Tiny Triplet Finder

A Pattern Recognition Scheme for

Large Curvature Circular Tracks

and Its FPGA Implementation Using Hash Sorter

Jinyuan Wu

Fermilab

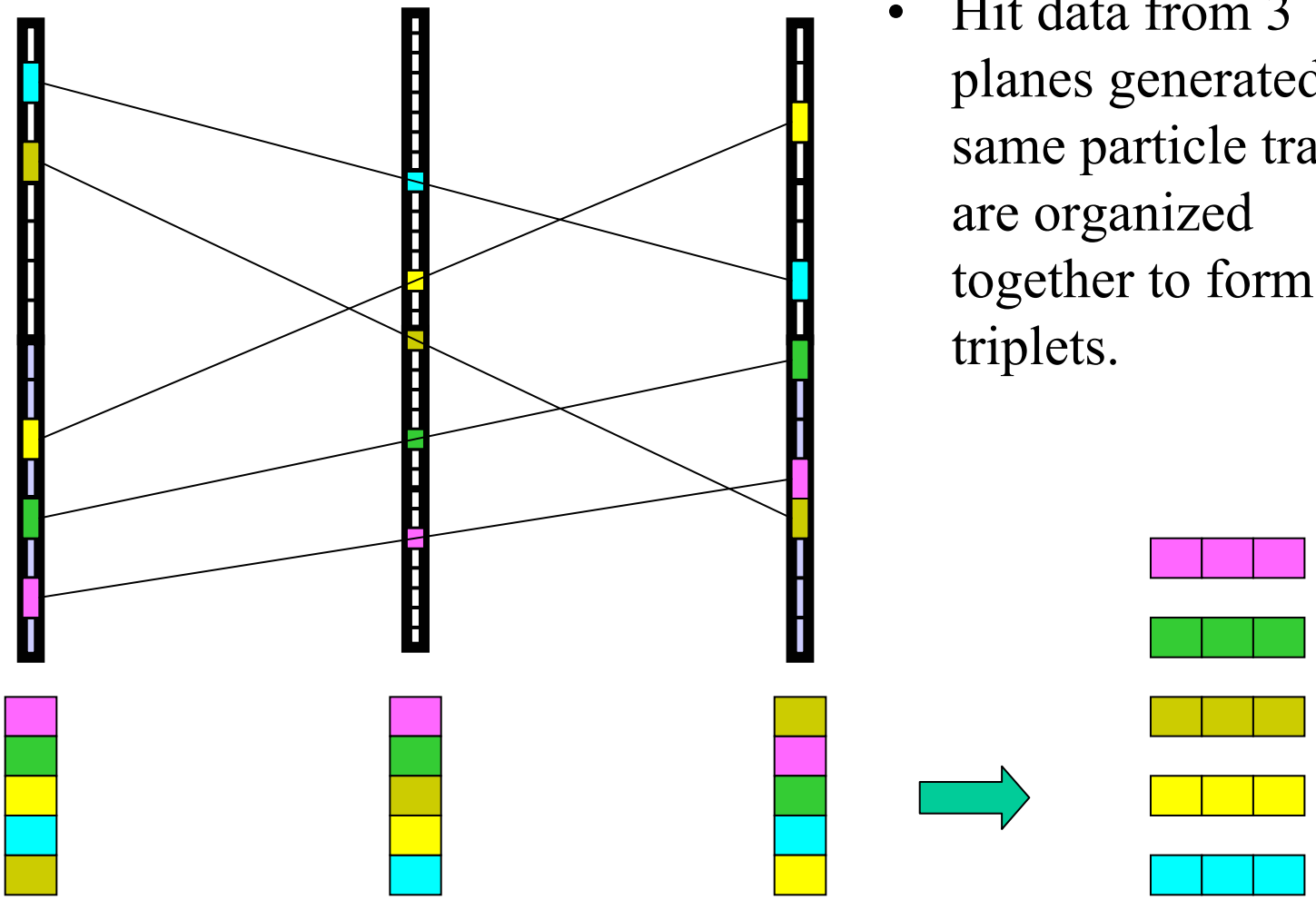
Sept. 2004

Tiny Triplet Finder

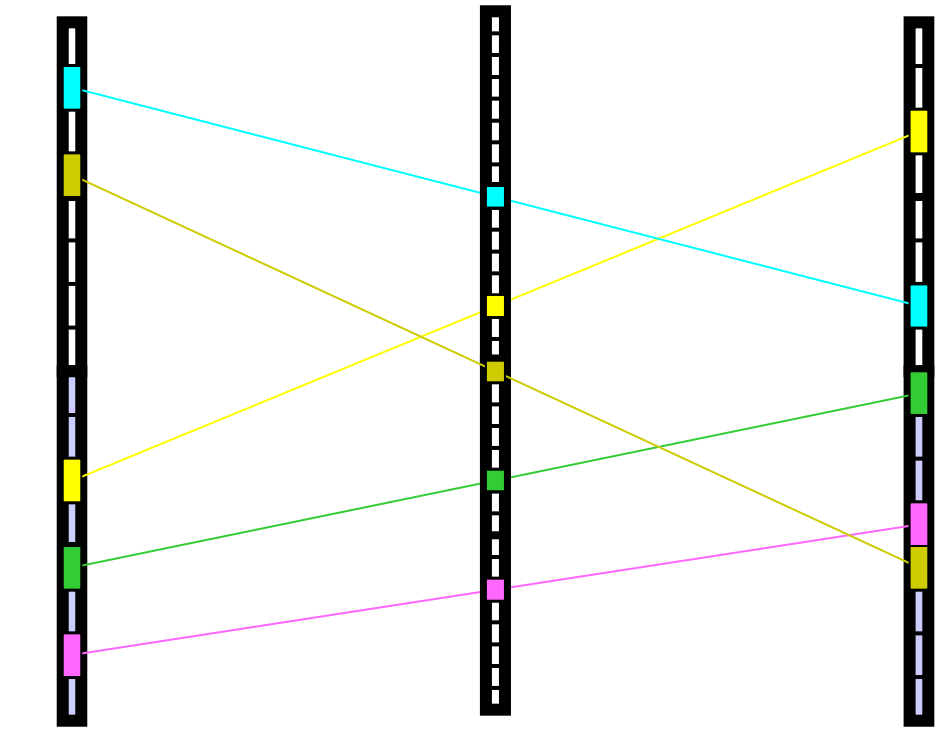
Jinyuan Wu,
Z. Shi, M. Wang, H. Garcia and E. Gottschalk
For BTeV Collaboration
Fermi National Accelerator Laboratory
Sept. 2004

Hits, Hit Data & Triplets (in BTeV)

- Hit data come out of the detector planes in random order.
- Hit data from 3 planes generated by same particle tracks are organized together to form triplets.

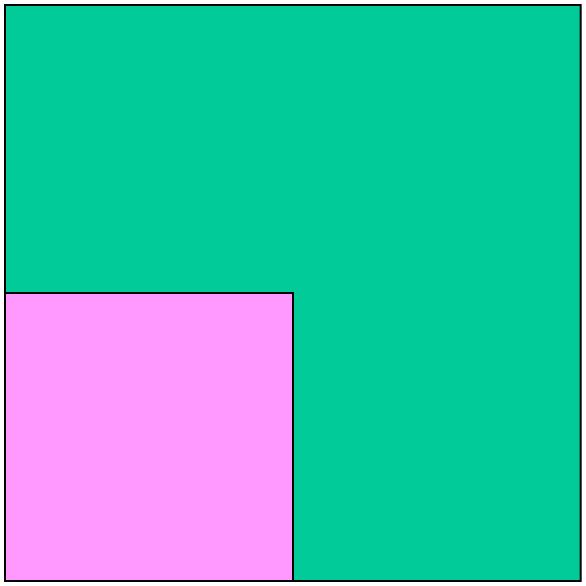


Triplet Finding



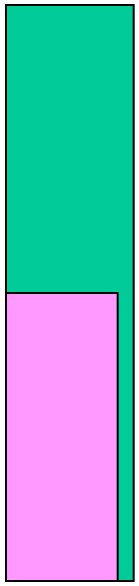
- Three data items must satisfy the condition: $x_A + x_C = 2\ x_B$.
- A total of n^3 combinations must be checked (e.g. 5x5x5=125).
- Three layers of loops if the process is implemented in software.
- Large silicon resource may be needed without careful planning: $O(N^2)$

Tiny? Yes, Tiny! – Logic Cell Usage:



CAM, Hough Transform

etc., $O(N^2)$

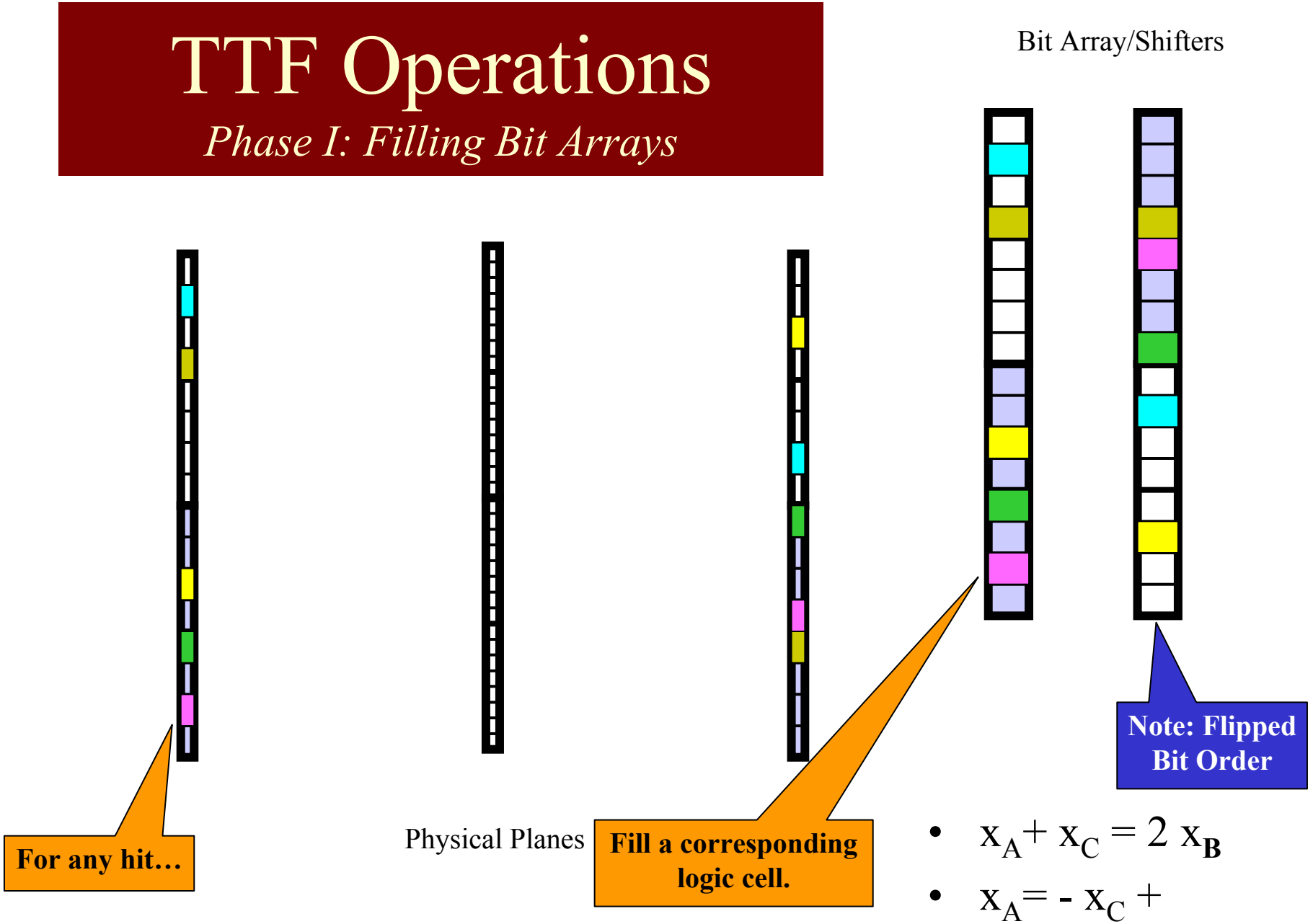


Tiny Triplet Finder

O(N*logN)

