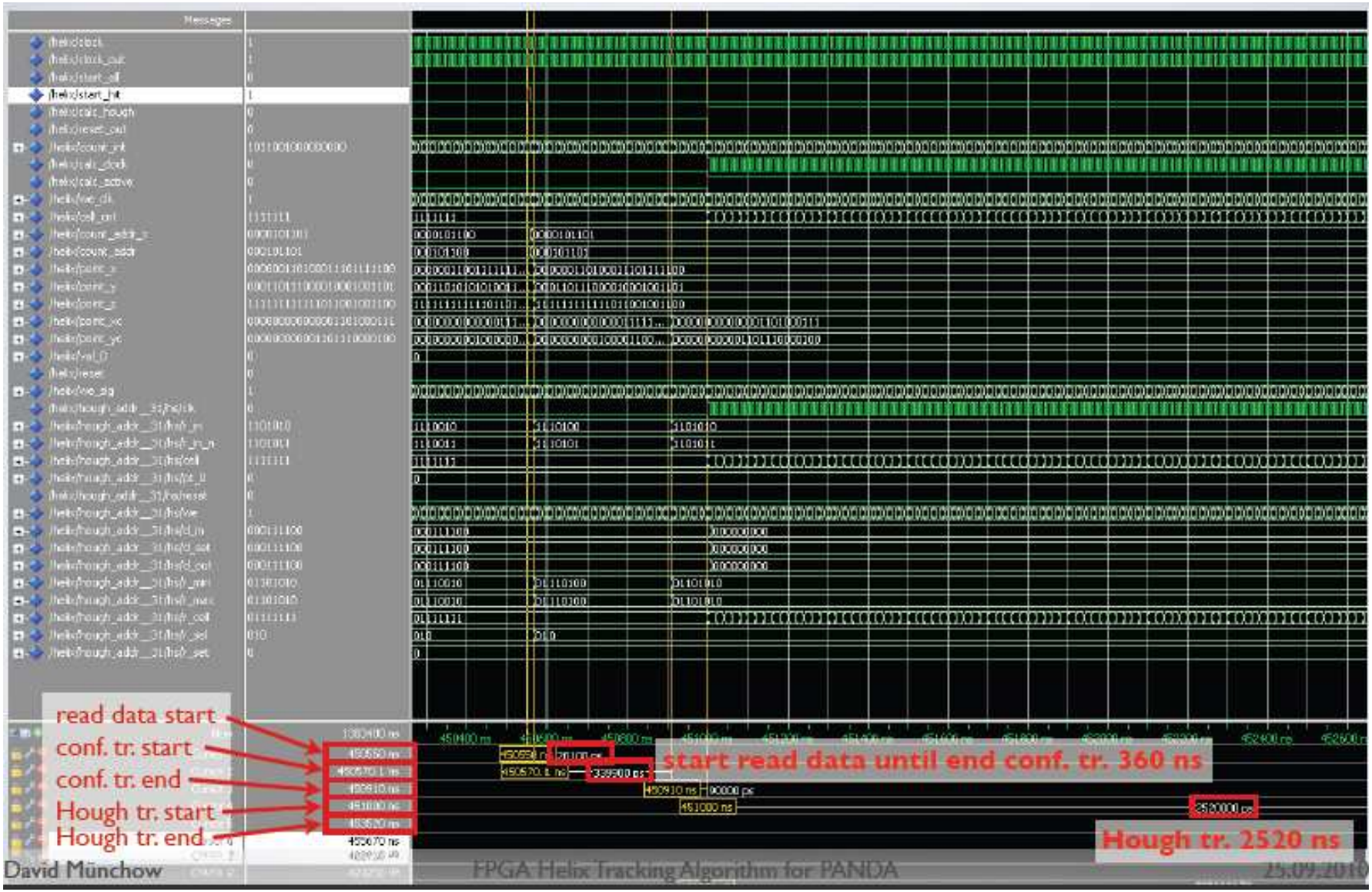
- Describing points in real space by parameter
  - For lines: r and ϑ



$$r = x \cdot \cos(\vartheta) + y \cdot \sin(\vartheta)$$

- Use all possible angles
- Save data in histogram
- Peaks in histogram represent possible lines in point set

