H1-09/99-576
PRC 99/06,07

Level 1 Fast Track Trigger Feasibility Study

S.A. Baird[2], Y.H. Fleming[1], S. Kolya[3], D. Mercer[3], P.R. Newman[1], D.P.C. Sankey[2], A. Schöning[4], and R.J. Staley[1].
[1]University of Birmingham, Birmingham, UK
[2]CLRC Rutherford Appleton Laboratory, Oxfordshire, UK
[3]University of Manchester, Manchester, UK
[4]ETH Zürich, Switzerland

April 28, 2000

1 Introduction

In January 2000, the UK Particle Physics Experiments Selection Panel formally approved the proposal for the first level of the H1 Fast Track Trigger (FTT) [1]. Funding was agreed at the level of $\pounds 175\mathrm{k}$ and 2.5 staff years of engineering effort were promised from Rutherford Appleton Laboratory. This level of funding leaves a shortfall of $\pounds 85\mathrm{k}$ relative to the overall estimated cost of the level 1 system.

Design studies for the level 1 FTT have been in progress for several months now. This stage of the trigger is responsible for digitisation of drift chamber signals, identifying pulses corresponding to charged tracks passing through the chambers, measurement of pulse arrival times and z-coordinates and searching for patterns of hits on groups of three wires that correspond to genuine track segments originating from the primary vertex region. Three dimensional track segments have to be provided to the level 2 trigger within around $2-3\ \mu\mathrm{s}$ of the interaction. It is also intended that the FTT should provide information to the first level central trigger. This would extend the functionality of the FTT compared to that originally proposed [2], making the track information available to the central trigger significantly earlier, albeit with relatively poor resolution.

This document has been produced at the time of a 'break-point' identified by the PRC. The request was that the collaboration should show the feasibility of the segment finding stages of the algorithm. Since the design of our preferred solution including level 1 trigger information remains in a state of flux, we concentrate here on demonstrating the feasibility of the project at the more fundamental level of feeding track segments to the level 2 (precision track linking) stage of the trigger. If no information is provided to the level 1 central trigger, there is no stringent time limit on the track segment finding and the first real time constraint is that the tracks be fully linked, three dimensional track reconstruction performed and trigger decision logic be applied inside the combined level 1 and 2 latency of $2.5(\mathrm{L1}) + 19.8(\mathrm{L2}) = 23.3\ \mu\mathrm{s}$.

In this report, we first discuss the status of the level 1 studies and introduce the general principles involved in the design. We then address the specific feasibility question by assessing the time and resources required to implement the minimal solution to the level 1 problem in which no level 1 trigger is incorporated.

2 Status of the L1 Studies

A class of algorithms have been found which fit the basic level 1 requirements in terms of processing speed and cost of electronic resources. The plan is to implement the algorithm using Altera Field Programmable Gate Arrays (FPGAs). For the purposes of this report, we assume we will use the 20K600E devices [4]. These contain 600,000 programmable gates, organised into around 24,000 'logical elements' and around 150 'Embedded System Blocks' (ESBs) to be used for product term logic. Each ESB can be configured to perform a variety of tasks (CAM, RAM, and 8 FIFO as explained below). Three PC's have recently been purchased for DESY and Birmingham with sufficient memory to perform detailed simulations of FPGAs with these numbers of gates using the Altera Quartus software package. We are currently at the stage of converging towards a design for the full implementation of the algorithms in FPGAs.

