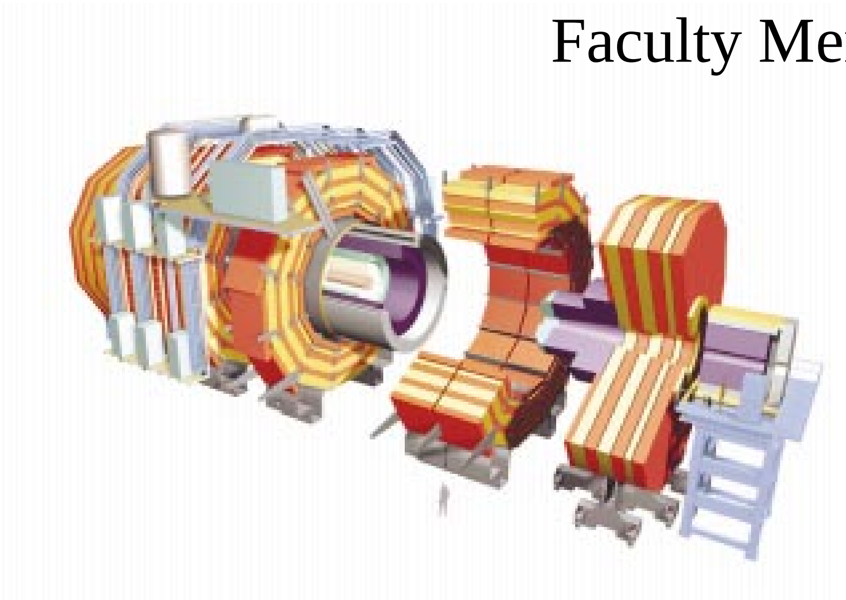
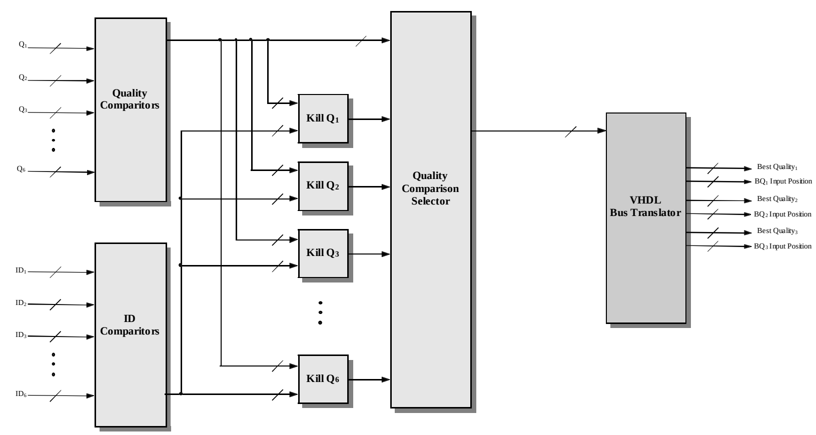
CMS and the Number Sorter

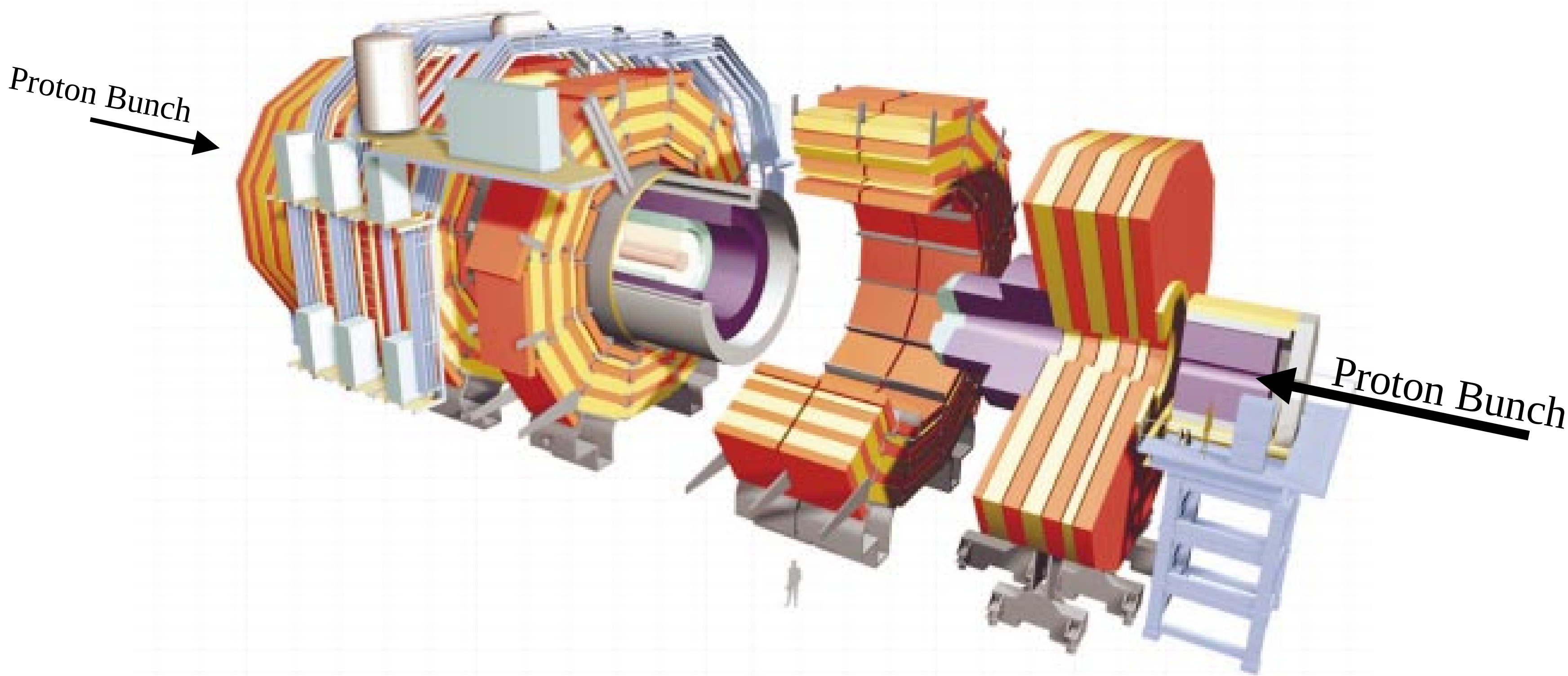
Presented to you by: Bobby Scurlock

Faculty Mentor: Darin Acosta





The CMS Detector



40,000,000 Bunch Collisions each second

Event = *total collection* of detector measurements produced resulting from 1 proton-proton bunch collision.

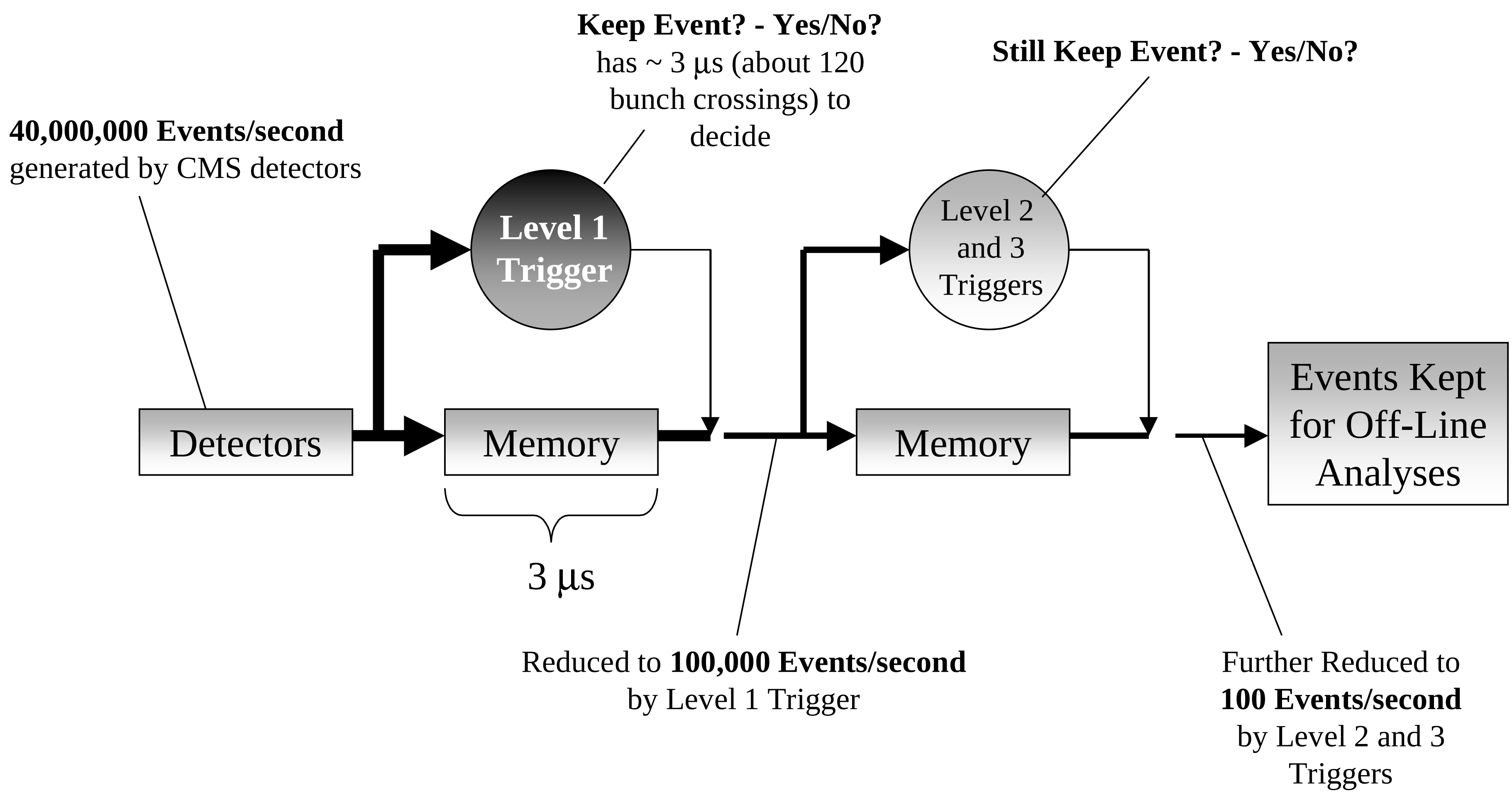
Do We Keep it All?!!