TTF Operations

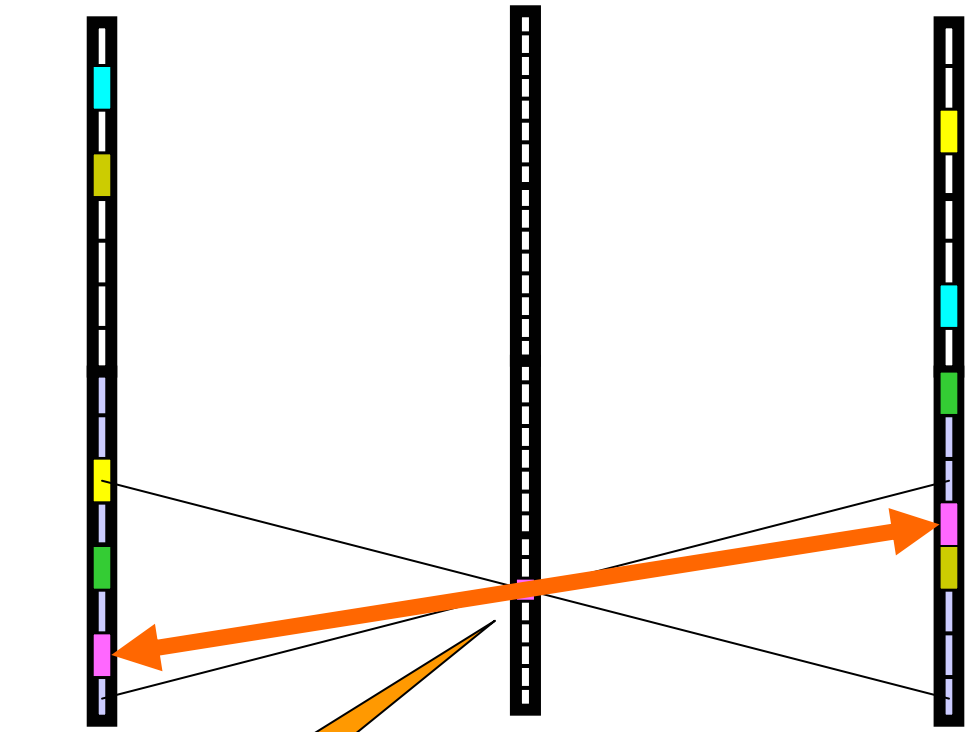
Phase I: Filling Bit Arrays



- $x_A + x_C = 2\ x_B$
- $x_A = -\ x_C +$ constant

TTF Operations

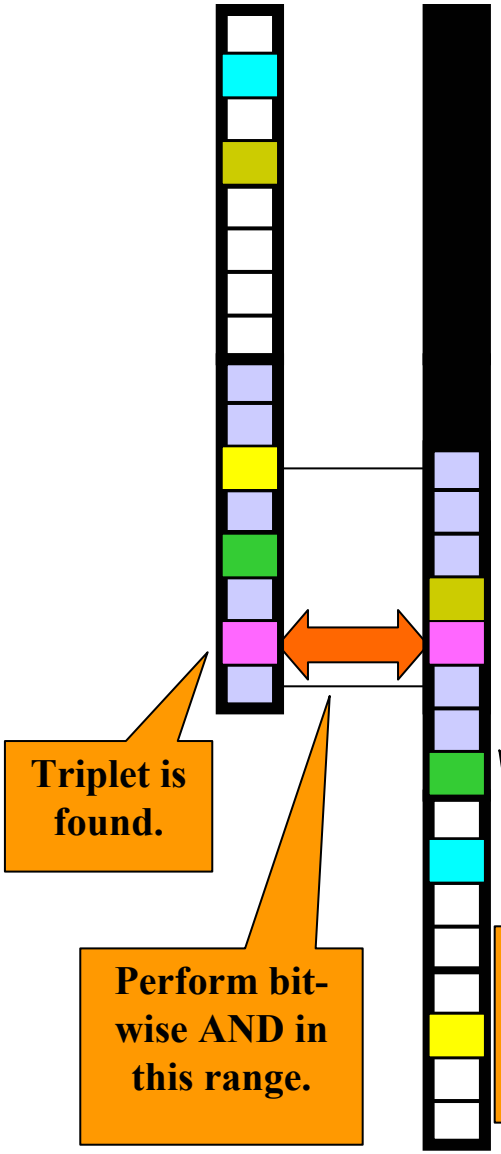
Phase II: Making Match



Physical Planes

For any center plane hit…

Bit Array/Shifters



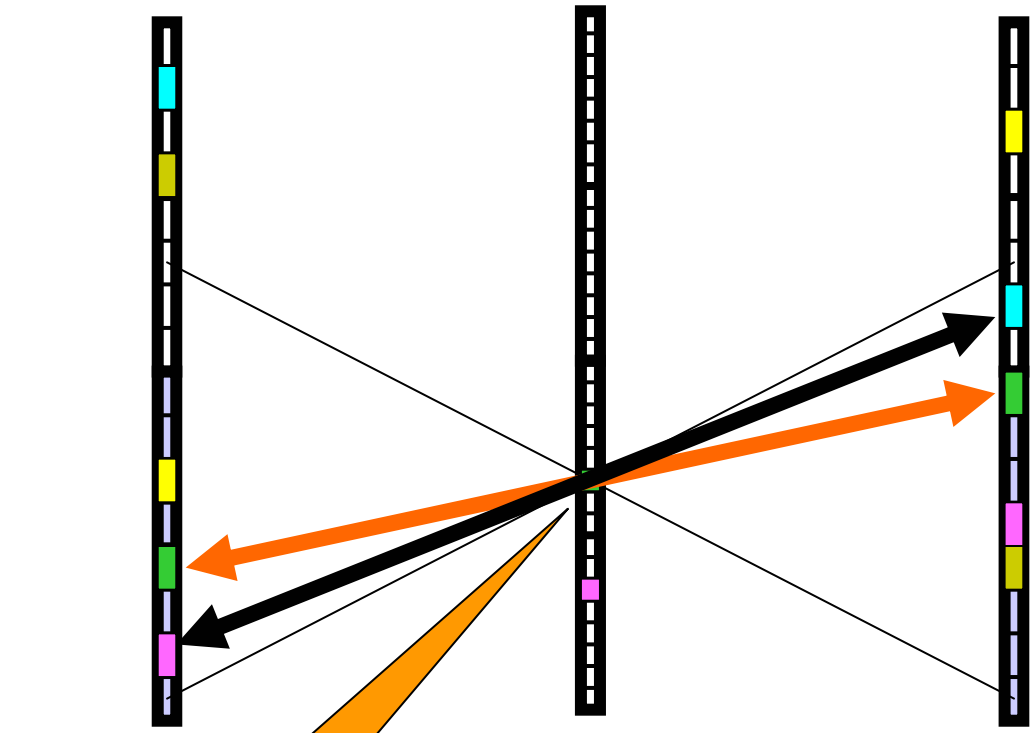
Triplet is found.

Perform bit-wise AND in this range.

Logically shift the bit array.

TTF Operations

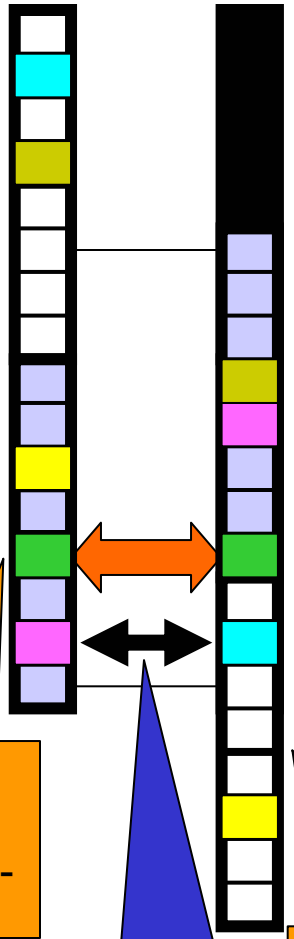
Phase II: Making Match (Next Hit)



Physical Planes

Loop to next center plane hit…

Bit Array/Shifters



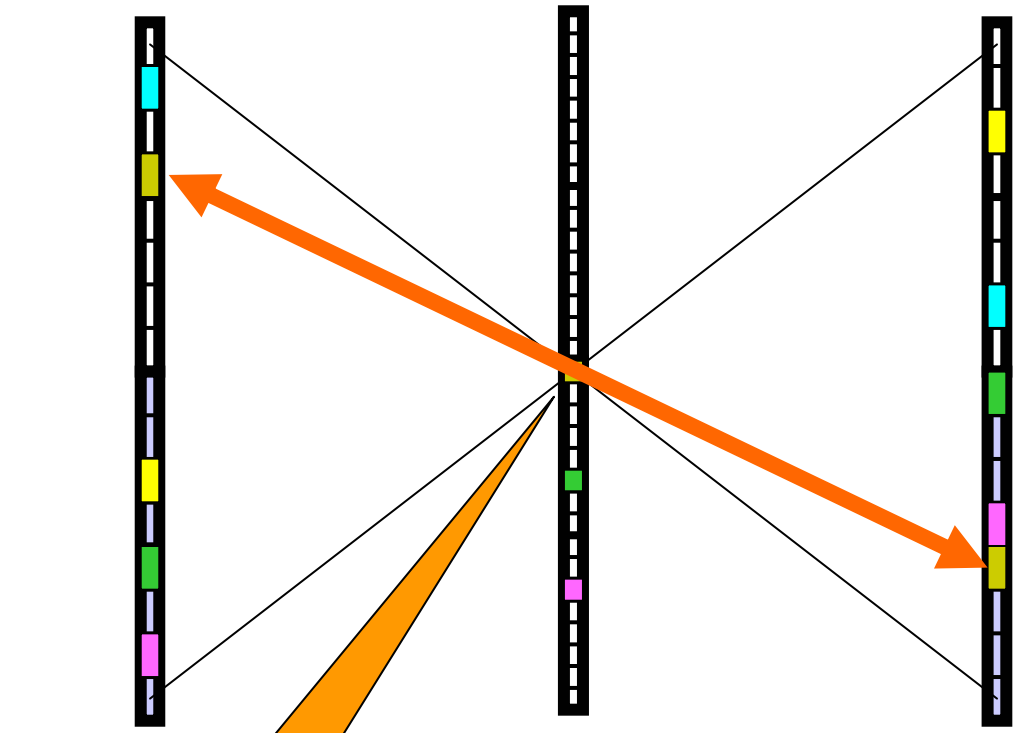
Triplet is found through bit-wise AND.

Fake triplet may exist. It will be cut out in the later stages.

Logically shift the bit array.

TTF Operations

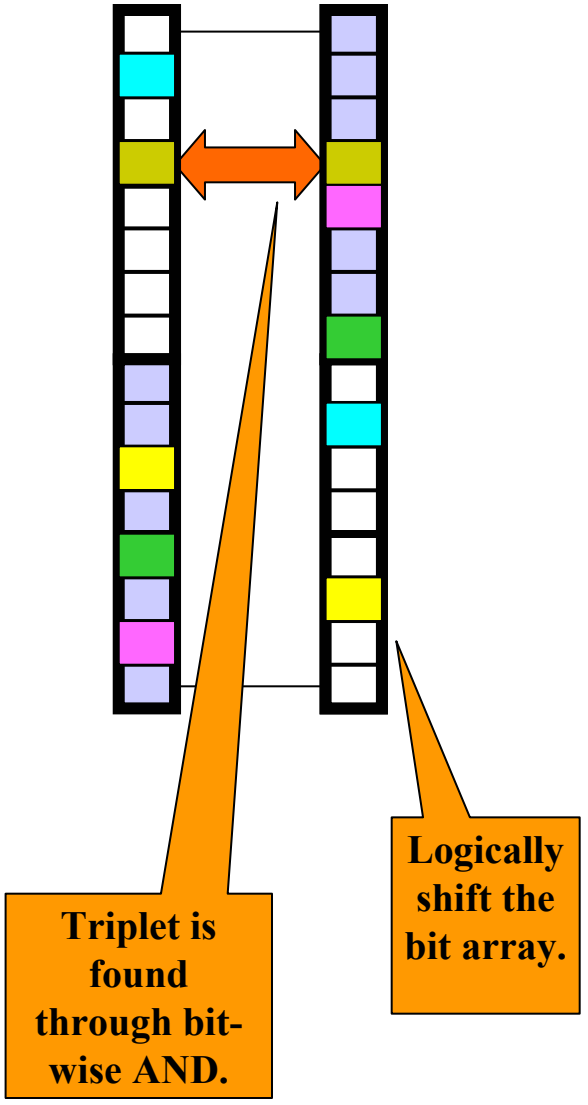
Phase II: Making Match (More…)



Physical Planes

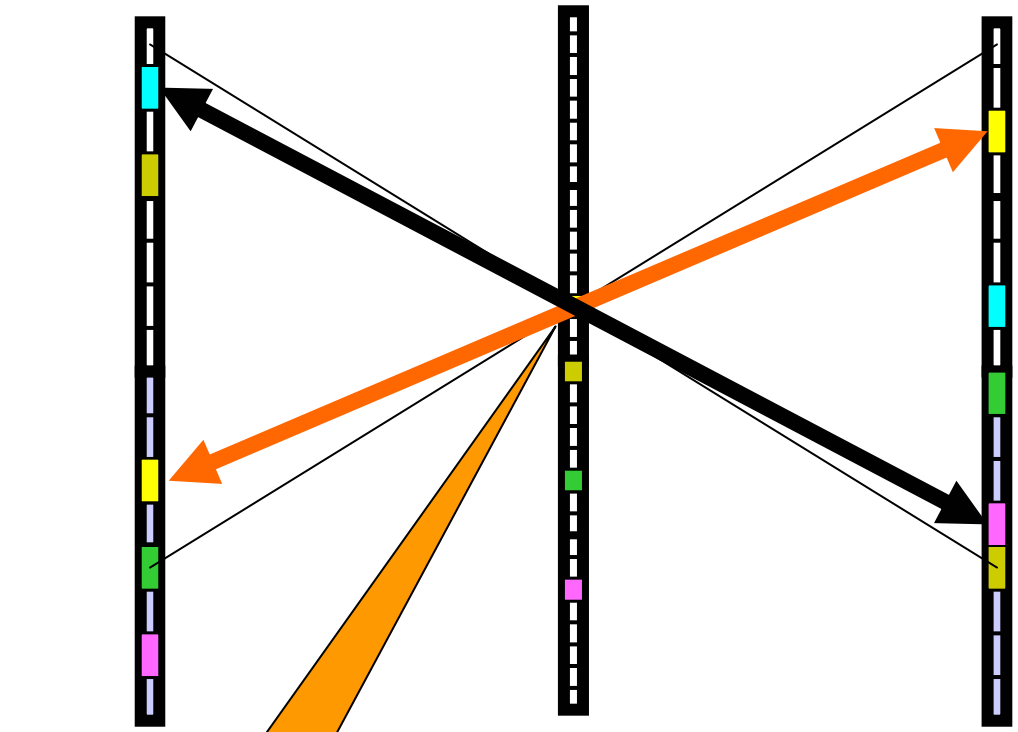
Loop to next center plane hit…

Bit Array/Shifters



TTF Operations

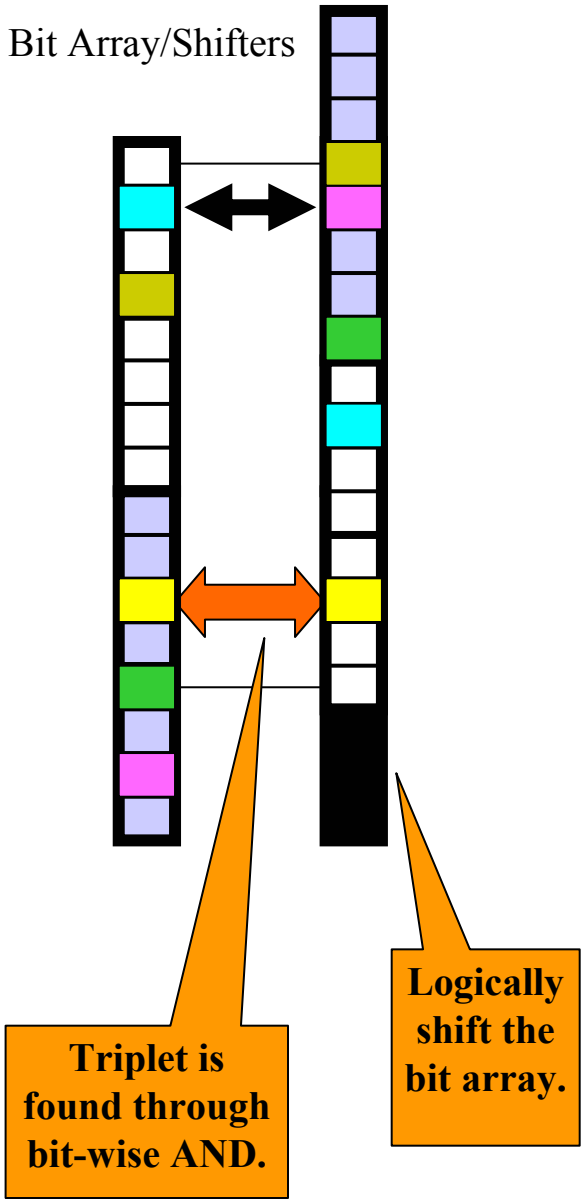
Phase II: Making Match (and More…)



