The DØ Silicon Track Trigger
Track Fitting Design

John Hobbs, Charles Pancake, Wendy Taylor
SUNY – Stony Brook

June 6, 2001 (Version 9)

Abstract
This note describes the fitting algorithm and custom hardware implementing the track fitting portion of the DØ Silicon Track Trigger. The physics constraints and performance are also summarized.

Contents

1 Algorithm, Physics Performance ........................................ 2
  1.1 Fit Algorithm ......................................................... 2
  1.2 Beam Position Correction ........................................... 3

2 Algorithm, Implementation .............................................. 3
  2.1 Execution Time ....................................................... 4
  2.2 Implementation ..................................................... 5

3 Fitting Hardware .......................................................... 7
  3.1 Data Flow ............................................................. 8
  3.2 Memory Maps ........................................................ 11
    3.2.1 Registers ......................................................... 14
    3.2.2 Data Transfer: Event Data and Dual Port Memory Layout ... 16
    3.2.3 Data transfer: Internal FIFO transfers ....................... 18
    3.2.4 Accessing External Memory from a DSP ...................... 19
  3.3 Control Logic – Partial description .............................. 20
  3.4 Initialization ....................................................... 20
    3.4.1 Overall Initialization ......................................... 20
    3.4.2 Programmable Logic Initialization .......................... 23
  3.5 Monitoring .......................................................... 23
  3.6 Error Handling – Place holder .................................... 25
  3.7 Power Consumption ................................................ 25
  3.8 Cost Estimate ....................................................... 27

4 Summary ................................................................. 27
This document presents a detailed description of the DØ silicon track trigger (STT[1])
track-fitting algorithms and electronics. The first section contains a summary of the physics
issues driving the algorithm. The second section describes timing and details regarding the
software implementation of the algorithm. The final section describes the hardware design
and monitoring plans for the track fit card (TFC).

1 Algorithm, Physics Performance

The physics choices driving the fitting algorithm selection have been described extensively
elsewhere[2][3], as has the detector geometry[4][5]. This section contains a summary of the
fitting algorithm and a description of the correction arising because the beam will not be
perfectly centered in DØ.

1.1 Fit Algorithm

The major points of the fit algorithm are recapped here. A $\chi^2$ fit is performed in the $r\phi$ plane
using data from the barrel detectors. The fit function is a linearized form of the equation
of a circle expressed in terms of track curvature $\kappa$, impact parameter $b$\(^{1}\) and track direction
$\phi_0$ at the point defining the impact parameter. The fit function, in polar coordinates $(r, \phi)$,
is\(^{4}\)

$$\phi(r) = b/r + \kappa r + \phi_0. \tag{1}$$

The dominant source of confusion in the fitting is selecting which hits belong to the
track. Tracks are seeded using CTT tracks found by the level one trigger, and only the silicon microstrip tracker (SMT) strips within a 2 mm road of the central track trigger (CTT)
tracks are allowed to contribute to the track. The average number of hits in a layer of
silicon in a given road is significantly more than one. A number of different hit selection
algorithms have been tried[3], and most have similar performance. The most significant
variation is whether all four SMT layers are required to have hits, or if tracks having hits
from only three layers are allowed. Permitting three layers increases the acceptance by
roughly 15%. The final algorithm for hit selection thus uses either four-layer or three-layer
candidate tracks, and within each layer, the hit closest to the circular trajectory defined by
the two central fiber tracker (CFT) points (from Level 1) and the origin is used in the fit.
In addition the hits must either all be in the same barrel segment or in adjacent segments
in a logically consistent manner. If a four-layer fit is possible, it is made. If the resulting
$\chi^2$ is unacceptably large, the fit is repeated, this time without the hit having the largest
contribution to the original four-layer $\chi^2$. This combination of hit selection and two-pass
fitting is called the “Static Road/3a” algorithm.

The main output from the silicon track trigger (STT) is the impact parameter $b$. The
dominant resolution effects on the impact parameter are a contribution of roughly 30 $\mu$m
from the beam spot size and detector resolution of 15 $\mu$m. The fitting algorithm numerical
precision introduces negligible change on these scales.

\(^{1}\)The magnitude of the impact parameter is the shortest distance between the track and origin, and the
sign is determined by the particle charge.
1.2 Beam Position Correction

The TFC uses the detector coordinates to reconstruct the track impact parameter with respect to the detector origin. However, the impact parameter relevant to physics is that measured with respect to the $p\bar{p}$ interaction point. We approximate this by the beam spot position in the $x-y$ plane. The geometry of the situation yields the formula for the corrected impact parameter

$$y_{\text{corr}} = \frac{\text{sign}(\kappa)}{2\kappa} - \sqrt{A^2 + r_B^2 + 2Ar_B \sin(\phi_B - \phi_0)},$$

where $(r_B, \phi_B)$ are the coordinates of the beam spot and $A = \text{sign}(\kappa) b - 1/(2\kappa)$. The square root in the formula is replaced by the lowest-order term of a Taylor expansion and the terms quadratic in $\kappa$ are neglected, yielding

$$b_{\text{corr}} = b + \text{sign}(\kappa)r_B \sin(\phi_B - \phi_0).$$

The $\sin(\phi_B - \phi_0)$ will be determined using a small lookup table.

Similarly, we can correct the $\phi_0$ of the track using

$$\phi_{0\text{corr}} = \phi_0 - \arctan \left( \frac{r_B \cos(\phi_B - \phi_0)}{r_B \sin(\phi_B - \phi_0) + A} \right).$$

The right-hand term is small for large $A = \text{sign}(\kappa) b - 1/(2\kappa)$. Assuming the worst case scenario of $\kappa = 0.002$ cm$^{-1}$, $b = 0.2$ cm and $r_B = 0.1$ cm, we find that the correction term is less than 0.0004, which is almost two orders of magnitude smaller than the 8-bit precision of the $\phi_0$ output by the TFC. Clearly, we can neglect such a correction for all reasonable beam spot offsets.

The beam position will have to be determined immediately before the run begins and conveyed to each TFC as part of the general the run–specific download. The download should be the $(r, \phi)$ position of the beam spot at the center of each of the six SMT barrels. The processor used for the fitting suggests a 16–bit integer format for these numbers. There seems to be little utility in decreasing the size, so we plan to receive six 32–bit numbers, with the lower 16–bits of each being the radius (in $\mu$m) and the upper 16–bits the $\phi$ divided into equal bins of $2\pi/2^{16}$ (chosen to maximally pack the 16 bit format). This format gives significantly more precision than is necessary, so if download times become too large, it can be made more compact. The TFC will be placed in initialization/configuration state prior to the download, and the vertex position data will be written to all processors.\(^2\)

2 Algorithm, Implementation

The fitting algorithm has three phases:

- hit selection
- parameter calculation, and
- $\chi^2$ calculation

\(^2\)See section 3.2.
<table>
<thead>
<tr>
<th>Processor</th>
<th>Algorithm Type</th>
<th>Execution time (μs)</th>
<th>1 fit</th>
<th>2 fits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Altera 10k130E</td>
<td>float, no LUT</td>
<td>36</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>Altera 10k130E</td>
<td>float, w/LUT</td>
<td>18</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td>TI6701@167 MHz</td>
<td>float, no LUT</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TI6701@167 MHz</td>
<td>float, w/LUT</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TI6202/3@300 MHz</td>
<td>integer, w/LUT</td>
<td>7.4</td>
<td>8.7</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Fitting algorithm execution time per road in various processors.

