Readout Chip for an L1 Tracking Trigger Using Asynchronous Logic

J Hoff,[a] M. Johnson,[a,*] R. Lipton,[a] and G. Magazzu[b]

[a] *Fermi National Accelerator Laboratory[†], Batavia, Il., USA 60510*
[b] *University of California, Santa Barbara*
E-mail: mjohnson@fnal.gov

ABSTRACT: Adding a silicon based tracker to the level 1 trigger systems for LHC detectors can substantially increase the ability of these systems to find events with patterns of high Pt tracks. This is especially true for high luminosity running where there may be several hundred interactions per crossing. Cooling and mass constraints require that the readout chips have low power and generate little electrical noise. The LHC crossing clock and experiment trigger latency requires that a trigger be able to be made in less than 100 LHC crossings of 25 ns each. One way to minimize power and noise is to use asynchronous logic. We present a readout chip design for both level 1 trigger and event readout that is entirely asynchronous. The only clock used is the LHC crossing clock.

KEYWORDS: Triggers; mousetrap; pipeline; asynchronous.

[*] Corresponding author.
[†] Operated by Fermi Research Alliance, LLC under Contract No. De-AC02-07CH11359 with the United States Department of Energy.

Contents

1. L1 Trigger design

There are at least two methods of forming tracking triggers for hadron colliders. One method is to take all the hits from the detector and search for patterns that fit high-energy tracks. This is the method used by CDF in the SVT level 2 trigger. The pattern matching is typically done using a large content addressable memory that can simultaneously search all the data for specified patterns. This can work well for systems that have enough time to send all the data to a common location and that are small enough so that all the patterns can be contained in a few hundred chips.

Another approach is to find the track incrementally [1]. This is similar to most offline reconstruction systems where one finds a seed track and then extrapolates the seed to the next tracking layer and looks for hits that may belong to the seed track. This is typically a very time consuming process but if the detector is designed so that seed tracks are created very easily, such a system can overcome many of the time limitations of the first approach. This is the design approach described in this paper.

A crucial feature of this approach is to eliminate low energy tracks as early in the process as possible. This reduces the amount of data that must be sent to the next level of processing. This is done by using 3-D chip technology together with the 4 Tesla magnetic field of CMS. The readout chip is located in the region between two sensors and takes data simultaneously from both sensors (figure 1). The 4 Tesla field is sufficient to bend all the tracks with $Pt < 2.5$ GeV/c by several strips so these tracks can be immediately eliminated. Monte Carlo simulations indicate that the chance of having more than 4 track candidates (called stubs) with $Pt > 2.5$ GeV/c in a 100 mm by 100 mm sensor is less than 0.01%. This is a substantial reduction in data rate.

Stubs provide a useful first level filter but they do not provide the measurement accuracy needed for a trigger. Several stubs must be combined together to get sufficient accuracy. We must also keep the search window small enough so that the search can be done in a few tens of ns. We accomplish this by deploying a row of stacks on each side of a 40 mm support structure. The 40 mm separation was chosen so that the transvers displacement of a high Pt track across the gap is small enough to allow a narrow search window but large enough so that the combination of two stubs (called a Tracklet) is accurate enough to project the track to another tracking layer with an accuracy of a few millimetres. The length in the Z direction of the sensor strips is of order 1 mm so that both the Phi and the Z coordinates are constrained in the projection. This then limits the search area in the destination layer.