Physical Planes

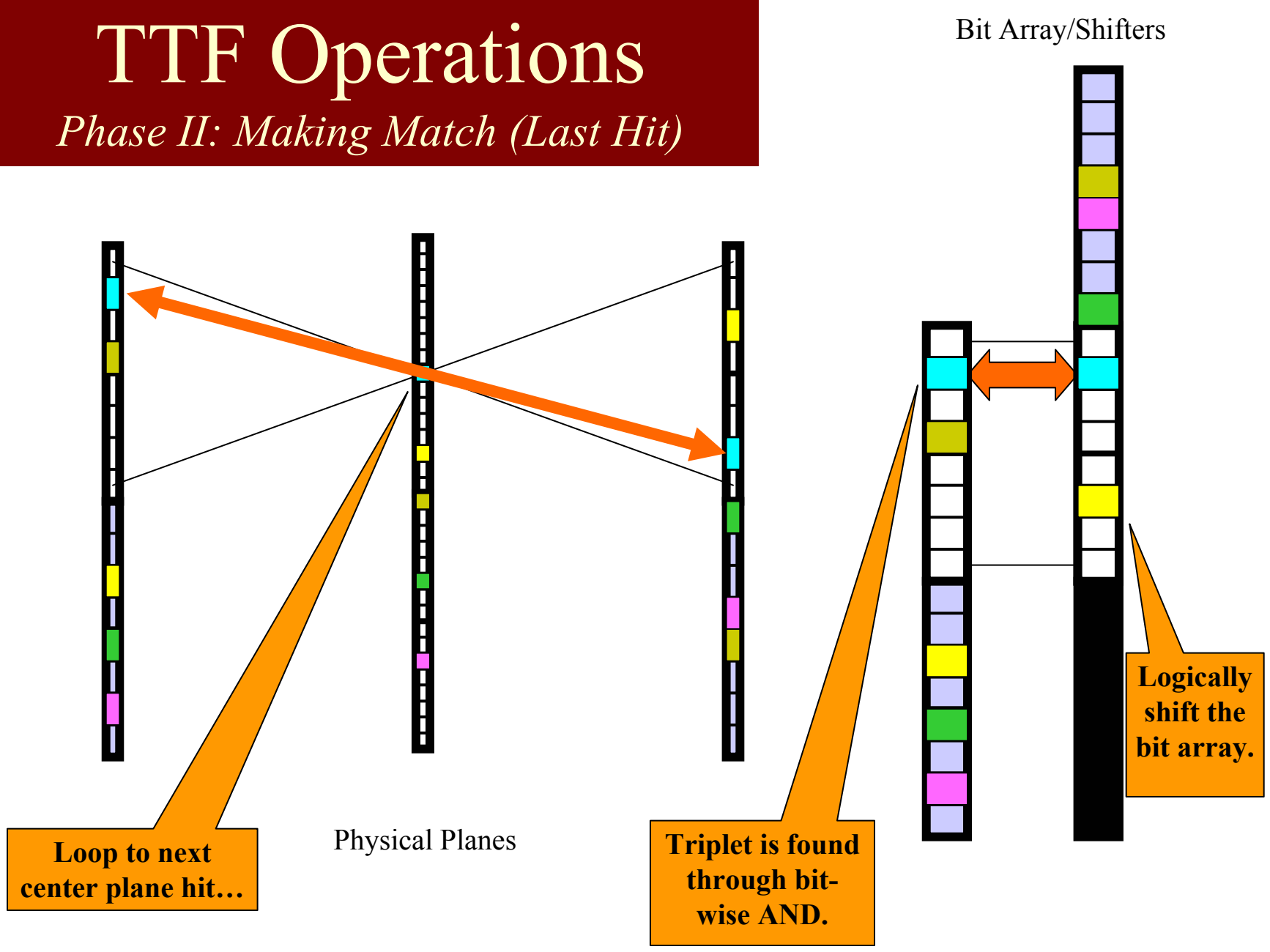
Loop to next center plane hit…

Bit Array/Shifters



TTF Operations

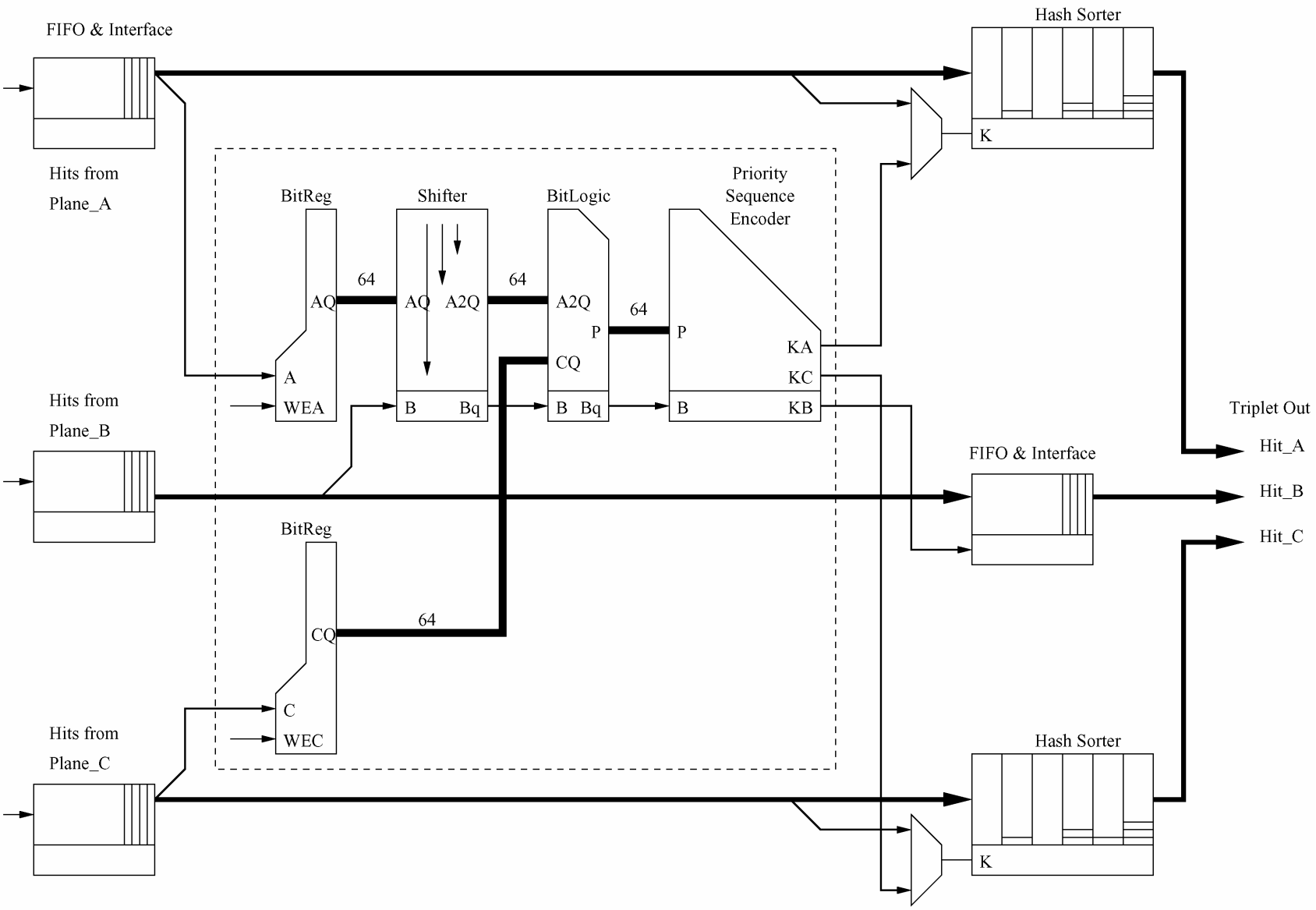
Phase II: Making Match (Last Hit)



Tiny Triplet Finder Operations

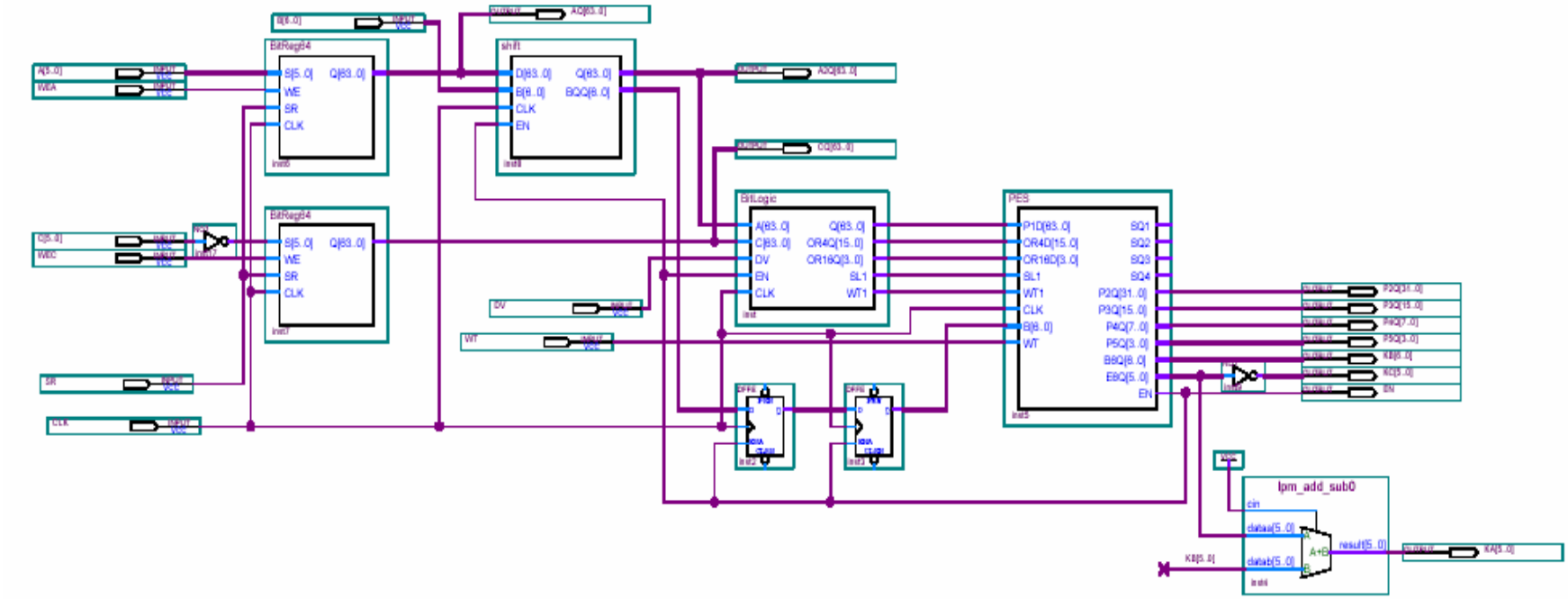
- Step 1:
  - Fill the bit arrays.
- Step 2:
  - Shift the bit array and check for bit-wise coincident.
- Step 3:
  - There is no step 3.

