NUCLEAR INSTRUMENTS AND METHODS 160 (1979) 165-171 ; © NORTH-HOLLAND PUBLISHING CO.

A HIGH-SPEED, ECONOMICAL MINICOMPUTER

D. E. CARTER

Physics Department, Ohio University, Athens, Ohio 45701, U.S.A.

Received 9 May 1978 and in revised form 2 November 1978

A high speed, low cost 20-bit minicomputer has been designed and constructed. It is based on the standard 7400[1]) series of transistor–transistor logic.

1. Introduction

Current work in experimental physics requires sophisticated equipment not only to record the events but also to process them before storing. Multiplication of signals as in the case of telescope detector systems or division of signals as in the case of position sensitive proportional detectors are examples of the above. To meet these requirements and at the same time replace a multichannel analyzer system, a low cost 20-bit minicomputer was developed.

2. Design objectives

Our application required a high speed, inexpensive and reliable on-line data taking device which could be interfaced to our existing IBM 1800 computer system. Since an existing computer system was available, the new minicomputer would not need to contain large amounts of memory, it could use the I/O devices and bulk storage of the existing system, and the high level software could be used to write assemblers, loaders, etc. for the minicomputer.

A word length of 20 bits was desired which would allow integer arithmetic operations having a magnitude in excess of one million. This word length would allow most arithmetic to be performed as single word integer operations, saving time, hardware and memory.

The microprocessor approach seemed to be the least expensive but suffered in the speed and word length requirements. A purchased minicomputer could have the required speed, but would be too expensive and most have unneeded features.

For the above reasons, a 20-bit machine was designed around the standard 7400 TTL logic. The 7400 series gave the needed speed to execute all machine instructions in 1 μs using a 20 MHz clock. The 7400 series is also inexpensive, reliable and readily available.

The 250 ns version of the INTEL[2]) 2102A-2 RAM (random access memory) integrated circuits were used to construct the 8k words of 20-bit memory. These were chosen as a compromise between speed and cost.

3. System architecture

A block diagram of the OU-8000 minicomputer is shown in fig. 1, views of it are given in figs. 2–4. Data flow and sequence is controlled by the ROM (read only memory) and system clock. Specialized instructions may be formed by altering the ROM code, as it controls all the data flow. Four Harris[3]) 7603 ROMs are used to decode the instruction sequence. The ALU (arithmetic logic unit) is composed of five 74181 chips. One maskable hardware interrupt is provided. This interrupt may be software vectored to obtain many interrupt entry points. Interrupt levels are not re-entrant, i.e., when executing on the interrupt level no additional interrupts are accepted. Interrupts are serviced within 1 μs of receipt of the interrupt request.

Eight 20-bit output ports and eight 20-bit input ports are provided. Each output is provided with an output strobe to indicate when the output data is valid. Each input is provided with an input acknowledge which signifies that the input has been read. This is valuable when handshaking is desired between the OU-8000 and its peripherals.

4. Instruction set

The designed instruction set has proven to be powerful, efficient and easy to use. It contains 49 basic instructions. These instructions may be divided into eight groups. These are direct memory reference, indexed, register to register, branch, input/output, shift, interrupt control and miscellaneous instructions. Four general purpose registers are used in conjunction with these instructions as