This section begins with a brief description of each phase. The next portion details the approximations made to arrive at an algorithm which executes in a reasonable time. The final portion gives the formats and sizes of the major tables (arrays) used in the calculation.

The fit uses at most one hit from each of the four SMT layers. Most of the time, however, some layers have more than one hit in the road. The hits used in the fit are those which are closest to the center of the road. The center is defined as the circle passing through the origin and the centers of the A- and H-layer CFT fibers defined by the input CTT road.

The parameter calculation[4] can be reduced to solving the matrix equation

\[ \vec{p} = M \cdot \vec{\delta \phi} \]  

in which \( \vec{p} \) is the 3-element vector of the fit parameters, \( M \) is a \( 3 \times (N_{hit} - 1) \) matrix depending only on the hit radii and resolutions and \( \vec{\delta \phi} \) is an \( N_{hit} - 1 \) element vector containing hit \( \phi \) residuals. The residuals are defined by subtracting the \( \phi \) for one of the hits from all other hits. Thus, one of the hits is guaranteed to have \( \phi \equiv 0 \), and the others are nearly zero. The final track \( \phi_0 \) is obtained by adding back the offset subtracted at the beginning. This method permits a significant reduction in the number of bits required to represent the angles when performing the fit.

The \( \chi^2 \) is evaluated using the form

\[ \chi^2 = \sum_{i=1}^{N_{hits}} c_i \left( (b - \kappa r_i^2 + (\phi_0 - \phi_i) r_i)^2 \right) . \]  

Here \( c_i \) is the inverse of the square of the hit resolution in length units.

### 2.1 Execution Time

An exhaustive number of processing options have been considered, including standard DO DEC/Compaq alpha processors, fully custom programmable logic devices and digital signal processors(DSP’s). Both floating point and integer versions of the calculation have been tested. Table 1 gives the per track fit times for a variety of processors using the “Static Road/3a” algorithm. The final processor choice, based on execution time, cost, processor density and programming simplicity, is the TI 320C6203 16-bit integer DSP. An integer form of the algorithm with matrices stored in a large look up table (LUT) will be used.

We expect 2.3 CTT tracks\(^3\) per 60° sector on average, and six CTT tracks in the highest occupancy sector (again, on average), and we have roughly 50 μs per event for all fitting.\(^4\)

---

\(^3\)Based on luminosity weighted QCD events.

\(^4\)See the results from the queueing simulation elsewhere.
Thus, a single processor is insufficient for the task. We will build dedicated processor cards each of which processes all tracks in a 30° \( \phi \) sector. Each card will have eight DSP’s. If all tracks in a given event require two iterations but assuming no more than one track per processor, it takes 16 \( \mu s \) to process one event. This time includes a contribution of 5.8 \( \mu s \) from worst case bus contention and assumes the maximum allowed hit counts in each road. When more than eight tracks are present, the additional fits will be performed as individual processors become available after completing their first fit. The hardware is described in more detail in the following sections.

### 2.2 Implementation

The best processing times come from the C6203 family of DSP’s running an integer form of the algorithm. The integer form is used because floating point arithmetic is emulated in software in the C62 family resulting in unacceptably long execution time if a floating point form is used. Additionally, were the highest precision integer arithmetic used, the dynamic range would be insufficient to calculate the matrix \( M \) on-the-fly, so the matrices will be computed offline and stored in a look up table. The DSP naturally performs 16 bit integer multiplies and 32 bit sums. Given this, the algorithm is designed to use matrix elements and hit \( \phi \) positions which require only 16 bits. In addition, we currently assume that no more than 58 hits are allowed per road. Monte Carlo studies indicate this is more than enough.\(^5\)

Figure 1 gives a comparison between the fit results obtained using a full floating-point calculation and the results from an integerized form. For the integer calculation the matrices were initially calculated in double precision and then converted to the 16 bit integers used in the calculation. The distributions shown are the differences between the integer-based result and the full floating-point result. The data are tracks in the SMT acceptance from 2500 \( WH \rightarrow q\bar{q}b\bar{b} \) events. This version of the integer algorithm was designed to produce answers with 1 \( \mu m \) precision for impact parameter, 1 mrad precision in \( \phi_0 \), and the curvature result in 256 even bins in the range \(-0.003 \leq \kappa \leq 0.003 \) cm\(^{-1}\). The results for \( b \) and \( \phi_0 \) are substantially more precise than physics demands: the beam spot alone contributes 30 \( \mu m \) to the impact parameter resolution the high-\( p_T \) track resolution is 15 \( \mu m \), and the standard \( \phi \) resolution in the trigger is 8 bits or roughly 24.5 mrad. The curvature binning was chosen to match the target output format.\(^6\)

The previous figures showed the precision of an integer calculation. For those plots the coefficient matrices \( M \) were calculated using the true hit radii and then converted to integer format for the calculation. If all possible matrices were computed in this manner and stored in a matrix look up table (MLUT), the table would be far too large. Each road requires a set of matrices because different assigned hit patterns result in numerically different matrices.\(^7\)

If a complete set of matrices were computed for each road, the MLUT for a 30° sector would be 0.5 Gb, clearly too large to be practical. Matrix sets will instead be (pre)computed with one matrix for a range of roads. Figure 2 shows the reconstructed impact parameter distribution for a look up table consisting of 160 \( \phi \) super roads, indexed only by H-layer.

---

\(^5\)The upper limit is determined by the size of buffer memory on the TFC. The main constraint on the memory size is not the upper bound number of hits/road, which could reasonably be as small as 32, but rather by the desire for efficiency during initialization when the memory is used as a transfer buffer between the PCI bus and DSP memory and the Matrix LUT.

\(^6\)It could easily be made more precise if needed, for example, in the \( \chi^2 \) calculation.

\(^7\)For example, the matrix for a four-layer track is different than that with the outer layer not used, and the matrix also depends on which SMT barrels the track crosses.
Figure 1: The difference in track parameter between the integer and double precision calculations. The resulting RMS values are significantly smaller than what is needed for the trigger. The lower right hand plot indicates whether the hit assignment in the integer algorithms is different from (zero bin) or the same as (one bin) if it were determined using a floating point calculation.
Figure 2: The track parameter resolutions calculated using a matrix look up table having 160 separate $\phi$ slices. The tracks used here were generated with a simple simulation having the correct CTT and SMT geometries but no multiple scattering or false hits. Tracks in the ranges $-0.2 \text{ cm} \leq b \leq 0.2 \text{ cm}$ and $p_T > 1.5 \text{ GeV}$ were generated. For the upper right hand plot only tracks satisfying $p_T > 3 \text{ GeV}$ and $|b| < 0.1 \text{ cm}$ were used. See the text for a discussion of the distribution widths.

fiber. Within a given super road, all hits in a given sublayer are assumed to come from the same radius. This approximation clearly breaks down for sufficiently wide roads. One sees that the impact parameter distribution in fig. 2 is somewhat wider than the 15 $\mu$m expected in the absence of multiple scattering. This has occurred for tracks with either $p_T < 3 \text{ GeV}$ or $|b| > 0.1 \text{ cm}$. For both of these cases the additional width is not likely to affect the results. However the effect can be substantially reduced by using the CFT A-layer offset in addition to the H-layer in defining the super road.