Block Diagram

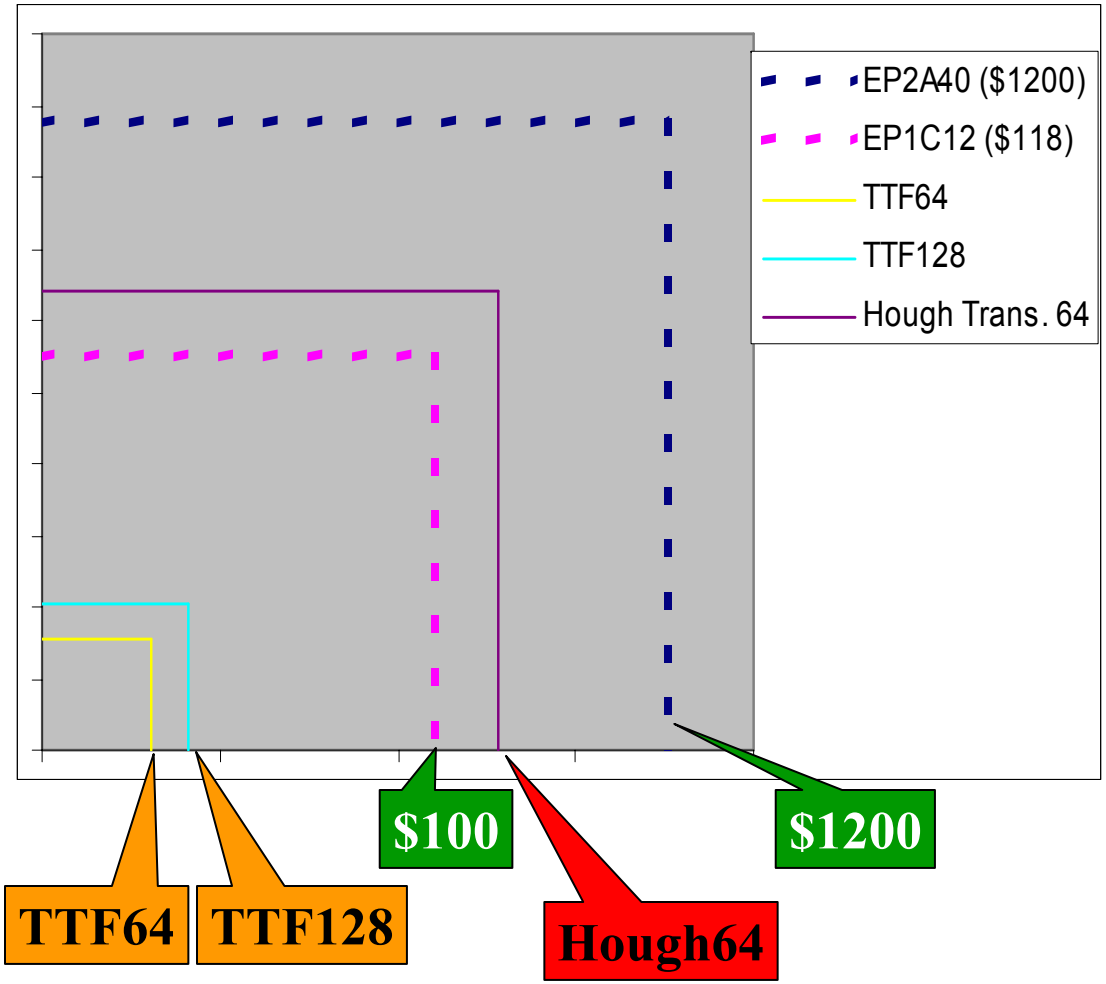


Schematics

Date: June 2, 2004 ttf.bdf Project: ttf

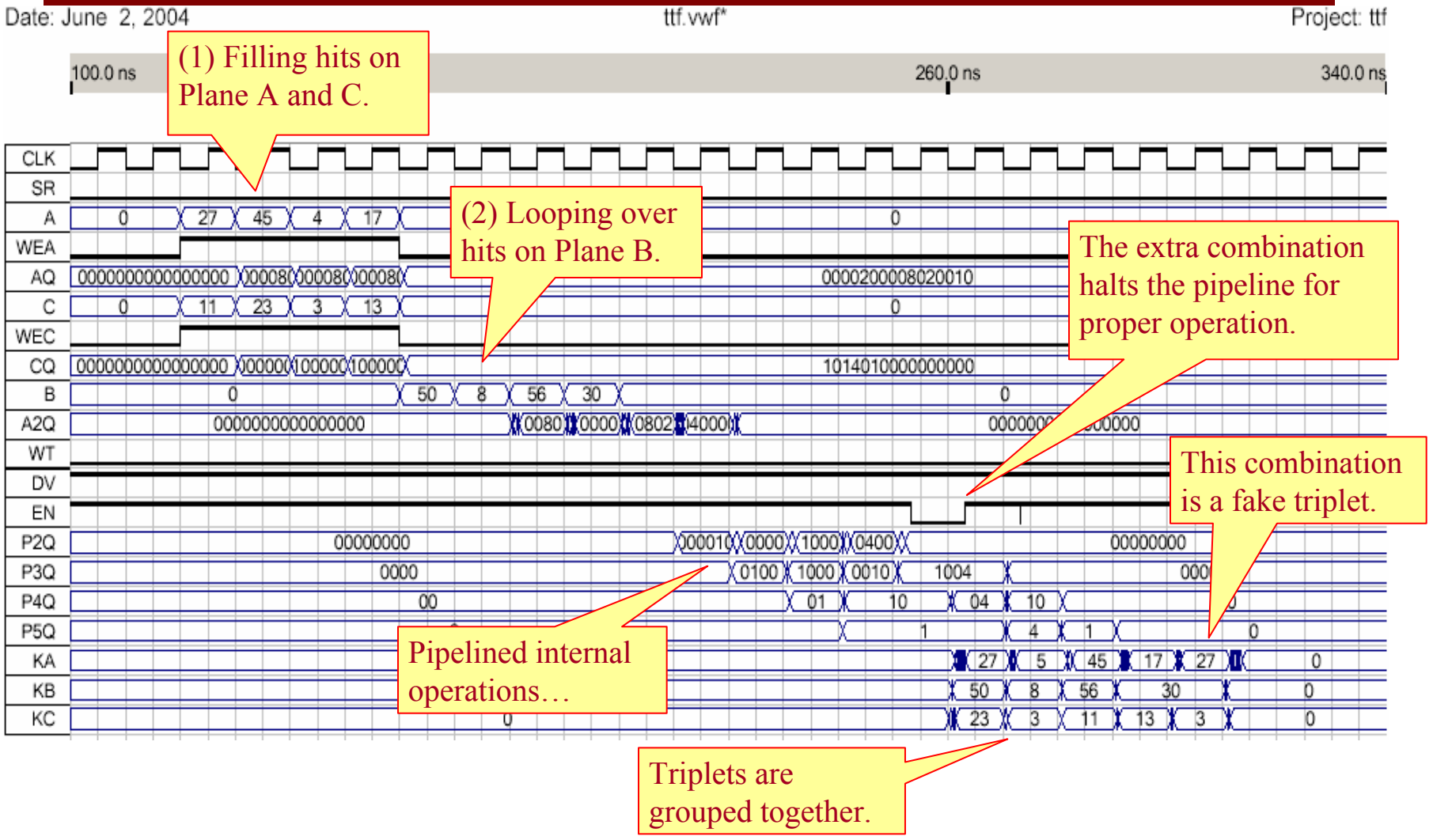


Logic Cell Usage



- Both 64- and 128-bit TTF designs fit $100 FPGA comfortably.
- A simple 64-bit Hough transform design is shown for scale.
- A $1200 FPGA is shown for scale.

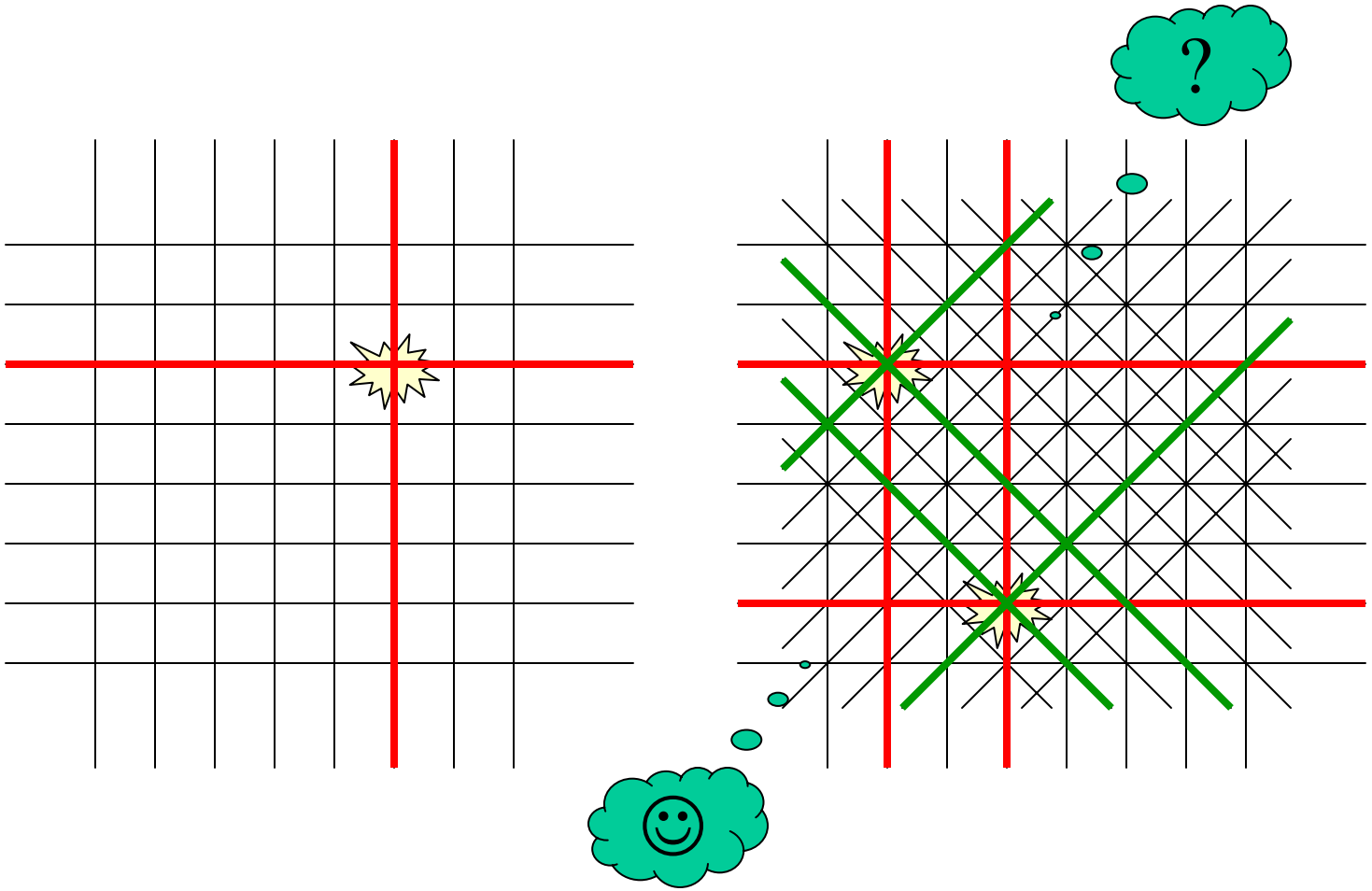
Simulation


Date: June 2, 2004
ttf.vwf*
Project: ttf
100.0 ns
260.0 ns
340.0 ns
(1) Filling hits on Plane A and C.
(2) Looping over hits on Plane B.
The extra combination halts the pipeline for proper operation.
This combination is a fake triplet.
Pipelined internal operations…
Triplets are grouped together.
CLK
SR
A
WEA
AQ
C
WEC
CQ
B
A2Q
WT
DV
EN
P2Q
P3Q
P4Q
P5Q
KA
KB
KC

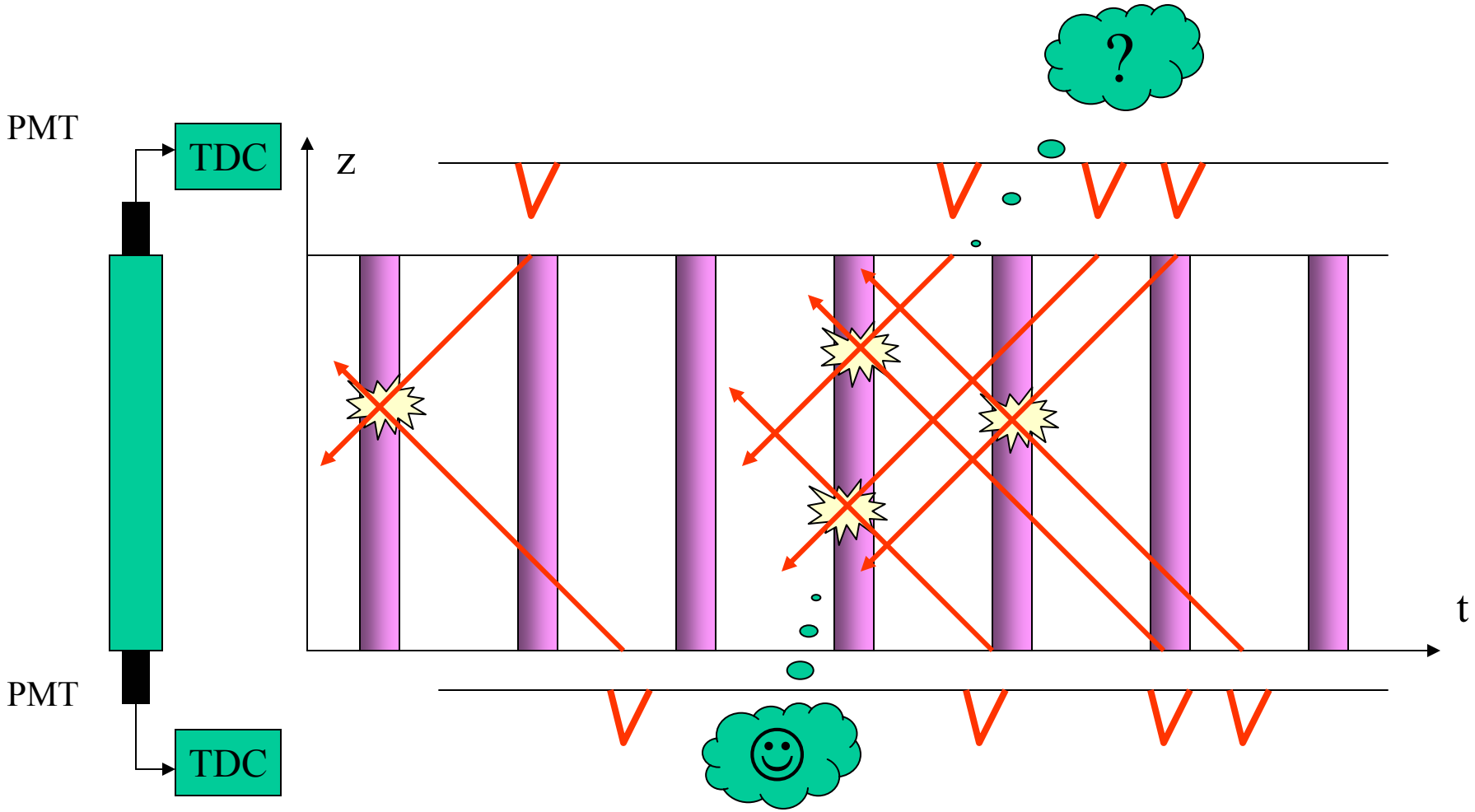
Other Applications of TTF

- There are other applications using same algorithm as TTF.
- Examples:
  - Wire chambers.
  - Time of flight counters.
  - GEM/MICROMEGAS

