# Fast Track Trigger L1 Algorithm Implementation

**Scheme proposed by A. Baird, D. Sankey**
**after discussions in Birmingham 18 Feb 2000**
**Notes (and misconceptions) by P. Newman**

## 1 Introduction

This is a very rough attempt to describe the FTT L1 algorithm on the basis of diagrmas produced by Dave and Adam and some notes from Dave. - It surely needs lots of refinement and considerably more detail before it can be called a design! The aim here is to provide a starting point from which discussion can follow at the next meeting (21st March).

## 2 Overview

Figure 1 shows a basic overview of the proposed system and its interconnections (thanks Yves!). The timing output from the $Q - t$ feeds the segment finding shift registers. The integrated charge information associated with each hit (leading to the $z$ coordinate) is buffered to be matched with segments later and fed to L2.

After a hit in the pivot element of the shift registers, the register contents are interrogated (CAMs?). Successfully found track segments are passed to the L1 trigger logic and also to the L2 pattern finding buffer. Here, valid patterns are stored, with the following information ...

- Time at which the pivot element was hit (20 MHz precision) (5-6 bits)
- Fine pattern of this segment (80 MHz precision) (12 bit)
- Pattern number (? bit)
- Cell number (5 bit)

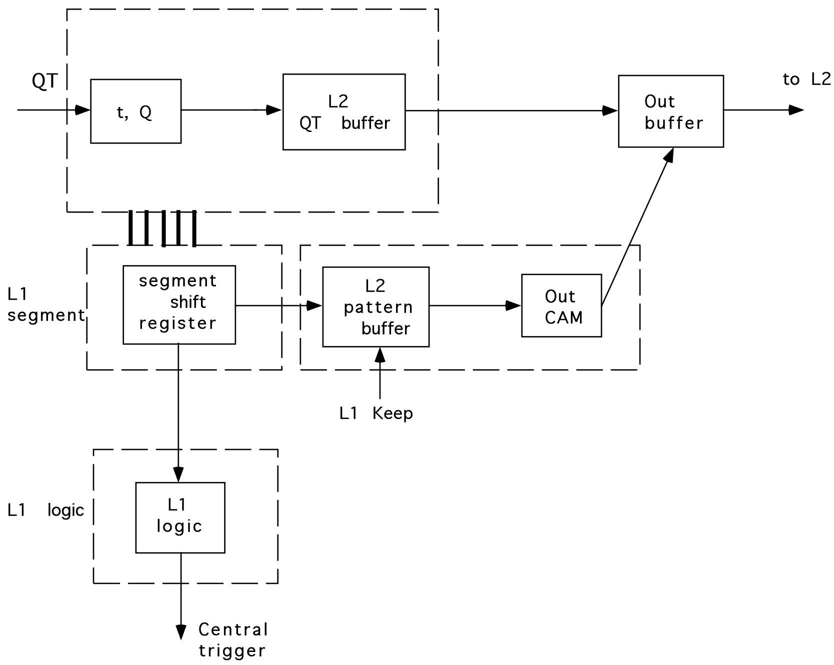


Figure 1: System Overview

A L1 keep signal from the central trigger activates the large output CAM, which gives accurate $\kappa$-$\phi$ information as a function of $t_{L1Keep} - t_{pivot}$, the time difference between the pivot element activation and the L1Keep. The fine (80 MHz) pattern, corrected in $\phi$ for cell number and the pattern number are given. The information from the large CAM has to be indexed by $t_{pivot}$ and cell number.

# 3 $Q - t$

An overview of the proposed $Q - t$ algorithm is shown in figure 2, which shows the manipulations performed on a set of three wires forming a trigger group. - Each wire has two inputs to the $Q - t$ algorithm, corresponding to the two wire ends.