3 Fitting Hardware

The track fitting card (TFC) will be a custom daughter board for the standard STT motherboard. The inputs to the TFC from the fiber road card (FRC, one channel) and silicon trigger card (STC, 6 channels) will be received on low voltage differential serial (LVDS) links using the standard STT PCMPIP receivers. The single output channels from a TFC will use a Cypress Hotlink based transmitter. One of the transmitter/receiver-equipped motherboard
PCI busses will be dedicated to the inputs, and the second will be dedicated to L2 output. The third PCI bus will be used for output to L3. The block diagram of the TFC is shown in figure 3. An input controller will read all input data, reformat it on-the-fly to provide SMT coordinate and format conversion, and store it in a DPM memory prior to transfer to a DSP for fitting. The input controller will also act as DSP input bus arbiter. When the event data is fully loaded into the input dual port memory (IDPM), data for individual tracks will be sent to DSPs by logic in the input controller. When a DSP is finished processing a track, the data will be written by the DSP to a predetermined location in an output dual port memory (ODPM). At this point the DSP can be loaded with data for another track. The new track can be from either the same event or, optionally, from a new event. When all tracks for a single event have been processed, the output controller will read the fit results from the ODPM, add headers and trailers, and transmit completed fit results to L2 and L3. Communication between the input and output controllers will provide event synchronization and DSP scheduling. The input and output controllers will be implemented in programmable logic.

The eight DSP’s on a TFC are arranged as two independent “columns” of four DSP’s. The DSP’s within a column share a common input bus and a separate common output bus. The DSP expansion bus will be used as the input (and initialization) bus, and the DSP external memory bus will be used as the output and external memory bus. The IDPM and ODPM thus each have two independent banks, one for each column. Track data will be sent to both columns alternating between the two for load balancing. All data will be in little-endian format, corresponding to that used in the L2 global processor.

The detailed hardware description is divided into seven sections. The first section describes in detail the data flow outlined above. The second section gives the system memory map for external PCI access, and the third section describes the control logic used for data routing and processor scheduling. The fourth and fifth sections present the monitoring, error handling and board initialization. The sixth section describes the power requirement. The final section is a cost estimate.

### 3.1 Data Flow

The input controller will first read the L1CTT data transmitted from the FRC, and then read all channels of STC data. Both FRC and STC data will be read from the input LVDS receiver boards using the “event” format, and data availability will be detected using user-defined pins on the PMC/PCI interface.

The L1CTT information will be copied unmodified to predetermined locations in the input dual port memory (IDPM)\(^8\) and copied to a DSP with the rest of the road data. Reformatting needed during the fit algorithm will be handled by the DSP code using internal look up tables.

The SMT hit positions will be loaded into the IDPM after undergoing an on-the-fly coordinate conversion from hardware address to the coordinates used in the fitting. The conversion will be provided by a look up table having the format given in table 2. The 20-bit look-up table address is formed from the input STC data word as shown in table 3. Because the conversion occurs during the read/store cycle, no additional time is lost. The three bits of hit dE/dx information in the STC input word will be stored in the IDPM with the 28 bits of coordinate information from the LUT.

---

\(^8\)See section 3.2 for the IDPM memory layout and data format
Table 2: SMT physical address to fit coordinates look up table format. Three additional bits are reserved for the dE/dx data in the STC input hit data. The remaining bit is reserved to augment the layer count in the event that additional SMT layers are added. The radius is an offset to the minimum radius for each ladder. A ladder number can be extracted from the layer, barrel and \( \phi \) values.
Table 3: The address for the coordinate conversion look up table. The memory has only 20 bits of address, so the eight bit sequencer ID has been arranged such that within a 30° sector, only four (7,2,1,0) bits are needed. The remaining sequencer ID bits are checked against the expected value. This leaves one available hardware address bit for future expansion.

All L1CTT and STC data for a given road are stored in consecutive memory locations in the IDPM. Because the first and last hits to arrive from the STC’s may belong to the same road, fitting cannot begin until the hits have all arrived. Once the hit transmission has finished, road data will be pushed from IDPM into DSP internal data memory with one road per DSP. The hits for each road are stored in consecutive locations, so the reads from buffer to internal DSP memory are implemented using DMA mode. The transfer efficiency will be reduced somewhat because the four DSP’s in one column share a common input bus. An upper bound on time needed to read from the buffers into the four DSP’s sharing a bus is 5.6 μs, including bus arbitration. This is for the case in which all roads have the maximum number of hits present. In general, the number will be three to four times lower.

The fit algorithm is then run independently in each DSP. Each fit requires one or two accesses to a look up table containing the coefficient matrices M. A look up table will hold all M’s for a single 30° sector, and each TFC will have two copies of the LUT, one for each column. All 15 elements for a single matrix are stored in 8 consecutive 32-bit words, with two elements per word. The SMT radial assembly precision is such that barrel-to-barrel variations must be taken into account. Because of this, the matrix used for a given road will depend not only on whether a layer is skipped, but also on the barrels in which the hits occur. Table 4 gives the address format for the matrix look up table. The road portion of the address requires 17 bits and each matrix has 8 32-bit words adding 3 bits, for a total memory requirement of 1M addresses of 32 bit words or 4 Mb.

The address size can be reduced, if needed by defining the sublayers uniquely for each road. In this case, the sublayer indices can be dropped. The total memory requirement for the MLUT is then only 256 kb. The 6203 DSP has 512 kb of internal data memory and the matrix look up could fit wholly in internal memory, eliminating external memory access during fitting.

When processing is finished, the DSP requests the output data bus, and the fit results are written to a DPM memory area using an address determined by the input controller and placed in fields in the road data initially copied to the DSP. The input controller is then notified that the DSP is free, and if roads remain to be processed, new road data can be copied to the DSP and processed.

All output formatting for the individual track is done as part of the fit. When all tracks in a given sector are finished, the data are read from the buffer by an output controller. A standard header and trailer are added, and the data transmitted to L2 via Cypress hot link. The header, data and trailer formats are shown in tables 5 to 7. The data type (L2 ID)

<table>
<thead>
<tr>
<th>19...16</th>
<th>15...13</th>
<th>12...9</th>
<th>8...0</th>
<th>bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seq ID</td>
<td>HDI</td>
<td>Chip</td>
<td>Strip</td>
<td>field</td>
</tr>
</tbody>
</table>

\footnote{Matrices for fits with a missing layer require only 12 elements and could be stored in 6 consecutive words. For simplicity, these will be stored in 8 words.}

\footnote{All other look up tables currently envisaged for the DSP algorithm require roughly 32 kb.}
<table>
<thead>
<tr>
<th># of bits</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Super Road ID (160 roads, 30° logical sectors)</td>
</tr>
<tr>
<td>3</td>
<td>Barrel index for hit in 4th layer</td>
</tr>
<tr>
<td>2</td>
<td>Barrel transition layer (0, no change)</td>
</tr>
<tr>
<td>1</td>
<td>Barrel transition direction, ±</td>
</tr>
<tr>
<td>4</td>
<td>Sublayer hit within each layer</td>
</tr>
<tr>
<td>1</td>
<td>Skipping a layer (Y/N)</td>
</tr>
<tr>
<td>2</td>
<td>ID of skipped layer</td>
</tr>
<tr>
<td>3</td>
<td>Matrix element pair (See caption)</td>
</tr>
<tr>
<td>20</td>
<td>TOTAL = 1M addresses of 32-bit words (4 Mb memory)</td>
</tr>
</tbody>
</table>

Table 4: Matrix inverse look up table address format. The look up table is assumed to be generated separately for each 30° sector. Each matrix has 15 elements of 16 bytes. The lowest order 3 addressss bits are used to access individual element pairs. A sector actually covers $360°/8 = 45°$.

field contains a preassigned 8-bit number identifying the data source. The TFC data types are 162 - 173 inclusive. The highest six numbers will be used in those TFC’s which also retransmit the L1CTT data. The STT will have 8 bytes of data per road. Using data from reference [3], the average sector has 16 bytes of fit output, and the highest occupancy sector will have 48 bytes.

