SLAC-111
UC-37
(EXPI)

SLAC SPIRAL READER

CONTROL SYSTEM REFERENCE MANUAL

MICHAEL J. C. HU
STANFORD LINEAR ACCELERATOR CENTER
STANFORD UNIVERSITY
Stanford, California 94305

PREPARED FOR THE U. S. ATOMIC ENERGY
COMMISSION UNDER CONTRACT NO. AT(04-3)-515

January 1970

Reproduced in the USA. Available from the Clearinghouse for Federal Scientific and Technical Information, Springfield, Virginia 22151.
Price: Full size copy $3.00; microfiche copy $. 65.

ACKNOWLEDGEMENTS

It is almost impossible to give each individual the amount of credit he deserves on a project of this size, but the contributions from the following people have made the spiral reader system a success at SLAC:

Physics

Brandt Kehoe
David Leith
James Loos
Fred Martin
Benjamin Shen

Programming

David Budenaers
Robert Good
Michael Hu

Electronics

Hanoch Brafman
Joseph Caponera
Michael Hu
Mark Jensen
Russell Matheson

Mechanical

Marty Anderson
Ronald Cochran
Charles Hoard

Also, the contribution from the staff of technicians is gratefully acknowledged. Finally, the author would like to thank the friends at the Lawrence Radiation Laboratory at Berkeley for their generous assistance and guidance throughout the development of our spiral reader. Furthermore, their participation in the designing and building of the stage and the periscope assembly is greatly appreciated.

TABLE OF CONTENTS

LIST OF FIGURES

SLAC SPIRAL READER CONTROL SYSTEM REFERENCE MANUAL

INTRODUCTION

The spiral reader is a semiautomatic film digitizing machine which is used to measure photographs of elementary nuclear particle interactions created in a hydrogen bubble chamber. The entire spiral reader system is a man-machine-computer system, where man plays the most important role. The communication between the man and the machine (spiral reader) is through the computer.[1,2,3,4] The function performed by each subsystem is described in the following paragraphs.

The spiral reader is a measuring machine whose design was based on the fact that the particle tracks associated with an event appear to radiate from a central point of origin called the vertex. Once centered on the vertex of an event, the spiral reader scans the film along a spiral path starting at the vertex. A pulse is generated each time the spiral reader scans over a track. The pulse then goes through a pulse height discriminating circuit for processing. If the pulse has sufficient height and sufficient width it will be passed. Also the radial and angular information for this corresponding pulse will be made available. Thus, any track or spot on the film that intersects the spiral scan is digitized into polar coordinates relative to the vertex center. In this manner, the spiral reader digitizes the entire picture independent of the number of tracks there are in the picture. The spiral reader can be thought of as a vertex-oriented measuring machine.

Before the digitized data can be used to compute the space reconstruction of an event, it is necessary to filter out the data pertaining to the unwanted tracks. To extract the desired data points a filtering program "POOH" is used. The filtering program utilizes the fact that the event tracks start from zero radius and are arcs of circles. The technique for determining the desired track points is by histogramming the region of small radius. Once a track is roughly determined in this manner, relatively simple curve fitting criteria can be used to extract the remaining length of the track. The histogramming technique is inadequate in cases of very short tracks. Consequently additional data called "crutch points" must be supplied to aid the filtering program in filtering short and/or confusing tracks.

The computer performs several functions to help man to speed up the entire event-measuring process. The type of functions that are performed by the computer are: controlling the measuring sequence, fiducial centering, vertex-centering, data acquisition, data display, data storing on magnetic tape, film advancing, keeping track of events measured and events rejected, and monitoring the state of the spiral reader for malfunctions. Furthermore, the computer communicates with the human operator through the teletype/keyboard and/or console buttons.

Last of all, the human operator makes the final adjustment before any measurement is made, or decides the event is not measureable and therefore rejects the event. In cases of mechanical and/or computer failure, he has to make the final decision as to what to do.

Although the computer has taken a lot of routine work out of the hands of the human operator, the fact still remains that there are still many details the operator has to know in order to make the man-machine-computer system an efficient system. It is the purpose of this report to present the details of the design and the operational aspects of the spiral reader computer control program—"SYSTEM GENIE," and to serve as a reference manual for those poeple who are operating, maintaining and using the spiral reader.

The report itself is divided into three parts. The first part touches the hardware design and describes the software design in detail. The second part describes the operational aspects of the spiral reader computer control program. The third part is the appendices which explains the instruction codes and status registers, then automatic priority interrupt and data channel assignments, and the calling sequence for some of the commonly used routines.

PART 1

SYSTEM GENIE'S DESIGN ASPECTS

I. System Design Specifications

i) Hardware System Design

A. Design philosophy

The hardware system that will be discussed in this report will only cover the computer and its interface electronics. The optical and the mechanical system for the spiral reader will be described in a separate report. Also the detailed description of the PDP-9 computer, which is given in the "PDP-9 User's Reference Manual" will not be presented here.

Since there is a computer completely devoted to the spiral reader, it is quite obvious that one should make the most use from this computer. All the control functions should be implemented with the minimum amount of electronic hardware, thus making the interface hardware simple to design and easy to maintain. The interface design should take advantage of some of the options that were purchased with the computer.

B. Design specifications

a) All electronically-controlled mechanical devices should have interlock protections.

b) All moving devices should have a manual control and a computer control mode.

c) All devices with similar control characteristics should have identical hardware, if possible.

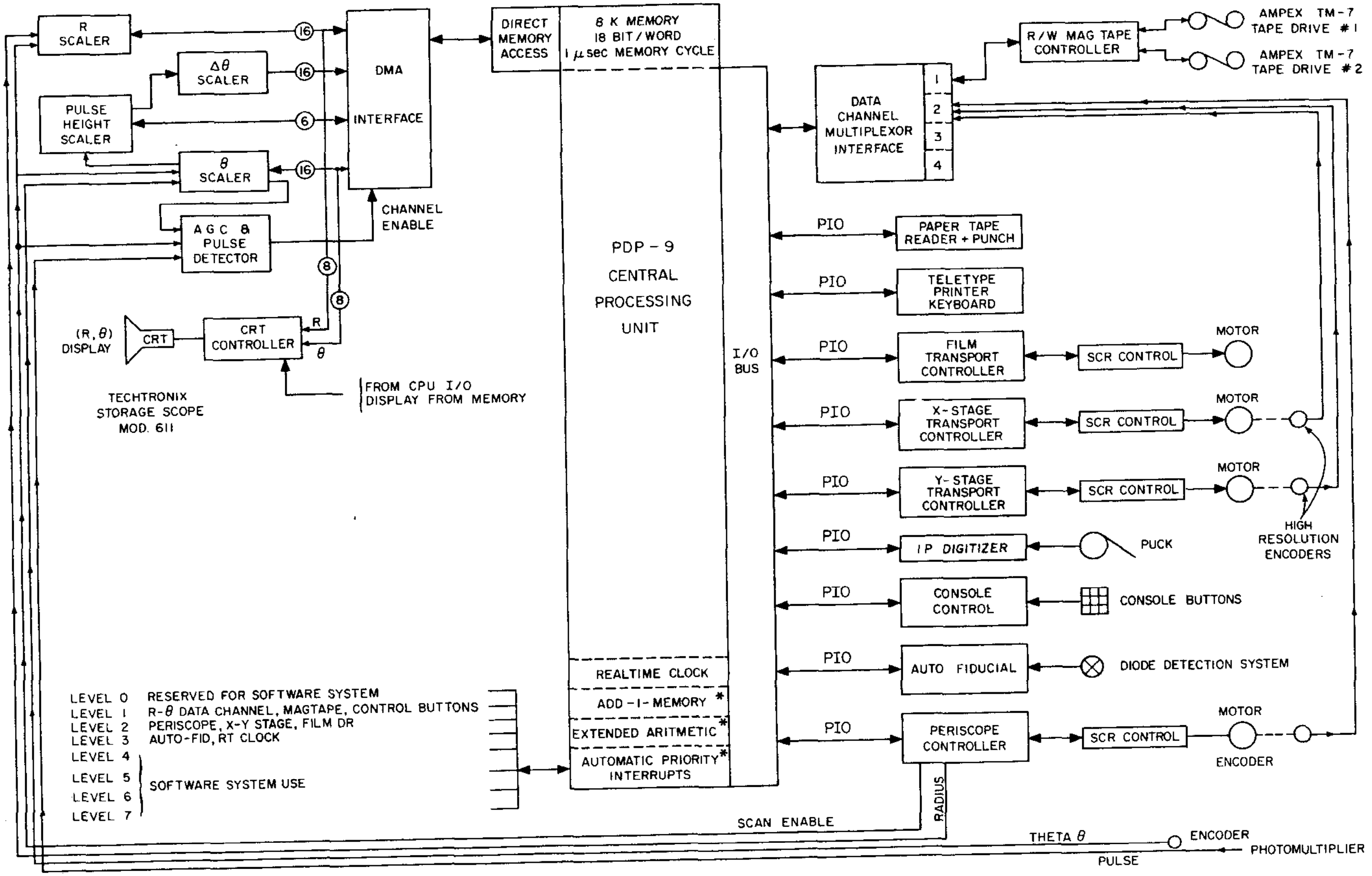
d) All I/O devices interfaced to the computer must have a status register that could be easily read and/or written by the computer.

e) All devices should take advantage of the automatic priority interrupt (API) feature of the PDP-9 computer.

f) Implement external scalers through use of the computer whenever possible.

C. System configuration

Figure 1-1 shows the complete computer system and its interface to the spiral reader. This hardware system configuration can be subdivided into three parts, namely: the computer and its peripheral equipment, the spiral reader control and



* ADDED OPTIONS

FIG. 1-1--Spiral Reader computer control system.

interface logic, and the data acquisition logic. An itemized list of the peripherals and interface logic for each part is given below:

a) Computer and its peripherals

   1) One PDP-9 computer with 8K of 18 bit word memory, and a memory cycle of 1 μsec.
   2) One real-time clock with 1/60 sec of resolution.
   3) One add-to-memory feature.
   4) One extended arithmetic element (EAE).
   5) Four levels of automatic priority interrupts, and four levels of software priority interrupts.
   6) One program interrupt level.
   7) One programmed input/output part for external devices.
   8) One data channel with 3 μsec/word input time, and a 4 μsec/word output time.
   9) One direct memory access (DMA) channel with a 1 μsec/word input/output time.
   10) One KSR 33 teletype printer keyword.
   11) Two Ampex TM-7, 9 channel, 800 bpi IBM compatible tape recorder. An inexpensive magtape controller that is capable of performing the following functions: read, write, backspace single record, generate and detect end of file, generate and check vertical and longitudinal parity, and rewind. The controller can only operate one tape unit at a time.

b) Spiral reader control and interface logic

1) The periscope, X-stage, Y-stage, and three identical film transport position controllers are quite similar in design. All six controllers have both manual and computer mode. The manual control of the periscope is through a potentiometer; the X-Y stage through an air-supported tracking ball; and the three separate film drives through a joystick and control buttons. The computer control for all six controllers is through 6-bit velocity registers (± 32 steps), 6-bit D-A converters and summing amplifiers. All six position controllers use PMI 400 motors driven by SLAC-developed SCR triggered circuits. Each control has its own interrupt with four possible interrupt conditions. The four interrupt conditions are reference, add-to-memory data overflow, complete, and limit-switch. The position scalers for all six

position controllers are in the computer memory. The encoders used are as follows: (a) The periscope uses a small Heidenhain circular encoder with 1000 lines per revolution logically doubled to give 64,000 over the 32 revolution total travel of the periscope. (b) The X-Y stage uses two Heidenhain linear encoders with 8μ line spacing. This spacing is reduced logically to 2μ in the present case, although the logic has the capability of 1μ least count. There are two diodes mounted on X-Y stage for reference information. (3) The film transport controller has the capability of using Gurley optical encoders with 2000 counts per revolution. However, presently the film positioning is accomplished through the use of diodes to detect brenner marks on the film.

2) One console control logic using 10 control buttons and an interrupt.

3) One image plane digitizer (Mangiaspago) using two Gurley optical encoders with 2000 counts per revolution, and 14 bit scalers for each encoder.

4) One auto-fiducial control logic which uses light sensitive diodes for sensing. Interrupts are generated whenever diodes detect the transitions from light to dark.

c) Data acquisition logic

The spiral reader film digitization is accomplished through the use of an automatic gain control (AGC) for equalizing film background variation, and a high speed pulse discrimination logic. Both the AGC and the pulse discrimination logic are designed and built by SLAC. At present AGC is available for both dark and bright field illumination, and each one is on a single logic card. The high speed pulse detection logic was designed to detect accurately pulses as narrow as 10μs, and the pulse height to six 6-bit accuracy. The pulse must exceed a certain threshold before it can be accepted, and likewise on the trailing edge of the pulse this same threshold must be exceeded from the peak down before the pulse is input to the computer. When this occurs four words are input to the computer: 1) Pulse height and threshold; 2) 16-bit angle count; 3) 16-bit radius count; and 4) 15-bit pulse width. These four words enter the computer memory via the Direct Memory Access Channel at 1μs per word or 4μsec per data point. Furthermore, two 6-bit D-A converters are used to drive one Tektronix 611 storage scope for digitized data display.

ii) Software System Design

A. Design philosophy

The type of computer control program for the spiral reader has to be a flexible one for a simple reason, because of the different film format requirements.

Different film formats might require different mechanical and/or electronic changes, which undoubtedly require some software changes in the control program as well. Perhaps the word "flexible" does not specify completely the nature of the control program in terms of programming. The control program should be designed in such a way that it is easy to add and/or delete part of the program without affecting the performance of the entire control program. Furthermore, the control program should have some kind of monitoring feature which can respond to the operator's requests, and direct subprograms to perform the desired tasks.

B. Design specification

a) All operators' requests should be requested either from the teletype printer/keyboard or the console buttons.

b) Since the response time of most of the electro-mechanical devices on the spiral reader are very slow compared to the computer, all I/O control routines should be interrupt driven.

c) The control program should be designed so that it can execute several jobs and be able to monitor the progress of these jobs simultaneously via interrupts.

d) Each I/O device should have its own interrupt handling routine.

e) All I/O device interrupt service routines should be kept as short as possible.

f) For those I/O devices which require lengthy computations, their interrupt service routines should request software priority interrupts to perform the computations. This way it will leave the automatic priority interrupt system free to respond to other interrupt requests.

g) Each interrupt service routine should have a means of returning to either the interrupted routine, or back to the main control program.

h) All I/O devices that have similar control characteristics should be programmed alike, if possible.

C. System configuration

The design of the spiral reader control program "SYSTEM GENIE" is based on both the hardware and software system design specifications. Basically, System Genie can be divided into three parts: the executives, the I/O control programs, and the spiral reader support programs. The executive part of System Genie deals with answering job requests from the operator and/or from I/O devices, queuing these jobs, and then executing them in their proper order. Furthermore, it sees that the interrupts from the I/O devices get properly serviced. The executive part of System Genie is designed to be self-sustaining. That is, this part of the control program can run independently of the rest of the program, hence adding and/or deleting I/O control programs could not affect the operation of the executives. Moreover, because of this self-sustaining feature, the operator can often recover from his own error through the teletype which is one part of the executive control program. Secondly, the I/O control programs are control programs for the PDP-9 peripherals and spiral reader controls. These programs can be added and/or deleted without affecting the main operation of the executive part of the control program. Lastly, the spiral reader support programs contain programs that deal with data formatting, event measurement sequencing, vertex centering, error recovery procedures, etc. The operation and performance of this part of System Genie depends heavily on the operation of the previous two parts, simply because lots of spiral reader support programs use I/O control programs. Similarly, the programs in this part can also be added or deleted without affecting the executive part of the control program. The details of each part will be described in the following sections. Unless otherwise stated the spiral reader computer control program — SYSTEM GENIE — will be referred to as "the system."

II. The Supervisory Routines for the System

i) The Executives

In the previous section it was stated that the main function of the executives of the systems were: answering job requests from the operator and/or from I/O devices, queuing these jobs, and then executing them in their proper order. In this section the function of each executive will be described. The following supervisory routines make up the executives of the system: job supervisor, job executor, teletype job supervisor, magtape job supervisor, software priority interrupt supervisor, program interrupt supervisor, and CAL supervisor.

A. The job supervisor

The purpose of having a job supervisor routine is to provide a means of centralizing job requests from several different sources, and storing these job requests in some orderly fashion. The function of this routine is to stack up job requests into a first-in first-out job buffer. The job buffer is circular and it can store up to a maximum of 16 jobs. To prevent external device interrupts from interrupting the job supervisor routine while it is trying to store one or more job requests into its buffer, the job supervisor is programmed to raise its priority to the highest priority level above all external hardware interrupts. This is a precautionary measure against any lost job requests. The job supervisor is also programmed to lower itself from the highest priority whenever it has finished storing one or more job requests.

a) Single job calling sequence:

```
JMS JOBSVR          NOTIFY JOB SUPERVISOR
LAW AA              ADDRESS OF JOB AA
DBK                 RELEASE JOBSVR FROM HIGHEST PRIORITY
```

b) Multiple job calling sequence:

```
JMS JOBSVR          NOTIFY JOB SUPERVISOR
LAW BB              ADDRESS OF JOB BB
LAW XYZ             ADDRESS OF JOB XYZ
LAW KK              ADDRESS OF JOB KK
DBK                 RELEASE JOBSVR FROM HIGHEST PRIORITY
```

B. The job executor

The purpose of this routine is to provide a means of executing jobs in the order that they are stored in the job supervisor's job buffer. Each job should always contain the JMP JOBXCT as its last instruction in the routine. The last instruction (JMP JOBXCT) will cause the computer to jump back to the job executor routine to excute the next job in the job buffer. As long as there is a job in the job buffer, the job executor is going to execute it. When the job executor finds that there is no more job in the job buffer, it then enters the system wait loop.

a) The proper calling sequence for the job executor should be:

```
JMS JOBSVR          NOTIFY JOB SUPERVISOR
LAW AA              ADDRESS OF JOB AA
DBK                 RELEASE JOBSVR FROM HIGHEST PRIORITY
JMP JOBXCT          GO EXECUTE THE NEXT JOB
```

The first three instructions request the job supervisor to store the job AA into its job buffer. The last instruction requests the job executor to execute the job AA. Suppose during the execution of job AA, the job supervisor is requested to store job BB into its job buffer. Upon completion of job AA, job BB is executed. This is because at the end of job AA the job executor is requested to execute the next job in the job buffer. Similarly at the completion of job BB, and also if there is no more job stored in the job buffer, the system goes into the wait state.

C. The teletype job supervisor

The functions of the teletype job supervisor are to keep track of the order in which the teletype requests are being requested by various jobs, and to make sure that a message is completely printed out before the next message can even be started. Furthermore, because of the way in which the teletype request buffer was designed, there is really no limitation as to how many teletype requests the teletype job supervisor can store.

a) The proper calling sequence for an alpha-numerical message typeout is:

```
JMS TTMH          ALERT TELETYPE MESSAGE HANDLER
LAW TTA           ADDRESS OF ALPHA-NUMERICAL ROUTINE
LAW XXM           ADDRESS OF MESSAGE
NOP
JMS TTRS          ALERT TELETYPE SUPERVISOR
JMP JOBXCT        GO EXECUTE THE NEXT JOB
```

The first instruction is to call the teletype message handler subroutine to store the next three instructions into its buffer. The second instruction indicates alphanumerical typeout is required. The third instruction indicates the address of the message, and the fourth is a dummy instruction which has no significance in this case. The fifth instruction requests the job supervisor to put the teletype job supervisor into its job buffer. The last instruction jumps to the job executor which starts the teletype typing.

b) The proper calling sequence for a numerical message typeout is:

```
JMS TTMH          ALERT TELETYPE MESSAGE HANDLER
LAW TTD           ADDRESS OF NUMERICAL ROUTINE
LAW XXN           ADDRESS OF NUMERICAL VALUE
NOP
JMS TTRS          ALERT TELETYPE SUPERVISOR
JMP JOBXCT        GO EXECUTE THE NEXT JOB
```

Everything is similar to the previous case except the second instruction indicates decimal typeout is required, and the third instruction indicates the address where the binary numerical value is located.

c) The proper calling sequence for an alpha-numerical message typeout which also expects a numerical value in return is:

```
JMS TTMH         ALERT TELETYPE MESSAGE HANDLER
LAW TTA          ADDRESS OF ALPHA-NUMERICAL ROUTINE
LAW XXM          ADDRESS OF MESSAGE
LAW XXR          ADDRESS OF RETURNED DATA
JMS TTRS         ALERT TELETYPE SUPERVISOR
LAW XX           ADDRESS OF NEXT JOB STEP
DAC KBDDR4       STORE IN KEYBOARD ROUTINE
JMP JOBXCT       GO EXECUTE THE NEXT JOB
```

Again the three instructions are similar to (a), and the fourth instruction instructs the teletype supervisor where to store the binary value after it has made the conversion from decimal to binary. Also the sixth instruction instructs the teletype job supervisor what to do next after it has finished storing the converted data away. This routine is frequently used whenever the system needs numerical data from the operator. A much more simplified version of the subroutine calling sequence will be given in the next section.