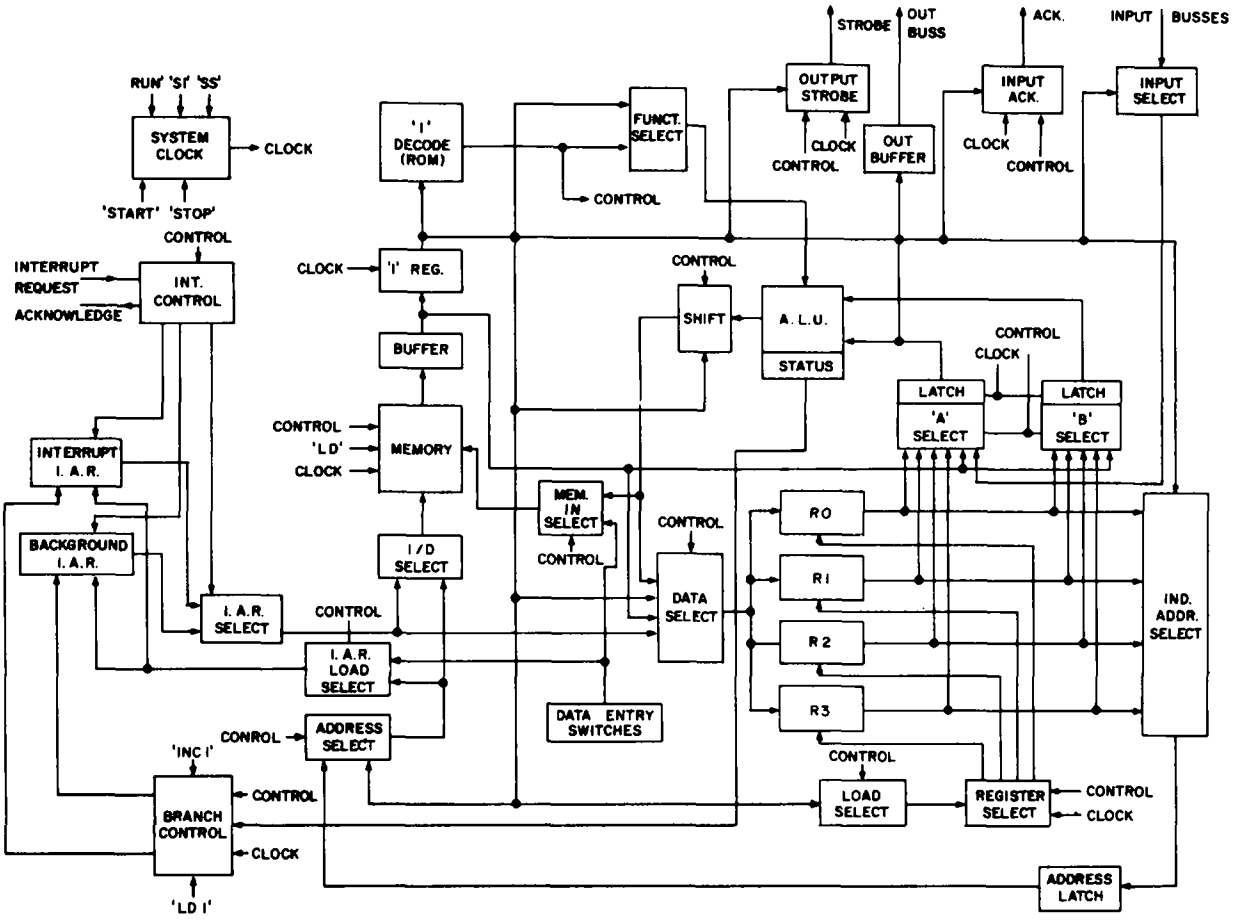

Fig. 1. OU-8000 block diagram.