The ODPM and matrix look up table share the DSP external memory interface, so matrix reads and data output cannot occur simultaneously. However, each look up and output write are expected to take less than 1.0 µs, to be compared with the overall processing time of roughly 10 µs. It appears that this will not cause problems.

In addition to the STT information with its header/trailer, the original L1CTT information will also be transmitted from the first of the two TFC’s in a sector. Thus, one cable will have both L1CTT and STT header/trailer blocks for each event, and the second will have only the STT information. A number of other options have been considered, but lead to nearly doubling the transmitted data size at high luminosity and also require additional hardware complexity. Transmission of the L1CTT data will be controlled by a bit in the TFC global configuration register.\textsuperscript{11}

The level 3 output data for normal events will be a copy of the level 2 information. Under special conditions signaled by serial command link (SCL) qualifier bits inserted in the original data transmission from the FRC, the input SMT hits, converted SMT hits and matrix LUT addresses will also be sent. The number of 32-bit words transmitted in this case is $(1 + 1) \times \Sigma_{\text{roads}_h} N_h + (1 + 3) \times N_{\text{roads}_h}$. For those events, the average data size will be roughly 160 bytes per sextant, with a strict per sextant maximum of 11.4 kb.

### 3.2 Memory Maps

This section describes the TFC and DSP memory layouts. The PCI accessible memories and registers are shown in table 8. The DSP memory map is shown in table 9. Detailed descriptions of data formats not specified elsewhere are given in remainder of this section.

\textsuperscript{11}See section 3.2.
<table>
<thead>
<tr>
<th>Word</th>
<th>Bits</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-7</td>
<td>Object Count</td>
</tr>
<tr>
<td></td>
<td>8-15</td>
<td>Header Length (DWORDS) = 3</td>
</tr>
<tr>
<td></td>
<td>16-23</td>
<td>Object Length (DWORDS) = 2</td>
</tr>
<tr>
<td></td>
<td>24-28</td>
<td>Object Format version</td>
</tr>
<tr>
<td></td>
<td>29-31</td>
<td>Header/Trailer Format version</td>
</tr>
<tr>
<td>2</td>
<td>0-7</td>
<td>Data Type, L2 ID</td>
</tr>
<tr>
<td></td>
<td>8-15</td>
<td>Bunch #</td>
</tr>
<tr>
<td></td>
<td>16-31</td>
<td>Rotation #</td>
</tr>
<tr>
<td>3</td>
<td>0-7</td>
<td>Algorithm Major version</td>
</tr>
<tr>
<td></td>
<td>8-15</td>
<td>Algorithm Minor version</td>
</tr>
<tr>
<td></td>
<td>16-23</td>
<td>Processor specific bits</td>
</tr>
<tr>
<td></td>
<td>24-31</td>
<td>Error bits</td>
</tr>
</tbody>
</table>

Table 5: STT data block to Level 2 header format.

<table>
<thead>
<tr>
<th>Word</th>
<th>Bits</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-6</td>
<td>$p_T$, GeV (unsigned, encoded, $p_T &lt; 100$ GeV)</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>Sign ($\kappa$)</td>
</tr>
<tr>
<td></td>
<td>8-15</td>
<td>Impact parameter significance</td>
</tr>
<tr>
<td></td>
<td>16-23</td>
<td>$\phi_0$</td>
</tr>
<tr>
<td></td>
<td>24-28</td>
<td>$\chi^2$</td>
</tr>
<tr>
<td></td>
<td>29-30</td>
<td>spare</td>
</tr>
<tr>
<td></td>
<td>31</td>
<td>Fit failed</td>
</tr>
<tr>
<td>2</td>
<td>0-11</td>
<td>$b$, $\mu$m (signed, $</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>spare</td>
</tr>
<tr>
<td></td>
<td>13-15</td>
<td>dE/dx (binned)</td>
</tr>
<tr>
<td></td>
<td>16-21</td>
<td>CFT track index (0-45)</td>
</tr>
<tr>
<td></td>
<td>22-23</td>
<td>spare</td>
</tr>
<tr>
<td></td>
<td>24-25</td>
<td>truncated layer count</td>
</tr>
<tr>
<td></td>
<td>26-27</td>
<td>Skipped layer (if topology flag =1)</td>
</tr>
<tr>
<td></td>
<td>28-30</td>
<td>barrel number</td>
</tr>
<tr>
<td></td>
<td>31</td>
<td>SMT topology flag. $1 \Rightarrow 3$ layers</td>
</tr>
</tbody>
</table>

Table 6: STT data block to Level 2 format. Two 32-bit words are used per road. The beam position correction is made in the TFC. For $p_T < 25$ GeV, the encoded $p_T$ is in 0.25 GeV increments, and for $p_T > 25$ GeV, the interval is 3 GeV.
<table>
<thead>
<tr>
<th>Bits</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-7</td>
<td>Bunch #</td>
</tr>
<tr>
<td>8-15</td>
<td>data type (L2 ID)</td>
</tr>
<tr>
<td>16-23</td>
<td>Longitudinal parity (even bytes)</td>
</tr>
<tr>
<td>24-31</td>
<td>Longitudinal parity (odd bytes)</td>
</tr>
</tbody>
</table>

Table 7: STT data block to Level 2 trailer format.

<table>
<thead>
<tr>
<th>Name</th>
<th>Mode</th>
<th>Size(DWORDS)</th>
<th>Location</th>
<th>Base</th>
<th>Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coordinate IDPM, Banks 1+2†</td>
<td>R/W</td>
<td>32k (See Sec. ??)</td>
<td>PCI-1, BAR0</td>
<td>0x0</td>
<td></td>
</tr>
<tr>
<td>conversion LUT†</td>
<td>R/W</td>
<td>2M</td>
<td>BAR1</td>
<td>0x0</td>
<td></td>
</tr>
<tr>
<td>PCI data write†</td>
<td>W</td>
<td>1</td>
<td>BAR2</td>
<td>0x0</td>
<td></td>
</tr>
<tr>
<td>L2 Xfer FIFO†</td>
<td>W</td>
<td>1</td>
<td>BAR3</td>
<td>0x0</td>
<td></td>
</tr>
<tr>
<td>Input target BAR’s</td>
<td>R/W</td>
<td>N(STC)+1</td>
<td>BAR4</td>
<td>0x0</td>
<td></td>
</tr>
<tr>
<td>IMXFER register</td>
<td>R/W</td>
<td>1</td>
<td>BAR4</td>
<td>0x8</td>
<td></td>
</tr>
<tr>
<td>ODPM, Banks 1+2†</td>
<td>R/W</td>
<td>32k (See Sec. ??)</td>
<td>PCI-2, BAR0</td>
<td>0x0</td>
<td></td>
</tr>
<tr>
<td>Output target BAR</td>
<td>R/W</td>
<td>1</td>
<td>BAR1</td>
<td>0x0</td>
<td></td>
</tr>
<tr>
<td>output data†</td>
<td>R</td>
<td>1</td>
<td>BAR2</td>
<td>0x0</td>
<td></td>
</tr>
<tr>
<td>L2 Xfer FIFO†</td>
<td>R</td>
<td>1</td>
<td>BAR3</td>
<td>0x0</td>
<td></td>
</tr>
<tr>
<td>L3 Xfer FIFO†</td>
<td>W</td>
<td>1</td>
<td>BAR4</td>
<td>0x0</td>
<td></td>
</tr>
<tr>
<td>L3 data output</td>
<td>R</td>
<td>1</td>
<td>PCI-3 BAR0</td>
<td>0x00</td>
<td></td>
</tr>
<tr>
<td>L3 Xfer FIFO†</td>
<td>R</td>
<td>1</td>
<td>BAR1</td>
<td>0x00</td>
<td></td>
</tr>
<tr>
<td>MON_START</td>
<td>R/W</td>
<td>1</td>
<td>BAR2</td>
<td>0x00</td>
<td></td>
</tr>
<tr>
<td>MON_DONE</td>
<td>R/W</td>
<td>1</td>
<td>0x04</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCL_READY</td>
<td>R/W</td>
<td>1</td>
<td>0x08</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCL_DONE</td>
<td>R/W</td>
<td>1</td>
<td>0x0C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configuration</td>
<td>R/W</td>
<td>1</td>
<td>0x10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating Mode</td>
<td>R/W</td>
<td>1</td>
<td>0x14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Reset</td>
<td>W</td>
<td>1</td>
<td>0x20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Counter/Timer reset</td>
<td>W</td>
<td>1</td>
<td>0x24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Monitor data</td>
<td>R</td>
<td>10</td>
<td>BAR3</td>
<td>0x00</td>
<td></td>
</tr>
<tr>
<td>Monitor histograms</td>
<td>R</td>
<td>??</td>
<td>BAR4</td>
<td>0x00</td>
<td></td>
</tr>
</tbody>
</table>