3 Level 1 overview

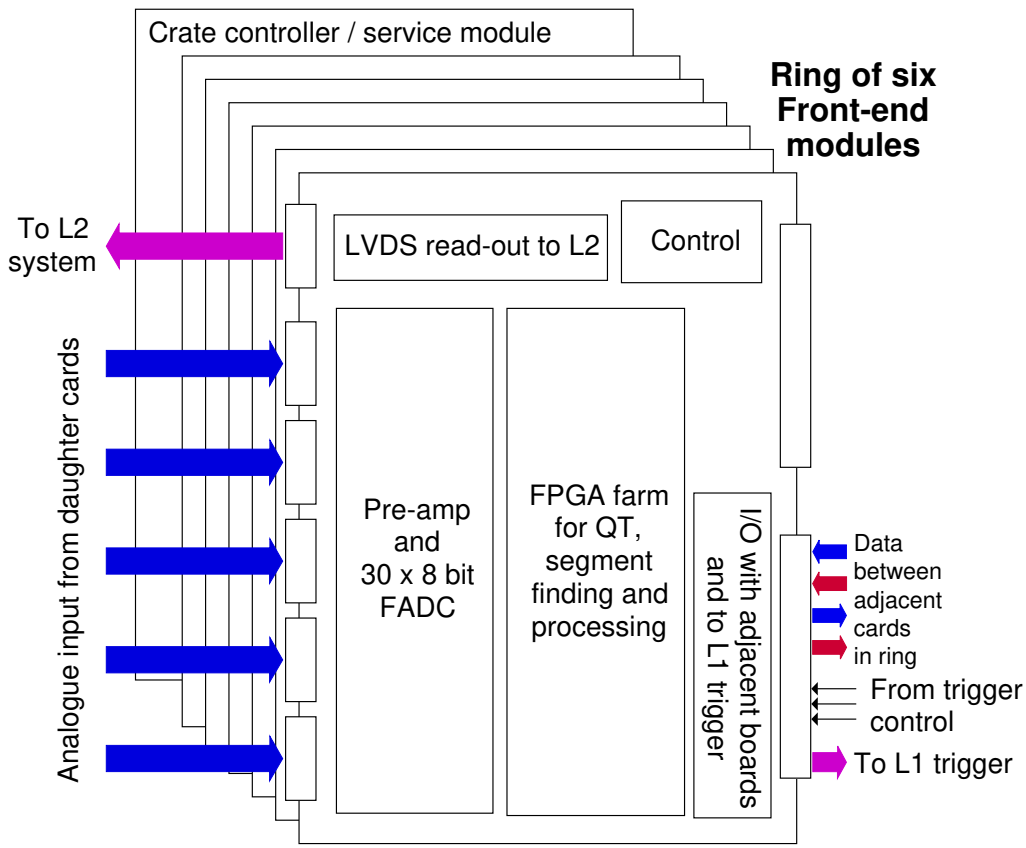

Figure 1: Overview of the level 1 FTT, showing a single crate. The Front End Module layout is shown in the foreground.

The general concepts of the level 1 FTT have not changed substantially since the original proposal. Details can be found in [2]. A short summary is provided here for completeness. The data flow within one crate of the system and external interfaces is summarised in figure 1.

Analogue drift chamber signals are picked up from the input to the existing DCrϕ track trigger system using new "plug through" adapter cards. This operation produces minimal disruption to the existing H1 configuration. The functioning of the DCrϕ trigger will not be jeopardized, so that the FTT and DCrϕ triggers can

operate simultaneously during the commissioning phase. The "plug through" adapter cards drive the analogue preamplified signals of a single trigger group (3 wires read out at both ends) up to 5 m to the front-end module cards (FEM). Signals from 5 trigger groups, adjacent in the ϕ coordinate, are fed to one FEM resulting in 30 analogue signals in total. These signals are digitized using 8-bit linear FADCs. The digitized data are passed to an FPGA farm, where all subsequent processing to produce track segments takes place. The functionality of the FPGAs is summarised in figure 2. The first step is the $Q - t$ analysis (see section 4). The timing output from the $Q - t$ algorithm is synchronised and written into shift registers clocked at 80 MHz. Based on the shift registers, a search takes place in the $r - \phi$ plane for valid track segments originating from the vertex region yielding signals in a group of three wires. Determination of the z-coordinate of the segment takes place in parallel, using the charge division technique.

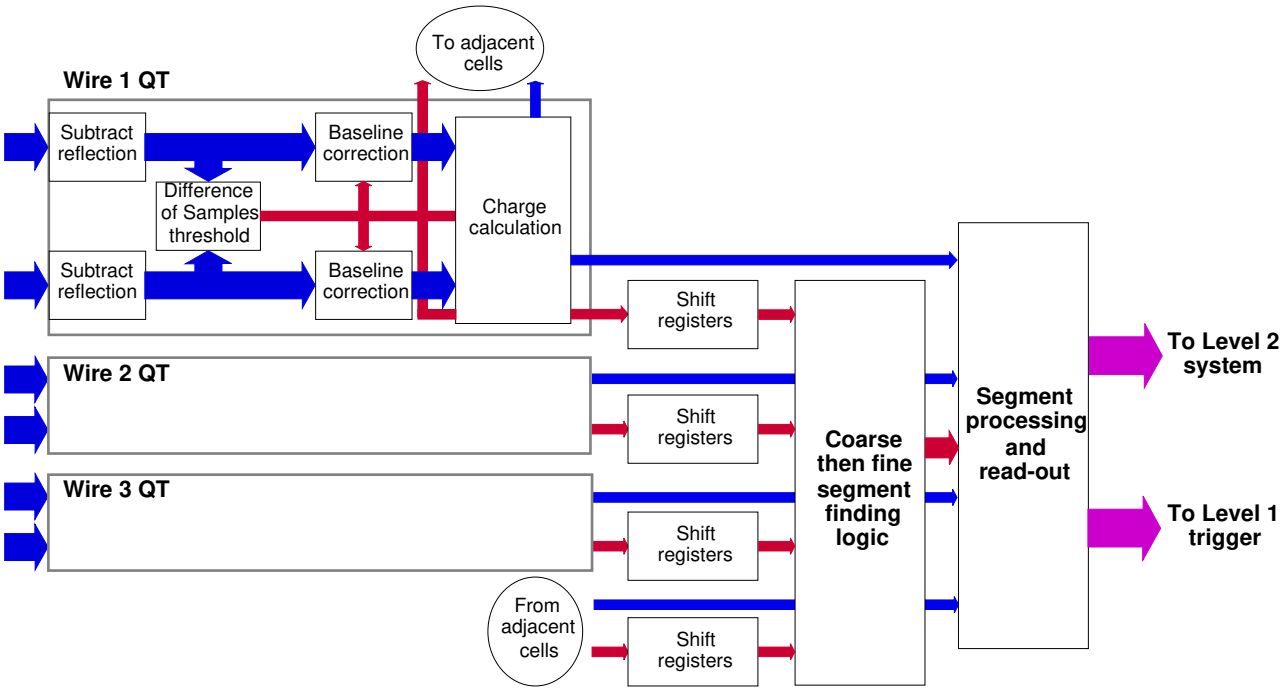

Figure 2: Block diagram illustrating the data Flow through the FPGAs.

The track segment finding is divided into two steps. In the first stage, the shift register bins are logically ORed in groups of four, such that the sychronisation frequency is effectively reduced to 20 MHz, one bin corresponding to approximately $1/20$ of the drift space.

