LEGAL NOTICE

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or

B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

EXPERIENCE WITH THE WILLIAMS LABORATORY ON-LINE COMPUTER SYSTEM*

MASTER

Russell K. Hobbie and Robert W. Goodwin

University of Minnesota

ABSTRACT: A CDC 3100 was installed in 1965 and has been used for all experiments since the accelerator became operational. Fast data access is obtained by a direct route to memory which is not under program control. The advantages and disadvantages of such a system are discussed. A 2500-word operating system reserves data arrays, loads relocatable programs, processes real-time interrupts and provides typewriter control of program execution. For flexibility, all programs can be called either by other programs or by the control typewriter. Most user-written programs are in Fortran. Fortran programs can reference the data arrays reserved by the resident. A simple program storing system using the end of the library tape facilitates secondary computer use during long runs. Measurement of excitation functions has been almost completely automated with computer control of the accelerator energy. Improvements to the Digitek Fortran-II are described. (auth)

Our computer system, built around a CDC 3100, was installed in July 1965. This paper summarizes our three years experience with the computer and the operating system which we developed.

The first decision to be made in designing a system is the means of rapidly transferring data into the computer. One way to do this is under program control on one of the regular input/output channels. This method prevents writing in the wrong location, but it requires a priority interrupt system, a fast data channel, and rapid response time. Alternately, one can provide an interface which, on a cycle stealing basis, writes directly in the computer's memory at an address provided by the outside world. This method is often rejected because of the danger of altering the wrong cell of the memory. In spite of this danger, the method has several advantages. It is fast, and the necessary interface is fairly simple to make. The cycle stealing takes no overhead time from the program being executed. Since other demands from the experiment (such as starting and stopping counters and reading values from scalers) can tolerate delays of a few milliseconds, they can be serviced by interfaces attached to a regular data channel and the normal interrupt handler. We chose this latter method and have not regretted it. The dreaded mistakes which were predicted have proved to be quite infrequent. If the worst does happen, re-loading the operating system and one's programs takes only a minute or two. Programming the computer to calculate while taking data is simplified, since the data collection is independent of the processor. We have, for example, collected data, displayed a spectrum on the oscilloscope (updated, say every ten seconds, by an interrupt from the real time clock) and compiled a Fortran program, all at the same time.

fey

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.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Almost any experiment can be handled by building the interface, (which we call the Direct Memory Unit), to transfer data to the computer in two modes. In Mode A, the data given to the DMU is treated as an address and one is added to that word of memory. The system functions as a pulse height analyzer, with the least significant bits of the address determined by the ADC output and the upper bits set to the desired starting address of the array. In the other mode, Mode B, the descriptor of each event is written into a buffer list in the memory (the current address in that list being stored in the DMU), and an interrupt is sent to the computer when the list is full. This list is then processed while an adjacent list is being filled. Should more than 24 bits be needed to describe an event, adjacent words in the list can be used to contain the description.

The hardware configuration is shown in Fig. 1. The CDC 3100 is a 24 bit machine with a 1.8 μ sec cycle time and a very complete instruction set. We have 16K of memory. After starting with 2 magnetic tapes, paper tape and typewriter, we found that a line printer, card reader (homemade) and third tape were essential. Since these were added, the paper tape has fallen into disuse. A block diagram of the DMU is shown in Fig. 2. It consists of a memory contents register, C, and two address registers: one for the add-one mode (A) and one for the buffered list mode (B). Input to the DMU consists of the datum, a request signal called Ready To Transfer, and a line indicating whether the datum is to be analyzed in Mode A or Mode B. A Transfer Accepted signal is sent to the input device 0.5 μ sec after RTT becomes true. Further RTT signals are ignored until the DMU operation is complete. This takes 2-3 μ sec for Mode B or 5-7 μ sec for Mode A. The DMU is fed by a scanner which multiplexes 7 inputs to the DMU input. If no input requires servicing the 7 inputs are scanned every 4 microseconds. ADC's are connected to scanner inputs via small individual patch panels. The total cost of the DMU-Scanner was \$5600 for parts and \$5000 for design, construction and documentation.