D. The magtape job supervisor

The function of the magtape job supervisor is quite similar to that of the teletype job supervisor. It also keeps track of the order in which the magtape requests are being requested by various jobs, and it makes sure that one magnetic tape transport completely finishes one job before the same magnetic tape transport or the second magnetic tape transport can start a new job.

a) A typical magtape calling sequence would look like:

```
JMS MTJH         ALERT MAGTAPE JOB HANDLER
XX               WORD COUNT
XX               BUFFER ADDRESS-1
LAW MTXX1        ADDRESS OF MAGTAPE FUNCTION
LAW XXNS         ADDRESS OF NEXT JOB STEP
JMS MTRS         ALERT MAGTAPE JOB SUPERVISOR
JMP JOBXCT       GO EXECUTE THE NEXT JOB
```

The first instruction calls the magtape job handler to store the next four instructions into its job buffer. The second instruction gives the buffer size, and the third instruction gives the starting address of the buffer. If functions like REWIND, generate EOF (end of file), require no buffer size of buffer address, then the NOP instruction should be used in place of XX. The fourth instruction indicates the type of magtape function to be performed. For instance, MTWR1 commands magnetic tape transport No. 1 to write, and MTWR2 commands magnetic tape transport No. 2 to write. A complete list of these commands is given in Appendix III. The fifth instruction instructs the magtape job supervisor where to go after it has completed the function. The sixth instruction requests the job supervisor to put the magtape job supervisor into its job buffer. Finally the instruction requests the job executor to execute the magtape job. A set of simplified magtape job subroutines will be given in the next section.

E. The software priority interrupt supervisor

The purpose of the software priority interrupt supervisor is to keep track of the software interrupt requests. These software interrupt requests are generated by some hardware interrupt service routines which require lengthy computations but could not afford the time to tie up the interrupt system. The software priority interrupt supervisor always assigns the highest available software priority interrupt level to the requesting hardware interrupt service routine. Usually the software interrupt gets serviced immediately after the hardware interrupt service routine has made the request. One major advantage of the software priority interrupt supervisor is that several hardware interrupt service routines could be requesting software priority interrupts, but only one of them gets serviced, and the rest of them are serviced automatically in order. No damage can be done to the program in which the interrupt occurred or to the job supervisor's job buffer when software priority interrupts are being serviced.

a) The calling sequence for the software interrupt supervisor is:

```
JMS SISVR          ALERT SOFTWARE INT. SUPERVISOR
LAW XX             ADDRESS OF SERVICE ROUTINE
  .   .
  .   .
```

The first instruction calls the software interrupt supervisor to store the address of the service routine into the highest available interrupt level. The address of the service routine is given in the second instruction.

F. The program interrupt supervisor

The function of the program interrupt supervisor is to determine whether the teletype or the keyboard made the interrupt. If it is the teletype that made the interrupt, then the program interrupt supervisor has to determine whether the

B. The keyboard

The keyboard provides a communication link from the operator to the computer. Keyboard command codes are: /, #, $, =. Each of these performs a specific task. For instance,

/: clears the keyboard buffer and takes the system out of any scaler display loop,

#: execute the subroutine,

$: convert numerical data from decimal to binary and store it away in some specified location in memory ,

=: printout the decimal value of any address location specified in octal.

Consult "System Genie's Operational Aspects" for further details as to how to use these commands.

C. The realtime clock

The realtime clock subroutines consist of routines that compute the time of day and incremental time difference

a) Print time of day

```
JMS TIMER          GO REQUEST TIME
JMP JOBXCT         GO EXECUTE THE NEXT JOB
```

b) Time computation and print

```
JMS TCR            GO COMPUTE TIME
JMS TPR            GO PRINT TIME
JMP JOBXCT         GO EXECUTE THE NEXT JOB
```

Before this set of instructions can be executed, the accumulator (AC) should contain the incremental time in seconds, and seconds must be in binary.

ii) Nonstandard PDP-9 Computer Peripherals

The nonstandard PDP-9 computer peripherals consist of two Ampex TM-7 magnetic tape transports and a controller. Since the function of the magtape job supervisor was described briefly in the previous section, only its name might be used in this section to explain subdevice interrupt service routines and the magtape subroutines.

A. The magtape

The magtape interrupt is assigned to hardware priority level once, and its interrupt address is location 45 in the computer memory. The subdevice interrupts consists of magtape complete interrupt, data overflow interrupt, parity interrupt, and end-of-tape interrupt. The magtape complete interrupt is generated whenever the magtape has finished reading a record, or writing a record. The magtape complete interrupt is used for directing the system where to go next after this interrupt has occurred. A data overflow interrupt is generated whenever the number of words per record read off the magtape exceeds the buffer size. When this interrupt occurs the magtape controller is prevented from reading any further, and a warning message is typed out on the teletype. The magtape parity interrupt is generated whenever a vertical parity or a longitudinal parity occurs while the magtape is doing a read. When a magtape parity occurs, a warning message is typed out on the teletype, and the magtape stops at the end of record. The magtape end-of-tape interrupt is generated whenever the end of the tape is reached. When this interrupt occurs, the magtape automatically stops, and a warning message is typed out on the teletype. The magtape subroutines consist of subroutine-like write one records, read N records, backspace N records, generate end-of-file, and rewind. N stands for the number of records. Since both magnetic tape transports share the same controller, their I/O commands and subroutines are identical with the exception that all the I/O commands and subroutines for magnetic tape transport #1 end with the letter 1, and for magnetic tape transport #2 end with the letter 2. Only magnetic tape transport #1's subroutine will be described below.

a) Magtape 1 write one record routine

```
JMS   WREC1      MAGTAPE1 WRITE 1 RECORD
LAW   -X         WORD COUNT = -X (Buffer size)
LAW   BUF-1      BUFFER ADDRESS -1
```

This set of instructions writes one record beginning from address called BUF, and of length X words long into the magtape 1. The word count must be expressed in negative numbers, and buffer address should always be expressed one less than the original address. These two requirements are really computer data channel requirements and have nothing to do with the subroutine itself.

b) Magtape 1 read N records routine

```
JMS   SFNR1      MAGTAPE1 READ N RECORDS
LAW   -(X + 1)   WORD COUNT = - (X + 1)
LAW   BUF-1      BUFFER ADDRESS -1
```

```
LAW    -N           -N= READ N TIMES
SAD    (046114)     SKIP IF NOT END OF FILE
 .       .
 .       .
```

This set of instructions reads N records of length X + 1 words into the same buffer located at the address labeled BUF. The word count is X + 1 instead of just X; this is because during a read command the magtape controller reads in the longitudinal check character, so consequently, the buffer has to be one location bigger. The number of records to be read must be expressed in negative number. This subroutine is really used for spacing forward N records, because the magnetic tape transport will not stop until the Nth record is reached, and the information in the buffer is changed after each read. In order to find out what is stored in each record, the number N must be set to 1, and the magnetic tape transport will stop after one record, so the buffer could be examined. While reading, the subroutine is also checking for end-of-file (EOF) after every record. If no EOF was encountered during read, the magnetic tape drive will stop at the end of N records, and before going on the accumulator (AC) is checked for possible EOF code (046114_8). However, if an EOF was encountered during a read, the subroutine will exit and leave the EOF code in the AC. Again, before going on the AC is checked for possible EOF code, and this time the EOF code exists and it is up to the programmer to decide what action he should take at this point.

c) Magtape 1 backspace N records

```
JMS    BSNR1        MAGTAPE1 BACKSPACE N RECORDS
LAW    -N           -N=N BACKSPACES
 .       .
 .       .
```

This set of instructions backspaces N records. The number of backspaces should be expressed in negative numbers. No parity can occur while the magtape transport is backspacing, because the magtape controller does not detect parity during that time.

d) Magtape 1 generates end-of-file

```
JMS    WEOF1        MAGTAPE1 WRITE END-OF-FILE
 .       .
 .       .
```

This instruction generates an end-of-file (EOF) code (046114), and writes it on the magtape.

e) Magtape 1 rewind

```
JMS     REW1            MAGTAPE1 REWIND
 .       .
 .       .
```

This instruction rewinds magtape. Information concerning magtape status register, instructions, and commands are given in Appendix III.

iii) Spiral Reader Controls

The spiral reader controls consist of the radius and angle high speed data channel, periscope, X-Y stage, film drive, console, image-plane-digitizer (IPD), and auto-fiducial control. The interrupt service routines for all spiral reader controls have one feature in common, that is, their hardware interrupt is always enabled at all times, but their subdevice interrupt service routines are normally disabled and are only enabled whenever their services are required. This feature could eliminate lots of undesired interrupts from various I/O devices due to noise. The subdevice interrupt service routines and control programs for these spiral reader I/O devices are presented below.

A. The radius-angle high speed data channel

The radius-angle high speed data channel interrupt is assigned to hardware priority level one, and its interrupt address is location 100 in the computer memory. The subdevice interrupts consist of data rate too high interrupts, angle error interrupt, and complete interrupt. The data rate too high interrupt is generated whenever data coming into the memory are faster than the magtape controller can write them out into the magtape. When this situation occurs, data coming into the memory will be temporarily stopped, so as to give the magtape controller a chance to finish writing. Moreover, a warning message is typed out on the teletype. The angle error interrupt is generated whenever two or more angle counts are lost. When angle error occurs, a warning message is typed out on the teletype. The complete interrupt is generated whenever data from the radius-angle high speed data channel have filled up the buffer of a specified size in the computer memory. This interrupt is used primarily for switching buffers around, and the next paragraph explains how this is done.

The radius and angle high speed data channel routine is activated by the periscope control program. Once this routine is activated, digitized data enter the memory via the direct memory access at 4μsec per data point. A double buffer technique is used for saving memory space. The two buffers, each of size 512 words, are switched back and forth through interrupts. As soon as the first buffer is filled up, a complete interrupt is generated, and the second buffer is switched in. While data are entering the second buffer, the first buffer is being written onto the magtape. When the second buffer is full, a complete interrupt is generated and the first buffer is switched back in again. Meantime the data in the second buffer are being written onto the magtape. This process ends whenever a spiral scan is finished. On a normal scan, approximately 600 to 700 data points are recorded, and this would mean each buffer would have to be switched back and forth three to four times. Furthermore, after each buffer is written onto the magtape, the same buffer is automatically cleared with all ones.

B. The periscope

The periscope interrupt is assigned to priority level two, and its interrupt address is location 101 in the computer memory. The subdevice interrupts consists of reference interrupt, complete interrupt, limit interrupt, and add-to-memory data overflow interrupt. The reference interrupt is generated whenever the periscope position encoder makes one complete revolution. This interrupt is used for keeping track of the periscope's position throughout the spiral scan. The complete interrupt is generated whenever a zero velocity is loaded into the velocity register for more than 100 msec. This interrupt is used for indicating that the periscope is completely stopped and the system can proceed to the next operation. The limit interrupt is generated whenever the periscope has traveled forward or backward beyond its intended length of travel. This interrupt is used for providing warning to the operator. The add-to-memory data overflow interrupt is not used. While in the computer mode, the enabled interrupt service routines are the reference, the complete, and the limit; and the disabled interrupt service routine is the add-to-memory data overflow. While in manual mode, all the interrupt service routines are disabled except the reference.

The periscope scaler resides at location 200 inside the computer memory. The counts are incremented or decremented via the add-to-memory feature of the PDP-9 computer. The periscope produces 2000 counts per revolution, and there are 32 revolutions per spiral scan, so this would mean the periscope would produce 64,000 counts. Presently, fine positioning of the periscope is not required, so the periscope scaler is used mainly for keeping track that the periscope encoder is putting out 64,000 counts for every spiral scan. This is important because the radius scaler in the pulse discrimination logic uses the same encoder.

The periscope position control program uses references interrupts generated by the periscope encoder to control the position and the velocity of the periscope. During the entire linear travel of the periscope 33 reference interrupts are generated. The periscope always sits behind the optical zero, and when the first reference is generated, the periscope scaler in the memory is preset to (2000_8) counts and the radius and angle high-speed data channel are activated. The periscope velocity *vs* periscope position (in terms of reference counts) in both forward and reverse direction is shown in Fig. 1-2.

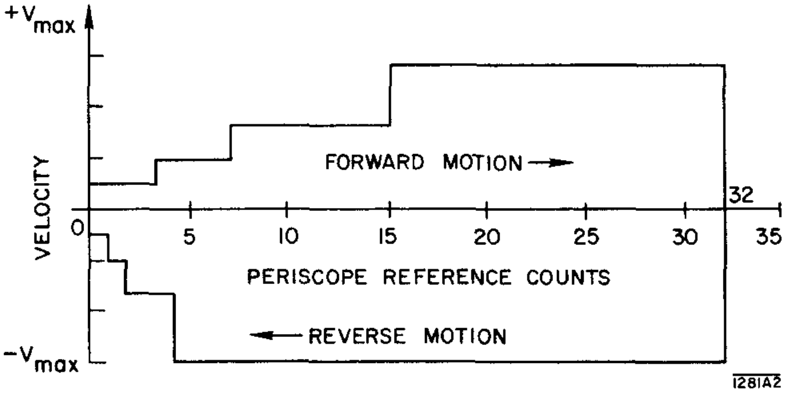


FIG. 1-2--Switching curve for periscope velocity control.

The proper calling sequence for the periscope:

```
JMS   JOBSVR    NOTIFY JOB SUPERVISOR
LAW   PDR       ADDRESS OF PERISCOPE ROUTINE
DBK             RELEASE JOBSVR FROM HIGHEST PRIORITY
LAW   JOB1      ADDRESS OF JOB1
DAC   PSFIN     STORE IN PSFIN
LAW   JOB2      ADDRESS OF JOB2
DAC   PFIN      STORE IN PFIN
JMP   JOBXCT    GO EXECUTE THE NEXT JOB
```

This set of instructions first calls the job supervisor to store the address of periscope drive routine "PDR," and then supplies the address of any job "JOB1" to be executed after the periscope has traveled the distance equivalent to 32 revolutions, and also provides the address of any job "JOB2" to be executed after the periscope is completely homed. Finally the last instruction requests the job executor to execute the periscope drive routine. So after 32 revolutions, the periscope is heading home, JOB1 is executed, and when the periscope is homed, JOB2 is executed.

C. The X-Y stage

The X-Y stage control program actually consists of two identical control programs, one for the X-stage and the other for the Y-stage, working simultaneously. Since both X and Y have identical control programs, it is only necessary to explain how one of them works.

Both X-stage and Y-stage interrupts are assigned to priority level two, and their interrupt addresses are location 102 and 103 in the computer memory, respectively. The subdevice interrupts, for either stage, consist of add-to-memery data overflow interrupt, reference interrupt, complete interrupt, and limit interrupt. The add-to-memory data overflow interrupt is generated whenever the sign bit of either scaler changes from 0 to 1, or from 1 to 0, resulting directly from 2's complement addition. This interrupt is used for updating the stage's position and for switching its velocity accordingly. The reference interrupt is generated whenever the stage passes a fixed point causing a diode to be momentarily in the dark. This interrupt is used for presetting the scalers to ($100,000_8$) counts. The complete interrupt is generated whenever a zero velocity is loaded into the velocity register for more than 100 msec. This interrupt is used for indicating that the stage is completely stopped and the system can proceed to the next operation. The limit interrupt is generated whenever the stage has traveled forward or backward beyond its intended length of travel. This interrupt is used for restoring the scaler and providing warning to the operator. While in the computer mode, the enabled interrupt service routines are the add-to-memory data overflow, the complete, and the limit, and the disabled interrupt service routine is the reference. While in manual mode, all the interrupt service routines are disabled except the reference.

The X-scaler and Y-scaler reside at locations 201 and 202 inside the computer memory, respectively. The counts are incremented or decremented via the add-to-memory feature of the PDP-9 computer. Each scaler is capable of accumulating 2^{17} counts, which, computed over the distance that the X-stage has to travel through, would yield a 2μcon per stage count resolution.

The X-Y stage position is controlled by distance-sampling, rather than by time-sampling. That is, the velocity is programmed to verify as a function of how close the stage is with respect to its destination. In order to move the stage from one position to a desired position, the first step is to compute the distance between the two positions. The next step is to place a set of pre-determined zone distances at the desired position. Then the distance in which the stage must travel to reach its destination is compared with these zones. If the distance is larger than the largest zone distance, then the stage is controlled to move at its maximum velocity to the first zone boundary. From then on, the stage is controlled to move to each zone boundary with the corresponding velocity until it has reached its destination. However, if the initial distance lies inside one of the boundaries, then the stage is controlled to move to the next zone boundary at the corresponding velocity. For the rest of the distance, the stage is controlled in the same manner as in the previous case. In order to get an add-to-memory overflow interrupt at each of the zone boundaries, the initial stage position in the scaler is temporarily stored away in some other location in the computer memory, and the difference between the largest number (377777_8 or 400000_8) and the distance to be traveled is computed and stored in the scaler. Since the initial stage position is stored away, that position has to be updated accordingly each time a zone boundary is crossed. Only until the final destination is reached, the updated stage position is restored back into the scaler, and the scaler should read the desired position.

Figure 1-3 shows a set of concentric zones placed at the desired position for computing relative distances.

The following equations are used to control the position of the stage:

Let X = present position of stage,
XDEST = desired position of stage,
DELTAX = actual distance between the two.
so DELTAX = X - XDEST.

Since velocity of the stage has to be switched at each zone boundary, the algorithm used to approximate DELTAX is given by:

$$\mathrm{DELTAX} = \mathrm{X} - \mathrm{XDEST}$$

$$\cong \sum_{i=1-5}^{5} (\Delta X_i + XOF_i) + DZ$$

where

ΔX_i = distance between zone boundaries,

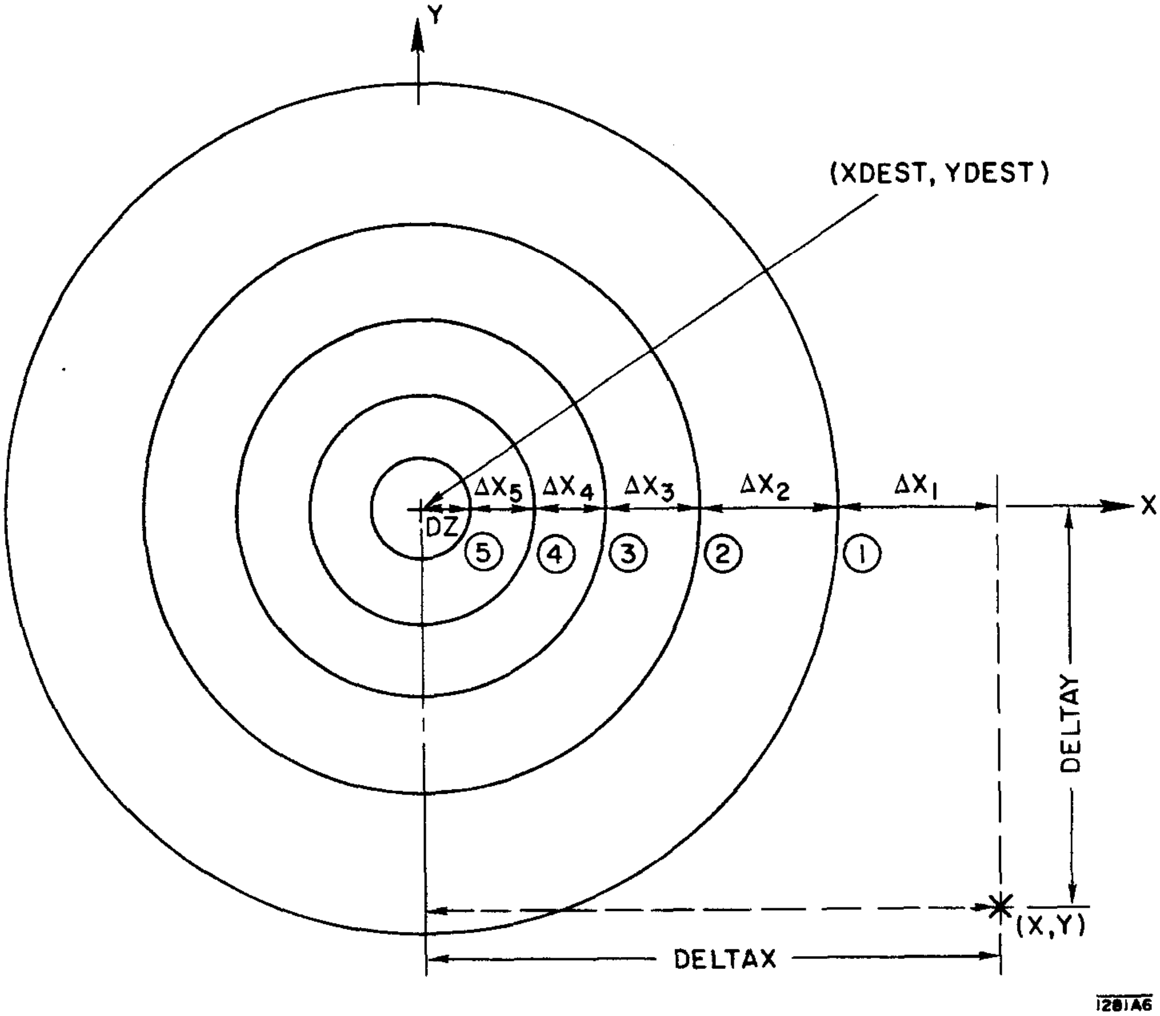


FIG. 1-3--Distance zones for controlling X-Y stage.

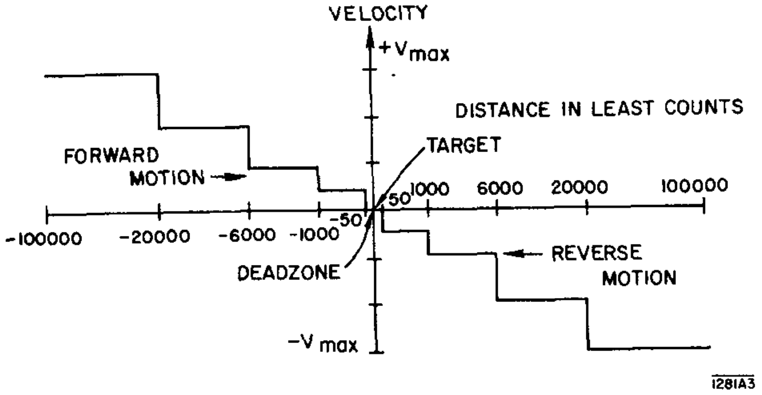


FIG. 1-4--Switching curve for X and Y stage velocity control.