In the minimal solution presented in section 5, this coarse segment finding does not commence until the Level 1 Keep (L1Keep) signal arrives from the central trigger. The L1Keep signal is distributed from the central trigger a fixed 2.5 μs after the interaction, such that the bunch crossing of origin is known to the FTT *a priori*. In the preferred solution incorporating a level 1 trigger the bunch crossing of origin is not in general known. An "on the fly" analysis of the coarsely segmented shift registers is then made using a 'pivot element' technique, where the arrival of a hit at a given register position in the middle of the three shift registers triggers a search for all track segments that include the pivot element. Most of the resulting track segments then have a 'validity' spanning several bunch crossings. The coarse track segments are then written into output buffers and become the starting point for the FTT L1 trigger. The solution incorporating FTT level 1 trigger will not be discussed further in this feasibility study.

Irrespective of whether a level 1 trigger is produced, the track segment refinement step starts only after a L1Keep. The coarse track segments are collected in output buffers and are fed to an SRAM. The full 80 MHz

sampling is restored by expanding each bin at $20\ \mathrm{MHz}$ to the 4 sub-bins contained. A large fraction of track segments can be rejected at this stage and for those that remain valid, the precision on the p_{t} and ϕ coordinates is improved. The output from the refinement step is merged with the appropriate z-coordinate derived from the charge division exercise. Finally the validated track segments are written to output buffers and then sent to the level 2 stage of the FTT.

4 QT Analysis

Figure 2 shows the current design of the $Q-t$ (charge and time) analysis. After digitisation using an 8 bit linear 80 MHz FADC (correcting for signal reflections along the 40 m analogue cables between the detector and the trailer is also being considered) hits are detected using a DOS (difference of sample) technique, illustrated in (see figure 3). A hit is defined as the time slice (clock cycle) of the first maximum in the DOS of the summed digitised data from both ends of the wire for each time slice in a region of at least two adjacent time slices where the DOS is above a threshold. The hit remains active, vetoing following hits, until there are at least two adjacent time slices where this DOS is negative. The hit finding is expected to be done in 5-6 cycles (about 75 ns) after the signals have been digitised in the FADC. The signals are synchronised and can be directly filled into the shift registers.

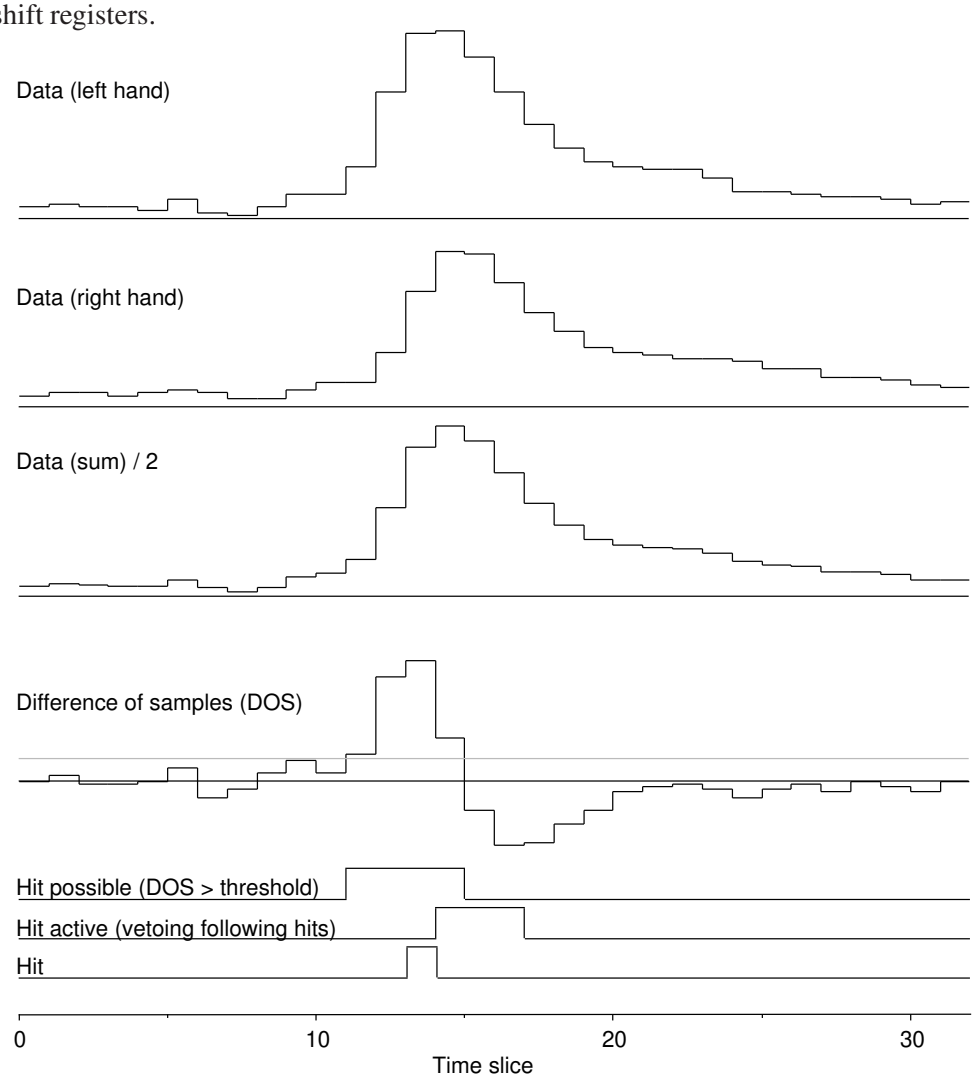

Figure 3: Sketch showing the DOS (difference of sample) technique. The FADCs are clocked at 80 MHz. See text for details.

