Systolic Associative arrays for track finding

Associative memories can be used to implement a fast track finder. A simple associative memory (Fig 1) consists of an array of memory cells and comparators. Data stored in the memory cells is compared to data that appears on the Query bus. If any cell matches, its contents are placed on the read bus. Some arbitration is usually needed if more than one match is possible.

Fig 1 Simple Associative Memory

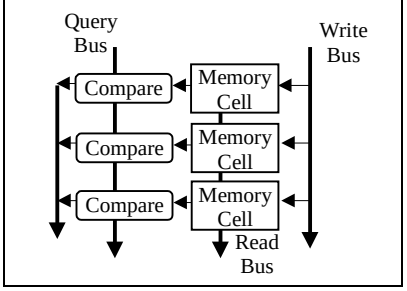
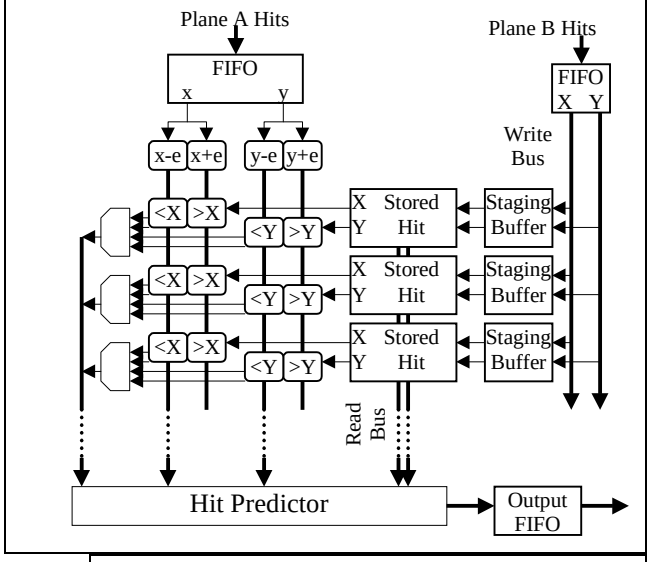


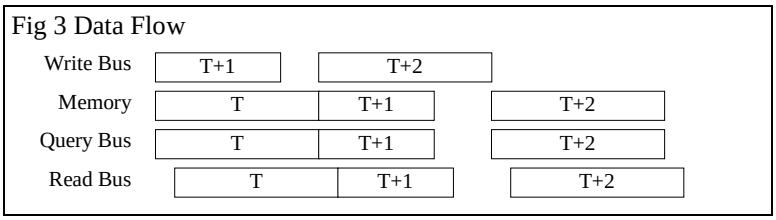
Figure 2 shows additional circuits needed to make an associative track finder. FIFOs are added to the inputs. A staging buffer is added to the write bus. As data comes in on the query bus, upper and lower bounds are computed for X and Y coordinates. Each memory cell has 4 comparators to check upper and lower bounds for both X and Y. The results of the match are sent to a circuit that predicts where the two matching plane hits would project into the next plane.

Fig 2 Hit Matching Associative Memory



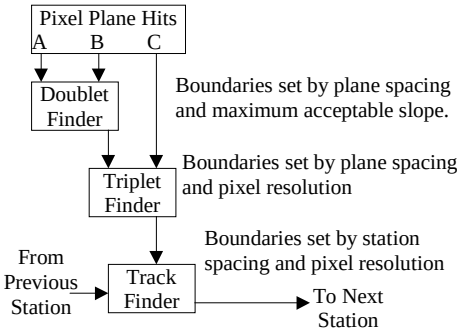
The staging buffer allows the next event to be received while the current event is processed. As soon as all of the hits from plane A have been used, the entire next event is moved from the staging buffer into the memory cells. Figure 3 shows the data flow for 3 events.

Fig 3 Data Flow



The same circuit can be used for finding doublets, triplets, tracks, and perhaps even vertices. Figure 4 shows how three associative units can be used to process station hits. Doublets are formed by matching hits from planes A and B based on the maximum acceptable track slope. The resulting matches are projected into plane C and matched with actual plane C hits to find triplets. The match boundaries in this case are set by the pixel resolution and plane spacing.

Fig 4 Station track finder



In the third unit, triplets are matched to partially-formed tracks that come from the neighboring station. Each matching triplet is projected to the next station and sent along with some additional track parameters. Any unmatched triplets are sent as possibly starting new tracks.

Associative memories have many advantages over other proposed track finders. In addition to speed and size advantages, there are advantages in data flow and parallelizability. An FPGA-based associative memory can process 50 million hits per second. A cheap custom chip could run 2-4 times faster. Preliminary simulations show that a station-quadrant will need to process an average of 21 million hits per second. An entire station-quadrant could be handled with only 2 or 3 chips, making it possible to build a single board that processes several stations at a time.

