MASTER

OAK RIDGE NATIONAL LABORATORY
operated by
UNION CARBIDE CORPORATION • NUCLEAR DIVISION
for the
U.S. ATOMIC ENERGY COMMISSION

ORNL-TM-3683

DATE - February 4, 1972

SETUP AND CONTROL ROUTINES FOR A PULSED NEUTRON TIME-OF-FLIGHT SPECTROMETER

L. W. Gilley

BLANK PAGE

TABLE OF CONTENTS

SECTION 1

1.1 Introduction

A magnetically pulsed[1] neutron time-of-flight spectrometer[2,3] has been installed at the HFIR by the Solid State Division with the setup and control, collection of data, and data analysis all implemented with a PDP-15/30 computer programmed by the Mathematics Division. This report describes the programming required for setup and control of the experiment. The hardware is described to the extent necessary to understand the programming. The data collection and analysis routines were written by Sandra Merriman of the Mathematics Division and will be published at a later date.

This report is divided into three sections. The second presents a general description of the hardware and software such that this section alone probably covers the interest of persons not actually participating in running experiments. The third section describes some routines in greater detail and gives examples of experiment setup, etc.

The spectrometer mechanical hardware was designed and installed by the General Engineering Division, and the electronic interface hardware was designed and installed by the Instrumentation and Controls Division.[4]

SECTION 2

2.1 Hardware

2.11 Spectrometer

The spectrometer hardware, except for the electronics, is diagramed in Fig. 1. It consists of a pulsing crystal, experimental sample mount, three neutron detector mounts, which support a total of up to 16 detectors, and a mount which supports the three detector mounts, sample mount, and pulsing crystal mount as a unit. All mounts can be positioned at various angles about an axis with computer controlled pulsing motors. The three detector mounts and the sample mount move about one axis, and the pulsing crystal and mount supporting the sample and detector mounts (and their motors) as a unit rotate about another axis. The detectors and pulsing crystal are enclosed in appropriate shielding, allowing a collimated beam of neutrons to fall on the pulsing crystal.

2.12 Positioning System

Each mount is rotated about its axis with a pulsing motor which requires 200 pulses for each rotation of its shaft. Each motor can be pulsed forward or backward under computer control with a specific IOT instruction. All motor drives are geared such that 200 pulses produce one degree of rotation of the respective mount about its axis. The names of the motor drives and their functions are listed in Table 1.