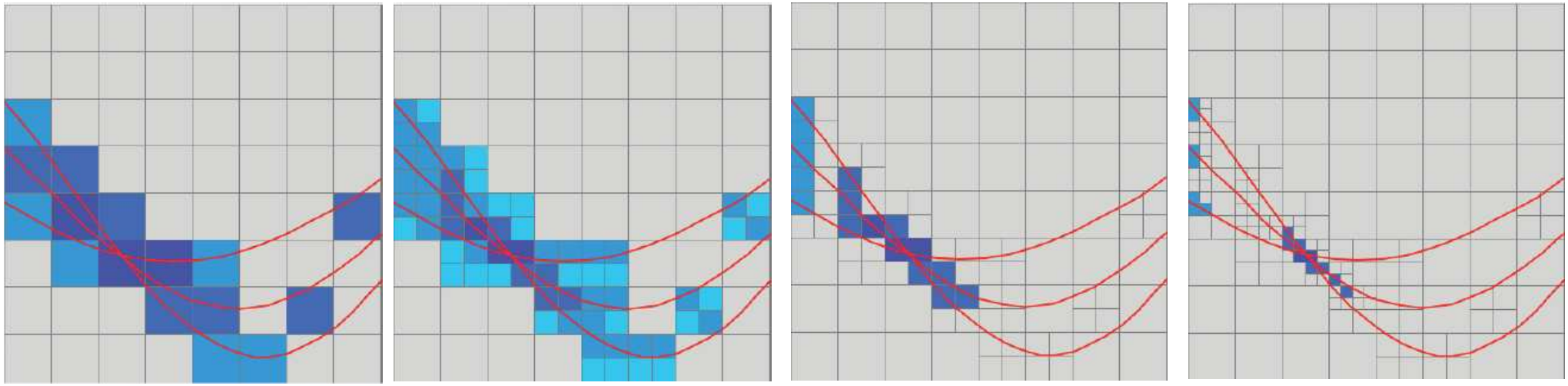


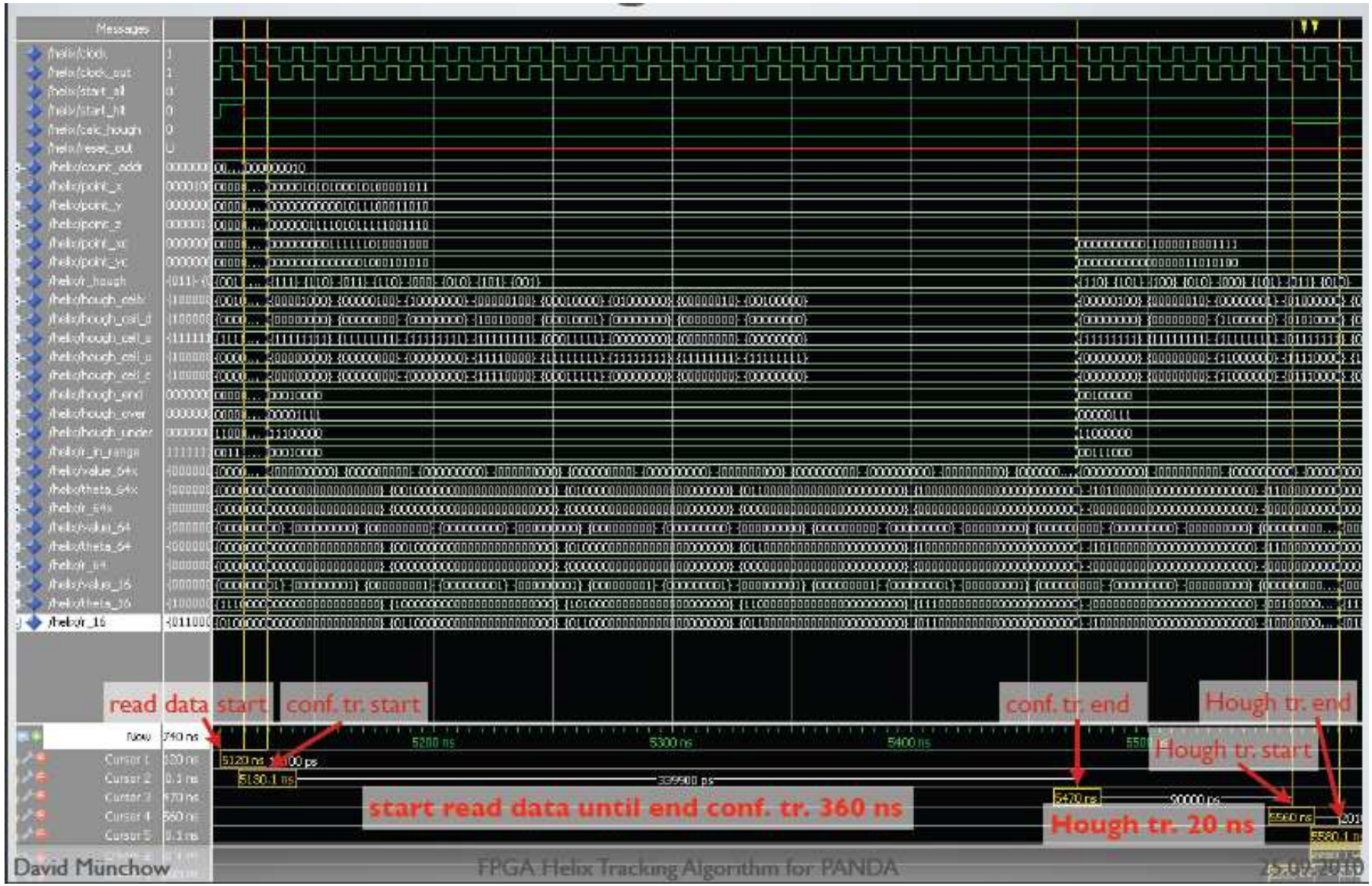

read data start
conf. tr. start
conf. tr. end
Hough tr. start
Hough tr. end
start read data until end conf. tr. 360 ns
Hough tr. 2520 ns
David Münchow
FPGA Helix Tracking Algorithm for PANDA

Fast Hough Transform:

= two tasks at same time:

1. adaptive iterative Hough transform („zooming“)
2. peak finder



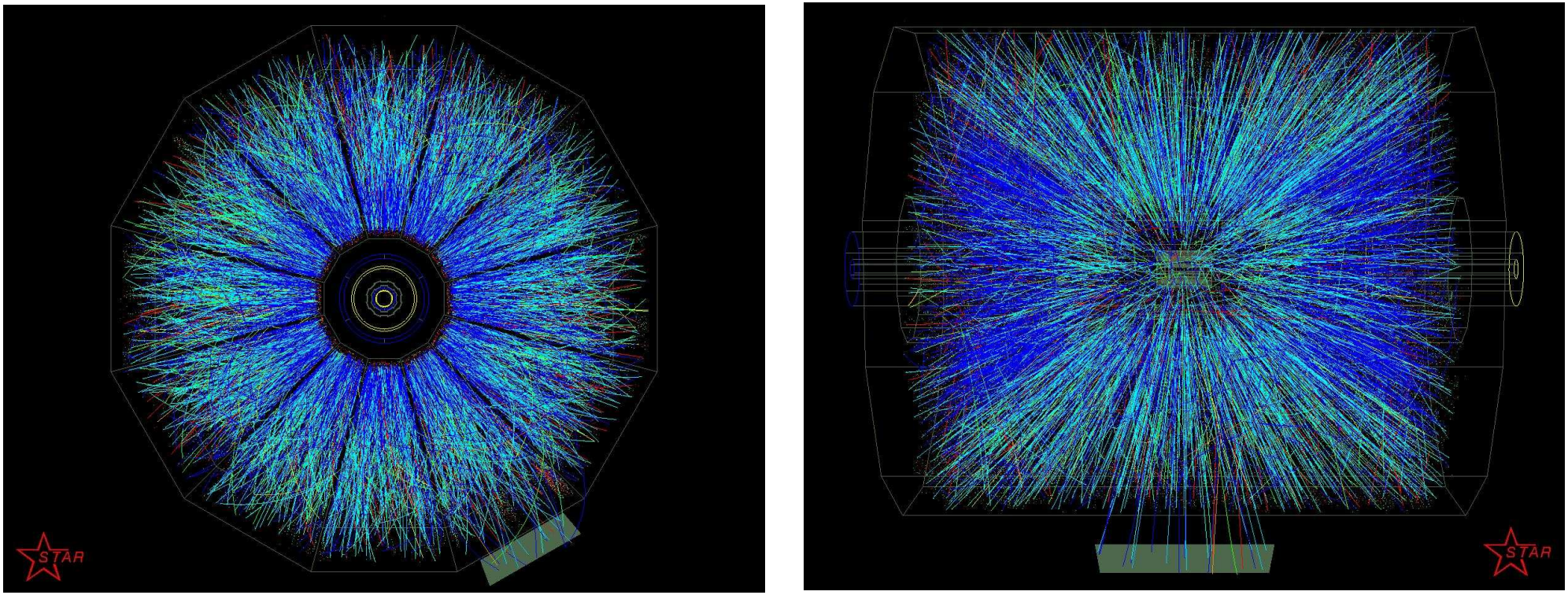

read data
start
conf. tr. start
conf. tr. end
Hough tr. end
Hough tr. start
start read data until end conf. tr. 360 ns
Hough tr. 20 ns
David Münchow
FPGA Helix Tracking Algorithm for PANDA

FPGA Algorithm Timing results

- Read data + conformal transformation
  360 ns per 1 hit
- Hough transform → Fast Hough transform
  2520 ns → 20 ns per 1 hit
  (64 cells parallel)
- Fast Hough transform requires hit sorting
  sorting algorithm was implemented
  unclocked (!)
  sorting is included in 20 ns
- Comparison between PC and FPGA:
  scaled to 800 x 800
  (instead of 128 x 128 for fast Hough, 5 steps)
  2nd z Hough transform
  max. 512 hits, 10 muon mit 2 GeV (~300 hits)
  2.5×10^3
- large factor because of parallelization
  - Hough space in ϑ is parallized (but r serial)
  - per step 64 cells parallel (in fast Hough transform)
  - divider is not parallized yet

Why „conformal map“ pre-step?

tested and works very well for high track density



Au+Au collision at √s=200 GeV

STAR „HLT"