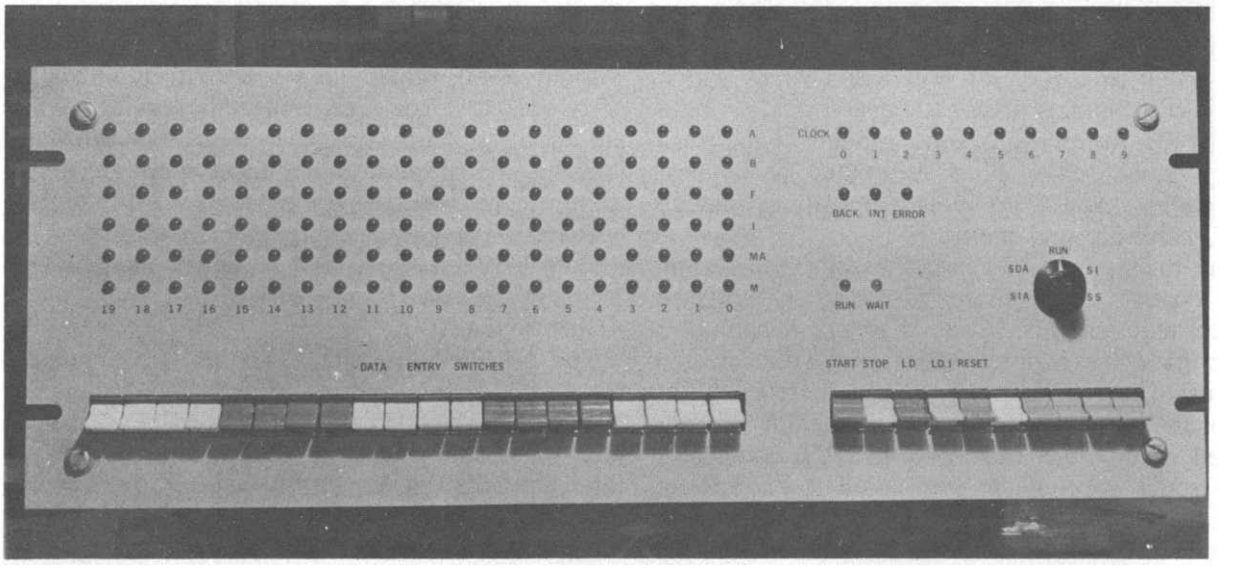

Fig. 2. OU-8000 front panel.