ORNL-DWG 69-13340

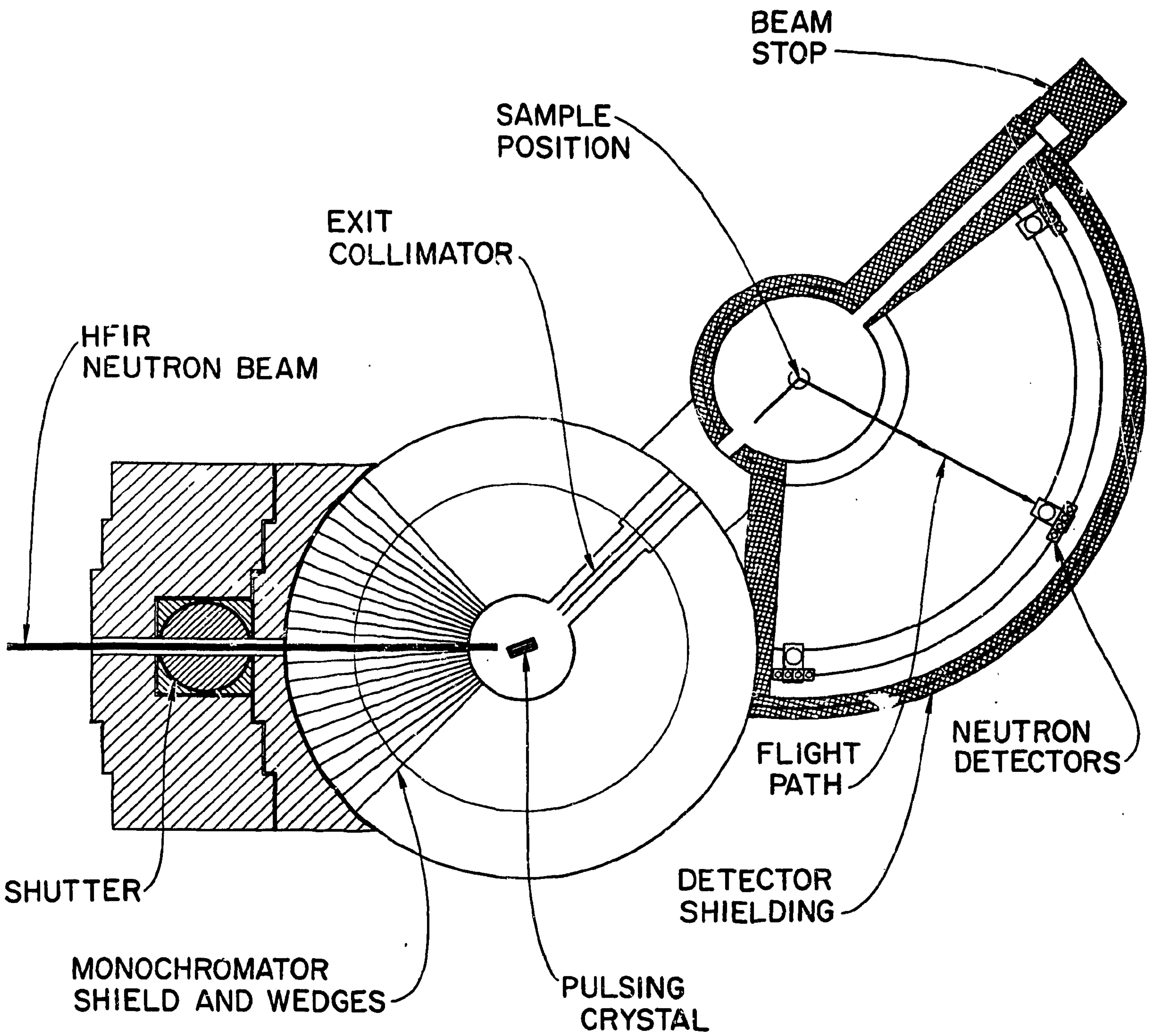

Figure 1

Table I

Motor Name	Function
2 THETA M	Rotates spectrometer with respect to neutron beam
OMEGA	Spare (not installed)
PSI	Rotates the sample mount
PHI 1	Rotates (moves) detector mount nearest exit neutron beam
PHI 2	Rotates (moves) detector mount next nearest exit neutron beam
PHI 3	Rotates (moves) detector mount farthest from exit neutron beam

The position of each mount is indicated by a scale graduated in degrees and fractions of a degree, and a dial mounted on the shaft of each motor graduated in 0.005 of a degree (one pulse). The 2 THETA M and PSI scales are graduated in 0.5 degrees while the scale for the three PHI drives is graduated in 0.1 degrees. During an experiment, the PHI motor dials and scale are enclosed in a shield of about 2 in. of B_4C on bottom, top, and sides, and therefore, cannot be read by the experimenter. Both scales and dials for 2 THETA M and PSI are exposed at all times. The heavy shield must move with the rest of the apparatus when 2 THETA M is driven. Therefore it is supported with a steel framework having three air pressure plates, about 18 x 24 inches, mounted on the bottom of the framework, so that, when 2 THETA M is being driven, air pressure supports the weight of the shield.

Stepping motors may fail to rotate for various reasons. Therefore as a means to provide for check on the rotation of the stepping motors, a disk having a narrow slit is mounted on each shaft such that once each revolution a signal from a photodiode in the path of the slit is sent to the computer indicating the shaft has rotated 1 degree (200 steps) since the last signal. The signal causes an API interrupt at level 2 and is designated the Degree Marker Interrupt.

Pulsing motors also must be pulsed at a rate determined by the load. If pulsed too fast or too slow, the motor will fail to rotate some pulses or fail to move at all. Optimum rate must be determined experimentally.

2.13 Interrupt Register

The computer is interfaced to an Interrupt Register[4] which can accept up to 16 input signals which set a flag, any one of which in turn will produce an interrupt flag producing a level 2 API interrupt at trap address 77. Table II shows the correspondence between the Interrupt Register input connector numbers, accumulator bits, and a given signal. The degree marker interrupts correspond to the designated motor. The hardware includes a motor clock which, when turned on by the computer, interrupts 300 times a second. These interrupts provide a uniform timing indication to the program so that the program can issue IOT's to drive the motors. Also included in the hardware is a prescaler, designated the Neutron Monitor in Table 1, which can be set up under computer control to give an interrupt after a preset number of counts and thereby determine the duration of an experiment as a function of the incident neutron beam intensity.

2.14 PPG and TOF Interface

Hardware[4] is provided which will collect data in two modes, normal and scan. In normal mode a Programmed Pulse Generator (PPG) generates a

Table II

INTERRUPT REGISTER INPUT	ACCUMULATOR BIT POSITION	INTERRUPT
	0	NOT USED
	1	NOT USED
16	2	SPARE
15	3	SPARE
14	4	SPARE
13	5	SPARE
12	6	SPARE
11	7	SPARE
10	8	SPARE
9	9	SPARE
8	10	PHI 3 DEGREE MARKER
7	11	PHI 2 DEGREE MARKER
6	12	PHI 1 DEGREE MARKER
5	13	PSI DEGREE MARKER
4	14	OMEGA DEGREE MARKER
3	15	2 φ M DEGREE MARKER
2	16	CLOCK INTERRUPT
1	17	NEUTRON MONITOR

pseudorandom sequence of signals and transmits this code to a magnetic pulser to chop the neutron beam, and the uncorrelated neutron time-of-flight data is collected through the TOF scaler. A pulse from a given detector causes the contents (12 bits of time information) of the scaler to be set into one of the 16 buffer registers containing 4 bits of detector identification. This information is subsequently transferred into the computer. This report is not concerned with normal mode data collection, and it will not be discussed further.

