CONF-781159--4

SLAC-PUB-2220
October 1978
(I)

ATROPOS - A VERSATILE DATA ACQUISITION AND ANALYSIS SYSTEM†

C.A. Logg and R.L.A. Cottrell
Stanford Linear Accelerator Center
Stanford University, Stanford, California 94305

ABSTRACT

At SLAC we frequently need versatile, portable, rugged, and compact test and control modules to use in the development and testing of detection equipment for high energy physics experiments. The basic system we have developed is based on an LSI-11 microcomputer with 24K RAM, 4K ROM, 2 serial interfaces (one to the console terminal, the other to the large SLAC IBM computer complex (the 'TRIPLEX')), a programmable clock, and a CAMAC crate controller. Data logging support is provided for magnetic tape, floppy disk, and an interactive program ACQUIRE which runs on the TRIPLEX under the timesharing system ORVYL. The data is read from various CAMAC modules, collected, buffered, and optionally logged. At a lower priority, the data read is sampled a l analyzed in real time on the LSI-11 to produce various histograms and tables. Concurrently a more extensive analysis can be performed by the TRIPLEX program on the . ita which is logged to it. Interactive facilities provided b/ the microcomputer operating system enable the user to change CAMAC module addresses and the function codes used with them, specify various data cuts and transformations that are to be performed on the sample data, and specify new histogram limits and titles. Results of the real-time analysis, by both the microcomputer and the TRIPLEX program (if it is attached), may be displayed in graphical or tabular form on the console terminal.

The basic system hardware cost (exclusive of the magnetic tape drive and floppy disk drive) is around $7000. The software is written in a modular fashion so that the user can supply his own data reading and analysis routines. This system has been in use for two years by various groups on several LSI-11s at SLAC.

INTRODUCTION

ATROPOS is an LSI-11 microcomputer[1] based software system for data acquisition and analysis which is now in use at SLAC. The design aims have been to provide a system that:

* is CAMAC compatible
* is portable
* is simple to use
* is modular in software and hardware so it can be easily tailored to individual needs
* can make use of the various facilities[2] of the large SLAC IBM computer complex (the 'TRIPLEX')†† without modification to the TRIPLEX hardware or system software; in particular:
  + can be programmed by making use of the sophisticated software development facilities available on the TRIPLEX for source code preparation and storage, object code production and linking, and down line loading
  + can be used interactively with the TRIPLEX to provide a more sophisticated online analysis than the LSI-11 can provide on its own
* can be used effectively in stand alone mode (i.e., independent of the TRIPLEX and any logging facilities)
* is low in cost.

The basic pattern around which the system must work is that when an 'event' happens, the data resulting from that event must be read in, analyzed locally, and sometimes logged to some form of mass storage. A grouping of events in time (from time A to time B) is called a 'run'. When we 'BEGIN' a run, various initializations must occur. The user must be able to 'PAUSE' the run, possibly to examine the data already taken, to change or fix the hardware, or maybe to change slightly the analysis which is happening on an event by event basis. Then the user will want to 'RESUME' the run. At some point the user will want to 'END' the run, and henceforth the data that was taken will be referred to as the data from RUN N where N is some number. During a run, the user must be able to examine in graphical and

† Work supported by the Department of Energy under contract no. EY-76-C-03-0515.

†† The SLAC computer complex is composed of two IBM 370/168's which run OS/VS2 Release 1.6, and one IBM 360/91 which runs OS/MVT Release 21.8, all under the control of ASP version 3.2.

(Contributed to DECUS Fall Mini/Midi Symposium, San Francisco, CA, November 29-30, 1978.)

tabular form the results obtained so far. He may selectively restart some of the analysis (clear histograms) or modify the analysis (redefine or define some data transformations and/or histograms where the results are binned, or even reconfigure the CAMAC devices, i.e., change their addresses and/or function codes).

The following sections describe the system we have developed.