XOF_i = counts accumulated while stage is crossing the zone boundaries.

DZ = counts accumulated while the stage is coasting to a stop inside the dead zone.

Lastly, the final updated stage position is given by

$$XDEST \cong X + \sum_{i=1-5}^{5} (\Delta X_i + XOF_i) + DZ$$

The stage velocity vs stage position (in terms of least counts) is shown in Fig. 1-4.

The proper calling sequence for the X-Y stage:

```
LAC XXXX        LOAD X DESTINATION
DAC XDEST       STORE IN XDEST
LAC YYYY        LOAD Y DESTINATION
DAC YDEST       STORE IN YDEST
JMS JOBSVR      NOTIFY JOB SUPERVISOR
LAW XSDR        ADDRESS OF X-STAGE ROUTINE
LAW YSDR        ADDRESS OF Y-STAGE ROUTINE
DBK             RELEASE JOBSVR FROM HIGHEST PRIORITY
LAW XYDONE      ADDRESS OF JOB1
DAC XSFIN       STORE IN XSFIN
DAC YSFIN       STORE IN YSFIN
JMP JOBXCT      GO EXECUTE THE NEXT JOB.
```

The first instruction loads the X destination from location denoted by XXXX, and the second instruction stores the value into XDEST to be used in the X-store routine. Similarly, the second and third instructions do the same thing to the Y-stage. The fifth instruction requests the job supervisor to store X-stage and Y-stage drive routines. The ninth instruction provides the address of job "XYDONE" to be stored in locations XSFIN and YSFIN of the X-stage and Y-stage drive routines. Finally, the last instruction requests the job executor to execute the X-Y stage. Once the X-stage has reached its destination, it will go to the next job "XYDONE," and similarly when Y-stage has reached its destination, it will also go to the next job

"XYDONE." Thus the job "XYDONE" acts as a common exit routine for both the X-stage and Y-stage routines.

D. The film drives

The three film drives' interrupts are assigned to priority level two, and their interrupts' addresses are locations 104, 105, and 106 in the computer memory, respectively. The subdevice interrupts, for each film drive, consist of reference interrupt, complete interrupt, limit interrupt, and add-to-memory data overflow interrupt. The reference interrupt is generated whenever a brenner mark on the film passes a diode mounted on the film platen. This interrupt is used for counting the number of frames. The complete interrupt is generated whenever a zero velocity is loaded into the velocity register for more than 100 msec. This interrupt is used for indicating that the film drive is completely stopped and the system can proceed to the next operation. The limit interrupt is generated whenever the film drive has reached the end of the roll. This interrupt is used for rewinding the film whenever the film was advancing, and for stopping the film whenever the film was reversing. The add-to-memory data overflow interrupt is not used in the film drive routines. While in the computer mode, the enabled interrupt service routines are the reference, the complete, and the limit; the disabled interrupt service routines are the add-to-memory data overflow. While in the manual mode, all the interrupt service routines are disabled.

The scalers for the three film drives reside at locations 204, 205, and 206 inside the computer memory, respectively. Presently, the positioning of the film is done by stopping on the brenner mark, and this method of positioning is adequate for our purpose, so the encoders are not being used at all. The wiring and logic for the encoders do exist, if they are ever needed.

The film drive control program uses references interrupts to control the position and velocity of the film. The distance between the brenner mark and the center of the picture is fixed, so the problem of film advancing and stopping on the right frame number becomes a problem of simply counting the number of brenner marks and then stopping. On reversing, the film drive control program always adds one to the number of frames to be reversed. When that number of frames is properly reversed, the film drive control program then advances the film by one frame, and the net result is the same; however the backlash in the film storage system is minimized. The velocity of the film drive is programmed according to the number of frames away from the desired frame number. The velocity vs the number of frames for both advancing and reversing is shown in Fig. 1-5.

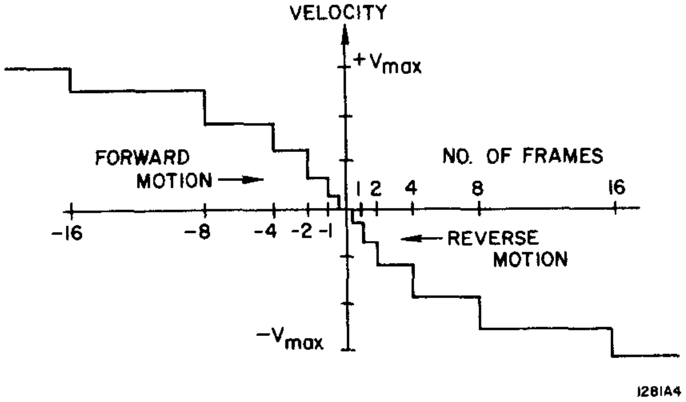


FIG. 1-5--Switching curve for film velocity control.

The proper calling sequence for all three film drives:

```
LAC NF          LOAD NO. OF FRAMES
DAC FLDEST      STORE IN FLDEST
DAC GLDEST      STORE IN GLDEST
DAC HLDEST      STORE IN HLDEST
JMS JOBSVR      NOTIFY JOB SUPERVISOR
LAW FDR         ADDRESS OF FILM DRIVE 1
LAW GDR         ADDRESS OF FILM DRIVE 2
LAW HDR         ADDRESS OF FILM DRIVE 3
DBK             RELEASE JOBSVR FROM HIGHEST PRIORITY
LAW FLDONE      ADDRESS OF FLDONE
DAC FFIN        STORE IN FFIN
DAC GFIN        STORE IN GFIN
DAC HFIN        STORE IN HFIN
JMP JOBXCT      GO EXECUTE THE NEXT JOB
```

The first four instructions load and store the number of frames to be advanced, if the number NF is positive, or to be reversed if the number NF is negative. The fifth instruction requests the job supervisor to store the address of film drive 1, 2, and 3 routines. The tenth instruction provides the address of job "FLDONE" to be stored in locations FFIN, GFIN, and HFIN of film drive 1, 2, and 3, respectively. Finally, the last instruction requests the job executor to execute the three film drives. When each film drive completes its job, it proceeds on to the next job "FLDONE." So the job "FLDONE" acts as a common exit routine for all the film drive routines.

E. The console buttons

The console button interrupt is assigned to priority level 1, and its interrupt address is location 111 in the computer memory. The subdevice interrupt consists of view 1 interrupt, view 2 interrupt, and view 3 interrupt, manual fiducial interrupt, vertex interrupt, image-plan-digitizer interrupt, event reject interrupt, crutch point interrupt, flagged crutch point interrupt, and advance interrupt. The view 1 interrupt is generated whenever button 1 is pushed. This interrupt is used for requesting the system to move the stage so that view 1 gets projected onto the table and the periscope cone. The view 2 interrupt is generated whenever button 2 is pushed. This interrupt is used for requesting the system to move the stage so that view 2 gets projected onto the table and the periscope cone. Similarly, the view 3 interrupt is generated whenever button 3 is pushed. This interrupt is used for requesting the system to move the stage so that view 3 gets projected onto the table and the periscope cone. The manual fiducial interrupt is generated whenever button 4 is pushed. This interrupt is used for requesting the system to read and record the X, Y position of the stage as a fiducial point. The vertex interrupt is generated whenever button 5 is pushed. This interrupt is used for requesting the system to read and record the X, Y poisition of the stage as a vertex point, and also set the periscope in motion to start the spiral scan. The image-plane-digitizer (IPD) interrupt is generated whenever button 6 is pushed. This interrupt is used for requesting the system to read and record the IPD scalers. The event reject interrupt is generated whenever button 7 is pushed. This interrupt is used for requesting the system to reject the event. The system responds to the operator's request by typing out a message which requires the operator to enter a reject code. Once the system enters the reject code into the ID (identification) buffer, then it does some bookkeeping before it proceeds onto the next event. The flagged crutch point interrupt is generated whenever button 8 is pushed. This interrupt is used for requesting the system to read and record the X, Y position of the stage as a flagged crutch if it happens that view 2 is being measured. However, if view 1 or view 3 is being measured immediately after a spiral scan, the system uses the flagged crutch point measured in the previous view to predict where the crutch point should be in the present view, and then it automatically proceeds to move the stage to the predicted position. The crutch point interrupt is generated whenever button 9 is pushed. This interrupt is used for requesting the system to read and record the X, Y position of the stage as a crutch point. In a completely different application, such as doing a calibration on the X-Y stage, this interrupt is used for requesting the system to advance to the next event, view, or frame as the case may be, and also to write IPD coordinates, or crutch points onto the magtape if it is required to do so.

The console control program consists of interrupt service routines which implement the functions described in the previous paragraph. The console buttons are program interlocked, that is, each button can be enabled or disabled individually by programming. During the normal operation only the view select button is enabled, and the rest of the buttons are disabled. The rest of the buttons are individually enabled whenever they are needed in the event measurement sequence. Furthermore, all console buttons are disabled whenever the stage or the periscope is in motion. All these are precautionary measures to guard against unnecessary or accidental requests from the operator, and random interrupts produced by noise.

F. The image-plane-digitizer

The image-plane-digitizer (IPD) does not have an interrupt assigned to it, but rather it has a subdevice interrupt connected to button 6 of the console buttons. The function of this interrupt was described in the previous paragraph. The main function of the IPD is to allow the operator to measure crutch points during a spiral scan. There is a slight saving in the total measuring time, since a more conventional method would be to have the operator to wait until the spiral scan is finished, and then let the stage free so he could move the stage around to measure the crutch points. In this manner, the IPD programs are generally simpler, and consist of routines that read, store, and write IPD data onto the magtape.

G. The auto-fiducial

The auto-fiducial is assigned to priority level three, and its interrupt address is location 112 in the computer memory. The function of the auto-fiducial control is to allow the system to be able to locate and measure the position of the fiducials automatically. The system obtains its information through diode detectors, and then moves the stage in such a direction as to obtain maximum signal from the diode detectors. The software, auto-fiducial control program has not been implemented yet.

IV. Spiral Reader Support Programs

The spiral reader support programs consist of manual fiducial measurement routines, vertex measurement routines, MRQ (measurement request tape) routines, ID (Indicative Data) routines, crutch point routines, and miscellaneous routines.

i) The Manual Fiducial Measurement Routines

The manual fiducial measurement routines consist of upper to lower fiducial measurement routine, view 1 manual fiducial routine, view 2 manual fiducial routine, view 3 manual fiducial routine, manual fiducial estimating routine, and lower fiducial measurement tolerance check routine.

A. The upper to lower fiducial measurement routine

The function of the upper to lower fiducial measurement routine is to allow the operator to tell the system which set of fiducials is needed in each of the three views, and also the distance between each set of fiducials. This routine requests the operator to measure view 1 upper fiducial by typing out V1UF on the teletype. The operator responds to this request by moving the stage manual to the upper fiducial and pushing the manual fiducial button. Once this routine records the stage position, it will then request the operator to measure view 1 lower fiducial by typing out V1LF on the teletype. The operator should respond accordingly. Similarly, this routine requests the operator to do the same thing for view 2 and view 3. The proper calling sequence for this routine is:

```
JMS     ULMFR
 .        .
 .        .
```

B. The view 1, 2, and 3 manual fiducial routines

The function of the view manual fiducial routine is to have the system automatically move the stage to the fiducial positions for operator's verification. Since there is no assurance that the film will stop at the same position on the film platen after every film advance, this means that the position of the upper fiducial position of each view has to be estimated. The system can only move the stage to approximately where the upper fiducial should be. But once the operator has measured the upper fiducial position exactly, the estimated upper fiducial position can be updated for the next frame. The lower fiducial position can be computed exactly, provided the upper fiducial position is located exactly, since the distance between the upper and lower fiducial is always fixed. After the operator has measured the lower fiducial exactly, this routine also performs a lower fiducial measurement tolerance check to determine whether the operator has measured the upper and lower fiducial correctly. The lower fiducial measurement tolerance check routine simply performs the comparison between the measured upper to lower fiducial distance to the known distance. If the measured upper to lower fiducial distance is within the specified tolerance, then the system is permitted to continue. However, if the measured upper to lower fiducial distance exceeds the specified tolerance, then this routine will drive the stage back to the upper fiducial and typeout a message suggesting the operator remeasure the fiducials. The proper calling sequence for view 1, 2, and 3 manual fiducial routines are:

```
JMS   V1MFR    VIEW1 MANUAL FIDUCIAL ROUTINE
JMS   V2MFR    VIEW2 MANUAL FIDUCIAL ROUTINE
JMS   V3MFR    VIEW3 MANUAL FIDUCIAL ROUTINE
 .     .
 .     .
```

The proper calling sequence for manual fiducial estimating routine is:

```
JMS   MFEST    GO TO MANUAL FIDUCIAL EST. ROUT.
XX             VALUE OF OLD EST. VALUE
XX             VALUE OF MEASURED VALUE
DAC   XXXX     STORE NEW ESTIMATED VALUE
```

This first instruction calls the manual fiducial estimating routine to estimate the position of the upper fiducial location. The next two instructions are the old estimated value, and the measured value is the average of the old estimated and the measured values. The new estimated value is left in the accumulator and should be saved. The proper calling sequence for the lower fiducial measurement tolerance check routine is:

```
JMS   LFMTCR   GO TO LOWER FIDUCIAL CHECK ROUTINE
XX             CALCULATED VALUE
XX             MEASURED VALUE
JMP   XXXX     GO REMEASURE FIDUCIALS
 .     .
 .     .
```

This first instruction calls the lower fiducial check routine to determine whether the measured lower fiducial is within the tolerance of the computed lower fiducial value. The next two instructions are the calculated value, and the measured value. If the measured and computed values agree closely, then the fourth instruction is skipped; otherwise the fourth instruction is executed and the fiducials have to be remeasured again.

ii) The Vertex Measuring Routines

The vertex measuring routines consist of vertex computation routines for view 1, view 2, and view 3, stage to vertex routines for view 1, view 2, and view 3, multivertex routines for view 1, view 2, and view 3, and beam track control sequence routines.

A. The vertex computation routine for view 1, 2, and 3

The vertex centering routine requires the knowledge of two things: (1) camera orientation in the bubble chamber, and (2) view-to-view measuring sequence. The measuring sequence is chosen to begin in view 2, since the initial scanned data were taken from view 2, and the measuring sequence is from view 2 to view 1 to view 3. Figure 1-6 gives the camera orientation in the bubble chamber, and Fig. 1-7 gives the film orientation on the stage. In view 2 the system can only move the stage as close to a vertex as the grid size would permit.

View 2 vertex computation can be given by the following equations:

$$\overline{\mathrm{V2VTX}} = \frac{(2N-1)GZ}{2} + \mathrm{V2LFX} - \mathrm{GDXLF} + \mathrm{GDXFF}$$

$$\overline{\mathrm{V2VTY}} = \frac{(2N-1)GZ}{2} + \mathrm{V2LFY} - \mathrm{GDYLF} + \mathrm{GDYFF}$$

where

$\overline{\mathrm{V2VTX}}$ and $\overline{\mathrm{V2VTY}}$ are the estimated vertex centers for view 2,

GZ is the grid-zone in stage counts,

N is the grid-zone number in binary,

V2LFX and V2LFY are the coordinates of view 2 lower fiducial position,

GDXLF and GDYLF are the distances between the edges of the lower fiducial,

GDXFF and GDYFF are fudge factors for adjusting the grid orientation.

GZ, GDXLF, and GDYLF are measured with the stage,

V2LFX and V2LFY are determined at the time when the fiducials are measured, and

GDXFF and GDYFF are obtained by trial and error until the estimated vertex position and the scanned vertex coincide in the same grid zone.

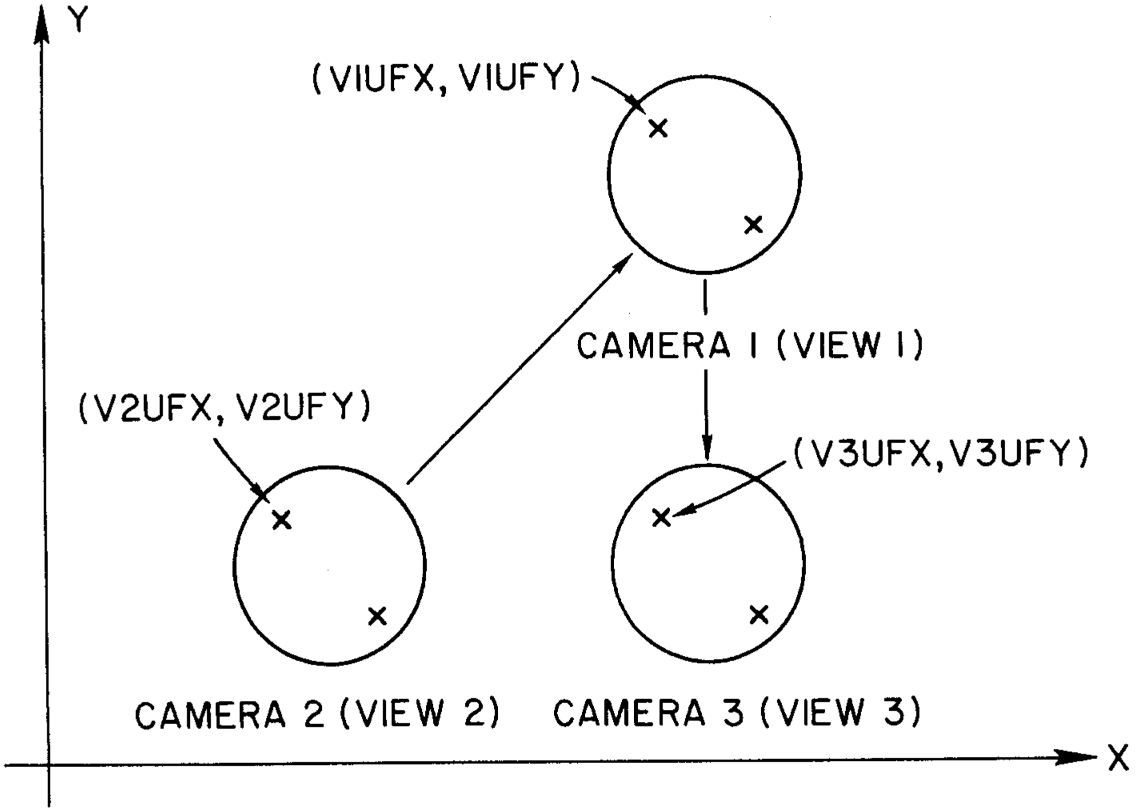


FIG. 1-6--Camera orientation in bubble chamber.

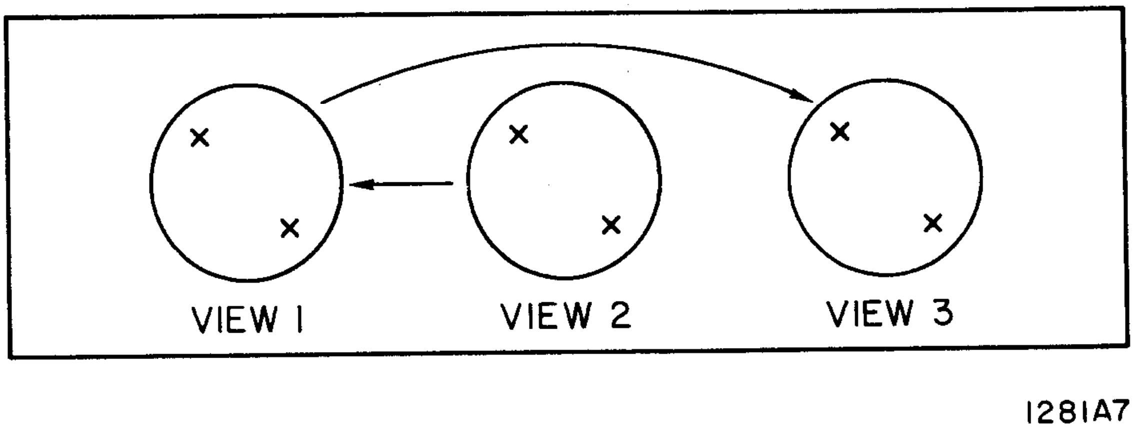


FIG. 1-7--Film orientation on stage.

View 1 vertex computation can be given by the following equations:

$$\overline{V1VTX} = V1UFX + (V2VTX - V2UFX) + V1XFF$$

$$\overline{V1VTY} = V1UFY + (V2VTY - V2UFY) + V1TFF + V1CORY$$

where

$\overline{V1VTX}$ and $\overline{V1VTY}$ are the estimated vertex centers for view 1,
V2VTX and V2VTY are the measured vertex centers in view 2,
V1UFX and V1UFY are the measured upper fiducials in view 1,
V2UFX and V2UFY are the measured upper fiducials in view 2,
V1XFF and V1YFF are view 1 fudge factors,
V1CORY is a correction factor for view 1 in the Y direction.

The view 1 fudge factors are obtained by trial and error until the estimated vertex falls in the TV screen.

