RD–27 note 31
October 1994

Beam Tests of a Single-Channel Bunch-Crossing Identification Module for the Level-1 Trigger – November 1993.

I.P. Brawn, R.E. Carney, Y. Ermolin, J. Garvey, R.J. Staley, A.T. Watson
The University of Birmingham, Birmingham, B15 2TT, UK

E. Eisenhandler, M. Landon
Queen Mary and Westfield College, University of London, London, E1 4NS, UK

C.N.P. Gee, A.R. Gillman, R. Hatley, V. Perera
Rutherford Appleton Laboratory, Chilton, Didcot, Oxon, OX11 0QX, UK

N. Ellis
CERN, 1211 Geneva 23, Switzerland

Abstract

Identifying a pulse which spans several sampling periods with only one of those sampling periods is a non-trivial problem which must be addressed in the context of level-1 triggering for the LHC. We have built and tested a single-channel bunch-crossing identification demonstrator which implements digital algorithms to extract such timing information.

1. Introduction.

Extraction of timing information for the ATLAS level-1 calorimeter trigger [1,2] is a non-trivial task due to several factors which are outlined below. Two separate approaches could be used to process incoming data from the detector front-end, either before or after the digitisation by the ADCs. The analogue method (as previously studied by the GEM collaboration [3]) has the advantages that it uses fast, simple and possibly cheap circuitry. Additionally there is no quantisation noise added to the signal as would be present post digitisation. Alternatively, techniques using digital filters have the advantages that they can be reprogrammed to meet changing experimental conditions and are thus more flexible. Also, non-linear algorithms can be employed which can be very effective at combating noise.

If the FERMI [4] front-end readout system is adopted by ATLAS, many tasks, including that of bunch-crossing identification (BCID), may be carried out digitally within the FERMI system. However, if FERMI is not chosen for the readout technology, the role of BCID falls into the domain of level-1 [5] which will in this case require its own ADC system. BCID could then be performed using analogue techniques before the ADCs, or digitally after them. We are currently investigating digital BCID technologies.

The main problems to be addressed are as follows:

- The expected interaction rate for proton–proton collisions at centre-of-mass energies ≈ 14 TeV and luminosities of 10^{34} $cm^{-2}s^{-1}$ is expected to be of the order of 25 events per bunch crossing. This pile-up is potentially a source of problems.

- The liquid argon (LAr) technology chosen for the ATLAS electromagnetic calorimetry [6] uses signal shaping which gives pulses spanning several bunch crossings (typically 4 crossings or 100 ns). For level-1 triggering each calorimeter signal must be identified as belonging to a unique bunch crossing. Its contribution to other bunch crossings must be suppressed. Only the signal corresponding to the peak position of the pulse should be passed onto the level-1 trigger processor.

- The problems of electronic noise, pile-up and quantisation errors become increasingly important as the pulse energy decreases.

Studies have been made of digital implementations of BCID technologies [7]. Simulation of several algorithms has been performed using data collected in previous beam tests of the bit-parallel trigger processing module [8,9] in conjunction with LAr prototype calorimeter modules. The data exhibited the correct bi-polar shaping of a calorimeter pulse and contained the effects of noise and digitisation as described above, but had no pile-up contribution.

A hardware BCID demonstrator module has been constructed which performs bunch-crossing identification for one input channel (out of 36) of the bit-parallel trigger demonstrator. The module has been tested at the LHC operating frequency of 40 MHz in a beam line at CERN.

2. Algorithms.

The role of the BCID module can be summarised as: To take a pulse which spans several bunch crossings and identify it with only one of those bunch crossings. If digital techniques are employed then the signals will be digital samples of analogue data taken synchronously with each bunch crossing. The BCID filter is then required to examine a pulse occupying a finite number of samples and transform it to an impulse occupying only one sample.

Although digital signals are represented by a series of discrete time samples of analogue data, timing information which can be extracted from the signal is not limited to the set of times at which the samples were taken. i.e. the known shape of a bi-polar pulse would allow the location of the zero crossing point (even if it was not directly on a sample) and information on the phase between the pulse centroid and the samples of that pulse could be obtained.

Sampling will be driven by the LHC clock from which the ADC strobe will be generated. Programmable delays will be used to adjust the phase of the ADC strobe to ensure that each calorimeter pulse is sampled at its maximum. However, some variation between the phase of the calorimeter pulse and the samplings will always be present, due to particle time-of-flight differences etc. This will result in small errors in the transverse energy (E_T) measurement, but these measurements should still be possible to an acceptable accuracy. This is sufficient for the level-1 trigger so for reasons of latency, all jitter corrections will be left to the level-2 trigger system. Extraction of timing information at level-1 will be limited to discrete values – the bunch-crossing numbers.

Many BCID algorithms have been considered. Excepting deconvolution filtering and stand-alone peak-finding algorithms, all of the algorithms investigated use a finite impulse response (FIR) filter followed by a peak-finding algorithm. The deconvolution algorithm consists solely of FIR filtering methods.

2.1 The Finite Impulse Response Filter.

The finite impulse response (FIR) filter is a discrete linear system which is invariant in time. Formally stated

- A discrete system is one which converts an input data set, $x(n)$, into an output data set, $y(n)$:
  $$x(n) \dashrightarrow y(n)$$
  where n is an integer representing the sample number.
- The definition of a linear system is that
  $$x_1(n) + ax_2(n) \dashrightarrow y_1(n) + ay_2(n)$$
  should be true for any value of a.