Scan mode is essentially field survey type data collection involving only one detector. In this mode a detector may move to a series of positions collecting data for a fixed time or number of monitor counts at each position, while the sample etc., do not move. Alternately the sample or pulsing crystal may be moved with the detector fixed. In this mode a pulse from a given detector does not transfer any time information from the TOF register to the corresponding detector buffer. Rather only the detector identification (4 bits of information) is transferred to the computer via a 3 cycle data channel.

2.2 Software

2.21 Introduction

Experiment setup and initiating data taking are implemented by typing in two character commands on a teletype interfaced to the computer. A routine, KBSRM, contains a keyboard decoder, which transfers control either to a routine in KBSRM or to another subroutine which implements the command. A list of the commands with defining comments is given in Table III. The routine MAINM does necessary preliminary initializing for other routines. Teletype I/O is handled through three routines: .FIRD (Floating or Integer Read), .FWRIT (Floating Write), and CIBCD (Convert Integer to Binary Code). Motors are driven by the routine MDRIVE and associated routines while data are collected by the routine SCAN and associated routines. In addition to the above routines, all of which are in core at the same time, a degree marker test routine, DMTEST, can be loaded separately and used to adjust the position at which a degree marker interrupt occurs. All of these routines are discussed further below.

All the routines are loaded from DECtape and executed with the PDP-15 ADVANCED MONITOR[5] in core. This permits use of the Linking Loader[6] which loads routines relocatably. Index mode addressing[7] is always used. The system monitor also provides system macro's such as .READ, .WRITE, etc., which are used frequently. The MONITOR provides a number of commands which are used principally in editing, printing out system information, etc. The command LOAD brings into core the Linking Loader which will then load the users binary programs from DECtape in response to typing in a list of routines to be loaded. After the routines have been loaded, the command CTRL S transfers control to the first location of the main routine. The main routine is defined as the first name typed in response to the Linking Loader request for routine names to be loaded.

Subroutines can be stored on DECtape either under their own file name individually, or they can be stored in the user's library, .LIBR5 BIN. When stored individually, each routine name must be typed in the list of names in response to the Linking Loader request for names.

Table III

COMMAND*	FUNCTION
IM	INPUT CURRENT 2 THETA M ANGLE
IO	INPUT CURRENT OMEGA ANGLE
IS	INPUT CURRENT PSI ANGLE
IP	INPUT CURRENT PHI 1 ANGLE
IQ	INPUT CURRENT PHI 2 ANGLE
IR	INPUT CURRENT PHI 3 ANGLE
DM	DRIVE 2 THETA M TO DESIRED ANGLE
DO	DRIVE OMEGA TO DESIRED ANGLE
DS	DRIVE PSI TO DESIRED ANGLE
DP	DRIVE PHI 1 TO DESIRED ANGLE
DQ	DRIVE PHI 2 TO DESIRED ANGLE
DR	DRIVE PHI 3 TO DESIRED ANGLE
PA	PRINT ALL CURRENT MOTOR ANGLES
CS	ZERO ALL SCAN STEP SIZES
NS	INPUT NUMBER OF SCAN STEPS
SM	INPUT 2 THETA M STEP SIZE
SQ	INPUT OMEGA STEP SIZE
SI	INPUT PSI STEP SIZE
SP	INPUT PHI 1 STEP SIZE
SQ	INPUT PHI 2 STEP SIZE
SR	INPUT PHI 3 STEP SIZE
SW	SET SCAN TIME OR MONITOR OPTION SWITCH
DN	INPUT DETECTOR NUMBER FOR SCAN
RN	INITIATE SCAN EXPERIMENT
CT	OUTPUT MARK ROUTINE INFO FOR 2 THETA M
CO	OUTPUT MARK ROUTINE INFO FOR PSI
CP	OUTPUT MARK ROUTINE INFO FOR PSI
C1	OUTPUT MARK ROUTINE INFO FOR PHI 1
C2	OUTPUT MARK ROUTINE INFO FOR PHI 2
C3	OUTPUT MARK ROUTINE INFO FOR PHI 3

*As currently programmed each command must be preceded by CTRL S and terminated with a carriage return.

2.22 Main Routine

MAINM, which is the main routine for all routines except DMTEST, serves two purposes, initialization and providing the Linking Loader with .GLOBL definitions so that the loader will load all needed routines and .GLOBLS when the user types MAINM in response to the Linking Loader's request. These definitions consist of listing the .GLOBL names in a .GLOBL statement. If these .GLOBL definitions were not included, several routines not defined outside the library would have to be typed in each time.