- 1 Event generates 1 Megabyte of data
- **40 Terabytes of data/second!!**
- Requires the storage capacity of 40,000 average home PCs for every second of operation!!

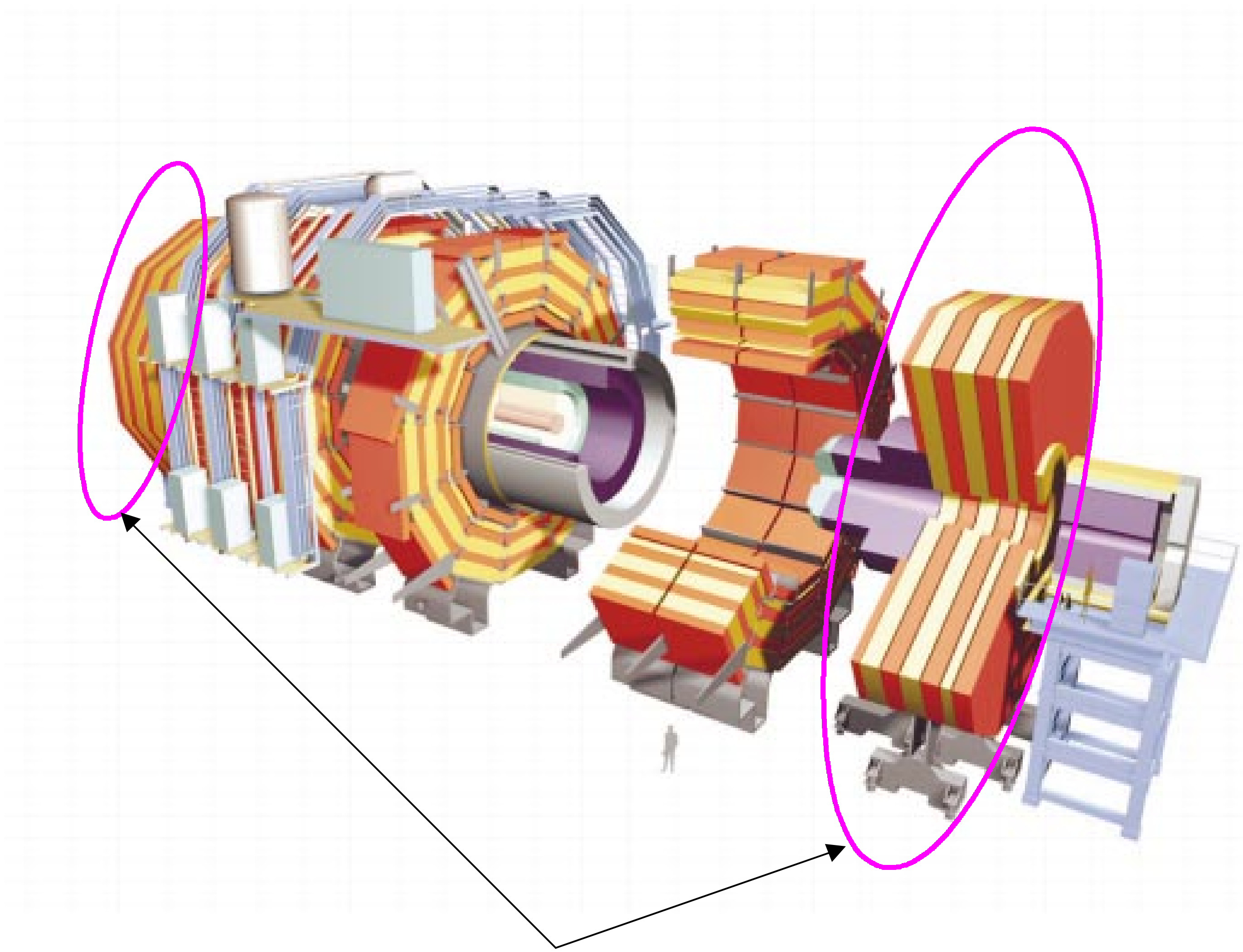
NOT ALL EVENTS ARE INTERESTING

Need an Electronic trigger to decide which events are important

Trigger and Data Acquisition Scheme of CMS

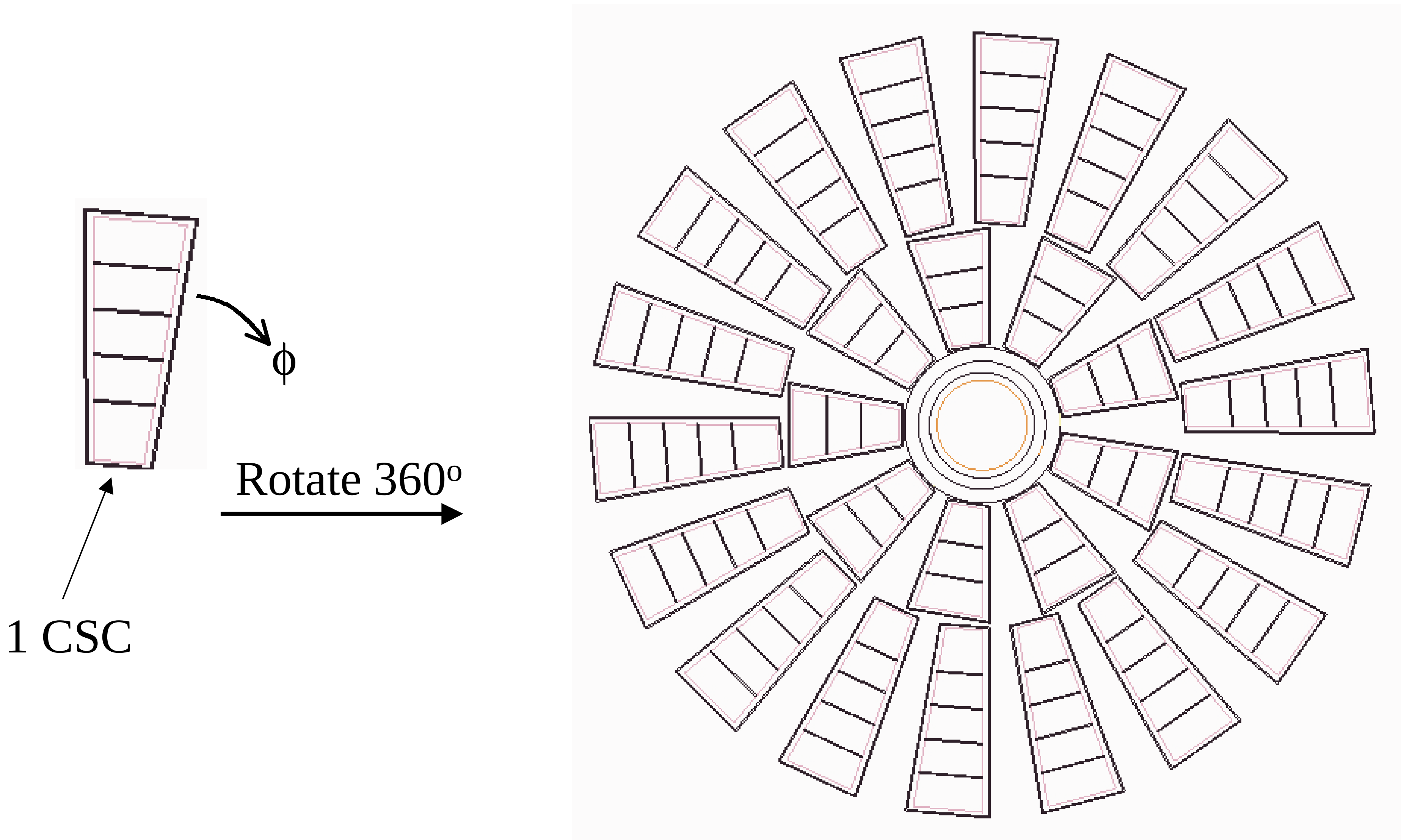


My Project



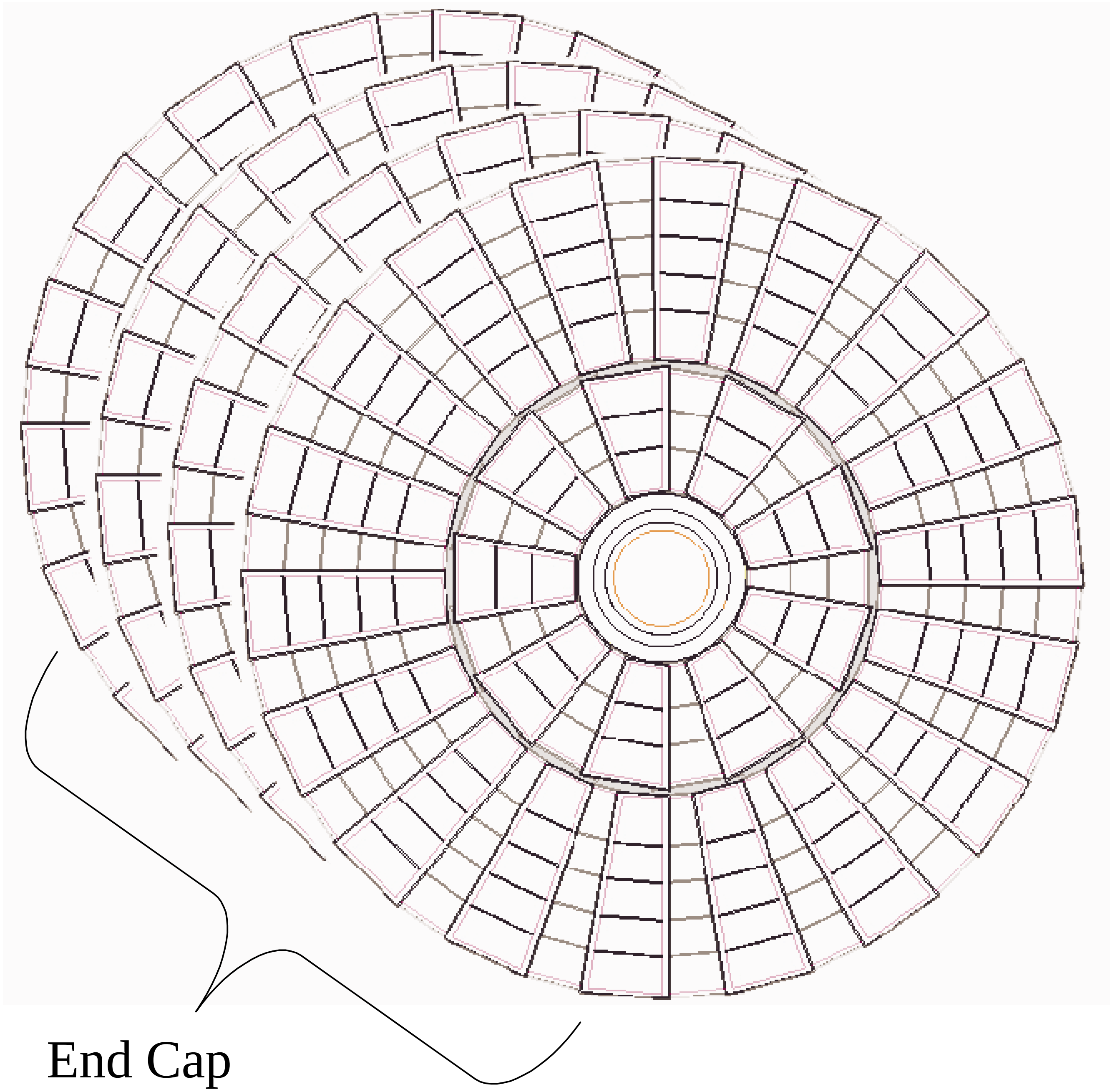