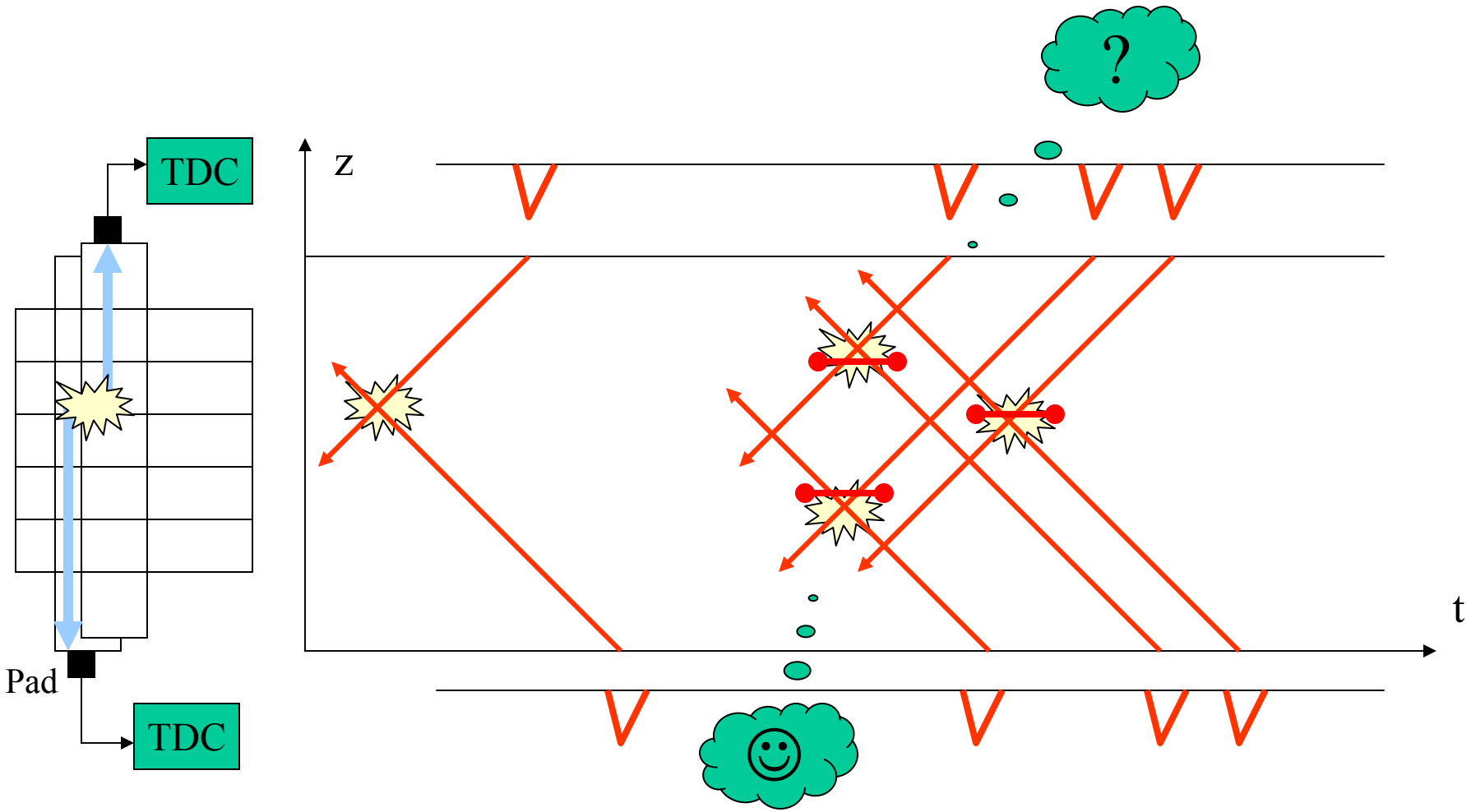
Other Applications: Wire Chambers



Other Applications: TOF



Other Applications: GEM/MICROMEGAS



Why Doing Something Tiny? Instead of Waiting for Moore's Law?

- Today, we hear Moore's law more often than Maxwell Equations.
- But in FPGA world, kilo- logic cells are still not so cheap – A quick argument.
- Consider FFT ($O(n^2)$->$O(n*\log n)$), 40 years ago:
  - If it were not developed, it would not be developed today.
  - Fortunately it was developed and we still use it today even though it is not necessary in many places.
  - We fully respect our grandparents for developing FFT.
- Now Tiny Triplet Finder ($O(N^2)$->$O(N*\log N)$):
  - If we do not developed it, it will not be developed in the future.
  - This piece of human knowledge will not ever exist.
  - Our grandchildren may say we are retarded – What's reputation of scientists of our generation?

A Pattern Recognition Scheme for Large Curvature Circular Tracks and Its FPGA Implementation Using Hash Sorter

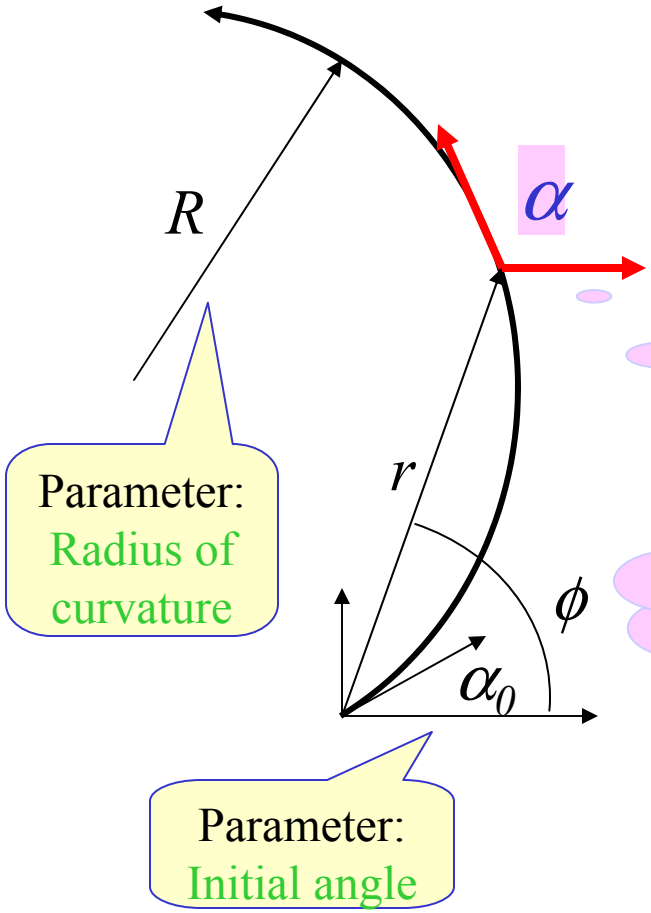
Jinyuan Wu and Z. Shi

Fermi National Accelerator Laboratory,

Batavia, IL 60510, USA

A Large Curvature Track

- A soft track hits large ϕ region.
  - A global algorithm is better suited.
- The "high-p_T" approximation is not valid globally.
  - Exact track equation is needed.



$$r = 2R\sin(\phi - \alpha_0)$$

Measure the tangent angle..

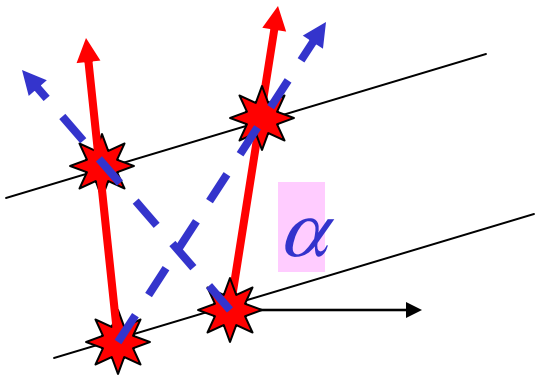
~~$r = 2R(\phi - \alpha_0)$~~

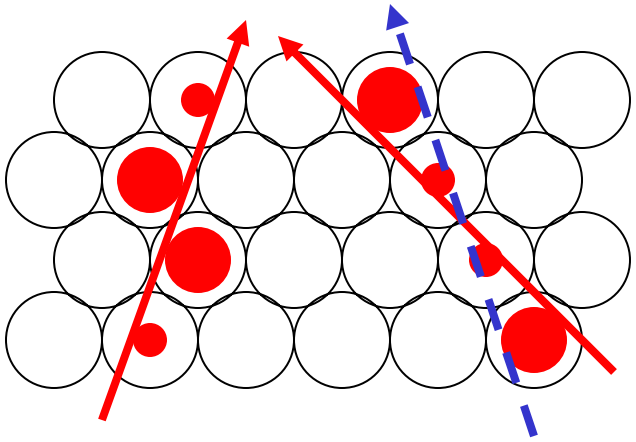
$$\alpha_0 = 2\phi - \alpha$$

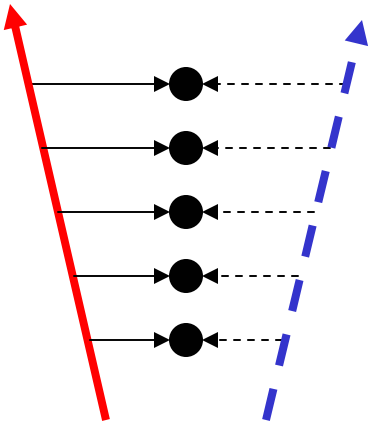
$$\frac{r}{2R} = \sin(\alpha - \phi)$$

Tangent Angle Measurements

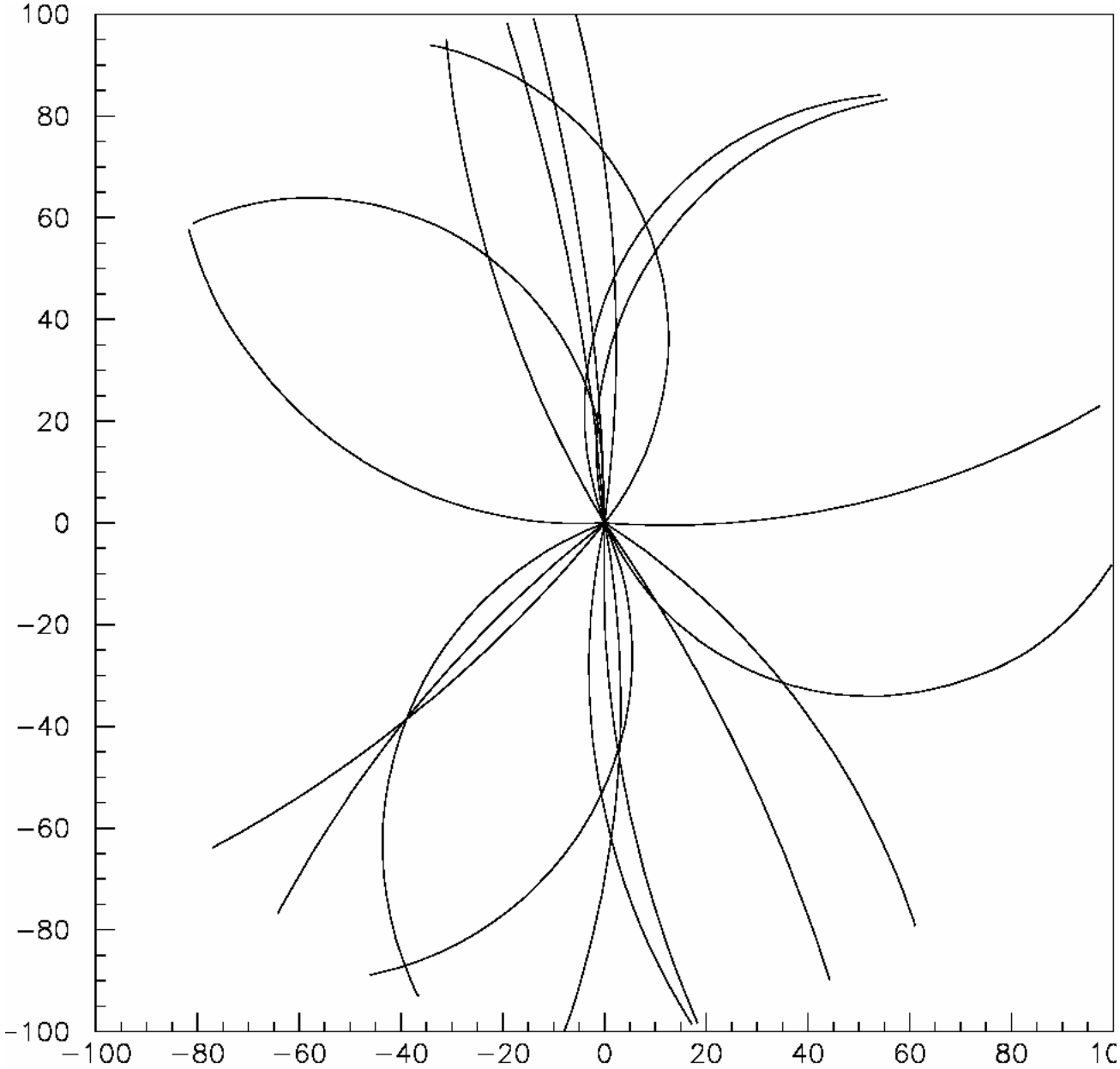
- There are various techniques to measure the tangent angle of the track segment (or “doublet”, or “cluster”).
- Sometimes extra “ghost” segments may exist.
- The ghost segments may be resolved in track recognition process later.





