



**Fermi National Accelerator Laboratory**

**FERMILAB-Conf-91/298**

## **A Verilog Simulation of the CDF DAQ System**

**K. Schurecht and R. Harris**

*Fermi National Accelerator Laboratory  
P.O. Box 500, Batavia, Illinois 60510*

**P. Sinervo and R. Grindley**

*University of Toronto, Dept. of Physics  
60 St. George, Toronto, ON, Canada, M5S 1A7*

**November 1991**

\* Presented at the *IEEE Nuclear Science Symposium*, Santa Fe, New Mexico, November 2-9, 1991.



## **Disclaimer**

*This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.*

# A VERILOG SIMULATION OF THE CDF DAQ SYSTEM

K. Schurecht and R. Harris, Fermi National Accelerator Laboratory<sup>1</sup>  
P. O. Box 500, Batavia, IL 60510

P. K. Sinervo and R. Grindley, University of Toronto  
Dept of Physics, 60 St. George, Toronto, ON, Canada M5S 1A7

## Abstract

A behavioral simulation of the CDF data acquisition system was written in the Verilog modeling language in order to investigate the effects of various improvements to the existing system. This system is modeled as five separate components that communicate with each other via Fastbus interrupt messages. One component of the system, the CDF event builder, is modeled in substantially greater detail due to its complex structure. This simulation has been verified by comparing its performance with that of the existing DAQ system. Possible improvements to the existing system were studied using the simulation, and the optimal upgrade path for the system was chosen on the basis of these studies. The overall throughput of the modified system is estimated to be double that of the existing setup. Details of this modeling effort will be discussed, including a comparison of the modeled and actual performance of the existing system.

## I. INTRODUCTION

The CDF data acquisition system at Fermilab [1] is a complex series of components that record digital data from the CDF detector over a large Fastbus network. The throughput of the system is being increased by improvements to various components in the system. The goal is to improve the event readout rate into the level 3 trigger system from 7 Hz to 30 Hz with 90% livetime. Since some components are complex and do not conform to standard queuing models, the DAQ system has been simulated using the Verilog modeling language. The main goals of the simulation were to decide how to configure the event builder, how to distribute the scanners for maximum parallelism, and to determine where present and potential bottlenecks exist.

## II. DAQ SYSTEM OVERVIEW

The goal of the CDF DAQ system is to achieve an event readout rate in excess of 30Hz with 90% livetime for the entire detector. To achieve this goal, no single component should

contribute more than 5% downtime. A simplified layout of the DAQ system is shown in Figure 1.