View 3 vertex computation can be given by the following equations:

$$\overline{V3VTX} = V3UFX + (V1VTX - V1UFX) + V3XFF + V3CORY$$

$$\overline{V3VTY} = V3UFY + (V1VTY - V1UFY) + V3YFF + V3CORY$$

where

$\overline{V3VTX}$ and $\overline{V3VTY}$ are the estimated vertex centers for view 3,
V1VTX and V1VTY are the measured vertex centers in view 2,
V3UFX and V3UFY are the measured upper fiducials in view 3,
V1UFX and V1UFY are the measured upper fiducials in view 1,
V3XFF and V3YFF are view 3 fudge factors, and
V3CORX and V3CORY are view 3 correction factors in both the X direction and Y direction.

The view 3 fudge factors are obtained by trial and error until estimated vertex falls in the TV screen.

View 1 and view 3 correction factors can be given by the following equations:

$$V3CORX = V1VTX - \overline{V1VTXE}$$

$$V3CORY = (V2UFY - V2VTY) - (V1UFY - V1VTY)$$

$$V1CORY = V1CORY \pm \frac{|V1CORY - V3CORY|}{4}$$

$$+ \text{ if } (V1CORY - V3CORY) < 0$$

$$- \text{ if } (V1CORY - V3CORY) \geq 0$$

where

$\overline{V1VTXE}$ is the estimated view 1 vertex center in X direction, and the rest of the variables have been defined previously.

A word of caution: These equations have been derived for camera orientation and measuring sequence shown in Figs. 1-6 and 1-7, respectively. Any attempt to change the measuring sequence, or to change the camera orientation in the bubble chamber would require a new set of equations for view 1, 2, and 3 vertex computation routines.

B. Vertex buffer design

The heart of the vertex control routine is the ability to allow the operator to measure several vertices before he has to remeasure the fiducials again. The system is presently designed to allow the operator to measure up to four events in each view, and each event can have one primary vertex and three secondary vertices. This means in each view a maximum of 16 vertices could be measured, and furthermore all 16 vertices would have to be remembered for the next view. To keep track of the order of these vertices, two buffers are used: one buffer is used for storing computed vertices, and the other is used for storing measured vertices. In order to keep track which set of vertices belong to which beam track, several pointers and buffers are used and they have to be initialized properly each time. To make sure the buffer pointers get initialized properly, the beam track control sequence routine is used for keeping track of the buffer pointers for each event, and the multivertex routine is used for keeping track of the buffer pointers for each vertex within the event. Finally, the stage to vertex routine drives the stage to the vertex coordinate provided by the event and vertex buffer pointers.

iii) The MRQ (Measurement Request Tape) Routines

The MRQ routines consist of read MRQ file content routine, roll search and confirm routine, locate MRQ from roll-frame-beamtrack (RFB) information, film advance from MRQ information, and read one MRQ record. Before describing the function of each of the routines, it is necessary to present the MRQ tape format. First, the MRQ file content format serves very much like the table of contents in a book. It always occupies the first two records of the MRQ tape. The two records themselves contain information concerning the location of the beginning of each roll on the MRQ tape. Figure 1-8 shows the MRQ file content format. For instance, the beginning of roll # X is located on the third record of the tape, and roll # Y begins on the 50th record of the tape, and etc. Secondly, the content of a MRQ record is given in Table 1. Each record contains roll, frame, beamtrack number, scan information, and vertex locations, and etc. and the last two records must be identical.

A. The MRQ search routines

The MRQ search routines consist of read MRQ file content routine, roll search and confirm routine, locate MRQ from roll-frame-beamtrack (RFB) information.

The read MRQ file content routine reads the first two records of the MRQ tape into a buffer. This routine should only be used every time when a new MRQ tape is mounted, or unless the file content information in the buffer is wiped out.

The roll search and confirm routine is used for locating the roll number. The way it works is by looking through the MRQ file content buffer to decide how many records to advance or reverse. If the proper number of records has been advanced, or reversed, it then confirms that it is on the correct roll by comparing the desired roll number to the one it read from the magtape. Furthermore, once in awhile this routine might get lost while searching for the proper roll number. To overcome this difficulty, the routine will automatically rewind the tape and start searching the tape from the very beginning, and no operator assistance is required.

The locate MRQ from roll-frame-beamtrack information obtains the RFB information from the operator via the teletype. This routine uses the roll search and confirm routine to locate the proper roll number on tape, and once the roll number is found, it then continues to search for the proper frame and beamtrack number. Moreover, it will also let the operator know whether it has found the RFB, or the requested RFB is missing on the MRQ tape.

B. Miscellaneous MRQ routines

The miscellaneous MRQ routines consist of film advance from MRQ information, and read one MRQ record routine. The film advance from MRQ information routine computes the difference between the present roll-frame numbers and the ones in the MRQ buffer, and then advances the film accordingly. The read one MRQ record routine merely reads one MRQ record into the MRQ buffer.

iv) ID (Indicative Data Routines)

ID routines consist of ID request routine, convert fiducials for ID record routine, convert vertex for ID record routine, write vertex and view number into ID record routine, copy partial MRQ into ID record routine, write beamtrack number and event type into ID record routine, and write ID record routine. Before describing the function of each of the routines, it is perhaps useful to explain the function of an ID record and its format. First of all, an ID record is used for providing information concerning the event being digitized. This ID record must be the first

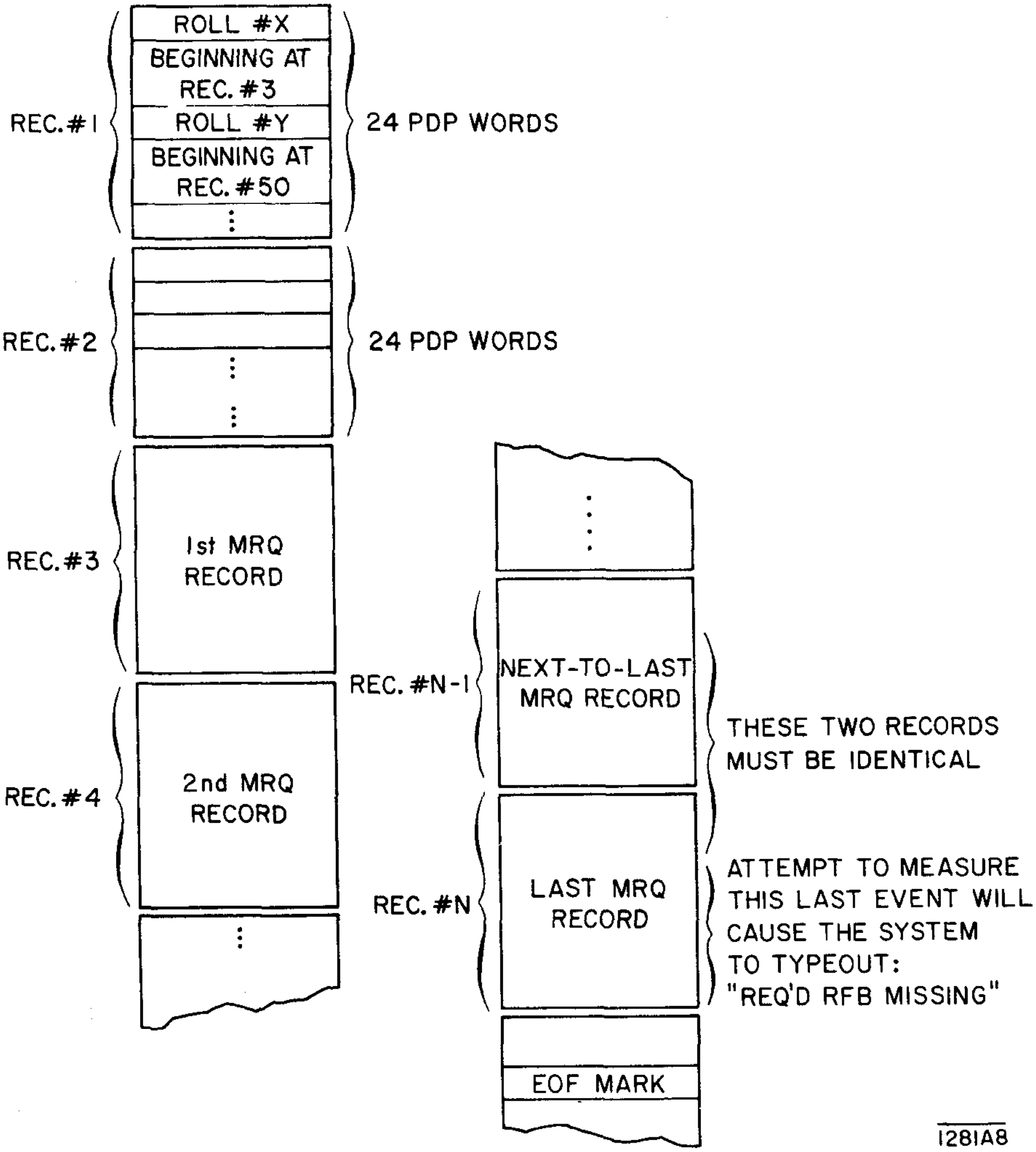


FIG. 1-8--MRQ file content format.

TABLE 1

FORMAT OF MEASUREMENT REQUEST (MRQ) TAPE FOR SLAC SPIRAL READER

Each quantity should occupy one-half-word integer (16 bits).

WORD	
1.	Format number
2.	Roll number
3.	Frame number
4.	Beamtrack number
5.	Event type
6.	Measurement number
7.	Experiment number
8.	Scanner number
9.	Year scanned
10.	Month scanned
11.	Day scanned
12.	Binary switches
13.	Binary switches
14.	X coordinate lst vertex
15.	Y coordinate lst vertex
16.	X coordinate 2nd vertex
17.	Y coordinate 2nd vertex
18.	X coordinate 3rd vertex
19.	Y coordinate 3rd vertex
20.	X coordinate 4th vertex
21.	Y coordinate 4th vertex

Record length: A record length of at least 48 bytes (24 PDP-9 words) should provide enough space for the 21 MRQ items above and those which might be added later, and assure proper IBM 360 word boundary alignment.

record of each spiral scan, and the format of an entire spiral scan is shown in Fig. 1-9. Secondly, the format of each ID record is shown in Table 2. The location of each item in the record is fixed and not interchangeable. Each ID record is 512 words long with the ID occupying the first 50 words of the record, and the rest of the record is filled with ones. When bit zero of the first word of the ID record is set to one, it means the beginning of a set of events. This bit is set at the beginning of each new measuring sequence in view 2, and it is reset to zero at the end of each spiral scan. Note also that each coordinate occupies two words. This is necessary because the stage coordinate uses all 18 bits, and the magtape controller can only accept 16 bits, so the 18 bit coordinate is divided up into two words with the first word containing the two most significant bits, and the second word containing the latter 16 bits.

The <u>ID request routine</u> is used for requesting information from the operator to be stored in the ID buffer. A list of information is given in "the operational aspect of system genie." When all the necessary information is stored away in the ID buffer, this routine then continues to read the MRQ file content information.

The <u>convert fiducials for ID record routine</u>, and convert vertex for ID record routine are similar in respects. That is, they both call the same routine which converts the coordinate from a single word into two words and stores them in the ID buffer.

The <u>write vertex and view number into ID record routine</u> is used for updating the vertex and view number information in the ID buffer just before each spiral scan.

The <u>write beamtrack number and event type into ID record routine</u> is used during view 1 and view 3 event measurement for writing the beamtrack number and event type into ID buffer from a buffer called the beamtrack number and event type buffer. The information was stored into this buffer during view 2 measurements.

The <u>copy partial MRQ into ID record routine</u> is used for copying a part of the MRQ buffer containing the roll, frame, beamtrack number, event type, etc., information into the ID buffer. This routine is used in view 2 measurement sequence whenever a MRQ record is read into the memory.

The <u>write ID record routine</u> is used prior to each spiral scan. This routine simply writes the ID buffer onto the magtape.

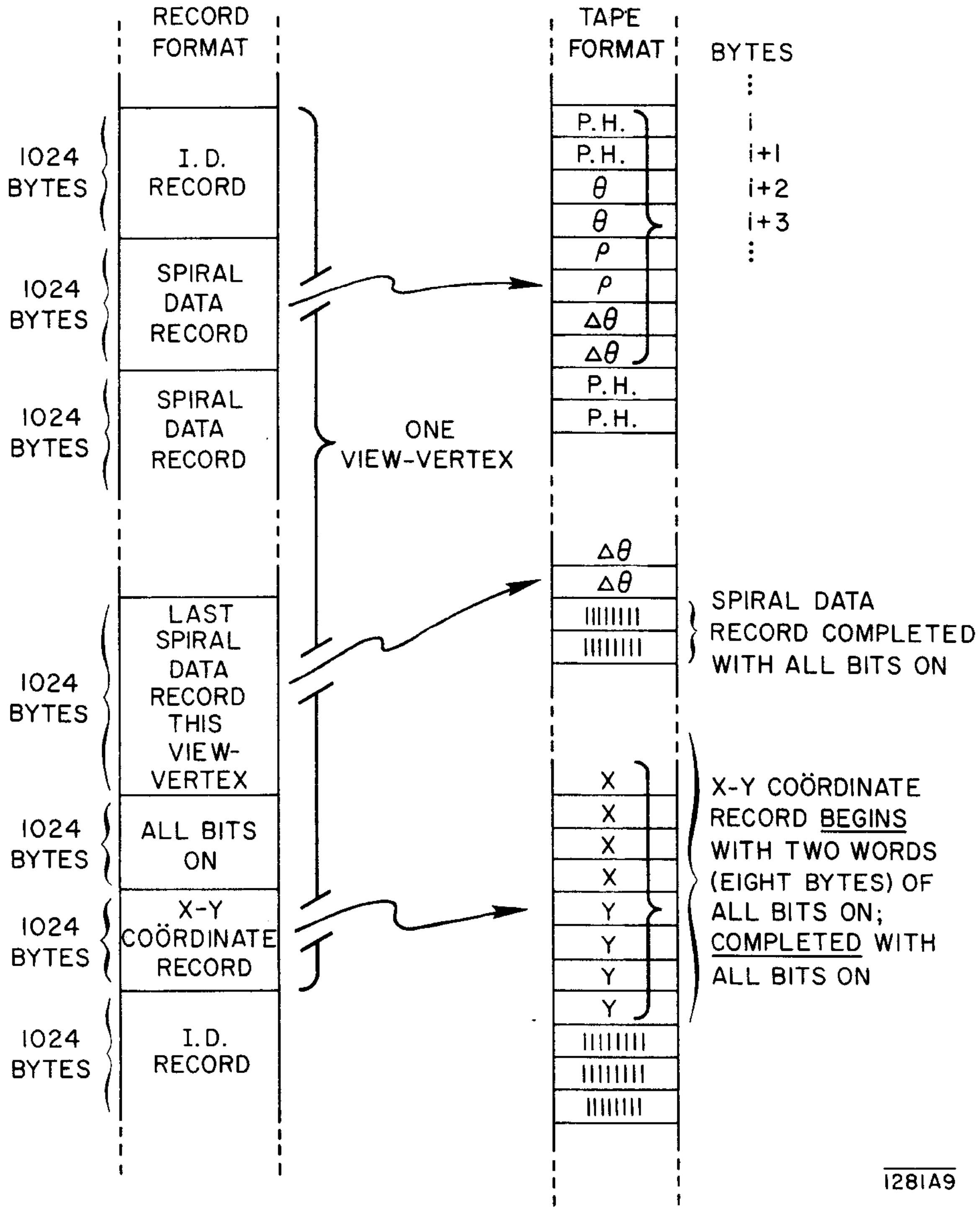


FIG. 1-9--Spiral Reader output tape record format.

TABLE 2

SPIRAL READER OUTPUT TAPE: I.D. RECORD FORMAT

50 18-bit words

1.	Bit 0 on for beginning of a set of events
2.	
3.	Vertex number
4.	View number
5.	Number of fiducials
6.	
7.	
8.	
9.	
10.	
11.	
12.	
13.	X coordinate 1st vertex
14.	Y coordinate 1st vertex
15. 16.	X coordinate Fid. 1
17. 18.	Y coordinate Fid. 1
19. 20.	X coordinate Fid. 2
21. 22.	Y coordinate Fid. 2
23. 24.	X coordinate Fid. 3
25. 26.	Y coordinate Fid. 3
27.	
28.	Chamber number
29.	Magnetic tape number
30.	Operator number
31.	Year measured
32.	Month measured
33.	Day measured
34.	Machine number
35.	Roll
36.	Frame
37.	Beamtrack
38.	Event type
39.	Measurement number
40.	Experiment number
41.	Scanner number
42.	Year scanned
43.	Month scanned
44.	Day scanned
45. 46.	Binary flags
47. 48.	X coordinate vertex
49. 50.	Y coordinate vertex

v) Crutch Point Routines

Crutch point routines consist of IPD (Image Plane Digitizer) routines, crutch point measurement routine, and flagged crutch point measurement routines.

A. IPD routines

The IPD routines consist of IPD initialization routine, coordinate button interrupt service routine, and advance button interrupt service routine. The main advantage of IPD is that it allows the operator to measure crutch points during a spiral scan, thus decreasing the total event measurement time by several seconds. The disadvantage of IPD is that new coordinate conversion and calibration routines are needed to correct for optical and alignment errors. Aside from the advantages and the disadvantages, the IPD initialization sets up the buffer for storing the IPD coordinates, and enables console button 6 whenever a spiral scan is initiated. A maximum of 16 crutch points could be measured for each event. The coordinate button interrupt service routine keeps track of the number of crutch points that are being recorded, and the advance button interrupt service routine requests the magtape supervisor to write out a record containing the IPD crutch point information.

B. Crutch point measurement routine

The crutch measurement routine allows the operator to measure crutch points with the stage. The advantage here is that no conversion and calibration routines are needed. The disadvantage is that the crutch points can only be measured after each spiral scan, thus prolonging the measuring time. The crutch measurement routine initializes the crutch point output buffer, and enables the console button 9. Once console button 9 is pushed, the crutch point measurement routine also depends upon the coordinate button interrupt service routine to keep track of the number of crutch points that are being recorded, and the advance interrupt service routine to request the magtape supervisor to write out a record containing the crutch point information.

C. Flagged crutch point measurement routines

The main function of the flagged crutch point measurement routine is to allow the operator to measure one or more crutch points in view 2, and to have the coordinates of these crutch points remembered so that the system can move the stage automatically to these corresponding positions in view 1 and view 3. The flagged crutch point measurement routines consist of flagged crutch point initialization routine, locate flagged crutch point buffer address from beamtrack and vertex number information routine, flagged crutch point measurement in view 1, view 2, and view 3 routines.

a) Flagged crutch point measurement buffer design

Since it is possible to measure 4 vertices per event and up to 4 events per frame, then it is assumed that each event can have a maximum of 7 flagged crutch points. This means that a primary vertex with 3 secondary vertices may have a maximum of 7 flagged crutch points for each event. This means the buffer for flagged crutch points should be capable of storing at least 4 × 7 = 28 pairs of (x, y) coordinates. Furthermore, in order to be able to quickly indentify a particular vertex and its associated flagged crutch points, it is necessary to build another buffer whose function is to store the address of crutch points. Since there can exist 16 vertices per frame, a buffer of size 16 words is constructed to store 16 flagged crutch point buffer pointers. Thus two buffers, one for storing flagged crutch point buffer pointers, and the other for storing flagged crutch point coordinates are required to do the job. In order to retrieve a flagged crutch point coordinate it is only necessary to know the beamtrack number and the vertex number. With these two pieces of information, a table look up scheme is used to locate the correct flagged crtuch point buffer pointer. Whenever a particular beamtrack and one of its corresponding vertices has no flagged crutch to be measured, then its corresponding location in the buffer for flagged crutch point buffer pointer is set to zero to indicate that no flagged crutch point is stored in the flagged crutch point coordinate buffer. A word of all ones in the flagged crutch point coordinate buffer means the end of a set of flagged crutch points for a given beamtrack and a vertex.

b) Flagged crutch point routines

The flagged crutch point initialization routine is used for clearing the buffer that contains crutch point buffer pointers and also the buffer that contains crutch point coordinates with zeros. Moreover, it also reinitializes counters and pointers for the flagged crutch point measurement routines. This routine is used in view 2 at the beginning of each new measurement sequence.

The locate flagged crutch point buffer address from beamtrack and vertex number information routine is used for locating what the title suggests. In view 2, this routine allows the proper flagged crutch point coordinate buffer address to be stored correctly in the buffer for flagged crutch point buffer pointers. In view 1 and view 3, this routine allows the proper flagged crutch point coordinate buffer address to be retrieved correctly from the buffer for flagged crutch point buffer pointers.

The flagged crutch point measurement in view 2 routine allows the operator to measure and store the flagged crutch points. This routine makes sure that the flagged crutch point coordinate buffer pointer gets properly sequenced, and a word of all ones is properly placed in the flagged crutch point coordinate buffer at the end of every set of flagged crutch point measurements. The flagged crutch point measurement in view 1 routine automatically moves the stage to the crutch point location predicted from the view 2 measurement after each spiral scan. Once the operator has positioned the stage over the desired crutch point location by pushing console button 8, if a word of all ones in the flagged crutch point coordinate buffer was not detected, the stage automatically moves on to the next crutch point location. However, if a word of all ones in the flagged crutch point coordinate buffer was detected, then advance button is enabled to permit the operator to move on to the next event. The flagged crutch point measurement in view 3 routine is similar to the flagged crutch point measurement in view 1 routine, except the flagged crutch point measurement in view 3 automatically moves the stage to the crutch point location predicted from the view 1 measurement after each spiral scan.

vi) Miscellaneous routines

Miscellaneous routines consist of routines that are frequently used throughout the whole system. Routines such as view drive routine, film drive routine, system recovery routine, stage coordinate routine, input buffer right justifier routine, output buffer left justifier routine, and memory-to-memory data transfer routine.