My Project involved the End Cap Regions of CMS

What are End Caps Anyway?



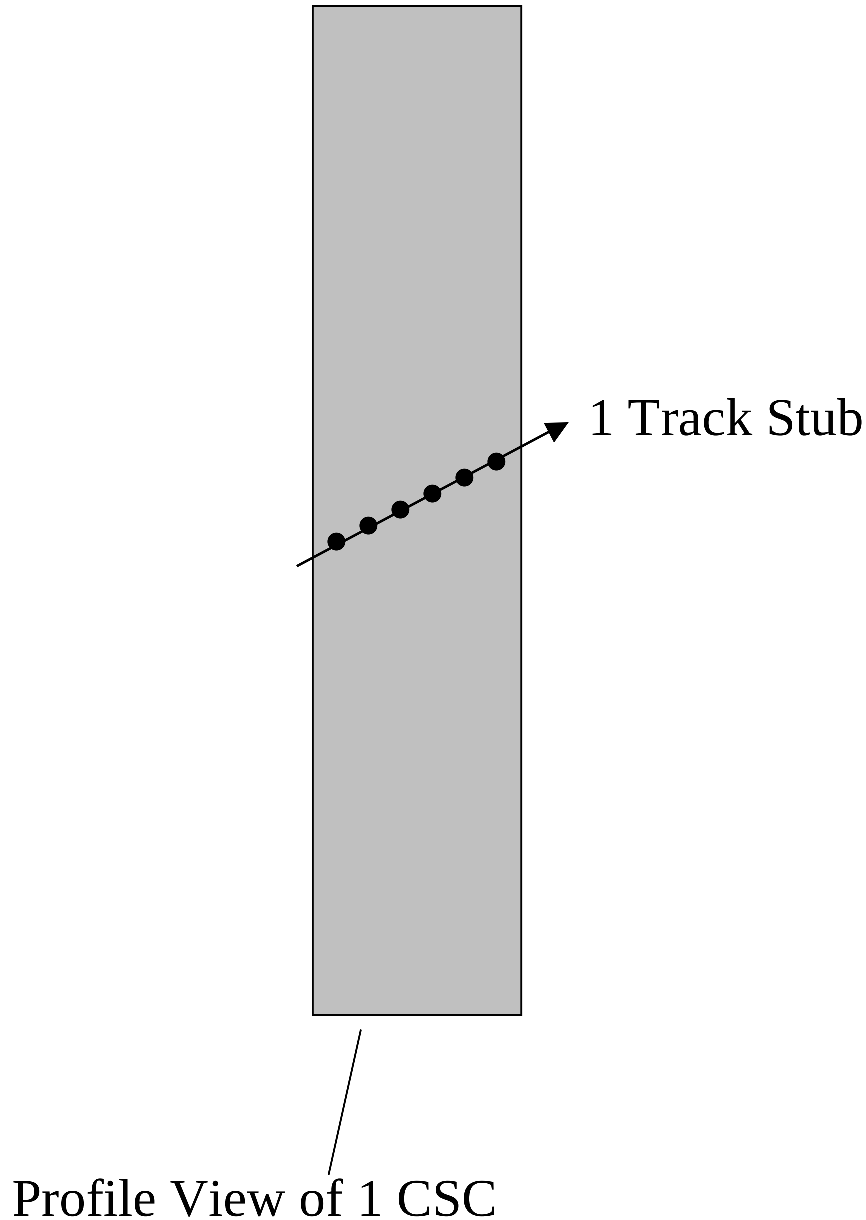
1 Disk = 1 Station

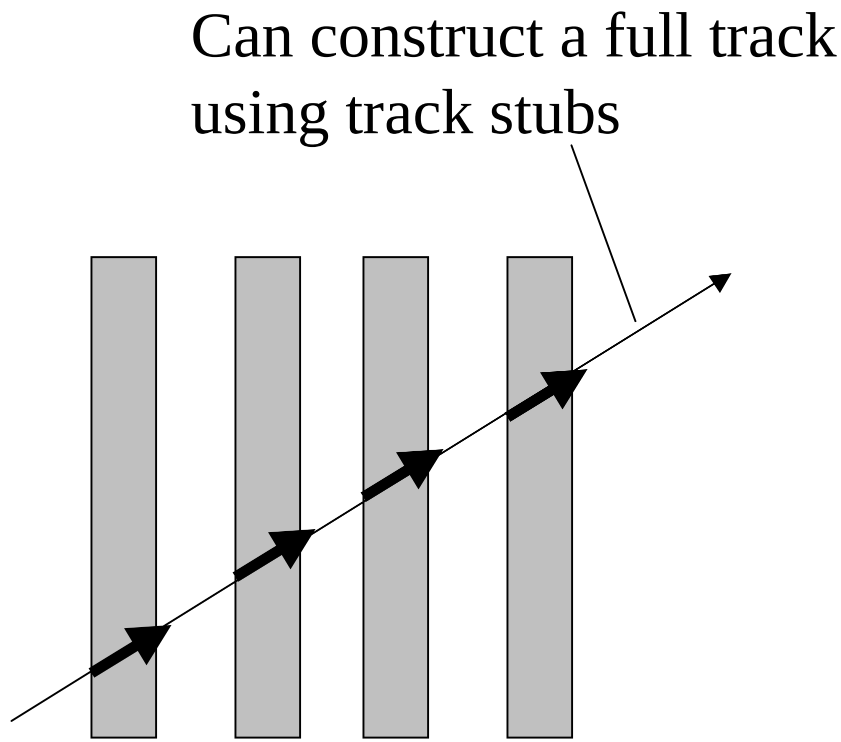
Combine 4
Stations and Get 1
End Cap



End Cap

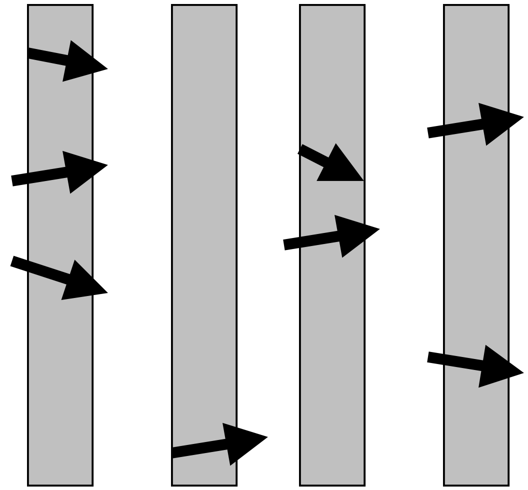
Review of CSC Operation





Profile View of 4 CSCs

Profile View of 1- 60^{o} Sector



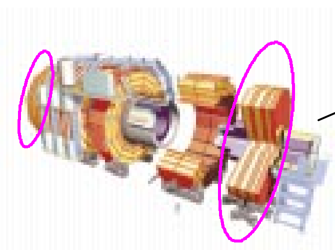
Real Track Stubs may actually look like this

Why Not Always Perfect Lines?