The behaviour of a discrete, linear, time-invariant system is governed by a convolution equation, i.e. the output from the system is given by a convolution of the input data with the system's impulse response. The impulse response is defined as the output produced by the system when the input consists of a unit impulse at time zero. The set of values $h(n)$ which constitute the impulse response completely define the system. The definition of a FIR filter is one that has a finite set of such values. The equation describing the behaviour of a FIR filter is

$$y(n) = \sum_{i=0}^{N-1} h(i)x(n-i)$$

where	$h(i) = a_i$	if	$0 \le i \le N-1$
and	$h(i) = 0$	otherwise.	

The impulse response of the FIR filter consists of N coefficients a_i. The output sample $y(n)$ is formed from the sum of the N previous input samples, weighted by the coefficients a_i.

2.2 Peak Finding.

Peak-finding algorithms can either be used following FIR filtering, in which case the peak finder receives filtered data, or in stand-alone mode. Local peaks are identified in the data and a flag is generated. It is only these flagged peaks that are passed onto the trigger processor hardware.

The peak finder receives as input consecutive samples of data $x(n)$, $x(n-1)$, $x(n-2)$ .... However, for clarity and convenience the samples are better denoted as R_1, R_2, R_3 ... where the sample with the largest subscript is the most recent. i.e. R_2 is more recent than R_1. With each clock cycle the samples shift one position such that R_p ----> R_{p-1}. Any number of samples from three upwards can be used in the peak-finding process such that

$$R_1 \; op \; R_2 \; op \; R_3 \; op \; ..... \; R_{p-2} \; op \; R_{p-1} \; op \; R_p$$

where R_p are the successive samples and op is any of the operators <, >, =, ≤, ≥. Typical examples are:

a) $R_1 < R_2 > R_3$

b) $R1 < R2 < R3 > R4 > R5$

c) $R1 < R2 > R3 > R4 > R5$

d) $R_1 \le R_2 > R_3$

e) $R_1 < R_2 \ge R_3$

In case (a) a local peak exists at sample R_2 and will be flagged. However, the position of the peak sample in the data stream is dependent on whether filtering has been applied, as peak positions in filtered data do not necessarily correspond to the peak of the original calorimeter pulse. This offset is fixed for any particular FIR filter that may be applied. This has to be taken into account when flagging a peak as it is the sample of the original calorimeter signal that has to be sent to the trigger processor.

The number of samples and the order of the operators can be used to control the functionality of the peak finder. i.e. its response to differing signal shapes and noise.

Of the cases listed above (a) is the simplest and will find any single-sample peaks in the data. However, it is particularly susceptible to noise, ADC pedestal fluctuations and quantisation errors, all of which could result in the generation of false flags or loss of real ones. Case (b) finds similar peaks but is more selective and thus far less likely to generate false flags. Case (c) or a variant can be used if the pulse is asymmetric and variations of (d) and (e) will find peaks where the maximum spans more than one sample. However, the following factors should be considered when choosing an algorithm.

- Noise, fluctuations and errors become increasingly more important as the pulse size decreases. In the low energy regime these effects will produce false peaks where none are really present, or flatten true peaks thereby losing them.

- Use of a larger number of coefficients reduces the number of false flags generated by fluctuations etc. However, it will also reduce the number of true peaks found in the low energy region, where a low signal to noise ratio causes signals to be increasingly distorted from their ideal shape. False flags generated by noise are less serious than incorrect flags generated due to distorted pulses. If a pulse is incorrectly identified, a large amount of energy may be added to the wrong bunch crossing and omitted from the correct one. A flag generated solely by background however will only involve a small energy discrepancy in the signal, resulting in increased noise passing to the trigger processor.

The consequences of misassigning pulses to bunch crossings depends on the sizes of the pulses involved. If a large pulse is assigned to the wrong bunch crossing, a trigger for the wrong crossing or a missed trigger could result. However, for small pulses the effects are less serious if misassignment occurs. Small pulses are more important for the isolation threshold of the cluster-finding algorithm. Here, a pulse assigned to the wrong bunch crossing would reduce the effectiveness of the isolation criteria, but not produce a trigger for the wrong bunch crossing.

In summary, the best peak-finding algorithm will depend on the characteristic shape of the pulse being examined. However, the digital techniques being studied will allow on-line reconfiguration of the algorithms to best match changing experimental conditions and pulse shapes.

2.3 FIR Algorithms.

Four algorithms have been considered which are described more fully elsewhere [7] (and refs. therein). However, a brief description of each is given below.

2.3.1 Deconvolution Filters.

The purpose of a deconvolution filter is to deconvolve a shaped calorimeter pulse back into an impulse signal. This is achieved by convolving the pulse with the inverse of its own system function. By performing a single convolution, an ideal filter should be able to extract information on both the timing and magnitude of the signal. There are however two major drawbacks of this algorithm. Its effect of increasing any noise present in the signal means it is only suitable for signals with a low noise content. Also the algorithm can only be applied to signals with a certain characteristic shape.

2.3.2 Zero Crossing Identifiers.

These algorithms are best suited to analogue technology and require a bi-polar pulse as input. It is preferable to extract timing information from the point of zero crossing, on the falling edge of the pulse where the gradient is non-zero. This point is amplitude independent. To implement this type of algorithm digitally, a FIR filter is used to integrate the input calorimeter pulse which generates peaks at points where zero crossing occurs. Integration is performed by convolving the pulse with a set of unitary coefficients (1, 1, 1, ...). The minimum number of coefficients required is that which covers the positive region of the pulse. A peak finder is then used to locate these maxima. A major advantage of this form of the zero crossing algorithm is that it requires no knowledge of the ADC pedestal value. This independence means that the algorithm is unaffected by any pedestal drift.