Table 8: TFC PCI memory map. The three interfaces are labelled PCI-1, PCI-2 and PCI-3. During normal data taking PCI-1 handles input from the FRC and STC, PCI-2 is used for output to L2 global and PCI-3 is used for L3 output. Locations marked with a † are inaccessible when in data-taking mode, and the locations marked with a ‡ are used only when in PCI-based data-taking mode (See section ???.

13
<table>
<thead>
<tr>
<th>Memory</th>
<th>Size (DWords)</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDPM</td>
<td>32k</td>
<td>DSP is target (Host Mode)</td>
</tr>
<tr>
<td>Matrix LUT</td>
<td>2M</td>
<td>Memory Bus, CE0+CE1</td>
</tr>
<tr>
<td>ODPM</td>
<td>32k</td>
<td>Memory Bus, CE2</td>
</tr>
</tbody>
</table>

Table 9: TFC DSP external memory map.

3.2.1 Registers

Each TFC has a number of registers in the PCI address space space. The registers are listed as part of table 8, and are described in more detail in the following paragraphs. Overall control and configuration of the TFC’s is handled by the Configuration, Operating Mode and IMXFER registers as described below. The status registers are described in section 3.5. Writing any value to the Software Reset register begins the system reset\(^\text{12}\). Writing any value to the Counter/Timer Reset register clears all monitoring counters and histograms.

**Input BAR registers.** The read/write input BAR registers contain the PCI addresses at which the FRC and six STC input buffers are found. These input buffers reside on the LVDS receiver cards and are the target addresses for reading the input data. The contents of this register must be written following the PCI configuration phase but before any data-taking is enabled.

**Output BAR register.** The read/write output BAR register contains the PCI base address at which the Cypress transmitter resides. Data for the L2 global processor is written to this address for transmission. As with the input BAR registers, the contents of this register are written after the PCI configuration phase but before any data-taking is enabled.

**Configuration register.** The configuration register contains all information needed for normal data-taking mode. The main purpose of this register is to enable sector-specific data to be written at run time, thereby allowing the programmable logic (FPGA) and DSP code to be identical for all sectors. The configuration register bit definition is shown in table 10.

**Operating Mode register.** The Operating Mode register allows the TFC to be placed in a variety of modes. The fundamental two modes are initialization and data-taking. All other modes are for testing. In addition to specifying the mode, this register also allows selection of a single DSP (in each column) to be used for some test and initialization operations. Upon reset or power up, the TFC is placed in initialization mode. In this mode, no data-taking will occur, and the various memories can be read and written. When the mode is set to standard data-taking, all external accesses to the TFC memory except those defined as part of the read out and monitoring will result in a bus error. The Operating Mode register is defined in table 11, and the modes are listed in table 12.

**IMXFER Register.** The IMXFER register provides the hand shaking between a CPU and the DSP that is needed when loading DSP programs or accessing any of the locations in the DSP memory map shown in table 9. The IMXFER register layout is defined in table 13.

Writing data to the DSP memory map from an external CPU is accomplished by first by

\(^{12}\text{This is currently identical to that process initiated by an SCL\_INIT request.}\)
<table>
<thead>
<tr>
<th>Bits</th>
<th>Quantity</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-7</td>
<td>data type (L2 ID)</td>
<td>L2 output</td>
</tr>
<tr>
<td>11-15</td>
<td>RESERVED</td>
<td></td>
</tr>
<tr>
<td>16-19</td>
<td>STC input channel count = 9</td>
<td>L2 input</td>
</tr>
<tr>
<td>20-23</td>
<td>RESERVED</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>echo L1 CTT</td>
<td>L2 output</td>
</tr>
<tr>
<td>25</td>
<td>enable input pipeline</td>
<td>L2 input</td>
</tr>
<tr>
<td>26</td>
<td>enable fitting pipeline</td>
<td>L2 input</td>
</tr>
<tr>
<td>27</td>
<td>enable L3 output pipeline</td>
<td>L3 output</td>
</tr>
<tr>
<td>28</td>
<td>permit STT skip (TRIGQUAL)</td>
<td>L2 input</td>
</tr>
<tr>
<td>29</td>
<td>transmit empty roads</td>
<td>L2 input</td>
</tr>
<tr>
<td>30</td>
<td>transmit failed fits</td>
<td>L2 output</td>
</tr>
</tbody>
</table>

Table 10: Configuration register, definition.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-3</td>
<td>Operating mode</td>
</tr>
<tr>
<td>4-5</td>
<td>Select DSP, column #1</td>
</tr>
<tr>
<td>6</td>
<td>Enable column #1 (initialization and test modes)</td>
</tr>
<tr>
<td>7</td>
<td>Enable column #2</td>
</tr>
</tbody>
</table>

Table 11: Operating mode register, definition.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Initialize</td>
</tr>
<tr>
<td>1-11</td>
<td>RESERVED</td>
</tr>
<tr>
<td>12</td>
<td>Data-taking, L2 input/output via CPU write/read on PCI bus</td>
</tr>
<tr>
<td>13</td>
<td>Data-taking, L2 input from CPU, output to Hotlink transmitter</td>
</tr>
<tr>
<td>14</td>
<td>Data-taking, L2 input from LVDS receiver, output by CPU read</td>
</tr>
<tr>
<td>15</td>
<td><strong>Data-taking, L2 input/output using LVDS/Hotlink</strong></td>
</tr>
</tbody>
</table>

Table 12: TFC operating modes.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>14-0</td>
<td>Word counter (R/W)</td>
</tr>
<tr>
<td>15</td>
<td>R/W flag (W)</td>
</tr>
<tr>
<td>15</td>
<td>IDPM busy (R)</td>
</tr>
</tbody>
</table>

Table 13: IMXFER register, definition.
reading the IMXFER register and testing the busy flag (bit 15) to see if the transfer IDPM\textsuperscript{13} is available. When it is free, data to be transferred is written to the IDPM by the CPU. The first word of the IDPM memory is interpreted as a destination address\textsuperscript{14}, and the last word of the written to the IDPM is an interrupt word. All other words are treated as data and are written to DSP memory starting at the address specified in the first word. The IMXFER register is then written with the number of data words plus one and with the R/W flag (bit 15) = 0. If the least significant bit in the interrupt word is set, the DSP interrupt DSPINT is raised at the end of the transfer, and the processor is restarted. All other bits in the interrupt word are ignored.