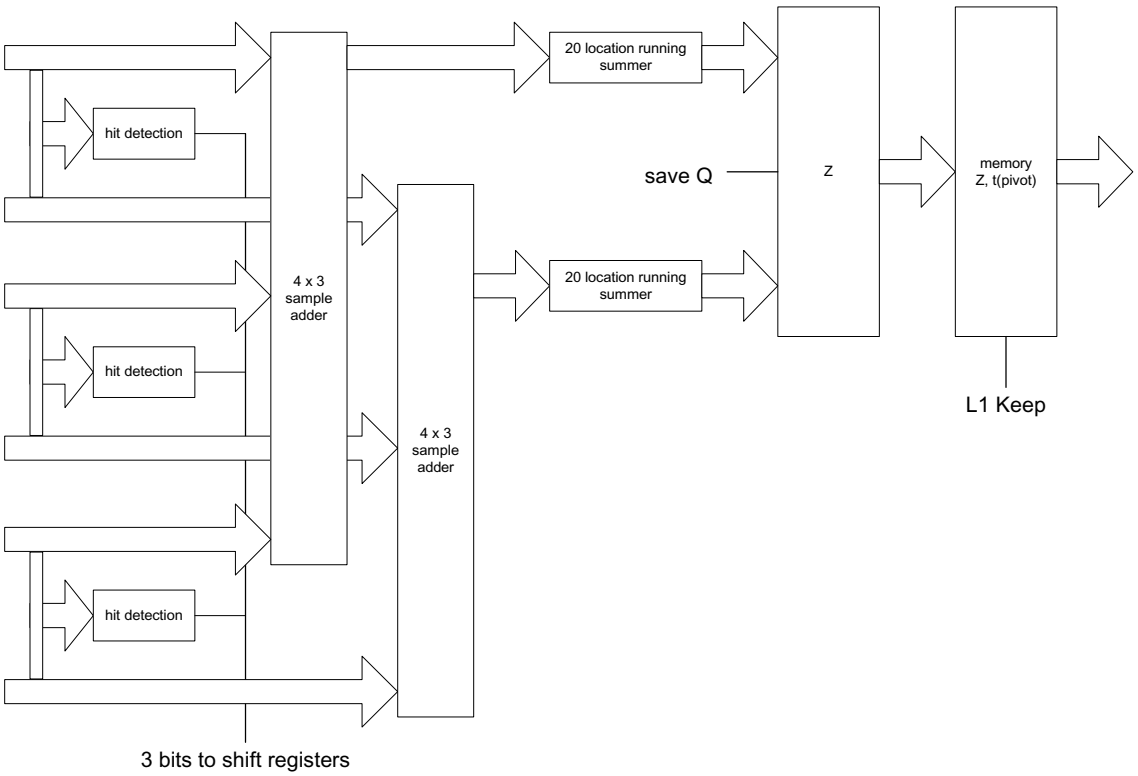

Figure 4: Overview of the cell-wise calculation of the z-coordinate using charge division. Charges measured in the group of three wires are summed up before making the division step.

At moment two methods are considered for the charge measurement. In a simple method the charge is measured cell-wise by using a running summer (see figure 4). In this case the charge is integrated over a period of about $1\ \mu$s. A hit, detected on the centre wire (layer) (a necessary condition for a track segment as defined in the next section), causes the instantaneous contents of the running summers to be passed via a buffer to the division step, where the z-coordinate is then calculated. This is determined by the relative fractions of charges measured at each set of wire ends. A big advantage of this method is that the charges measured by the group of three wires can be added and a common z-coordinate can be inferred before the hit pattern recognition is performed. Consequently, only one charge division operation has to be performed. The alternative approach would be the hit-wise measurement of the z-coordinate. In such a scenario the z-coordinates have to be associated to hits, entailing additional book keeping and buffering and a baseline correction of the signals is also required to minimise the influence of preceding hits.

It is presently felt that a single z-coordinate from the three wires comprising a trigger group is sufficient. FPGA space and time are saved by performing only a single division per group of 3 wires, rather than three. Potential disadvantages arise where additional hits at different z-coordinates are present on the registers within the range of the 20 bin running summer, skewing the z-coordinate in one direction or the other. However, these double hits, generally a result of synchrotron radiation hits, are not expected to be a frequent occurrence. Analysis of existing drift chamber data is in progress to assess the relative importance of the double hits and the extent to which subsequent particle searches are affected by the degraded track information.

5 Minimal Segment Finding Solution

In this section, a minimal solution to the track segment finding is presented and the necessary FPGA resources and the time taken are estimated. In this scenario, the track segment searches do not begin until after a L1Keep signal is received from the H1 central trigger. The bunch crossing of origin for an event is thus already known

when the segment finding begins, considerably reducing the complexity of the required algorithms. This exercise can be considered first as a proof of principle, demonstrating that the segment finding can be performed sufficiently quickly. It is also a good starting point for the development of more sophisticated algorithms that are capable of finding track segments and linking them to form full tracks without the prior knowledge of the bunch crossing of origin. An overview block diagram of the algorithm in question is shown in figure 5. Each separate stage of the algorithm is discussed in detail below.