A. The view drive routine

The view drive routines automatically move the stage to the desired position in the requested view, and also the desired position is remembered for future reference. That is, after the stage has moved to the desired position, the operator then moves the stage manually to some other location; in order to bring the stage back to its initial position automatically, all the operator has to do is to push the corresponding view select button. The proper calling sequence is:

```
LAC   aaaa                    LOAD X-DESTINATION
DAC   VIEW1X/VIEW2X/VIEW3X    STORE IN VIEW1 OR VIEW2 OR VIEW3
LAC   bbbb                    LOAD Y-DESTINATION
DAC   VIEW1Y/VIEW2Y/VIEW3Y    STORE IN VIEW1 OR VIEW2 OR VIEW3
```

```
JMS    V1SR/V2SR/V3SR          GO TO VIEW1 OR VIEW2 or VIEW3
 .           .
 .           .
```

The X-destination is stored in location named aaaa, and similarly the Y-destination is stored in location named bbbb. The desired X-position is stored in the corresponding view storages VIEW1X, or VIEW2X, or VIEW3X location. The routine that does the actual driving of the stage is V1SR, or V2SR, or V3SR.

B. <u>The film drive routine</u>

The film drive routine automatically advances or reverses all three film drives simultaneously by any desired number of frames. The proper calling sequence is:

```
LAC    aaaa       NUMBER OF FRAMES TO BE MOVED
DAC    FLDEST     STORE IN FLDEST FOR FILM DRIVE 1
DAC    GLDEST     STORE IN GLDEST FOR FILM DRIVE 2
DAC    HLDEST     STORE IN HLDEST FOR FILM DRIVE 3
JMS    FLDR       GO MOVE FILM
 .       .
 .       .
```

The number of frames to be advanced or reversed is stored in the location labeled by aaaa. If the value stored in aaaa is positive then the film drives will advance; otherwise if aaaa contains negative value, the film drives will reverse. The routine that does the driving of the film is FLDR.

C. <u>The system recovery routine</u>

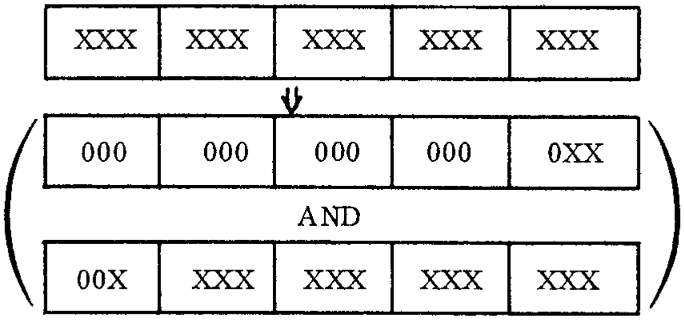
The system recovery routine is used for recovering from any system or operator error. This routine requires a predetermined location to be stored in the recovery routine, such that whenever the recovery routine is called, it will reorientate the system to begin execution from the predetermined location. The system recovery routine consists of system control sequence pointing routine, and system control sequence recovery routine. The system control sequence pointing routine is used for storing pointer in the system control sequence recovery routine. The proper calling sequence is:

```
GAMMA   JMS   SCSPR   GO TO SYSTEM CONTROL   SEQUENCE POINTER ROUT.
         .      .
         .      .
```

This routine stores the address of GAMMA into the system control sequence recovery routine. So if recovery is required, the system will recover to the address labeled GAMMA.

D. The stage coordinate routine

The stage coordinate routine is used for converting X coordinate and Y coordinate from a single 18-bit word into two 16-bit words as indicated below:



where X's represent the useful bits. This routine must be used whenever X or Y coordinate must be recorded onto the magtape. This is because the magtape controller automatically drops 2 bits whenever information is transferred from the computer to the magtape controller. The magtape controller has a 16-bit buffer. The proper calling sequence is:

```
JMS   SCWCR   GO TO X-Y STAGE COORDINATE WORD CONVERTER ROUT.
LAW   XXXX    ADDRESS WHERE DATA SHOULD BE RETURNED
XX            X-COORDINATE VALUE
XX            Y-COORDINATE VALUE
 .       .
 .       .
```

The address where converted data should be stored is labeled by XXXX. That is, the two X and Y coordinates are going to be converted from two 18-bit words into four 16-bit words and they are to be stored consecutively from the address starting with XXXX. This routine also requires that the X and Y coordinate be stored in the locations labeled XX's prior to the execution of this routine.

E. The input buffer right justifier routine

The input buffer right justifier routine is used for shifting each word in a set of words to the right by two bits. This is necessary because when the magtape controller transfers a word to the computer, it always transfers the word into bits 0-15 of the word, and leaves the two least significant, bits 16 and 17, with zeros. In order for the data to make sense, they have to be right justified before they can be used. The proper calling sequence is:

```
JMS    IBRJR    GO TO INPUT BUFFER RIGHT JUSTIFIER ROUT.
LAW    -N       INPUT BUFFER SIZE
LAW    XXXX     INPUT BUFFER ADDRESS
 .       .
 .       .
```

The buffer size has to be a negative quantity and the address of the input buffer is labeled by XXXX. These two parameters must be determined before the routine is executed.

F. The output buffer left justifier routine

The output left justifier routine is used for shifting each word in a set of words to the left by two bits. This is necessary because when the magtape controller receives a word from the computer, it always accepts data from bits 0-15, and information in bits 16 and 17 are lost. In order to record the data accurately onto the magtape, they have to be left justified before they are actually written out onto the magtape. The proper calling sequence is:

```
JMS    OBLJR    GO TO OUTPUT BUFFER LEFT JUSTIFIER ROUT.
LAW    -N       OUTPUT BUFFER SIZE
LAW    XXXX     OUTPUT BUFFER ADDRESS
```

The buffer size has to be a negative quantity and the address of the output buffer is labeled by XXXX. These two parameters must be determined before the routine is executed.

G. The memory-to-memory data transfer routine

The memory-to-memory data transfer routine is used for transferring a block of data from one part of the memory to a different part of the memory. The proper calling sequence is:

```
JMS    MMDTR    GO TO MEMORY-TO-MEMORY DATA TRANSFER ROUT.
LAW    -N       NUMBER OF WORDS TO BE TRANSFERRED
LAW    XXXX     ADDRESS WHERE DATA TO BE TRANSFERRED FROM
LAW    YYYY     ADDRESS WHERE DATA TO BE TRANSFERRED TO
```

The number of words to be transferred must be expressed as a negative quantity, the address from which the data should be transferred is labeled by XXXX, and the address to which the data should be transferred is labeled by YYYY.

PART 2

SYSTEM GENIE'S OPERATIONAL ASPECTS

There are two communications links between the operator and the control program "System Genie." The main communication link, which works both ways, is the teletype-keyboard-printer, and the other communication link is the console buttons. Figure 2-1 shows these communication links

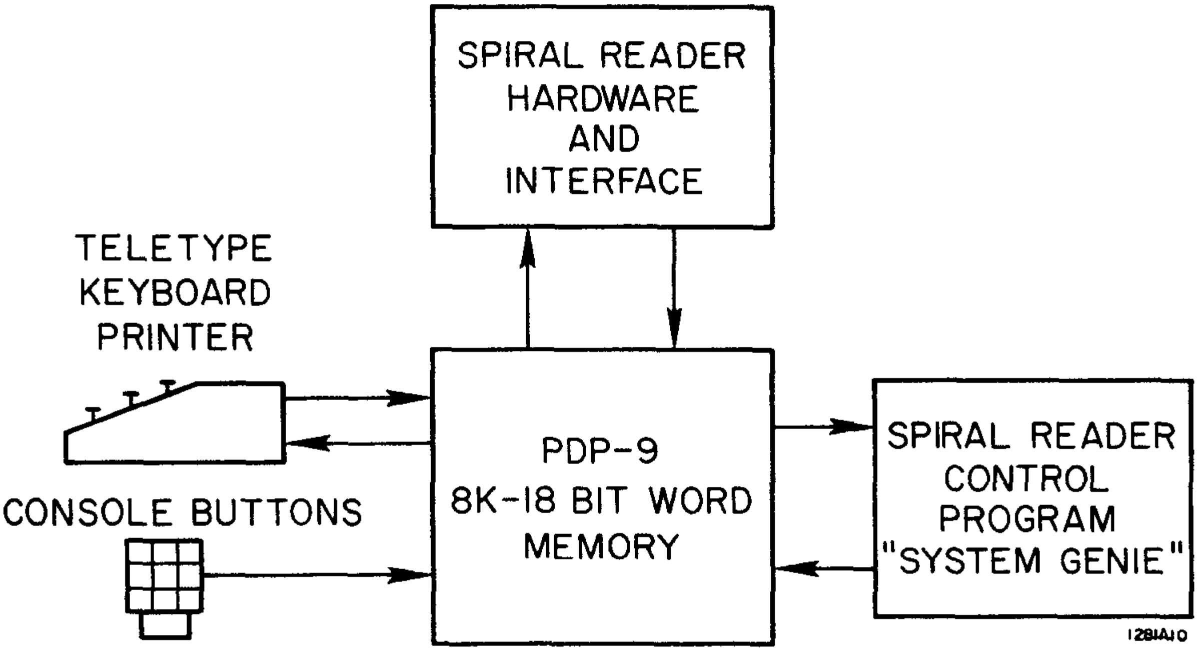


FIG. 2-1--Communication links for the Spiral Reader system.

The following paragraphs will describe in length how an operator can communicate with the system. Unless otherwise stated, the name "teletype-keyboard-printer" will just be referred to as "the teletype," and also "System Genie" will just be referred to as "the system."

I. Teletype Communication

i) Teletype Command Codes

The main communication link between the operator and the control program "System Genie" is through the teletype printer/keyboard. System Genie obeys only a set of special characters, which are /, #, $, =. Each special character is a command code which dictates to the system what to do. The following describes the function of each command code:

/: commands the system to clear and initialize the keyboard buffer, and also takes the system out of any scaler display loops.

#: commands the system to execute indirectly the routine specified in the keyboard buffer.

Ex. A#. The system will execute the routine whose address was previously stored in the location named A.

$: commands the system to convert decimal data stored in the keyboard buffer into binary data and store in some predetermined location in the computer memory.

=: commands the system to printout the decimal value of the location whose address is specified in octal.

Ex. 207 = the system will printout on the teletype the decimal value of the location 207 in the computer memory.

ii) Executing Routines From the Teletype

System Genie is capable of executing twenty-six different routines through the teletype. However, twenty-six routines are not the maximum number of routines which the system can handle. The number of routines executable from the teletype can be expanded to any number, if desired. Because of the core memory limitation, it is not possible for the operator to execute routines by their names, but rather by their codes only. The twenty-six executable routines take on the code A through Z for simplicity. Thus, whenever the operator types A# on the teletype, the system will execute the routine asigned to code A. There is a certain amount of flexibility built into the system as far as adding, replacing, and/or deleting routines with respect to their assigned codes.

At present, there are only twenty-two routines which have assigned codes to them, and the rest of the codes are free for future assignments. It is appropriate at this time to call the routine with an assigned code to be a teletype command routine. The following is a list of teletype command routines which are executable from the teletype

CODE	NAME OF ROUTINE	MNEMONIC NAME
A	INITIALIZE SYSTEM CONTROL FLAGS AND COUNTERS	ISCFC
B	X-Y STAGE AUTOMATIC REFERENCE CHECK ROUTINE	XYARC
C	PERISCOPE REFERENCE AND R-THETA CHANNEL CHECK	PDTR
D	ID REQUEST ROUTINE	IDRR
E	LOCATE MRQ FROM RFB INFORMATION	LMRQR

CODE	NAME OF ROUTINE	MNEMONIC NAME
F	DATA TAPE MOUNT ROUTINE	DTMR
G	MANUAL FIDUCIAL MEASUREMENT ROUTINE	MFMR
H	SPIRAL READER CONTROL SEQUENCE	SRCS
I	OPERATOR SIGN IN ROUTINE	OSIRR
J		
K		
L		
M	REMEASURE PRESENT EVENT ROUTINE	RPER
N	REMEASURE EVENT FROM DESIRED RFB INFORMATION	RERFB
O	OPERATOR SIGN OFF ROUTINE	OSFRR
P	DATA TAPE RELEASE ROUTINE	DTRR
Q	UNLOCK X-Y STAGE	UXYSR
R	SYSTEM RECOVERY ROUTINE	SCSRR
S	STAGE COORDINATE MEASUREMENT ROUTINE	SCMR
T	IPD COORDINATE MEASUREMENT ROUTINE	IPCMR
U	MAGTAPE 1 TEST ROUTINE	MT1TR
V	MAGTAPE 2 TEST ROUTINE	MT2TR
W	AUTOMATIC FILM DRIVE REQUEST ROUTINE	AFDRR
X	X-Y STAGE MANUAL AND DISPLAY SCALER	XYSTR4
Y	PERISCOPE MANUAL CALIBRATE ROUTINE	PMRC
Z	MANUAL FILM DRIVE ROUTINE	FLMAN

iii) Functional Description of the Teletype Command Routine

A. System check-up routines

a) A# ⇒ initialize system control flags and counters

Hardware-wise routine A resets all the external device status registers. It sets periscope, X-Y stage and film drives to automatic mode. It enables the interrupt flip-flop on console button control, magnetic tape interface, R-theta channel, periscope, X-Y stage, film drives, auto-fiducial, and image plane digitizer. It enables the encoder counts from the periscope and the X-Y stage to come into their respective scalers inside the computer memory. Finally it turns on the realtime clock, and enables the API and PI facility in the computer.

Software-wise routine A clears the job slots and reinitializes the pointer in the Job Supervisor routine. It reinitializes pointers and resets flags in the Job

Executor routine, teletype routines, keyboard routines, and magtape routine. Finally it sets the interrupt service requirements for the console button control.

b) B# ⇒ X-Y stage automatic reference check routine

Routine B performs two functions. The first function it performs is to check all four primary limit-switches on the stage. The second function it performs is to check the X and Y scaler reference. Figure 2-2 shows the limit-switch boundaries drawn in solid lines, and reference boundaries drawn in dotted lines. Routine B drives the stage at a fixed velocity towards XLS1 boundary until the stage hits the XLS1 limit-switch. The system senses the XLS1 limit interrupt and returns the control to the routine which then drives the stage at a fixed velocity toward YLS1 boundary until the stage hits the YLS1 limit switch. The process repeats itself for XLS2 and YLS2 limit-switches. The reference is always sensed before the stages hit XLS2 and YLS2 limit-switches. Whenever the reference interrupt is sensed by the system, the system will always compare the value of the scaler to some reference value to determine whether or not the scaler is out of tolerance (± 2μ), and then it sets the scaler back to the reference value.

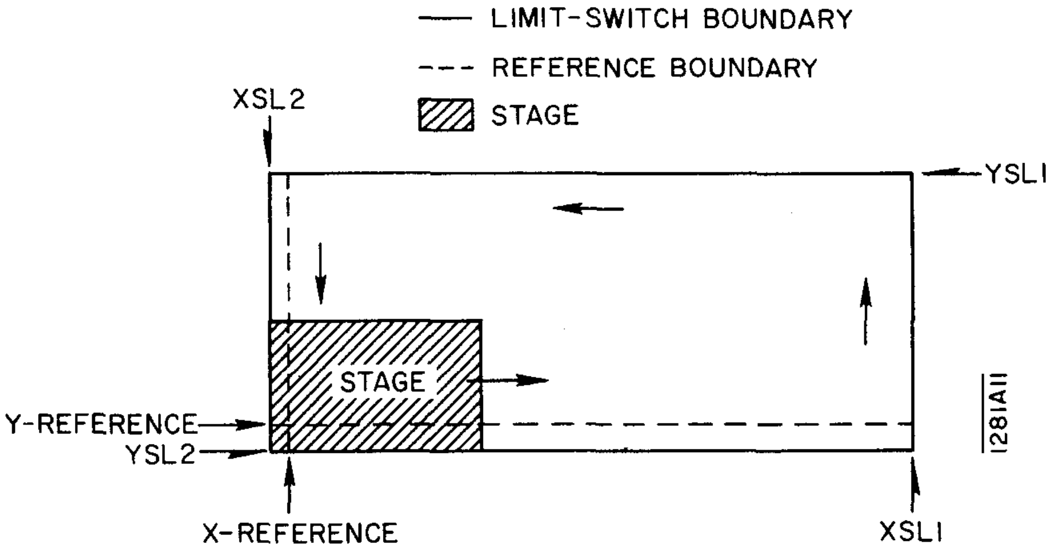


FIG. 2-2--X-Y stage boundaries.

c) C# ⇒ periscope reference and R-theta channel check

Routine C is a dual purpose routine. The primary purpose of routine C is to check the periscope reference and the R-theta channel (R-angle high speed data channel). The secondary purpose of routine C is to record digitized calibration pattern on magnetic tape. The primary function of the periscope reference test is to make sure that the periscope generates 33 references interrupts on its way out and 33 references interrupts on its way in. When the test is successful, the periscope scaler will be displayed on the computer console, and this assures the operator that the periscope is homed properly. Routine C should not be repeated until the periscope is completely homed or when computer console displays the periscope scaler. While the periscope is moving out, the R-theta channel should be activated, meaning, digitized data should be coming through the R-theta channel and being displayed on the Tektronix 611 storage scope. Once the scan is completed the data displayed on the storage scope should be interpreted. If the data make physical sense, then R-theta is working properly. Otherwise, further tests have to be made on the R-theta channel.

B. System initialization routines

a) D# ⇒ ID request routine

Routine D requests the operator for information such as bubble chamber size, operator number, year, month, date, and time of the day. When this information is stored properly in a reserved section for ID information, the system will then try to read the MRQ file content on magtape drive #1. When the MRQ file content is read successfully the system types out MRQM. Otherwise, the system will type out a message to the operator alerting him that something has to be done. Unless the system is damaged by an irrecoverable operational error or by external device malfunction, routine D has to be repeated only once each day.

b) E# ⇒ locate MRQ from RFB information

Routine E allows the operator to search any desired roll, frame, and beamtrack number stored on the MRQ tape. The operator will be notified whether or not the event specified by its roll, frame, and beamtrack can be found. Routine E is executable from the teletype only if it is preceded by a successful execution of routine D. Once the MRQ file content is read successfully by routine D, then routine E can be executed anytime the operator feels that it is necessary (for instance, when the operator changes to a roll of film whose roll number does not follow the roll of film that was measured previously, or whenever it is necessary for him to skip several frames on film).

c) G# ⇒ manual fiducial measurement routine

The purpose of routine G is to allow the operator to tell the system which set of fiducials he wants the system to help him to locate during event measuring time. The initial six measurements have to be performed completely manually with the stage-ball and console button #4. Routine G starts off by requesting the operator to measure V1UF (view 1 upper fiducial). The operator moves the stage to first fiducial in view 1, then he pushes button number 4 to record the X-Y coordinate point of the fiducial. Once the system has recorded the position of the fiducial, routine G then requests the operator to measure V1LF (view 1 lower fiducial). The measuring process repeats itself for V2UF, V2LF, V3UF, and V3LF. When all six measurements are completed, the system then moves the stage to each of the fiducials measured previously for confirmation. The system moves on to the next fiducial only when the operator pushes console button #4. After all six fiducials are completely confirmed, the system then moves the stage back to view 2 center to indicate to the operator that routine G is completed.

C. Operator sign-IN/OFF routines

a) I# ⇒ operator sign-in routine

Routine I requests the operator for his number. Once the operator number is recorded in the ID buffer stored in the computer memory, the system then types out the time of the day, and also the roll and frame number that the operator should be measuring next. Assuming routine D was done previously during the day, whenever a different operator signs in, routine I changes only the location concerning the operator number in the ID buffer, and nothing else gets changed in the ID buffer.

b) O# ⇒ operator sign-out routine

Routine O tells the operator the time when he signed off. It also tells him how many events he measured (NEM), how many events he rejected (NER), and also the actual measuring time (AMT) it took him to measure those events. Furthermore, routine O tells the operator the total number of events measured (TNEM), the total number of events rejected (TNER), and the total actual measuring time up to and including that of the operator's own. The acutal measuring time only measures operators' efficiency, but it does not include film transport time, or view changing time.

D. Date tape ON/OFF routine

a) F# ⇒ data tape mount routine

Routine F requests the operator to assign a number to the tape on which the digitized data is to be stored. Once the number is loaded into computer memory, the system types out the time of the day so the operator knows exactly when he has loaded the tape. Routine F must be requested by the operator every time when he needs to replace another data tape. The frequency at which the data tape needs replacement depends on the average rate of events that can be measured per hour.

b) P# ⇒ data tape release routine

Routine P generates an end-of-file (EOF) at the end of the digitized data and rewinds the tape. Furthermore, it also types out which operator released that tape, tape number, total number of measured events stored on the data tape, total number of ejected events stored on the data tape, and finally the time when the data tape was released. Routine P must be requested by the operator every time he has to replace the data tape. Otherwise, the EOF will never be generated and this might present problems to the IBM 360 when it tries to process the data tape. Furthermore, all the useful bookkeeping information could be lost.