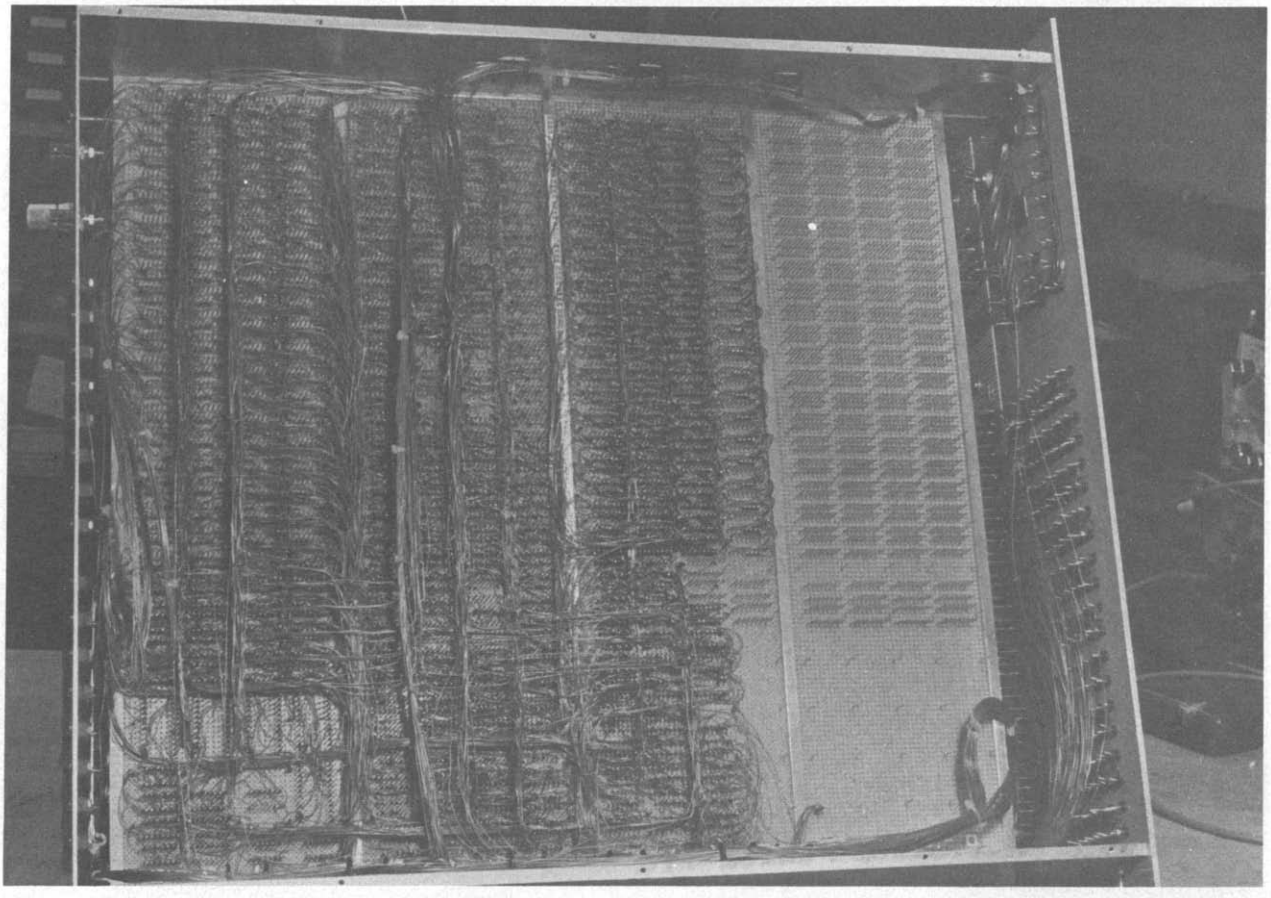

Fig. 3. OU-8000 top view.

accumulators, index registers, loop counters, subroutine linkage registers or for temporary data storage. The general purpose, multiple register structure has proven to be a very powerful system architecture.

The arithmetic and logical operations include add, subtract, AND, OR, EXCLUSIVE OR, increment, decrement, complement, set to zero, and set to minus one. Conditional branch instructions include branch if negative, branch if positive, branch if zero and branch if not zero.

The direct memory reference group of instructions performs the indicated operation using the operand as a memory address. This address is calculated by the assembler at assembly time and inserted into the machine instruction. For example, the assembly instruction *A 1 OPERND* would add the contents of memory location OPERND to register one.

The indexed instruction group normally requires three registers to be specified. The first register entry specifies a register which contains data to be operated on. The second register specified contains an address of data in memory. The indicated operation is performed on these two data words and the result is stored in the third register specified. For example, the assembly instruction *AX 1 2 3* would add the contents of register one to the data at the address contained in register two and store the result in register three.

The register to register instruction group operates similar to the indexed instruction group except that the second register specified contains data instead of an address. For example, *AR 1 2 3* would add the contents of register one to the contents of register two and place the result in register three.

It should be noted that any combination of registers is allowed in these instructions. For example, the contents of register two could be shifted left one bit by executing the instruction *AR 2 2 2*. Table 1 lists a summary of the instruction set.