The routines INDCSR (Initialize Data Channel Service Routine), INIRM (Initialize Power Fail Recovery), and INKBM (Initialize Keyboard Monitor) are all routines entered from an API or PI interrupt. All except INKBM must be initialized by storing their entry address in a pointer and storing a JMS* POINT, where POINT is the pointer address in the API entry cell. This is necessitated by the fact that the routines are not located on the page with the API entry cells. In order to issue commands on the keyboard with the present program organization, it is necessary to transfer program control from the main routine wait loop to the keyboard decoder, KBSRM. This is done by taking advantage of a system MONITOR feature whereby program control is transferred to the address located in cell 106 in response to typing CTRL S. The Monitor originally loads 106 with the user's starting address. Hence when the program is initially loaded, typing CTRL S transfers control to MAINM where initialization takes place. INKBM loads 106 with KBSRM+1; so that, thereafter typing CTRL S transfers control to the subroutine KBSRM, where the program sits in a loop in a .READ macro waiting for command input. INKBM prints out

READY FOR COMMANDS after initialization takes place. Most routines after execution return control to an entry point of MAINM so as to print out READY FOR COMMANDS; hence, CRTL S is essentially a part of the command code. In some instances CTRL S may transfer control from a wait loop in another routine to MAINM. There are no strong arguments favoring this organization over having MAINM transfer control directly to KBSRM, then returning to the keyboard from another routine using the system CTRL P feature which transfers control to the user's starting address. Utilizing the CTRL S feature as implemented is mostly a debugging convenience, allowing repeated transfer of control to KBSRM without repeated print out of READY FOR COMMANDS and reinitialization. Furthermore, when debugging using DDT[8], some errors are non-recoverable using the DDT restart command but are recoverable using CTRL S. Utilizing CTRL S for transfer to KBSRM will probably be eliminated in the future, and if it is, will require only minor coding changes.

2.23 Teletype I/O

DEC supplied routines[9] to input or output floating point or integer data are designed to process the formatted READ and WRITE statements in FORTRAN IV programs. As such they are more general and hence bulky than required for this application. Furthermore, they are designed to output complete logical records rather than one word at a time. This is a requirement for this application because in scan mode of data collection it is desirable to print out each data point as soon as it is collected, drive any motors desired, collect another data point, etc., printing out 8 data points per line. It is desirable to print each data point as collected because frequently it can be determined changes should be made after 2 or 3 points. Therefore, to save space and improve flexibility teletype I/O routines were written to input and output integer data and floating point data in F format.

Input is handled by .FIRD (Floating or Integer Read) which utilizes the system .READ Macro to input characters into a 32 word buffer. The calling sequence is:

```
JMS*        .FIRD
JMP         .+3
     (0 or 1)
   (users address)
```

Zero indicates floating point input, and 1 indicates integer input. Users address is the address (or label) of the first word of the array into which input is to be stored. The required .INIT and .CLOSE Macros are not contained in .FIRD but must be coded by the user, the purpose being to give the user greater control over formatting I/O. When control is transferred to .FIRD, execution proceeds to a wait loop in .READ and waits for input. As many words can be input in one line, i.e., preceding a carriage return, as can be typed within 78 characters. Words must be separated by at least one space (spaces are included in the 78 characters limit). If integer input has been specified and a decimal point is typed after the integer, the routine prints the error message INVALID INPUT. INPUT AGAIN. and then reinitializes for new input. The invalid input will be immediately before the user on the teletype, so the correct input can be typed in. If a decimal point is not input after a floating point number with no fraction, the number will be converted to floating point and stored correctly with no message print out. If a character other than a number, period, minus sign, or space, or if two periods or minus signs are input, the message INVALID INPUT. INPUT AGAIN will be printed out and the routine reinitialized for input. It should be pointed out that if there is an error in any word of a line of words such that the routine reinitializes, the entire line must be input again. Also all words in a single line of input must be either floating point or integer. The largest integer that can be input is 131071. Floating point numbers occupy two 18 bit words and have 6 decimal digit accuracy.

Teletype output is implemented through the routines CIBCD (Convert Integer to Binary Code) and .FWRIT (Floating Write). The calling sequence for CIBCD is:

```
JMS*            CIBCD
JMP              .+4
(character code)
(address of integer)
(field width in octal)
```

The character code is loaded into each word of the output buffer before conversion, and for integer output would be 40 (space). The address is the in-core address of the integer, usually specified with a label. Field width is the total number of characters to be printed per word. Present buffer size allows 12 characters. If the width specified is less than the number of converted characters in the buffer, the high order digits will be truncated. If the field width is greater than the number of converted characters, plus sign if present, the left most positions printed will be spaces. A minimum of 5 spaces are available for spacing output words. The routine checks the number in the specified address to see if it is a two's-complement negative or positive number, and converts accordingly without the programmer having to specify which it is. Each time a number is converted and stored in the output buffer, the specified field is automatically printed out. The routine therefore outputs one word per call; and if a number of words are to be printed, the programmer must include the calling sequence in an appropriate loop, including carriage return.

