Lawrence Berkeley Laboratory
UNIVERSITY OF CALIFORNIA

Engineering & Technical Services Division

A CAMAC INTERFACE FOR TPC DATA-ACQUISITION ELECTRONICS

S. Sidman, S. Olson, and R. Jared

June 1983

MASTER

S. Sidman*, S. Olson, and R. Jared

Lawrence Berkeley Laboratory
University of California
Berkeley, California   94720 U.S.A.

INTRODUCTION

The Time Projection Chamber (TPC) is a detector used for high-energy physics research at the Stanford PEP Accelerator. TPC requires about 17,000 channels of data acquisition, which samples on command the input to each channel at a 10 MHz rate. This high data rate is made possible by means of Charge Coupled Devices (CCDs),[1] intelligent digitizers,[2] and a sophisticated trigger system.

Analog data from the shaping amplifiers[3,4] are sampled and stored in the CCDs every 100 ns. The CCDs are analog shift registers, which always contain the most recent 455 data samples. When the trigger system determines that a significant event has occurred, the CCD clock period is changed from 100 ns to 50 μs, and the digitizers begin digitizing each sample as it is outputed from the CCD. The digitizers are of Wilkinson Ramp type and perform an analog to digital conversion in 40 μs. Each digiti∠ed value of each channel is compared with a threshold value stored in RAM for that channel. This value is written to an output RAM for readout if the digitized value exceeds threshold. Readout is done during the 40 μs digitization time, so that while sample n is being digitized, sample n-1 is being read out.

The TPC-CAMAC interface described here was developed to allow experiments of smaller scale than the complete TPC to use the standard data acquisition portion of the TPC electronics, namely the amplifier, CCD and digitizer bins. These three bins, when properly interconnected and controlled by the interface control bin, form a transient digitizer with a depth of 455 samples and a maximum width of 256 channels per bin set.

SYSTEM OVERVIEW

Before going into the description of the interface itself, it is first necessary to describe in greater detail the TPC data acquisition system which it controls. As mentioned above, the data acquisition electronics consist of three types of bins: an amplifier bin, a CCD bin, and a digitizer bin. All bins have 17 slots. Sixteen of the slots in each bin house cards dedicated to the bin function, each card having sixteen channels of that function. The seventeenth slot in each bin houses the control card for the bin register control unit (RCU) interface, which is necessary for control functions and test pulser gating. The RCU bus in the full size TPC is an 8 bit bi-directional data highway which is used to set up the various data acquisition system operating parameters. Figure 1 is a block diagram of the Experimental Time Projection Chamber ETPC system. The three bins containing the shaping amplifiers, CCD's and digitizers are shown across the top of the figure. Each

bin supports 256 channels. The fourth bin (interface) and the CAMAC module that buffers the dataway are shown across the bottom.

The amplifier bin contains up to 16 cards of shaper amplifiers which provide signal conditioning for the rest of the data acquisition system. The programmable controls on the seventeenth card in the bin consist of RCU registers that control discriminator levels and gating for the test pulses.

The CCD bin contains up to 16 CCD cards and a 17th card. The CCDs are analog shift registers which hold 455 samples of input voltage. CCDs are put in the signal path of the data acquisition system for time expansion so that slow A/Ds can be used. The CCD devices are normally clocked at 10 MHz (one voltage sample every 100ns) while the system is in a reset state awaiting a trigger and during the data acquisition state. The clock frequency is reduced to 20 KHz (50 μs per sample) during the readout state. The programmable controls that effect this bin and their significance are:
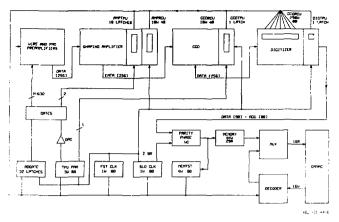
a) CCD Fast Clock Period – Sets the data sample rate.
b) CCD Slow Clock Period – Sets the readout rate for A/D input.
c) Fast Clock Counter – Acts as a pretrigger, setting up the number of fast clocks which will be allowed after a trigger is received. If 455 clocks are selected, then the first sample read out of the CCD when it is switched to the lower readout frequency will be the trigger sample.
d) Slow Clock Counter – Sets up the number of slow readout clocks which will be allowed; that is, the number of CCD samples to be read. Readout terminates either on slow clock terminal count or buffer memory full.
e) Pedestal Control – An RCU function on each CCD card which adjusts the CCD zero level on a board wide (16 channel) basis.