For even higher performance, it's possible to parallelize processing by zones without losing data. Since the algorithm predicts where it will find a hit in the next station, a hit query can be sent only to the chip that processes the predicted zone. Zone overlap is very small (~300 microns x 800 microns) and any resulting reduntant tracks will be eliminated at the receiving station.

Simulations have been done for this algorithm. Figures 5 through 14 show some results from these simulations. Figures 5-10 show a single reconstructed event. Figures 5 and 6 show the X and Y outputs for the doublet finder versus actual station hits. Each blue dot represents a prediction for where a hit can be found in plane C based on hits from A and B. Figures 7 and 8 show triplets found by the triplet finder. Figures 9 and 10 show track predictions for each station.

Figures 11-13 are histograms that give a measure of bandwidths for each stage. They are based on 2400 simulated events for a station-quadrant. These show an average bandwidth of less than 5 words per event Figure 14 is a histogram of the number of triplets found minus the number of plane-A hits for each station. This gives a measure of efficiency: producing exactly 1 triplet per plane hit is good. Producing more ore less is not as good.

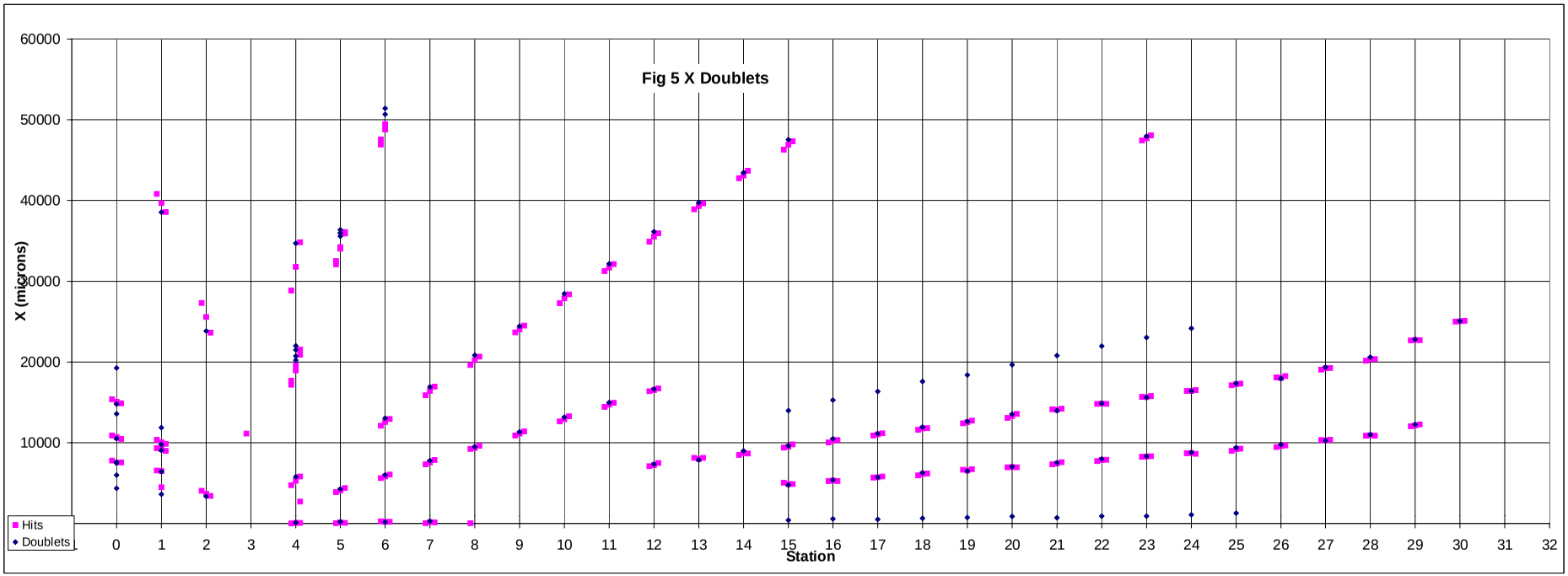


Fig 5 X Doublets
X (microns)
60000
50000
40000
30000
20000
10000
Station
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
Hits
Doublets