Floating point output is handled through .FWRIT and the calling sequence is:

```
JMS*    .FWRIT
JMP     .+4
        (FA)
        (W)
        (D)
```

FA is the address of the 1st word of the 2 word floating point number, W is the total field width to be printed, and D is the number of decimal places. In summary .FWRIT fixes the floating point number and converts it to a binary integer then calls CIBCD to convert and print it out. Then .FWRIT prints a period, the fraction portion is multiplied to one more decimal place than specified, rounded back to the specified decimal places, and CIBCD is called again to print this integer. The routines are executed sufficiently fast that there is no visible lack of mechanical smoothness of the teletype on printout. In the call to CIBCD the character with which the buffer is to be loaded is specified as zero (60), so that if the fraction is smaller than the number of digits specified, the appropriate number of zeroes will be printed between the decimal point and the converted integer. Since the buffer in CIBCD is used for both the integer portion output and the fraction output, a minimum of 5 spaces can precede the output number. Negative numbers are output with a minus sign, but positive numbers have no sign.

2.24 Keyboard Command Service Routine

The keyboard service routine for the motor drives, KBSRM, decodes two character commands* and transfers control, through a dispatch table, to appropriate service routines. Routines to input motor drive current angle positions and desired angle positions, as well as routines to output marker test information, are in KBSRM. A list of KBSRM commands and their functions have been given in Table III.

The current angle is input through the subroutine POSN, which sets up a call to SETUP, and which allows the user to input the desired angle position in floating point format to three decimal places with the teletype I/O routine. Control is then transferred to SETUP, which divides the floating point number into degrees and 200ths of a degree, and stores this data in designated addresses in the motor drive routine MDRIVE. The desired angle is input through a call to POSN also, then a call to MOTOR (Motor Drive Setup Routine). MOTOR also calls SETUP, making the call in POSN redundant, but does no harm and allows POSN to input both current and desired positions. MOTOR sets up the motor drive routine for pulsing by calculating and transferring to the appropriate motor drive routine, the desired degrees and 200ths, setting the drive flag, setting an overshoot indicator which causes a drive to overshoot the desired position by one degree when driving backward, then approach the desired

---

*In the present organization each command must be preceded by CTRL S and terminated with a carriage return.

position going forward always to minimize gear backlash, and enabling the motor clock and interrupt register. Control is then transferred to a wait loop allowing pulses from the motor clock to cause execution of the drive routine and hence the given motor. What has just been described allows driving a given motor to a desired position by command input at the teletype. MOTOR is also called by the SCAN routine which will be discussed below.

2.25 Motor Drive Routines

There are six motor drive routines (MDRIVE) although only five motors are presently installed. These routines are similar to PDP-8 routines[10] used by the Solid State Division to drive motors on somewhat similar spectrometers. The drive routines are set up, as stated, by MOTOR and are executed at a uniform pace in response to API interrupts from the motor clock. The Interrupt Register Service Routine (IRSRV) only passes every other pulse to the motor drives, cutting their pulse rate to 150 per sec. The Two THETA M drive routine further reduces its pulse rate, by ignoring pulses, to about 18.8 per sec. Each motor routine updates its 200ths position, and degree position if required, each motor pulse and terminates driving when these are equal to the desired degree and 200ths position, unless it is driving backward in which case the routine sets up to drive forward one degree to compensate for the 1 degree overshoot. Each routine has a slow start feature which causes the motor to accelerate the first few pulses up to its normal speed. The Two THETA M motor also decelerates the last few pulses. The acceleration and deceleration are for about four pulses but are subject to change as experience dictates. The first time the Two THETA M routine is executed, i.e., on the first pulse, for a given traverse, it turns on a solenoid allowing air pressure to the air plates and delays for about 8 seconds to allow pressure to build up before beginning actually driving the motor. If a scan mode experiment is in progress, the solenoid is not turned off until the end of the experiment. If a scan experiment is not in progress, the solenoid is turned off when the Two Theta M motor has completed its drive.

When driving backward, the PHI 1 drive routine checks each pulse if it has reached a lower position limit, currently 14°, and if so, terminates driving whether the desired position as specified by command input has been reached or not, to prevent driving into the apparatus. PHI 2 similarly checks each pulse, if it is at the separation limit set to prevent running into PHI 1 or PHI 3; and PHI 3 checks a limit between it and PHI 2 and an upper limit. When IRSRV determines that all motors have stopped driving, it compares the current motor position as stored in the drive routines for PHI 1, PHI 2, and PHI 3 with the desired positions; and if they are not equal, an error message is printed out giving the motor name incorrectly positioned and its actual position. The program does not terminate.