E. System recovery routines

a) M# ⇒ remeasure present event routine

Routine M allows the operator to remeasure the event which he has just measured. This routine backs up the data tape, so on the next measurement the system will write over the data that is in error. Once the tape is properly backed up, that is when the information like view number, beamtrack number, and vertex number in the ID buffer matchup one-to-one on what was read back from the tape, the system types out the message "REMEASURE EVENT." The system also moves the stages to approximately where it thinks the vertex should be. Routine M is only executable after the operator has pushed the vertex button and before he pushes the advance button. Because once the advance button is pushed, the system is then instructed to move on to the next vertex, or to the next view. So the previous information in the ID buffer and the location of the vertex have completely been changed, and it is impossible for the system to recover.

b) N# ⇒ remeasure events from desired RFB information

Routine N allows the operator to start remeasuring everything from view 2. This is necessary because sometimes the operator might measure the events in view 2 in one way, and say in view 1 he got mixed-up in his measuring sequence, the procedure is such that he has to remeasure everything from the starting point in view 2. The controlling sequence in the system for measuring successive vertices and events is rather complicated, and rather than messing-up the entire measuring sequence it would be much easier just to have the operator to start all over again. Routine N requires routine E to be successfully executed. This is necessary because routine E requests roll, frame, and beamtrack (RFB) number from the operator, and routine N needs RFB information from the MRQ buffer. Routine N backs up the data tape until the RFB information read-back from the tape matches the RFB information stored in the MRQ buffer. Then the system stops the tape search and types out the message "REMEASURE EVENT." When the operator sees this message he should execute routine H to start the measuring sequence again.

c) R# ⇒ system recover routine

Routine R allows the operator to recover from errors that he made accidentally. For instance, the system makes a request, and the operator enters the wrong information. He can make a recovery from his error provided if he did not type "$" on the teletype; otherwise he has to start all over again from the beginning.

EX.

```
Y  = 67/
R#
Y  = 69$
M  =
```

System request year information, operator made an error. He hits "/" on the teletype and does a routine R. System recovers, operator enters correct information. System continues.

Furthermore, whenever magtape drives are not in remote control the system types out "MT NOT READY," and the operator can recover from this error by using routine R. In this case, routine R simply allows the system to continue on with the routine that it was executing before it detected the error. Routine R can also be used to help the operator to bring him back to the proper view and event, after, for some reason or another, he got sidetracked and he forgot what he was supposed to do. In Routine S, X-Y stage coordinate measurement routine, the operator can remeasure a previously measured X-Y coordinate simply by requesting routine R. In this case, the system simply moves the stage back to the last measured X-Y coordinate. The operator can backup as many X-Y coordinates as he wishes, simply by requesting routine R repeatedly from the teletype.

d) Q# ⇒ unlock X-Y stage routine

Routine Q allows the operator to free the X-Y stage whenever the stage is locked in the automatic mode and the console buttons are disabled. This case can happen whenever the system is requested to move the stage, and usually happens after the system has moved the stage to the target. For each stage lock-up only one request for routine Q should be made, and repeated requests of routine Q would not serve any purpose.

F. Spiral reader control sequence

a) H# ⇒ spiral reader control sequence.

Routine H is the control sequence which directs the operator what to do and also keeps track of what he does during the measurements. Once this routine is executed from the teletype, communication between the operator and the system is done through the console buttons. Routine H through the system is capable of performing the following functions:

1) It directs the measuring sequence from VIEW2, to VIEW1, and to VIEW3.

2) In each view, it starts off by requesting the operator to measure the upper and the lower fiducials. This is accomplished by directing the stage to move the fiducial mark into the TV screen. The operator responds by pushing the manual fiducial button.

3) After the fiducial measurements, it directs the system to read the MRQ tape only if it is directing measurements in VIEW2; otherwise, it directs the system to check the vertex buffer if it is directing measurements in VIEW1, or VIEW3. In view 2, after the MRQ reading, it moves the stage to approximately where the vertex lies. The operator sometimes has to look on the measuring table to find the vertex and bring the vertex into the TV screen with the stage-ball to perform the final adjustment, before he pushes the vertex button to measure that event. However, in view 1 and view 3, the system usually moves the vertex to the right into the center of the TV screen, so the operator just has to perform the final adjustment and pushes the vertex button to measure.

4) Routine H is capable of storing up to four vertices (one primary vertex, and 3 secondary vertices) per event. Furthermore, it allows the operator to measure up to four events in each view without having the operator to measure fiducials again. Hence, the system is really capable of storing up to 16 vertices before it moves on to the next view. However, if there are move than four events per view, routine H directs the operator to measure all four events in view 2 first, and then it directs him to measure the events in the rest of the two views. Once the measuring sequence is finished, the system will return to view 2 to direct the operator to remeasure the fiducials and to finish the rest of events in view 2, then VIEW1, and VIEW3. There is really no limit to the number of events that can be measured in each frame, because the process always repeats itself after every four events.

5) After each spiral scan, the system allows the operator to measure crutch points. There is a maximum number of crutch points per vertex that the operator can measure, and the number is 50. Measuring a crutch point is accomplished by moving the stage manually to a point on the track and pushing the crutch point button. Crutch points may be measured anytime after every vertex measurement if the operator feels that it is necessary to do so. Furthermore the operator is free to measure a crutch point in one view and not in the other two views.

6) After each spiral scan, the system also allows the operator to measure stopping-track crutch points. The maximum number of stopping-track crutch points that can be measured and remembered by the system is 28 points per view. This would really allow something like 7 stopping-track crutch points per event. Measuring a stopping-track crutch point is accomplished by moving the stage manually to the end of the stopping track and pushing the crutch point button twice and flagged crutch point button once. Stopping-track crutch points are mandatory crutch points, and they must be measured in all three views. The order in which they have to be measured is important to the system; otherwise some of these stopping-track crutch points could be lost. In order to take the load off the operator's mind, the system remembers those stopping-track points for him, provided those points are measured in view 2 to begin with. In view 1 and view 3, right after each spiral scan, the system moves the stage to approximately where the stopping track should be; the operator then pushes the crutch point twice and the flagged crutch point button once. If there is another stopping track in the same event, the system moves the stage to that point; otherwise the system permits the operator to push the advance button.

7) After the system has moved the stage to where it thinks the vertex should be, due to one reason or another, the event is impossible to measure and the operator has to reject that event. To reject an event the operator has to push the reject button, and the system will request the operator to supply the reject code. Once the system records the reject code in the ID buffer, it puts the ID buffer out onto the magtape, and it moves the stage to the next event. Furthermore, if none of the events is measurable in view 2, and the operator has already rejected every single one of them, the system instead of moving to view 1, will just move on to the next frame. However, if none of the events is measurable in view 1, and the operator has already rejected

every single one of them, the system will still move to view 3. In this case, the operator would only have to measure the fiducials in that view, and the system will move on to the next frame. Finally, if none of the events is measurable in view 3 and the operator has rejected them all, the system will just move on to the next frame.

8) Finally, at the end of each measuring sequence (VIEW2, VIEW1, and VIEW3), routine H tells the operator how long it took him to measure all the events, and also tells him which roll and frame number he should be measuring next. Simultaneously the system advances the film to the proper frame number and moves the stage to VIEW2 upper fiducial to begin the next measuring sequence. Furthermore the film drive is in manual mode, so the operator can adjust the film position should the film be out of registration. Throughout the entire measuring sequence, other teletype command routines like A, B, E, M, N, O, I, R, P, W, Y, and Z are executable from the teletype. A list of possible operator errors, system errors will be mentioned in Section III.

G. Calibration measurement routines

a) S# ⇒ stage coordinate measurement routine

Routine S allows the operator to measure the X-Y stage coordinate of any calibration pattern. The maximum number of points that can be measured is 100. However, this number can be expanded very easily. Initially, the operator has to measure all the points manually by moving the stage to each point and to push coordinate measurement button (console button 9) to record the X-Y position. When the operator has measured enough points, he pushes the advance button, the system writes the measured data on tape and proceeds to move the stage to the first point. From then on, all the operator has to do is to make the final adjustment and to push the coordinate measurement button; likewise, all the system does is to record the X-Yposition andto move the stage to the next point. When the last point is recorded, the system does not move the stage to anywhere, and the coordinate button is disabled. At this time, the operator should realize he has finished measuring all the points, and he should push advance button to write the measured data out onto the tape. Each time the advance button is pushed, the system types out a number indicating the number of records that are written on the tape. Furthermore, whenever the operator feels that he has to remeasure a previously measured X-Y coordinate, he can do so by requesting routine R.

b) T# => IPD coordinate measurement routine

Routine T allows the operator to measure the image plane digitizer coordinate of any calibration pattern. The maximum number of points that can be measured is 100. However, this number can be expanded very easily. In this routine, every point has to be measured manually, and every point is recorded by pushing IPD button (console button 6). After the operator has finished each set of measurements, all he has to do is push the advance button and the system records the measured data onto the magtape. Each time when a record gets written out onto the tape, the system types out a number indicating the number of records that are stored on the magtape.

H. Miscellaneous routines

a) U# => magtape 1 test routine

Routine U allows the operator to checkout the performance of magnetic tape drive 1 and its interface electronics. This routine uses every instruction that was designed for the magtape. Basically the routine does 10 file gaps and 10 back spaces; then it writes short records of all $(7)_8$, $(6)_8$, $(5)_8$, $(4)_8$, $(3)_8$, $(2)_8$, $(1)_8$, and $(0)_8$. To see if the data is written on the tape correctly, the best way is to try to read it back and see. The first record should be $(7)_8$, then once the operator hits the continue button on the computer console, the routine rewinds the tape and displays the second record which should be $(6)_8$, and the process repeats itself for the rest of the patterns. Once that part of the program is checked out, the routine starts backing the tape to check the backspacing. The other part of routine U is used for maintenance of the magtape drive #1. For instance, routine U is capable of writing fixed record length of 512 words continuously. Furthermore, routine U also allows the operator to read backwards continuously.

b) V# => magtape 2 test routine

Routine V is identical to routine U in every respect, except it is for magnetic tape drive #2.

c) W# => automatic film drive request routine

Routine W allows the operator to transport film on any or all of the film drives. It also allows the operator to either advance or reverse film drive via the teletype. If a positive number is entered, this means advance film, or if a negative number is entered, this means reverse film. However, if zero is entered, this means do not move the film on that drive. Routine W may be executed during the measuring sequence.

d) X# ⇒ X-Y stage manual and display scaler

Routine X is used mostly for maintenance of the X-Y stage scalers or their encoders. This routine allows the operator to see if he can repeatedly come back to the same position without losing or gaining any stage counts. This is achieved with the help of the displayed scalers on the computer console.

e) Y# ⇒ periscope manual calibrate routine

Routine Y allows the operator to preset the periscope scaler to a certain value from the teletype. Unless the periscope is malfunctioning, it should always come back to the same location after a spiral scan. The system makes a check on the periscope scaler before each spiral scan, to ensure that each spiral scan does begin before the first reference. Routine Y may be executed during the measuring sequence.

f) Z# ⇒ manual film drive routine

Routine Z puts the film drives to manual mode from automatic mode, thus allowing the operator to transport the film manually. To transport film manually the operator has to push one or all three view buttons (V1, V2, V3) and the joystick. This routine is executable during the measuring sequence.

II. Console Button Communication

i) Description of the Console Button Layout

The secondary communication link between the operator and the system is through the console button control. The console button control is a one-way communication link to the system, so each control button can only request the system to perform a specific task. Figure 2-3 shows the console button layout and the abbreviated name for each button.

V1 (1)	V2 (2)	V3 (3)
MF (4)	VTX (5)	IPD (6)
RJT (7)	FCP (8)	CP (9)
	ADV (10)	

FIG. 2-3--Console button layout.

The following is a list of abbreviated names for the console buttons and their corresponding nonabbreviated names.

1) V1 = View 1 button,
2) V2 = View 2 button,
3) V3 = View 3 button,
4) MF = Manual fiducial button,
5) VTX = Vertex button,
6) IPD = Image plane digitizer button,
7) RJT = Event reject button,
8) FCP = Flagged crutch point button,
9) CP = Crutch point or coordinate measurement button,
10) ADV = Advance button.

Since the console button control is a one-way communication link to the system, the only way for the operator to be sure that the system has received the request is to have the system acknowledged. For V1, V2, and V3 the system acknowledges the operator's request by simply moving the stage to the corresponding view. For the rest of the console buttons, the system always acknowledges the operator's request by ringing the teletype bell.

ii) Functional Description of Each Console Button

A. View buttons

View 1, view 2, and view 3 perform similar functions except in different views only. The main function of the view buttons is to allow the operator to select different views through stage movements. Furthermore, whenever the view button is pushed, the system always moves the stage to the last position. This last position could be either a predetermined point, or an operator measured point. This function is quite useful whenever the operator for one reason or another forgot where the stage was before he moved the stage away manually. The view buttons are usually enabled as long as the system is in operation. The only time when the views are disabled is during every spiral scan, that is, when the stage must be locked up both in the manual mode and in the automatic mode.

B. Manual fiducial button

The manual fiducial button allows the operator to record the X-Y coordinate of the fiducials, and it also directs the system as to what to do after each fiducial measurement. During manual fiducial measurement (routine G), this button records the X-Y coordinate of the fiducials, and directs the system to type out V1LF, V2UF, V2LF, V3UF, and V3LF messages only if the X-Y coordinate is successfully recorded. During event measurement sequence (routine H), a successful recording of the X-Y coordinate of a fiducial point is indicated by a stage movement from the upper fiducial to the lower fiducial, and from the lower fiducial to the vertex. Suppose somewhere along the measuring sequence, the operator has made a bad measurement for the upper fiducial; the system records the point and moves the stage toward the lower fiducial. There he makes a good measurement, but the system performs a computation and discovers the fiducial distance is out of tolerance; it therefore moves the stage back to the upper fiducial and types out "REMEASURE FIDUCIALS" message. At this point, the operator has to remeasure the fiducials before he can go on with the other measurements. Manual fiducial buttons are enabled only during the fiducial measurement time; otherwise, this button should always be in the disabled mode.

C. Vertex button

The vertex button allows the operator to record the X-Y coordinate of the vertex point and directs the system to perform a spiral scan. The spiral scan is activated only after the X-Y coordinate of the vertex is successfully measured. During the spiral scan the following things are taking place inside the computer:

a) The periscope is moving and references are being generated every revolution. The computer has to keep track of the periscope references in order to change the periscope's velocity.

b) Periscope encoder counts are coming into the scaler inside the computer memory. The periscope scaler keeps track of the periscope position, and it is important to make sure that the periscope always starts moving somewhere behind the optical zero. This position is always checked before each spiral scan.

c) R-theta channel is activated, and the data is coming into the computer through the direct memory access facility (DMA).

d) R-theta channel interrupts have to be serviced by the computer.

e) The system is busy transferring the digitized data from its output buffer onto the tape and clearing the output buffer with ones.

f) The realtime clock is still keeping the time down to 1/60 sec and interrupts the computer every second.

While the spiral scan is in progress the system is using approximately 60 to 90 percent of the computer's time, depending on the type of event that is being digitized. Furthermore, during the spiral scan the system locks up the stage, and disables the entire console control. Only right after the spiral scan, when the periscope is heading for home is the console button activated and the stage freed to be moved manually. While the periscope is heading for home, the system is also busy performing computations for the next vertex position.

D. Image plane digitizer button

The image plane digitizer allows the operator to record the image plane coordinates. This button is enabled only during the measuring sequence (routine H), or during IPD calibration measurement (routine T).

E. Event reject button

The event reject button allows the operator to enter a code from the teletype as to why he is rejecting the event. Once the code is successfully recorded into the ID buffer, the system then writes that ID buffer out onto the tape and moves the stage to the next event. The system also keeps track which event and in what view when the operator made the reject. So the same event will not be presented in the next two views if the event is rejected in view 2; or in view 3 if the event is rejected in view 1. The event reject button is enabled only during the vertex measuring time and disabled after the spiral scan is completed.

F. Flagged crutch point button

The flagged crutch point button allows the operator to record any X-Y point or points to be remembered by the system following a spiral scan. The decision has to be made by the operator while he is measuring a vertex in view 2. Once all the flagged crutch points are recorded in view 2, the system will be responsible to remember how many and the locations of those points after each spiral scan in view 1 and view 3. To accomplish what was discussed before, the system has to finish a spiral scan on an event in view 2, and the operator moves the stage manually out to the point of interest and pushes this button. If there are other points of interest he wants, he can record them all before pushing the advance button to go on to the next event. This same procedure can be repeated for the next event, if the operator is required to do so. However, if all the events are measured the system moves the stage to the next view and finishes the spiral scan on the event; instead of going onto the next event it moves the stage to the first flagged crutch point. When this happens the operator cannot ignore measuring that point, since the system does not permit him to advance to the next event. Consequently, the operator has to measure all the flagged crutch points by pushing this button, and all points are measured only when the stage does not move to the next point after this button is pushed. It is only then the system allows the operator to advance to the next event. The flagged crutch point button is used extensively in measuring stopping-track events. The maximum number of flagged crutch points the system can remember is 28 points per view, or approximately 7 flagged crutch points per event. The flagged crutch point button is enabled after each spiral scan and disabled after the advance button is hit.

G. Crutch point button

The crutch point button allows the operator to record the X-Y coordinate of any track point to keep the filtering program "POOH." The operator has to push this button once to record a crutch point, and twice to signify stopping-track. This button can be used at anytime after a spiral scan and is not useable after the advance button is hit. Furthermore, this button is used to record the X-Y coordinate in the stage calibration measurement routine (routine S).

H. Advance button

The advance button allows the operator to tell the system to move on to the next event, view, or frame. This button also checks to see whether IPD coordinates or crutch points, or neither, has been measured. If either one of the two types has been measured, the system writes out the data onto the tape. Otherwise, the system determines whether a flagged crutch point has to be measured. If there is, the system moves the stage right to that point, and processes all the flagged crutch points before it allows the operator to hit the advance button in order to move on to the next event. The advance button is enabled after each spiral scan and is disabled by itself.

III. System Message and Their Meaning

There are four types of messages which the system can type out on the teletype. The four types of messages are machine malfunction, system operation, system information request, and system bookkeeping messages. Machine malfunction messages deal with hardware operation conditions. System operation messages allow the operator to know whether the system has successfully executed the routine or not. System information request messages are messages that require the operator to return the necessary numerical data which the system must have. Finally, system bookkeeping messages deal with operators' efficiency and number of events measured or rejected.

The routine in which the message appeared will be designated by " routine code ." For example, if a message occurred during routine H, it will be noted as (H).

i) Machine Malfunction Messages

A. Magnetic tape drive

a) MT1-INTERRUPT = An unidentified interrupt occurred while magtape #1 is in the process of doing a read or a write instruction.

Operator's Response (O. R.): A# and repeat the routine that caused this error. If error repeats, notify the maintenance.

b) MT2-INTERRUPT = An unidentified interrupt occurred while magtapė #2 is in the process of doing a read or a write instruction.

O. R.: Same as (i-A-a).

c) MT1-WC TOO SMALL = The length of record read off the magtape #1 is longer than the input buffer size.

O. R.: Increase the input buffer size or determine why the record is longer than the specified buffer size; then repeat routine.

d) MT2-WC TOO SMALL = The length of record read off the magtape #2 is longer than the input buffer size.

O. R.: Same as (i-A-c).

e) MT1-EOT = Beginning of tape or of the tape condition on magtape #1 was reached.

O. R.: If beginning of tape condition do routine D. If end-of-tape condition do REWIND in local mode.

f) MT2-EOT = Beginning of tape or of the tape condition on magtape #2 was reached.

O. R.: If beginning of tape condition do routine F. If end-of-tape condition do REWIND in local mode.

g) MT1-PARITY = A parity is detected during a read instruction on magtape #1.

O. R.: Repeat the routine or routines that caused this error. If error repeats, notify maintenance.

h) MT2-PARITY = A parity is detected during a read instruction on magtape #2.

<u>O.R.</u>: Same as (i-A-g).

B. <u>Radius—angle high speed data channel</u>

a) DATA RATE EXCEEDED = Data are being digitized faster than they can be written out on the tape.

<u>O.R.</u>: (C), repeat C#.

(H), do M#.

If the error persists after several spiral scans, notify the maintenance.

b) ANGLE ERROR—Two or more counts off the baldwin disc somehow got lost during a spiral scan.

<u>O.R.</u>: Same as (i-B-a).

C. <u>Periscope drive</u>

a) CALIBRATE P—SCALER = One of four things could have happened.

1) Periscope did not get homed properly from the previous spiral scan.

<u>O.R.</u>: (C) do Y#, /, and repeat C#.

(H) do Y#, /, and R#.

If the error persists after several spiral scans, the following things could also be causing this message.

2) The periscope is setting at some position other than its home position. This is caused by manual pot for the periscope being set incorrectly.

<u>O.R.</u>: Call maintenance to check the pot setting, and repeat (i-C-a-1).

3) There should be 33 reference counts per spiral scan. If for some reason, one or more references got lost while the periscope is moved out, the periscope will hit the outer limit switch.

<u>O.R.</u>: Same as (i-C-a-1).

4) Periscope counts to the scaler, if not properly adjusted, can cause the periscope scaler to lose or gain counts. When this happens the periscope will generally lose control and crash into the limit switches.

<u>O.R.</u>: Call maintenance to check periscope scaler adjustment. Repeat (i-E-a-1).

D. X-Y stage drive

a) X-SCALER REF. OFF = Two or more counts got lost or gained while the X-stage was moving, and the system tells the operator exactly how many it was off. It is expected that every time that the system is initially loaded into the computer, the X-stage scaler will be off, and this is why routine B has to be executed first. Because of power supply drifts, which might affect the lamps inside the encoders, it is necessary to have the operator to check the X-scaler regularly. This is accomplished by moving the X-stage manually toward the XSL (left) boundary as shown in Fig. 2-2. Because each time the X-stage crosses the X-reference boundary in the manual mode, the system automatically checks the X-scaler to see if it is out of tolerance, and presets the X-scaler to (100000_8).

O. R. : (X-Y stage in manual mode), push any view button and bring X-stage towards its left boundary.

(B), repeat B#.