A feasibility study for the level 2 FTT [3] has demonstrated that the track segment linking can be performed inside the level 2 latency of 19.8μs, with 0.6μs to spare (see table 8 of [3]). In this calculation, 0.48μs were assigned for L1-L2 latency and a further 2.6μs for receiving the data at level 2. Time can be probably saved by optimising the L2 fit algorithm or by making a faster L2 trigger decision which was conservatively estimated to be about 3μs. Taking [3] as a guide, any level 1 solution that takes less than 0.6μs would ultimately be viable. However, it would clearly be desirable to use as much of this 'spare' time as possible for final trigger logic at level 2, so the level 1 segment finding should be fitted into a much shorter time after the L1Keep if possible.

5.1 Shift Registers

The shift registers are clocked at 80 MHz and a double hit resolution of 50 ns is defined by the hit finding algorithm. Figure 5 shows the algorithm for the full drift space on both sides of the sense wire in a single cell. The hits can be considered logically as entering from the both sides of the register and simultaneously shifting to the centre. When a L1Keep signal is received, the registers are frozen. There is then a one-to-one correspondence between hit positions on the frozen registers and location of the hits inside the drift space.

For the first stage of the segment finding algorithm, four adjacent positions in the registers are logically ORed, such that the effective active length of the register decreases from a maximum 128 bits to a maximum 32 bits [1].

5.2 First Stage Track Segment Finding

Studies have been performed of the numbers of patterns in the groups of three shift registers that could correspond to genuine tracks. Considering only tracks with $p_{\rm t} > 90$ MeV and an acceptable χ^2 when fitted to a circular path in the $r-\phi$ plane, the resulting numbers of valid patterns of one hit in each of the three registers are shown in table 1, both where all three struck wires are in the same cell and for the case where one of the hits is in a neighbouring cell in the ϕ-coordinate. Due to the geometry of the chambers, cases where the outermost wire is struck in the left hand neighbour (case 00-1 in table 1), or where the innermost wire is struck in the right hand neighbour (case 100) are very rare. With the high level of redundancy in the planned system (presently demanding a coincidence of track segments in two out of the four possible tracker layers), these two cases can safely be ignored. However, it is clear from table 1 that it is very important to trigger cases where the outermost wire in the right neighbour (case 001) or the innermost wire in the left neighbour (case -100) are struck.

The search for valid patterns in the three registers is realised using Content Addressable Memories (CAMs). These CAMs will be used to store the predefined valid patterns in the groups of three registers. Invalid patterns do not have entries in the CAMs. Where a pattern from the registers matches one of those stored in the CAM, the address in the CAM corresponding to the match is output to the next stage of the algorithm.

The most recent release of the Quartus software contains a facility for easy building of 'Multiple Match' mode CAMs. This allows output for more than one valid match per CAM. In this case, two or more valid patterns,

[1] The necessary active lengths of the registers vary slightly from trigger layer to trigger layer, following the geometry of the cells. In fact, the shift registers have to be considerably longer than 32 elements in order to retain (pipeline) the data until the L1Keep signal arrives. However, when this occurs, at most the last 32 elements of the registers need be considered, as any other register positions do not physically correspond to hits in the chambers at the bunch crossing of interest.