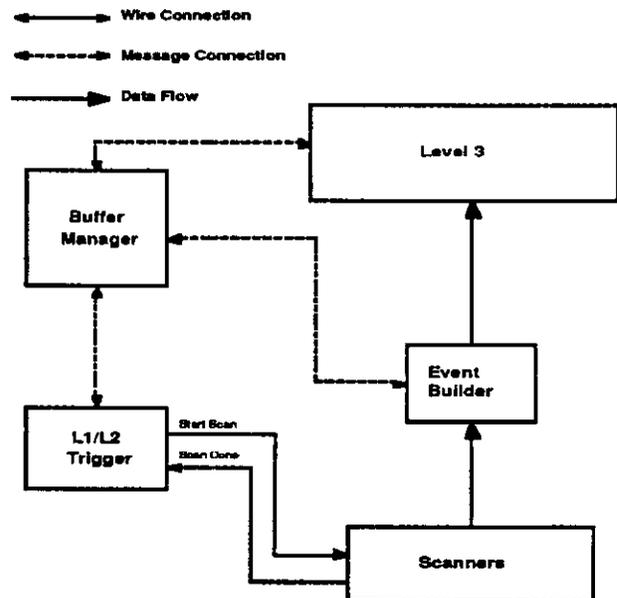


Figure 1: Block Diagram of CDF DAQ System

The L1/L2 trigger is a set of specialized Fastbus boards. They examine an event that is stored on the detector and decide if it should be read out by the front end scanners. The front end scanners are Fastbus-based processors that digitize and read out the data stored on the detector. They format this data into one of four buffers that are then read out by the event builder. The event builder [2,3] is a set of Fastbus boards that reads ("pulls") the data out of the front end scanners, reformats the data into a single YBOS record [4,5] and writes ("pushes") the data into the Level 3 farm. Any two of the three operations, pulling, reformatting, or pushing, can be performed in parallel, which is achieved by having two data buffers (or "engines") in the event builder, each of which can store a complete event.

The Level 3 trigger is a set of six Silicon Graphics multiprocessor computers, with each housing eight processors (for a total of 48 processors). Level 3 has the final decision to reject the event or store the event on tape. The buffer manager [6] is a process that runs on a VAX workstation. It sends Fastbus interrupt messages to and from the L1/L2 trigger, the event builder and the Level 3 system. The buffer manager coordinates the flow of data through the DAQ system and acts

<sup>1</sup> Work supported by the U.S. Department of Energy under contract No. DE-AC02-76CHO3000. Work supported in part by the Texas National Accelerator Research Laboratory and the Natural Sciences and Engineering Research Council of Canada.

as the central control process in the DAQ system. All interactions between the L1/L2 trigger, event builder, the Level 3 system and the buffer manager use Fastbus interrupt messages.

A typical event would flow through the system in the following way. A interaction occurs in the detector and the analog data is temporarily stored. The L1/L2 trigger analyze a subset of this data to see if the interaction should be accepted. If the trigger decides to discard the event, the front end flushes the data and another event is read in. If a trigger is accepted, the front end scanners read out the event from the front end into one of four buffers. When the event builder has a free engine (i.e. buffer), the buffer manager sends it a message to read out the scanner buffer. The event builder reads out the buffer and reformats the data. It then sends a message to the buffer manager and the buffer manager responds with the address of a buffer in level 3 to push the event record to. The event builder then pushes the event to level 3 where it is further processed and either rejected or saved to tape.

### III. EVENT BUILDER ARCHITECTURE

Of the five components in the simulation, the event builder is the most complex. The event builder is composed of three types of Fastbus boards: crate controllers, cable controllers and reformatters. The crate controller acts as the event builder supervisor. It handles all interaction with the buffer manager and works with the reformatter board to write data to Level 3 from the reformatter buffers. During a push operation, the crate controller passes sequentially through each of the reformatter boards, pushing the data from one of the event builder buffers via the Fastbus crate segment to the level 3 system. A complete event builder system only requires one crate controller.

Each cable controller acts with one or more reformatters to pull data from the front end scanners. Every cable controller is attached to its own Fastbus cable segment and is responsible for reading out the scanners on that cable segment. The cable controllers pull data in parallel into one of the two buffers on a reformatter attached to the same cable segment.

The reformatter determines how the data in its buffer should be reorganized into YBOS format. The data itself is reorganized during the push operation into the Level 3 system. Since there are two buffers on the reformatter, any two of the above actions may be done at the same time. For example, the event builder could be reading data into one buffer while reformatting in another. Alternately it could be reading in one buffer and pushing out the second buffer.

Figure 2 shows the CDF event builder setup using two event builders in parallel. In this setup, events are alternated between the two event builders. The event builder is expandable to have as many as fifteen boards. This would allow for 7 cable segments of scanners.

