

## ***Pixel Trigger System specifications.***

The trigger system must receive hit data from the pixel detector, process the data and generate trigger decisions to go to the global trigger.

The input pixel hit data will be a stream of event packets consisting of an event number, chip identification, some status information, and the hits from that event. The hits will have row, column and amplitude information for each hit.

The output will be a message to the global trigger or DA system accepting or rejecting the event. An accepted event will also store data containing the event track segment pairs to the equivalent of a level 1 buffer. The L2 trigger system processor assigned this event will retrieve the data to provide starting points for the level 2 processing.

In the trigger system, any individual element will be processing one event at a time. However, it is not a requirement that events must be processed in sequence nor is it required that similar processing elements are all working on the same event at the same time. All the data from one event will be collected by one DSP before vertex processing.

To identify tracks, the trigger system will connect the hits across adjacent planes to create segments. The segments will have to meet criteria that will eliminate some false tracks and focus on tracks entering and leaving the detector planes. This work will be performed in a Field Programmable Logic Array (FPGA). Then the segments will be connected to create tracks in a Digital Signal Processor (DSP). Also in a DSP, all the tracks in an event will be processed to find their vertices in (or near) the collision region and the vertices will be evaluated to look for vertices that are distinctly separated but within a small number of millimeters apart. This is the primary criterion for a trigger output.

The trigger systems can be split into two hardware regimes. The first processing that produces segments must process almost all the data and can be characterized as repetitive and time critical. This will be done with Field Programmable Logic Arrays (FPGA) and some supporting digital logic. There will be some monitoring logic to implement error reporting and a simple control system. The segments will pass to Digital Signal Processors (DSP) to be made into tracks, project vertices and analyze separation of the vertices. This processing must be more flexible

The FPGA processing can be split into pixel specific tasks in the Pixel Processor and segment tasks in the FPGA Tracker.

The Pixel Processor will perform the following tasks:

1. Group the pixel hits by event and expand the time data field to expand the range.
2. Buffer the events until all the hits have been read out of the detector.
3. On a separate data stream, buffer all the raw hit information into a large buffer until a level one decision is made on each event and provide data to the rest of the data acquisition system if it is accepted.
4. On the trigger data stream the hits are passed through the cluster logic which replaces sequential hits in the same column (sequential row numbers) with a single hit in the middle of those hits. This is probably the place where the hit analog information is replaced by tags used to pass additional information about the hits. One tag bit will be

to mark a cluster larger than the largest expected size or to encode the cluster size if that helps later processing.

5. Next on the trigger data stream the hit position information must be translated from the chip, row and column raw data to x-y positions that will allow processing hits across planes.

The FPGA Tracker will perform the following tasks:

1. Connect hits in each y plane to hits in the immediate next y plane. In all but the end planes, the starting hits must lie in one of two seed regions. The seed regions are the detector area immediately surrounding the beam hole and the detector area at the outermost edges of the planes. Thus segments created in the tracker will either be the part of the track entering the detector region or the part where it exits the detector.
2. Connect a third y hit to the previous track.
3. Confirm the segment with two of the three corresponding x plane hits.
4. Pass the resultant segments to the DSP processor.

## Detector description

The detector has 31 stations each consisting of two planes of detector. The pixels are rectangles 50 um by 400 um so the pixel position information has a higher resolution in one dimension than the other. One plane will have the narrow pixel dimension in the x (horizontal) direction and the other plane will have higher resolution in the y (vertical) direction. The pixel detector is in a magnetic field that will bend charged particles in the vertical or y plane. Each plane will have an active area approximately 10 cm by 10 cm with an  $X$  by  $X$  hole in the center to allow the beam through. The active area will be made up of shingles approximately  $X$  by  $X$  cm. Each shingle will provide a data stream output on a LVDS cable but the shingle with sensors near the beam region will have much more data than the shingle near the outer edge of the plane. The shingle data streams from a plane will be combined close to the detector before the data is serialized onto fiber optic links for the trip from the collision hall up to the counting room. This will allow balancing data flows through the many links carrying the data.

## Pixel Processor

The data into the pixel processor will be received on fiber optic cables from the collision hall. The fibers will carry data from one plane [half plane, quadrant] and the data is not guaranteed to be in event time sequence. The data rate from one pixel plane will average  $X$  gigabytes per second based on 10 hits per plane, three pixels per hit, four bytes of data per pixel and a 7.6 MHz beam crossing rate. [correct the numbers] As the data flows in it will be stored in a group of buffers. Data from the same event will be placed in the same buffer. Circuitry will monitor data into a buffer and time out either after a fixed time following the first data arrived or after a fixed time break in the data flow. When the buffer of data times out the data is pushed through the next steps to the FPGA tracker assigned this event. It is also pushed into Level 1 data acquisition system buffers to be held until requested by the DAQ system or specifically rejected.

Comparing the hit column and row numbers and accumulating sequential hits in a shift register performs cluster finding. Individual hits are passed right through and hits with the same column number and sequential row numbers are put into a shift register. The number of hits in the register determines the offset that is used to make a mean row value for the one data hit that is passed to the next step. This data moves to the input buffer of the FPGA Tracker.

## **FPGA Tracker.**

A segment finder will take hits from one event from three stations, six planes, of pixels. The data from one of the end Y planes will be the seed hits. The segment finding will first load hits from the Y plane closest to the seed plane into a content addressable memory (CAM). It will then select hits from the seed plane that are in the seed region and, one by one, look for matches in the CAM. A match will be any hits that have x/y positions that are within a fixed range window of the seed hit. The window will limit matches to hits that have a high likelihood of being part of an interesting track. These matching hit pairs will be passed to the next section of logic that has loaded the third Y plane of hit data into a CAM. The matched pairs will be projected into the third plane and the CAM searched for hits that match that predicted x/y position. Successful matches are then used to similarly search for the X plane hits that fit the developing segment. A good segment will have two of the three X hits. Good segments are passed on to the DSP processing. Checking circuitry early in the logic will confirm that all the hit data is from the same event. If not, an error message is passed to the Trigger Control and Monitoring (PTCM) processor. Granularity of the FPGA tracker could be eighth, quarter, or half planes. There will also be a temporal multiplexing, in that (four, eight) banks of FPGA trackers will rotate processing events to allow the time necessary to do the segment finding.

## **DSP.**

DSP process control will include messages to the DSPs defining which events they will process. This will allow error checking to look for lost or misrouted data. Then when an event times out and is started into the FPGAs, a message can be sent to enable processing on the DSP. All event numbers need to be accounted for by the DSPs. The monitor processor should look for messages from each DSP on a regular basis. The DSP application software will be controlled by a real time operating system that will control the application software. Applications will be track and vertex processing, command and error message processing, and diagnostics and testing.

### **Misc Notes and information.**

- All messages will include the event number as a time tag.
- L2 software is Paul. L2 hardware is?
- Data flow out of pixel FEBs -- with  $n$  fiber ports, port one gets event 1,  $n+1$ ,  $2n+1$ , etc. Port 2 gets events 2,  $n+2$ ,  $2n+2$  etc.
- Pixel chips are capable of reading out about 5 hits per BCO.
- Pixel chips have 16 rows X 180 cols or 2880 pixels.
- Data from detector is 16 Gbits per sec per plane
- Data from whole detector 62 planes is 1 Gbit per millisecond.
- Diagram of rows and columns. Rows are the long pixel dimension and columns are the short dimension. So the clustering will be more prevalent along neighbor rows in a column.