The Display Driver, another device built by us, is connected to an I/O channel. It operates a point display oscilloscope and light pen, a Moseley x-y plotter, scaler start/stop/reset lines, 24-bit input and output busses, and 24 relays. The oscilloscope and plotter are driven by common DAC circuits with 10 bit resolution in both directions. The plotting speed of the oscilloscope is limited by the speed of the data channel to 1 point every 24 μ sec. The digital output is presently used to set a preset scaler for beam current integration, transmit the desired NMR frequency to the accelerator automatic energy control, and send simulated data to the DMU for systems tests. The digital input is presently being used to read a digital voltmeter. The cost of this unit was \$11,000 for parts, including the oscilloscope and plotter and about \$5000 for design and construction.

One of the devices fed by the Display Driver is a unit to control the accelerator's energy.¹ When supplied with the desired frequency by the computer, it sets the NMR oscillator and then adjusts the magnet to produce resonance at that frequency. It can make energy changes of 25 keV in 30 sec.

The operating system which we developed to provide an interface between man and the machine is called RESIDENT and occupies about 3000 words of storage.² It is designed to accept a typewritten program request from the Operator, execute it, and then wait for the next request. Execution may be forced to terminate by pressing a manual interrupt button. The simplicity of this structure allows for considerable flexibility. The operator is able to change his mind at any time (and often does). Programs may be loaded from a user's tape or the library tape all at once or separately. Some or all of the programs loaded may be deleted to make room for additional programs. There is no job structure, such as exists in many batch processing operating systems. Programs in core may be called either by the operator or by another program. Assignment of logical unit numbers to physical devices may be changed by the operator, for example, to read data from tape rather than cards. RESIDENT also allows some foreground/background operation.

A major portion of RESIDENT is devoted to providing buffered or non-buffered I/O and interrupt handling. Routines called READ and WRITE simplify I/O programming by providing automatic error checking at the end of the operation. The general interrupt handler can be requested to call user interrupt

routines on detection of some interrupt condition such as the Mode B list interrupt from the Direct Memory Unit. RESIDENT also includes the floating point arithmetic simulation routines, which require about 250 microseconds per operation.

The operator, in setting up his experiment, must reserve certain areas of memory for data accumulation by the DMU. A program in RESIDENT assigns this storage, and tells the experimenter how he must set his patch panels so that data are written at the proper addresses in core. The system then reserves those areas of memory until they are released by a special control statement. In effect, then, these areas become a part of the RESIDENT.³ Programs may then be loaded, executed and cleared at will without effecting the data arrays. (Fig. 3) The experimenter may, by use of a special declarative statement added to our Fortran compiler (discussed below), refer to these data arrays in his programs. There are actually two classes of these arrays--DEFINE and DEFINEAB arrays. DEFINEAB arrays are used for direct data accumulation as we have just described and must have starting addresses which are a power of two and an exact multiple of the length of the array. DEFINE arrays do not have this restriction and may start in the first available memory location. They are often used as a kind of labeled COMMON storage, since any program in core may refer to these arrays.

Under the RESIDENT system, programs remain in core and may be called by the operator any number of times. Thus, on one call, the user can set parameters for use in subsequent calls. Therefore, once a program is initialized, the experimenter is reluctant to discard it so that someone else can make use of some processor time that may be available while a run is in progress. A means for quickly (10 seconds) saving all of core beyond the DEFINED arrays on the end of the library tape was developed in a program called SAVE.⁴ Then this memory becomes available for loading and executing other programs. When control is to be returned to the experimenter, program RESTORE retrieves his programs exactly as they were before.

The compact version of Fortran supplied by CDC was the Digitek Fortran which has been used fairly widely on several different computers. It was designed to fit a 4096 word machine, and consisted of a compiler, a loader, and a collection of routines called by the object code to handle DO-loops, I/O, etc., known as RUNTIME. None of these were relocatable. The I/O equipment used for compilation could not be changed by the user. Operation was as follows: the library tape was autoloading to bring in the compiler. The compiler generated object code in one pass through the source deck. The loader was then read in over the compiler, the object tape rewound, and the object program loaded into memory starting at location 4000g. The necessary library routines were loaded, RUNTIME was read into the lower 4000g words of memory, and the program was executed. The object program generated by the compiler contained undetermined addresses, since the code was generated one source statement at a time. These were filled in by the loader, which ought really to be regarded as doing the second phase of the compilation.