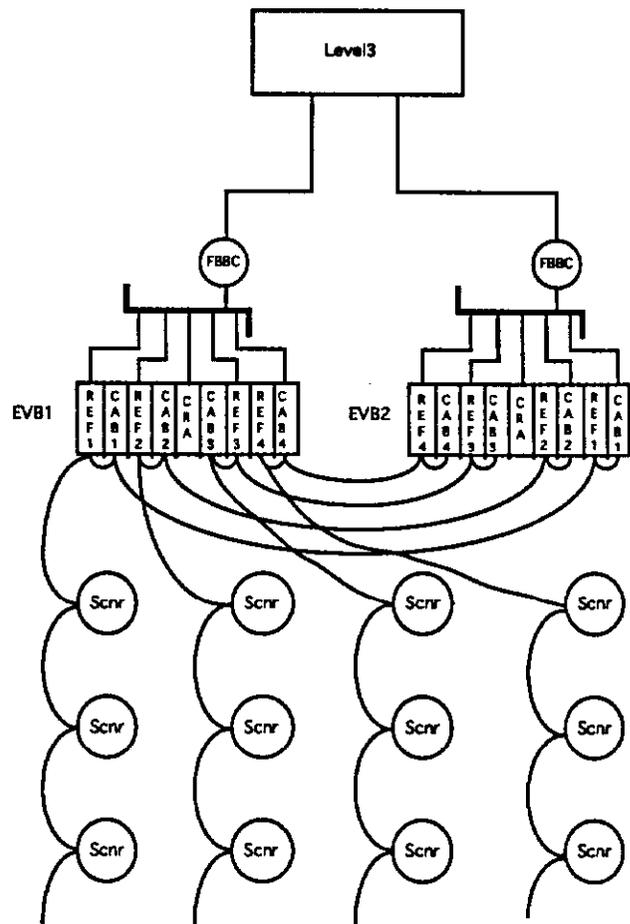


Figure 2 CDF Dual Event Builder System

### IV. DEADTIME ANALYSIS

There are three components that contribute to the deadtime in this system: the L1/L2 trigger system, the front end scanners, and the DAQ system. The trigger system incurs deadtime by inhibiting event taking while a determination is being made on whether or not to accept an event stored on the detector. This deadtime is directly proportional to how long it takes to make the Level 1 and Level 2 trigger decisions (~ 10 microseconds). The deadtime due to the scanners is proportional to the length of time it takes to digitize and read out the detector data into a scanner buffer, since the detector cannot be reenabled until this operation has been completed. Since all scanners must have completed before the detector can be reenabled, the scanner with the longest readout time determines the scanner deadtime (~ 2 milliseconds). Even though the trigger decision time and scanner readout time are very short, they are significant when the rate of events being considered by the L1/L2 trigger is of order 5 kHz and the rate of events being read out by the DAQ system exceeds 30 Hz.

The DAQ deadtime comes from not being able to read out the scanner buffers as fast as they are filled up. If there is no buffer available in the scanners, the L1/L2 trigger is inhibited and any interactions during that time are not recorded. The

event builder, buffer manager and Level 3 system are all potential sources of DAQ deadtime and the slowest device in this chain determines its value. For example, if the buffer manager is unable to keep up with the incoming messages, the event builder will sit idle until the buffer manager finds time to give it another task.

The DAQ deadtime is the focus of the simulation studies. Whereas the L1/L2 trigger and scanner deadtime can be readily determined by formulae, the unique architecture and interaction of the buffer manager, event builder and Level 3 system cannot be analyzed by simple queuing theory and a stochastic simulation is necessary to better understand their contribution to deadtime.

## V. GOALS OF THE SIMULATION STUDY

The original CDF DAQ system achieved a maximum readout rate of approximately 7 Hz during the 1988-89 Tevatron Collider run. For the 1992 Collider run, an improvement in this readout rate to 30 Hz is required. Several possible upgrade paths for the DAQ system are possible: (i) increasing the number of cable segments used for the readout, (ii) expanding from a system with one reformatter per cable segment to two reformatters, and (iii) expanding from one event builder in the DAQ system to two event builders. Factors like the cost of the modifications to the buffer manager software needed for two event builders and the cost of extra reformatters or controller boards were important considerations in the upgrade choice. The primary goal of the simulation studies were to estimate the performance improvement of each upgrade choice without the cost of implementing each configuration.