2.3.3 Constant Fraction Discriminators.

Again this algorithm is more widely used in analogue signal processing. Timing information is extracted from the rising edge of the pulse where the gradient of the pulse is at its maximum. Essentially the constant fraction discriminator measures the time at which the pulse height rises above a threshold value. A fixed threshold can not be used as the time at which it is crossed will vary with the amplitude of the pulse. Instead, a threshold is set at a constant fraction of the total pulse height. The initial input pulse from the calorimeter is sent along two channels, one of which is delayed by a time Δt and multiplied by some fraction X. The value of Δt is chosen such that the rising edge of the delayed pulse coincides with the peak of the undelayed pulse. The delayed pulse is then subtracted from the undelayed pulse and the result is a pulse which has a point of zero crossing at the time where the delayed pulse had a magnitude of $1/X$ of its total amplitude.

For a digital constant fraction discriminator, the generation of the delayed pulse and the subtraction from the undelayed pulse are performed using a FIR filter. A second FIR filter is then used to integrate the resultant pulse. However, the two convolutions required for this process can be combined into a single FIR filtering process.

It should be noted however that due to the nature of the digital signals, Δt is limited to a discrete set of values. Also the height of a digital pulse is not a continuous variable. The threshold could be set to correspond to the height of a sample or to lie between two consecutive samples. If the threshold is set to the height of a sample, even the smallest amount of noise in that sample will force it above threshold, shifting the time at which the constant fraction discriminator fires by a whole time slice. To best counter this effect the threshold should be set midway between consecutive sample heights.

2.3.4 Comparison of a Pulse with its Integral.

This algorithm functions in a similar manner to the constant fraction discriminator. It generates a flag when the pulse height is equal to a constant fraction of its own integral. Two convolutions are required to implement this method digitally. The first multiplies the original pulse by a factor *X* and subtracts from it the integral of the original pulse. The resulting pulse has a point of zero crossing where the original pulse height had a magnitude of *1/X* of its integral. A second convolution then integrates this pulse before passing it onto a peak finder. Again these two convolutions can be combined into a single FIR filtering process. The same considerations as for the constant fraction discriminator apply when optimising the value of *X*.

3 Hardware.

With the exception of the deconvolution filter and stand-alone peak-finding algorithm, all of the BCID processes consist of a FIR filter followed by a peak finder. A schematic diagram of the hardware required to implement such a system is shown in Figure 1.

The BCID module hardware is based around a pipelined systolic architecture. As seen in Figure 1a, the incoming data are stored in a pipeline memory whose length is determined by the latency of the FIR filter and peak finder. A data sample should arrive at the multiplexer (MUX) in synchronisation with the BCID flag generated for that sample. If no flag is produced, the output is suppressed to zero. The FIR filter (shown in more detail in Figure 1b) inputs data from the pipeline memory. It uses this data in calculating the weighted sum. The latches form the pipeline memory which contain the *N* previous data samples. The samples are multiplied by the coefficients a_i and summed in an adder tree as described in section 2.1 above. The FIR filter output is fed into a second pipeline memory which acts as input to the peak finder (Figure 1c). The logic shown in the figure only uses three inputs but can be expanded to use any of the algorithms listed above. The peak finder output is the BCID flag which determines whether data or zero is sent to the trigger processor.

3.1 The Demonstrator Module.

A schematic layout of the one-channel demonstrator is shown in Figure 2. The module was built in Birmingham [10] and has been tested with the RD27 phase-1 bit-parallel trigger processor in a test beam at CERN during November 1993. The module operates at 40 MHz clock rate and has a latency of ten pipeline steps to allow the maximum flexibility in testing the different algorithms. The theoretical minimum latencies for the individual algorithms are listed in Tables 1 and 2 in numbers of bunch crossings. Programmable FIR and peak-finder logic has been used to enable testing of all of the algorithms.

Multiplication of data samples in the FIR filter is performed via Look-Up-Tables (LUTs). In the demonstrator module this was implemented in RAM to achieve fully programmable logic. The FIR filter was able to convolve input data with up to five coefficients. A fully programmable five inputs peak finder allowed many algorithms to be tested. The peak finder consists of a set of comparators with a LUT to combine the outputs and produce the overall peak finder result (BCID flag).

The BCID module generates three outputs:

- the unfiltered input data — E_{Tout}.
- convoluted data produced by the FIR filter — E_{Tconv}.

- data from the full BCID algorithm chain — E_{Tzsup}.

The peak finder output flag acts as an enable input to the MUX so that data is only output if a peak is found. The MUX receives data from one of four positions in the pipeline memory (to allow for offsets between actual samples and flagged samples) where the correct sample will depend on the choice of FIR filter and peak finder used.