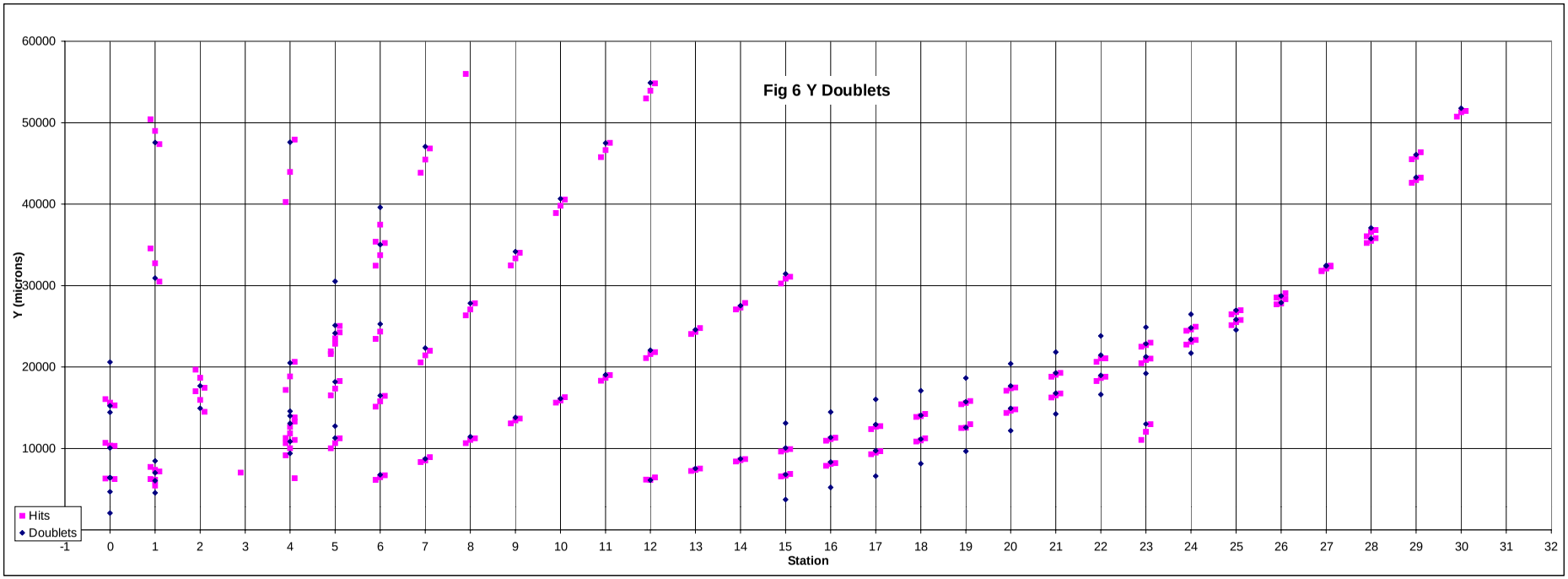


Fig 6 Y Doublets
Y (microns)
60000
50000
40000
30000
20000
10000
Station
-1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
Hits
Doublets

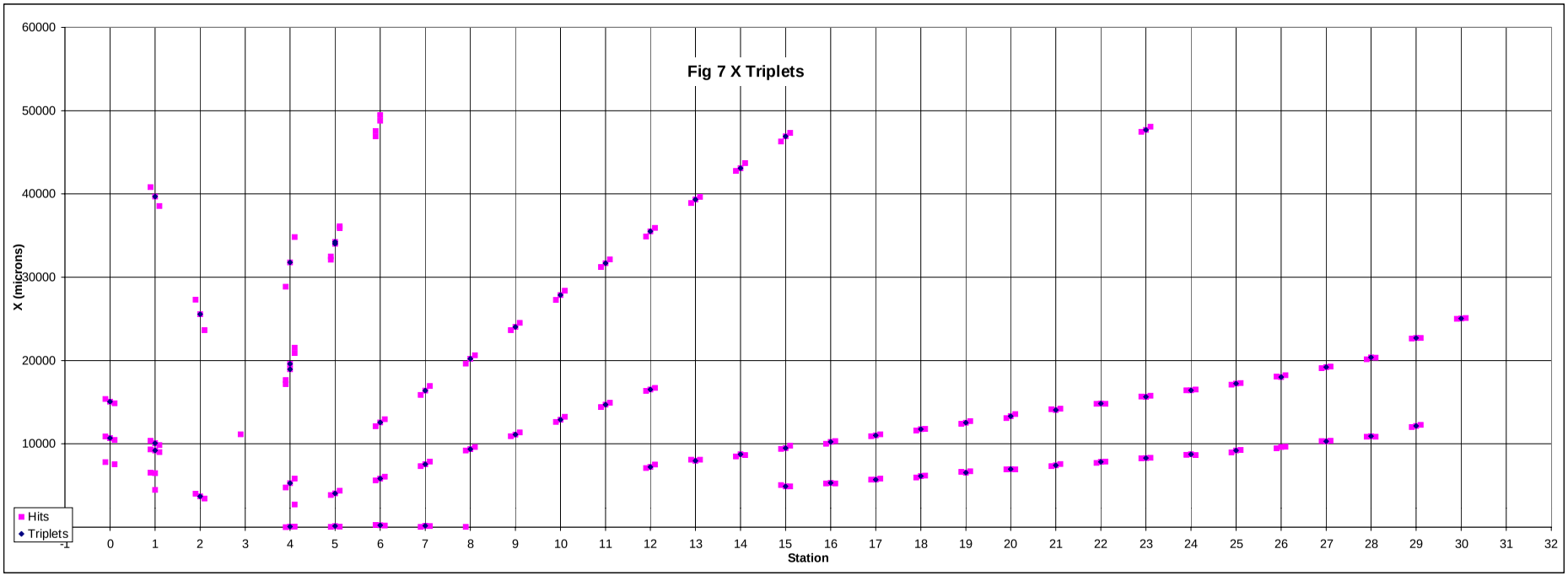


Fig 7 X Triplets
X (microns)
60000
50000
40000
30000
20000
10000
Station
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
Hits
Triplets