Measurements taken when reading out small portions of the detector were found to be misleading when naively extrapolated to the full detector. This results from the capability of the event builder to execute two tasks in parallel. We found that the length of time required for the pull and push operations and how much reformatting is required can radically affect how efficient the event builder is in overlapping operations. Therefore, the other use of the simulation was to fine-tune the behavior of the system under typical operating conditions.

There were four configurations that were simulated. They were:

- One event builder, one reformatter per cable segment
- One event builder, two reformatters per cable segment
- Two event builders, one reformatter per cable segment
- Two event builders, two reformatters per cable segment

Each of these configurations were tested and the results are shown in section 7.

## VI. SOFTWARE DESIGN OF THE SIMULATION

The simulation was written as five separate components that interacted with each other via messages. The L1/L2 trigger

first creates an interaction in the detector that in principle would pass the L1/L2 trigger. The time difference between subsequent interactions is a random variable with a Poisson distribution. The L1/L2 trigger also acts as the monitor for deadtime. When an interaction occurs three conditions are checked. If the L1/L2 trigger is still processing an event the deadtime is charged to the trigger system. If the scanners are still reading out a previous trigger, the interaction is considered lost and is considered scanner deadtime. If there are no buffers available in the scanners, it is due to a bottleneck in the DAQ system. When none of these inhibiting conditions are present, the L1/L2 trigger module sends a start scan signal to the scanners.

The model for the scanners is a process that upon receiving a start scan, waits a fixed amount of time representing the scanner digitization and formatting, and then returns a scan done signal to the L1/L2 trigger. The L1/L2 trigger then sends a Fastbus message to the buffer manager informing the buffer manager that an event is ready for readout.

The buffer manager receives and sends Fastbus interrupt messages to coordinate the system. When it receives a message, it waits a fixed amount of time (representing the overhead involved in the receipt of the message), updates its queues and responds by starting the next action.

Due to the event builder's ability to overlap operations and its interaction with the buffer manager, the simulation incorporated sufficient detail in the event builder to accurately model how this overlapping takes place. One such key feature was how the messages sent between the event builder boards interacted. Depending on how they are coordinated, there could either be a large or small amount of overlap in operations.

Since the Level 3 system was not considered to be a factor in these studies, it only acted as a receiving point for data sent from the event builder. The Level 3 system is considered to have enough processing power to handle the rates that the event builder can produce.

## VII. RESULTS OF SIMULATION

The performance of the simulation and the real DAQ system was first compared in order to verify how well the simulation was able to reproduce the properties of the system. We found that the simulation reproduced the observed deadtime to within 1% over a wide range of data rates and event sizes. This gave us confidence that the model could be used to predict the behavior of other system configurations.

The results of the simulations are shown in Figure 3, where the livetime shown is only the DAQ system livetime. The important point is where the DAQ system livetime crosses 95%. This is the trigger rate at which the DAQ system will induce 5% deadtime into the system. From the graph, it is obvious that as hardware is added to the event builder, the livetime increases.

In these results the relative time that each of the three types of event builder component boards and the buffer manager are busy becomes important. The crate controllers' busy time is based on how long it takes to push data plus how long it takes

to handle the communication overhead. For the amount of data expected in the 1992 run, a crate controller is busy an average of 31ms per event. The cable controller takes about 11ms to handle pulling data out of the scanners on its cable segment. The average time a reformatter is busy during one event is 27ms. This is the time that the reformatter is actively working and does not include any pull or push time during which the reformatter could be actively reformatting. These times are based on a performance study of the event builder [7] on how to optimize cable segments. The buffer manager busy time is derived from how long it takes to handle one event worth of messages. For a single event it takes the buffer manager about 25ms to handle all of the messages. The theoretical maximum rate of a configuration comes at the point one of the boards is 100% busy. However, due to interaction of messages and the need to make certain functions wait until others are finished, this is an unreachable goal.