HARDWARE CONFIGURATION

The basic system (Fig. 1) consists of an LSI-11 with EIS/FIS,[1] 24K 16 bit words of RAM, 4K 16 bit words of Intel 2708 EPROMs, 2 serial interfaces (DLV11),[1] a programmable line time clock (KW11L),[3] a CAMAC crate, a dedicated type U CAMAC crate controller (Schlumberger JLSI-10), and an alpha-numeric video terminal (usually an Ann Arbor 4080D).

The 4K of EPROM contains a kernel of routines which support the link to the TRIPLEX, and provide some basic I/O and conversion facilities for use by the stand alone systems which are down line loaded into the RAM. A further description of the EPROM programs can be found in Reference 4.

One of the serial interfaces is for the connection to the TRIPLEX. Communication to the TRIPLEX is via an EIA RS232C asynchronous serial line, utilizing a telephone and modem or a hardwired line. The second serial interface is for the console terminal.

More elaborate ATROPOS systems (Fig. 2) have included:

* a Tektronix 4013 as the console terminal; it is used concurrently as the graphics display device
* a third serial interface that goes to an Ann Arbor terminal (without keyboard) which is used to provide a constantly refreshed tabular display of the event data, current run statistics and 24 bit CAMAC scaler readings
* an 800 bpi 9 track tape drive for local logging of data
* a floppy disk for backup reloading of the program and local logging of data. †††

SOFTWARE DEVELOPMENT AND MAINTENANCE

The programs are written and maintained by using the software development facilities available on the TRIPLEX. The facilities include a PL-11 cross compiler,[5] a MACRO-11 cross assembler,[6] a cross linker,[6] the WYLBUR text editing system,[7] IBM OS data sets, and all the peripherals of a large computer center. A further description of how we utilize these facilities for our software development can be found in Reference 4.

The programs are edited, compiled and/or assembled and linked together on the TRIPLEX to form an absolute load module which is then down line loaded into the LSI-11. This allows us to have complete access to our software development facilities from any site where we have access to a hardwired line to the TRIPLEX or a telephone and modem.

OPERATOR INTERACTION FACILITIES

Operator interaction is provided via the console keyboard and optionally via a specially designed experiment control panel (ECP) (see Fig. 2) which

††† The ROM kernel has a facility for dumping the RAM contents onto floppy disk and restoring it from floppy. It takes about 3 minutes to down line load ATROPOS (16K words) into the LSI-11 over a 9600 baud TRIPLEX line and about 90 seconds to reload from the floppy disk.

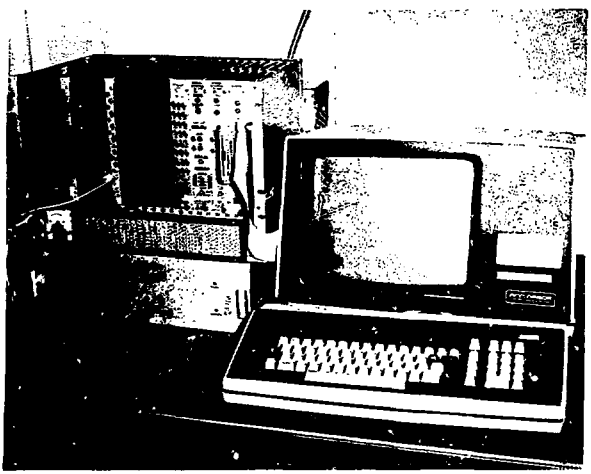

Fig. 1. The basic system is composed of an LSI-11, a CAMAC crate, and an Ann Arbor terminal.

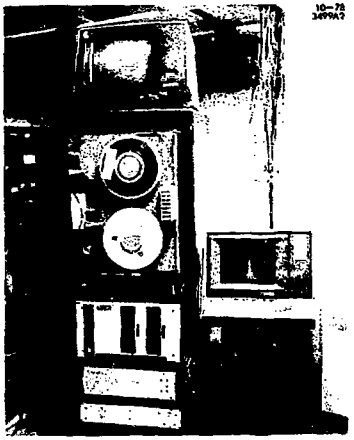

Fig. 2. One of the more elaborate ATROPOS systems consists of (from top to bottom): an Ann Arbor terminal (without keyboard) as an auxiliary display, a magnetic tape drive, an ECP, a dual floppy disk drive, the LSI-11 itself, and on the right, the Tektronix 4013 terminal which is used as the LSI-11 console and the graphics display device.

interfaces to the system via a single width CAMAC module.