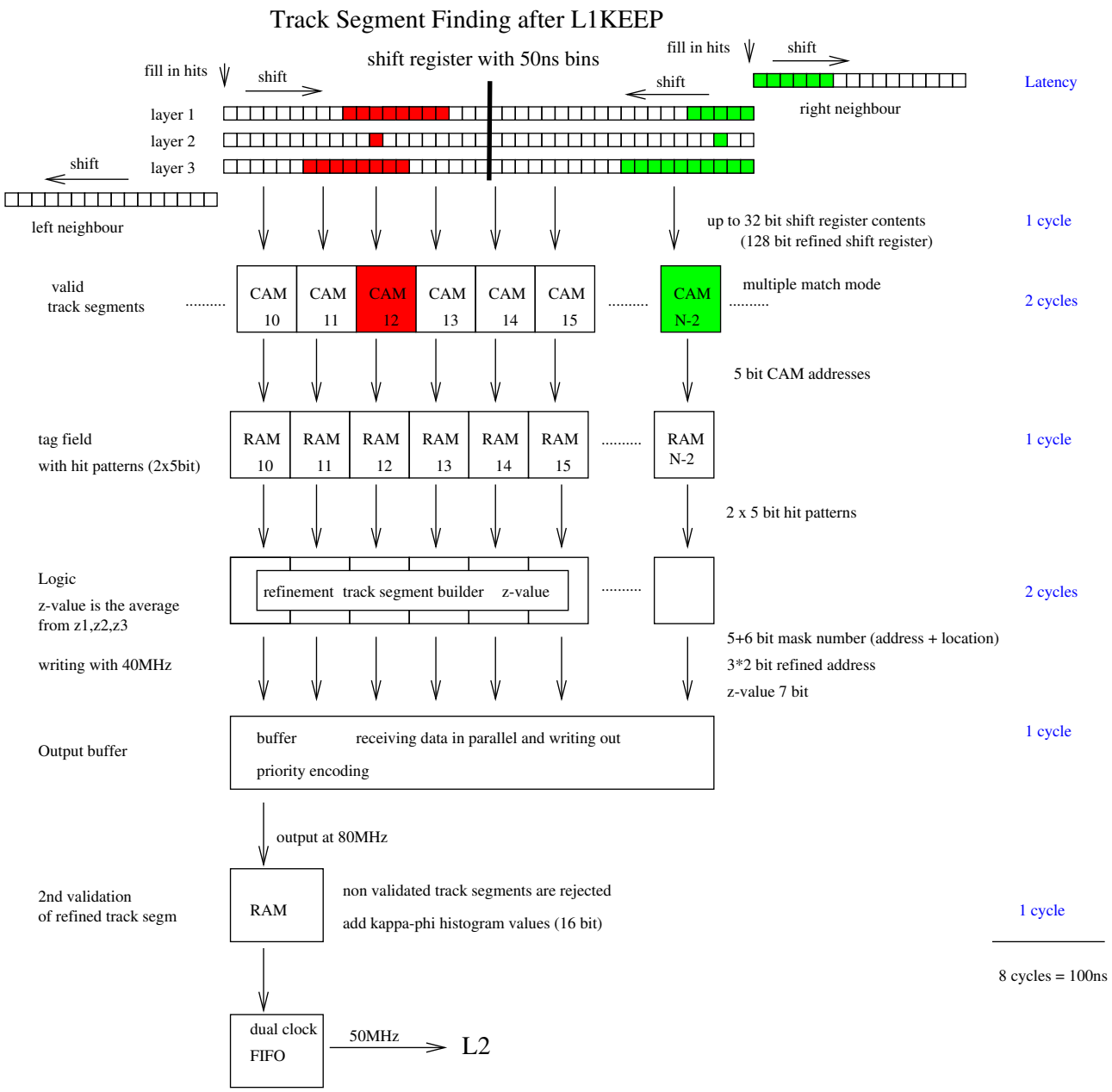

Figure 5: Overview block diagram for the segment finding, starting on the arrival of the L1Keep signal.

Trigger Layer	Cells	Valid Masks
1	000	127
1	-100	155
1	00-1	11
1	100	9
1	001	133
2	000	143
2	-100	243
2	00-1	36
2	100	30
2	001	199
3	000	169
3	-100	323
3	00-1	56
3	100	50
3	001	288
4	000	283
4	-100	405
4	00-1	43
4	100	52
4	001	393

Table 1: Numbers of combinations of entries in the three shift registers which could correspond to genuine track segments with $p_{\mathrm{t}} > 90\ \mathrm{GeV}$ at 20 MHz synchronisation frequency. Equivalently, these numbers represent the number of entries required in the CAMs for the coarse segment finding. The 'cells' column specifies which cells each register is taken from. The '000' entries correspond to patterns with all three hits in the same cell. Entries '-1' correspond to hits in the left neighbouring cell and '1' to hits in the right neighbouring cell, such that e.g. the entry for '00-1' corresponds to the case where the first two wires struck are in the same cell, with the third (outermost) wire being struck in the left neighbouring cell.

potentially corresponding to distinct track segments in the same drift space, can be found. The resulting CAM addresses are driven out in successive cycles. Since the final output from the FPGAs is a serial stream of track segments, this separation in time-slice of the data is not a problem.

At the first level of segment finding ($20\ \mathrm{MHz}$ sampling frequency), it is intended to configure one or maximal two multiple match CAMs for each register position on the central layer of the trigger group (see figure 5). The contents of the first and third layers provide the input to the CAM. The range of valid track segments including the single element on the central layer covers only a small fraction of the first and third shift registers as shown by the shaded areas of the registers in figure 5. Simulations have shown that for valid track segments involving any given element in the middle layer, it is never necessary to feed more than 32 shift register bins in total from the first and third layers (e.g. 16 bits from each of layer 1 and layer 3). The numbers of valid masks to be stored in each CAM are also generally less than 32 (see table /refvalid, which is for the full register). By implementing a single ESB as a CAM, 32 track segments with 32 bit width can be stored. The CAMs thus fit easily into a single ESB, making efficient use of the available gates. In the rare cases where there are more than 32 valid patterns involving a single register position in the middle layer, it will be necessary to use two ESBs. We conservatively estimate that 40 ESBs are needed for a given triplet of shift registers. Since there are never more than 32 valid patterns involved in a single ESB 5 bits of information are therefore sufficient for the CAM addresses.

The output from the CAMs is the one or more addresses in the CAM of the valid matches. These addresses are passed to tag fields (RAM), from which the output data from a successful search are the two original register positions from the first and third shift register. The positions within these 32 bit registers can be encoded in maximal 5 bits each. At this stage, the full algorithm is still proceeding in parallel, such that the position of the hit in the central layer is implicit in the CAM / RAM pair that gives an output.

5.3 Track Segment Refinement

As illustrated in figure 5, the output from the first stage of the segment finding is fed to the refined track segment builder. Here, the original $80\ \mathrm{MHz}$ shift register binning is restored just by ignoring the previous ORing of the shift register entries. That yields an extra 2 bits for each layer [2], describing the register position at the finer granularity.

The z-coordinate, calculated in parallel as explained in section 4, is re-associated with the track segment using e.g. the position in the centre layer, and is stored using an additional 7 bits. Here it has not been decided yet whether z-values are stored hit-wise or cell-wise. In the simplest approach the cell-wise calculated z-positions using the 20 bit running summer have already been averaged. In case of a hit-wise processing of the z-values a second shift register parallel to the hit shift register as used for the pattern recognition could be implemented. That register has to contain the z-position of already reconstructed hits as function of the drift time.