Each time a motor drives its degree marker slit past its photodiode, a degree marker flag is generated in the Interrupt Register corresponding to that motor, and a level 2 interrupt transfers control to IRSRV. IRSRV then decodes which motor caused the interrupt and transfers control to an appropriate calling sequence to transfer control to MARK (a degree marker check routine). When the motor clock issues a pulse, the drive routine will issue a corresponding IOT to the motor pulser and update the position recorded in the program. If the motor pulser should fail occasionally, the motor will not move for that pulse, and therefore, the program will run ahead of the degree marker. That is, the degree marker is set to interrupt between positions 0 and 1 on the dial, which corresponds to the value 1 in the 200ths variable of the drive routine driving forward, and 0 driving backward. If the program runs ahead, this variable will be greater than 1. Each time MARK is executed, it checks if the program agrees with the degree marker; and if not, corrects the program, i.e., changes the degrees and 200ths to agree with the actual position. Even with corrections being made, if pulses are being missed, when driving is completed the motor position will be in error by the number of pulses missed since the last degree marker interrupt. Actually, since the degree marker interrupt can be off by a pulse due to hardware limitations, a correction is not made unless the error is greater than 1 pulse. The MARK routine also aids in maintenance checking when trouble is suspected either in the motor pulser or cables. Each time the routine is executed, a flag is set so that if one knows a degree marker has been crossed, then the flag indicates that the photo-cell, cables, etc. are working. If pulses are missed, the number of pulses in error is stored before correction and this information, as well as the above flag can be printed out by a command from the teletype. A detailed example of driving motors and obtaining test information from MARK is given in Section 3.

2.26 Scan Mode Data Acquisition Routines

Data concerning neutron distribution about the sample as a function of the relative orientations of pulsing crystal, sample, and detector are required to set up for data acquisition in the normal mode. The scan mode data acquisition procedure consists of positioning a mount (detector, sample, etc.), accumulating neutron counts for a specified interval of time or monitor counts, printing the results on the teletype, repositioning mounts, etc., until the specified number of data points are obtained, then terminating the run and reinitializing for new command input. The SCAN routine is set up with commands in which the user specifies the detector number to be used, whether time or counts interval and the duration of the interval, and the angle the specified mount is to be moved between data points. The detector number ranges from 1 to 16 inclusive, and is input with command DN. If a detector number other than 1-16 is input, the experiment will not terminate, but no data will be collected. SCAN is set to take data for a time or monitor interval according to a flag input with command SW, where 0=TIME and 1=MONITOR. When the value for the switch is input, the routine automatically types out either TIME= or MONITOR= depending on the choice. Time should be typed in seconds and monitor interval as number of counts.

Data is input to the computer through a 3 cycle data channel setup and serviced by DCSR (Data Channel Service Routine). Two input buffers are used, and when a level 0 buffer-full interrupt occurs, DCSR switches buffers and debrakes immediately to mainstream, so data can be accepted by the new buffer within 25 micro seconds after the buffer-full interrupt. The data is input as 18 bit words containing the detector number, and therefore, DCSR must examine each word to see if it contains the detector number specified for the experiment, and if so, increment the count by one. The routine will handle the specified 10^5 counts sec.

As each data point is obtained, control is transferred to a routine OUTPUT which controls output for the experiment. OUTPUT unloads the buffer into which data were being inputted when the interval was completed. Furthermore, in the event the interrupt occurred before DCSR had completed unloading the alternate buffer OUTPUT will complete unloading that too. Each data point is printed out as it is obtained, and if a check shows more data points are desired, control is transferred to the entry point REPEAT in SCAN. OUTPUT prints 9 words per complete line. The leftmost word is the number of the leftmost data point on the line (1, 9, etc.), and the next 8 words are data. Printout is in I8 format providing a minimum of 2 spaces between words since an 18 bit word can contain no more than a 6 decimal digit word. The maximum number of data points that can be obtained per experiment with present buffer size is 256. If the last data point printed out is all that was requested, OUTPUT prints out SCAN COMPLETED and returns control to the wait loop for command input. A detailed example of data acquisition in scan mode is given in Section 3.

2.27 Degree Marker and Motor Test Routine

As stated, each pulsing motor has mounted on its shaft a disk with a thin slit in it through which a light beam can fall on a photodiode giving an interrupt signal. This interrupt needs to occur near midway between pulses so the routine servicing it will not affect other routines. Also it is desirable to drive motors at various speeds to test how well they drive for a given load. This test would be required not only initially but if a motor or load were changed. These adjustments and tests are made with the routine DMTEST.

This routine drives one motor at a time, and hence, has only one sequence of drive coding setup by a keyboard command which inserts the appropriate drive IOT for the desired motor and direction, forward or backward. Table IV lists the keyboard commands available and their function. A motor is driven a given number of pulses (rather than degrees) specified on command input. In addition to commands to drive motors forward or backward, there are commands to oscillate each motor a specified number of pulses and oscillations and a command to change the rate at which the motor is driven. The default rate is one half clock rate, i.e., 150 pulses/sec. The command RF allows inputting other rates, the number typed in minus 1 giving the number of motor clock pulses skipped between motor pulses.

Each time a motor is driven the specified number of pulses, the routine prints out the number of pulses called for on command input, the number of pulses executed before a degree marker interrupt occurred, and in addition when a degree marker interrupt occurs, the time interval between the last pulse executed and the degree marker interrupt. This last time interval is measured in units equal to about 3.2 microseconds, or the time to execute an ISZ and JMP instruction. This information can be used in setting pulse rate of a motor for a given load and for adjusting the degree marker slit to occur between pulses. A detailed example of using DMTEST is given in Section 3.