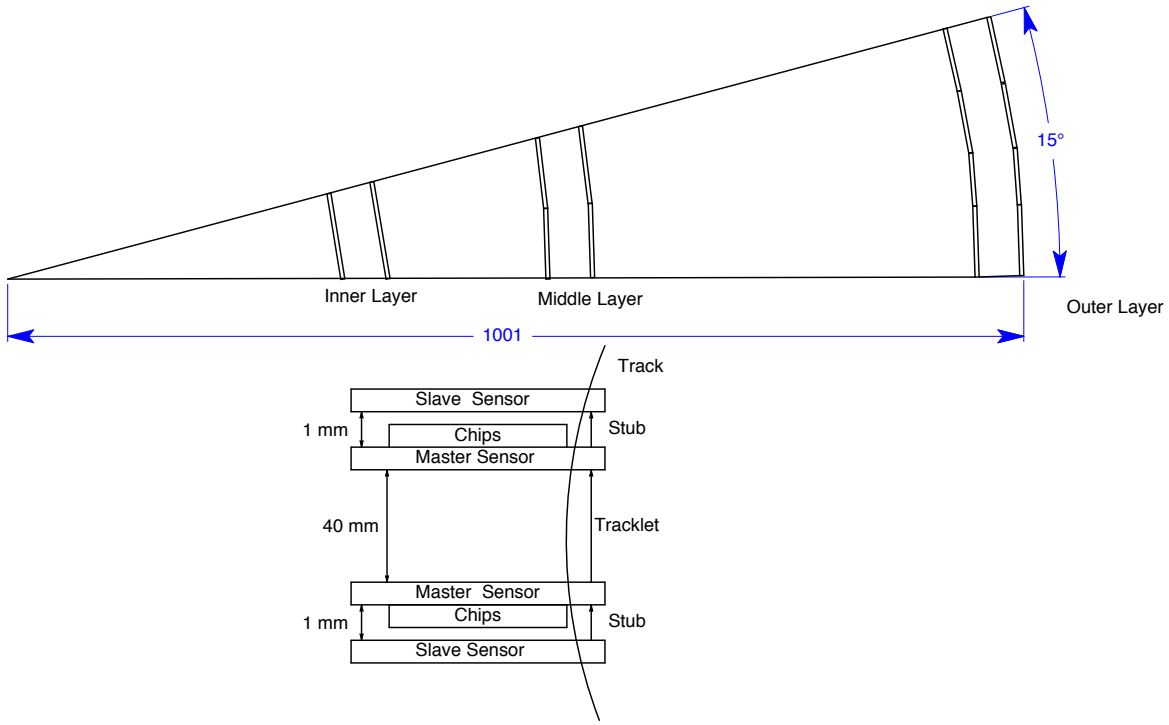

Figure 1: 15-degree sector of the tracker showing 3 super layers. The lower part of the figure shows a detailed view of one of the 3 super layers. There are 2 stacks of 2 sensors each. The readout chip is located between the two sensors of a stack. A stub is the portion of a track found in a stack of sensors. A Tracklet is formed from 2 stubs in the same super layer.

Track finding then proceeds as follows. After a bunch crossing, the readout chips simultaneously gather data from both sensor layers in their stack. Each chip then finds all stubs with Pt >= 2.5 GeV/c and sends these stubs off detector to the Tracklet processor that receives data from 6 stack pairs (24 sensors) and matches stubs to form Tracklets. It then projects these Tracklets to an adjacent layer based on the position and on the Pt of the Tracklet. This is done via look-up tables. Once the destination location in the other layer is known, the Tracklet data is routed to the Tracklet

finding processor in the adjacent detector layer. This processor then compares the incoming Tracklet with the ones that it has found. If there is a match, a track candidate is sent to the global level 1 trigger. Table 1 lists the pipeline steps for the readout chip. It is likely that some of these steps will be combined once detailed timing measurements have been made. Each step takes 25 ns so the total delay is 275 ns. The total L1 delay in CMS is 6.4 μs so this takes only about 4% of the available time.

Table I Pipeline steps for Readout Chip

25 ns Step	Operation
1	Discriminate input data
2	Load data for transmission between chips
3	Transmit data between chips in phi direction
4	Form Clusters
5	Send cluster data in Z direction
6	Encode cluster data in each of the short z sections
7	Encode 4 hits from all short z sections (4 hits/chip)
8	Load output pipeline
9	Transmit data in Phi direction to edge chip
10	Transmit data in Z direction to optical transceiver
11	Send data over the optical transceiver

The pipeline steps for the off-detector processing is still under development but a preliminary design requires 16 pipeline steps. Adding a 50% contingency and including 500 ns for cable delay to the off-detector processing electronics gives a total delay of 1.4 μs. This is about 22% of the time available.

2. Readout Chip Design

The readout chip is located between two sensor layers so a low noise design is required. The data rates are large so a design that minimizes power consumption is also important. The stubs required in the track finding process described in Section 1 are generated in the readout chip as the result of the following sequence of operations.