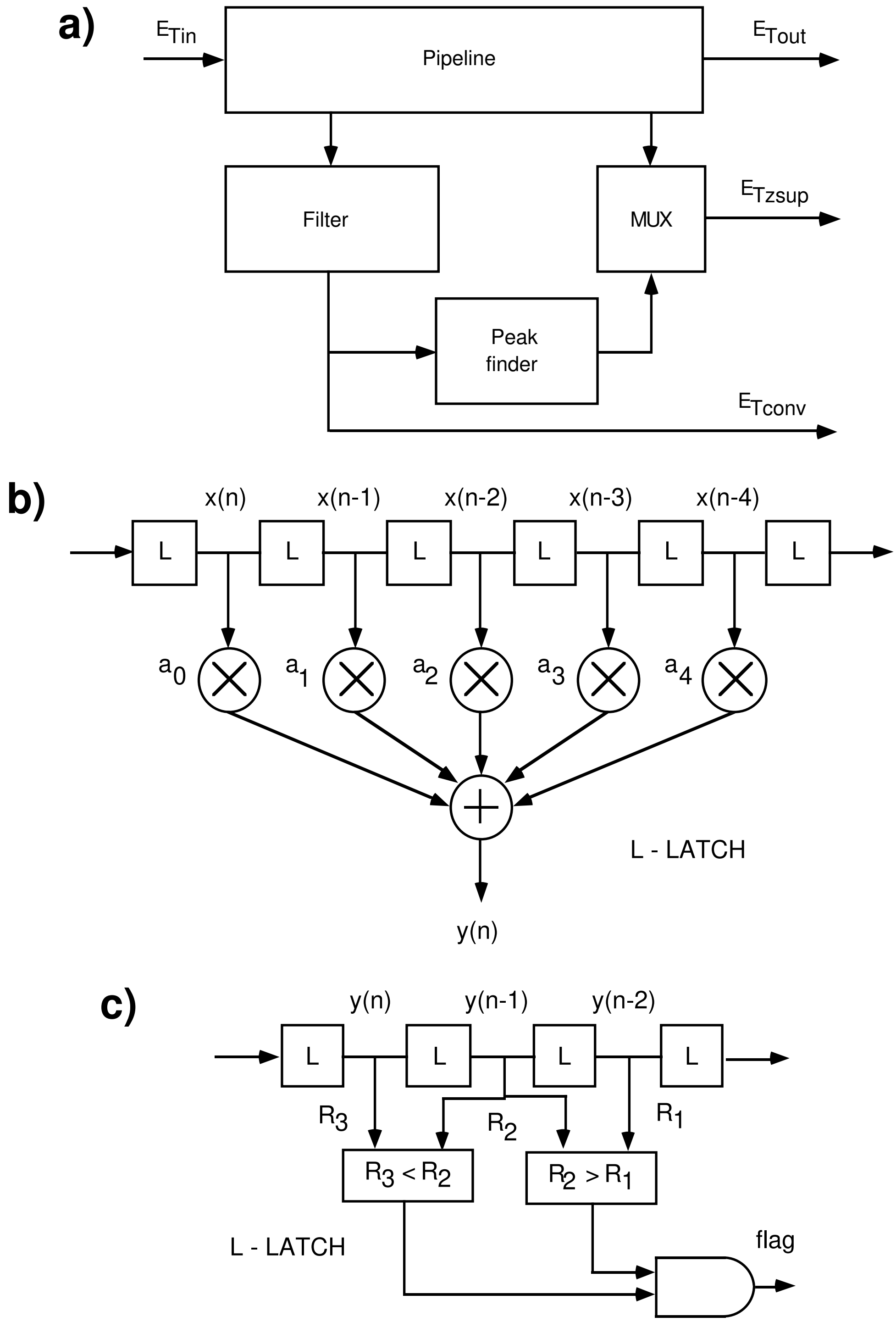
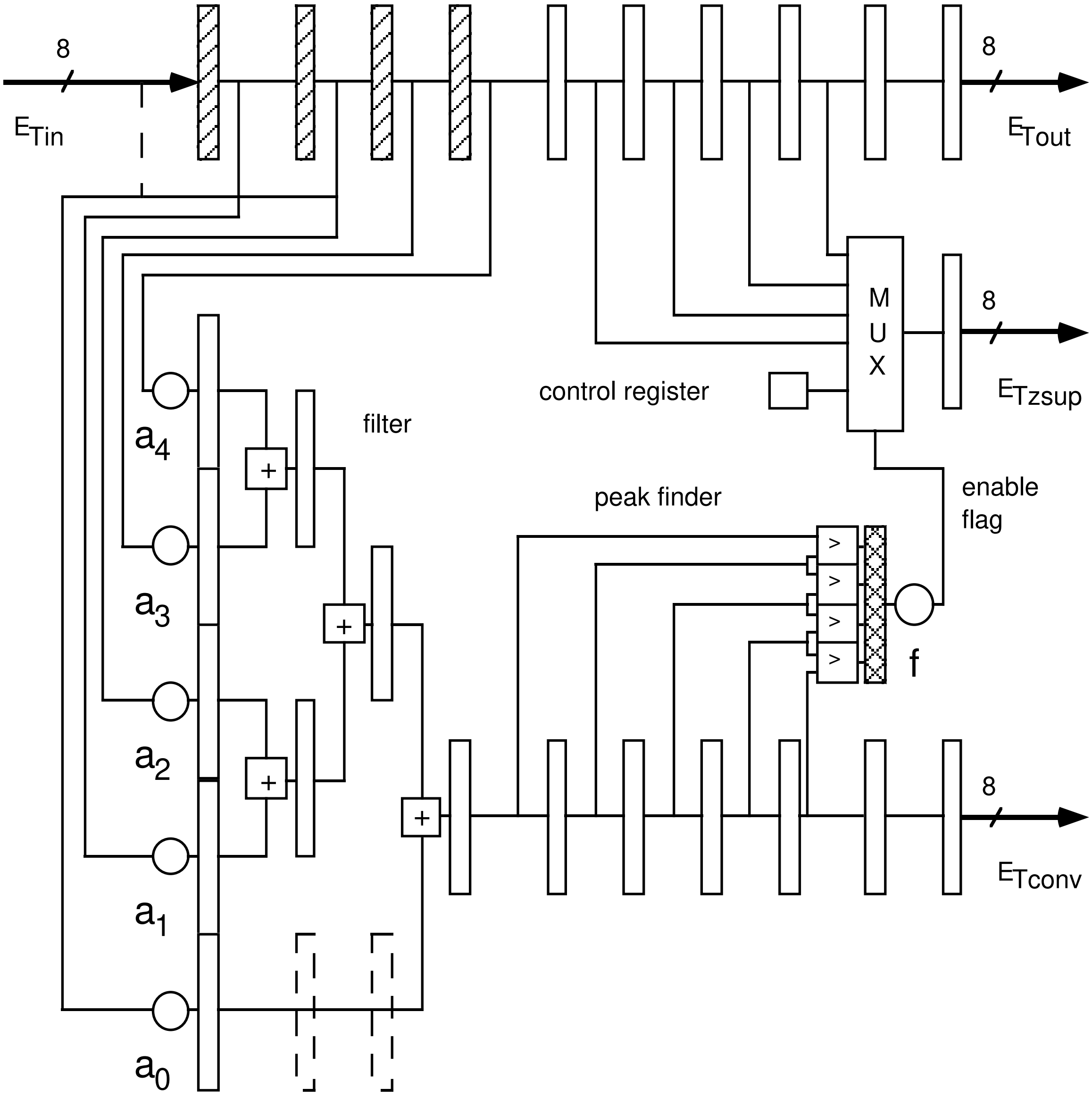


Figure 1: a) General hardware implementation of the BCID process, b) Hardware implementation of the FIR filter, c) Hardware implementation of a peak finding algorithm.