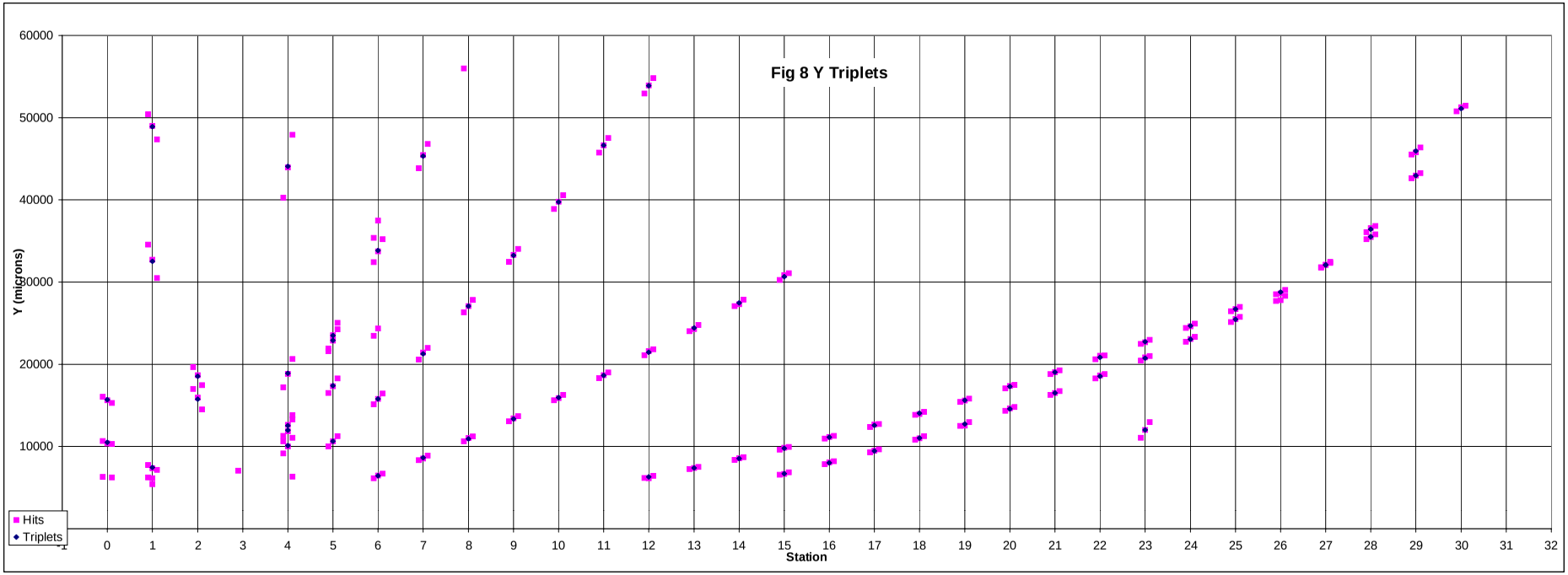


Fig 8 Y Triplets
Y (microns)
60000
50000
40000
30000
20000
10000
Station
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
Hits
Triplets