If the error persists, notify the maintenance to check the X-scaler adjustments and power supply drift.

b) Y-SCALER REF. OFF = Similar to (i-D-a), except everything applies to Y-stage.

O. R. : (X-Y stage in manual mode), push any view button and bring Y-stage towards its lower boundary.

(B), repeat B#.

If the error persists, notify the maintenance to check Y-scaler adjustments and power supply drift.

c) CALIBRATE X-SCALER = While in automatic mode, the X-stage hits one of its primary limit switches, and as a consequence, the X-scaler gets wiped out. Any further attempt to move the stage either through the view buttons, or through the system could cause the stage to get out of control, and this message to be repeated. One of four things could have happened:

1) An attempt was made by the system to drive the stage out of its limits because an error was made on the MRQ tape.

O. R. : (H), B# twice, E#, (reject button), A#, B#, R#.

2) X-scaler is losing or gaining counts rapidly due to unstable power supply or badly adjusted X-scaler which can cause the X-stage to get out of control.

O.R.: (View button control), A#, B# twice and any view button.
(G), A#, B# twice, and G#.
(H), A#, B# twice, Q#, and manually move the stage to the desired position.
If the error persists, notify the maintenance to check the X-scaler adjustments and the power supply drift.

3) For some reason the system could get wiped out, in which case it is necessary to reload the system and start from the beginning.

O.R.: Reload the system and do A#, B#, D#, E#, G#, N#, H#.

4) X-stage could be losing overflow interrupts, which can cause the X-stage to get out of control. This is the last thing to suspect.

O.R.: Same as (i-D-c-2), except the maintenance has to check overflow interrupts.

d) CALIBRATE Y-SCALER = Similar to (i-D-c), except everything applies to Y-stage.

O.R.: Same as (i-D-c-1, 2, 3, 4).

E. Film drive

No messages.

F. Console button control

No messages.

G. Image plane digitizer (not used presently)

a) IPD NOT HOMED

b) IPD LT-SCALER OFF

c) IPD RT-SCALER OFF

H. Auto-fiducial (not in the system yet)

ii) System Operation Messages

A. System acknowledgement messages

a) SYSTEM GENIE = System is initialized and ready to accept teletype command routine.

b) MRQM = MRQ file content is properly read into the computer and the operator may begin routine E.

c) REQ'D RFB FOUND = Requested roll, frame, and beamtrack number were located successfully on the MRQ tape.

d) REQ'D RFB MISSING = Requested roll, frame, and beamtrack number cannot be located on the MRQ tape. During routine H the last event listed on the MRQ was reached. Operator should verify.

e) DTMD = Data tape is properly mounted and the time when it got mounted is also typed out on the teletype.

f) DTRD = Data tape is released and the time when it got released is also typed out on the teletype.

B. Messages that require the operator's cooperation

a) ILLEGAL REQUEST = The operator has made an error while making a request or trying to enter a numerical value with the wrong teletype command code.

O.R. : (D), or (E), or (F), or (H), do R#.

b) MT NOT READY = One or both magnetic tape drives are not in remote control.

O.R. : (E), or (F), or (H), operator should set magtape drives to remote and do R#.

c) REMEASURE FIDUCIALS = One or both fiducial measurements were measured incorrectly.

O.R. : (G), or (H), remeasure the fiducials.

d) REMEASURE EVENT = The system has successfully backed up the data tape, so it is ready for the operator to remeasure the event.

O.R. : (M), push vertex button to remeasure.
(N), do H#.

e) C-P BUFFER FULL = Maximum number of crutch points was exceeded, or the maximum number of coordinate points was exceeded, and the last coordinate point was not recorded.

O.R. : (H), (S), (T), push advance button to write data points out onto the tape. Do not attempt to exceed the maximum number on the next try.

f) FCP BUFFER FULL = Maximum number of flagged crutch points was exceeded. The last flagged crutch point was not recorded.

O.R. : (H), push advance button. Do not attempt to exceed the maximum number on the next try.

g. RELOAD FILM = The system has detected a change in the roll number on the MRQ, and it also requests the operator to reload the proper roll of film on the film transport.

   O. R. : (H), rewind film automatically or manually, mount the proper roll of film, locate the proper frame number, do H #.

iii) System Information Request Messages

A. Non-numerical requests

   a) V1UF = Measure view 1 upper fiducial.

      O. R. : (G), move stage manually to view 1 upper fiducial and push manual fiducial button.

   b) V1LF = Measure view 1 lower fiducial.

      O. R. : Same as (iii-A-a), except for lower fiducial.

   c) V2UF = Measure view 2 upper fiducial.

      O. R. : Same as (iii-A-a), except for view 2.

   d) V2LF = Measure view 2 lower fiducial.

      O. R. : Same as (iii-A-c), except for lower fiducial.

   e) V3UF = Measure view 3 upper fiducial.

      O. R. : Same as (iii-A-a), except for view 3.

   f) V3LF = Measure view 3 lower fiducial.

      O. R. : Same as (iii-A-e), except for lower fiducial.

B. Numerical requests

   a) C = = Chamber size

      O. R. : (D), type in numerical value.

   b) O = = Operator number

      O. R. : Same as (iii-B-a).

   c) Y = = Year information

      O. R. : Same as (iii-B-a).

   d) M = = Month information

      O. R. : Same as (iii-B-a).

   e) D = = Date information

      O. R. : Same as (iii-B-a).

   f) S = = Spiral number

      O. R. : Same as (iii-B-a).

   g) H = = Hour of the day information

      O. R. : Same as (iii-B-a).

h) M = = Minute information

<u>O. R.</u> : Same as (iii-B-a).

i) R = = Roll information

<u>O. R.</u> : (E), type numerical value.

j) F = = Frame information

<u>O. R.</u> : Same as (iii-B-i).

k) B = = Beamtrack information

<u>O. R.</u> : Same as (iii-B-i).

l) RC = = Reject code information

<u>O. R.</u> : (H), type numerical value.

m) NF = = Number of frames to transported information

<u>O. R.</u> : (W), type numerical value.

iv) <u>System Bookkeeping Messages</u>

a) R = = Roll number that the operator should look for next.

<u>O. R.</u> : (H), (I), no operator's response required.

b) F = = Frame number that the operator should look for next.

<u>O. R.</u> : Same as (iv-a).

c) AMT = = Actual measuring time taken by the operator to measure an event, or a set of events.

<u>O. R.</u> : (H), (O), no operator's response required.

d) TIME ON = = Time when the operator signed in.

<u>O. R.</u> : (I), no operator's response required.

e) TIME OFF = = Time when the operator signed off.

<u>O. R.</u> : (O), no operator's response required.

f) NEM = = Number of events measured by the operator.

<u>O. R.</u> : Same as (iv-e).

g) NER = = Number of events rejected by the operator.

<u>O. R.</u> : Same as (iv-e).

h) TNEM = = Total number of events measured by several operators.

<u>O. R.</u> : Same as (iv-e).

i) TNER = = Total number of events rejected by several operators.

<u>O. R.</u> : Same as (iv-e).

j) TAMT = = Total actual measuring time by several operators.

<u>O. R.</u> : Same as (iv-e).

k) MEDT = = Total measured events stored on data tape.

O.R. : (P), no operator's response required.

l) REDT = = Total rejected events stored on data tape.

O.R. : Same as (iv-k).

IV. System Start-Up and Error Recovery Procedures

i) System Start-Up Procedure

The following procedure must be followed after loading or reloading the system. Look up the section called "Initialize System Control Flags and Counters" in the listing to find the starting address of the system labeled as "START." Load the start address into the address on the console and push the start button. At this point the system will type out "SYSTEM GENIE" and the operator is ready for the start-up procedure. The operator should execute the following teletype command routine in that exact order:

A#	Clear system flags.
B#	Calibrate X-Y scaler.
C#	Check periscope and R-theta channel.
D#	ID information request.
E #	Locate MRQ.
F#	Initialize data tape.
W# or Z#	Locate frame and position film.
G#	Manual fiducial measurement.
H#	Start measuring sequence.

Routine R may be used whenever the operator has made a wrong numerical entry.

ii) Error Recovery Procedures

A. System got wiped out, but the teletype is still alive. This can happen if hardware malfunction causes a certain section to be wiped out. The procedure should be:

O#	Get operator sign-off information, and reload the system.
A#	
B#	
C#	
D#	
E#	
N#	

P#
F#

Do this only if there are many events on the data tape; otherwise just continue on.

H#

B. System got wiped out, but the teletype is dead. Do exactly as (ii-A), except do not attempt to do O#.

C. Remeasure vertex position after a spiral scan, but before advancing to the next event, the recovery procedure should be:

M#

Reposition vertex and push vertex button.

D. Remeasure crutch point or flagged crutch point after the point has been measured incorrectly, but before advancing to the next event, the recovery procedure should be:

M#

Reposition vertex and push vertex button. Reposition for crutch point or flagged crutch point after the spiral scan.

E. Remeasure events because sequence got mixed up, after several events had already been measured. The recovery procedure should be:

E#
N#
H#

F. Remeasure vertex position, crutch points, or flagged crutch point after a spiral scan, and also after advancing to the next event. The recovery procedure should be:

E#
N#
H#

G. Rejecting a vertex, or crutch points, or flagged crutch point, after a spiral scan, but before advancing to the next event. The recovery procedure should be:

M#

Push reject button and enter reject code.

H. Rejecting a vertex, or crutch point, or flagged crutch points, after a spiral scan, but after advancing to the next event. The recovery procedure should be:

E#
N#
H#

PART 3

APPENDICES

I. Instruction Codes and Status Register Layout

i) Radius-Angle High Speed Data Channel

A. Instructions

DWRA	702021	WRITE RADIUS AND ANGLE
DWSC	702022	WRITE SCOPE POINT
DESC	702024	ERASE SCOPE
DRDP	702041	READ DATA POINT (4 WORDS)
DWST	702044	WRITE STATUS
DRST	702052	READ STATUS

Notes:

1) Write radius and angle instruction, DWRA, requires the AC bits 0-8 to contain radius information, and AC bits 9-17 to contain angle information.

2) To write a scope point requires two instructions in the following sequence:

DWRA	WRITE RADIUS AND ANGLE
DWSC	WRITE SCOPE POINT

. .

. .

The accumulator should contain radius and angle information, AC $= RRR\theta\theta\theta_8$. The first instruction DWRA causes radius and angle information to be properly loaded into the radius and angle output register, and the second instruction causes a $(R-\theta)$ point to be written onto the scope.

3) To read a data point instruction, DRDP, reads a set of words corresponding to pulse height, angle, radius, and pulse width information into the computer memory starting from location 10000_8. The instruction sequence is:

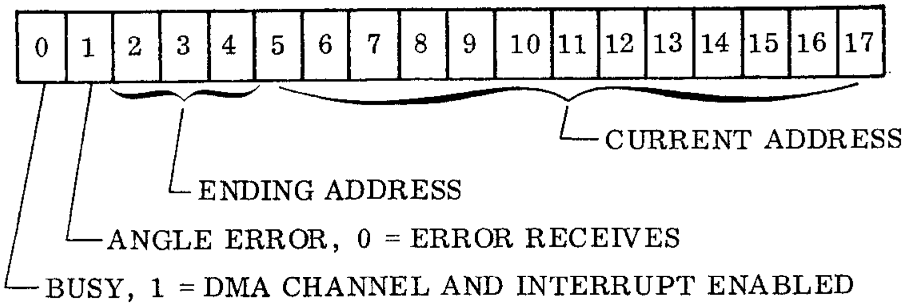
DWSC	WRITE STATUS
DRDP	READ DATA POINT

. .

. .

The accumulator should be AC = 610000. This set of instructions is primarily used for checking the radius scaler for possible loss of counts.

B. Status



Notes:

1) Busy, bit 0, must be set to 1 whenever one wants to activate the DMA channel. This bit is automatically reset to 0 whenever the required number of words are transferred into the computer memory.
2) Angle error, bit 1, must always be set to 1. This bit is automatically reset to 0 whenever an angle error is detected, and this bit must be set to 1 by reissuing another DWST instruction.
3) Ending address, bits 2-4, causes the busy flip-flop to be turned off whenever these three bits and bits 6-8 are identical. This information must be supplied prior to the execution of the status.
4) Current address, bit 5-17, causes the DMA to transfer data starting from the address indicated by the bit setting. Suppose it is required that the data to be transferred starts from memory address 16000 and ends at 16777; then the status word should be AC = 756000. Furthermore, bit 5 is always set to 1 by the hardware to ensure that data will never get transferred into the lower 4K of the computer memory.

C. Data

First word:	Pulse height word
Bits 0-5	Pulse height
Bits 6-8	Separation (lowest = 1/32 full scale, highest = 1/4 full scale).
Bits 9-16	Unused, always = 0
Bit 17	Unused, always = 1
Second word:	Angle word
Bits 0-15	Angle (measured clockwise from vertical)
Bits 16-17	Unused, always = 0

Third word:	Radius word
Bits 0-15	Radius (64000 cts/32 rev, full scale = 65536)
Bits 16-17	Unused, always = 0
Fourth word:	Pulse width word
Bit 0	Unused, always = 0
Bits 1-15	Pulse width (counts between threshold points)
Bits 16-17	Unused, always = 0

Notes:

1) The radius scaler is an up-count scaler and does not clear itself at the beginning of each spiral scan. However, radius scaler must be reset to zero at the beginning of each spiral scan, otherwise the radius information would be in error. To overcome this problem, the following sequence of instruction must be executed prior to activating the radius-angle high speed data channel:

```
CLA     ZERO AC
DWRA    WRITE RADIUS AND ANGLE
  .
  .
```

ii) Periscope, X-Y Stage, and 3 Film Drives

A. Instruction

a) Periscope

PSKP	701021	SKIP ON $\overline{\text{INTERRUPT ENABLED}}$ + $\overline{\text{COUNT ENABLED}}$
PRST	701032	READ STATUS
PØR1	701024	OR 1's TO STATUS
PSAS	701041	SKIP ON AC · STATUS = 0
PWVE	701042	WRITE VELOCITY
PØR0	701044	OR 0's TO STATUS
PWST	701064	WRITE STATUS

b) X stage

XSKP	701121	SKIP ON $\overline{\text{INTERRUPT ENABLED}}$ + $\overline{\text{COUNT ENABLED}}$
XRST	701132	READ STATUS
XØR1	701124	OR 1's TO STATUS
XSAS	701141	SKIP ON AC · STATUS = 0
XWVE	701142	WRITE VELOCITY

XØR0	701144	OR 0's TO STATUS
XWST	701164	WRITE STATUS

c) Y stage

YSKP	701221	SKIP ON $\overline{\text{INTERRUPT ENABLED}}+\overline{\text{COUNT ENABLED}}$
YRST	701232	READ STATUS
YØR1	701224	OR 1's STATUS
YSAS	701241	SKIP ON AC $\cdot$ STATUS = 0
YWVE	701242	WRITE VELOCITY
YØR0	701244	OR 0's STATUS
YWST	701264	WRITE STATUS

d) Film drive 1

FSKP	701321	SKIP ON $\overline{\text{INTERRUPT ENABLED}}+\overline{\text{COUNT ENABLED}}$
FRST	701332	READ STATUS
FØR1	701324	OR 1's TO STATUS
FSAS	701341	SKIP ON AC $\cdot$ STATUS = 0
FWVE	701342	WRITE VELOCITY
FØR0	701344	OR 0's TO STATUS
FWST	701364	WRITE STATUS

e) Film drive 2

GSKP	701421	SKIP ON $\overline{\text{INTERRUPT ENABLED}}+\overline{\text{COUNT ENABLED}}$
GRST	701432	READ STATUS
GØR1	701424	OR 1's TO STATUS
GSAS	701441	SKIP ON AC $\circ$ STATUS = 0
GWVE	701442	WRITE VELOCITY
GØR0	701444	OR 0's TO STATUS
GWST	701464	WRITE STATUS

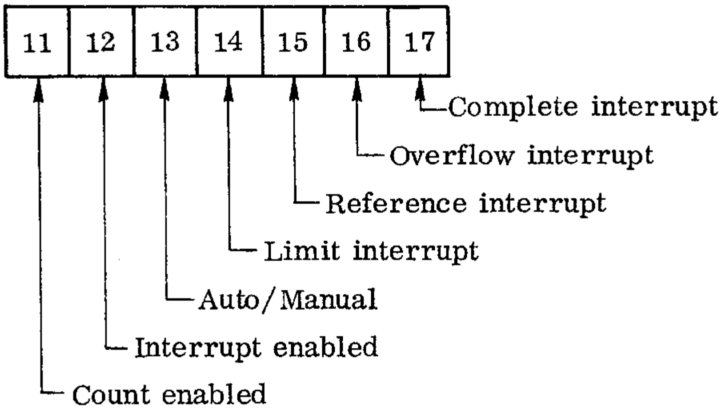
f) Film drive 3

HSKP	701521	SKIP ON $\overline{\text{INTERRUPT ENABLED}}+\overline{\text{COUNT ENABLED}}$
HRST	701532	READ STATUS
HØR1	701524	OR 1's TO STATUS
HSAS	701541	SKIP ON AC $\circ$ STATUS = 0
HWVE	701542	WRITE VELOCITY
HØR0	701544	OR 0's TO STATUS
HWST	701564	WRITE STATUS

Notes:

1) SKIP ON AC · STATUS = 0: If any of the bits in the status registers and the AC are a "1", skip will not occur. AC = MASK.
2) OR 1's TO STATUS: Those bits in the status register corresponding to 1's in the AC are set. Those corresponding to 0's in the AC are unchanged. AC = STATUS 1's.
3) OR 0's TO STATUS: Those bits in the status register corresponding to 0's in the AC are cleared. Those corresponding to 1's in the AC are unchanged. AC = STATUS 0's.
4) WRITE STATUS: This instruction combines both OR 1's and OR 0's into a single instruction, which essentially jams the AC into the status register.
5) READ STATUS: At the end of this command, the desired status register will be contained in bits 11-17 of the AC.
6) WRITE VELOCITY: At the end of this command, the velocity register will contain the contents of AC bits 12-17.

B. Status



Notes:

1) The status register for the periscope, the X-Y stage, and all 3 film drives are similar.
2) Bit 11: 1 = enabled, 0 = disabled. Count enabled allows data channel requests to update scalers. This bit can be turned off for ± 3 counts without losing any counts since the buffer can hold ± 3 counts.

   Bit 12: 1 = enabled, 0 = disabled. Interrupt enabled allows any one of the four interrupt conditions (Limit, Reference, Overflow, and Complete) to interrupt the computer. Upon recognition of the interrupt, this bit is cleared and the four interrupt conditions are left unchanged.

Bit 13: 1 = auto, 0 = manual. Auto mode must be set by the computer. It is reset in the X, Y, and periscope systems when any one of them runs into the limit switches. It is reset in the film drives whenever either reel box carriage bottoms out.

Bit 14: 1 = limit condition, 0 = normal. The limit interrupt is generated whenever the periscope, X, and Y stage return automatically out of the limit switches. In the film drives, the limit condition is set when the leader is encountered; that is, after several frames of light and dark combinations detected by the photodiodes.

Bit 15: 1 = reference condition, 0 = normal. Reference interrupt is generated on the periscope once per revolution by a photodiode. On the X and Y stages it is generated by photodiode pairs near the lower end of the scalers. For the film drive it is generated by a diode pair once per frame.

Bit 16: 1 = overflow condition, 0 = normal. Overflow interrupt is generated whenever any of the six scalers is incremented or decremented to cause bit 0 to change sign.

Bit 17: 1 = complete condition, 0 = normal. Complete interrupt is generated 50 msec after velocity goes to 0 if the velocity is still at the end of the 50 msec.

3) The masks used for detecting interrupts in octal representation are as follows:

000010	LIMIT INTERRUPT
000004	REFERENCE INTERRUPT
000002	OVERFLOW INTERRUPT
000001	COMPLETE INTERRUPT

4) The masks used for resetting individual interrupt in octal representation are as follows:

777767	LIMIT INTERRUPT
777773	REFERENCE INTERRUPT
777775	OVERFLOW INTERRUPT
777776	COMPLETE INTERRUPT

5) The masks used for setting auto mode and manual mode:

$$\left.\begin{array}{l}\text{count enable}\\ \text{interrupt enable}\\ \text{auto mode}\end{array}\right\} \qquad (000160)_8$$

$$\left.\begin{array}{l}\text{count enable}\\ \text{interrupt enable}\\ \text{manual mode}\end{array}\right\} \qquad (000140)_8$$

C. Velocity register

The velocity register is a 6-bit register occupying bits 12-17. Prior to the execution of the write velocity, PWVE, command, the AC should contain the velocity information. Bits 0-11 are not used, and bits 12-17 contain velocity information. Velocity range is given below:

$$\left.\begin{array}{l}00_8 = 0\\ 37_8 = \text{positive full scale}\end{array}\right\} \qquad \text{Positive Velocity}$$

$$\left.\begin{array}{l}77_8 = -1\\ 40_8 = \text{negative full scale}\end{array}\right\} \qquad \text{Negative Velocity}$$

iii) Console Buttons

A. Instructions

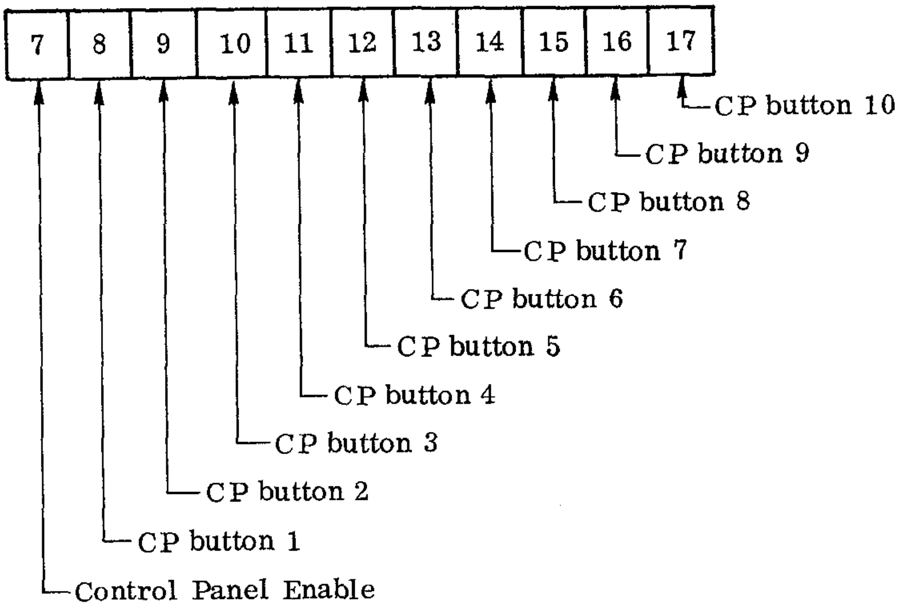
CPS	702101	SKIP ON CP ENABLED
CPW	702104	WRITE CP STATUS
CPR	702112	READ CP STATUS

Notes:

1) Write CP status instruction writes the contents of the AC into the status register.