Reading data from the DSP memory map is accomplished in a similar manner. First the IMXFER register is read, and the busy flag is checked to see if the IDPM is free. The DSP memory source address is then written to the first word of the IDPM. The IMXFER register is then written with the word count and with the R/W flag = 1. The CPU must then poll the IMXFER busy flag to see when the transfer from the DSP to IDPM is complete. When the transfer is done, the data may now be directly read from the IDPM, starting at the lowest address.

The transfer from the IDPM to the DSP memory takes place on the DSP expansion bus under control of logic in the PCI-A programmable device. The full memory address range of the ‘6203 is accessible through this means.

3.2.2 Data Transfer: Event Data and Dual Port Memory Layout

There are four individual dual port memory\textregistered (DPM) areas on the TFC, two for input (IDPM) and two for output (ODPM). Each of the two DSP columns has one area of each type. In normal data-taking mode the DPM’s are logically divided into fixed length blocks with each block holding data for one event. Each event block is subdivided into additional fixed size blocks with each of these holding data for a single road. Because one column holds even numbered tracks, and the other holds odd numbered tracks, no DPM will ever hold more than half of the tracks for a given event. Each road data block is 64 DWORDS long, and each event block contains up to 32 road data blocks (of which only 46/2 = 23 will ever be used). Thus, the 32k DWORD memories can hold data for up to 16 events. The general DPM layout is shown in figure 4. The data format for a single road differs between the input and output DPM banks. Figure 5 shows the road data layout for both input and output banks.

Input road data has a “packed word” for each road. The field layout is shown in figure 6. The field data are defined as the following:

- **Hit Count** Number of SMT $r_{\phi}$ hits in the road.

- **Flag bit** In processing bit. This flag is clear when the road data is written into the DSP, and it should be set by the DSP when the road processing begins. After the DSP processing finishes, the DSP program should wait for this bit to be cleared before processing the next road.

- **CTT Road No.** CTT index of the current road (0-45). The index is assigned for the event using the order in which the road data is received from the FRC. The first road

\textsuperscript{13}See section 3.2.2.

\textsuperscript{14}The destination address is a byte address, but the lower two bits must be cleared. This implies all transfers word aligned.
Figure 4: Division of the DPM into event and road blocks. The road indices shown correspond to one DSP column of the TFC. The second column will hold data for the even numbered roads. This division pertains during normal data taking. When in initialization or test modes, each DPM is addressed as a normal 32k DWORD memory.

Figure 5: Data format for a single road. Both the input and output DPM formats are shown. The lighter colored entries in the output road data are present only for unbiased or forced-write triggers. The “packed word” in the input DPM is described in Fig. 6.
Figure 6: The fields in the packed data word of the road data transmitted to a DSP. The field definitions are described in the text.

in a given event has index = 0; the second has index = 1 and so on. The assignment is made by the event loader control logic.

- **Road Out & Event ID**. The data output address (within the ODPM) is formed by the logical operation Event ID << 11 + Road Out << 6.

- **TQ** Trigger qualifier bits extracted from the SCL information

  Bit 29 = Unbiased or ForcedWrite
  Bit 30 = CollectStatus
  Bit 31 = L2STTNeeded

All DPM's are directly accessible to external sources via the PCI bus\(^\text{15}\). DPM's of the same type in both columns appear in the same 32kb address range; the IDPM's from both columns overlap with each other, and the ODPM's from both columns overlap each other. This implies that the DPM's in the two columns can be simultaneously accessed. Accesses to a given column are controlled by the column enable bits in the *Operating Mode* register. The only ill-defined situation arises when a read occurs with both columns enabled. In this case, the data from column 1 is transferred, and no information from column 2 is gotten.

**IDPM's used to transfer data into DSP memory** The IDPM's are also used to transfer data to memory in the DSP memory map, using the method described in the IMXFER register definition in section 3.2.1 above. This mechanism is used for all transfers to DSP addresses, including the matrix LUT and internal DSP program and memory.

### 3.2.3 Data transfer: Internal FIFO transfers

Two FIFO's are used on the TFC. The first is used for copying data from the TFC L2 input controller to the L2 output controller, and the second is used for copying data between the L2 output PCI interface and the L3 output interface.

The data sent from the input controller to the L2 output controller includes the original L1CTT information via the FRC and, for unbiased monitor events, the input STC data. The data sent from the L2 output controller to the L3 FIFO is a copy of the STT data sent to the L2 global processor, and for unbiased triggers it also contains the STC input data, and an augmented the full road data record.\(^\text{16}\) For this FIFO, the data for an event are in blocks of L1CTT, STC and TFC in this order. The STC block is present only for monitoring events.

\(^{15}\)When the *Operating Mode* register is set to a data-taking mode, this access is disabled, and access to the DPM’s is controlled by the InputController and L2OutputController FPGA’s.

\(^{16}\)See section 3.5.
Table 14: Internal FIFO data format. The bit labelled LD is set in the last word of each of the LiCIT, STC and TFC data blocks (for each event). The field labelled “TQ Flags” contains flags extracted from the full trigger qualifier word. The flags are used to indicate an unbiased read out event, a monitoring event and an event for which STT processing is optional.

Both FIFO’s are 64k x 36 bits. The upper four data bits will be used for out-of-band data indicating start-of-event, LiCIT data, STC data, FIT data, end-of-event and a last-word-of-type flag. The start and end record also contain a 4-bit number indicating event region in the DPM’s used for the road data and the trigger qualifier bits pulled from the FRC header. The various formats are shown in figure 14.

3.2.4 Accessing External Memory from a DSP

The memories for the matrix look up table and output data buffer(ODPM) used by the DSP’s are external to the DSP’s. The memory for these functions is shared by the four DSP’s in a given column, so arbitration must be used to prevent simultaneous access to the same memory from multiple DSP’s. The ’620x series provides limited support for sharing the external memory interface bus. The DSP has a simple hold-off which can be asserted by an external bus master via a pin on the DSP package. When an external master requires the bus, it drives the hold-off pin (/HOLD), and the DSP tristates its external memory interface pins and drives /HOLDA. There is no dedicated bus request line from the DSP which would allow the DSP to request the bus. We have thus implemented our own software arbitration scheme based on this hold-off interface and a single I/O pin on the DSP arbitrarily defined as a bus request line. The scheme is described in the remainder of this section.