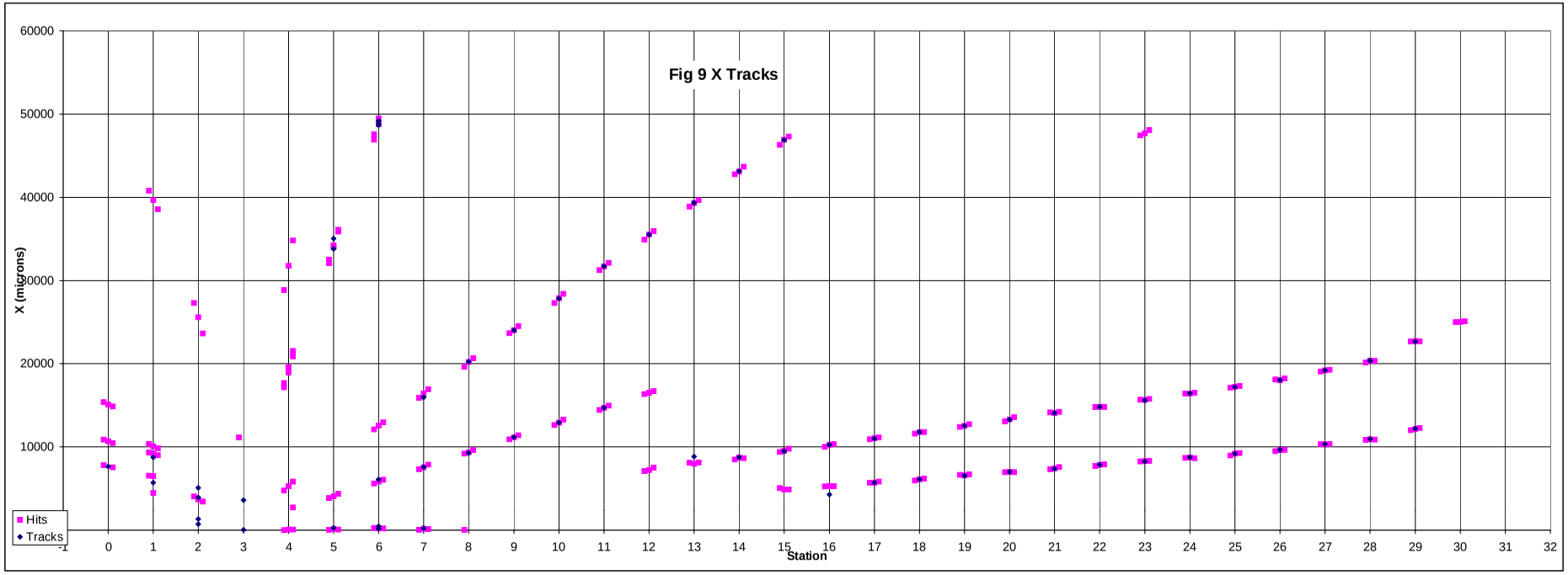


Fig 9 X Tracks
X (microns)
Station
Hits
Tracks

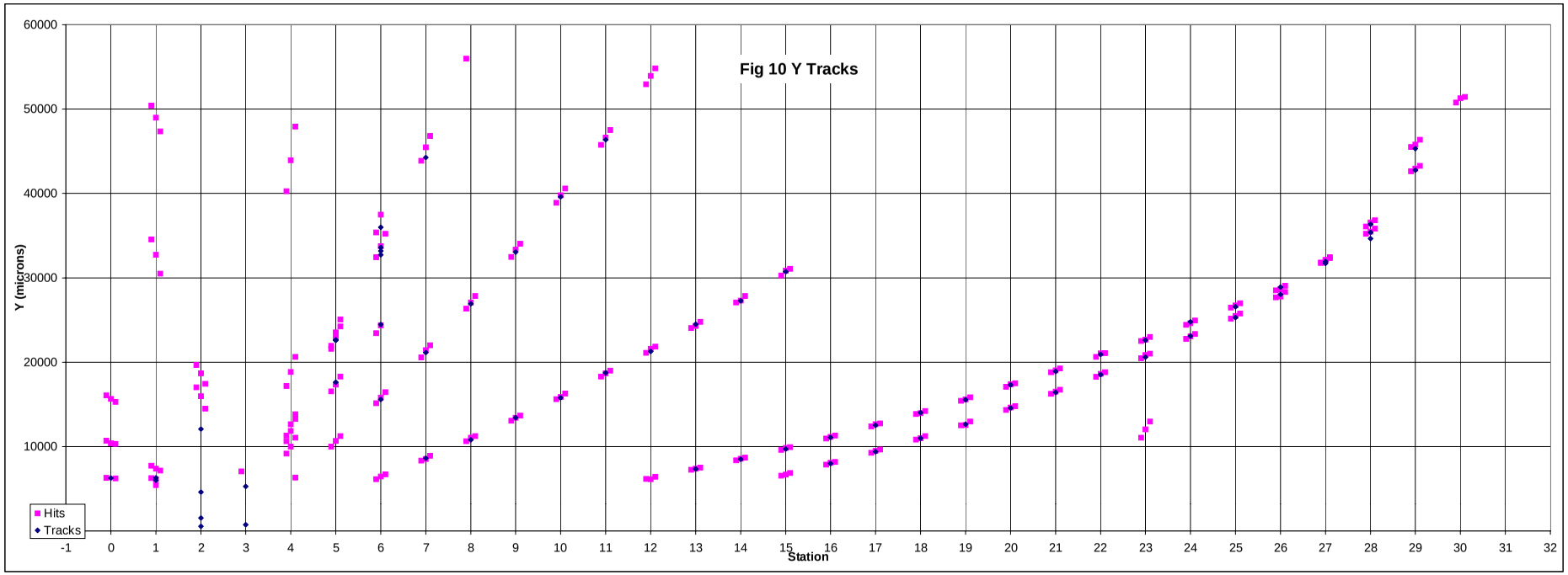


Fig 10 Y Tracks
Y (microns)
Station
Hits
Tracks

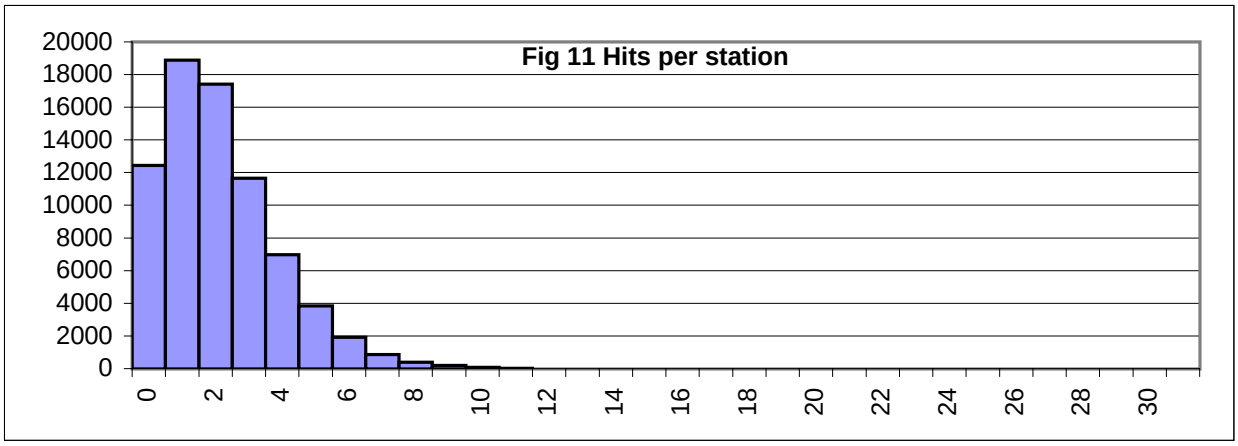


Fig 11 Hits per station
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30