ELSEVIER

Nuclear Instruments and Methods in Physics Research A 453 (2000) 397–404

NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH Section A

www.elsevier.nl/locate/nima

The STAR level-3 trigger system

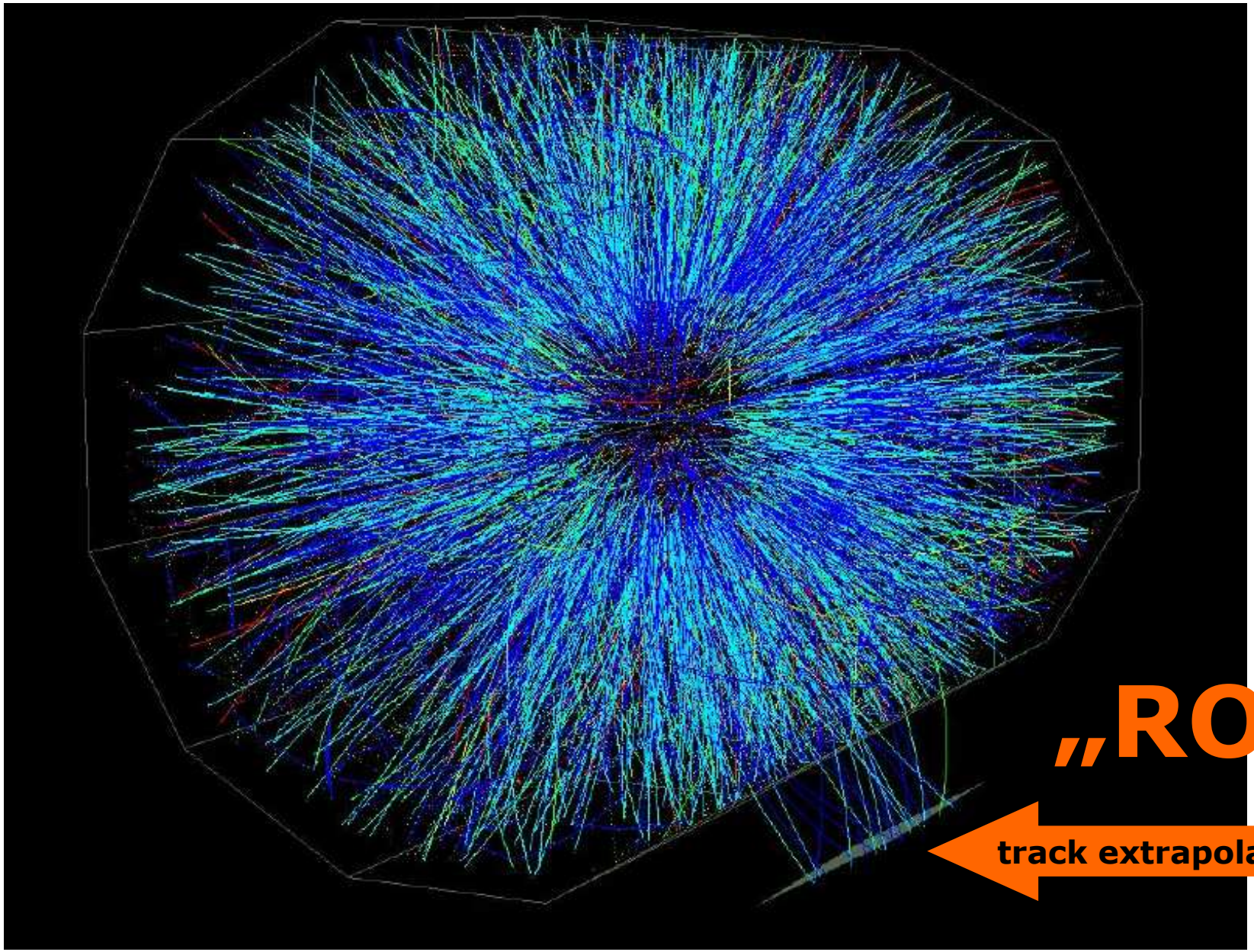
J.S. Lange[a,*], C. Adler[a], J. Berger[a], M. Demello[b], D. Flierl[a], J. Landgraf[c], M.J. LeVine[c], A. Ljubicic Jr.[c], J. Nelson[d], D. Roehrich[e], J.J. Schambach[f], D. Schmischke[a], M.W. Schulz[g], R. Stock[a], C. Struck[a], P. Yepes[b]

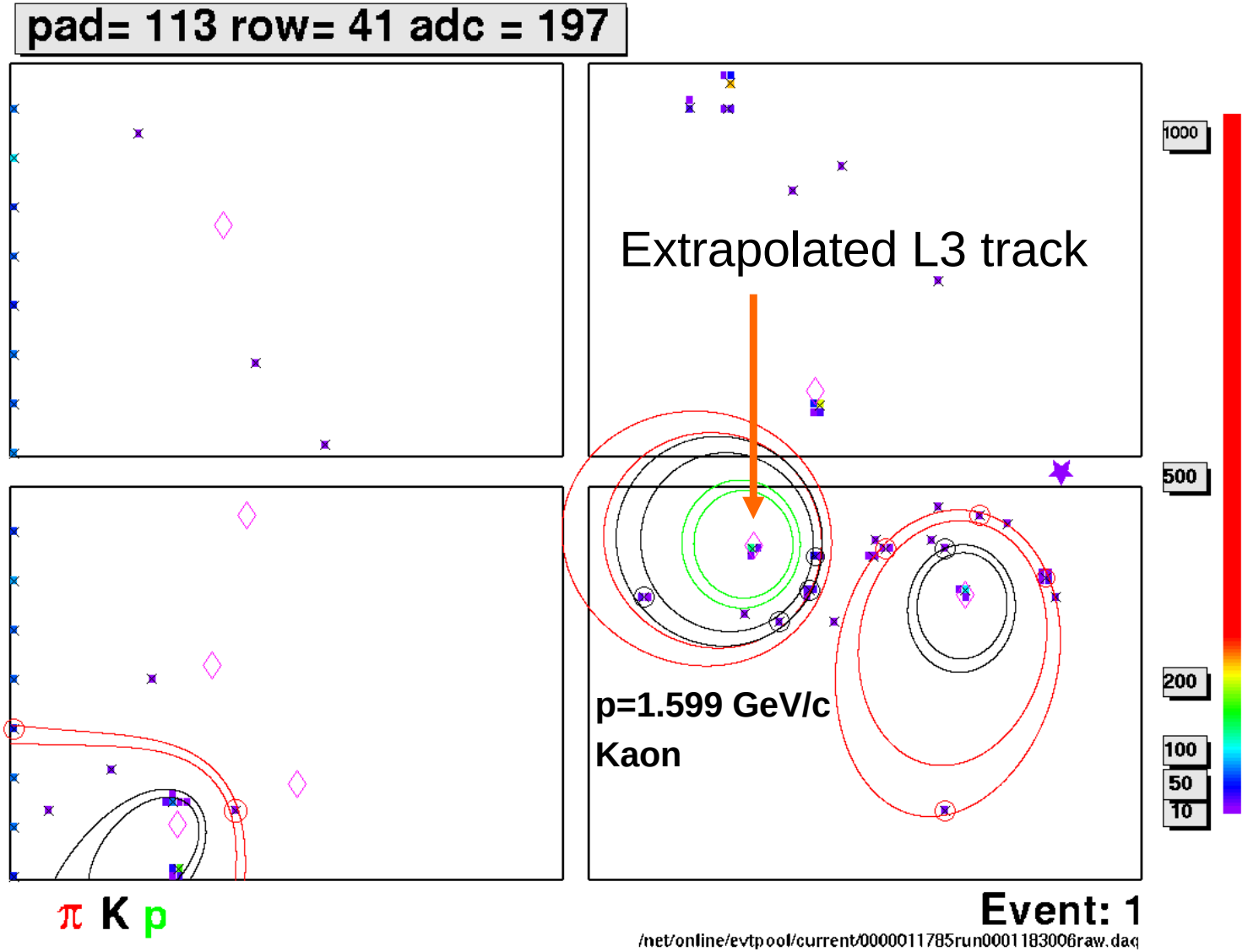
[a]*University of Frankfurt, August-Euler-Straße 6, D-60486 Frankfurt, Germany*
[b]*Rice University, Houston, Texas 77251, USA*
[c]*Brookhaven National Laboratory, Upton, New York 11973, USA*
[d]*University of Birmingham, Birmingham B15 2TT, United Kingdom*
[e]*University of Bergen, Allegaten 55, 5007 Bergen, Norway*
[f]*University of Texas, Austin, Texas 78712, USA*
[g]*University of Heidelberg, Philosophenweg 12, 69120 Heidelberg, Germany*