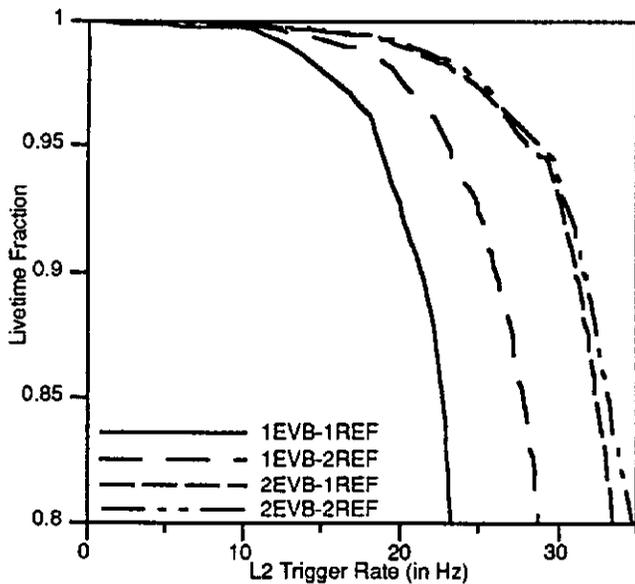


Figure 3: DAQ System Livetime

The results show that one event builder with one reformatter per cable segment is the slowest configuration, with a 95% livetime rate of 19Hz. The flat out rate of this configuration is 25Hz (the flat out rate is the fastest rate the configuration can go with no concern for downtime). This is the base configuration with no improvements. The other three cases are measured against this to see how much is to be gained by each of the improvements. In this configuration the crate controller is busy about 60% of the time, the cable controller about 21%, the reformatter about 50%, and the buffer manager 50%. Figure 4 shows how the boards interact during a typical slice of time for a single event builder system. The figure is not drawn to scale, but does show where boards need to wait for other actions to finish and why. All of the boards have some time that is wasted. The wasted time is from waiting for some other process to complete.

The second configuration is one event builder with two

reformatters per cable segment. The effective change is that there can be twice as much reformatting going on at one time, so the reformatting time is effectively cut in half. Nothing is done to decrease the effective pull and push times though. Therefore the reformatter busy time decreases to about 30%, while the crate busy time increases to 70% the cable time to 25%, and the buffer manager time to 60%. The 95% livetime rate for this setup is 24Hz. The flat out rate is 29Hz. Figure 5 shows the cpu usage of this configuration.

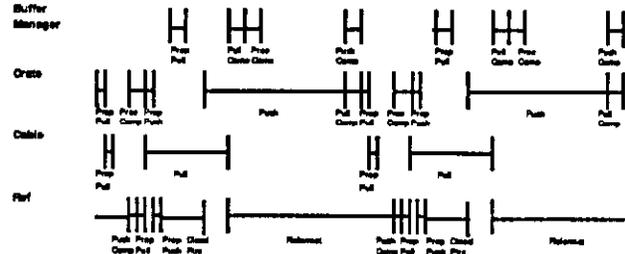


Figure 4: Cpu Usage for 1 EVB / 1 REF

When a second event builder is added to the system, all of the event builder busy times are effectively cut in half since alternating events go to different event builders. If the buffer manager could completely absorb this added load, the rate into level 3 should double. Unfortunately at this point the event builder stops being the bottleneck in the system. The increase from one to two event builders only increases the rate from 19Hz to 30Hz. At 30Hz, the buffer manager starts getting overloaded with messages and can no longer keep up. Another 5 hz can be squeezed out of the system (35Hz total), but only at the expense of severe downtime. At 35Hz the buffer manager is completely saturated and can no longer run any faster.