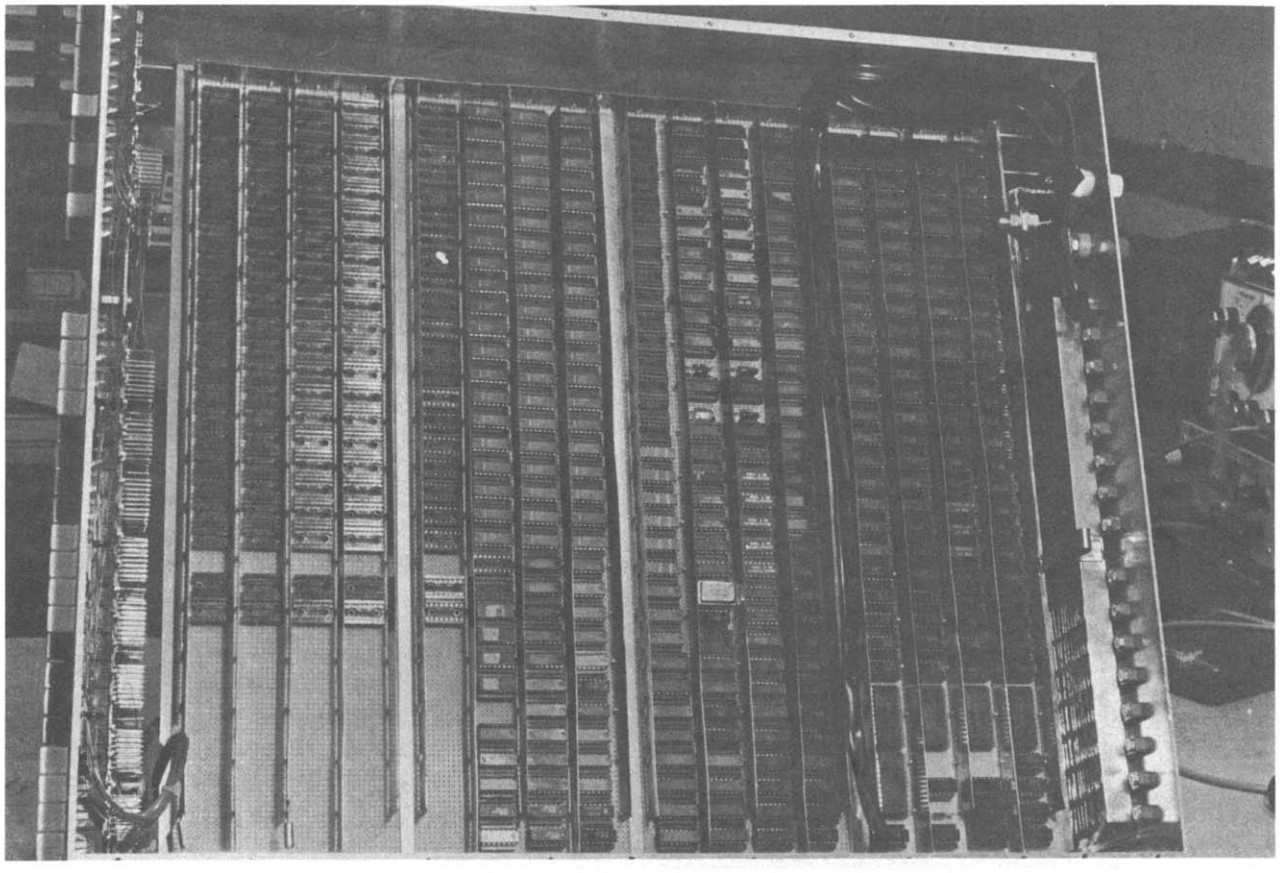

Fig. 4. OU-8000 bottom view.

One specialized instruction is included in the instruction set to perform the increment memory function. This instruction requires three memory cycles, (instruction fetch, data fetch, and data store), and is performed in 1 μs. This instruction greatly reduces the processing time when performing the MCA (multichannel analyzer) function.

5. Speed

Using 250 ns memory, all instruction times in the system are 1 μs. If higher speeds are required, faster memory will result in total instruction times less than 1 μs. The 1 μs instruction execution time allows data acquisition at over 200 kHz rates when the system is running as a multichannel analyzer.

6. Application of the OU-8000

The data acquisition system designed around the OU-8000 is proving to be a versatile, high speed and reliable system. Since construction of the OU-8000, we have interfaced it with a keyboard, CRT display scope, joystick, four high speed, multiplexed ADCs (analog-to-digital converters) and an IBM 1800 computer. In operation, the OU-8000 operates independently of the IBM 1800 system, using the 1800 as a I/O (input/output) device. This interface to the 1800 allows the minicomputer to access the 1800 I/O devices including: direct access disk drives, typewriters, magnetic tape drives, storage CRT displays, etc.

Software support for the OU-8000 now includes: an assembler, written in FORTRAN which allows assembly of minicomputer programs on any computer which supports FORTRAN; a loader, which loads the object code output from the assembler into the OU-8000 memory; an extensive subroutine library to support the I/O devices, code conversion, etc; an on-line editor to facilitate on-line debugging of programs; and at present three operating systems with a fourth in the development stage.

The first operating system was designed to ac-

TABLE 1
Summary of instruction set.