- Detectors not 100% Efficient
- Many particles travel through detectors
- A lot of radiation being produced in throughout the detectors

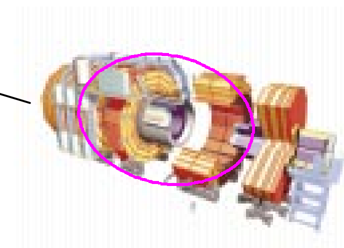
Level 1 Trigger

Level 1 Trigger Scheme



CSC Detector Logic

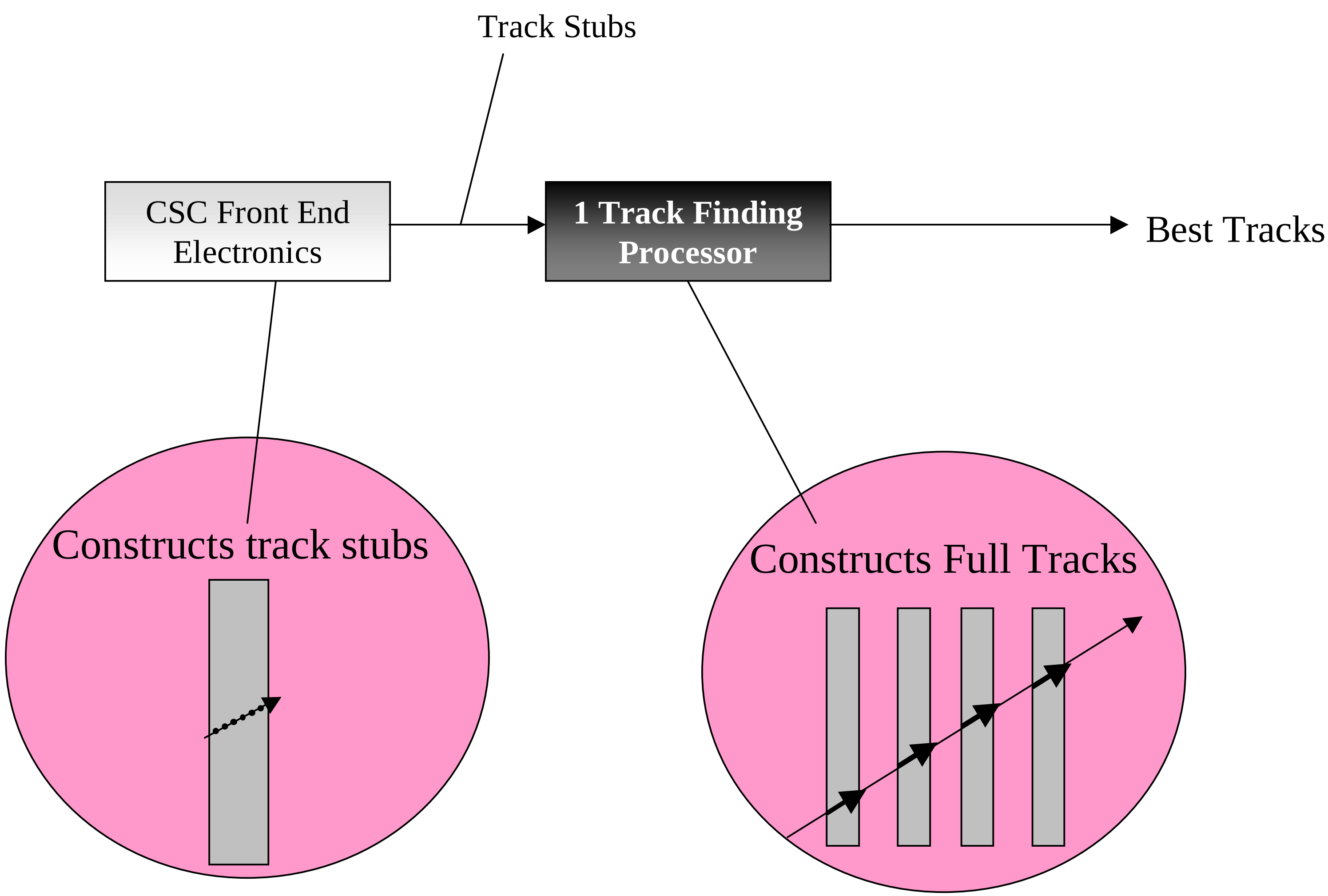
Other Detector Logic

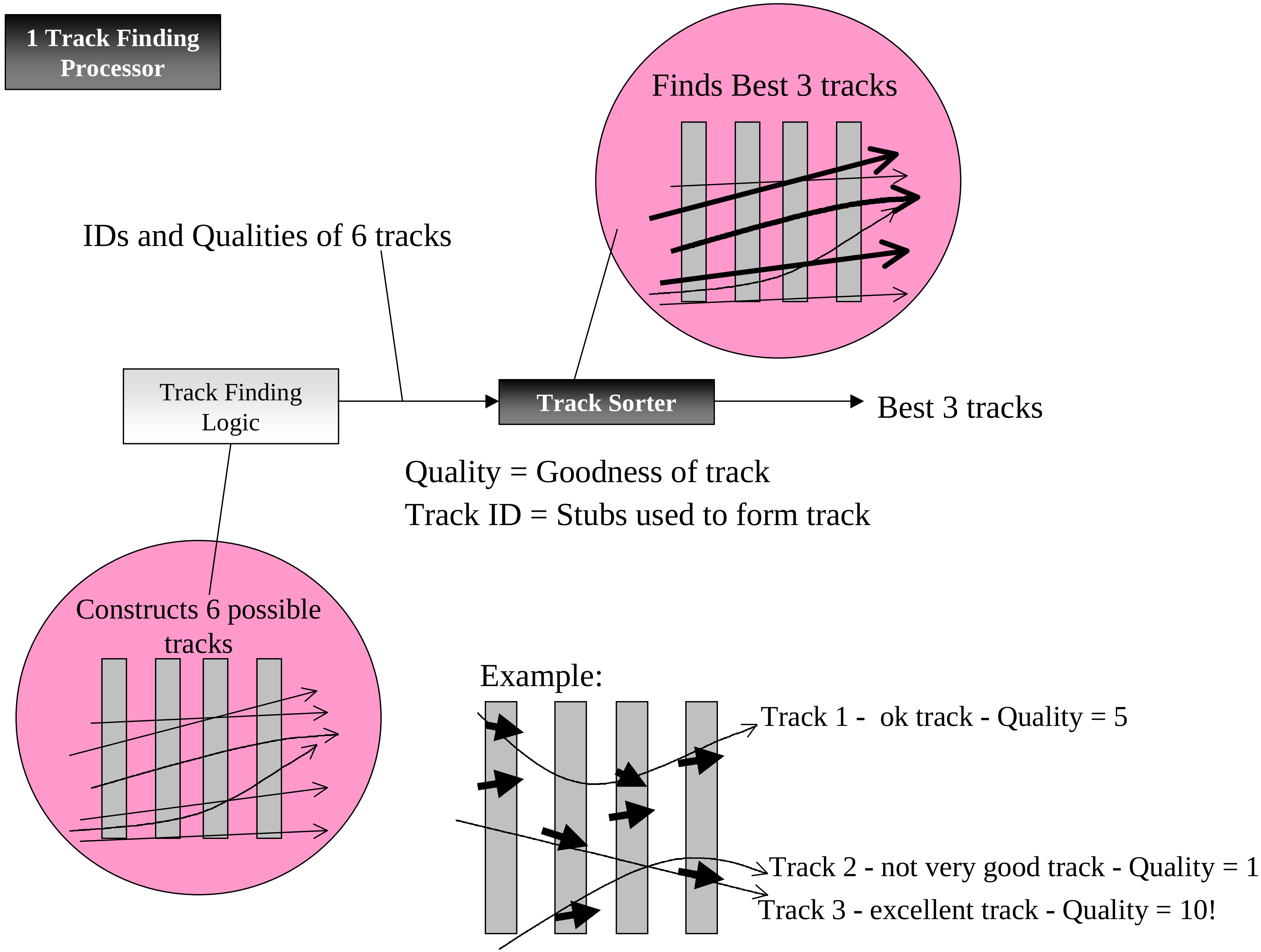


Final Decision Making Logic

Keep Event? - Yes or No?