The digitizer bin contains of up to 16 cards of Wilkinson ramp run down (single slope) A/Ds with 16 channels per card, and a 17th card. Each digitizer card has logic which compares each digitized channel with a value set in a lower limit RAM. Only those channels which exceed their limit are read out to digitizer buffer stored in RAM memory, thus providing a time-efficient "sparse data readout". The typical application is to set the lower limit RAM value on a per channel basis just above the expected system noise level. These lower limit RAM values are loaded via the RCU interface.

The readout to buffer memory involves the signals Readout/Reorder and Data Present. During the readout state (40 μs) all channels are digitized and all validated channels are read out from digitizer buffer RAMs to the buffer memory. While sample n is being digitized, sample n-1 is being read out. During the reorder state (10 μs) all channels are compared with

1

their respective lower limits, and all channels which exceed those limits are written to digitizer buffer RAM.

Data Present is a board-by-board status signal which indicates during readout state the presence of valid data in a digitizer buffer RAM. Buffer RAM read out is done at a 2.5 MHz rate, so that if all 256 channels had valid data (an unlikely occurrence), 102.4 us would be required to read them all out. In fact, even if only half the channels had valid data the nominal 40us readout time would still be exceeded. This problem is resolved by freezing the CCD clock which advances the CCDs to the next sample until the data present signal is globally false. The interface contains the logic for this interlocking protocol involving Readout/Reorder, Data Present and CCD Clock.

THE INTERFACE

The prototype interface was packaged as a single large CAMAC module of three boards. It is presently in use at the MUSIC box experiment at the HISS facility of the Berkeley BEVALAC.[5]

The present interface consists of a CAMAC interface module and a TPC style interface bin comprising seven boards. Two cards are used to decode CAMAC commands into specific data acquisition system set-up conditions or actions, and to recreate and distribute the RCU bus.

Three cards are concerned with self-test. Self-test in the interface exists at three levels: static, memory, and dynamic. Static self-test is the capability to read back any register in the interface or in the data acquisition bins and verify it. Memory self-test is the ability to generate buffer memory test patterns and read them back. Dynamic self-test

is the ability of the interface to generate and inject test pulses into the TPC data acquisition bins and simulate a complete data acquisition cycle. By using the appropriate maintenance software the complete signal path from amplifiers all the way to interface buffer memory may be rigorously checked out.

One card is dedicated to sequence control. It supplies the CCD clock of appropriate frequency and number depending on system state, and also manages the digitizer readout protocol.

The final card is the buffer memory, which serves as the repository for validated digitized data before its transmission to slower mass storage devices. All but the first word of this memory are divided into three fields. They are:

1) An address field which identifies the digitizer board and channel number.
2) A Z-bucket field which identifies the CCD bucket number.
3) A data field which holds the nine bits of digitized data.

All three fields have a parity bit for error detection. The first word (word 0) is actually the last word written, and contains two fields. They are the word count, which gives the amount of buffer memory used, and the phase, which is a digitized value produced by a built-in time to digital converter. The phase parameter is the time between the event trigger and the first system clock. Typically this value is used to more accurately locate the event within the chamber.

All words are written into the memory broadside, i.e., all 29 bits at a time, over the interface internal tri-state bus. All data are read back to the host computer via CAMAC a field at a time. The method of readout is as follows:

2

A memory reset instruction is issued, which puts the memory in the read mode and resets the buffer memory address counter to zero. The two fields of word zero are individually addressable. The word count is read back with one f-code, the phase with another. The read f-code is then sent n times, where n is three times the word count after one memory address increment f-code. The first two occurrences of this command select out and read back the first two fields of the current word. The next occurrence of the f-code advances the memory address counter and reads out the third field. Every third occurrence of the f-code will advance the counter.

The issuance of this memory read f-code is presently done via programmed I/O through an MBD branch driver, and so is somewhat slow. Modifying the MBD branch driver software to decrement the word count value and issue the read command would permit faster operation. A modification to the interface made by workers at CERN copies the word count to a separate hardware register, which is then decremented at every read. When the word count is exhausted, the CAMAC Q signal is put up and readout is halted.