The bus arbitration is performed by logic in (the PCI-B) programmable logic. Under normal circumstances, the arbitration logic asserts the bus hold-off to all DSP’s, thereby preventing any DSP from accessing external memory. When a DSP needs to use the external memory, it asserts a bus request signal on a dedicated I/O pin, the DX1 McBSP pin\(^\text{17}\). The external arbiter waits until no DSP is using external memory, and then removes the hold-off for the DSP which requested the bus. Once the DSP has made the request, its program must poll the /HOLD and /HOLDA flags (bits 8 and 9 ) in the EMIF global control register (at address 0x01800000).\(^\text{18}\) When both flags equal zero, the DSP has ownership of the external bus, and can make as many transfers as desired. When all transfers are completed, the DSP

\(^{17}\)This pin is driven by bit 5 of the PCR register at address 0x01900024. See section 11.8 and table 11-22 in reference [5]

\(^{18}\)In principle, the DSP does not have to poll these flags, and any external writes will simply stall the processor until it has the bus. We choose to make the test at least during the development phase.
program deasserts the bus request signal, and proceeds. Once the bus request is released, the arbitration logic reasserts the hold-off to the DSP. If multiple requests are pending, they are handling sequentially. The sequence described here is illustrated in Fig. 7. Note that if a DSP fails to remove the bus request, all other DSP’s are locked off the bus. It is crucial that no program fails when the request is asserted or the system will hang until a reset is issued.

3.3 Control Logic – Partial description

The actual track fits are performed within the DSP’s, but the data input and output, the SMT hit coordinate conversion, processor scheduling, synchronization, most error handling and most of the monitoring is handled by control logic implemented in programmable devices. Figure 8 shows the various functional blocks of the logic needed during normal data-taking. Additional blocks, not shown, are used to provide initialization and testing.

3.4 Initialization

Initialization of each TFC has three phases: (1) setting configuration registers defining the operating conditions, (2) downloading large look up tables and (3) downloading the DSP program and initialized data and starting the DSP execution. Figure 9 shows a flowchart of the initialization procedure. The description assumes familiarity with the TFC memory map described above.

3.4.1 Overall Initialization

The first step, the operating mode register is set to zero placing the system in the initialization state. Run-time constants used in data taking are then written to the configuration register. The digits marked as “M” and “N” in Fig. 9 for the configuration register vary
Figure 8: TFC control logic block diagram. The boxes indicate logic blocks, and the ovals indicate memory external to the programmable logic. Control signals are indicated using solid lines, and external data paths are shown using dashed lines. The boxes drawn using lines with short dashes indicate the grouping into physical chips.
Figure 9: Initialization flowchart. Numerical values containing “M” or “N” depend on which TFC is being initialized. Operations marked by “*” cannot be performed by direct PCI transfers. Instead, data must be written to the IDPM and then transferred from the IDPM to the actual destination. See the description of the IMXFER register in section 3.2.1 for details of this two-step transfer.
from TFC to TFC. The least significant two hex digits, marked as “MM” define the level 2
cable identifiers[6]. The digit marked as “N” is either 0xE or 0xF; the latter is used if CTT
information is to be transmitted to L2 global. The third step in the initialization enables
the input controller to find the FRC and (six) STC link receiver base addresses. The values
of the BAR’s in the LRB interfaces are written into consecutive registers in the TFC PCI-A
interface, BAR4. The fourth step loads the 2Mb coordinate conversion look up table into
the CCLUT memory. The tables are defined in 30° slices, so a different table is required
for each TFC. The next step(s) places the 4Mb look up tables containing coefficients used
in the track parameter linear algebra into the MLUT memories. Because each TFC has
two “columns” of four DSP’s, the matrix LUT data must be loaded twice, once for each
DSP column. Furthermore, the MLUT memory is not directly accessible from PCI, so the
indirect loading mechanism described in the IMXFER register section is used. As with the
CCLUT, the matrices are defined for 30° slices, so different tables are needed for each TFC.
Finally, the DSP code itself is loaded into each DSP in turn, and each DSP is started. To
load a DSP, the appropriate bits are loaded into the mode register, and then the program
data transferred to the selected DSP via the IMXFER method. The parameter “M” in the
“select DSP” block of Fig. 9 takes on the eight values 0x40, 0x50, 0x60, ..., 0xF0, one value
corresponding to a specific DSP. 10 A DSP is started by setting the least significant bit of
the interrupt word in the indirect transfer described in the IMXFER register section.

Initialization is required only when the crates are powered up or if data in the look up
tables changes. A global reset of the TFC simply returns the system to a known state,
but does not require complete reinitialization. After the reset, only the “start DSP” step is
required (for each DSP).

3.4.2 Programmable Logic Initialization

Each of the three programmable logic devices on the TFC (See Fig. 8) will be automatically
bootstrapped at power up by reading a dedicated serial EEPROM located on the TFC.
There is one EEPROM for each of the three programmable logic devices. If the code in the
programmable logic needs modification, the corresponding EEPROM will be (re)loaded and
the system reset. The EEPROM can be loaded either by a dedicated byte-blast compatible
connector on the TFC or by JTAG via the corresponding PCI interface. For testing, the
programmable logic can be directly loaded from JTAG, bypassing the EEPROM. In this case,
a system reset will restore the original EEPROM program to the programmable device.

3.5 Monitoring

There are effectively two types of monitoring required by the TFC. They are:

1. Collect Status (snapshot diagnostics)

2. Unbiased and/or forced write (large data size for offline analyses)

These are indicated as part of the SCL word. The FRC will echo these as necessary in
the extra header in the L1CTT road transmission. Table 15 shows the information to be
included in the COLLECT STATUS data. There are 16 32-bit words per TFC card. The
information includes event counters providing consistency checks and processor state and

10 See the mode register definition in table 3.2.1 to understand the eight possible values.
<table>
<thead>
<tr>
<th>Word #</th>
<th>Bit Pos’n</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic status. See below</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Processor/Bus status. See below</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Buffer status. See below</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 words/TFC, cumulative counts, in I/O controllers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0-15</td>
<td>Events in</td>
</tr>
<tr>
<td></td>
<td>16-31</td>
<td>Events out</td>
</tr>
<tr>
<td>2</td>
<td>0-15</td>
<td>Good events in</td>
</tr>
<tr>
<td></td>
<td>16-31</td>
<td>Corrupt events in</td>
</tr>
<tr>
<td>3</td>
<td>0-15</td>
<td>CTT roads in count</td>
</tr>
<tr>
<td></td>
<td>16-31</td>
<td>CTT roads out count</td>
</tr>
<tr>
<td>4</td>
<td>0-15</td>
<td>Populated road count</td>
</tr>
<tr>
<td></td>
<td>16-31</td>
<td>Underpopulated road count</td>
</tr>
<tr>
<td>5</td>
<td>0-15</td>
<td>STT out, fit complete</td>
</tr>
<tr>
<td></td>
<td>16-31</td>
<td>STT out, fit failed</td>
</tr>
<tr>
<td>2 words DSP processing times (µs, 8 bits/DSP times 8 DSP’s</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 15: Monitoring information for COLLECT_STATUS.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Field Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-5</td>
<td>total hit count</td>
</tr>
<tr>
<td>6-17</td>
<td>3 bits x 4 hit count/layer</td>
</tr>
<tr>
<td>18-21</td>
<td>1 bit x 4 hit count overflow/layer</td>
</tr>
<tr>
<td>22</td>
<td>1 bit iteration count flag (one loop or two)</td>
</tr>
<tr>
<td>23-24</td>
<td>2 bits dropped layer id</td>
</tr>
</tbody>
</table>

Table 16: Additional fit information for unbiased/forced write events.

timing snapshots to diagnose system problems and watch for load induced problems. If space permits, some fields may be expanded into histograms.

The data for the Unbiased/forced write categories includes the standard L1CTT and L2STT fit output information plus the following:

- input from STC: 1 word/hit (via FIFO)
- original road data to DSP: 3 words/road + 1 word/hit (via DSP)
- Packed information: 1 word/road, internal DSP information
- Matrix LUT address(es): 2 words/road possible

The packed information included with each fit is shown in table 16.