We modified the compiler to fit our system. The new version of Fortran is maintained as an assembly language source deck which can be assembled on an 8K machine. The compiler operates as a relocatable library program which can be called any number of times with input and output units specified as arguments. RUNTIME has been replaced by a set of relocatable library routines, or, for DO-loop checking and argument transmission, by compiler-generated coding. The PROGRAM statement has been added and is treated like a SUBROUTINE statement. The END statement causes a return to the calling program. Array elements with constant subscripts are referenced directly. Hollerith and octal constants were added. Error stops have been converted into error messages which do not stop the computer. The basic compilation speed is 400 statements/minute.

Certain areas of memory, as discussed above, are reserved by RESIDENT for the direct input of experimental data. Since nearly all experimenters wish to write their processing routines in Fortran, it is necessary to provide some means to reference these arrays. This has been done by adding the statement

DEFINED Namel,Name2,.....

to Fortran.³ This statement must come immediately after a PROGRAM or SUBROUTINE

statement, and the names listed must be the names of DEFINED arrays in the entry point table at the time of execution. The variables may be further described by REAL, INTEGER, or DIMENSION statements. The DEFINED statement does two things. During compilation it reserves a temporary location in the subroutine for each defined variable and then treats it as an additional subroutine argument. It also produces coding which, during execution, calls the routine CHKDEFIN. This routine receives the BCD name of the defined array and returns with the first word address of the array, which it obtains from the Entry Point Table in RESIDENT. The subroutine then stores this address in the temporary location set aside for the Defined variable. If the variable with that name is not in the entry point table, an error message results. These changes have increased the size of the compiler from 3500 to 4360 words.

The writing of this system, the modification to Fortran and some modification to the manufacturer's assembler took about 1 man year to design, program and document. Details of the hardware⁵ and the programs⁶ are available on request.

Footnotes

* Work supported in part by the U.S. Atomic Energy Commission.

1. Clark Bergman and R. K. Hobbie, Control System for 90 Analyzing Magnet, AEC Report C00-1265-70 (1968).
2. R. W. Goodwin and R. K. Hobbie, "A Flexible Resident System for the Small 3100," SWAP-12 Proceedings (CDC Users' Group), p. 212 (1967).
3. R. K. Hobbie and R. W. Goodwin, Nucl. Instr. and Methods, 52, 119 (1967).
4. R. K. Hobbie and R. W. Goodwin, Nucl. Instr. and Methods, 58, 342 (1968).
5. R. K. Hobbie, 3100 Computer System Interface Instruction Manual; AEC Report C00-1265 (1966).
6. R. W. Goodwin and R. K. Hobbie, Programmers' Manual, Version 2, AEC Report C00-1265-48 (1967).

Fig. 1. Block diagram of the system.

Russell K. Hobbie and Robert W. Goodwin, Experience with the Williams Laboratory On-Line Computer System, C00-1265-76.

MODE A (ADD ONE) OPERATION.

MODE B (BUFFERED) OPERATION

Fig. 2. Block diagram of Direct Memory Unit. The DMU contains three registers: A and C are used in Mode A; B and C are used in Mode B.

Fig. 3. Map of the memory showing the location of DEFINED arrays and RESIDENT.

EXPERIENCE WITH WILLIAMS LABORATORY USERS PROGRAMS*

Robert W. Goodwin and Russell K. Hobbie

University of Minnesota, Minneapolis

ABSTRACT: We have found 60% of our operation is in a pulse-height analysis mode using up to eight detectors. The remainder involves multiparameter events from particle identification telescopes and position sensitive detectors. For pulse height analysis, special routines are provided which the user incorporates in Fortran programs suited to his experiment. These routines can be called from the typewriter for preliminary runs. Peak identification is facilitated by calculating the kinematics for each reaction and storing an energy calibration for each detector. This calibration is least-squares fitted to the data and is adjusted each time a new run is analyzed. For multiparameter work, machine language packages have been provided which can process events at average rates of 7500 counts/sec. Much to our surprise, the experimenters have not stored raw data on tape for subsequent analysis, although facilities to do so were provided. The computer has also been used extensively for phase shift fits and other routine calculations.

This paper will discuss our experience with the interaction of an experimenter with the system described in the previous paper.¹ Perhaps the most important question is computer availability. Most of our users have proved to be cooperative people who are willing to use SAVE and RESTORE (described in Ref. 1) and allow someone else on the computer while they are collecting data but not actually analyzing it. In spite of this, our biggest bottleneck is off-line data analysis when the computer is tied up with an experiment. Our plans to alleviate this include a small disk providing the buffer storage for two displays and a simple program swapping facility. We expect this to be in operation in about a year.