SOFTWARE

Any system is just an expensive collection of electronic parts until the appropriate software is used to put it to work. The development of checkout and maintenance software for a small system which has many of the complexities of the larger system from which it is derived can pose significant problems. The question is one of balance. The interface was designed to take advantage of the economic leverage of using an existing data acqusition system whose development cost had already been incurred. It would not make sense to pursue this course of system development if a great expenditure of time and money were needed to write the maintenance software. Some specific constraints which we faced were:

1) The software had to support the step-wise verification of the system that experience with the full size TPC had shown to be most efficient. In particular, the verification sequence is:

   a) Establish that the RCU dataway words for single register read/write.
   b) Prove that all RCU registers exist by using memory address tests.
   c) Prove that all RCU registers function under random access
   d) Verify the integrity of RCU registers by cycles of various bit patterns.
   e) Establish that the system can digitize and read out a signal.
   f) Verify specific system functions, such as readout of the phase information.
   g) Verify that the system operates over the entire physics data path by running a simulated experiment with test pulses.
2) The software had to provide for checkout of defects revealed by running any of the above tests.
3) The software had to provide for high repetition rates for various system functions so that scope loops would be possible.
4) The software had to be simple and convenient to use.
5) Inexperienced personnel would be using the software later on to maintain the system.
6) Funding was limited.

There are two basic approaches to take in the development of such software. One approach is to

assemble a team of programmers, maintenance personnel, and experimenters. The two user groups would present their concerns to the programming group, who, in turn, would write programs which deal with these concerns. Two basically incompatible kinds of work flow would then be established. Programmers would be caught in a loop of writing specific programs to cover specific test conditions as system checkout proceeded at the same time they were trying to write a comprehensive data acquisition operating system. There is an inherent conflict between short and long term goals. This form of software development also requires that all groups be relatively knowledgeable, and that the software group in particular become expert in all phases of system operation.

The other approach to software development avoids setting up incompatible work flow and competing goals by a two-fold strategy which seeks to make the best use of expensive programmer time. First, a software tool is written which has the minimum number of commands needed to control the interface and system in a user friendly form. These commands consist of simple mnemonics which are directly related to the system block diagram in both name and effect. For example, load fast clock register, or set board address for RCU operation. These commands are then grouped into command files by the user and the command files executed in an interpretive way by the software tool. The users need only to know the host computer's resident editor to create the command files. The programmer's task is reduced to writing a simple parser and a series of short routines which convert the command lines into one or more CAMAC CNAF (Data) commands to the interface. In the second porton of the development strategy the programmers then write the specific programs needed to verify those few aspects of system operation not adequately covered by the command file interpretive software tool.

This second approach to software development was, in fact, the one taken, with very positive results. The six constraints previously mentioned were all satisfied. Specifically, since the software tool was relatively easy to code and allows the user groups to solve most problems by themselves, an expensive software development budget was avoided and the funding constraint was not violated.

Since user groups need only a knowledge of the system block diagram with mnemonics and their own requirements to use the software tool, even inexperienced users quickly gain proficiency in the use of the software, and in system operation.

High repetition rates for scope loops are made possible by the repeat count option in the software. For example, 32000 repetitions of loading the fast clock register can be done with the following commands:

LOC; FSTCLK; SET FAST CLOCK REGISTER LOCATION
DAT; OO376; OATA FOR REGISTER
EXC; 32000; EXECUTE 3200O TIMES

After the first interpretive execution, all subsequent executions are done at full computer speed.

Stepwise system verification is made possible by four main functions which deal with successively higher levels of the system. The CAMAC function will issue a CAMAC command the number of times specified. For example, the sequence:

```
FUN ;24
    N;2
    A;0
    EXC
```

will execute a F24N2A0 command one time.

The Format 2 function provides for read, write or verify of individual or selected banks of RCU register with the same data. Minimum and maximum board and channel numbers may be specified for execution. The Format 3 function does a form of RCU memory address testing (not to be confused with the 4K physics data buffer) to ensure the existence of all RCU registers. *The Format 4 function performs random address and data-field testing of RCU registers. Other commands perform readout and testing of the buffer memory.*