The console keyboard is interrupt driven. The user enters, through the keyboard, a command text string and terminates it with a carriage return. The input text string is then matched against a set of command strings and the appropriate subroutine is called to execute that command. Some commands will prompt further for a text string, octal, decimal integer or floating point number accordingly.

The ECP has two 16 bits octal thumbwheels (referred to as ETW0 and ETW1), two 4 digit decimal thumbwheels (ETW2 and ETW3), and one hexadecimal thumbwheel (ETW4). It also has 16 sense lines (toggle switches), 16 push buttons, 4 digits of 7 segment LEDs, and 16 LED status lights. When one of the buttons is pushed, a CAMAC LAM interrupt occurs. The routine which responds to this CAMAC interrupt reads the push buttons and calls the appropriate routine to handle the push button command. Every 1/30th of a second, the ECP thumbwheels and toggle switches are read, and their values are saved in the appropriate CONSTANTs (ETW0, ETW1, ETW2, ETW3, ETW4, TOGL). The CONSTANTs are discussed in the section entitled User Accessible Variables. The 7 segment LEDs are used together with one of the octal thumbwheels to provide monitoring of memory locations. The status lights are used to indicate the status of the program and the data taking.

For an ATROPOS system which does not have an ECP, commands are added to the keyboard command list which enable the user to set the toggle switch bits in TOGL, thumbwheel CONSTANTs (ETW0, ETW1, ETW2, ETW3, ETW4), and issue, through the keyboard, commands that would have been provided by the ECP push buttons.

A list of the commands available follows. Those with a double asterisk are also provided by the ECP if one is available. The command HELP when entered via the keyboard will print a list of available commands together with a short description of each.

**	BEGIN	- Begin a run.
**	BKCLR	- Clear the background queue.
**	CONLIS	- List the CONSTANTs.
**	CONSET	- Set a CONSTANT.
*	DETACH	- Terminate the logging.
**	END	- End a run.
*	EXPDEF	- Define an expression.
*	EXPDEL	- Delete an expression definition.
*	EXPEXEC	- Execute the defined expressions. This is provided for system testing purposes.
*	EXPLIST	- List the expressions which are defined.
*	EXPSHOW	- List the values the expressions had after their last evaluation.
*	HALT	- Execute a HALT instruction and put the LSI-11 into ODT mode.
**	HCLR	- Clear a histogram.
*	HDEF	- Define a histogram.
*	HDEL	- Delete a histogram.
*	HELP	- Print a list of available commands.
**	HGET	- List the histogram definitions.
**	HLOG	- Log the contents of a specified histogram.
**	HOUT	- Display a histogram.
**	HSUM	- Display the statistics for a specified histogram.
*	JOBTIME	- List the time and date the load module was created.
*	ODT	- This command provides limited absolute address memory access and modification. This is provided for system testing purposes.
**	PAUSE	- Pause a run.
*	RECOVER	- Recover from a TRIPLEX crash.
*	RESTART	- Restart the logging.
**	RESUME	- Resume a run.
*	TOGL	- This allows the user to set a sense-line. This command is provided only on systems which do not have an ECP.
*	TRACEOFF	- Disable the trace trap. This is provided for system development.
*	TRACEON	- Set a trace trap. This is provided for system development.
*	WYLOFF	- Disable all TRIPLEX communication.
*	WYLON	- Reenable TRIPLEX communication.
*	CTRL U	- This is equivalent to pushing the UPDATE button on the ECP. It updates the display according to what is contained in ETW0.

The commands which are provided for system testing purposes are usually included in the final production system, since they may be useful in checking out the CAMAC hardware and other peripherals.