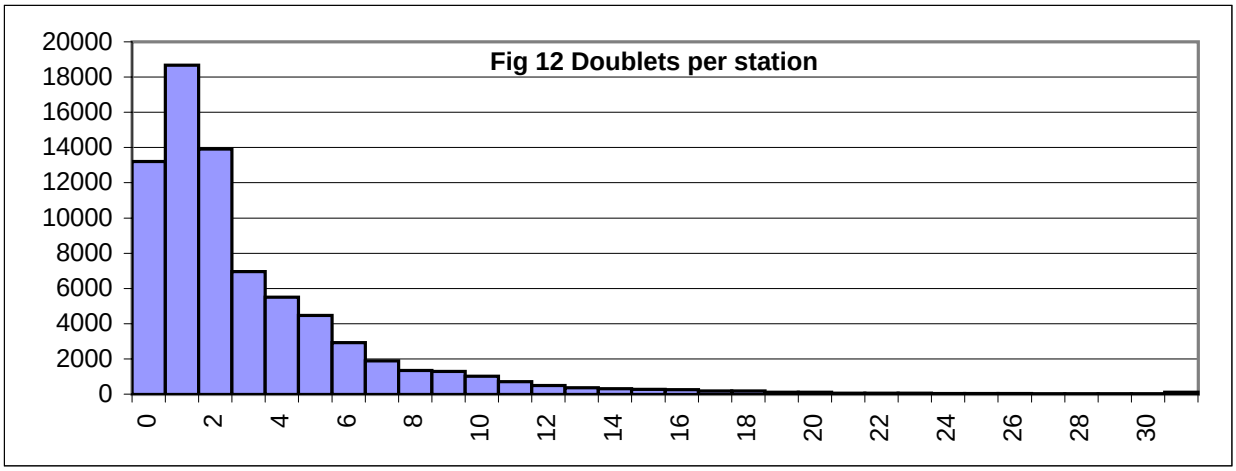


Fig 12 Doublets per station
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30

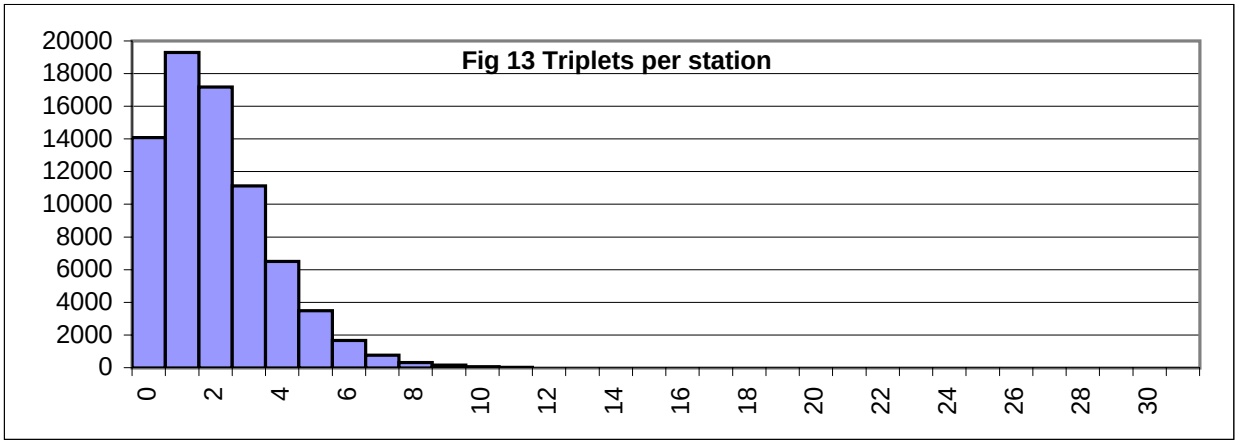


Fig 13 Triplets per station
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30