1. Detect hits in long strips and short strips belonging to top and bottom sensors respectively
2. Propagate hit information on edge strips to adjacent chips in the Phi direction (cluster finding in edge strips requires hit data from adjacent chips)
3. Identify clusters made by hits from up to three adjacent strips (wider clusters are ignored)
4. Propagate cluster information in the Z direction to preserve the capability to identify stubs associated with shallow angle tracks
5. Identify stubs and evaluate the Pt of the associated particle by correlating clusters in short strips (bottom sensor) and long strips (top sensor)
6. Propagate stub information to interfaces to optical links and then to the off-detector track finding processor

All these operations must be performed in a limited number of clock cycles to minimize the impact on the overall latency of the level 1 trigger. We propose the use of asynchronous pipelines,

an innovative approach in the readout of the front-end circuits for High Energy Physics experiments. The designers of integrated circuits often fear asynchronous architectures for data transfer because they can lead to unstable operation due to glitches and bus skew. However, they provide clear advantages in speed, low power and low noise: if no data must be transmitted, the circuitry remains inactive with a significant power saving and there is no need of high speed serial links that could be a potential source of noise.

2.1 Asynchronous readout

2.1.1 MOUSETRAP pipelines

We adopted the MOUSETRAP micro-pipelines developed at Columbia University [2] that provide features that fit our requirements. In our readout chips MOUSETRAP pipelines will be used to transfer hit and cluster data between adjacent chips and to send the stub data to the off-detector processor. The final readout architecture of our modules will be based on an array of 5x5 chips. Figure 2 shows a simplified array of 2x2 chips: in each chip Hit Pipelines (HP) and Cluster Pipelines (CP) will be used to propagate hit data in the Phi direction and cluster data in the Z direction to the adjacent chips. Chips in the same row will be daisy chained and Phi Pipelines (Phi-P) will handle the propagation of the stub data in the Phi-direction. Circuits belonging to the last column will be daisy chained and Z Pipelines (Z-P) will handle the propagation of the stub data in the Z direction to a final data concentrator circuit that will act as a buffer and as an interface to the optical transceiver.

The data flow control is implemented by one EXCLUSIVE NOR (XNOR) gate. It uses a non return to zero protocol so the bandwidth required for the control lines is minimized and when it is combined with an Earle gate[3], the propagation delay though each stage is quite uniform. Fig. 3 shows a 2-input Earle gate.