USER ACCESSIBLE VARIABLES

In an ATROPOS system there are several groups of variables, commonly referred to as the CONSTANTs, which the user has access to. Each group of CONSTANTs is identified by an ID in the range 100 thru 177 (octal). The CONSTANTs include:

* various run statistics such as the run number (RUN#), the number of events read so far (READ), the number of events logged (LOGD), the number of events not logged (LOST), and the number of events analyzed locally (SAMP)
* various parameters associated with the histogram display format (TOGL,ETW0,ETW1,ETW2,ETW3,ETW4)
* event data pedestals
* event data CAMAC addresses
* event data CAMAC read functions
* the event data
* event data histogram ID's

Commands are provided so that the user can easily look at, and, in some cases, change the values of the CONSTANTs. Each CONSTANT has associated with it an up to four character name, or an up to four character name and an index. A facility is provided whereby when the user tries to set a CONSTANT, a subroutine can be called to check the value it is to be set to. That subroutine then can either issue an error message to indicate a bad value (and possibly a reminder of the purpose of the CONSTANT) and set a flag (so the value will be reprompted for), or set the value to some reasonable value.

CONSTANT accessing commands are:

* CONSET - examine a CONSTANT and set its value
For example:*

```
. CONSET
CONST NAME= AA1   (AA1 is a CAMAC address variable.
                   164502 is its octal Q-BUS
                   address, and 1 10 1 is the
                   decoded crate, station, and
                   subaddress.)
(ABS.ADDR.,C,N,A)= 164502  1  10  1
CRATE=1
STATION=<cr>      (A null response does not change
                   the old value.)
SUBADDRESS=2
CONST NAME=AP1
0 : 75            (0 is current contents, replace
                   it with 75)
CONST NAME= <cr>  (The <cr> terminates the CONSET
                   dialogue.)
.
```

* CONLIS - lists a group of CONSTANTs in tabular form
For example:

```
. CONLIS
CONSTANT GROUP? ?  (A '?' gives a list of the
                    CONSTANT group titles.)
101-STATISTICS
102-ECP OPTIONS
103-ADC DATA PEDESTALS
104-TDC DATA PEDESTALS
105-ADC DATA CAMAC ADDRESSES
106-TDC DATA CAMAC ADDRESSES
107-ADC DATA READ FUNCTIONS
110-TDC DATA READ FUNCTIONS
111-ALC HISTOGRAM ID
112-TDC HISTOGRAM ID
113-DRIFT CHAMBER READOUT PARMS
114-SCALER CAMAC ADDRESSES
115-ADC DATA
116-TDC DATA
117-DRIFT CHAMBER TIME, ADR
120-SCALER DATA
121-DRIFT CHAMBER WIRE PEDESTALS
122-DRIFT CHAMBER WIRE GAINS
CONSTANT GROUP? 105
(These are the CAMAC address variables. The
numbers in parentheses are decimal and are the
crate, station, and subaddress values which have
been decoded from the octal QBUS address which is
listed just after them.)
AA0 : (1,10,0) : 164500; AA1 : (1,10,1) : 164502;
AA2 : (1,10,2) : 164504; AA3 : (1,10,3) : 164506;
AA4 : (1,10,4) : 164510; AA5 : (1,10,5) : 164512;
AA6 : (1,10,6) : 164514; AA7 : (1,10,7) : 164516;
AA8 : (1,10,8) : 164520; AA9 : (1,10,9) : 164522;
AA10 : (1,10,10) : 164524; AA11 : (1,10,11) : 164526;
CONSTANT GROUP? 115
AD0 : 115; AD1 :   0; AD2  : 129; AD3  : 145;
AD4 : 105; AD5 :   9; AD6  :   9; AD7  : 125;
AD8 : 115; AD9 : 110; AD10 :  29; AD11 : 105;
CONSTANT GROUP? <cr>
(The <cr> ends the CONLIS dialogue.)
```