These functions may be combined to form sequences which check out most of the data path and which represent a complete data acquisition cycle. Figure 2 is an example of a command file containing one such sequence. *In this example,* the system is initialized, the test trigger executed, and the buffer memory compared to an expected value. This level of complexity represents the upper limit of usefulness for the software tool. More complex or comprehensive checks of system function are best done by software written specifically for that purpose. Note that even the special software is made easier to write by the availability of the software tool's internal routines.


```
EXAMPLE OF COMMAND FILE

TEST PULSE            ; First line (Comment)
FUN ;24               ; Disable LAM Mask F Code
    N   ; 2           ; Slot 2
    A   ; 1           ; A Code
    EXC               ; Execute F24A0
FUN : 9               ; Function Code
    EXC               ; Reset/Abort F9A0
FN: ;32               ; Format 2
    N   ; 2
    CAD ;00 00        ; Set Channel Limits
    BAD ;00
    LOC ;FSTCLK       ; Set Fast Clock Location
    DAT ;000376
    MOD ; 3           ; Write Verify
    EXC               ; Set Fast Clock Req. to 176 Octal
    LOC ;SLWCLK       ; Set Slow Clock Location
    DAT ;000376
    EXC               ;
FUN ;25
    A   ; 1
    EXC               ; Test Trigger Execution
LAM ;
    N   ; 2
    EXC ;032100       ; Test 0 for True, Up to
                        32100 times
FUN : 9               ; Clear 3
    A   ; 1
    EXC
PRI                   ; Print Mode On
FUN ; 1
    A   ; 2
    EXC               ; Read Word Count and Print
FUN . 1
    A   ; 1
    EXC               ; Read Phase and Print
NPRI                  ; No-Printing Mode
FUN ; 27
    A   ; 2
    EXC               ; Increment Address Counter
F2  ;
    N   ; 2           ; F2 Read and Verify
    TST ;4000,4100    ; Print Errors for Memory
    DAT ;000377       ; Location 4000 to 4100 if
    EXC               ; Read Data is not equal to 377
END                   ; Close the File
```

Fig. 2. Example of command interpretive language use by ETPC software.

The maintenance and checkout software was originally devised by one of us and H. Ikeda of KEK and R. Enomoto of the University of Tokyo[6]. It was later revised by R. Benetta of CERN and T.C. Meyer of the University of Wisconsin as their unit was being debugged. The software consists of a command file interpreter called ETPC and four other programs. All of the software is written in FORTRAN.

True experimental data gathering software is presently available only under RSX-11, although the ETPC package could be modified for this task with relatively small effort. Indeed, experimenters have found that using the ETPC package is a good way of gaining familiarity with the data acquisition system operation, and of testing system set up values before incorporating them into the actual run time software.

SUMMARY

An approach to general particle and nuclear physics data acquisition has been implemented using the hardware originally developed for a very large experiment and a special CAMAC interface. Success in achieving a desired cost and schedule objectives is obtained by using a proven data acquisition system whose development cost has already been incurred, a CAMAC interface whose development cost is relatively small, and a simple yet flexible software package which allows all types of users to develop their own routines.

This approach of writing a software tool which permits users to interact with the system and solve their own problems represents a very efficient use of everyone's time and the experimenter's money. Everyone need not be an expert on everything in order to do useful work with the system. By using the system block diagram with mnemonics and the software tool, the knowledge base required of each group is more in line with that group's intended function. Maintenance personnel can concern themselves with system hardware functionality, experimenters can concern themselves with optimizing the set up of system variables for specific experiments, and programmers can occupy themselves writing those few special programs which are needed for the long term.

ACKNOWLEDGMENTS

Reference to a company or product name does not imply approval or recommendation of the product by the University of California or the U. S. Department of Energy to the exclusion of others that may be suitable.

REFERENCES

1. R.C. Jared, T.Y. Fujita, H.G. Jackson, S.B. Sidman and F.S. Goulding, "Use of CCDs in the Time Projection Chamber," IEEE Trans. Nucl. Sci., NS-29, No. 1, Feb. 1982, pp. 282-285.

2. M.R. Maier, S.R. Olson, M. Nakamura and F.S. Goulding, "Multichannel Analog-to-Digital Converter and Readout System for the Time Projection Chamber," IEEE Trans. Nucl. Sci. NS-26, No. 1, Feb. 1979, pp 697-700.

3. R.C. Jared, D.A. Landis, and F.S. Goulding, "Analog Signal Processing for the Time Projection Chamber", IEEE Trans. on Nucl. Sci., NS-29, No. 1, pp. 57-61.

4. D.A. Landis, R.S. Adachi, N.W. Madden, and F.S. Goulding, "Low Noise Preamplifiers/Amplifiers for the Time Projection Chamber", IEEE Trans. on Nucl. Sci., NS-29, No. 1, pp. 573-577.

5. E.F. Barasch, F.P. Brady, C.M. Castaneda, J. Drummond, J.L. Romero, C.D. Goodart, and R. Rothrock, H. Crawford, R.C. Jared, D.E. Greiner, P.J. Lindstrom, S.B. Sidman and H. Sann, "A Prototype Multiple Sampling Ionization Chamber," presented at the 1981 Asilomar Meeting of the American Physical Society, Oct. 18-30, 1981.

6. H. Ikeda and R. Enomoto, and revised by R. Benetta and T.C. Meyer, "ETPC Test Program Operating Manual," internal document, Lawrence Berkeley Laboratory, 1982.