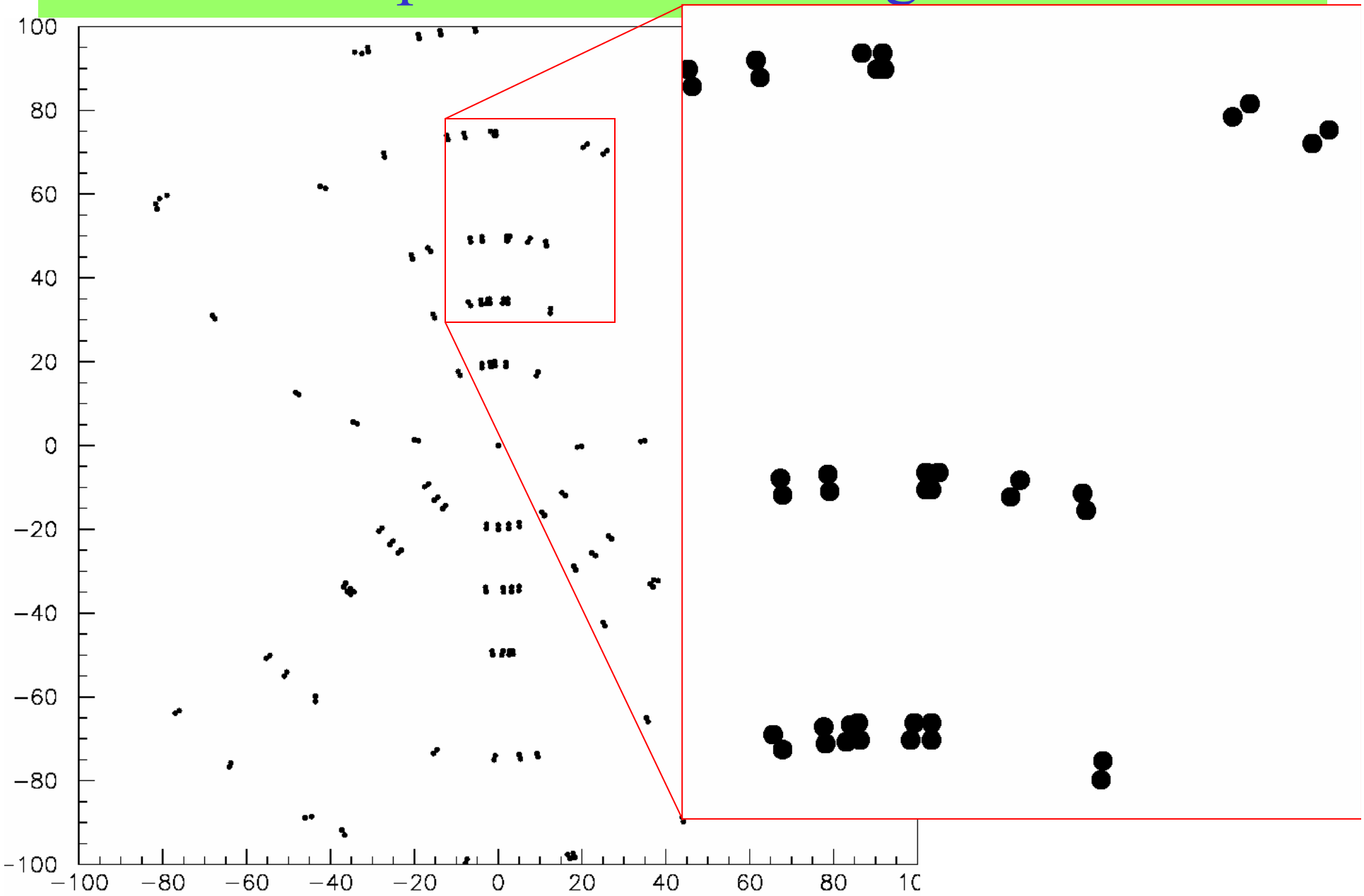


An Example of Track Recognition: *Event*

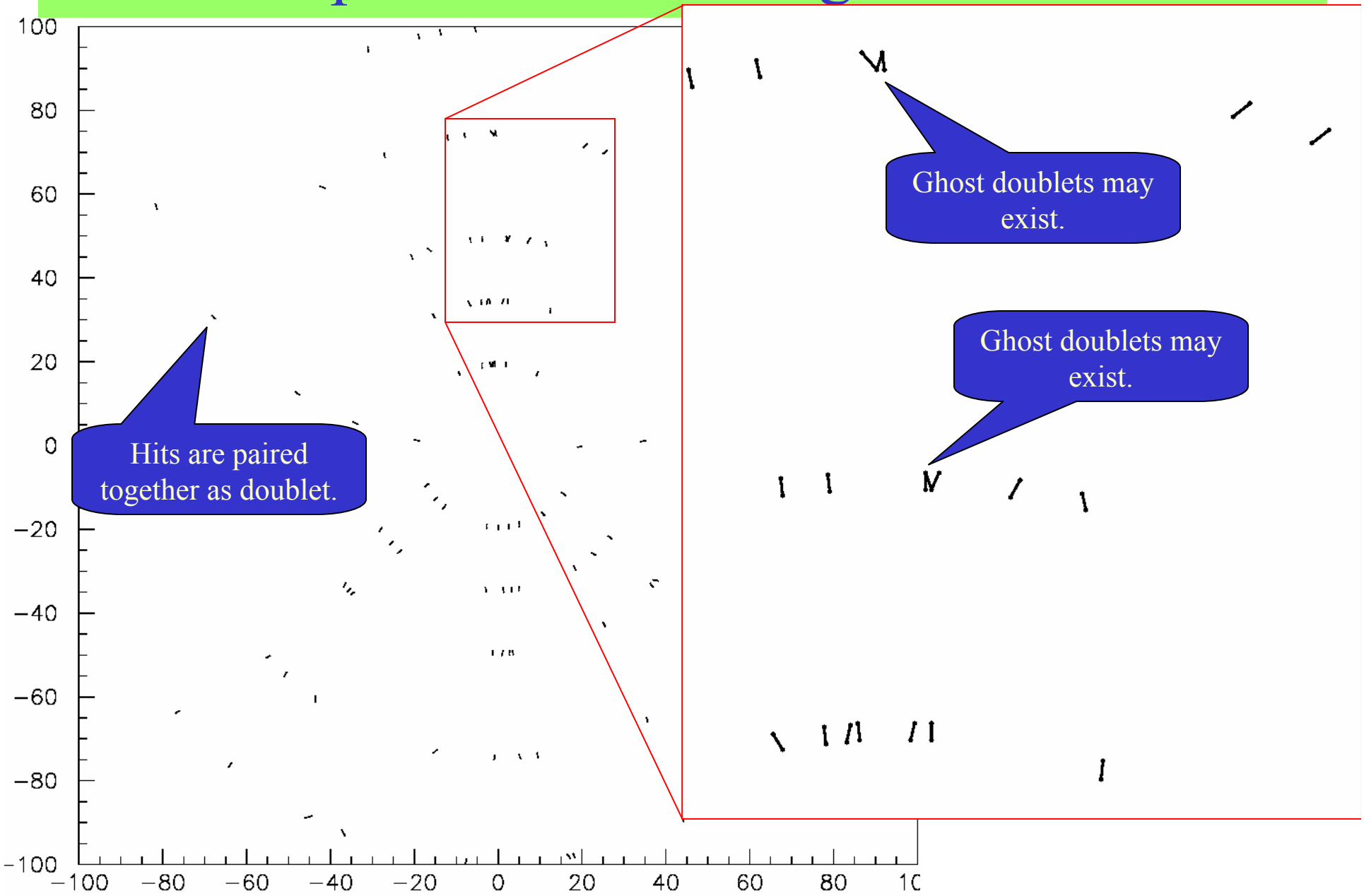


- We explain the track recognition process using this 20-track example.

An Example of Track Recognition: *Hits*



An Example of Track Recognition: *Doublets*



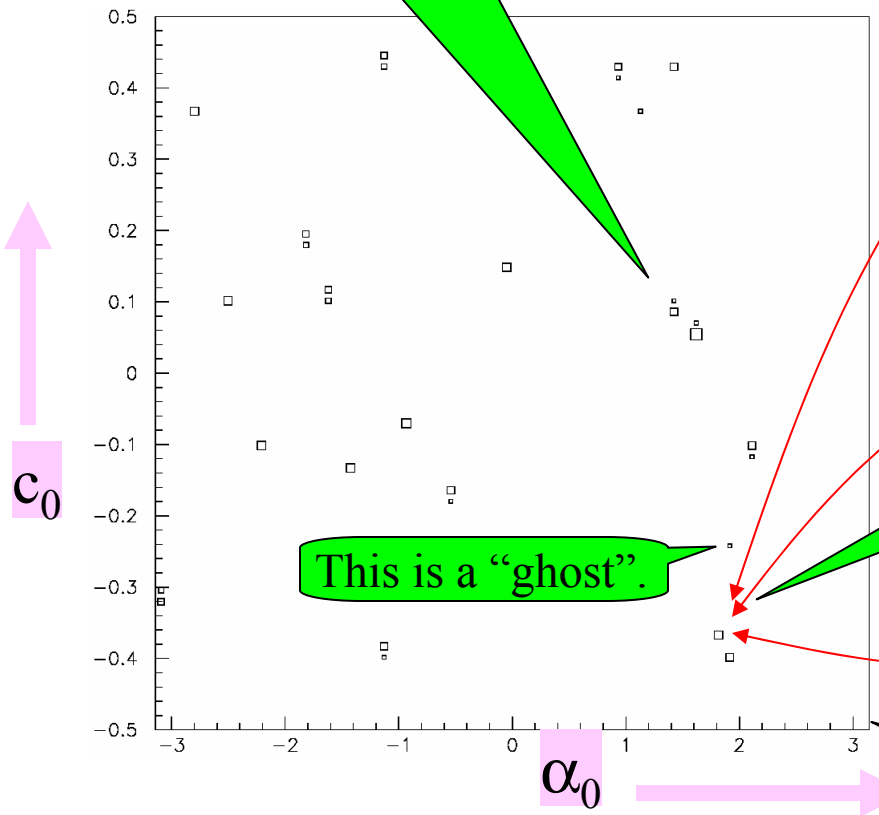
An Example of Track Recognition: *Histogram*

$$\alpha_0 = 2\phi - \alpha$$

$$c_0 = \frac{25cm}{R} = \frac{50cm}{r}\sin(\alpha - \phi)$$

Two track parameters can be calculated for each doublet.

Sometimes they are stored in clusters.



This is a "ghost".

Doublets from same track are entered into same bin, (since they have same track parameters).

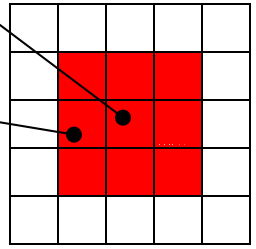
A 2-D histogram is booked.

An Example of Track Recognition: *Clustering*

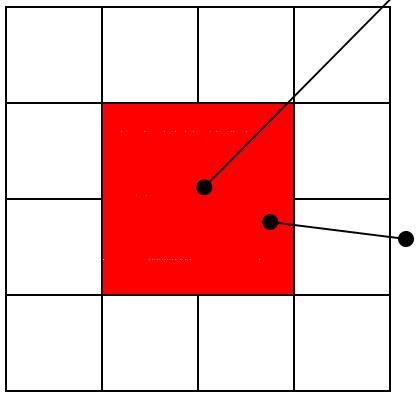
For doublets on the seeding super layer in this bin…

search for coincident in these 9 bins.

The 9-bin scheme



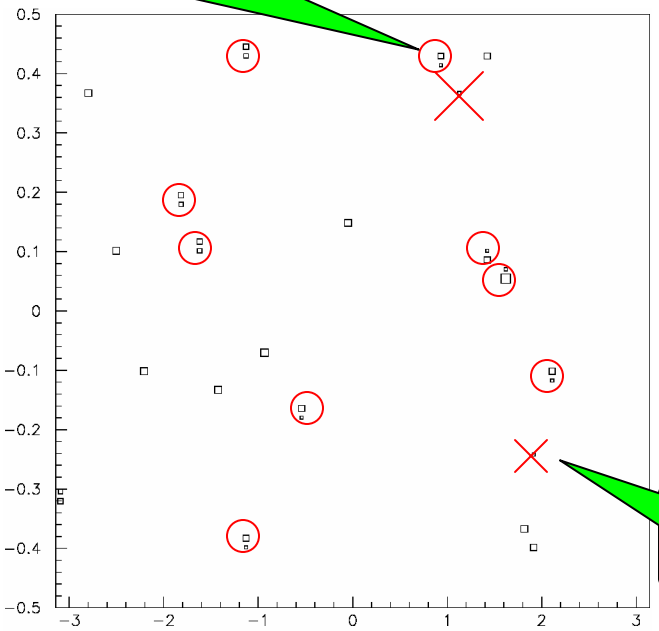
The 4-bin scheme



For doublets on the seeding super layer in this bin…

search for coincident in these 4 bins.

