A Description of the Control Program for the Lunar Ranging Experiment

J. D. Rayner

Technical Report No. 70-064

November 1969

UNIVERSITY OF MARYLAND
DEPARTMENT OF PHYSICS AND ASTRONOMY
COLLEGE PARK, MARYLAND

A DESCRIPTION OF THE CONTROL PROGRAM FOR THE LUNAR RANGING EXPERIMENT

J. D. Rayner

TECHNICAL REPORT NO. 70-064

November 1969

Work supported by NASA Grant #NGR 21-002-109

University of Maryland

Department of Physics & Astronomy

College Park, Maryland

TABLE OF CONTENTS

SECTION I

INTRODUCTION

The lunar ranging ground station is built around a Varian 620/i control computer. This arrangement gives a good deal of operating flexibility as well as providing a useful degree of computational capability. In the present system, the computer has three main functions: to control the firing of the laser and gate the return pulse to accumulate data from the return pulse, to calculate an approximate range from the data and/to record the raw data supplemented by calibration data on magnetic tape. A fourth function is possible in the future, namely, to analyze the return data after a run.

The computer controls the laser by means of two external control lines, one to fire the flash lamps and one to enable the Pockels cell. To determine the firing time, the computer reads the Astrodata time-of-day clock which derives its time base from the main oscillator. The gate for the return is provided by an Eldorado Time-Delay-Generator which can be loaded under program control by the computer. The projected range which is used to set the time delay generator is read by the computer from magnetic tape by the read-write tape deck.

For a complete description of the timing electronics, the reader should see technical report No. 70-049 by C. Steggerda. Only an outline will be given here. The return is timed to 50 nanoseconds by a time interval meter which counts 20 MHz clock pulses. This time interval is refined by measuring the time from the start pulse to the first counted pulse and from the stop pulse to the first uncounted pulse with Time-to-Pulse-Height Converters. The pulse height is then converted to digital form by an Astrodata Analog-to-Digital converter which can be read under program control by the computer.

Permanent data storage is on magnetic tape by means of a write-only tape deck. This deck can also be used in conjunction with the read-write deck to copy several output tapes onto one for data concentration.

As the program was originally conceived by Don Day and the author, the system would run continuously with provision for supression of laser action or teletype output under control of sense switches on the computer console. As the system evolved, it became obvious that this was not the ideal mode of operation as the system was being used in bursts of 50 to 100 shots rather than continuously. For this reason, the system as presently constituted has two phases, a control phase and a run phase.

Normally the program is in control phase which accepts and acts on commands entered from the teletype. Each command is in the form of a string of letters followed by a space. As only the first letter is scanned, there are 26 possible unique commands of which the following are implemented:

AID	-	enter the AID debugging program
CONTINUE	-	continue the present run
DELTA	-	set DELTA
FIRINGS	-	set number of firings

GO	-	start a run
KALIBRATION	-	enter the calibration program
STOP	-	terminate run
WARMUPS	-	set number of warmups

As can be seen, typing G or GO switches the program from the control to the run phase. This phase first fires the flash lamps a prespecified number of times to bring the laser system up to operating temperature. It then enters the main loop which fires the laser every 4 seconds and types the residual range and the associated time. The program is returned to the control phase on the completion of the specified number of shots or by a character from the teletype.

SECTION II

SYSTEM OPERATION

The operating system is normally in core but if it has been damaged it can be read in from magnetic tape using the program EASE. There are two starting points. Starting at memory location 1 causes the program to read a tape record. This is useful for initial start up and when mounting a new tape. Otherwise the system can be started in the command interpreter at location 4000_8 which is where it restarts after power down.

Once started, the program is ready to accept commands from the teletype keyboard. Each command consists of a key letter followed by an optional and arbitrary string of letters terminated by a space. Thus GO, G, and GORP all produce the same effect. Any non-alphebetic character before the space is treated as an error and the program types a carriage return and then waits for the next command. Any numeric parameters needed are typed after the command; each parameter being terminated by the first non-numeric character. The commands presently implemented are:

DELTA ddd - Sets the value of the constant DELTA which is subtracted from the projected range before it is set into the range gate generator. ddd represents any string of digits.

WARMUPS ddd - This sets the number of warmup flash lamp firings before the laser is fired.

FIRINGS ddd - This sets the number of laser shots in a run.

GO - This initiates a run with DELTA set by the last D command, the number of warmups set by the last W command and the number of shots set by the last F command.

STOP - This command terminates the present run and writes the run data on magnetic tape.

CONTINUE - Resumes an interrupted run. This should not be used after an S command since an S completely terminates a run. A C can be used to restart a run interrupted by any of the other commands or by a power drop.

AID - Transfers control to AID, the Varian system debugging program.

K - Transfers control to the calibration program.

The action of the program can also be modified by use of the three sense switches on the computer front panel. These allow the operator to enter the test mode and suppress lasing or magnetic tape output. They should normally be in the up position.

