DØ/TRIGGER/3684
September 10, 1999

The Level 2 STT Trigger Simulator and its Clustering Algorithm

Silvia Tentindo Repond
Florida State University

Abstract

This note describes the Simulator of the Level 2 Silicon Tracking Trigger detector (L2STT) as a component of the D0 RunII Trigger Simulator. After a brief descroption of its design and implementation, it describes the Clustering algorithm that creates clusters from the SMT hits in input to L2STT, and includes preliminary results of MonteCarlo studies.

1 *Introduction*

The L2STT Simulator is a component of the D0 RunII Trigger Simulator. Section 2 briefly describes the D0RunII trigger Simulator. Sections 3 and 4 talk about the design and the structure of the L2STT simulation package ('tsim_L2STT'). Section 5 and 6 describe one of the possible implementation of hit clustering,gives a concise scheme of the logical steps constituting the algorithm, and outlines a brief list of its possible advantages. Section 7 presents the results of the Clustring Algorithm applied to MonteCarlo data (single muons, ttbar with no minimum bias and ttbar with 2 minimum bias events). Specifically Section 8 studies the depen dency of the Clustering efficiency as a function of SMT layer and Section 9 studies threshold dependency. Section 9 mentions studies at present in development about cluster centroid position resolutions.

2 *The D0 RunII Trigger Simulator*

The STT Trigger Simulator (tsim_L2STT package) is a component of the RunII DØ Trigger Simulator whose goal is to read real or MonteCarlo data and to simulate the various components of the Level 1, 2 and 3 trigger. The main task of the Trigger Simulator is to study and help optimize the trigger. More specifically, the task of the D0 RunII Trigger Simulator is to simulate the input/output of each module and its functionality. Also, a single module is tested for its internal functionality, so that the input/outputs of its internal components are optimized.

Typical components for the D0 RunII Trigger Simulator are L1CTT, L1CAL, L1MUON, for the Level 1 trigger, L2CAL, L2MUON, L2STT, L2CTT and L2Global for the level 2 trigger, and L3 for the level 3 trigger.

Each trigger component is modeled by a 'package' and by a 'chunk'. The package is the set of methods that simulate the hardware processors and the algorythms they implement, the chunk is the object that contains the properties (as results of calculations and algorithms) of the trigger component. For example, for the L2STT processor, there is a package (tsim_L2STT) and a chunk (L2STT).

The simulation framework is simplified by the existence of two utili ties, called the Unpacker - that converts the raw data (Raw Data Chunk) into pieces (Unpacked Data Chunk), one for each detector or trigger component, and the Packer - that takes a given Unpacked Data Chunk and converts it into a Raw Data Chunk -.

3 *Brief description of the design for tsim_L2STT.*

Tasks of the L2STT Simulator 'tsim_L2STT' are:

- A) to provide a separate framework package that simulates the STT L2 preprocessor, written in C++, that operates entirely within the RunII software environment.
- B) to simulate all the components of the STT preprocessor:
  1. hit clustering in the Silicon Trigger Card (STC)
  2. hit filtering in STC (This includes road matching in the Fiber Road Card (FRC))
  3. tracking in the Track Fit Card (TFC)

4. vertexing in the Vertex Card (VTC)

- C) to format output to the L2 CTT processor and to L3
- D) to optimize monitoring parameters, strategies and algorithms
- E) to provide efficiencies for STT in different configurations and for different input conditions.

At least two clustering algorithms have been studied. In this note we will address the algorithm that is currently implemented and some initial results obtained by running that algorythm on MonteCarlo data.

4 *The tsim_L2STT package.*

The tsim_L2STT Package is made of Classes, each modeling a hardware components: the STC is described by L2STT_Cluster Class and a L2STT_Clusters_Road Class; the FRC is described by a L2STT_Road Class, the TFC is de scribed by L2STT_Cluster_on_Track and L2STT_Track_Fit Classes, and the VTC is described by the L2STT_Vertex Class. The components of the tsim_L2STT Package are interfaced to the other packages of the D0 trigger Simulator by a main tsim_L2STT package, that steers and activates the various components. In the steering package there are methods to read the inputs needed to STT, in particular the SMT (Silicon Microstrip Detector) chunk ('getSMTFrontEnd'), and the CTT L1 (Central Fiber Tracker Detector) chunk ('getFT_L1'). The steering package finally provides the method to create the STT output chunk (L2STT_fill_Chunk) where the results of the STT processors are stored, and ready to br passed to the L2 CTT processor, and also provides the resulting data in an Ntuple by using the general trigger package (TrigNtuple).