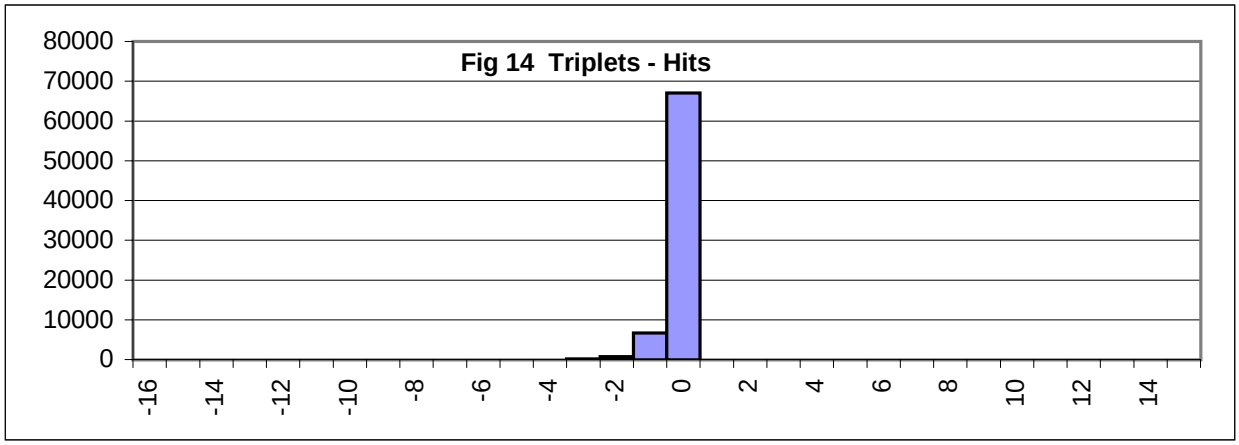


Fig 14 Triplets - Hits
80000
70000
60000
50000
40000
30000
20000
10000
0
-16 -14 -12 -10 -8 -6 -4 -2 0 2 4 6 8 10 12 14

Doublet Finder

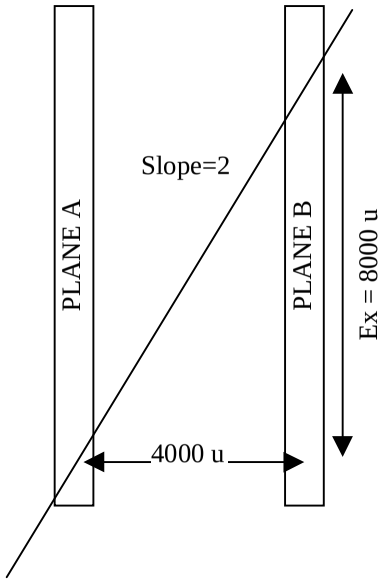
Find all pairs of hits from planes A and B such that

$$Xb > Xa - Ex \quad \text{and} \quad Xb < Xa + Ex$$
$$Yb > Ya - Ey \quad \text{and} \quad Yb < Ya + Ex$$

Where Ex and Ey are determined by the maximum allowable slope and plane spacing:

Then, use the pair to predict where to find a matching hit in plane C:

$$Xd = Xb + Xb - Xa$$
$$Yd = Yb + Yb - Ya$$

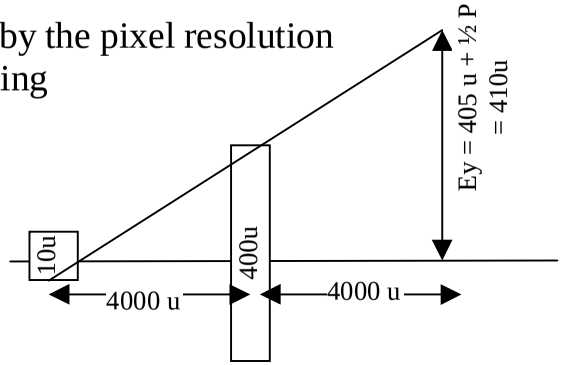


Triplet Finder

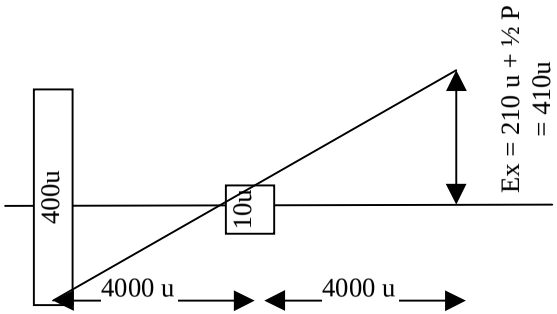
Find all matches of doublets (d) with plane C such that

$$Xc > Xd - Ex \quad \text{and} \quad Xc < Xd + Ex$$
$$Yc > Yd - Ey \quad \text{and} \quad Yc < Yd + Ex$$