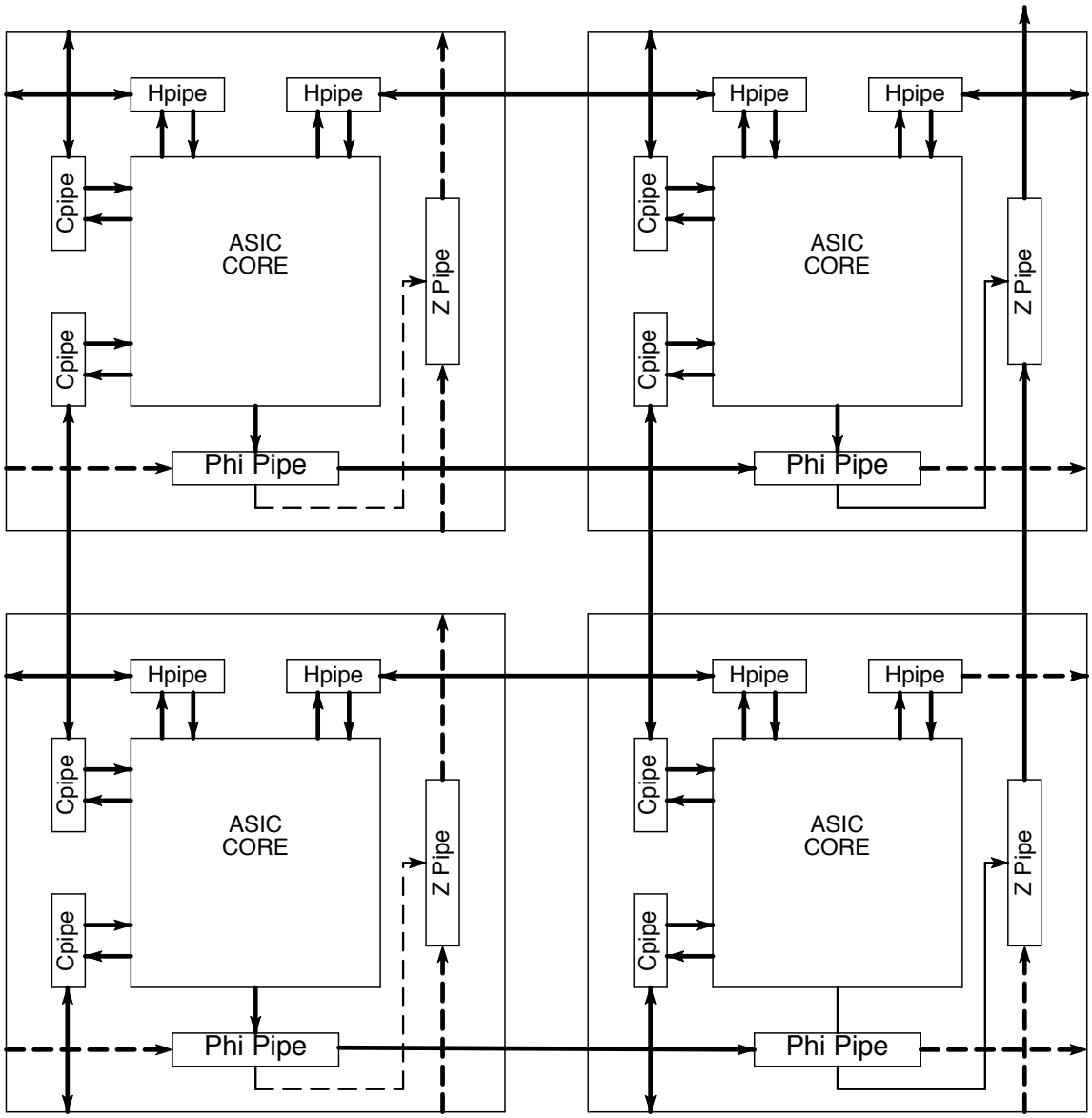

Figure 2: Simplified array of 2x2 readout chips. Cpipe is a pipeline of clusters and Hpipe is a pipeline of hits. Phi Pipe is the pipeline of stubs in the phi direction and Z Pipe is the pipeline of stubs in the z direction. Dotted arrows indicate data paths that are on the chip but not used in this configuration.

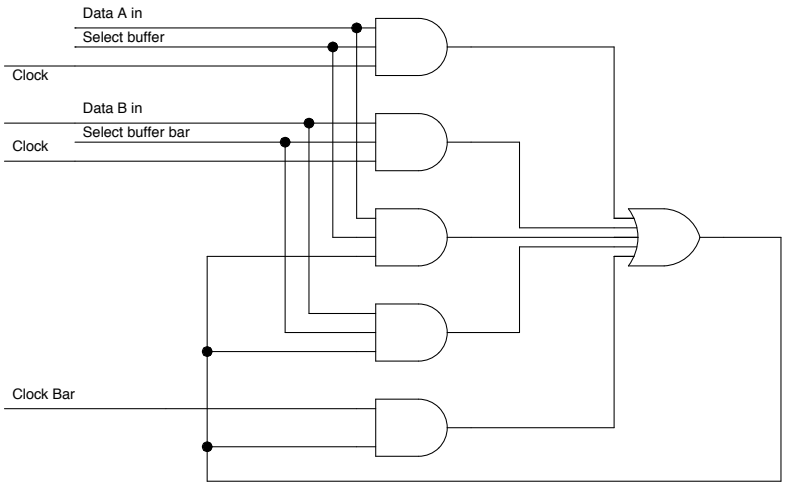

Figure 3: 2-input Earle gate.

Figure 4 shows a block diagram of a 3-stage MOUSETRAP micro-pipeline. Its operation can be understood as follows. First, assume that all the control lines (DONE and ACKNOWLEDGE) are at logic 0. The XNOR gate then holds both latches at logic 1 which is the transparent (non latched) mode. To send data in the pipeline, the requestor loads latch 0 with data and sets the control bit to 1. The logic 1 propagates through the transparent latch 1 and forces the XNOR to 0 thus latching the data at latch 1. The data continues to propagate to latch 2 that also latches the data. The DONE signal from the latch 2 is routed back to latch 1 as the ACKNOWLEDGE signal. It tells latch 1 that latch 2 has acquired the data so that latch 1 can go back to transparent mode. This is done by the XNOR gate that now has both its inputs at logic 1 so its output is at logic 0 and latch 1 is now transparent again. New data can be sent by changing the control of latch 0 to logic 0. This will cause the DONE signal in latch 1 to go to logic 0 and thus latch the new data at latch 1. Nothing will change at latch 2 unless its acknowledge line goes to logic 1.