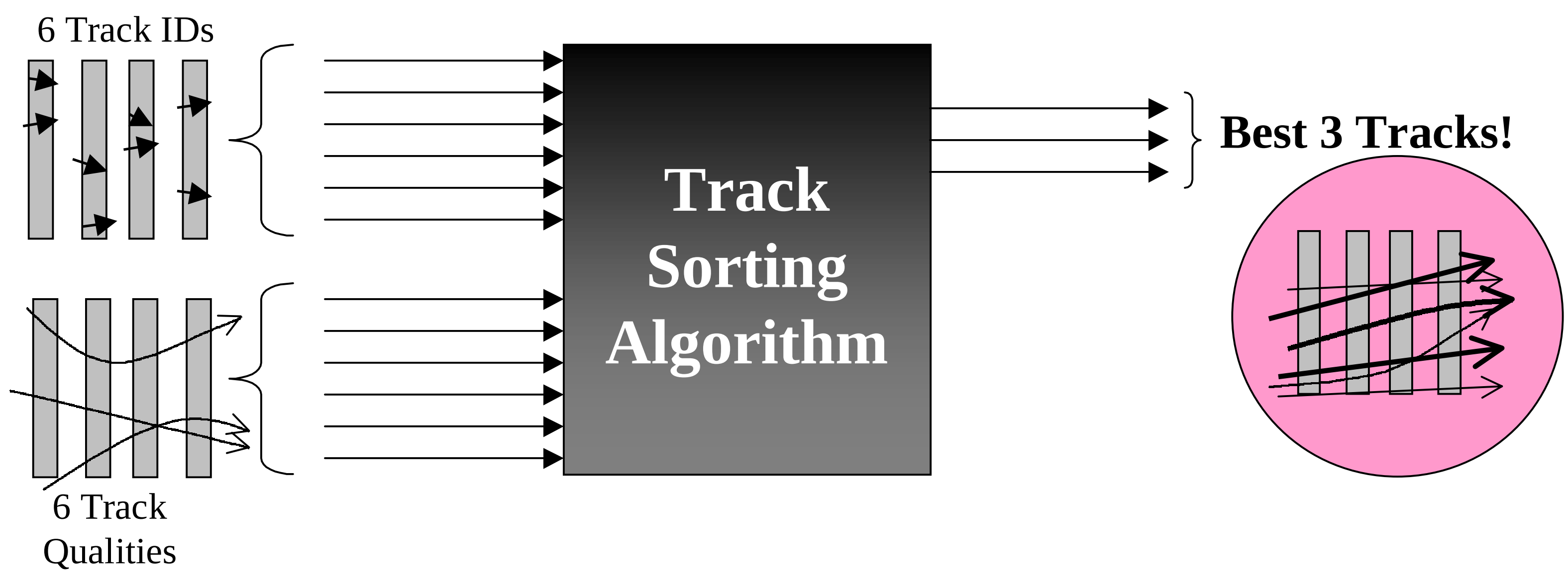
CSC Detector Logic Scheme




1 Track Finding Processor
Finds Best 3 tracks
IDs and Qualities of 6 tracks
Track Finding Logic
Track Sorter
Best 3 tracks
Quality = Goodness of track
Track ID = Stubs used to form track
Constructs 6 possible tracks
Example:
Track 1 - ok track - Quality = 5
Track 2 - not very good track - Quality = 1
Track 3 - excellent track - Quality = 10!

Best 3 tracks = 3 distinct tracks w/Highest Quality #

Must Sort Quality #s to find highest 3 => Need to use Binary Number Sorter



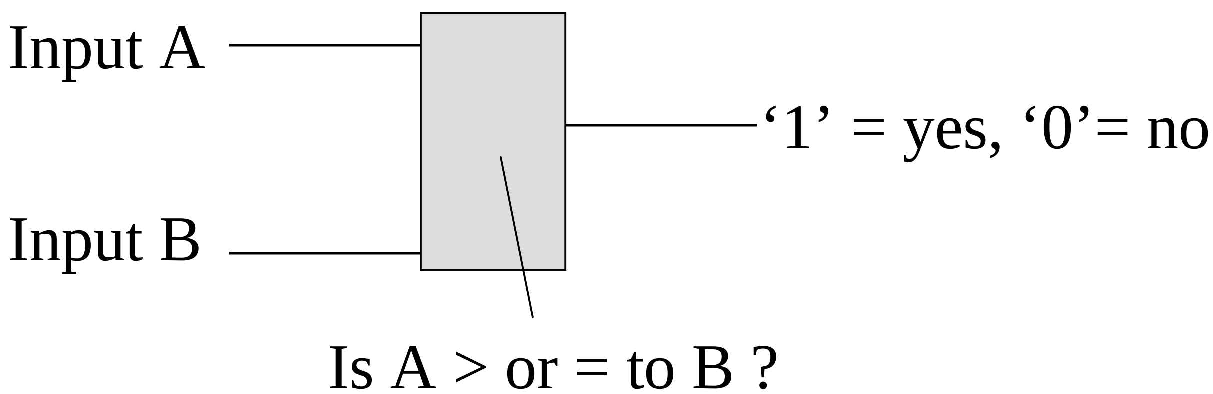
Tasks: 1) Sort these Tracks in the Shortest Time Possible

2) Make Sure These Tracks are Distinct

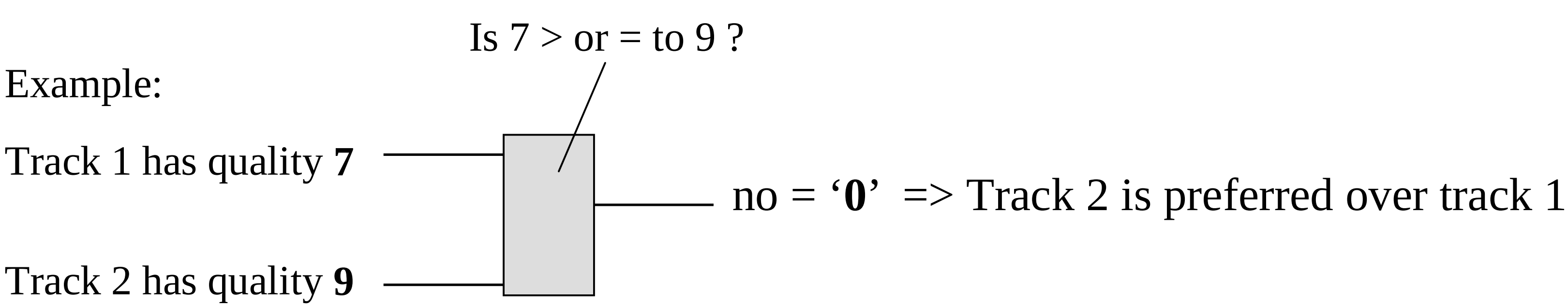
How do you Sort Tracks?

Use a number sorter to find 3 highest quality tracks

Basic Number Sorter:

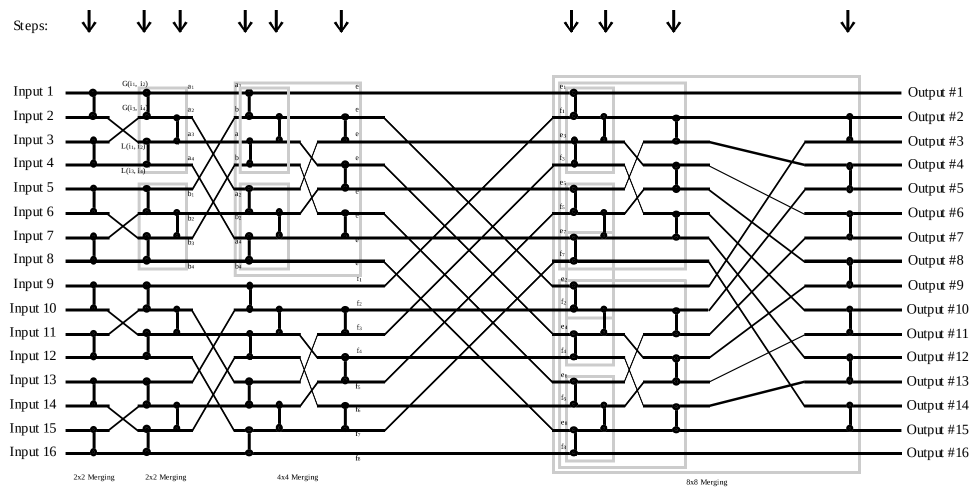


Example:



Other Sorting Methods

Arrange Basic Number Sorter into a tree to sort more numbers



For 16 inputs this needs 10 steps

For 6 inputs this needs 6 steps

Too many steps for 6 inputs
Steps increase as inputs increase

Very Bad News

There must be a better way

Use VHDL to Sort!

What is VHDL?

•(Very High Speed Integrated Circuits) Hardware Description Language

•Programming Language like C or Fortran

•Compiled by computer software like OrCAD to program chips

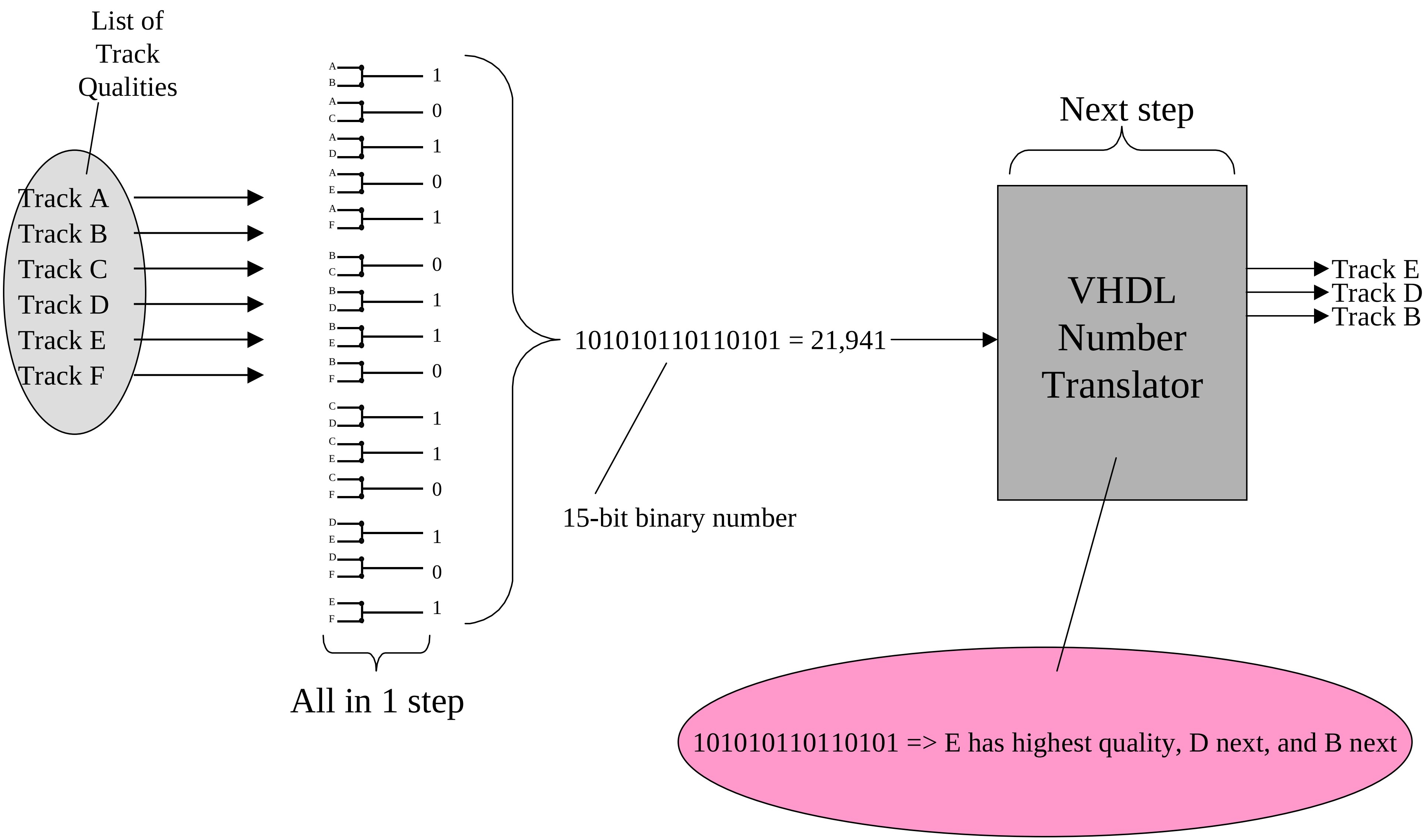
Using VHDL we can create a fast number sorter

Works in only 2 steps no matter how many inputs!