UA1 BUS PROTOCOL INTERFACE

ADDR 0	CONTROL REGISTER	3 BITS	R/W (S1,S0,R/L)
ADDR 1	LUT1 DATA REG (a_4).	12 BITS	R/W
ADDR 2	LUT2 DATA REG (a_3).	12 BITS	R/W
ADDR 3	LUT3 DATA REG (a_2).	12 BITS	R/W
ADDR 4	LUT4 DATA REG (a_1).	12 BITS	R/W
ADDR 5	LUT5 DATA REG (a_0).	12 BITS	R/W
ADDR 6	LUT6 DATA REG (f).	1 BIT	R/W
ADDR 7	ADDRESS REGISTER	8 BITS	R(Par.)/W(Ser.) 1 BIT

Figure 2: The one-channel BCID demonstrator module.

4 Beam Tests.

The module was tested in conjunction with the bit-parallel trigger processor in a beam line at CERN during November 1993. Real data were derived from a prototype liquid-argon end-cap (Spanish Fan) module of the RD3 collaboration.

Essentially the experimental configuration was identical to that described in reference 8, although a brief recap is given for completeness. The experimental layout can be seen in Figure 3.

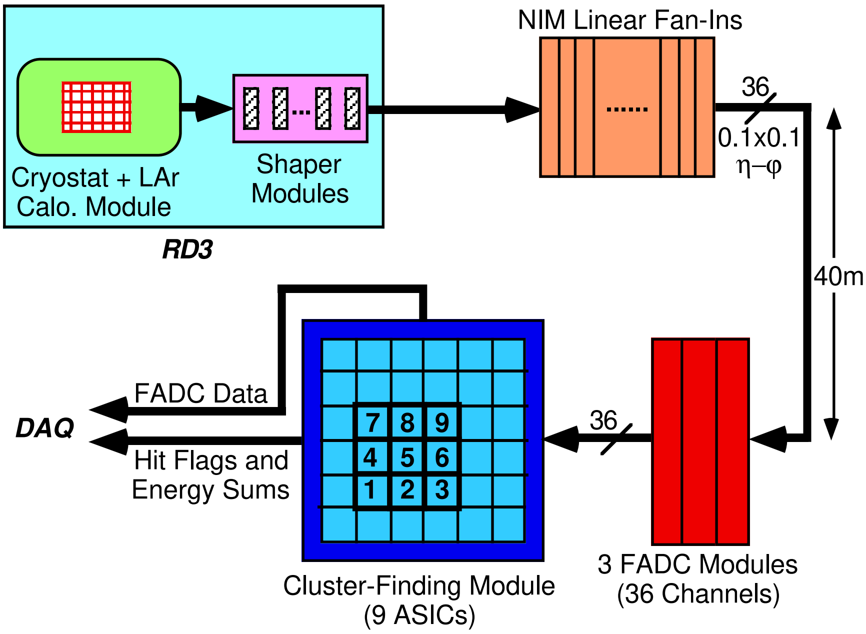


Figure 3: Block diagram of the calorimeter and trigger system.

Trigger cells of granularity $\Delta\eta \times \Delta\varphi \approx 0.1 \times 0.1$ in pseudorapidity–azimuth space are produced from the calorimeter cell data in an analogue summation tree (the cell sizes are constant in rapidity and thus vary in physical dimension). The resulting trigger-cell analogue signals are then digitised by linear 8-bit FADCs. These data are input to the Cluster Finding Module (CFM) which processes the input data as a 6×6 array of trigger-cells.

The BCID test module was a single-channel board. It was housed in the same crate as the FADCs and CFM and sat between them in the data-flow chain. The module accepted one channel of 8-bit digitised data from the FADCs and output three 8-bit words (E_{Tout}, E_{Tconv}, E_{Tzsup}.) to the CFM. In this test the CFM was used as a convenient memory trap to store the incoming data from the BCID module. The 256-deep input memory of the CFM was read out and recorded for later off-line analysis, where a comparison could be made between the output from the module (hardware) and the expected output obtained by software evaluation of the BCID algorithm.

Several FIR filter and peak-finding algorithms were implemented and are describe below. All test data were produced from a 200 GeV electron beam.