DATA ACQUISITION AND LOGGING

When an ATROPOS system is created, the appropriate routine must be supplied to respond to the event interrupt. The CONSTANTs contain the CAMAC addresses of the modules from which the data is read. These are set by default at system initialization time, but may be changed by the user interactively at any time. Basically all the routine does is read the data, call a system routine to log it, and pass the data onto the sampling routine which is also user supplied and invoked at a lower level. In order to change the 'device' to which the data is logged (the TRIPLEX, magnetic tape, or floppy disk), one has to specify the appropriate object module library and relink the program. No actual coding changes are necessary. Due to the memory required, one cannot create a system which contains simultaneously all three logging facilities. All logging is interrupt driven and there is provision for both synchronous and asynchronous operation. All logical records passed to the logging routine are buffered. In the case of the tape and floppy devices multiple buffers are used. The number (and length in the case of the tape device) is determined dynamically by the event data rates and the available memory. Each physical record includes information to identify the computer system which is generating it and the current run number. Data descriptors are included with the tape and floppy device data to simplify conversion of data types (e.g., floating point, integer, ASCII character strings) from LSI-11 format to IBM 370 format. Event data records also include the event number to provide a check on the data. In the case of the floppy disk logging, each run is written as a separate file in RT-11 format.[8]

* In the examples user responses are underlined, and all user responses are terminated with a carriage return. The symbol <cr> refers to a carriage return typed before any other input on that line (i.e., a null response). Lower case text is a comment on the example which would not appear in a real session.

ACQUIRE - THE TRIPLEX LOGGING TASK

For the logging to the TRIPLEX, ATROPOS communicates with a job ACQUIRE,[9] which is running under the interactive time sharing system ORVYL.[10] The TRIPLEX terminal communication protocol is basically half duplex with a DC1 character being sent by the TRIPLEX to turn the line over to the terminal. Characters sent by the 'terminal' when it does not own the line are ignored by the TRIPLEX apart from a BREAK which interrupts the TRIPLEX and, after it has been processed, causes the TRIPLEX to turn over the line. Characters may be sent by the TRIPLEX at anytime. When ACQUIRE is attached, it intercepts any messages from ATROPOS to the TRIPLEX, and processes them. There are three types of messages sent by ATROPOS to ACQUIRE: data messages, ACQUIRE commands, and regular WYLBUR commands. The protocol for the ATROPOS/ACQUIRE communication enables ACQUIRE to distinguish between the three kinds.

Data messages start with a DC3 character. They also have a checksum for the data record on the end of them. The communication line to the TRIPLEX only supports 7 bit character transmissions and some of those characters are used as control characters by the communications controller at the TRIPLEX. Therefore each 8 bit byte of binary data is encoded into two 7 bit ASCII hex characters before transmission. When ACQUIRE receives a message with a DC3 on the beginning of it, it computes a checksum, decodes the binary data, saves the data record in the data file (if the checksum is good), and sends a reply back to ATROPOS as to the status of that record. If the record had a checksum error, ACQUIRE notifies ATROPOS and the message is retransmitted. The transmission rate that can be reasonably sustained over a 2400 baud communication line is about sixty 8 bit bytes/second of logical data.

ACQUIRE commands are commands typed by the user that (in most cases) start with an equals sign ('='). These commands allow the user to control the analysis which ACQUIRE is performing on the data which is logged to it. For example, the user may define or redefine the ACQUIRE histograms, clear them, display them (on the LSI-11 terminal or any TRIPLEX graphic device (e.g., the Versatec plotter)), or manipulate the format in which the ACQUIRE histograms are displayed.

Any command not preceeded by an equals sign or a DC3 character is checked against a list of ACQUIRE commands, and if it is not there then it is sent by ACQUIRE onto WYLBUR for processing. In this fashion, the terminal can be used to communicate with WYLBUR and hence be used for program development etc. even while taking data.

Since the ACQUIRE task is written in IBM FORTRAN IV it may be modified by the experimentor. Further it has access to almost all the facilities of the TRIPLEX. For example, it uses the facilities of DPAK and HPAK[11] and Unified Graphics[12] for displays and histogramming, disk mass storage facilities (IBM OS data sets), and the output facilities (lineprinter, Versatec plotter, and other peripherals) available on the TRIPLEX.