		Direct memory reference instructions
A	α	OPERND MEMORY REFERENCED ADD
IM		OPERND INCREMENT MEMORY
L	α	OEPRND LOAD REGISTER
N	α	OPERND MEMORY REFERENCED LOGICAL 'AND'
O	α	OPERND MEMORY REFERENCED LOGICAL 'OR'
S	α	OPERND MEMORY REFERENCED SUBTRACT
ST	α	OPERND STORE REGISTER
X	α	OPERND MEMORY REFERENCED LOGICAL 'EXCLUSIVE OR'
		Indexed instructions
AX	$\alpha\beta\gamma$	INDEXED ADD
BX	β	UNCONDITIONAL INDEXED BRANCH
CLX	$\alpha\beta$	INDEXED SUBROUTINE CALL
IMX	β	INDEXED INCREMENT MEMORY
LX	$\alpha\beta$	INDEXED LOAD REGISTER
NX	$\alpha\beta\gamma$	INDEXED LOGICAL 'AND'
OX	$\alpha\beta\gamma$	INDEXED LOGICAL 'OR'
STX	$\alpha\beta$	INDEXED STORE REGISTER
SX	$\alpha\beta\gamma$	INDEXED SUBTRACT
XX	$\alpha\beta\gamma$	INDEXED LOGICAL 'EXCLUSIVE OR'
		Register to register instructions
AR	$\alpha\beta\gamma$	REGISTER TO REGISTER ADD
CR	$\alpha\gamma$	COMPLEMENT REGISTER
DR	$\alpha\gamma$	DECREMENT REGISTER
IR	$\alpha\gamma$	INCREMENT REGISTER
LR	$\alpha\gamma$	TRANSFER REGISTER
NR	$\alpha\beta\gamma$	LOGICAL 'AND' REGISTERS
OR	$\alpha\beta\gamma$	LOGICAL 'OR' REGISTERS
SM1	γ	SET REGISTER TO MINUS ONE
SR	$\alpha\beta\gamma$	SUBRACT REGISTERS
SZ	γ	SET REGISTER TO ZERO
XR	$\alpha\beta\gamma$	LOGICAL 'EXCLUSIVE OR' REGISTERS
		Branch instructions
B		OPERND UNCONDITIONAL BRANCH
BN	α	OPERND BRANCH IF REGISTER NEGATIVE
BNZ	α	OPERND BRANCH IF REGISTER NOT ZERO
BP	α	OPERND BRANCH IF REGISTER POSITIVE OR ZERO
BZ	α	OPERND BRANCH IF REGISTER ZERO
		Input output instructions
R	γ	OPERND READ INPUT
W	α	OPERND WRITE OUTPUT
		Shift instructions
SL2	$\alpha\gamma$	SHIFT LEFT TWO
SL6	$\alpha\gamma$	SHIFT LEFT SIX
SL12	$\alpha\gamma$	SHIFT LEFT TWELVE
SR1	$\alpha\gamma$	SHIFT RIGHT ONE
SR2	$\alpha\gamma$	SHIFT RIGHT TWO
SR6	$\alpha\gamma$	SHIFT RIGHT SIX
SR12	$\alpha\gamma$	SHIFT RIGHT TWELVE
		Interrupt control instructions
CI		RETURN TO BACKGROUND PROCESSING
M		MASK INTERRUPTS
U		UNMASK INTERRUPTS
		Miscellaneous instructions
CL	α	OPERND CALL SUBROUTINE
LI	α	OPERND LOAD OPERAND
WT	α	WAIT

$\alpha\beta\gamma$ may be registers 0, 1, 2 or 3.

SHIFT LEFT ONE is implemented by AR $\alpha\alpha\alpha$.

quire, store and display data from a single ADC. Commands were entered from the keyboard to the OU-8000 to control the data acquisition, storage and display of data acquired. A modification to this system allowed two parameter data arrays to be generated. Fig. 5 shows one such two parameter display.

The second operating system was designed to support a position sensitive detector system used in conjunction with a magnet spectrometer. This operating system had three input ADCs. The position sensitive detector had two linear outputs, and the third ADC contained time of flight (TOF) information. From these three inputs the OU-8000 was used to detect a coincidence with the three input signals, store each input in a separate array, generate a sum of the two position sensitive detector events and store it in an array; then it divided the sum input by one of the position sensitive detector inputs and stored this in another array. Using the minicomputer allowed the addition and division process to be done digitally instead of using linear electronics. This system also had the standard data storage and display features.