4.1 FIR Filters and Peak Finders.

The planned test strategy involved the peak finder and FIR filter algorithms listed in Tables 1 and 2 below, where the theoretical latency in bunch crossings for each algorithm is also shown. The programmable nature of the module allowed any combination of the peak finders and FIR filters. Table 2 shows the five coefficients used in calculating the weighted sum in the filter before data entered the addition tree. The filter *fil0* effectively disables the FIR block as the output is identical to the input. Filters *fil1* and *fil2* are integral weightings as would be used with a zero-crossing identifier. Two variations of constant fraction discriminator are implemented with filters *fil3* and *fil4*. Finally *fil5* effects a comparison-with-integral algorithm. Additionally the offset to determine which data sample the MUX selected was varied on a run-by-run basis to accommodate the choice of algorithms. Earlier simulation work [7] showed that deconvolution filters had only limited success with bi-polar pulses.

pf1	R1 < R2 > R3 > R4 > R5	3
pf2	R1 < R2 < R3 > R4 > R5	2
pf3	$R_1 < R_2 > R_3$	1
pf4	$R_2 < R_3 > R_4$	1
pf5	$R_1 \leq R_2 > R_3$	1
pf6	$R_1 < R_2 \geq R_3$	1
pf7	$R_2 \leq R_3 > R_4$	1
pf8	$R_2 < R_3 \geq R_4$	1

Table 1: Peak-finding algorithms to be tested with latency in bunch crossings.

	a_0	a_1	a_2	a_3	a_4	latency
fil0	1	0	0	0	0	0
fil1	1	1	1	0	0	1
fil2	1	1	1	1	0	2
fil3	1	1	-2	0	0	1
fil4	1	-2	-2	0	0	0
fil5	1	0	-1	0	0	0

Table 2: FIR filter algorithms to be tested with latency in bunch crossings.

5 Results.

Software was written to act on the same input data ($E_{Tin} = E_{Tout}$) that the BCID module processed. This data went through simulated FIR filters and peak finders to determine whether a BCID flag (peak found) should be set. Finally a simulation of the MUX determined whether zero (no flag) or data (flag) was output (E_{Tzsup}). The FIR filter output was also available as E_{Tconv}.

Some 13.5 k events were analysed. An event typically consisted of 248 recorded 8-bit data samples, therefore the module processed ≈ 1.14 million data samples. Of these, 98 samples differed between the simulation and hardware FIR filter output. These samples often followed a change of many bits in the data pattern. i.e. from few bits set to many set or vice versa. The simulation and hardware peak-finder output (i.e. that at the end of the full algorithm chain) showed 100% agreement. Although the 98 incorrect FIR samples did not affect the final output result, this was purely by chance and must still be considered as errors. (Often an existing peak sample was increased or reduced by the FIR error, leaving the peak in the data but with the wrong pulse height.) Figure 4 shows the output from each stage of the BCID process for a typical event spanning 248 samples. The top plot is the raw FADC data as input to the module, the middle the output from the FIR block after applying the FIR algorithm, and the lower, the final BCID output after the filtered data had passed through the peak finding block. Several noise hits can be seen in addition to the main pulse in the peak-finder output. There was no subtraction of ADC pedestal or any thresholding performed prior to processing the input data. It can be seen that subtraction of pedestal would greatly reduce the frequency of such hits, and this is to be tested in the phase-2 BCID module.

As previously described, at the LHC the ADC strobe will be derived from the LHC clock, and there will be a constant a phase difference between the strobe and the maxima of the pulses being sampled. Programmable delays will be employed to adjust the phase and ensure that each calorimeter pulse is always sampled at its maximum. During the beam tests a TDC was used to measure the difference in time between the arrival of a particle (the trigger) and the phase of the system clock running at 40 MHz. The TDC range spanned 100 counts (460 – 560) with a resolution of 0.25 ns per count. Figure 5 shows the TDC value versus the maximum pulse height for each event (in ADC counts). It is these maxima that the BCID module should locate. As the module only processed a single trigger channel, there is a wide spread in energies due to energy sharing between neighbouring trigger cells, but a distinct 'seagull' edge can be seen on the right hand side of the plot. This is where the maximum pulse height was recorded for any given TDC value. The dip in the centre of the edge (between TDC values ≈ 485 – 535) is caused by off-peak sampling. Therefore, ADC data with TDC phase values in this range are not on the peak of the pulse and should not be used. By selecting on the TDC value, it is possible to extract only those samples taken on the peak of the pulse. i.e. those with the correct phase relative to the clock. A selection of TDC count values in the ranges 460 ≤ TDC ≤ 485 or 535 ≤ TDC ≤ 560 (from now on referred to as TDC1 and TDC2 respectively) excludes the dip region in the centre of the right hand edge of Figure 5, which is the region corresponding to off-peak samples. This is indicated by the two horizontal lines drawn on the figure.

Figure 6 illustrates the TDC value versus the ADC sample or slice in which the pulse maximum was actually found (usually sample number 192 or 193). This can be seen to correspond to the right hand edge of Figure 5. Clearly selecting values in the regions TDC1 and TDC2 remove those data samples not taken at the pulse maximum, leaving only samples in bin 192 – the correct position of the peak sample.

In Figure 7a the slice containing the peak sample as found by the BCID module is plotted for any events with a TDC value in the range TDC1 or TDC2. The correct slice is found by the BCID module in all cases. Figure 7b shows the same information for events not included in the above TDC ranges. As can be seen from figures 5 and 6 there is some overlap in TDC value as the peak sampling moves from bin 192 to bin 193, thus explaining why some samples from bin 192 feature in this plot.