Accepted 20 June 2000

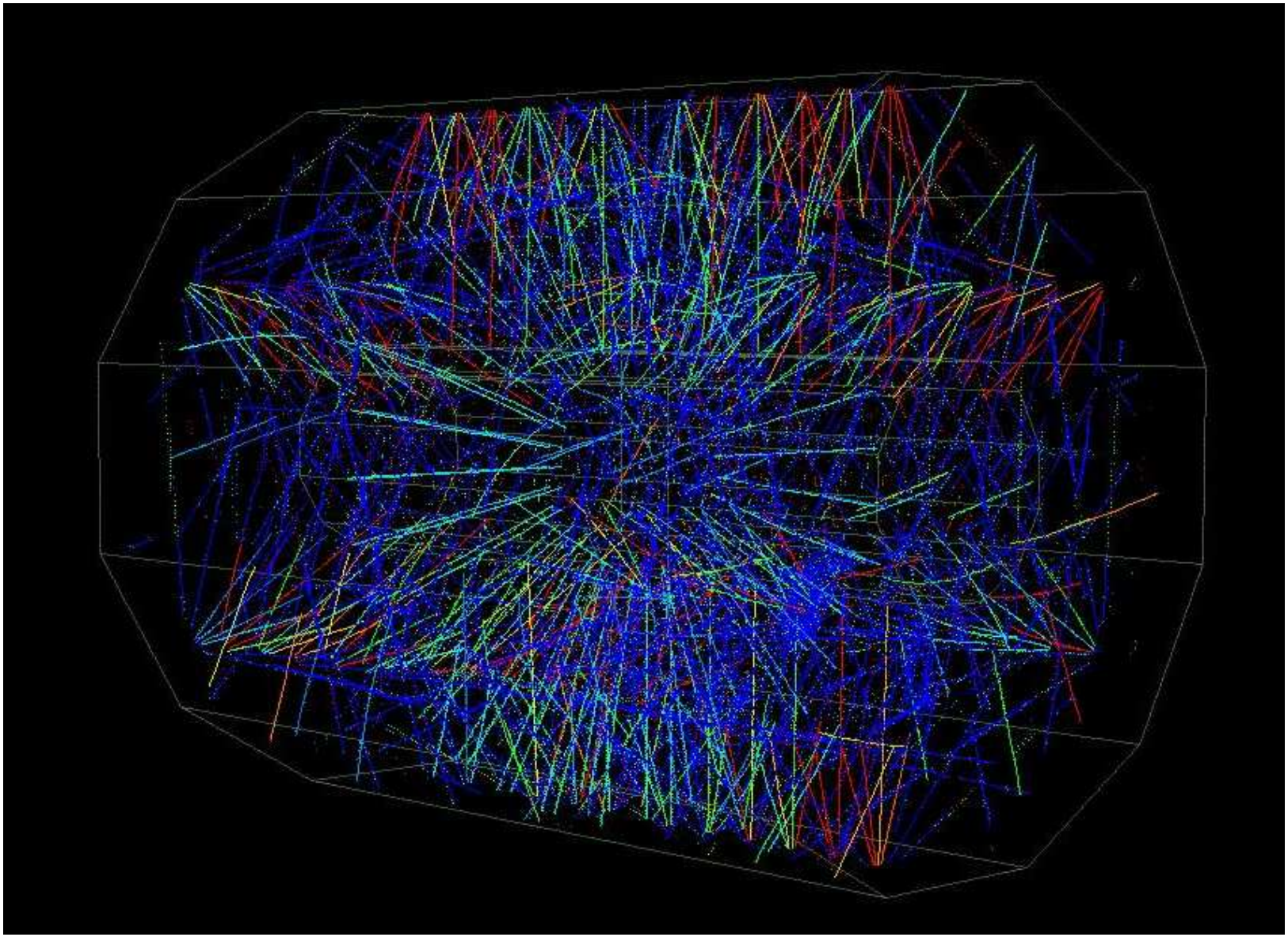
Abstract

The STAR level-3 trigger is a MYRINET interconnected ALPHA processor farm, performing online tracking of $N_{track} \geq 8000$ particles ($N_{point} \leq 45$ per track) with a design input rate of $R = 100$ Hz. A large-scale prototype system was tested in 12/99 with laser and cosmic particle events.


„ROI"
track extrapolation


pad= 113 row= 41 adc = 197
Extrapolated L3 track
1000
500
200
100
50
10
p=1.599 GeV/c
Kaon
π K p
Event: 1
/net/online/evtpool/current/0000011785run0001183006raw.daq

Laser Events for TPC Drift Velocity
Track Finder works for straight tracks!
Curvature = ∞



Conformal Map Track Finder and Track Fitter

- finding a straight line is always easier than finding a helix (circle)
- works for high occupancy
- works for secondary vertices
- works even for straight tracks (e.g. cosmics)
- Code:
  - stand-alone C/C++ version
  - version in PandaRoot
    (Oleg Rogachevsky, Mohammad Al-Turany, S.L.)
    - with genfit (Kalman filter) post-step
      (Christian Höppner, Sebastian Neubert, Lia Lavezzi)
      because genfit needs a pre-ordered chain of hits,
      i.e. it needs a pre-track finder
  - VHDL version (i.e. lookup tables)
    (David Münchow)