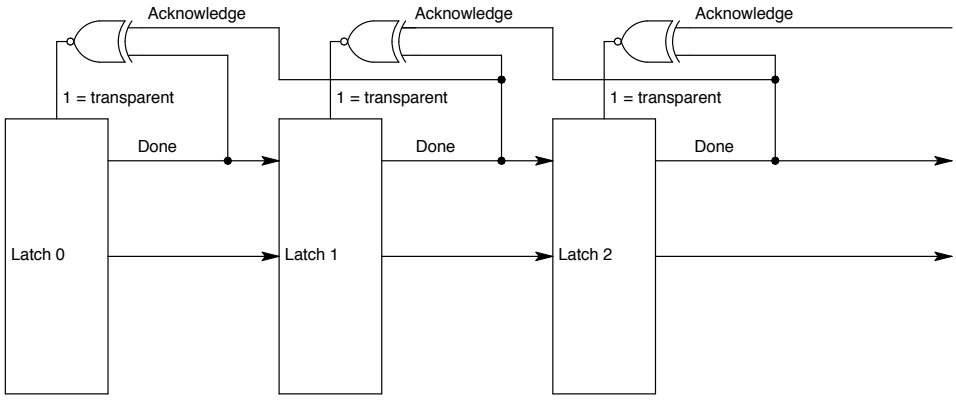

Figure 4: Simplified logic diagram for a 3-element MOUSETRAP pipeline. Data to be sent is loaded into latch 0 from external sources.

The basic micro-pipelines needed two modifications for use in our readout chip. The first modification is to allow direct loading of the pipeline latches. The pipeline is used to move data between chips so that during the first part of a transfer cycle, data is loaded into the latches on the sending chip and the latches are cleared in the receiving chip. When the data load is complete the input of the latches is switched back to the pipeline and the asynchronous pipeline sends the data to the second chip.

The second modification is to make the pipeline variable length. In the proposed readout architecture we daisy chain 5 chips in the Phi and in the Z readout pipelines and each chip must be able to accommodate up to 4 hits. Nevertheless the average number of hits in the pipeline will be much less than 4 so most of the registers will be empty most of the time. A variable length pipeline that transmits only valid data can clearly save the power and the time required to transmit null data in a fixed length pipeline.

Both modifications are achieved by using a multi-input Earle gate in the latch (figure 4). The two-input Earle gate is used for fixed length data transfer of hit data between chips. One input is from the preceding latch of the MOUSETRAP pipeline and the other input is the load data bus. Changing the select line to select the pipeline register as the input starts the pipeline. The three-input Earle gate is used for the variable length pipeline. This pipeline needs the same data loading capability as the hit transfer pipeline. But it also needs input from both an upstream pipeline register and an upstream chip. For example, with only one hit in the chip, only one register would be entered into the pipeline. Its output would go to the downstream chip and its input would come from the upstream chip. If there were two hits in the chip, the input would come from the upstream pipeline register rather than the upstream chip.

Figure 5 shows an example of a 4-stage pipeline with 2 registers in the source chip and two in the destination chip. Data is loaded into the first 2 registers during one part of the transfer cycle and then transferred in the second part. If the control registers are loaded with a logic 1 in the first register and a logic 0 in the second and the ACKNOWLEDGE line at the end of the pipeline in the second chip is held at the logic 1, the data will transfer to the destination chip when the load signal goes low.

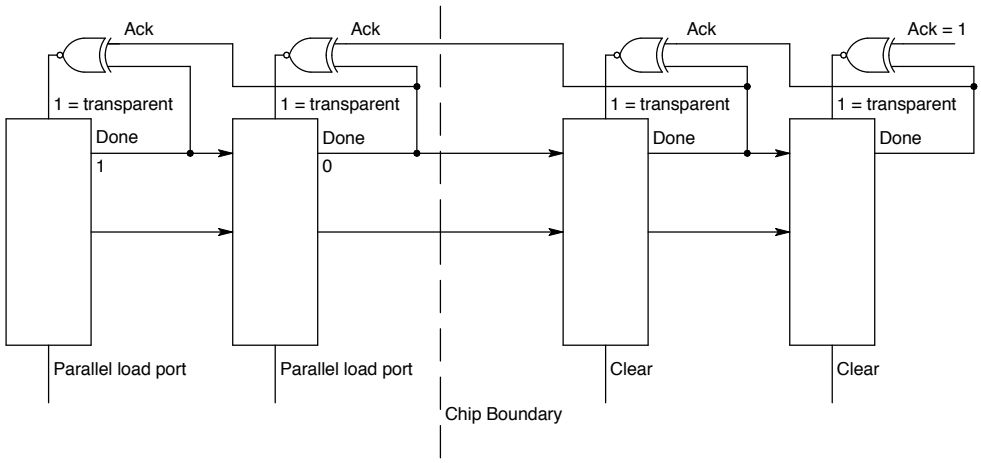

Figure 5: 4-stage MOUSETRAP pipeline split between 2 adjacent chips. This is the type of pipeline used to transfer edge hits between chips in the phi direction.