Completely flexible for any chip type and chip configuration software

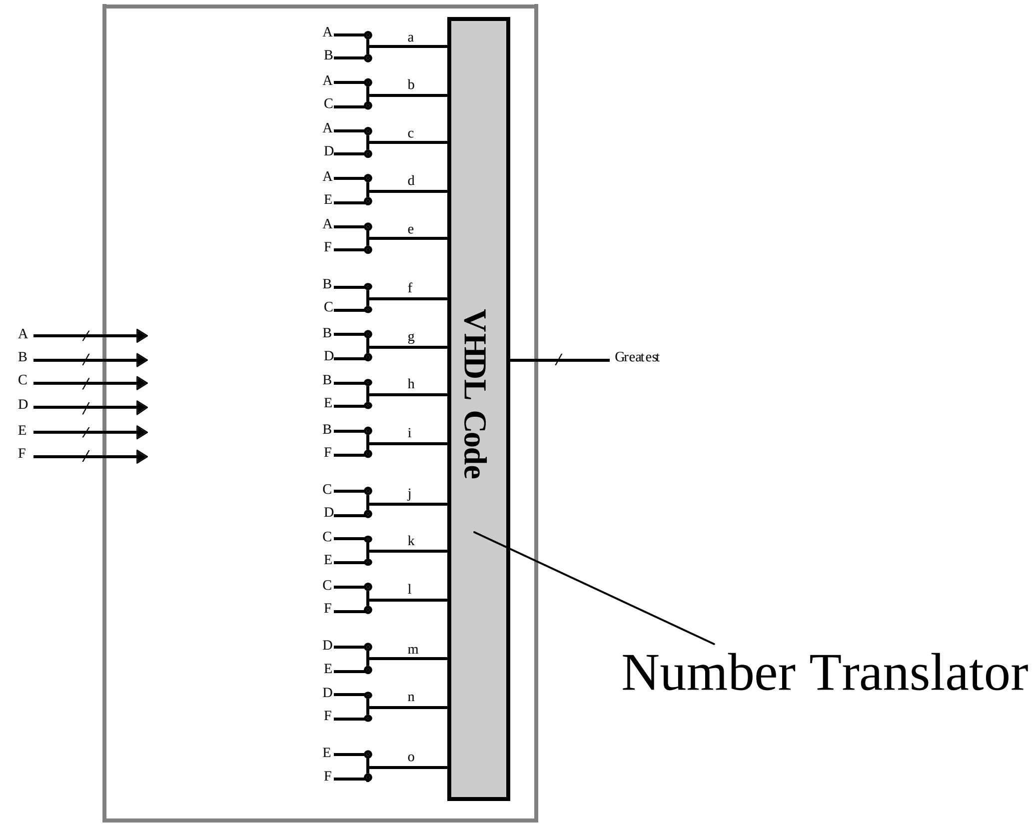
Track Sorter

How does it Work??



Example: Find Highest Value from Input List

Each 15-bit value corresponds to a unique ordering of the inputs



Bits that must be set for Track B to be the best

	compa	compb	compc	compd	compe	compf	compg	comph	compI	compj	compk	compL	compm	compn	compo
inputA	1	1	1	1	1	x	x	x	x	x	x	x	x	x	x
inputB	0	x	x	x	x	1	1	1	1	x	x	x	x	x	x
inputC	x	0	x	x	x	0	x	x	x	1	1	1	x	x	x
inputD	x	x	0	x	x	x	0	x	x	0	x	x	1	1	x
inputE	x	x	x	0	x	x	x	0	x	x	0	x	0	x	1
inputF	x	x	x	x	0	x	x	x	0	x	x	0	x	0	0

Translate Truth Table to VHDL if-then Statements

	compa	compb	compc	compd	compe	compf	compg	comph	compI	compj	compk	compL	compm	compn	compo
inputA	1	1	1	1	1	x	x	x	x	x	x	x	x	x	x
inputB	0	x	x	x	x	1	1	1	1	x	x	x	x	x	x
inputC	x	0	x	x	x	0	x	x	x	1	1	1	x	x	x
inputD	x	x	0	x	x	x	0	x	x	0	x	x	1	1	x
inputE	x	x	x	0	x	x	x	0	x	x	0	x	0	x	1
inputF	x	x	x	x	0	x	x	x	0	x	x	0	x	0	0

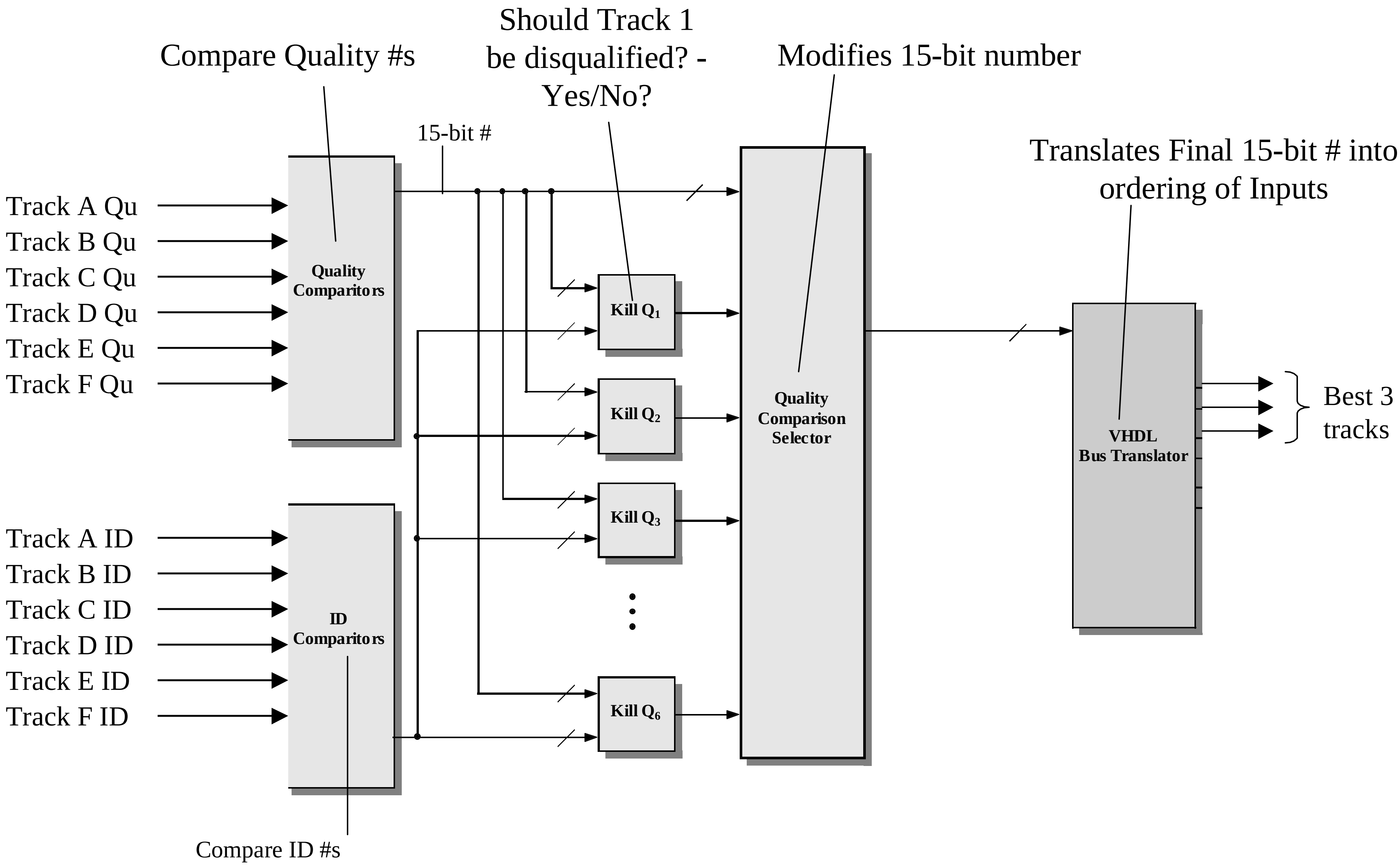
Truth Table

```
if ((compa = '1')and(compb = '1')and(compc = '1')and(compd = '1')and(compe = '1')) then
    inputA = BestTrack;
```

VHDL

Where do the ID #s Come in??