5 *The hit clustering in tsim_L2STT*

The cluster algorithm implemented into the STT Simulator has been designed taking into account the requirements of the hardware component in tehg Silicon Trigger Card that is going to perform the clustering (FPGA). More than one algorithm is envisaged, and the simulator will help to compare them and to optimize the final implementation. In this note we will refer to 'Algorithm I' as the one currently implemented in tsim_L2STT. Design studies for the clustering 'Algorithm I' have been initiated (Note 2-15-99, S.Tentindo Repond) taking into consideration all the work previously done on the subject. Since none of the existing proposals had satisfied the constraint of being 'on-the-fly', a modified algorithm had been designed. The requirement of being 'on-the-fly' means that the calculation must be done before the next SMT hit arrives to the STC (this means 2 clock periods or 37.6 nsec i.e. the time required to read two 8-bit words from the SVX chip of SMT).

The centroid (center of gravity) of a cluster of N hits, each of which is specified by a Channel Content (ADC_i) and a Channel Address (ID_i) is given by

```
                  sum_i ( ADC_i * ID_i )
CENTROID =    -----------------------          i = 1,N     (1.1)

                    sum_i ( ADC_i)
```

In the present 'Algorithm I', the procedure is the following:

- 1- read first hit (channel address and channel content)
- 2 -subtract pedestal from channel content
- 3 - if channel content - threshold bigger than 0 then:

```
store the partial quantities:
           nominator = sum_i (ADC_i * ID_i)
           denominator = sum_i (ADC_i)             (1.2)

where here obviously i=1
```

- 4 - read the next hit : check if it is a 'neighbohr' hit:

```
( i.e. if ID_i = ID_(i-1) )

4.1 if yes, set flag ‘HIT_IN_CLUSTER’
        and calculate the quantities above (1.2) and go to
        next hit
4.2 if not, set flag 'END_OF_CLUSTER'
        and calculate the (1.1)
        CENTROID
        save int (CENTROID)
        calculate OFFSET in 1/4 strip units
        reset cluster_size
        reset cluster_pulse_height
        store Cluster's attributes into class
        L2STT_Clusters
```

```
--------------------------------------------------
(Cluster attributes:
  cluster number
  hardware address
  centroid hadware address ( Channel ID + offset )
  cluster size
  cluster pulseheight)
--------------------------------------------------
```

- 5 - for each next hit, proceed as in 4.1. or as in 4.2 as needed

```
Remarks .

      In this simple algorithm, based on checking on the
      'neighbohr' hit, there is no fixed size for the cluster.
```

```
This means that a cluster can be as many hits as needed;
it also means that a smaller cluster will require less space;
it also means that there will not be hits belonging to
two different clusters in the centroid calculation.

A maximum cluster size should be imposed nevertheless  [ it is
senseless to calculate the centroid for a too wide cluster
like the one generated by delta ray or by a quasi-parallel
track ].

Another point is that the DIVISION needed by the centroid
formula is not required for each clock period of 37.6 ns,
but only for the hit that is the 'END_OF_CLUSTER'.
It could be possible to use the FPGA for other operations
during the previous clock cycles (to read out calibration
constants or to download lookup tables,etc.).

The DIVISION can be done by a lookup table.
A lookup table that doesn't require big memory space,
like the one proposed in D0Note...... would be applicable to
this Algorithm I. Even though such lookup table requires
a 'central' address for the cluster where the maximum ADC is
found, it simple to aknowledge that the same lookup table
would work in case that a 'side' address [ the first hit
of the cluster ] would be substituted to the central hit as
'address zero'.
```

6 *Remarks about Clustering Algorithm I.*

The Algorithm I as implemented in the tsim_L2STT Simulator, has also been implemented in the L3 software, as the SMT Clustering L3. Compared to the D0Reco SMT Clustering Algorithm it is much simpler and, in particular, satisfies the trigger timing requi rements. To provide the L2 clustering to the L3 clustering seems reasonable because L2 and L3 have very similar requirements. Nevertheless the L2 clustering could be taken as a 'simpler' algorithm, and some more sofisticated version of clustering, like the one in DF0Reco, could be reserved in L3 to those more rare and more complicated clusters (double peaks etc).