DDR2 RAM Access: Shared Memory Concept

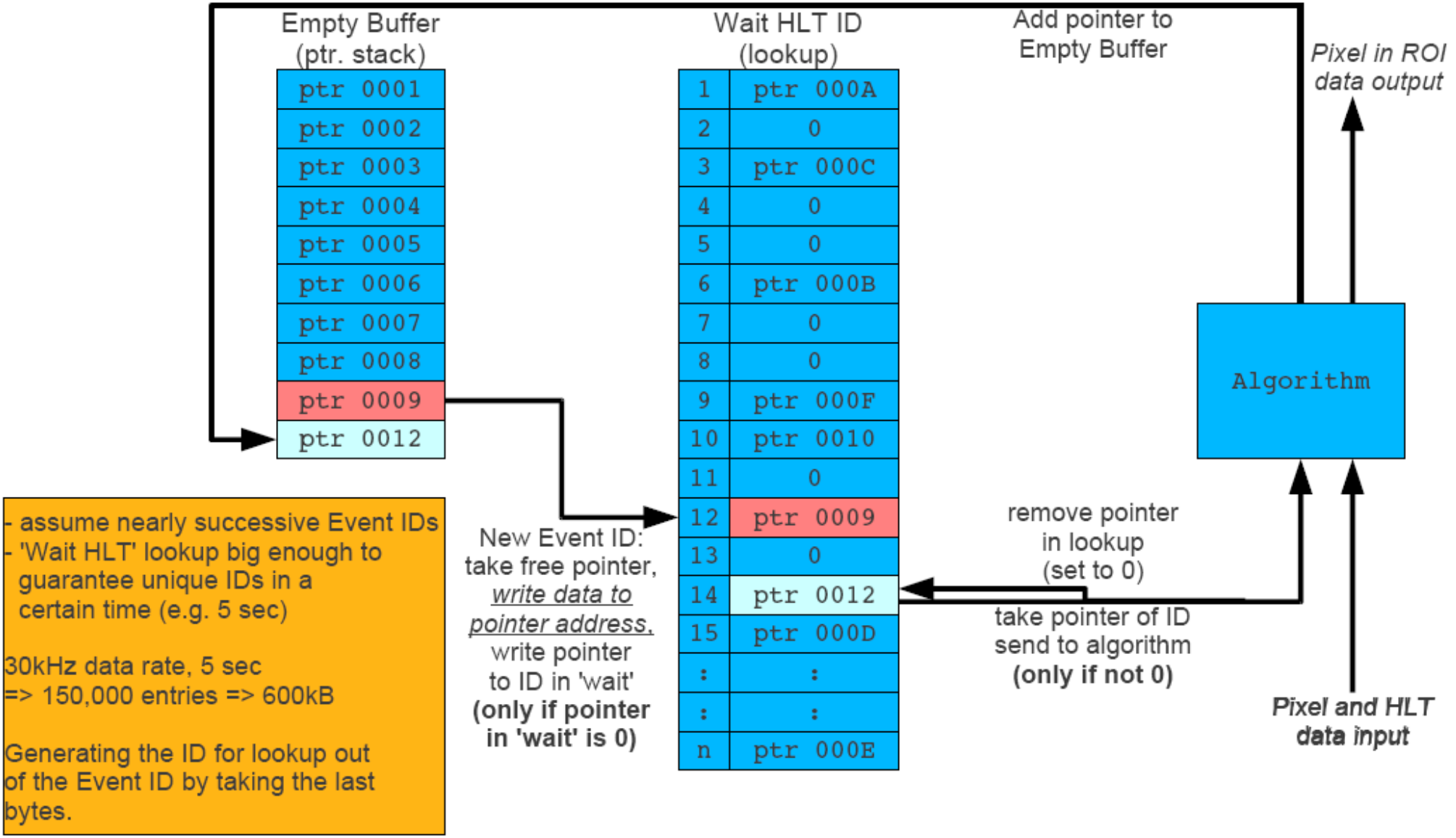
„Shared Memory“

- FPGA ip core (SVD-only ROI)
  write ROI
- FPGA ip core (receive ROI from HLT)
  write ROI
- FPGA ip core (receive PXD data by optical link)
  write data
- ROI selector
  read unfiltered data
  read ROI
  write filtered data
- …

- addresses to e.g. 30 kB blocks (fixed block size)
  (although variable event size, but otherwise too complicated)
- addresses exchanged by registers

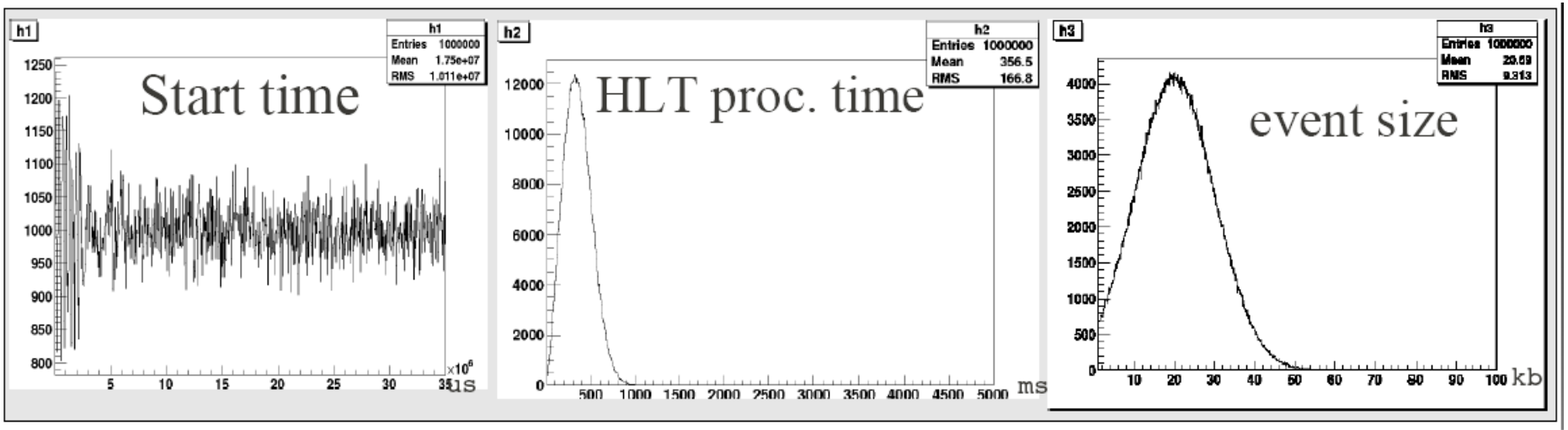
STACK principle

- *Data written to RAM once* → afterwards managed by Pointer



- Simulation to check the memory usage for different scenarios
  - On PC
  - Different Memory Management Algorithms
  - Input: Distributions for time, HLT processing time, event size
- KISS principle, using the simplest Algorithm possible
- Test Bench:
  - PC send data according to distributions
  - CN processes data as it would be in final stage
  - memory management
  - CPU usage, Network bandwidth, find performance or implementation problems, etc
- Simulate and check error conditions!
  - Missing pixel or HLT data, memory full, etc...

Björn Spruck



1M events in 35s (~30kHz), HLT proc time and size distributed to “some” functions