Information from both wire ends is used in hit detection, from which the timing information is obtained. The integrated charge calculations from both ends of each CJC wire are used to obtain the $z$ information. In the

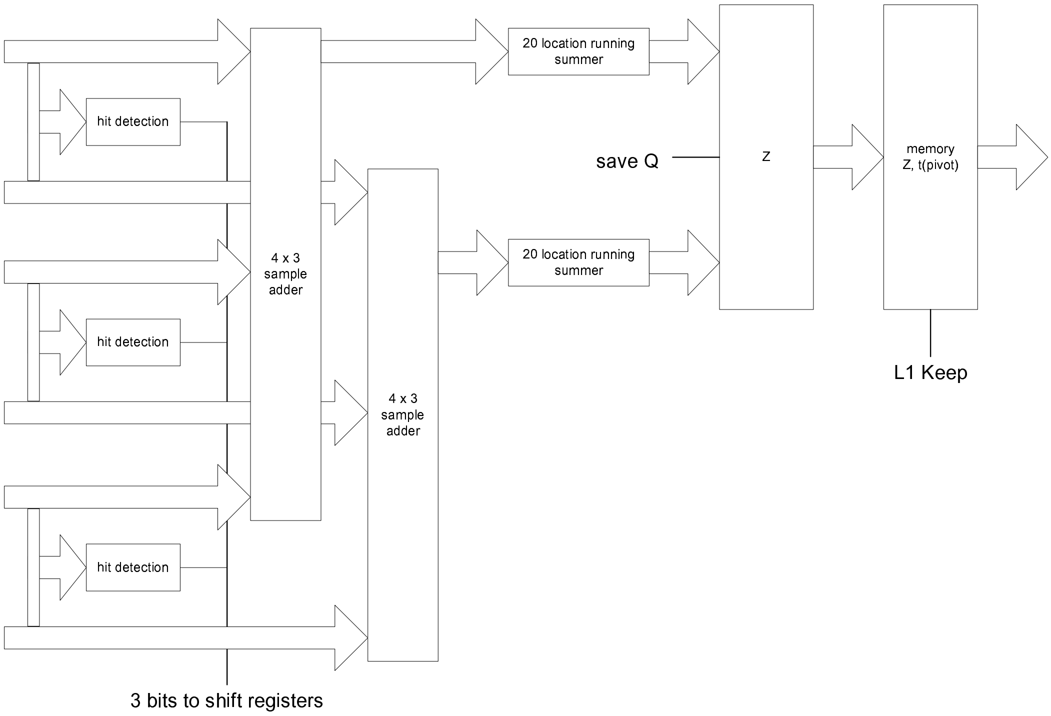


Figure 2: $Q - t$ Overview

example shown in figure 2, the $z$ coordinate of the segment is obtained by averaging over the three wires of a trigger group. The information from the left hand end of each of the three wires is summed (4x3 sample adder - four fold to convert between 20 MHz and 80 MHz) and similarly the right hand ends of the wires. The information from successive digitisations is summed using a sliding charge integration window centred in time on the hit in the middle of the three wires (i.e. the pivot layer). A valid segment match found in the shift registers / CAMs triggers the passing of the summed charge information to the '$z$' step. Here a $z$ coordinate for the segment is obtained by the standard charge division technique. The $z$ and $t_{pivot}$ information are then buffered until a L1Keep signal arrives.

Optionally, we could also latch on the hit in the centre wire, as is done in the segment finding algorithm (pivot layer). This would make the $z$ calculation fully independent of the segment finding.