HISTOGRAMMING FACILITIES IN ATROPOS

Histograms provide a convenient way of analyzing and displaying the data which is being accumulated. In order to provide as much flexibility as possible, our facilities allow the user to define and or redefine the histograms both at compile and execution time. Up to 63 histograms (ID's range from 1 to 77 octal) can be defined at any given time. For each the user must specify the ID, the number of bins, the low bin, the bin width, and a title which is displayed when the histogram is displayed. The output facilities are modularized so that one can easily change display devices without a lot of recoding. Typically we use a Tektronix 4013 terminal for a console terminal and display the histograms on that.

Histogram commands available are:

* HDEF which enables the user to define a histogram. He supplies an ID, number of bins, low bin value, bin width, and histogram title. For example:

```
·HDEF
ID? 1
# of BINS? 100
LOW? 0
WIDTH? 5
TITLE? TOTAL NUMBER OF WIRES FIRED EACH EVENT
·                (Up to 80 characters)
```

* HCLR which prompts for the ID of the histogram to be cleared. A response of 0 clears all the histograms. The push button HCLR command uses the contents of ETWO as the ID.
* HDEL which prompts the user for the ID of the histogram to be deleted. A response of 0 deletes all histograms.
* HOUT enables the user to display the contents of a histogram in a graphical (Fig. 3) or tabular form.
* HSUM displays the statistics of a specified histogram. For example:

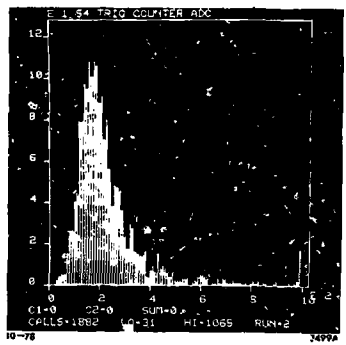

Fig. 3. Example of the graphical display of an ATROPOS histogram.

```
·HSUM
ID=30
HSUM FOR:  PESTOV 1 TDC
  ID : 24   LBIN : 250   HBIN : 448   CALL : 6280
UFLO : 28   OFLO : 6125
Y SUM= 6280.000   XMEAN= 444.5710   SIGMA= 24.23909
```

* HLOG provides the user with a way of saving (on the logging device) the histogram contents for future reference.
* HGET allows the user to examine all or one of the histogram specifications. For example:

```
· HGET
ID? ?   (Response of '?' or 0 results in the
         listing of all defined histograms)
ID=10,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=S1 TRIG COUNTER ADC
ID=11,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=S2 TRIG COUNTER ADC
ID=12,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=S3 TRIG COUNTER ADC
ID=13,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=S4 TRIG COUNTER ADC
ID=14,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=PESTOV 1 ADC
ID=15,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=PESTOV 2 ADC
ID=16,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=PESTOV 3 ADC
ID=17,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=PESTOV 4 ADC
ID=20,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=PESTOV 5 ADC
ID=21,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=PESTOV 6 ADC
ID=22,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=PESTOV 7 ADC
ID=23,TYPE=2@,#BINS=100,XLOW=0,XWID=10,
      TITLE=PESTOV 8 ADC
ID=30,TYPE=2@,#BINS=100,XLOW=250,XWID=2,
      TITLE=PESTOV 1 TDC
ID=31,TYPE=2@,#BINS=100,XLOW=250,XWID=2,
      TITLE=PESTOV 2 TDC
ID=32,TYPE=2@,#BINS=100,XLOW=250,XWID=2,
      TITLE=PESTOV 3 TDC
ID=33,TYPE=2@,#BINS=100,XLOW=250,XWID=2,
      TITLE=PESTOV 4 TDC
ID=34,TYPE=2@,#BINS=100,XLOW=250,XWID=2,
      TITLE=PESTOV 5 TDC
ID=35,TYPE=2@,#BINS=100,XLOW=250,XWID=2,
      TITLE=PESTOV 6 TDC
ID=36,TYPE=2@,#BINS=100,XLOW=250,XWID=2,
      TITLE=PESTOV 7 TDC
ID=37,TYPE=2@,#BINS=100,XLOW=250,XWID=2,
      TITLE=PESTOV 8 TDC
```