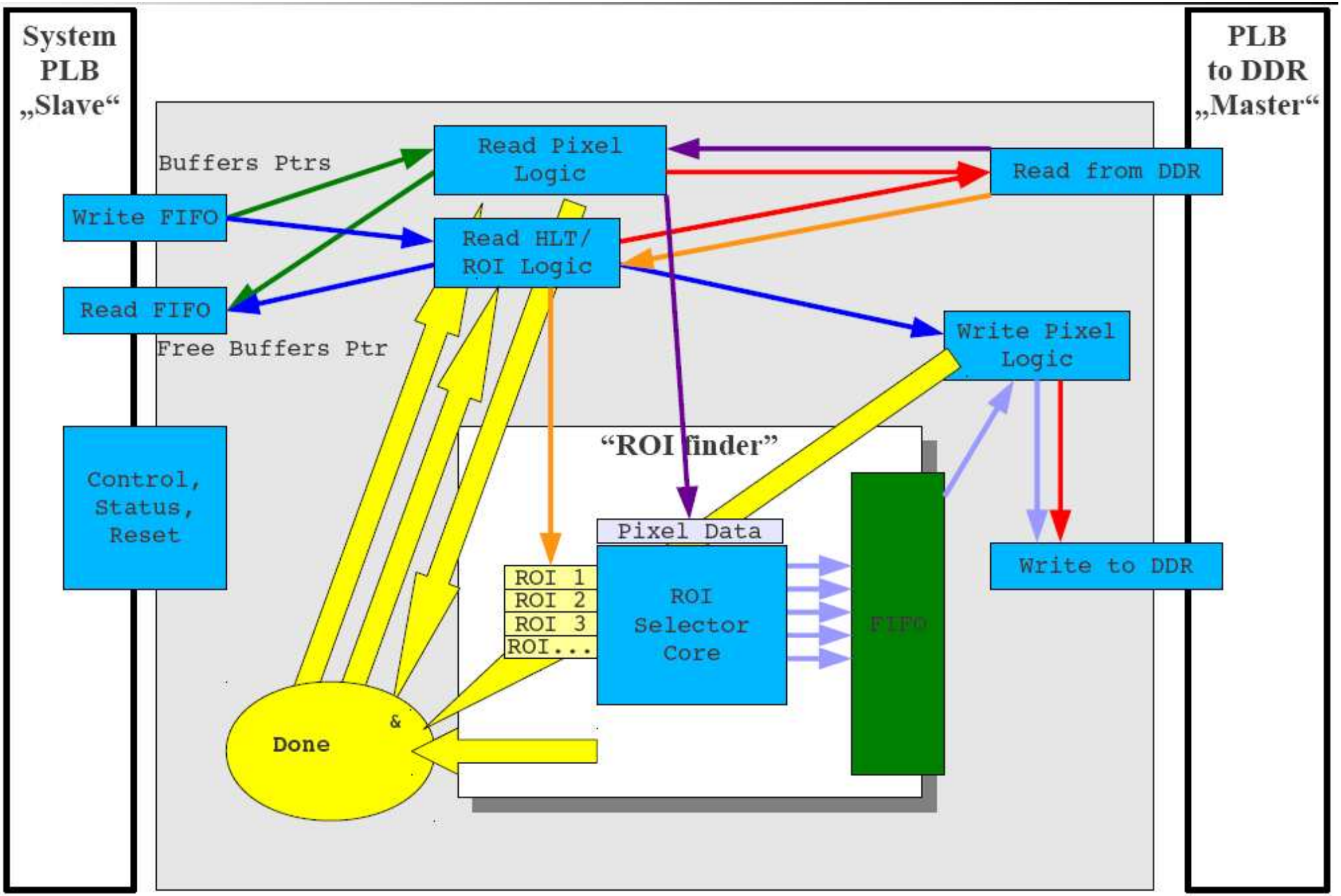
```
.. Nr of Buffers: 1250
.. Memsize: 100000
.. => Size of MemBlock: 80 [in kb]
Add: 329680 Pro: 329680 Rej: 670320 RejSize: 0 IDnf: 670320

.. Nr of Buffers: 1875
.. Memsize: 150000
.. => Size of MemBlock: 80
Add: 493895 Pro: 493895 Rej: 506105 RejSize: 0 IDnf: 506105

.. Nr of Buffers: 2500
.. Memsize: 200000
.. => Size of MemBlock: 80
Add: 658232 Pro: 658232 Rej: 341768 RejSize: 0 IDnf: 341768

.. Nr of Buffers: 3750
.. Memsize: 300000
.. => Size of MemBlock: 80
Add: 976560 Pro: 976560 Rej: 23440 RejSize: 0 IDnf: 23440

.. Nr of Buffers: 5000
.. Memsize: 400000
.. => Size of MemBlock: 80
Add: 1000000 Pro: 1000000 Rej: 0 RejSize: 0 IDnf: 0
```


System
PLB
„Slave“
PLB
to DDR
„Master“
Buffers Ptrs
Write FIFO
Read FIFO
Free Buffers Ptr
Read Pixel
Logic
Read HLT/
ROI Logic
Read from DDR
Write Pixel
Logic
Control,
Status,
Reset
“ROI finder”
Pixel Data
ROI 1
ROI 2
ROI 3
ROI...
ROI
Selector
Core
FIFO
Write to DDR
Done
&

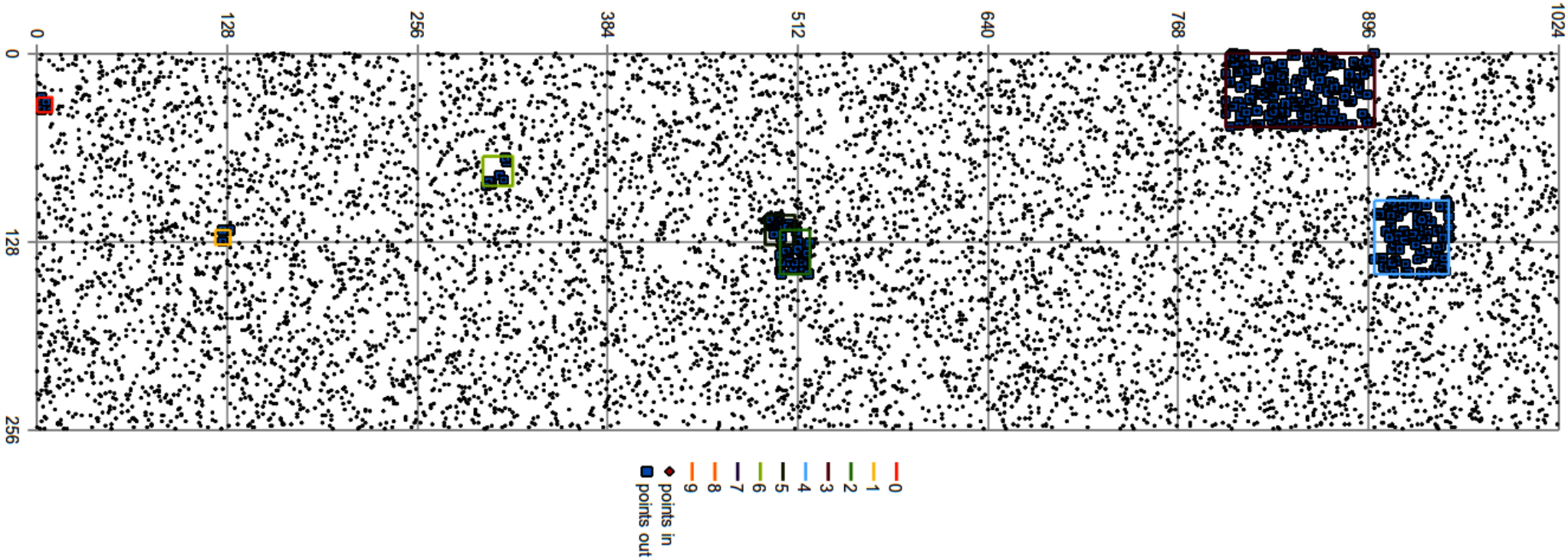
Algorithm on PXDDAQ #2: ROI selector (data filter)