From this stage on, the data flow switches from parallel to serial, so a further 6 bits (thus allowing up to 64 CAMs to be addressed) are required in order to encode the CAM-RAM pair originating the match (or equivalently the register position in layer 2). The fully built track segments at $80\ \mathrm{MHz}$ sampling frequency are thus encoded in a total of 24 bits of information. These patterns are passed to an output buffer which receives the information from up to 64 parallel CAM streams.

The output buffer collects these data in parallel and feeds them sequentially to an SRAM. The conversion from parallel to serial data flow will be realised using a priority encoding, whilst ensuring that all segments are ultimately passed to the SRAM. The SRAM decodes the input information into p_{t} and ϕ information for the track segments, which are stored as 16 bits. Invalid segments at the $80\ \mathrm{MHz}$ sampling frequency are rejected on the basis of the finer granularity information. The final information is passed at $80\ \mathrm{MHz}$ to a dual clock 'first in, first out' (FIFO) buffer, which outputs the data at the $50\ \mathrm{MHz}$ frequency as specified for the FTT Level 2 input.

6 Timing and Necessary Resources

For several individual processing steps of the above described algorithm (see figure 5) the speed and the resources of the implementation have been evaluated by running the Quartus simulation package. Unfortunately, due to the late arrival of the high memory PCs software development first started in March and, as a consequence, a simulation of the complete algorithm could not be performed as yet. Nevertheless, all important parts have been simulated separately and this approach of programming single modules instead of the whole design is assumed to be valid. Fast routing lines (*FastTrack Interconnect*) in the Altera APEX devices ensure a fast data transfer between different so called *MegaLAB* (logic array blocks) structures, in which the different processing tasks are implemented. This ensures a high performance even if a large number of logic elements or ESBs for CAM and RAM applications is used. Consequently, scalability of the device is assumed in the following. That

[2] Since the double hit resolution of $\sim 50\ \mathrm{ns}$ corresponds to 4 bins at $80\ \mathrm{MHz}$ synchronisation frequency, the situation where there are two hits in the four $80\ \mathrm{MHz}$ bins corresponding to one $20\ \mathrm{MHz}$ bin can not arise, and we encode the four register positions in two bits.

means that the speed of the algorithm does not depend on the size of the implementation and that the resources needed (number of logic elements and ESBs) scale linearly with e.g. the number of track segments to be used.

Simulations also showed that several tasks like reading of shift registers, CAM operations, buffering, and even more complex logical operations can be performed at clock frequencies up to 180 MHz. Making use of pipelining techniques even mathematical operations like adding and multiplying can be performed at that speed. In the current design a clock frequency of 80 MHz is coservatively assumed. The final algorithm is hoped to be optimised by doubling the clock frequency for some parts of operations and thus reducing the latency of the track segment finding (the time when the first track segment can be sent to level 2) by about 10-20%.

6.1 Timing before Level 1 Keep

Since the segment finding in the minimal solution does not start until after the L1Keep signal arrives, overheads for the collection of the signals from the drift chambers, $Q - t$ analysis and filling of the shift registers need not to be considered in detail. Nonetheless, estimates are given here to demonstrate that these stages can easily be completed before the L1Keep signal arrives. The estimates are summarised in table 2. The overall time taken for these steps is estimated to be less than $1.5\ \mu\mathrm{s}$, which is comfortably within the $2.5\ \mu\mathrm{s}$ that have elapsed when the L1Keep arrives. The maximum drift time for any hit in the chambers under consideration is approximately $1.1\ \mu\mathrm{s}$ (outer layer in CJC1). This unavoidable delay varies between trigger layers. A second unavoidable overhead arises from cable delays. We conservatively allow for $40\ \mathrm{m}$ of cables here, corresponding to an expected delay of a little less than $200\ \mathrm{ns}$. The actual delay incurred by the $Q - t$ analysis is that incurred in digitising the data, finding pulses above the pedestal and clocking them onto the shift registers. The z-calculation is performed in parallel with the segment finding in the $r - \phi$ plane, and thus is not included in this timing analysis. An additional delay of $50\ \mathrm{ns}$ is incurred in the write operation, whilst passing the hits to the shift registers. This number can be directly inferred from the 20 MHz clock cycles of the coarse shift register.

Task	**Latency first segment (ns)**	**Cumulative Time taken (ns)**
Ionisation drift to sense wires	< 1100	1100
Cable Delays	200	1300
t-Analysis	75	1375
Synchronisation into Shift Registers	50	1425

Table 2: Estimates of time required to perform all tasks that must be completed before the L1Keep signal arrives from the central trigger ($2.5\ \mu\mathrm{s}$).

6.2 Timing after Level 1 Keep

Here we start with the assumption that we freeze the shift registers on the L1Keep arrival and estimate the time taken for segment finding from this point. The times taken can then be added to the L1Keep latency to estimate the time at which the track segments are provided to level 2. For this exercise we assume that we are clocking the FPGAs at $80\ \mathrm{MHz}$, such that a single cycle costs $12.5\ \mathrm{ns}$. Table 3 summarises the times taken. It is estimated that the first track segment information can be output to the level 2 FTT within $113\ \mathrm{ns}$ of the L1Keep signal arriving. A maximum number of ten valid track segments per wire group, of which at most 4 come from each CAM, has been derived from simulations. Taken that the last is output to level 2 about $400\ \mathrm{ns}$ after L1Keep. Because track segments are more rapidly validated than they can be sent to L2 using the 50 MHz LVDS link only the initial delay of the first track segment is relevant here.