About half of our use has been in Mode A, for routine pulse height analysis. In a typical experiment, data from two or three peaks in each of eight detectors were converted to cross sections during the experiment. This analysis was done using the oscilloscope and light pen, with a general display program² called by the experimenter's own Fortran program. Since the major problems in this analysis were speed and peak identification (particularly when the cross section went through a minimum) we decided to store linear calibration constants for each detector in the user's program. His program calculated the kinematics for the reaction, and, using the calibration, identified the channel where the peak should be. The display program then displayed that region of the spectrum with the peak marked. The experimenter had only to adjust the peak limits with the light pen; this corrected information was then used to update the least-squares fit to the detector calibration. After a few runs, almost no corrections were necessary. This method allowed analyzing one peak every two or three seconds. After the analysis, both the raw data and calculated cross sections were stored on magnetic tape. When excitation functions were taken, this same program caused the magnet controller¹ to automatically change the beam energy at the end of each run. Enough library routines are available so that the casual user can use the system as a pulse height analyzer (including display and peak summing) by simple calls from the typewriter.

Most of the rest of our operation has been in the buffered list mode, Mode B, for two-parameter experiments. Some of this work has involved particle identification with a ΔE -E system,³ calculating $F = (E + \Delta E)^k - E^k$ using a stored table of values of E^k . Currently the program accommodates up to seven pairs of detectors. The other two-parameter experiments have involved position sensitive detectors in the focal plane of a magnetic spectrometer.⁴ The momentum of the particle is related to position x , and the type of particle to E , the energy it loses passing through the detector. The value of x is calculated from the input signals E and $x\Delta E$. Again, the program can handle up to seven detectors. Both two-parameter programs process lists of 128 events at a time, at an average rate of 6000-7500 events/sec. This time does not include the time required to store the raw data on magnetic tape; our slow tape units can write the data only at 2000-5000 events/sec. Very few of our experimenters have found it worthwhile to store their raw data in Mode B; they prefer to store spectra after the particle identification has been made.

The setting of parameters such as k , or thresholds between particle types, is again facilitated by the display and light pen. One can display either a two-parameter plot of the last 256 events or the projection of all events on the axis corresponding to particle type. Figure 1 shows an x - E plot for the position sensitive detector. One can see that the division improves the resolution by eliminating the effect of straggling for the particles passing through the detector.

One surprise has been the heavy use of the Moseley point plotter. This is connected in parallel with the oscilloscope, and pressing a jump switch on the console causes it to plot whatever was being displayed on the oscilloscope. This facility has been used much more than we anticipated; in fact, photographs of the oscilloscope are hardly ever taken.

It has proved useful in many cases to have a display which is normalized to the maximum count each time the display is updated.

In determining cross sections by measuring the decay rate of reaction products, it is necessary to know the history of the beam intensity so that one can correct for the finite exposure time. This was done formerly by using only runs in which the beam current was quite steady, or by using a "leaky integrator" on which the charge leaks off at the same rate that the desired state decays. The same thing can be done digitally by causing each count from the beam current integrator to generate an interrupt. Comparing its time of arrival with the one-millisecond real time clock, one can generate a profile of the beam intensity as a function of time.⁵ This profile is used to correct for beam intensity fluctuations, and it has the advantage over the "leaky integrator" that the same beam exposure can be used for several different half life measurements.

The system has also been rather extensively used for phase shift searches, optical model and Coulomb excitation calculations. We have been pleasantly surprised that these calculations have not been unduly long, even with our 250- μ sec programmed floating point instructions.

Footnotes

*Work supported in part by the U.S. Atomic Energy Commission.

1. R.K. Hobbie and R.W. Goodwin, "Experience with the Williams Laboratory On-Line Computer," Proceedings of Conference on Computer Systems in Experimental Nuclear Physics (1969), AEC Report C00-1265-76.
2. The display programs and other special library routines are described in R.W. Goodwin and R.K. Hobbie, Programmers Manual, AEC Report C00-1265-48 (1967).
3. D.C. Weisser, Ph.D. Thesis, University of Minnesota, 1968 (unpublished).
4. P.H. Debenham, D. Dehnhard and R.W. Goodwin, Nucl. Instr. and Methods (to be published).
5. R. Barnett and J.H. Broadhurst, private communication.

Fig. 1. Display of data from a position sensitive detector showing E versus X.