# 4 L1 Segment Finding

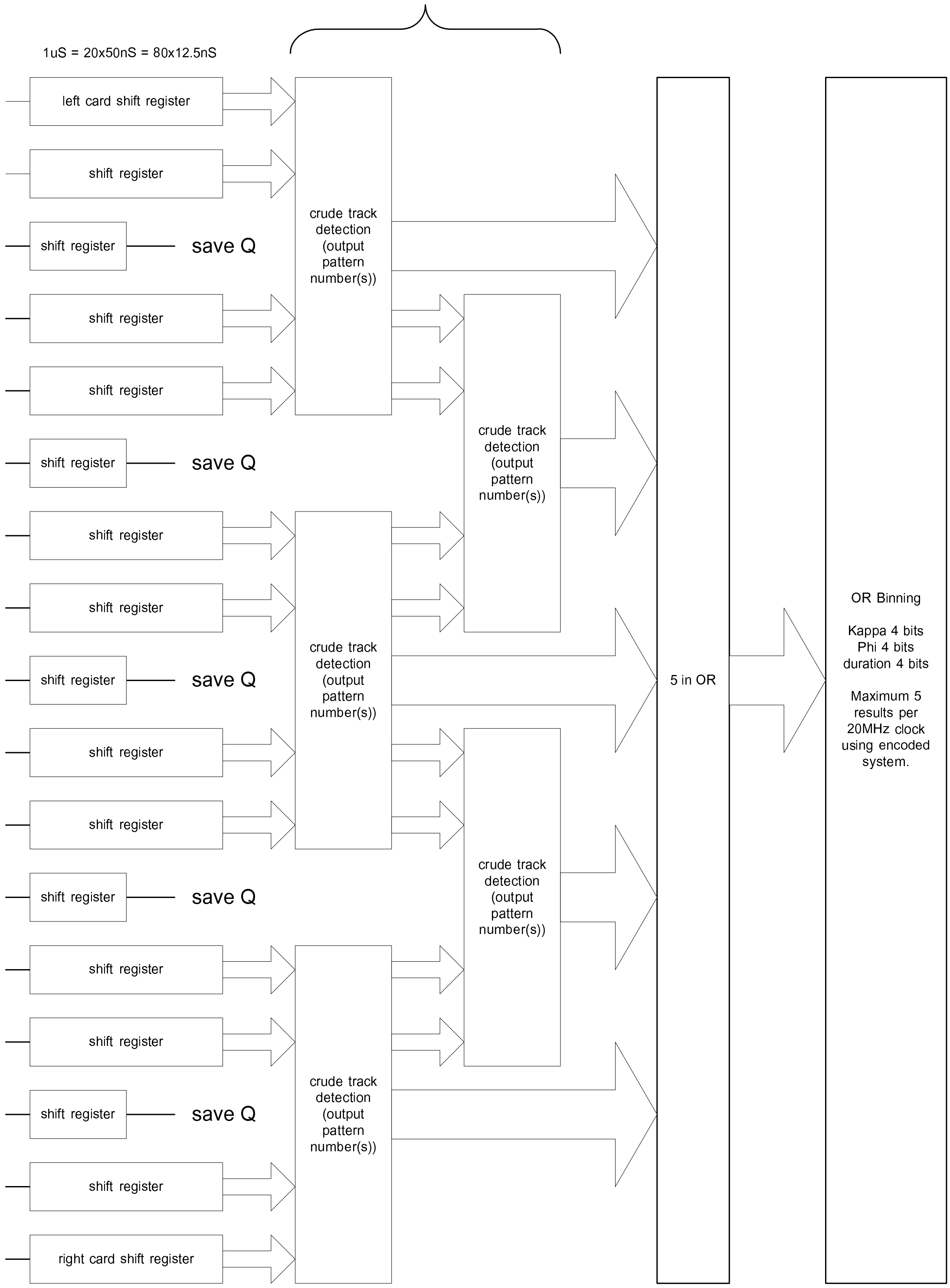

Figure 3: Segment Finding Overview

Figure 3 summarises the segment finding algorithm for a complete front end module (5 groups of 3 wires, plus external connections to neighbouring cards to deal with cases where tracks pass cell boundaries at the edge of the space covered by a card. The 'save $Q$' label feeds back in as shown in figure 2. The shift registers containing the identified hits feed into the first level (20 MHz) segment finding. This is done in 2 stages to ensure that the correlations between neighbouring cells covered by the same card are dealt with. The output from the segment finding is ORed together and encoded as shown in figure 3 for output to the L2 pattern buffer and the L1 rtigger card (see figure 1).