The third operating system was designed to be

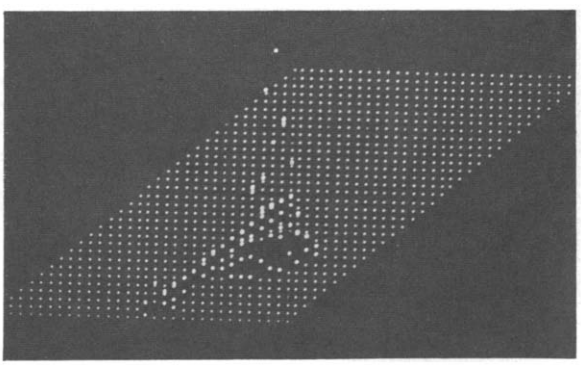

Fig. 5. Two-parameter display.

a general purpose multichannel analyzer system. The system contains a repertoire of 43 commands used to defined arrays, enable ADC's, control display of data, define I/O parameters, and control the data acquisition process. The OU-8000 sets up data areas for each defined ADC. Data areas are defined using the define command and a "MENU" is displayed on the CRT. The user then uses the joystick to intensify the parameter he wishes to define and then types in the desired value. The "MENU" allows the user to define; the ADC to use as input, whether the ADC is enabled or not, the logical name to be assigned to the array, the number of channels to reserve for the array, the logical lower and upper level discriminator channel, the digital offset and the conversion gain to be assigned to the ADC. Fig. 6 shows one such "MENU".

Data are displayed on the CRT as data acquisition is taking place. Since the ADC inputs operate on the interrupt level they take precedence over the display program and no dead time is introduced by the display program. Data arrays may be referenced by ADC # or logical array name. All ar-

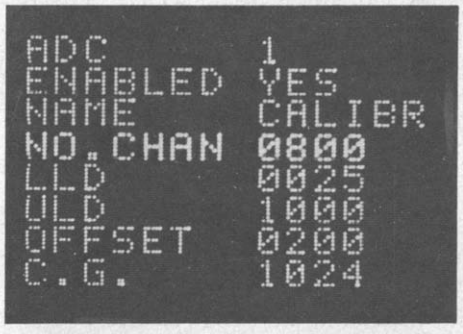

Fig. 6. Define array menu.