Table IV

DMTEST KEYBOARD COMMANDS

COMMAND	FUNCTION
TF	DRIVE TWO THETA FORWARD
TB	DRIVE TWO THETA BACKWARD
OF	DRIVE OMEGA FORWARD
OB	DRIVE OMEGA BACKWARD
PF	DRIVE PSI FORWARD
PB	DRIVE PSI BACKWARD
1F	DRIVE PHI 1 FORWARD
1B	DRIVE PHI 1 BACKWARD
2F	DRIVE PHI 2 FORWARD
2B	DRIVE PHI 2 BACKWARD
3F	DRIVE PHI 3 FORWARD
3B	DRIVE PHI 3 BACKWARD
TO	OSCILLATE TWO THETA
OO	OSCILLATE OMEGA
PO	OSCILLATE PSI
1O	OSCILLATE PHI 1
2O	OSCILLATE PHI 2
3O	OSCILLATE PHI 3
RF	INPUT RATE FACTOR

SECTION 3

3.1 Introduction

In this section specific examples are described utilizing the features of the motor drive routines, the MARK routine, the scan mode data acquisition routines, and the degree marker and motor load test routine, DMTEST. These descriptions are intended to supplement information in Section 2 for persons actually utilizing these routines for test or experiment setup. DMTEST and associated routines are loaded as one set of operating routines and MAINM, the motor drive, scan, etc., routines are loaded as a second set of operating routines. When loading DMTEST teletype handler TTA must be assigned DAT slot 3 and DTA2 to -4 and -5. For MAINM, etc., assignments are the same. To load DMTEST, etc., type ←DMTEST in response to the Linking Loader's request for program names and terminate with ALT MODE. To load the motor drives, etc., type ←MAINM and terminate with ALT MODE.

3.2 Operational Examples

3.21 Positioning Motors

When the motor drive routines are loaded into core, the current position (angle) of each motor must be input before it can be driven accurately because the loader zeroes all labeled cells having no source contents. Hence, in effect the default current motor location values are zero. If, under such circumstances, a motor was actually at 20° and a command input to drive it to 30°, it would drive to 50°.

An example of inputting the current 2 THETA M angle will be given, assuming MAINM has already printed

READY FOR COMMANDS

>

and is waiting for command input. The 2 THETA M scale gives the whole degree position and the motor dial gives the fraction, the finest marks on the dial representing 0.005 degrees. Having this information, the command IM (see Table 2, Section 2) should be typed in. In response to this the program types:

INPUT CURRENT 2 THETA M ANGLE

and waits for the angle to be typed in F format, e.g., 35.715. This value is then converted to degrees and 200ths and stored in appropriate 2 THETA M parameter locations. In the case of the PHI 1, 2, and 3 drives when the current angle is input, this angle is also stored in the desired angle parameter locations. If this were not done, and if all desired angles had not been input, when a motor was driven error messages would be output for any PHI drive not driven, since its current and desired locations would be different. After input of each current angle

READY FOR COMMANDS

>

will be typed out.

Assume we wish to drive 2 THETA M to 38.0. The command DM should be typed in. The program will type INPUT DESIRED TWO THETA M ANGLE and wait for input. If no fraction is involved, input can be typed 38. or 38 and the I/O routine will interpret it correctly. In response to typing in the desired angle, the motor drive routine will drive the motor to the desired angle.

3.22 MARK Routine Output

The MARK routine can provide information useful in maintenance testing. Assume the motor pulser is suspected of being erratic (missing pulses) or the degree mark detector is suspected of being faulty. Information can be obtained by positioning the drive just above (say .1 degree) the degree mark, then driving the motor forward across the degree marker slit. Available information can be printed out for 2 THETA M, say, with the command CT. The program will print out CHECK FLAG = followed by a 1 or 0. The 1 indicates that the routine MARK was executed, and hence that the interrupt signal was received, and 0 indicates no signal received. On the same line will be printed # PULSES IN ERROR = followed by a number from 0 to 200, which will be the number of pulses missed from the start till the degree marker was passed.

3.23 Scan Mode Data Acquisition

As an example of scan mode data acquisition, assume it is desired to take 20 data points of 5 seconds each from detector number 3 and the detector is to be moved 0.1 degree each step. All motors must be driven to their initial position for the scan as described above. Then the following steps will set up and initiate a scan experiment. If needed the command CS will zero all scan step sizes, which may remain from a previous scan. In response to the command NS the program types NUMBER OF STEPS = and waits for input, in this example 20.