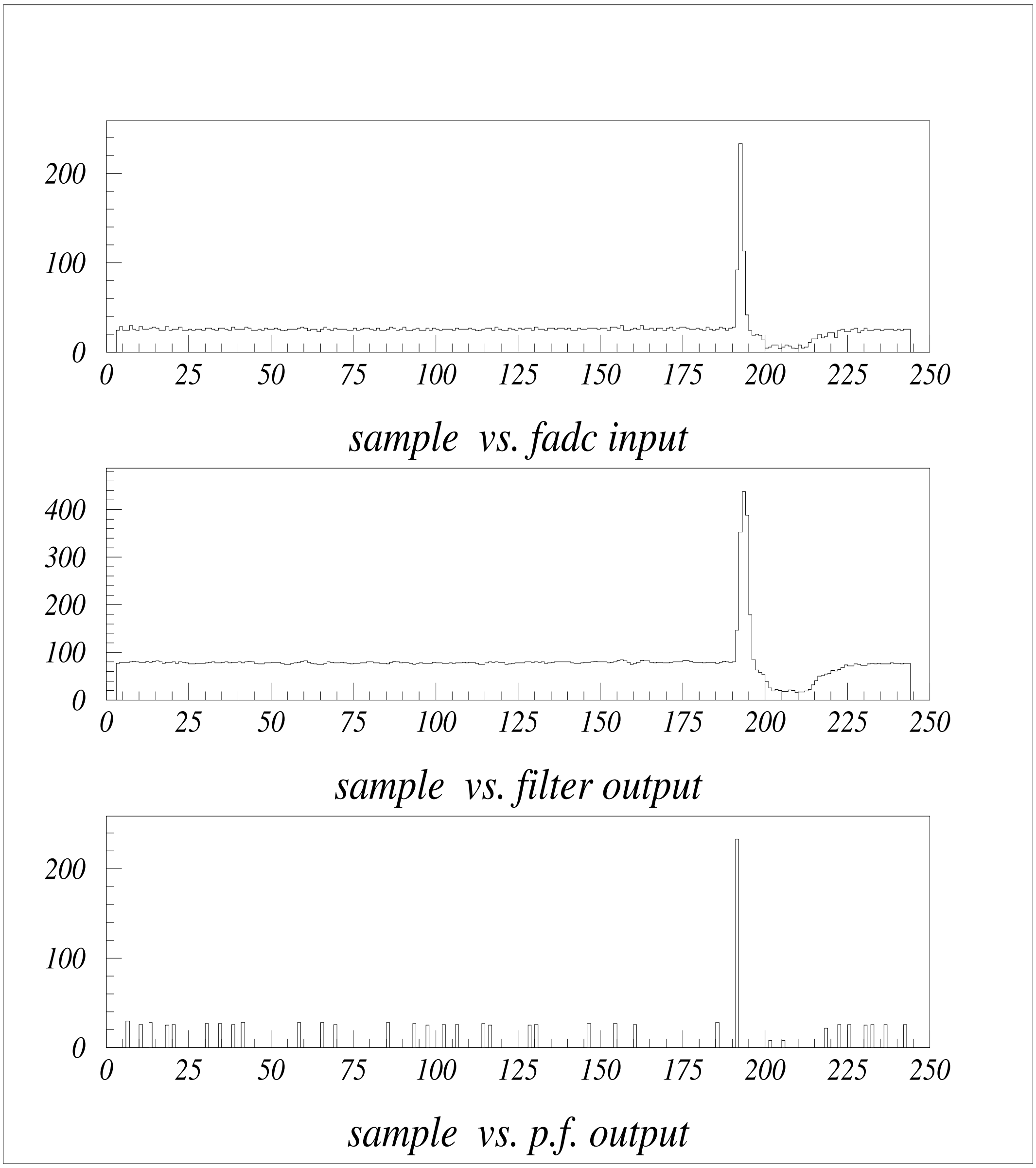


Figure 4: The value of each sampling after successive stages of the BCID process have been applied: upper) the raw ADC input data, middle) after applying the FIR algorithm (*fil1*), lower) final BCID output after peak-finding algorithm (*pf4*).

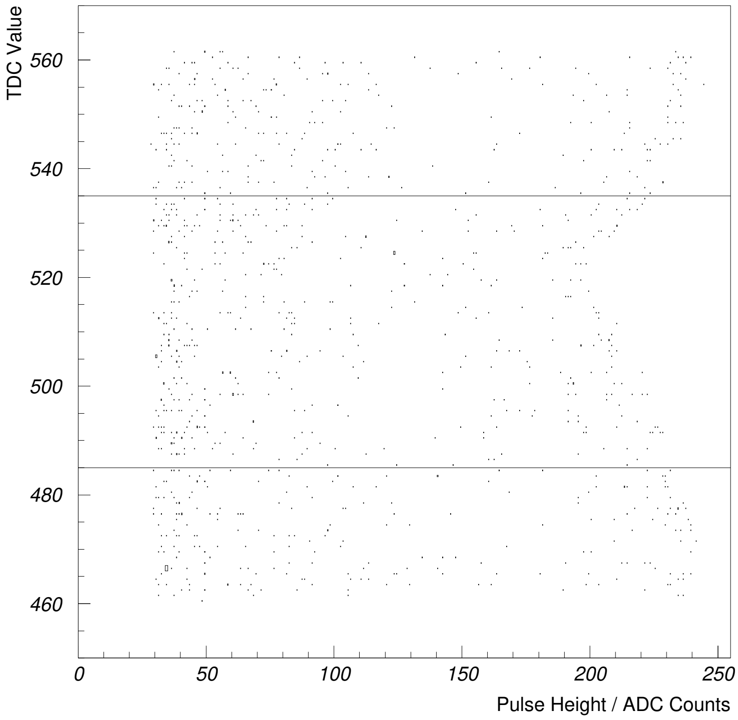


Figure 5: Pulse height maximum per event in ADC counts versus TDC value.