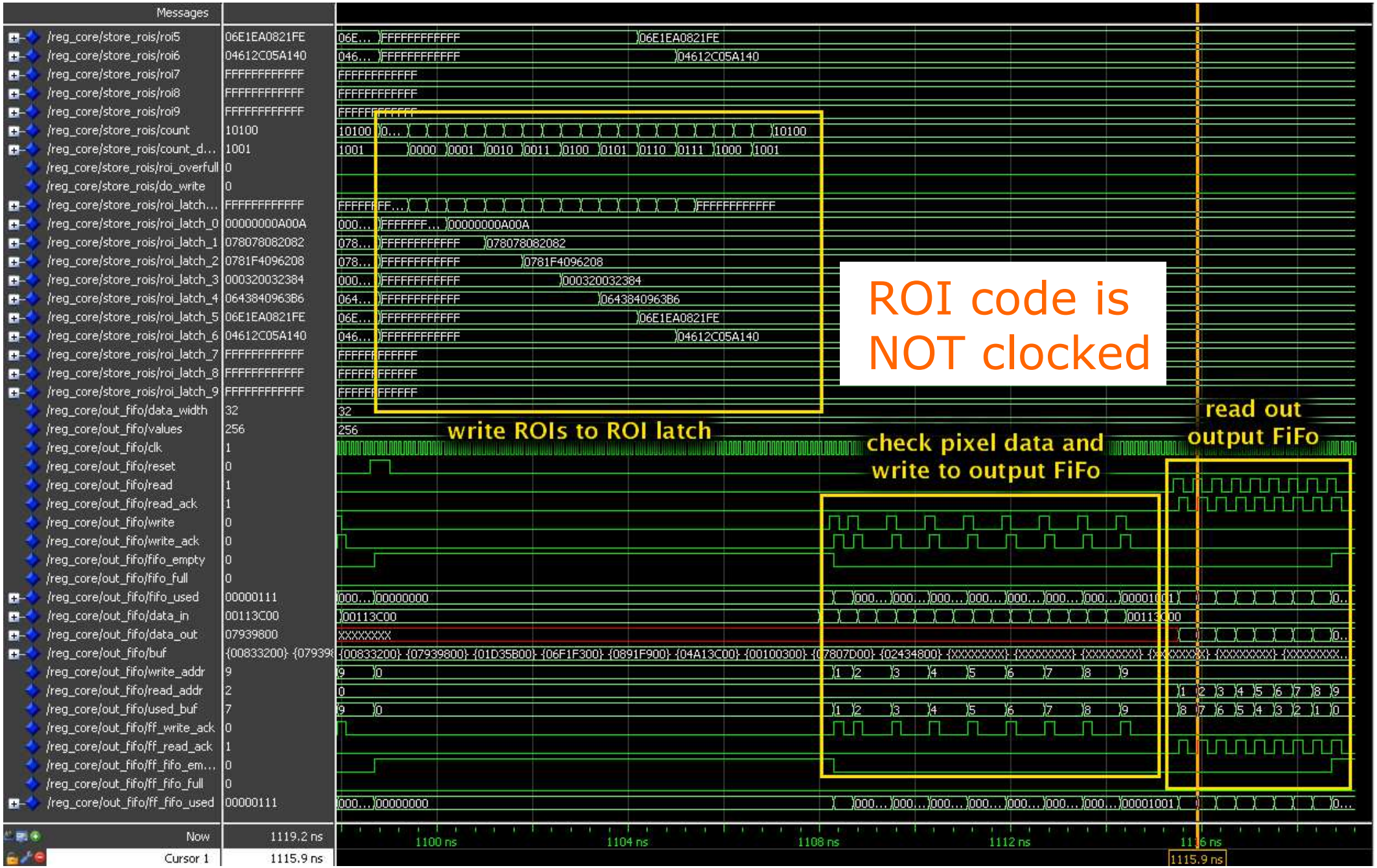
Event Example, 1 Half-Module

1024 x 256 space
(reality: 768 x 250, but address space needs to be bit aligned anyway, -> 10 bit x 8 bit)

3% occupancy

9 ROIs




Messages
/reg_core/store_rois/roi5
/reg_core/store_rois/roi6
/reg_core/store_rois/roi7
/reg_core/store_rois/roi8
/reg_core/store_rois/roi9
/reg_core/store_rois/count
/reg_core/store_rois/count_d...
/reg_core/store_rois/roi_overfull
/reg_core/store_rois/do_write
/reg_core/store_rois/roi_latch...
/reg_core/store_rois/roi_latch_0
/reg_core/store_rois/roi_latch_1
/reg_core/store_rois/roi_latch_2
/reg_core/store_rois/roi_latch_3
/reg_core/store_rois/roi_latch_4
/reg_core/store_rois/roi_latch_5
/reg_core/store_rois/roi_latch_6
/reg_core/store_rois/roi_latch_7
/reg_core/store_rois/roi_latch_8
/reg_core/store_rois/roi_latch_9
/reg_core/out_fifo/data_width
/reg_core/out_fifo/values
/reg_core/out_fifo/clk
/reg_core/out_fifo/reset
/reg_core/out_fifo/read
/reg_core/out_fifo/read_ack
/reg_core/out_fifo/write
/reg_core/out_fifo/write_ack
/reg_core/out_fifo/fifo_empty
/reg_core/out_fifo/fifo_full
/reg_core/out_fifo/fifo_used
/reg_core/out_fifo/data_in
/reg_core/out_fifo/data_out
/reg_core/out_fifo/buf
/reg_core/out_fifo/write_addr
/reg_core/out_fifo/read_addr
/reg_core/out_fifo/used_buf
/reg_core/out_fifo/ff_write_ack
/reg_core/out_fifo/ff_read_ack
/reg_core/out_fifo/ff_fifo_em...
/reg_core/out_fifo/ff_fifo_full
/reg_core/out_fifo/ff_fifo_used
Now
1119.2 ns
Cursor 1
1115.9 ns
write ROIs to ROI latch
check pixel data and
write to output FiFo
read out
output FiFo
ROI code is
NOT clocked
1100 ns
1104 ns
1108 ns
1112 ns
1116 ns
1115.9 ns

Stability Test

```
count events:   1758620 correct        0 with error  (rate 0.000000  ROIs(8):  ) 195010 195780 194950 195468 194624 195459 196079 195484 195766
count events:   1758630 correct        0 with error  (rate 0.000000  ROIs(4):  ) 195012 195781 194952 195469 194625 195459 196080 195485 195767
count events:   1758640 correct        0 with error  (rate 0.000000  ROIs(6):  ) 195014 195781 194953 195471 194626 195461 196080 195485 195769
count events:   1758650 correct        0 with error  (rate 0.000000  ROIs(1):  ) 195016 195784 194953 195472 194626 195463 196081 195486 195769
count events:   1758660 correct        0 with error  (rate 0.000000  ROIs(6):  ) 195017 195785 194954 195474 194626 195466 196082 195486 195770
count events:   1758670 correct        0 with error  (rate 0.000000  ROIs(6):  ) 195017 195787 194955 195475 194629 195469 196082 195486 195770
count events:   1758680 correct        0 with error  (rate 0.000000  ROIs(1):  ) 195018 195788 194956 195476 194629 195470 196085 195488 195770
count events:   1758690 correct        0 with error  (rate 0.000000  ROIs(3):  ) 195019 195789 194957 195478 194629 195471 196085 195490 195772
count events:   1758700 correct        0 with error  (rate 0.000000  ROIs(4):  ) 195020 195792 194957 195479 194630 195471 196085 195492 195774
count events:   1758710 correct        0 with error  (rate 0.000000  ROIs(6):  ) 195020 195793 194957 195480 194632 195473 196086 195494 195775
count events:   1758720 correct        0 with error  (rate 0.000000  ROIs(1):  ) 195022 195795 194960 195482 194632 195473 196086 195494 195776
count events:   1758730 correct        0 with error  (rate 0.000000  ROIs(7):  ) 195023 195797 194960 195482 194635 195474 196088 195495 195776
count events:   1758740 correct        0 with error  (rate 0.000000  ROIs(5):  ) 195026 195799 194960 195482 194636 195475 196088 195498 195776
count events:   1758750 correct        0 with error  (rate 0.000000  ROIs(8):  ) 195026 195799 194960 195487 194636 195477 196089 195499 195777
count events:   1758760 correct        0 with error  (rate 0.000000  ROIs(7):  ) 195027 195800 194960 195489 194637 195477 196090 195500 195780
count events:   1758770 correct        0 with error  (rate 0.000000  ROIs(8):  ) 195028 195800 194961 195489 194639 195479 196091 195503 195780
count events:   1758780 correct        0 with error  (rate 0.000000  ROIs(7):  ) 195029 195801 194962 195491 194640 195480 196093 195504 195780
count events:   1758790 correct        0 with error  (rate 0.000000  ROIs(5):  ) 195030 195802 194964 195492 194643 195481 196093 195505 195780
count events:   1758800 correct        0 with error  (rate 0.000000  ROIs(9):  ) 195031 195803 194965 195493 194645 195482 196093 195505 195783
count events:   1758810 correct        0 with error  (rate 0.000000  ROIs(7):  ) 195032 195806 194965 195493 194645 195483 196096 195506 195784
count events:   1758820 correct        0 with error  (rate 0.000000  ROIs(7):  ) 195034 195807 194965 195494 194645 195485 196097 195508 195785
count events:   1758830 correct        0 with error  (rate 0.000000  ROIs(6):  ) 195035 195807 194967 195496 194646 195487 196097 195509 195786
count events:   1758840 correct        0 with error  (rate 0.000000  ROIs(6):  ) 195037 195809 194968 195496 194647 195490 196097 195510 195786
count events:   1758850 correct        0 with error  (rate 0.000000  ROIs(4):  ) 195037 195810 194968 195499 194648 195490 196099 195512 195787
count events:   1758860 correct        0 with error  (rate 0.000000  ROIs(8):  ) 195037 195811 194970 195500 194649 195490 196101 195513 195789
count events:   1758870 correct        0 with error  (rate 0.000000  ROIs(9):  ) 195038 195811 194971 195501 194650 195492 196101 195514 195792
count events:   1758880 correct        0 with error  (rate 0.000000  ROIs(4):  ) 195040 195811 194971 195502 194650 195494 196104 195515 195793
count events:   1758890 correct        0 with error  (rate 0.000000  ROIs(8):  ) 195041 195811 194971 195502 194651 195496 196105 195518 195795
count events:   1758900 correct        0 with error  (rate 0.000000  ROIs(5):  ) 195042 195813 194972 195503 194653 195496 196107 195519 195795
count events:   1758910 correct        0 with error  (rate 0.000000  ROIs(7):  ) 195042 195816 194972 195503 194653 195497 196109 195520 195798
```