2.1.2 Dual Rail Protocol

Possible problems in MOUSETRAP pipelines can occur because of the skew between the control and the data lines due to the different propagation delays of logic 0's and 1's and because of glitches on the control lines, due to noise or to Single Event Transient (SET) effects induced by radiation. Glitches can cause momentary latch transitions that can corrupt the information stored in the pipeline. One way to eliminate these effects is to use a "Dual Rail" encoding protocol: two lines are used to encode each bit, logic 0 is encoded as "01", logic 1 as "10" and states "00" and "11" are illegal. This approach eliminates any differences in propagation speed between logic 0's and 1's and also eliminates glitches since a momentary transition to "11" or "00" is ignored. Of course it costs both power and chip area so our current design uses dual rail only on the control lines. Errors in the data are corrected by using Hamming encoding with error detection and correction capability for the data.

2.1.3 Mephisto Priority Encoder

To evaluate the hit address we use Mephisto encoders [4] which are priority encoders that include prioritizing in both directions (low to high and high to low) and a feedback mechanism that sends a signal back to the input channel that has been selected. We use this signal both to gate the associated Pt onto a readout bus and to suppress the selected bit in a second implementation of the encoder. Two daisy chained Mephisto encoders can handle up to four hits in each readout circuit.

2.2 Readout of event data

The maximum event rate is 100 KHz so the average readout time is 10 μs or 400 crossings. This large number of crossings allows us to use the trigger data path for event readout. We append one event cluster from one chip at every crossing. Thus, we can read out 100 event clusters in 100 crossings. It is possible that there will be too many trigger stubs in a crossing so that the event data is not transmitted. We solve this by having an "event data received line" that is used to tell the sender that the data was successfully received. A token is passed from one chip to the next to determine which chip can send the data. A 3-bit crossing number is included with each data word so that we can check that the correct event is being sent.

2.3 Protection against chip failures

The design is robust enough with 3 layers so that a few per cent of random chip failures are acceptable. What we want to avoid is having a single failure disable several other chips. We do this by making every chip have a reversible data path. If the chip circled in figure 6 fails, we simply enable the spare optical transmitter, reverse the direction of the chips in blue and readout the data on the spare line. This method also protects against an optical transmitter failure.

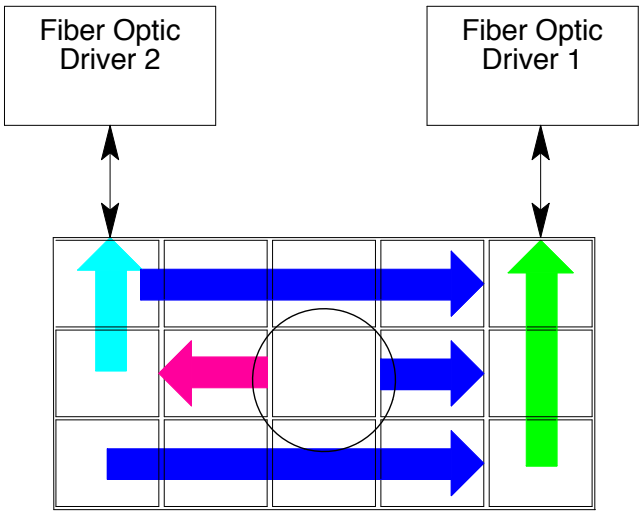

Figure 6: protection against chip failures.

3. Summary

Developing a level 1 trigger for the LHC upgrade is a challenging task. There is a large amount of data and very little time to find the tracks. Asynchronous design has the potential to deliver very high-speed circuits at modest power levels and reduced noise.

References

[1] E. Hazen et al., *Architecture of a level 1 track trigger for the CMS experiment*, JINST 5 (2010) CO8004

[2] A. Pietropaolo et al., *MOUSETRAP: Ultra High Speed Transition Signalling Asynchronous Pipeline*, Proc. International Conf. Computer Design (ICCD), Pages 9-17, Nov. 2001

[3] S. R. Kunkel and J.E. Smith, *Optimal Pipelining in Supercomputers,* Proc. 13th International Symposium on Computer Architecture, ACM SIGARCH **14** 2 (1986) 404

[4] P. Fischer et al.,NIM. A431(1999) 134.