2) Read CP status instruction reads the contents of the status register in the AC.

B. Status



Notes:

1) Bits 0-6 are not used.
2) Bits 7: 1 = control panel enabled, 0 = control panel disabled. This bit is reset to 0, whenever any one of the control panel buttons are pushed, and it will remain reset until this bit is set in the AC and the write CP status, CPW, instruction is issued.
3) Bits 8-17: 1 = control panel button enabled, 0 = normal. Whenever a control panel button is pushed, its corresponding bit will remain set until it is cleared with zero in the AC and the CPW instruction.
4) The masks used for detecting interrupts in octal representation are as follows:

001000	CP BUTTON 1
000400	CP BUTTON 2
000200	CP BUTTON 3
000100	CP BUTTON 4
000040	CP BUTTON 5
000020	CP BUTTON 6
000010	CP BUTTON 7
000004	CP BUTTON 8
000002	CP BUTTON 9
000001	CP BUTTON 10

5) The mask used for enabling the control panel interrupts: $(002000)_8$.

iv) Image Plane Digitizer (IPD)

A. Instructions

MSS	702331	SKIP IF IPD NOT HOMED
MSLW	702324	WRITE LEFT REGISTER
MSLR	702332	READ LEFT REGISTER
MSRW	702344	WRITE RIGHT REGISTER
MSRR	702352	READ RIGHT REGISTER

Notes:

1) Skip if IPD not homed instruction is used for checking whether the puck of the IPD is placed back to its home position or not.
2) IPD uses two Gurley recorders to give the coordinate readings. Both scalers can be written and read via the AC register.

B. Status

4	5	6	7	8	9	10	11	12	13	14	15	16	17

Notes:

1) IPD does not have a status register.
2) IPD has two identical 14-bit up-down scaler registers.
3) The scalers can be written or read under program control via the accumulator.

v) Auto-Fiducial

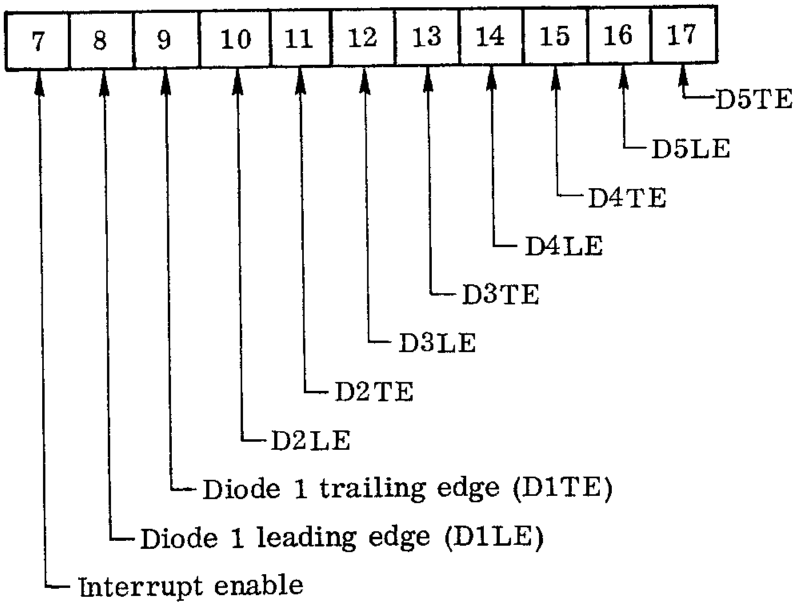
A. Instructions

AFS	702201	SKIP ON-AF ENABLED
AFW	702204	WRITE AF STATUS
AFR	702212	READ AF STATUS

Notes:

1) Write AF status instruction writes the contents of the AC into the status register.
2) Read AF status instruction reads the contents of the status register into the AC.

B. Status



Notes:

1) Bits 0-6 are not used.

2) Bit 7: 1 = interrupt enabled, 0 = interrupt disabled. This bit is reset to 0 whenever any one of the diodes changes its state from dark to light, or from light to dark. This bit is set to 1 by setting the same bit in the AC and then issuing the AFW command.

3) Bits 8-17: 1 = diode transition, 0 = normal. The bits are set by diode transitions. The bit will remain set until zero is loaded into the AC, and AFW command is issued. An interrupt is generated whenever the interrupt is enabled, and any one of the bits is set.

4) The masks used for detecting interrupts in octal representation are as follows:

001000	Diode 1 Leading edge
000400	Diode 1 Trailing edge
000200	Diode 2 Leading edge
000100	Diode 2 Trailing edge
000040	Diode 3 Leading edge
000020	Diode 3 Trailing edge
000010	Diode 4 Leading edge
000004	Diode 4 Trailing edge
000002	Diode 5 Leading edge
000001	Diode 5 Trailing edge

5) The masks used for enabling the auto-fiducial interrupts: (00200$_8$).

vi) Magnetic Tape Controller

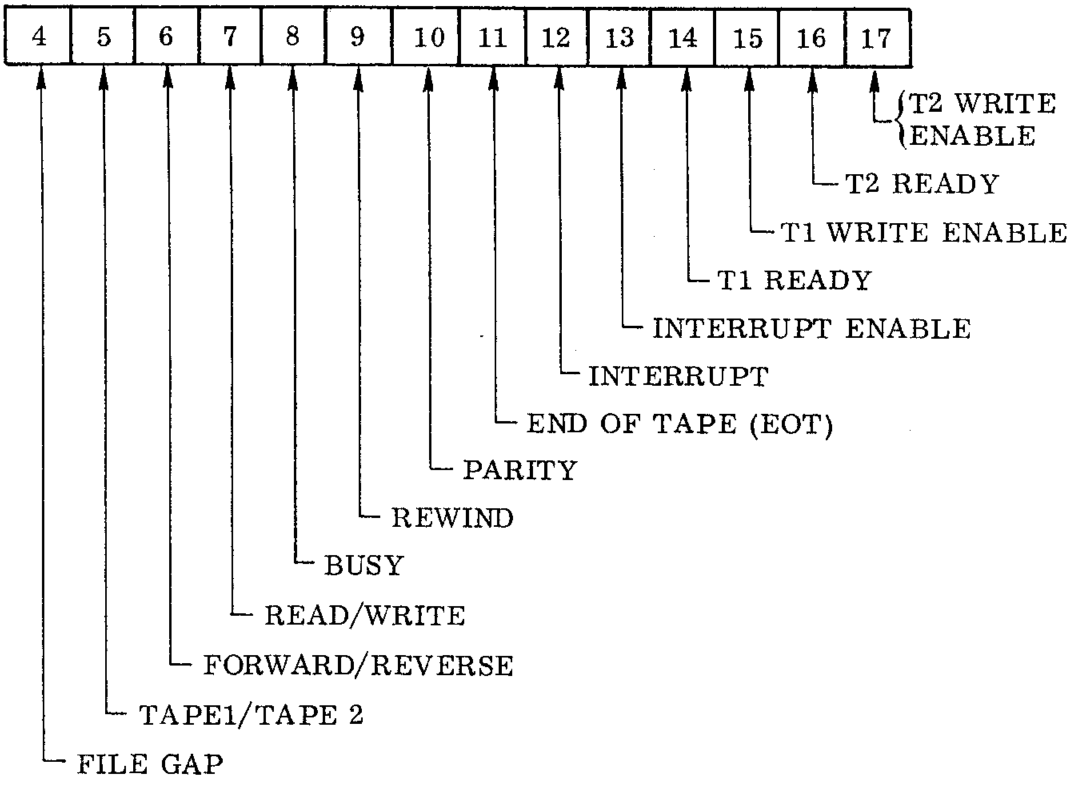
A. Instructions

MSK2	703621	SKIP IF TAPE DRIVE # 2 READY
MSK1	703641	SKIP IF TAPE DRIVE #1 READY
MWST	703624	WRITE STATUS
MRST	703632	READ STATUS
MWDB	703644	WRITE DATA BUFFER
MRDB	703652	READ DATA BUFFER

Notes:

1) The skip instructions are used to check whether the magtape drives are ready or not.
2) Read status and write status instructions require the use of the AC register.
3) Read and write data buffer instructions used for examining and modifying data in the data buffer.

B. Status



Notes:

1) Bits 0-3 are not used.
2) Bit 4: 1 = generate file gap, 0 = normal. The status of this bit can only be set but it cannot be read. Upon completion of the file gap, the busy is reset to 0, and the interrupt is set to 1.
3) Bit 5: 1 = tape 1 selected, 0 = tape 2 selected. This bit selects the tape drives and the controller can only serve one tape drive at a time.
4) Bit 6: 1 = forward, 0 = reverse. This bit selects the motion of the tape drives.
5) Bit 7: 1 = read, 0 = write. This bit selects the playback mode or record mode on the tape drives. Upon completion of read (data clock stopped) busy is reset to 0 and interrupt is set to 1. However, upon receipt of buffer full indication during read, interrupt is set to 1. Also upon completion of write (buffer emptied) busy is reset to 0 and interrupt is set to 1.
6) Bit 8: 1 = busy, 0 = not busy. This bit starts the selected tape drive moving. The resetting of this bit to zero occurs at the completion of file gap generation, read, or write.
7) Bit 9: 1 = rewind, 0 = normal. This bit starts the selected tape drive in rewind. Upon completion, this bit remains set and interrupt is set to 1. This bit must be reset by MWST instruction.
8) Bit 10: 1 = parity error, 0 = normal. This bit is set to 1 whenever vertical or longitudinal parity has occurred. No interrupt is generated. However, this bit remains set at the end of a read and must be reset by the MWST instruction.
9) Bit 11: 1 = EOT, 0 = normal. This bit is set to 1 whenever end-of-tape condition has occurred and interrupt is set to 1. This bit remains set and must be reset by the MWST instruction.
10) Bit 12: 1 = interrupt, 0 = normal. This bit is set to 1 at the completion of file gap generation, read, buffer full, write, rewind, or EOT. This bit is reset to 0 whenever the interrupt request to the computer is honored, and grant signal is used for resetting interrupt and channel enable bit (Bit 13).

11) Bit 13: 1 = interrupt and channel enable, 0 = disable. This bit must be set to 1 in order to initiate data transfer to and from the computer to the magtape controller. This bit is reset by the grant signal from the computer.

12) Bits 14-17 are not able to be set from the computer. They are just status to indicate that state of the two tape drives. For instance:

Bit 14: 1 = T1 not ready, 0 = T1 ready.
Bit 15: 1 = T1 write disabled, 0 = T1 write enabled.
Bit 16: 1 = T2 not ready, 0 = T2 ready.
Bit 17: 1 = T2 write disabled, 0 = T2 write enabled.

13) The octal bit patterns that should be in the AC before the execution of write status, MWST, instructions for various functions should look as follows:

TAPE 1	TAPE 2	
015020	005020	WRITE DATA
017020	007020	READ DATA
035020	025020	WRITE END OF FILE
013020	003020	BACKSPACE
012420	002420	REWIND
000000	000000	CLEAR STATUS

```
ex. LAC            (015020     LOAD T1 WRITE DATA
    MWST                       LOAD T1 WRITE DATA
     .              .
     .              .
```

These two instructions command tape 1 to write a record, whose length is given in location 32; and whose buffer address is stored in location 33.

14) To generate an end-of-file mark on the tape 1 the following instructions must be used:

```
LAC            (046114     LOAD END-OF-FILE
MWDB                       WRITE DATA BUFFER
LAC            (015020     LOAD T1 WRITE DATA
MWST                       WRITE STATUS
 .              .
 .              .
```

IBM tape format requires two consecutive 13_{16} words as end-of-file, and in octal this is represented by (046114_8).

15) The masks used for detecting interrupts in octal representation are as follows:

004000	WRITE INTERRUPT
006000	READ INTERRUPT
002000	BACKSPACE INTERRUPT
002400	REWIND INTERRUPT
007000	DATA BUFFER OVERFLOW INTERRUPT
000100	EOT INTERRUPT
000200	PARITY (NOT AN INTERRUPT)
010000	TAPE 1 OR TAPE 2 MASK

16) Locations 32_8 and 33_8 are locations for word count and current address information respectfully. Prior to the execution of read or write instruction, 32_8 must contain buffer size in two's complement form, and 33_8 must contain the actual buffer address less one location.

17) When reading a record back from the tape, the input buffer size should be one word larger than required, in order to accommodate the longitudinal check character at the end of the record.

II. Automatic Priority Interrupt (API) and Data Channel Assignment

i) Automatic Priority Interrupt Assignments

LEVEL	DEVICES	ADDRESS
0	Reserved for system use	
1	a) Radius-angle h.s. data channel	100
	b) Control panel button	111
	c) Magnetic tape controller	45
2	a) Periscope	101
	b) X-stage	102
	c) Y-stage	103
	d) Film drive 1	104
	e) Film drive 2	105
	f) Film drive 3	106
3	a) Auto-fiducial	112
	b) Realtime clock	51

LEVEL	DEVICES	ADDRESS
4	Software priorities	40
5		41
6		42
7		43

ii) Data Channel Assignments

DEVICES	WORD COUNT ADDRESS	CURRENT ADDRESS	SCALER ADDRESS
MAGNETIC TAPE	32	33	
PERISCOPE	140	141	200
X-STAGE	142	143	201
Y-STAGE	144	145	202
FILM DRIVE 1	150	151	204
FILM DRIVE 2	152	153	205
FILM DRIVE 3	154	155	206

III. Calling Sequence for Some Commonly Used Routines

i) Teletype

A. Alpha-numerical message request

```
JMS     TTAMR     GO REQUEST ALPHA-NUMERICAL TYPEOUT
LAW     XXM       ADDRESS OF MESSAGE
JMP     JOBXCT    GO EXECUTE THE NEXT JOB
```

B. Numerical type out request

```
JMS     TTNOR     GO REQUEST NUMERICAL TYPEOUT
LAW     XXN       ADDRESS OF NUMERICAL DATA
JMP     JOBXCT    GO EXECUTE THE NEXT JOB
```

C. Numerical input request

```
JMS     TTNIR     GO REQUEST NUMERICAL INPUT
LAW     XXM       ADDRESS OF MESSAGE
LAW     XXN       ADDRESS OF RETURNED DATA
LAW     XXJS      ADDRESS OF NEXT JOB STEP
JMP     JOBXCT    GO EXECUTE THE NEXT JOB
```

Notes:

The following two instructions must precede the calling sequence if simultaneous execution of another routine and the teletype type out is required:

```
LAW   XXXX   ADDRESS OF ROUT. TO BE RAN SIMULTANEOUSLY
DAC   TTRS1  STORE IN TELETYPE REQUEST SERVICE
```

ii) Magtape

A. Read 1 record/space forward N records

```
JMS    SFNR1     MAGTAPE 1 SPACE FORWARD N RECORDS ROUT.
LAW    -X        -X =WORD COUNT (BUFFER SIZE)
LAW    BUF-1     ADDRESS OF BUFFER -1
LAW    -N        -N = N FORWARD SPACE
SAD    (046114   SKIP IF NOT END OF FILE
 .       .
 .       .
```

B. Backspace N records

```
JMS    BSNR1     MAGTAPE 1 BACKSPACE IN RECORDS ROUT.
LAW    -N        -N = N BACKSPACE SPACE
 .       .
 .       .
```

C. Generate end-of-file

```
JMS    WEOF1     MAGTAPE 1 WRITE END OF FILE ROUTINE
 .       .
 .       .
```

D. Rewind

```
JMS    REW1      MAGTAPE 1 REWIND ROUTINE
 .       .
 .       .
```

E. Write 1 record

```
JMS    WREC1     MAGTAPE 1 WRITE 1 ROUTINE
LAW    -X        -X =WORD COUNT (BUFFER SIZE)
LAW    BUF-1     ADDRESS OF BUFFER -1
 .       .
 .       .
```

Notes:

1) This set of calling sequence applies to magtape 1 only, for it to apply to magtape 2 simply change the letter from 1 to 2 after each JMS instruction.
2) The following two instructions must precede the calling sequence if simultaneous execution of another routine and magtape routine is required:

```
LAW   XXXX   ADDRESS OF ROUT.  TO BE RANSIMULTANEOUSLY
DAC   MTRS1  STORE IN MAGTAPE REQUEST SERVICE
```

iii) X-Y Stage

```
LAC    aaaa     X-DESTINATION
DAC    VIEW1X   STORE IN VIEW1X
LAC    bbbb     Y-DESTINATION
DAC    VIEW1Y   STORE IN VIEW1Y
JMS    V1SR     GO TO DESTINATION POINT IN VIEW 1
 .      .
 .      .
```

Notes:

1) For view 2 the following names have to be used:
   VIEW2X, VIEW2Y, V2SR, and for view 3;
   VIEW3X, VIEW3Y, V3SR, respectively.
2) The following two instructions must precede the calling sequence if simultaneous execution of another routine and X-Y stage routine is required:

```
LAW   XXXX   ADDRESS OF ROUT.  TO BE RAN SIMULTANEOUSLY
DAC   XYDR1  STORE IN STAGE DRIVE ROUT.
```

iv) Film Drive

```
LAC    aaaa     NUMBER OF FRAMES TO BE MOVED
DAC    FLDEST   STORE IN FLDEST FOR FILM DRIVE 1
DAC    GLDEST   STORE IN GLDEST FOR FILM DRIVE 2
DAC    HLDEST   STORE IN HLDEST FOR FILM DRIVE 3
JMS    FLDR     GO MOVE FILM
 .      .
 .      .
```

Notes:

1) If the content of location aaaa is positive, then the film drive will advance. However, if the content of location aaaa is negative, then the film drive will reverse.
2) The following two instructions must precede the calling sequence if simultaneous execution of another routine and film drive routine is required:

```
LAW   XXXX    ADDRESS OF ROUT. TO BE RAN SIMULTANEOUSLY
DAC   FLDR1   STORE IN FILM DRIVE ROUT.
```

v) Stage Coordinate Routine

```
JMS   SCWCR   GO TO X-Y STAGE COORDINATE WORD CONVERTER
              ROUTINE
LAW   XXXX    ADDRESS WHERE DATA SHOULD BE RETURNED
XX            X COORDINATE VALUE
YY            Y COORDINATE VALUE
 .     .
 .     .
```

Notes:

1) This routine converts one 18-bit word to two 16-bit words.
2) The address as to where the routine should store the converted data and the value of X and Y coordinate value should be stored properly in the sequence prior to the execution of this routine.

vi) Memory-to-Memory Data Transfer

```
JMS   MMDTR   GO TO MEMORY-TO-MEMORY DATA TRANSFER
              ROUTINE
LAW   -N      NUMBER OF WORDS TO BE TRANSFERRED
LAW   XXXX    ADDRESS WHERE DATA TO BE TRANSFERRED FROM
LAW   YYYY    ADDRESS WHERE DATA TO BE TRANSFERRED TO
```

Notes:

1) The number of words to be transferred and the addresses of data to be transferred to and from should be stored properly in the sequence prior to the execution of this routine.

vii) Input Buffer Right Justifier Routine

```
JMS   IBRJR    GO TO INPUT BUFFER RIGHT JUSTIFIER ROUTINE
LAW   -N       INPUT BUFFER SIZE
LAW   XXXX     INPUT BUFFER ADDRESS
 .      .
 .      .
```

Notes:

1) This routine must be used after a record has been read into the computer memory.
2) The buffer size and buffer address should be stored properly in the sequence prior to the execution of this routine.

viii) Output Buffer Left Justifier Routine

```
JMS   OBLJR    GO TO OUTPUT BUFFER LEFT JUSTIFIER ROUTINE
LAW   -N       OUTPUT BUFFER SIZE
LAW   XXXX     OUTPUT BUFFER ADDRESS
 .      .
 .      .
```

Notes:

1) This routine must be used before a record is going to be written onto the magtape from the computer memory.
2) The buffer size and buffer address should be stored properly in the sequence prior to the execution of this routine.

ix) Clear Buffer With Ones

```
JMS   CLBWO    GO CLEAR BUFFER WITH ONES
LAW   -X       BUFFER SIZE
LAW   XXXX     BUFFER ADDRESS
 .      .
 .      .
```

Notes:

1) This routine is used whenever a buffer is needed to be cleared with all ones.
2) The buffer size and buffer address should be stored properly in the sequence prior to the execution of this routine.