**Logic Status register.** Status regarding event flow, configuration and logic states can be determined by reading the logic status register. This register is defined in table 17.

**Processor/Bus Status register.** State information for the DSP processors and bus is latched using a low frequency clock running asynchronously with the beam clock and trigger.
<table>
<thead>
<tr>
<th>Bits</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>In data taking mode</td>
</tr>
<tr>
<td>1</td>
<td>RESERVED</td>
</tr>
<tr>
<td>2-3*</td>
<td>Event loader state: 0=idle, 1=FRC reading, 2=STC read, 3=finalize</td>
</tr>
<tr>
<td>4-7</td>
<td>Current EL event ID</td>
</tr>
<tr>
<td>8-9*</td>
<td>DSP select-and-load state: 0=idle, 1=loading, 2=bus wait</td>
</tr>
<tr>
<td>10-13*</td>
<td>Current DSL event ID</td>
</tr>
<tr>
<td>14-15*</td>
<td>DSP select-and-load state: 0=idle, 1=loading, 2=bus wait</td>
</tr>
<tr>
<td>16-19*</td>
<td>Current DSL event ID</td>
</tr>
<tr>
<td>20-21*</td>
<td>Event writer state: 0=idle, 1=LTCTT copy, 2=STT copy</td>
</tr>
<tr>
<td>22-25</td>
<td>Current EW event ID</td>
</tr>
<tr>
<td>26-27*</td>
<td>FitTracker State</td>
</tr>
<tr>
<td>28-31*</td>
<td>FitTracker current event ID</td>
</tr>
</tbody>
</table>

Table 17: Logic Status register, definition. *These require extra connections to the input and output controllers.

The processor status register is defined in table 18. Monitoring of the input and output buffers will be provided on the receiver cards. All monitoring information will be accessible from VME.

**Buffer Status** The *buffer status* register contains information about the occupancy of the input and output DPM’s and also for the internal transfer FIFO’s. The register is defined in table 19.

### 3.6 Error Handling – Place holder

NEEDS WORK. Pull SCL_INIT if too many consecutive events with errors on input or loss of event number sync.

### 3.7 Power Consumption

A TFC will contain eight DSP’s, three large pin-count programmable logic devices and three significant external memories. The power consumption, based on conservative use of manufacturers data sheets, is estimated to be 30 W per daughterboard.

The TFC components require three voltages: (1) 1.5V for the DSP core, (2) 2.5V for the Altera FPGA cores and 3.3V for I/O pins on both DSP and FPGA and for the memories. The PCI interface supplies 3.3V and 5V. The 3.3V signals will be used directly, and the 2.5V will also be derived from the 3.3V using regulators. The 5V supply will be used with a DC-to-DC converter to generate the 1.5V for the DSP cores. By separating the 1.5V supply from the remaining voltages, the power load is approximately balanced, and possible effects of noise from the converters are minimized. The design requires 17W at 3.3V and 15W at 5V corresponding to currents of 5.3A and 3.0A respectively.

---

Footnotes:

20 Currently, the devices are Altera 10K100E484-1 parts
21 If necessary, 1.5V signal can be generated from the 3.3V PCI power, removing the need for 5V.
### Table 18: Processor/Bus status register, definition.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-23</td>
<td>8 DSP’s x 3 bits ea. Processor state</td>
</tr>
<tr>
<td></td>
<td>State: 0 = Idle</td>
</tr>
<tr>
<td></td>
<td>1 = Loading</td>
</tr>
<tr>
<td></td>
<td>2 = Processing</td>
</tr>
<tr>
<td></td>
<td>3 = Matrix LUT Read</td>
</tr>
<tr>
<td></td>
<td>4 = Writing to ODPM</td>
</tr>
<tr>
<td></td>
<td>5-7 = reserved</td>
</tr>
<tr>
<td>24</td>
<td>Column #1, Memory bus, SET = contention wait</td>
</tr>
<tr>
<td>25</td>
<td>Column #2, as bit 24</td>
</tr>
<tr>
<td>26</td>
<td>Column #1, Expansion bus, SET = contention wait</td>
</tr>
<tr>
<td>27</td>
<td>Column #2, as bit 26</td>
</tr>
<tr>
<td></td>
<td>SET = initializing</td>
</tr>
<tr>
<td>28-31</td>
<td>reserved</td>
</tr>
</tbody>
</table>

### Table 19: Buffer status register, definition. All unused bits are defined to read as zeroes.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-3</td>
<td>Number of events in IDPM</td>
</tr>
<tr>
<td>8-11</td>
<td>Number of events in DSP’s</td>
</tr>
<tr>
<td>16-19</td>
<td>Number of events wholly in ODPM</td>
</tr>
<tr>
<td>24-27</td>
<td>Number of events wholly in L3 output fifo</td>
</tr>
<tr>
<td>29</td>
<td>L2 transfer FIFO, full flag</td>
</tr>
<tr>
<td>30</td>
<td>L3 transfer FIFO, full flag</td>
</tr>
</tbody>
</table>

26
3.8 Cost Estimate

The dominant portions of the cost are the processors, the programmable devices and memories. The cost for these components is roughly evenly divided and the total is $4000. The board manufacturing and assembly is estimated to add another $1000. Thus, the cost for each daughterboard is roughly $5000. The original estimate was $7k/slot, but the costs here do not include the motherboard or the I/O transceivers. All parts for the prototype boards have been ordered, and for some of the smaller or long-lead time components, all needed parts have been ordered.

4 Summary

The algorithms and and hardware design for the silicon track trigger track-fitting have been described. The aggregate processing time is between 10 and 20 $\mu$s for a large events using a DSP-based solution. The conceptual design is complete, and initial power, space and cost requirements appear acceptable.

References

[1] DØ Note 3516, *A Silicon Track Trigger Processor for DØ*


[6] Private communication, Jim Linnemann. The L2 data types for the STT(TFC) are 162-173, one for each TFC in the system. The highest six values will be used by the TFC’s which retransmit the L1CTT information.
History

- Version 2: Used in Feb. 25, 2000 internal review
- Version 3: More detailed beam-spot correction section
- Version 4: Revised L2STT output format
- Version 5:
  - Add memory map and register definitions and format definitions
  - Add control logic section and figures
  - Add initialization section (and place holder for error handling)
  - Add L2 output header and trailer formats
  - Add information to power section
  - Revise processor status register
  - Revise LUT address and data formats to reflect switch back to 6 independent barrels
  - Remove alignment tolerance section (Separate document)
  - Many small changes to text
- Version 6:
  - Revise PCI memory map
  - Revise DSP memory map
  - Revise **Operating Mode** register definition
  - Add **IMXFER** register definition
  - Add internal transfer FIFO format
- Version 7:
  - Tweaks to memory map after implementing logic. Fix DSP memory map
  - modify bits def’ns in IMXFER register
  - Minor text changes
- Version 8:
  - Add L3-events-in-FIFO count to buffer status register and change fifo-full bit positions accordingly
  - Remove sector ID from configuration register
  - Reorder monitoring registers to put status registers at the start of the address range.
  - Double the size of coordinate conversion LUT to 2M DWords.
• Version 9:
  – Revise operating mode register definition
  – Add packed data word description to IDPM memory map section.
  – Change BAR order in memory map to reflect Altera code. (Rearranged to minimize address space occupied by PCI interfaces.)
  – Add external memory access handshaking section
  – Expand initialization section