Coarse Segment Finding: The complete shift register is read and shift register sectors are presented to the CAMs in a single step. The CAMs are operated in multiple match mode, which takes two cycles per match.

Task	Latency (ns)	Time first segment (ns)	Time last segment (ns)
Read Shift Register	12.5	12.5	12.5
Coarse Segment Finding	25	37.5	112.5
Decode combinations	12.5	50	125
Build Track Segments, get z-values	25	75	150
Write to Buffer	12.5	87.5	162.5
Validation in SRAM	12.5	100	387.5
Write to FIFO	12.5	112.5	400

Table 3: Estimates of time required to perform all tasks that are completed after the L1Keep signal arrives from the central trigger. The latency describes the actual delay in the processing. The cumulative times of the first and the last track segment do not include the $2.5\ \mu\mathrm{s}$ that have elapsed before the L1Keep arrives. The cumulative time for the last track segment assumes a maximum number of 20 non-validated track segments.

Up to 4 matches per CAM result in a maximum delay of $112.5\ \mathrm{ns}$ for the last segment.

Track Segment Refinement: In the refinement step about 50% of the track segments are re-validated. The original 80 MHz sampling information for each 20 MHz element in the track segments is decoded using a tag field (1 cycle), yielding a refined hit pattern. The so constructed refined track segments are merged with the z-information (2 cycles) and written to a buffer. Up to 20 not finally validated track segments have to be further processed in a second validation step using a discrete RAM which contains predefined valid masks and the track representation in κ and ϕ, which are needed for the subsequent L2 processing. The maximum delay comes from the serial processing in the refined validation step and results in $400\mathrm{ns}$ for the last track segment.

6.3 Required FPGA Resources

The choice of the size of the FPGA is driven by the number of logic elements (LE) and ESBs for the implementation of the algorithm and by the number of I/O pins. The number of LEs can be directly derived from the depth and the width of the objects. E.g. a single bit in a shift register or buffer can be implemented in one LE. For CAMs the number of logic elements needed depend on the operation mode. In the multiple match mode about 100 LEs are needed per ESB. Assuming that 40 CAMs are needed 4000 LEs have to be reserved. For the output buffer about 4000 LEs [3] and for the shift register 1000 LEs [4] are needed. The total number of logic elements per wire group including the $Q - t$ analysis, control logic, and I/O tasks is approximately 12000 (50% of APEX20K600 resources). ESBs are only used for CAM and RAM applications. In the current design each CAM and its tag field needs two ESBs resulting in about 80 ESBs (50% of APEX20K600 resources). In a minimum scheme inputs to the FPGA are just the digitized 8 bit FADC values from both ends of wires from all groups in the same cell and from the adjacent cells giving 80 input pins. Output pins are 24 bit track segments which have to be validated by the discrete RAM. More output bins are needed if the L1 trigger option is realised. These numbers have to be compared with the minimum 480 I/O pin FPGA packages which are available for the APEX 20K600 and so the required number of I/O pins does not impose any constraints on the choice of the FPGA.

[3] 40 output buffers containing a most 4 words a 24 bit

[4] 2.5μs L1 latency times 80 MHz clock frequency gives 200 bins per shift register. There are 3+2 shift registers from the group of three wires in the same cell plus the additional shift registers from the adjacent cells

7 Implementation Costs

The level 1 hardware costs are dominated by the price of the Altera FPGAs. In the current scheme 150 APEX 20K600 devices are needed which cost nowadays about 2000 DM on the open market. That price fits into the L1 cost estimates as given by the original FTT proposal. Two bigger FPGA series have already been announced by Altera and the APEX 20K1000 is expected to be deliverable in May/June this year. Prices for smaller FPGAs are expected to go down then significantly. To be conservative and for keeping the possibility of using even bigger FPGAs, e.g. to implement the L1 trigger option, we do not change the financing scheme at the current stage. Because of the decision to move the HERA upgrade shutdown by another 3 months to September 2000 the FTT schedule has also been shifted, which allows later purchase of the most recent FPGAs for even further decreased prices.

8 Summary

In this document, a simplified version of the planned level 1 track segment finding algorithm has been presented, in which the functionality is kept to the bare minimum required to feed the track segments to the level 2 segment linker. It has been shown that the extraction of track segment information can be done within around $100\ \mathrm{ns}$ of the L1Keep signal. This is much less than the 'spare' time calculated in the level 2 feasibility study. Though the estimates are fairly rough at this stage, it is clear that for example, even an extra $100\ \mathrm{ns}$ compared to our estimates would not be critical. The minimum hardware resources have been evaluated and it has been shown that the APEX 20K600E fits our basic requirements. Whether a bigger FPGA has to be used in order to implement a modified track segment finding for the L1 trigger option is not clear at the current stage and will be investigated in the next months.

References

[1] 'A Fast Track Trigger with High Resolution for H1', Proposal to the PPESP, number 750.

[2] 'A Fast Track Trigger with High Resolution for H1', and addendum, PRC 99/06, PRC 99/07.

[3] 'H1 Second Level Fast Track Trigger Feasibility Study', Supercomputing Systems (2000).

[4] Details of the Altera 20k series can be found at http://www.altera.com/document/ds/apex.pdf.