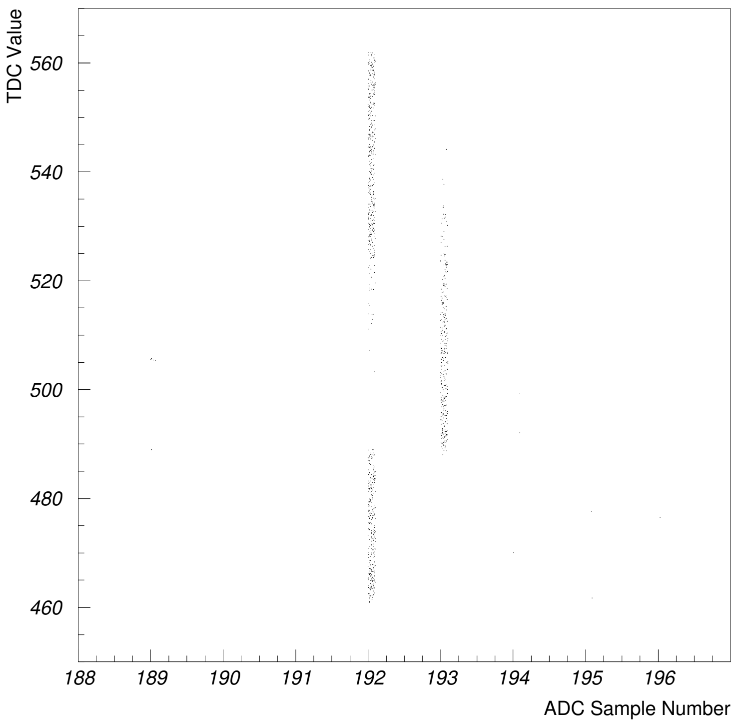


Figure 6: ADC sampling number containing the pulse maximum as found by the BCID module versus TDC value.

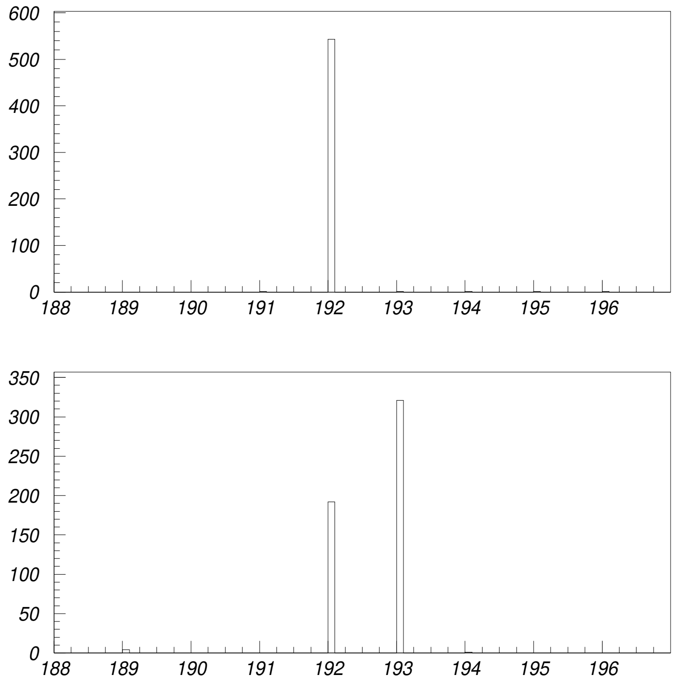


Figure 7: ADC sampling number containing the pulse maximum:
a) When using the TDC range described in the text, b) Data outside the TDC range.

6 Conclusions.

As far as can be discerned, the current single-channel BCID module has performed with almost 100% accuracy. Therefore digital implementations of peak finder and FIR filtering techniques can be successfully used on data at the full LHC speed of 40 MHz.

The RD27 phase-2 bit-parallel trigger demonstrator [11] will be a more comprehensive system than the current demonstrator. Its purpose will be to study several critical functions of the final level-1 system, i.e. high speed fibre optic links and serial–parallel data conversion. To accompany this, a phase-2 BCID module [12] will also be constructed. It will consist of 36 processing blocks – one for each input trigger-channel. The logic will be implemented in XiLinx [13] Field Programmable Gate Arrays. The phase-2 BCID module will be described more fully in a future note.

Acknowledgements.

We would like to express thanks to the RD3 collaboration for allowing us to test our hardware using signals from their calorimeter modules.

References.

[1] ATLAS letter of intent, CERN–LHCC/92–4.

[2] Nick Ellis et al., *A Calorimeter-Based Level-One Electromagnetic Cluster Trigger for LHC*, Proceedings of the International Conference on Computing in High Energy Physics 1992, CERN/92–07, 210.

RD27 status report, CERN–DRDC/93–32.

[3] GEM Technical design Report, GEM–TN–93–262.

[4] *A Digital Front-end and Readout Microsystem for Calorimetry at LHC*, CERN–DRDC/90–75, DRDC/P19, December 1990.

[5] RD27 proposal, CERN–DRDC/92–17.

[6] F. Gianotti et al., *The liquid argon Accordion electromagnetic calorimeter of the ATLAS experiment at the LHC*, to be published in Proceedings of the IVth. International Conference of Calorimetry in High Energy Physics, Elba, Italy, September 1993.

RD3 proposal, CERN–DRDC/90–31.

[7] I.P. Brawn, *Preliminary Investigations into Bunch-Crossing Identification for the Level-1 Trigger*, RD27 note 11.

I.P. Brawn, *Further Investigations into Bunch-Crossing Identification for the Level-1 Calorimeter Trigger*, RD27 note (unpublished).

[8] I.P. Brawn et al., *Beam Tests of a Prototype Level-1 Calorimeter Trigger Processor for LHC experiments*, Nuclear Instruments and Methods A349 (1994) 356–366.

[9] R.E. Carney et al., *The Level-1 Bit-parallel Electromagnetic Trigger Processor. CERN Beam Tests – November 1992*, RD27 note 12.

[10] Y. Ermolin, *Transparencies shown at RD27 (UK) meeting,* RAL, 13th. October 1993.

[11] V. Perera, *Requirements and Specification Document for the First-level Calorimeter Trigger Processor Demonstrator Phase-2,* RAL internal documentation, 1993.

[12] I. Brawn, *Requirements and Specification Document for a First-level Bunch-Crossing Identification Demonstrator System* , RAL internal documentation, 1993.

[13] *The Programmable Logic Data Book,* XiLinx Inc., 1993.