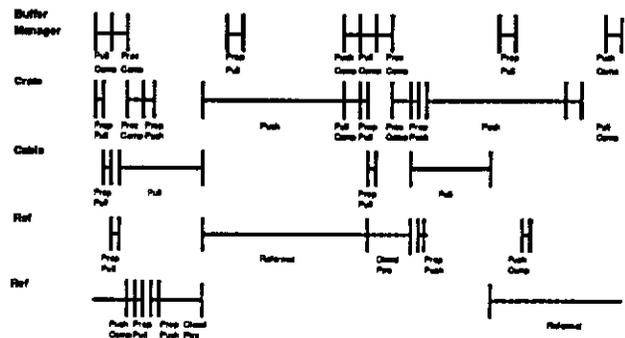


Figure 5: Cpu Usage for 1 EVB / 2 REF

When more reformatter boards are added to the two event builders, the gain is minimal. The event builder would be able to run slightly faster as shown in the changes in the first two configurations, but due to the buffer manager the rate is still 30Hz.

Since the buffer manager was unable to handle any faster rate than what was available by two event builders with one reformatter per cable segment, it was decided to build that

configuration for the 1992 run. The added cost in hardware and maintenance was well worth the cost of going to two event builders since there was such an improvement from both one event builder configurations. It was not worthwhile to go to the fourth configuration since the gain was so small.

By doing a detailed study of the event builder, certain internal improvements were identified. For example, when preparing to push the data, a set of pointers must be downloaded. This is a time consuming task and depending on exactly how much data is being reformatted can be best done either at reformat time or at push time. Making this an option makes it possible to make the best choice once exact data sizes are known. Also, much of the waiting time on the event builder boards can be diminished by allowing front panel messages to interrupt long processes, such as the push. Previously all messages had to wait until the push was complete to be handled. This can cause long waiting periods where another board is waiting for the crate board to handle the message. A set of priority messages have been determined that are important enough to interrupt ongoing processes to speed up the complete system.

### VIII. CONCLUSIONS

The behavioral simulation of the DAQ system has proven to be very helpful in determining ways to improve the system. It was able to predict the performance improvement for a number of different system configurations. Extensive hardware tests of these configurations with the CDF DAQ network were not necessary, freeing up valuable time on the Fastbus network that could be used profitably by others. Bottlenecks in the system were found that people can now work on trying to improve. An important result of the simulation study is that the buffer manager response time will now be the limiting factor in the DAQ system performance. Should the present DAQ system need to be further improved, this would be the first bottleneck to attack.

The simulations also provide a means of evaluating how the processing load is distributed within the event builder itself as the overall configuration of the system changes. Since the cable controller is busy so little of the time, its contribution to deadtime is negligible. When two reformatters are placed on each cable segment, the reformatter busy time is reduced to a level where its response did not contribute to overall deadtime. This leaves only the crate controller and the buffer manager with high busy times. When a second event builder is added, the buffer manager remains the only process with a high busy fraction. The simulations show that this fraction can increase to about 80% before the system is saturated.

The insights provided by the simulation will enable the system designers to make informed decisions about how to further improve the performance of the DAQ system. The simulation will therefore continue to play a significant role in future system studies.

### IX. REFERENCES

- [1] E. Barsotti et. al., "FASTBUS Data Acquisition for CDF", Nucl. Instr. and Meth. A269(1988)82.
- [2] A. W. Booth, M. Bowden and H. Gonzalez, "Specification for a Hardware Event Builder", CDF Note 452, Fermilab.
- [3] P. K. Sinervo, et. al., "Fast Data Acquisition with the CDF Event Builder", IEEE Transactions on Nuclear Science, Vol 36, No. 1, February 1989.
- [4] J. T. Carroll, "YBOS Scanner Bank Format", CDF Note 264, Fermilab.
- [5] D. Quarrie, et. al., "CDF Event Structure", CDF Note 152, Fermilab.
- [6] C. Day, "Buffer Manager Software Design", CDF Note 326, Fermilab.
- [7] R. Harris, et. al., "Estimation of Event Builder Execution Time for the 1991 Run", CDF Note 1242, Fermilab.