A nice simplification to the segment finding is possible, provided all wire groups in one card are from the same radial layer. Under these circumstances, the valid masks for each group of three wires are identical. The corresponding $\kappa$ and $\phi$ values differ only by the addition of a multiple of the cell number to the $\phi$ coordinate. It therefore seems reasonable to multiplex the input to the track segment finding. The precise details of how that could work need to be defined.

# 5 Trigger Logic

Figure 4 gives an overview of the suggested logic leading to the L1 trigger signal. This is presumably implemented in a single CAM(?) and all sits on the trigger card / service module. The input data from the Front End Module is the list of linked track segments and the range of validity in terms of bunch crossings. The array driver clocks out the ($\kappa$,$\phi$) results for each bunch crossing. This information is ORed within one layer and then summed with the other layers at different radii to look for coincidences (cluster detector). The output from the cluster finding is then fed to the combinatorial logic for the trigger.

Recall that the granularity is likely to be no better than 64 x 16 in ($\kappa$,$\phi$), such that the $\phi$ resolution is half a cell. For a given segment, the $\phi$ and $\kappa$ values therefore do not change over the range of bunch crossings for which they are valid. It is probably possible to have completely independent histograms (e.g. 11 x 16 in $\kappa$,$\phi$) for different regions of $\phi$.

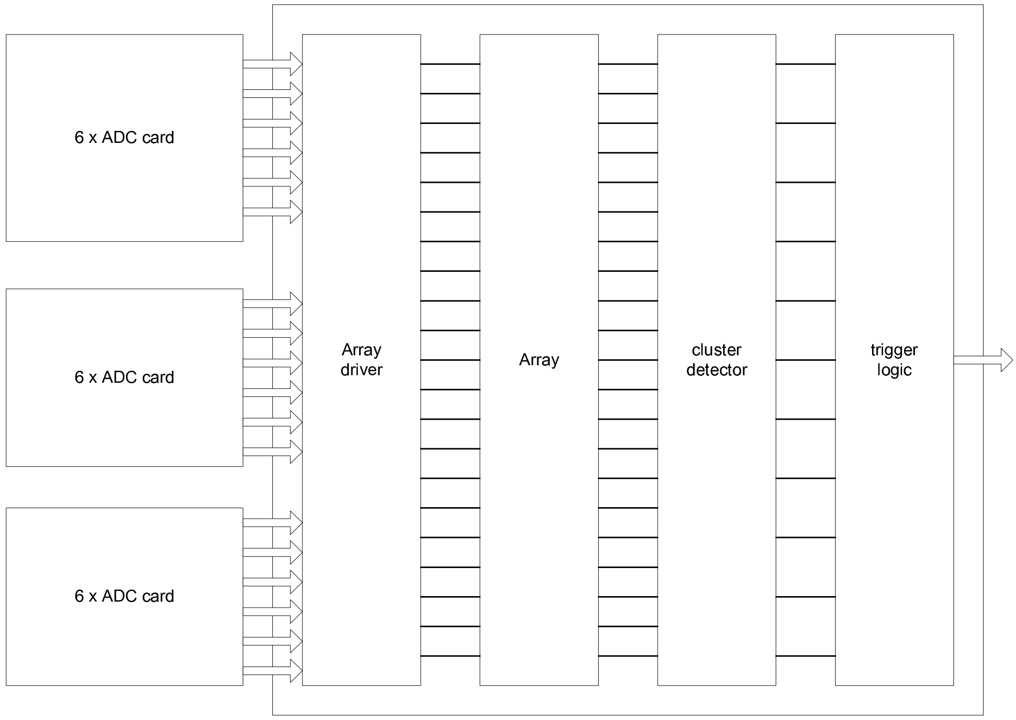


Figure 4: Trigger Logic Overview