EXPRESSION FACILITIES

Very often the user would like to look at various transformations of the data the user is taking, and the user wants to be able to change these transformations easily. The simple expression definition and evaluation facilities provided by ATROPOS allow him to do this. There are basically five parts to the expression facilities.

* The expression definition facility is entered by the command EXPDEF. The command EXPDEF prompts the user for an INDEX, an expression, any conditions the user wants to attach to the evaluation of that expression, and an ID of a histogram where the result is to be histogrammed. If no histogram of the result is desired, the user responds 0 to the ID prompt. The expressions are evaluated in strictly left to right order. There is no heirarchy of operators (they all have equal precedence) and parentheses cannot be used to group subexpressions. The result of any expression can be used in a later expression in which case the previous expression is referenced by preceding its index with '%:'. If, at evaluation time, the previous expression was not evaluated because one of the attached conditions was not satisfied, then a zero is used for its value. Elements of the indexed CONSTANTs are referred to by giving the name followed by a colon followed by the index. Every operator is denoted by a single character. The operators allowed in the arithmetic expressions are +,*,-,/, > (arithmetic shift right), < (arithmetic shift left), & (logical and), | (logical or), _ (an underscore for minimum function), and a ^ (a carat for maximum function). In the conditionals, > (greater than), < (less than), = (equal to), and # (not equal to) are permitted. Note that the conditional A>B>C is equivalent to A>B and A>C, and not A>B and B>C. An operand may be a named CONSTANT or a decimal, octal or hexadecimal integer. Each expression is checked for syntax when it is entered by the user. If at evaluation time an error (overflow or undefined CONSTANT) is detected, the entire expression string is printed on the console terminal with a pointer to the operator where the error occurred. Note that 32 bit intermediate results are propagated if the next operation is an addition, subtraction, or division. For all other operations, if the intermediate result is greater than 16 bits, then an overflow is presumed to have occurred and the evaluation of that expression is aborted with a warning. Although the expression facility is very simple, it appears to be adequate for our purposes and requires little memory for expression storage or for parsing and evaluation. As an example of the expression definition use: suppose we have 3 ADCs (analog to digital converter), each with a pedestal, and 3 associated TDCs (time to digital converter); we want to histogram the residual of each ADC reading and its pedestal, and the TDC/100 associated with the largest residual.

```
·EXPDEF
INDEX=1              (index by which the expression
                      is defined)
EXP=ADC1-APED:1      (the expression)
COND=<cr>            (no conditions attached)
ID=1                 (histogram result in histogram
                      with ID=1)

INDEX=2
EXP=ADC2-APED:2
COND=<cr>
ID=2
INDEX=3
EXP=ADC3-APED:3
COND=<cr>
ID=3
INDEX=4
EXP=TD1
COND=%:1 > %:2 > %:3
COND=<cr>
ID=0
INDEX=5
EXP=TD2
COND=%:2 > %:3 > %:1
```

```
COND=<cr>
ID=0
INDEX=6
EXP=TD3
COND=%:3 > %:1
COND=%:3 > %:2
COND=<cr>
ID=0
INDEX=7
EXP=%:4 + %:5 + %:6/100
COND=<cr>
ID=4
INDEX=<cr>
.
```

* EXPDEL prompts a user for the index of the expression which is to be deleted, then deletes it.
* EXPLIST lists all of the defined expressions, their associated conditions, and histogram ID. For example:

```
. EXPLIST
  : ID=000001 : ADC1-APED:1
2 : ID=000002 : ADC2-APED:2
3 : ID=000003 : ADC3-APED:3
4 : ID=000000 : %:1>%:2>%:3 : TD1
5 : ID=000000 : %:2>%:3>%:1 : TD2
6 : ID=000000 : %:3>%:1 AND %:3>%:2 : TD3
7 : ID=000004 : %:4+%:5+%:6/100
.
```