Suppose detector 3 is located on PHI 1 motor mount, then command SP causes PHI 1 STEP SIZE = to be printed out, in response to which in this example .1 should be typed in. The detector number is specified by inputting command DN in response to which DETECTOR NO. FOR SCAN = is printed out. Here it is 3. The program responds to the SW command with SWITCH OPTION = in response to which 0 should be typed for time interval and 1 for monitor interval. We have assumed time. When we type in 0 and carriage return, the routine responds with TIME = in response to which we type in 5 for our example. The above is all that is needed to set up for a scan, assuming only one motor is to be driven. The command RN then will initiate the experiment by transferring control to the routine SCAN. The above commands can be input in any order, except for RN which must be last. Continuing the example, SCAN then sets up to input data (from all 16 counters) into the first of two 256 word buffers through the 3 cycle data facility and enables the computer clock. When the first buffer is full an API interrupt transfers control to DCSR which switches buffers, resets and reenables the data channel, then counts the number of detector 3 counts in the buffer just filled. This latter buffer switching and unloading process continues until a computer clock interrupt (after 5 seconds in this example) transfers control to TSR (Time Service Routine), which disables the experiment and transfers control to an output routine OUTPUT. OUTPUT prints out the value of the count just accumulated from detector 3. OUTPUT does not issue a carriage-return line-feed until 8 data points have been printed on one line. Since we are assuming this to be the first data point, control would be transferred to the entry point REPEAT in SCAN. SCAN then drives the motor 0.1 degree, for our example, then enables the data channel for input to the first buffer, resets the clock for 5 seconds and reenables it and waits for the second data point to be accumulated. In this example this process would continue for 20 data points. A complete line of output will consist of the sequence number of the leftmost data point of a line followed by 8 data points. In our example 3 lines would be output consisting of the number 1 followed by 8 data points for the 1st line, then the number 9 followed by 8 data points, then 17 followed by 4 data points. After the 20th point is printed out, OUTPUT prints out SCAN COMPLETED then transfer control to MAINM for new command input.

3.24 Degree Marker and Motor Test

The routine DMTEST can drive a motor forward, backward, or oscillate, and change the rate of drive, all in response to commands as specified in Table IV, Section 2. Input to DMTEST is through .FIRD which is called to read in integer values, and output if through CIBCD which outputs integer values. If an integer I is input using command RF, then I-1 clock pulses will be skipped between motor pulses. Determining proper rate for a motor is a matter of trying various rates and observing motor performance.

As stated in Section 2, when a motor is driven with DMTEST, at the completion of the drive, the routine prints out the number of pulses called for on command input, the number of pulses executed before a degree marker interrupt did occur, the time interval between the last pulse executed and the degree marker interrupt. These features can be utilized in the following way. To adjust the degree marker slit, a motor is driven anywhere from 200 pulses (1 degree) down to about 20 pulses through a degree marker. If a motor is driven, say, 200 pulses forward and then 200 backward, and if the degree marker interrupt occurred on the 95th pulse forward, it should occur on the 106th backwards. If it does not occur here, the slit may be too wide, or the disk loose, etc. Given that the interrupt occurs at the correct pulse, the slit must be adjusted to occur near midway between two pulses. Information on this time interval, i.e., the interval from the time of the motor pulse immediately preceeding the interrupt to the interrupt, is printed out each time the motor is driven through a degree marker, so that if the motor is driven forward then backward, this information can be used to adjust the slit near center. If the motor is being driven at reduced speed, the time to service motor clock interrupts will be subtracted from the interval measured by the routine, even though no pulse is executed. In general, therefore, the two time intervals can be made nearly (but never exactly) equal with a reasonable effort, but the two time intervals when added do not equal the time interval between pulses. The possibility of intermittent failure of the degree marker hardware can be tested with the oscillate commands.

References

1. D. D. Bates and H. A. Mook, High Energy Magnet Pulser, Instrumentation and Controls Division, Ann. Progr. Rept. Sept. 1, 1971, ORNL-4734.

2. H. A. Mook and M. K. Wilkenson, "A Magnetically Pulsed Neutron Beam for Time-of-Flight Measurements of Inelastic Scattering," Journal of Applied Physics, Vol. 39, No. 2 (Part I) 447-449, February 1, 1968.

3. H. A. Mook and M. K. Wilkenson, "A Magnetically Pulsed Time-of-Flight Spectrometer for Inelastic Neutron Scattering," reprinted from "Instrumentation for Neutron Inelastic Scattering Research," Iaea, Vienna, 1970.

4. F. W. Snodgrass and H. A. Mook, Computer Control System for an Autocorrelation Neutron Time-of-Flight Spectrometer, Instrumentation and Controls Division, Ann. Progr. Rept. Sept. 1, 1971, ORNL-4734.

5. ADVANCED MONITOR SOFTWARE SYSTEM FOR PDP-15/20/30/40, Programmer's Reference Manual, Digital Equipment Corporation, Maynard, Mass.

6. PDP-15 UTILITY MANUAL, Programmer's Reference Manual, Digital Equipment Corp., Maynard, Mass.

7. PDP-15 SYSTEMS REFERENCE MANUAL, Digital Equipment Corp., Maynard, Mass.

8. Dynamic Debugging Technique Utility Program, Digital Equipment Corp., Maynard, Mass.

9. PDP-15 FORTRAN IV Programmer's Reference Manual, Digital Equipment Corp., Maynard, Mass.

10. W. R. Busing, R. D. Ellison, H. A. Levy, S. P. King, R. T. Roseberry, The Oak Ridge Computer Controlled X-ray Diffractometer, ORNL-4143.