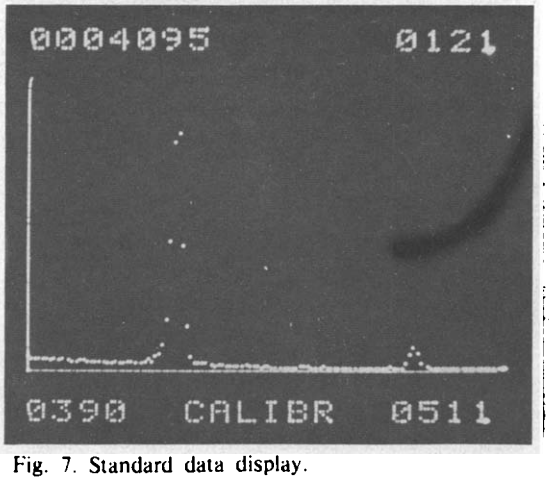

Fig. 7. Standard data display.

```
**********************************************************************
**                                                                  **
**      SUBROUTINE BNASCI                                           **
**                                                                  **
**********************************************************************
**                                                                  **
**       SUBROUTINE TO CONVERT A 20 BIT BINARY NUMBER INTO          **
**      A DECIMAL ASCII CHARACTER STRING.                           **
**                                                                  **
**      CALLING SEQUENCE              CL    3    BNASCI           **
**                                    DCA        NUMBR            **
**                                    DCA        NCHARS           **
**                                    DCA        LIST             **
**                        NUMBR       DC         00000            **
**                        NCHARS      DC         1-7              **
**                        LIST        BSS        1-7              **
**                                                                  **
**********************************************************************
**                                                                  **
**      D. E. CARTER    3/13/77                                     **
**                                                                  **
**********************************************************************
**                                                                  **
 24  100  1812C   1  BNASCI  ST   0       SAV0
 25  101  1A12D   1           ST   1       SAV1
 26  102  1C12E   1           ST   2       SAV2
 27  103  11000   0           LX   0 3
 28  104  10000   0           LX   0 0
 29  105  CE600   0           IR   3   3
 30  106  13000   0           LX   1 3
 31  107  12000   0           LX   1 1
 32  108  F4007   0           LI   2       #7
 33  109  CCA18   0           SR   2 1 1
 34  10A  F4131   0           LI   2       TABLE
 35  10B  CCC25   0           AR   2 1 2
 36  10C  CE600   0           IR   3   3
 37  10D  13000   0           LX   1 3
 38  10E  CE600   0           IR   3   3
 39  10F  1E12F   1           ST   3       RET
 40  110  F6131   0           LI   3       TABLE
 41  111  CF618   0           SR   3 2 3
 42  112  B611B   1           BNZ  3       LOOP
 43  113  C860E   0           SZ           3
 44  114  98120   1           BP   0       NOMILS
 45  115  41000   0           SX   0 2 0
 46  116  CE600   0           IR   3   3
 47  117  98120   1           BP   0       NOMILS
 48  118  31000   0           AX   0 2 0
 49  119  CE60E   0           SZ   3   3
 50  11A  B8120   1           B            NOMILS
 51  11B  C8600   0  LOOP     SM1      3
 52  11C  41000   0  LOOP1    SX   0 2 0
 53  11D  CE600   0           IR   3   3

 54  11E  9811C   1           BP   0       LOOP1
 55  11F  31000   0           AX   0 2 0
 56  120  4E130   1  NOMILS   O    3       #30
 57  121  26800   0           STX  3 1
 58  122  CA200   0           IR   1   1
 59  123  CC400   0           IR   2   2
 60  124  F6137   0           LI   3       TABLND
 61  125  CF618   0           SR   3 2 3
 62  126  9E11B   1           BP   3       LOOP
 63  127  0812C   1           L    0       SAV0
 64  128  0812D   1           L    1       SAV1
 65  129  0C12E   1           L    2       SAV2
 66  12A  0E12F   1           L    3       RET
 67  12B  C1000   0           BX     3
 68  12C  00000   0  SAV0     DC           00000
 69  12D  00000   0  SAV1     DC           00000
 70  12E  00000   0  SAV2     DC           00000
 71  12F  00000   0  RET      DC           00000
 72  130  00000   0  #7       EQU          00007
 73  130  00030   0  #30      DC           00030
 74  131  F4240   0  TABLE    DC           F4240
 75  132  186A0   0           DC           186A0
 76  133  02710   0           DC           02710
 77  134  003E8   0           DC           003E8
 78  135  00064   0           DC           00064
 79  136  0000A   0           DC           0000A
 80  137  00001   0  TABLND   DC           00001
 81  138  00000   0           END


  SYMBOL TABLE

          SYMBOL          DEFINED

          BNASCI            100
          LOOP              11B
          LOOP1             11C
          NOMILS            120
          SAV0              12C
          SAV1              12D
          SAV2              12E
          RET               12F
          #7                007
          #30               130
          TABLE             131
          TABLND            137


   0 ERRORS IN ABOVE ASSEMBLY
..END

PROGRAM
```

Fig. 8. Sample subroutine listing.

rays may be automatically scanned or each array may be displayed on command. The joystick allows the user to expand any region of interest in the displayed array. A sample array is shown in fig. 7. Arrays may also be overlayed to show relative peak positions.

The data acquisition may be controlled from the keyboard or from external inputs. Data may be stored on the 1800 disk for later reference.

A fourth operating system is currently being designed to control a triplet–quadrupole spectrometer to be used for neutron induced charged particle experiments. The system will control the magnet currents and sample position and control the data acquisition.

As can be seen, the applications using the minicomputer are many and varied. The advantage of using the minicomputer controlled data acquisition system over our hardwired analyzer system is that the computerized system may be programmed to handle specialized data acquisition and control tasks.

Using the assembler, subroutine library, and unique system architecture, programming is straightforward and efficient. Fig. 8 shows a standard subroutine listing.

I would like to express my sincere thanks to Dr. J. Cox for his help and guidance throughout this project.

The photographs were made by D. Daso.

References

[1]) *Signetics data book*, Signetics Corporation, Menlo Park, California, U.S.A.

[2]) *INTEL data catalogue*, INTEL Corporation, Santa Clara, California, U.S.A.

[3]) Harris Semiconductor, Melbourne, Florida, U.S.A.