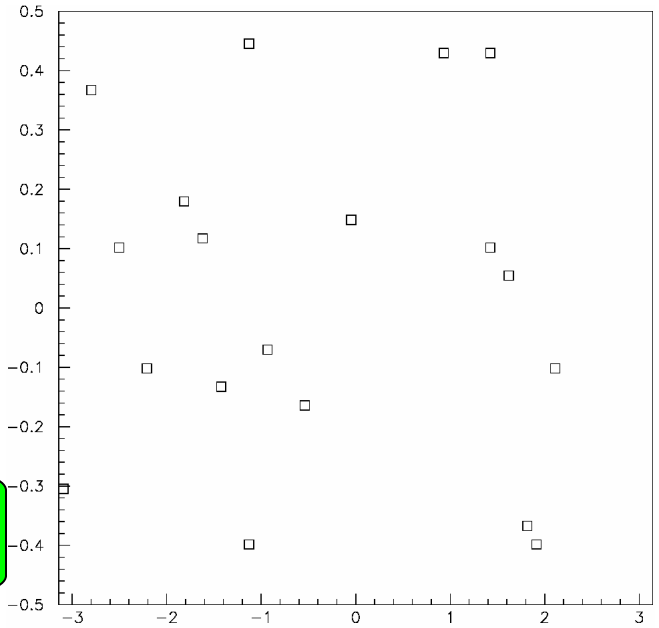
The doublets in clusters are grouped together.



c_0

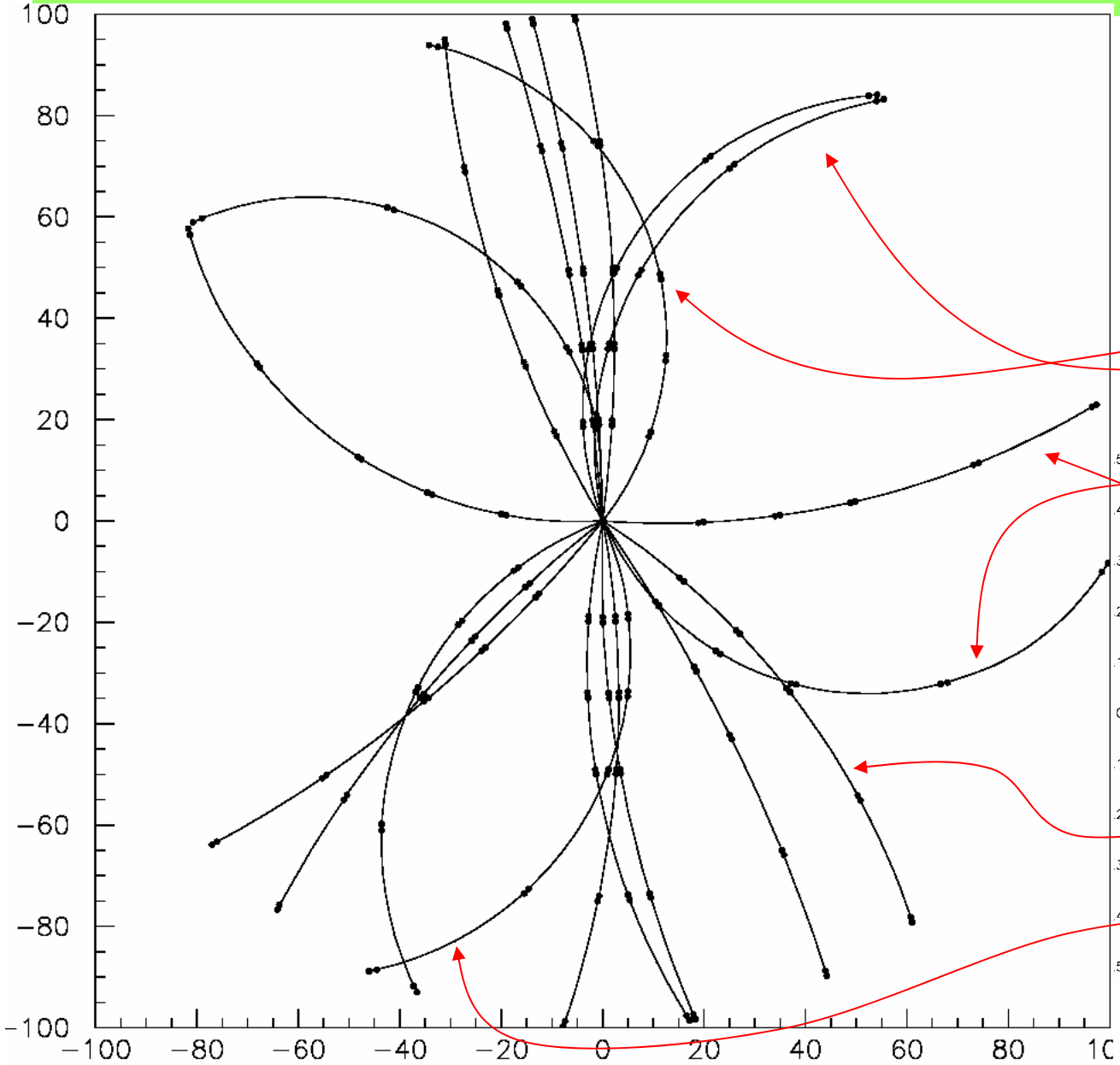
α_0

clustering

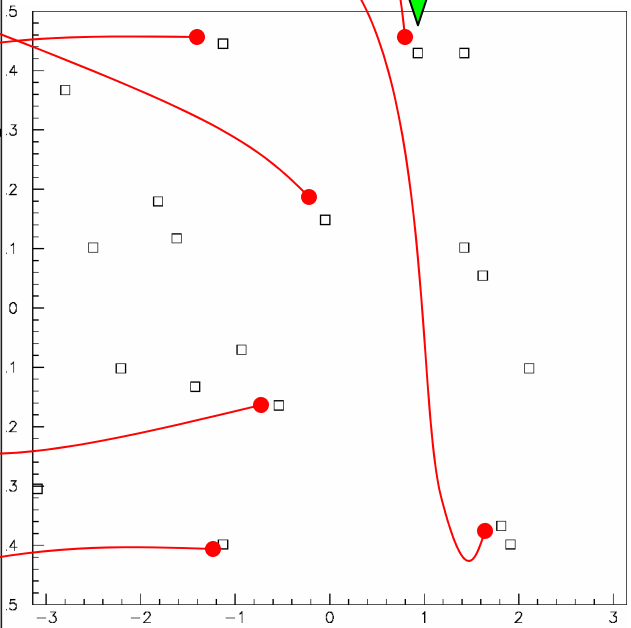


The "ghost" doublets are gone.

An Example of Track Recognition: *Tracks*



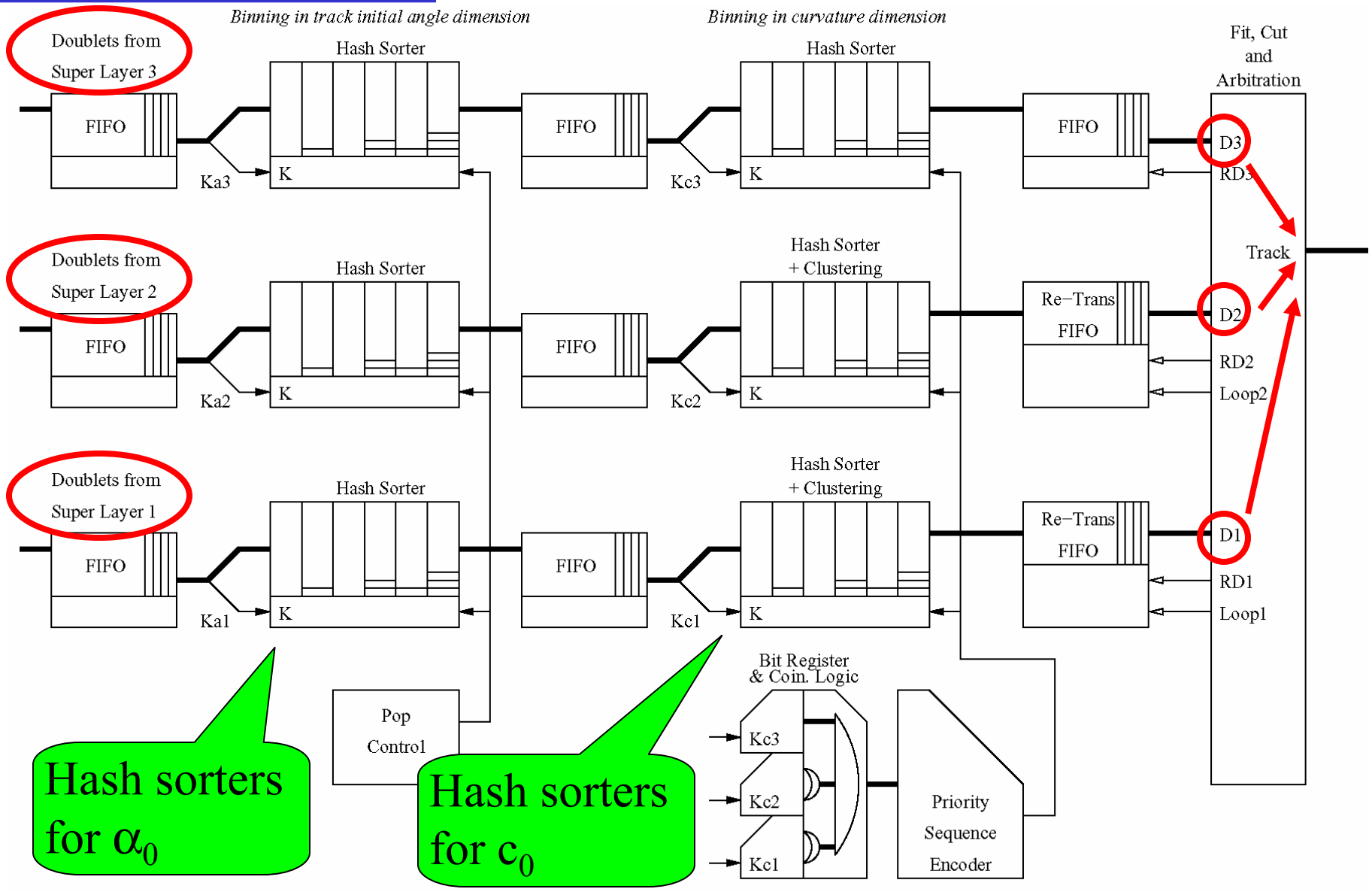
All doublets from a track are contained in a cluster.