Where Ex and Ey are determined by the pixel resolution and plane spacing



Then, use result to match tracks from neighboring stations



Track Tokens

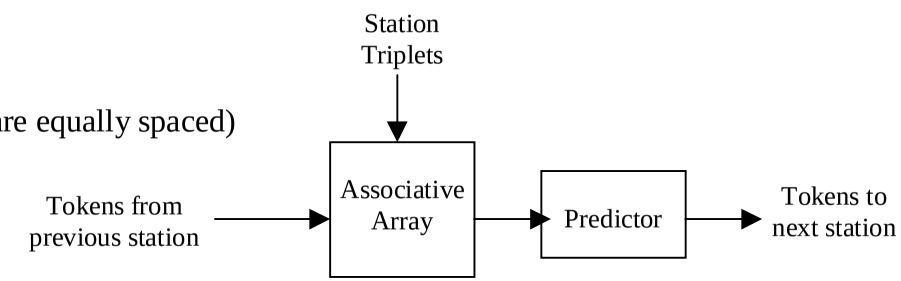
A track token contains a description of a partially-formed track:

15	Xq	Query value (Predicted X)
15	Yq	Query value (Predicted Y)
15	Xs	X from previous station
15	Ys	Y from previous station
5	S0	Station where first hit was found
5	Hits	Count of matched hits in track
2	Miss	Count of misses in track
10	ddY	Curvature of track (Optional)
82		Total bits in track token

When a match occurs, the next track token is computed from the incoming token and the matching station hit (Xt,Yt) as:

ddY= Yt-Yq
Xq= Xt + (Xt-Xs)
Yq= Yt + (Yt-Ys) + ddY
Xs= Xt
Ys= Yt

(Assume that stations are equally spaced)



READOUT

A track token contains enough information to describe two station hits. When combined with a third hit, there is enough information (including curvature) to project the track back to the Z-axis (or not).

It may be enough to simply read out only the ends of tracks that appear to be headed towards the Z-axis and send them to the virtex processor for further processing.

If necessary, we can send both ends of a track.

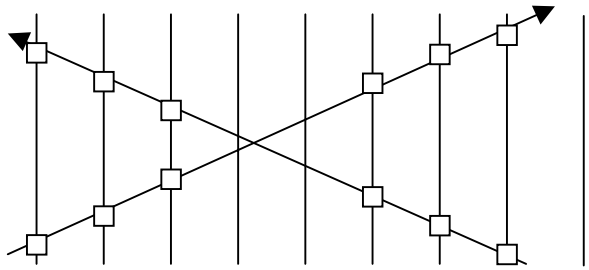


Table 2 outlines the features of the APEX 20K device family.

Table 2. APEX 20K Device Features, *Note (1)*

Feature	EP20K100	EP20K160	EP20K200	EP20K300	EP20K400	EP20K600	EP20K1000
Maximum gates	263,000	404,000	526,000	728,000	1,052,000	1,537,000	2,670,000
Typical gates	53,000 to 106,000	82,000 to 163,000	106,000 to 211,000	147,000 to 293,000	213,000 to 423,000	311,000 to 618,000	541,000 to 1,073,000
Logic elements	4,160	6,400	8,320	11,520	16,640	24,320	42,240
Embedded system blocks	26	40	52	72	104	152	264
Maximum RAM bits	53,248	81,920	106,496	147,456	212,992	311,296	540,672
Maximum macrocells	416	640	832	1,152	1,664	2,432	4,224
Maximum I/O pins	250	320	320	420	500	620	780
Packages, *Note (2)*	FineLine BGA 144-pin TQFP 208-pin QFP 240-pin QFP	FineLine BGA 144-pin TQFP 208-pin QFP 240-pin QFP	FineLine BGA 208-pin QFP 240-pin QFP	FineLine BGA 208-pin QFP 240-pin QFP	FineLine BGA 208-pin QFP 240-pin QFP	FineLine BGA	FineLine BGA

Content Addressable Memory

Content addressable memory (CAM) is a memory technology that accelerates search applications and is widely used in high-speed communication applications. Unlike RAM, which receives an address as input and outputs data, CAM takes a data input and outputs the address. Each APEX 20K ESB can be configured as a 32-word by 32-bit CAM, and multiple ESBs can be cascaded to support larger CAM blocks. APEX 20K CAM is faster than traditional CAM.