<=10 random ROIs, variable number of ROI, variable ROI size
random data, 3% occupancy
ML403 board: 3 days, >11 x 10^6 events
ATCA CN: >4.5 x 10^6 events
output data re-checked on PowerPC side

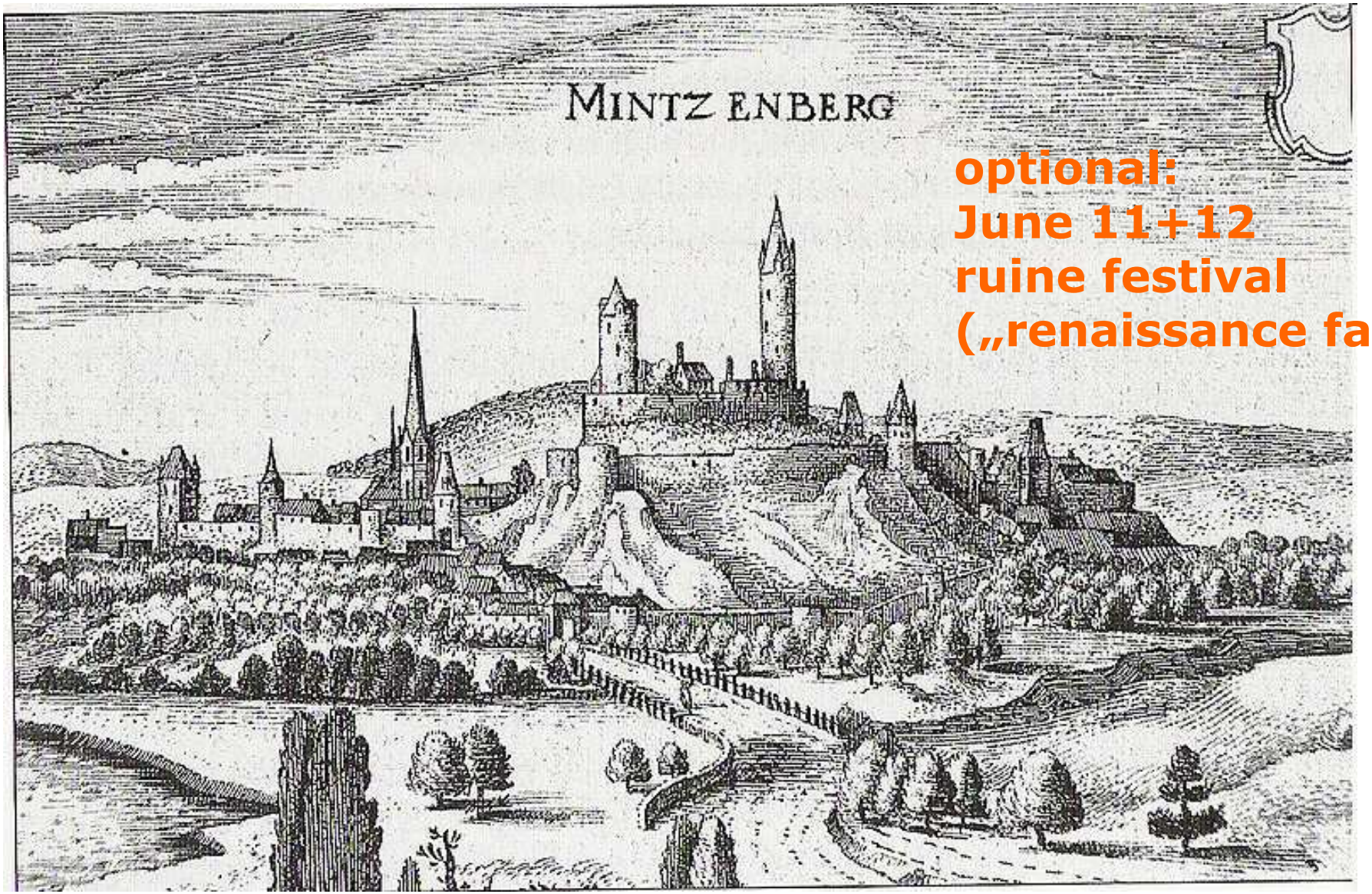
The location for the PXD DAQ Workshop
June 9 and 10, 2011
was fixed

Arrangement will be signed in the next few days.

http://de.wikipedia.org/wiki/Burg_Münzenberg
http://www.burghotelmuenzenberg.de/

arrangement incl.
- dinner outside in the „Biergarten"
- guided tour to the ruine

Mentioned in Documents since 1162.
This drawing by Merian dated 1620.



optional:
June 11+12
ruine festival
(„renaissance fair")

Münzenberg


Wettenberg
480
E40
Mücke
49
Gießen
Buseck
Grünberg
Ulrichstein
Linden
Fernwald
Laubach
Herbs
Pohlheim
Hungen
Schotten
Butzbach
489
455
45
Nidda
Gedern
E41
275
Bad
Nauheim
Ortenberg
5
Friedberg
Büdingen
E451
45
Nidderau
Wächtersbach
456
Friedrichsdorf
Gründau
Bad Homburg
vor der Höhe
Bruchköbel
Gelnhausen
40
Maintal
Frankfurt
am Main
Freigericht
Hanau
43
Alzenau
Obertshausen
3
Neu-Isenburg
Seligenstadt
Mömbris