Take the same basic VHDL Sorter and **add some disqualification Logic**



Does it Work?

- Simulations show the sorting method gives correct order
- Timing Analysis show a 20 ns delay in real chip
- Finishes within the allowed time
- Help the performance of Level 1 trigger => Less memory needed to pipeline detector data
- Less money!

Acknowledgements

Kevin Ingersent and Alan Dorsey

Professor Darin Acosta - Faculty Mentor

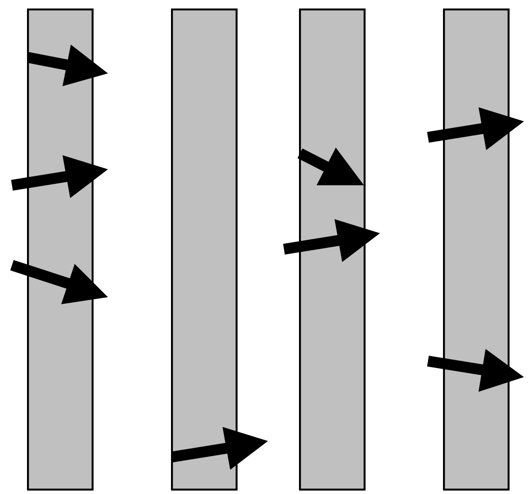
S. Ming Wang - Postdoc

Raymond Chow - undergraduate assistant

Mike Marquez - electrical engineer

1 Track Finding Processor

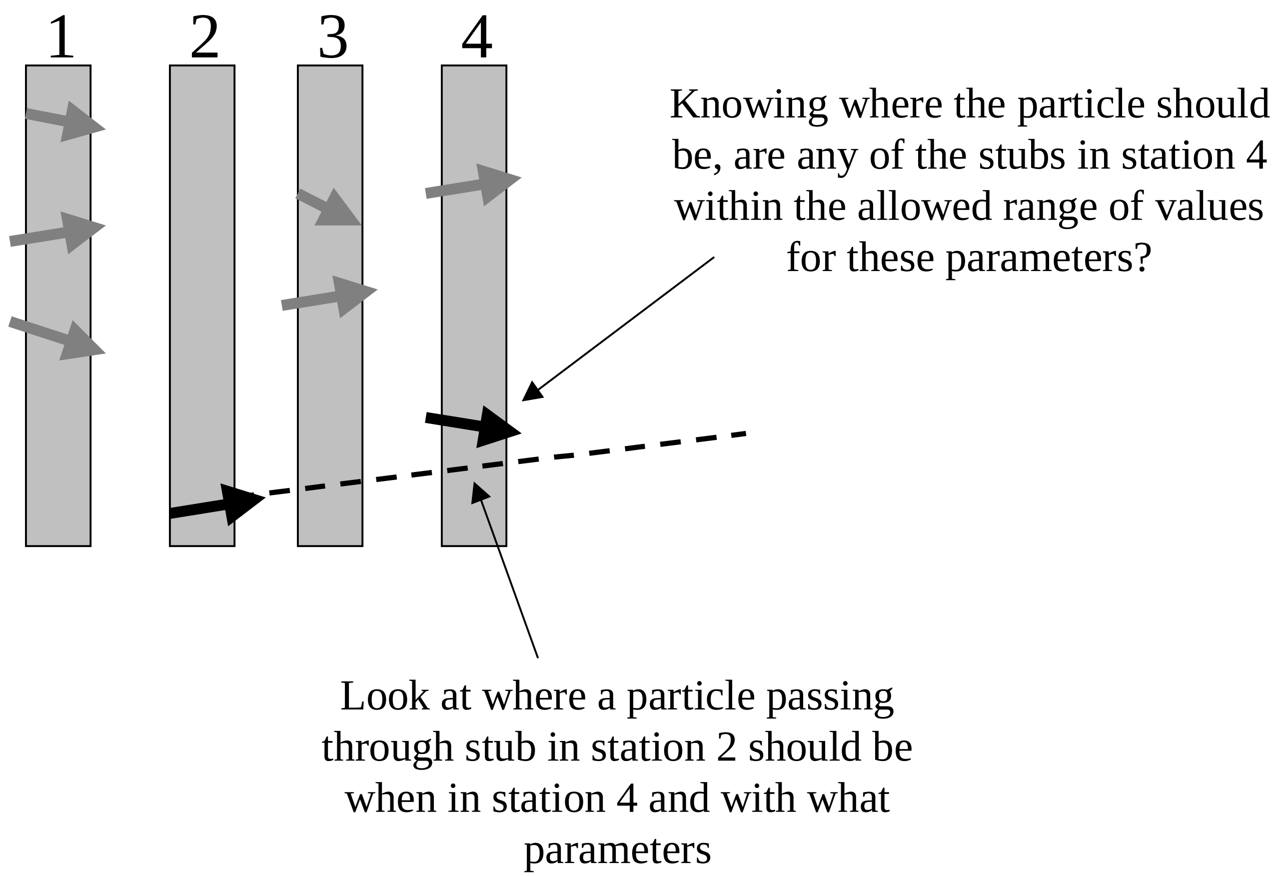
Profile View of 1- 60^{o} Sector



Real Track Stubs may actually look like this

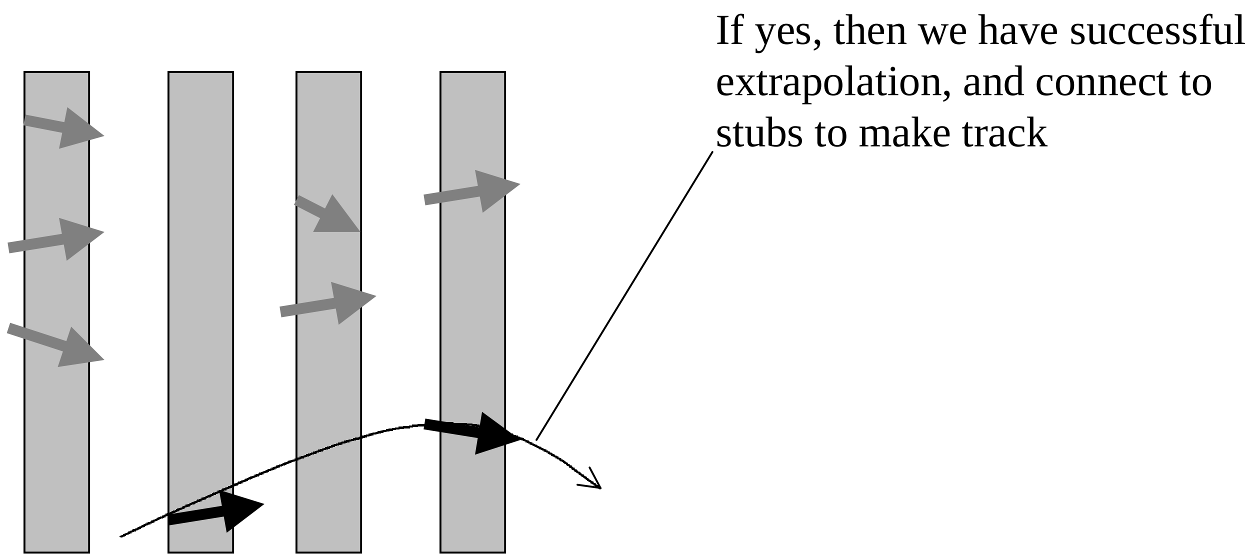
How can we possibly construct tracks from this mess!??