* To perform the evaluations at the appropriate time, the user puts a call to EXPXEQ in his code. This will usually be in the sampling routine. EXPXEQ will evaluate all the expressions, and histogram the results in the appropriate histogram for those which were successfully evaluated. The command EXPEVAL executes an EXPXEQ call. This is usually used for testing purposes.
* EXPSHOW lists the results from the last EXPXEQ call, along with a flag which indicates whether a given *expression was successfully evaluated.*

TAILORING AN ATROPOS SYSTEM

Over the past two years we have tailored and used three substantially different ATROPOS systems.++ In addition, at least two other ATROPOS systems have been tailored by other groups at SLAC. In general we have found the software structure makes it simple to tailor a system to meet a new equipment configuration. In fact we have used the ATROPOS structure as a base for two other unrelated monitoring (CLOTHO) and control (LACHESIS) systems we needed.+++

The following subsections describe the changes needed and the routines that must be supplied in the tailoring of a system.

* The CONSTANTs

The CONSTANTs (their names, memory allocation, check routines and group names) must be set up according to what the system being tailored requires. Although this must be done very carefully, it is a fairly simple process.

* UINIT

UINIT is the basic system initialization routine. It should load the CONSTANTs and initialize them, set up the interrupt vectors for the clock interrupt, the CAMAC interrupts, initialize the keyboard processing, initialize any auxiliary devices (such as tape drives, floppy disk, auxiliary displays), and define the default histograms.

* UMAIN

The group of routines we call UMAIN are the run control routines. UBEGIN, UEND, URESUME, UPAUSE are called when the user does a BEGIN run, END run, RESUME run, or PAUSE run, respectively. In these routines the programmer must enable/disable the event data interrupts, clear the histograms (usually at BEGIN run), and take any other appropriate actions.

* EVENTLAM

This is the routine which handles the event interrupt. It reads the data and does any logging that may be necessary, and then passes the data onto the sampling routine (also user supplied and user invoked).

* UCLOCK

The user must supply this routine to respond to the clock interrupt if the target machine has a clock. For an ATROPOS system which does not have a clock, this routine can be omitted.

* BKGRND

Although ATROPOS is completely interrupt driven, we have found it advantageous to do some things in background as CPU time allows. For example, updating the console display and driving the auxiliary display are done from the background. In the case of our auxiliary display, every 1/30th of a second, the clock routine posts a command to the background queue to update the display if a command is not already present in the background queue.

Currently the user supplied routines must be written in PL-11 or MACRO-11. Soon we hope to have PASCAL[16] and FORTRAN[17] programming capabilities added to our LSI-11 software development facilities on the TRIPLEX.

CONCLUSIONS

The ATROPOS system described herein is quite useful for our needs, although it does have its limitations. Since we do not run under a disk based operating system such as RT-11, we do not have all the flexibility that we could have with, for example, overlays. With the memory constraints relieved by

++ The first system we developed was used in the SLAC C-beam in December 1976 to test a new shower counter design.[14] In summer 1977 we tailored a system which we used to develop a hardware method of generating random bits.[15] That one entailed the logging of several million bits of data to the TRIPLEX. Currently we are using ATROPOS to develop detectors for a future experiment.

+++ CLOTHO, LACHESIS, and ATROPOS are the names of the three fates.[13] CLOTHO is a system which monitors the polarized electron source located in the SLAC injector. CLOTHO was the youngest of the three fates. She spun the thread of life. LACHESIS is the system which provides closed loop feedback position and energy steering of the SLAC beam in beam line 'A' of the beam switchyard. LACHESIS was the fate who determined the length of the thread of life. ATROPOS (the system described in this paper) provides analysis facilities for the result of the beam hitting a target. ATROPOS was the fate who cut off the thread of life.