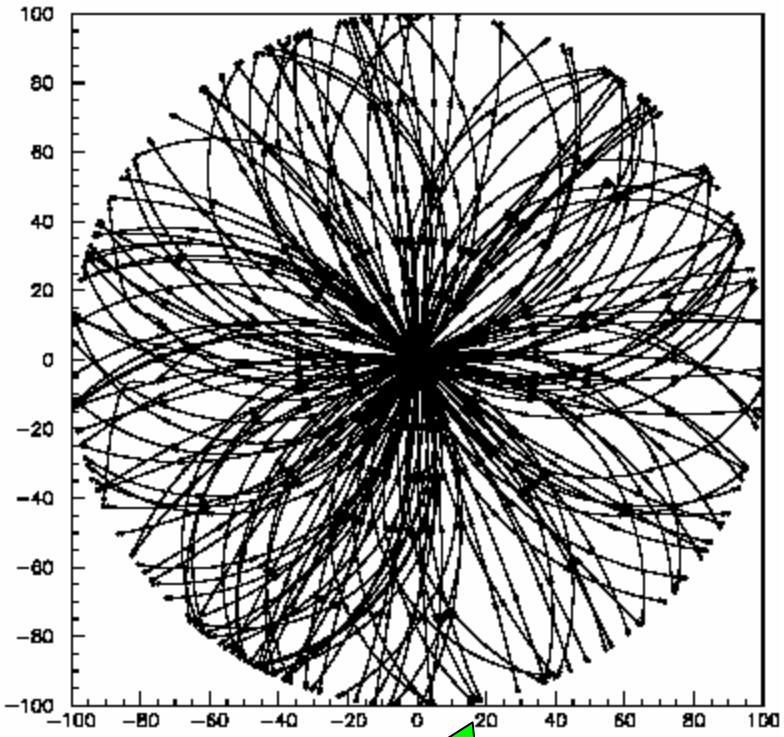
$$\alpha_0 = 2\phi - \alpha$$

$$c_0 = \frac{25cm}{R} = \frac{50cm}{r}\sin(\alpha - \phi)$$

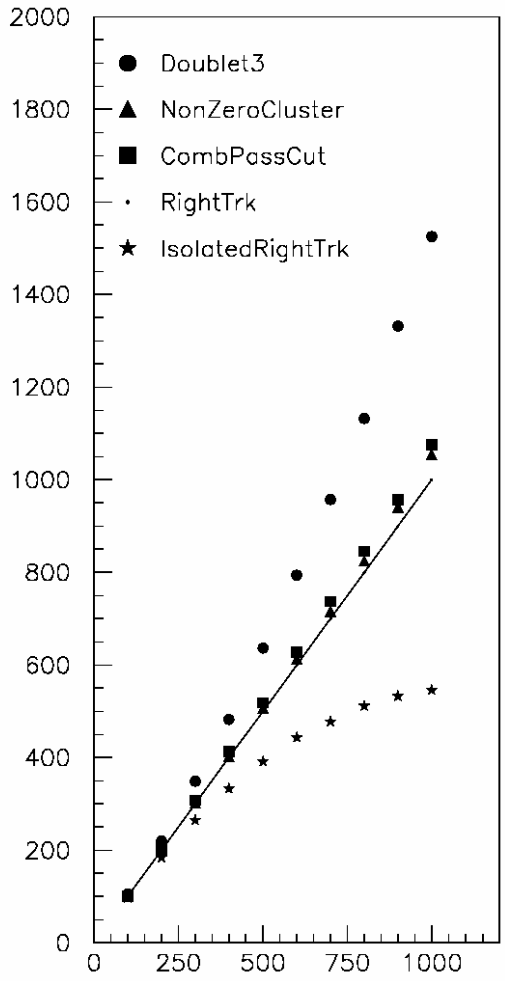
FPGA Block Diagram

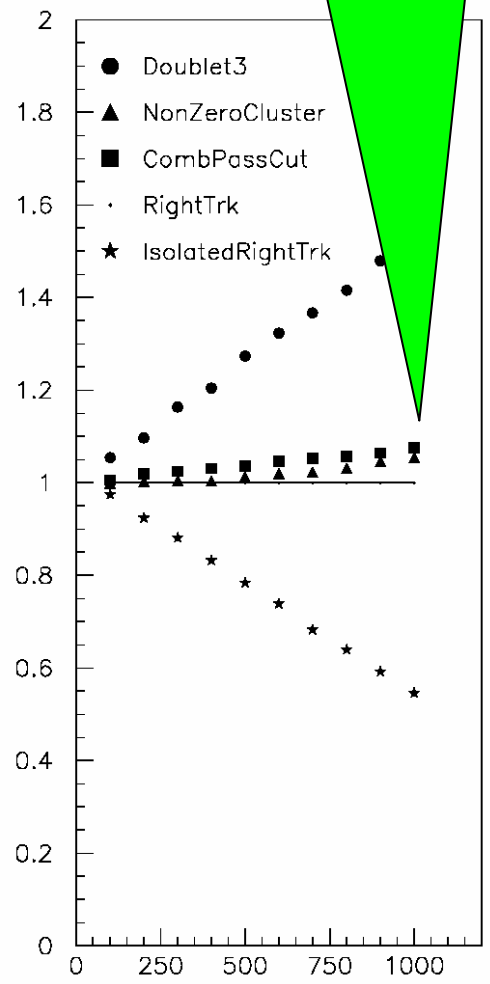


Simulation Results



An event with 200 tracks





It still works at 1000 tracks/event

Without Full Track Recognition

$$\alpha_0 = 2\phi - \alpha$$

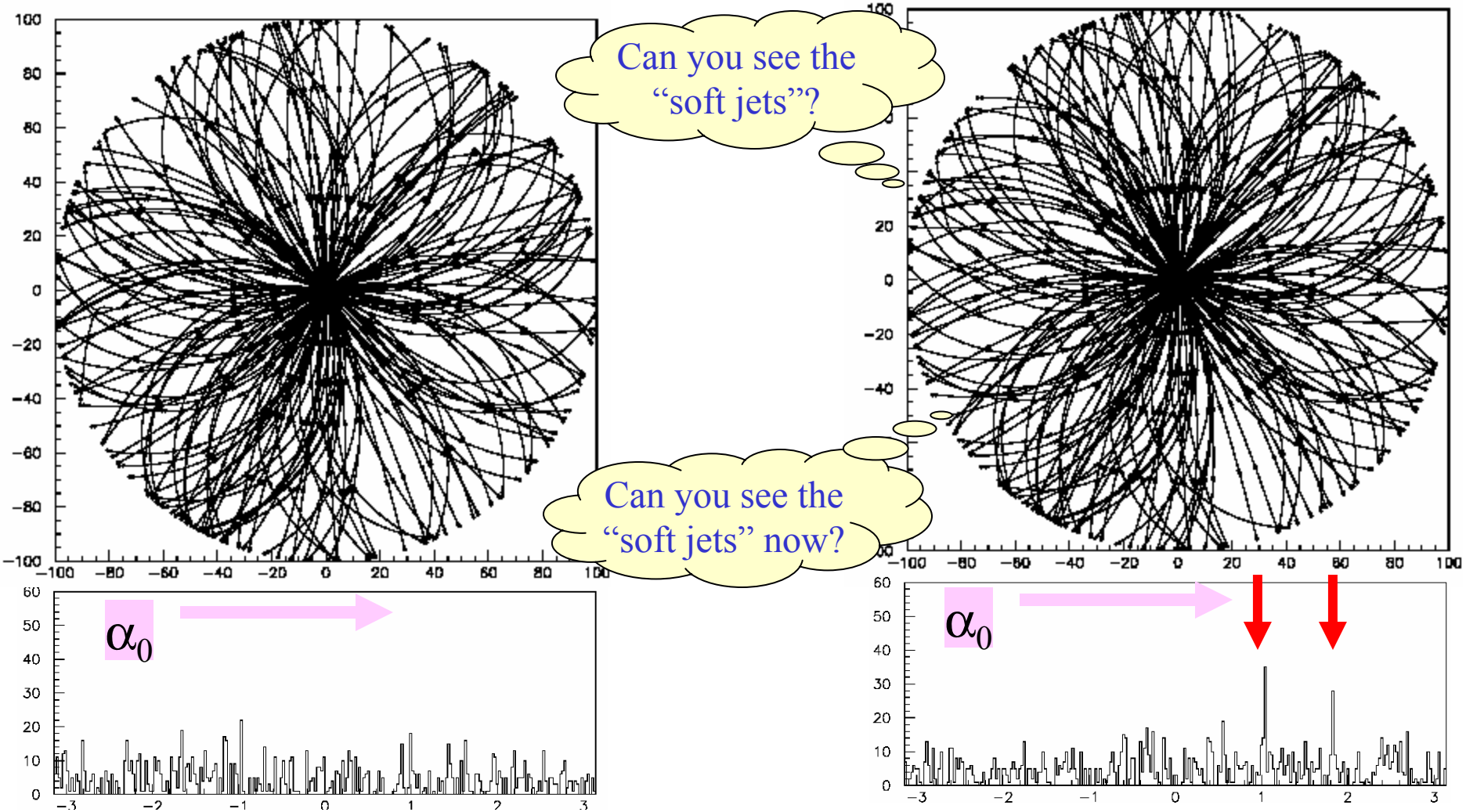
$$c_0 = \frac{25cm}{R} = \frac{50cm}{r}\sin(\alpha - \phi)$$

- Two track parameters can be calculated for each doublet.
- Useful trigger primitives can be found without full track recognition.
- For example…

Example: Finding "Soft Jets"

- A simulated event with 200 tracks.
- Flat distributions.
- Min. R = 55 cm

- 16 soft tracks are added.
- They are grouped in 2 small initial angle regions, i.e., 2 "soft jets".



Soft Tracks?

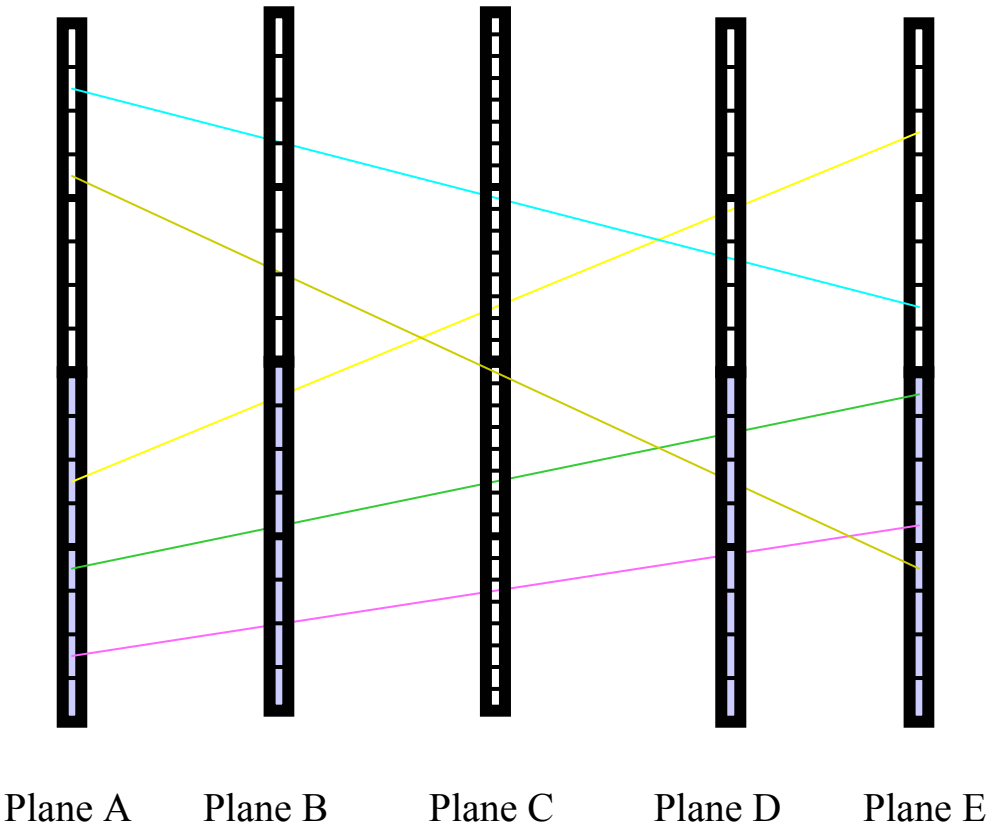
- Can we always anticipate high-p_T signatures?
  - Probably not.
- Do soft tracks carry useful information?
  - Maybe.
- In strong magnetic fields, (e.g. 4T in CMS), high-p_T tracks look soft.
- Isn't it too hard to use soft tracks in trigger stage?
  - It was, but now it is not too hard. ☺

The End

Thanks

Pentlet Finding

Beyond Just Bit-wise AND

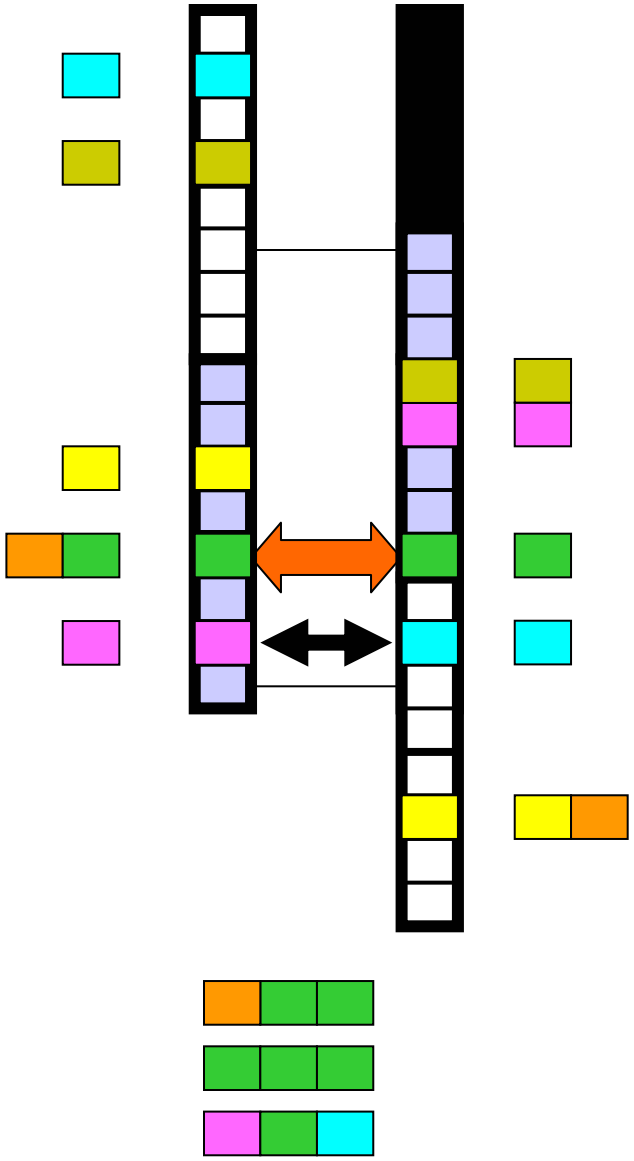


- Use 4 bit arrays.
- There are 3 constraints total.
- More constraints help to eliminating fake tracks.
- It is possible to use bit-wise majority logic (such as 3-out-of-4) to accommodate detector inefficiency issues.

Multiple Hits & Triplets

Keep Them All

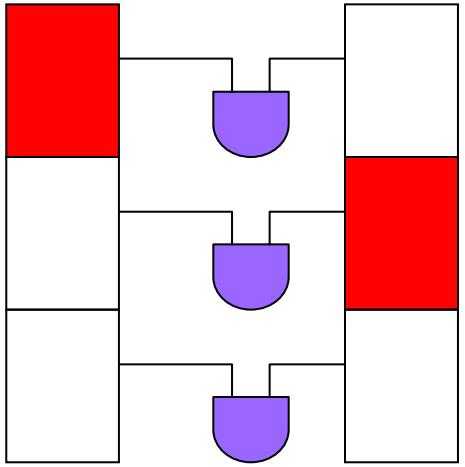
- Each bin may be filled with more than one hits.
- Hits data are kept in hash sorters – allowing multiple hits per bin.
- There are may be more than one match in each bit-wise AND operation.
- They are all sent out, one-by-one, to the later stages for fine cut and arbitration processes.

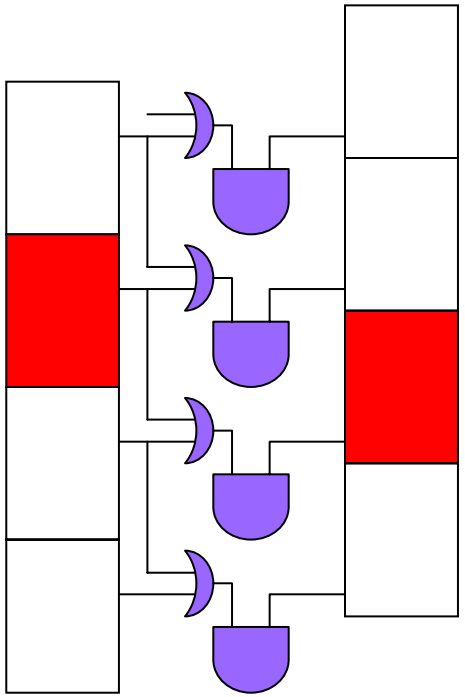


Boundary Issues

Beyond Just Bit-wise AND

- When the track hits near the boundary of a bin, simple bit-wise AND may miss the triplet.
- The bit-wise OR-AND logic will cover the boundary.
- The logic cells in most of today's FPGA's have 4 inputs. So the OR-AND bit-wise logic doesn't increase any resource usage.





Tiny Triplet Finder (Animation)