7 *Clustering with MonteCarlo data.*

The tsim_L2STT package has been run in its minimal configu- ration, i.e. with only the Cluster finder activated. Input MonteCarlo files contain 50 Gev muons single tracks, with t tbar events (no minimum bias) and t tbar events (2 minimum bias). Table I summarizes these studies. nc is the average number of reconstructed clusters per event. size is the average size of cluster, and ph is the average pulse height of the cluster. A cut has been set for a maximum of 2000 for nc, 40 for cluster size and 250 for pulseheight. The ttbar + 2 minimum bias sample shows an increase in number of hits (and reconstructed clusters) by about a factor 2 relative to the ttbar + zero minimum bias sample. Also cluster size (from 3.2 hits per cluster to 3.9) and cluster pulse height (from 55. ADC counts to 69.) increase.

TABLE I

	t tbar + 0 min bias	t tbar + 2 min bias
$\overline{\text{nc}}$/evt	547	1092
$\overline{\text{size}}$	3.2	3.9
$\overline{\text{ph}}$	55	65
LAYER 1		
nc/evt	548	1091
$\overline{\text{size}}$	3.8	4.9
$\overline{\text{ph}}$	59	>57
LAYER 8		
nc/evt	543	1092
$\overline{\text{size}}$	3.0	3.3
$\overline{\text{ph}}$	52	65

8 *Clustering efficiency vs SMT layer.*

A specific concern is the clustering as a function of layers (LAYER 1) where the density of hits could create contiguous clusters. Table 2 shows values of nc, size and ph wrt layer number (LAYER 1 to LAYER 8) for the sample of ttbar+0 min bias and for the sample of ttbat+2 min bias. As a general remark, the size of cluster as reconstructed by the algorithm decreases with increasing Layer number (from 3.6 hits/ cluster in LAYER 1 it goes to 2.9 hits/cluster in LAYER 8). ph goes from 59 ADC counts in LAYER 1 to 52 ADC counts in LAYER 8.

Still from table 2, in case of ttbar+2min bias, the size goes from 4.3 hits/cluster in LAYER 1 to 3.2 hits/cluster in LAYER 8. ph goes from 75 ADC counts fro LAYER 1 to 65 ADC counts for LAYER 8.

TABLE 2

t tbar + 0 min bias

	ALL BARREL	Layer 1	2	3	4	5	6	7	8
nc/evt	547	548	555	545	545	550	551	541	543
size	3.1	3.6	3.5	2.9	3.1	3.0	3.1	3.1	2.9
ph	55	59	60	51	56	55	53	55	52

t tbar + 2 min bias

	ALL BARREL	Layer 1	2	3	4	5	6	7	8
nc/evt	1092	1091	1092	1098	1083	1091	1094	1089	1092
size	3.6	4.3	4.2	3.1	3.4	3.7	3.8	3.4	3.2
ph	69	75	58	63	72	54	69	50	65

9 *Clustering efficiency vs channel threshold.*

The previous values have been calculated by setting a channel threshold to 3 ADC channels.

Table 3 compares results of the clustering in case the threshold is set ot zero, for the ttbar+0 mun bias sample.

TABLE 3

	t tbar + 0 min bias threshold = 3 ADC	t tbar + 0 min bias threshold = 0 ADC
$\overline{\text{nc}}$/evt	547	612
$\overline{\text{size}}$	3.2	3.2
$\overline{\text{ph}}$	55	59
LAYER 1		
nc/evt	548	614
$\overline{\text{size}}$	3.8	3.7
$\overline{\text{ph}}$	59	>56
LAYER 8		
nc/evt	543	608
$\overline{\text{size}}$	3.0	3.0
$\overline{\text{ph}}$	52	49

10 *Conclusions.*

Clustering studies have been initiated in the framework of the STT L2 Trigger Simulator. New Algorithms can be implemented in the same framework with minor modifiactions. Improvement in the code that analyzes the Montecarlo SMT hits and compares them to the reconstructed clusters is needed and on way of development.