1 Track Finding Processor



We must extrapolate each track stub to each station

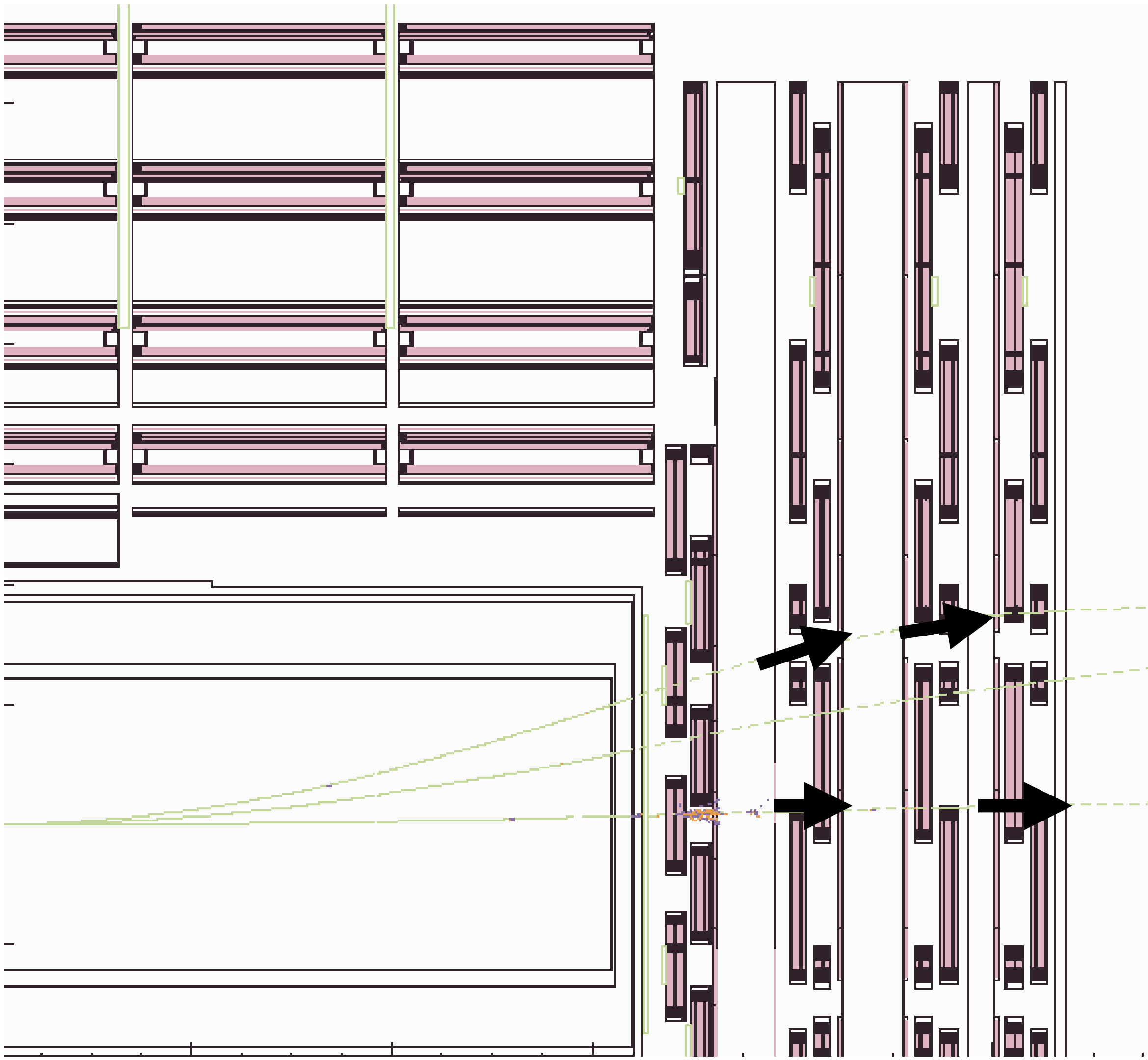
1 Track Finding Processor



After a successful extrapolation, we assign a quality value to the constructed path based on track features

1 Track Finding Processor

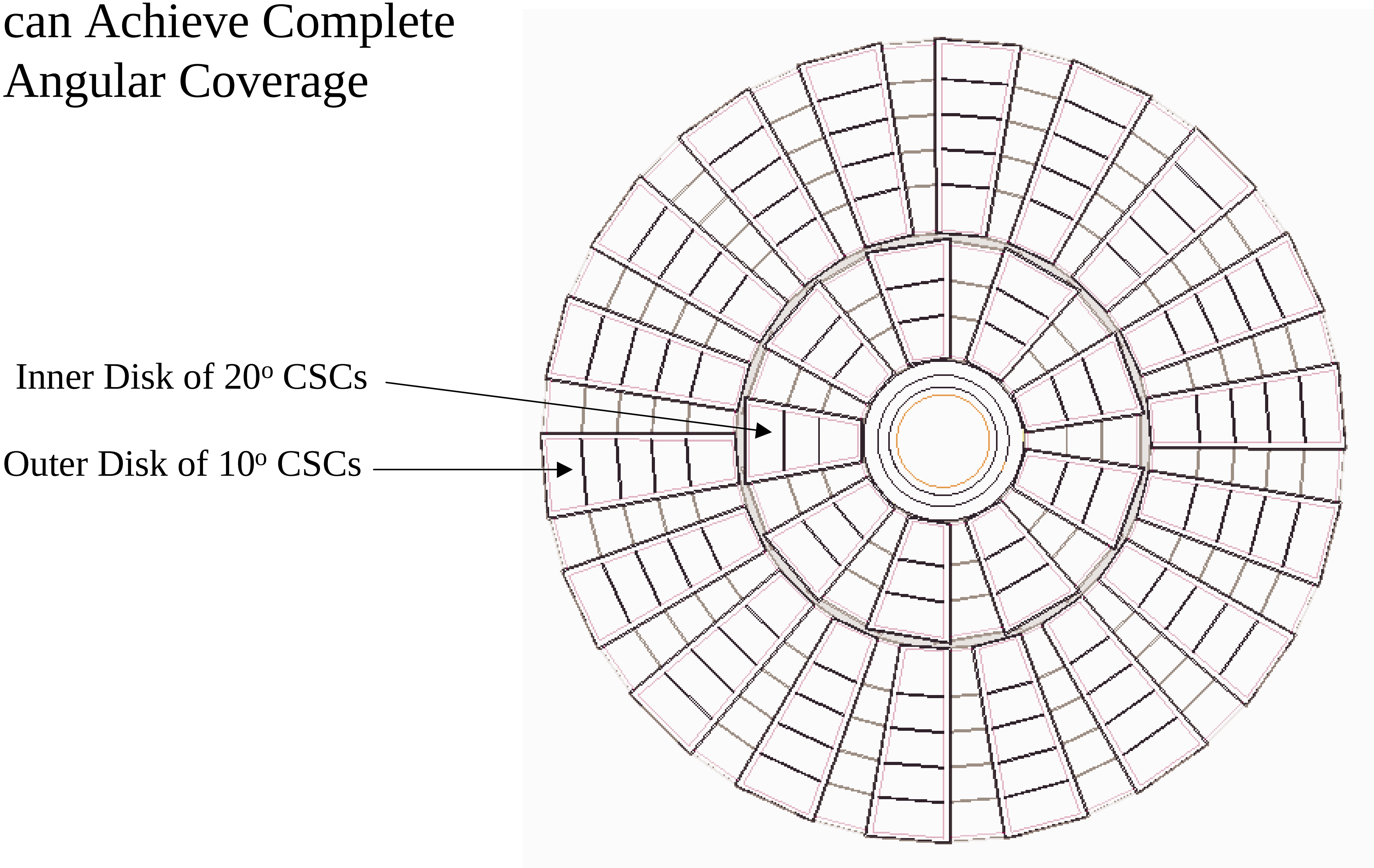
Quality is based on Momentum and Stations used to extrapolate



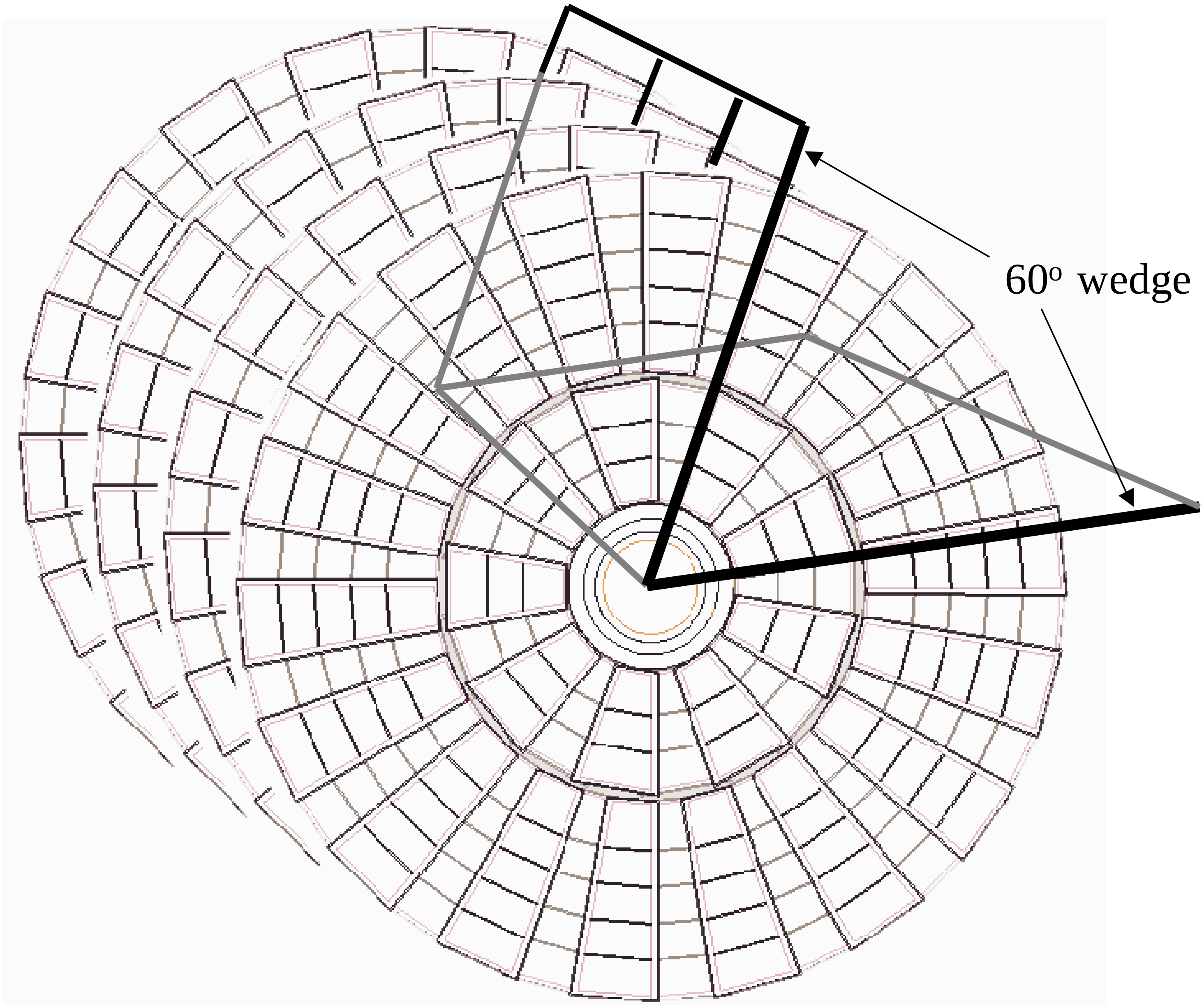
Higher Transverse Momentum => Less Bending => Higher Quality

Top View of 1 Sector

With Some Extra CSCs We can Achieve Complete Angular Coverage



1 Complete Disk (with complete Inner and Outer Disks) = 1 Station



1 Sector will contain 3 - 20° CSCs from inner disk, and 6 - 10° CSCs from outer disk