Once the various parameters and switches have been set, a run can be started with a G command. The computer will then type out the current value of DELTA and the time, and begin the warmup cycle. At the completion of the warmup, the computer will commence firing the laser once every 4 seconds. The result of each shot is typed on the teletype. If no start pulse is received, NO LASER will be typed out. Absence of a stop pulse causes NO RETURN to be printed. Otherwise the minutes and seconds of the time and the residual range to 0.1 nanosecond are typed out. For example:

```
34:16 0175.3
```

The 6 second version provides more complete output; listing the complete time (day:hour:minute:second), the range sent to the range gate in nanoseconds, and the residual range to 0.1 nanosecond. For example:

```
271:08:21:06 2513275947 0002371.1
```

When the required number of shots have been completed, the computer will cease firing the laser and write the complete raw data out on magnetic tape (See Appendix II). It will then revert to the command mode ready for the next run.

There are three main errors that can occur when running the program. They are all flagged by a message typed on the teletype. As the program takes remedial action, there should be no need for operator intervention.

SYNC ERROR - indicates that the time read with the last tape record is later than the time read from the clock. This can be due to the moon setting, or a missed or bad tape record. The computer will wait until the time read from the clock catches up with that read from the tape.

SHORT REC. - indicates that the last record read from tape was too short. The computer then reads the next record which may cause a SYNC. ERROR.

REDUNDANCY - there was a redundancy check error in the last tape record. The computer takes the same action as for SHORT REC.

SECTION III

PROGRAM OPERATION

Command Interpreter. The command interpreter (see Fig. 1) reads and acts on all commands from the teletype and while doing this keeps the magnetic tape synchronized with the time of day clock. It is normally in a loop alternately checking the clock and the teletype input flag. If the time read from the clock is later than that associated with the last tape record, UDR is called to catch the tape up to the clock.

When the teletype flag is detected, a character is read and echoed back to the teletype. If the character is a letter, it is stored and additional characters are read until a blank is detected. A non-alphabetic character causes a carriage return to be typed and the command interpreter to be restarted.

On detection of a blank, a branch is made on the initial letter of the command. This is accomplished by biasing the character so the code for Z is 0 and then simultaneously incrementing the character and an indirect pointer until the character goes to 0. The pointer is then pointing to the address of the routine corresponding to the character and is used to indirectly jump to the routine. Those entries in the 26 word address table corresponding to letters for which no commands have been implemented contain the address of the error routine.

Associated with the command interpreter are two utility routines that can be used to read numerical parameters.

RDND is not presently a subroutine but rather the routine associated with the command D. It reads a string of up to 10 digits and packs them right-justified 4 to a word in BCD using three words. RDNB is a subroutine which can be called by other service routines. It reads a string of digits and converts it to a binary number exiting on the first non-numeric character with the result in the accumulator. Both these routines use the same read loop as the command interpreter and thus keep the tape synchronized with the clock.

Run Initialization. The command G transfers control to the initialization section of the control program. The counters are set to their initial value and the current value of DELTA is inserted in the heading. The warmup loop is then entered. UDR is called to establish the next firing time - T0. Next WT0 is called to wait for T0 and type the heading and WT0 is excited. and the flash lamps fired. The loop is then repeated until the counter goes to zero. At the completion of the warmup the remainder of the heading is typed and the output pointers reset.

Main Loop. There are two things to be followed in understanding the main loop: the temporal sequence of events and the flow of data. First the sequence of events during one interval will be followed. (see figs. 2-3)

UDR (Update Range) established T0 and the associated range. This range then has DELTA subtracted from it and is sent to the time delay generator by AOR (adjust and Output Range). WT0 (Wait for T0) is then called. While waiting for T0, WT0 types the results of the last shot and looks for the final vernier. If a final vernier is found, the output pointer OPP1 is set to output data. At TO, EFT is called conditionally on sense switch 1 to fire the flash lamp. Then FIRE is called conditionally on sense switch 1. The shot counter NF1 is checked. If it is zero, control is transferred to the cleanup routine. Otherwise it is decremented and the laser fired. The output pointer OPP2 is then set to the NO LASER message and the verniers zeroed.

The data from the last shot is then transferred to the output buffer by LOB (Load Output Buffer). RVO (Read Vernier O) is then called to read the initial vernier. If no vernier is received by TO + 0.01 sec, RVO is exited leaving OPP1 set to NO LASER. Otherwise the vernier is stored and OPP2 reset to NO RETURN.

CRR (Calculate Residual Range) is then called to calculate the residual range for the last shot. OUT is called next to pack the time and residual range for teletype output. At this time the output pointers are pushed down. The actual output pointer TTYP is set equal to OPP1. OPPL is then set to OPP2 and OPP2 to the null message. The time and range of the shot is then transferred to the current data buffer by MOVE and the cycle repeats with UDR.

In following the data flow it is important to keep in mind that at any time there are three sets of data being processed; that pertaining the the previous shot, that pertaining to the present shot, and that pertaining to the next shot. To understand the program, it is necessary to unravel the flow of this data from the above sequence of events.

Data for a shot is first introduced in UDR when the time of the next shot (DAY, HM, SEC), the firing time of the next shot (DAYO, HMO, SECO), and the projected range (R1, R2, R3) are established. The range is then decremented by DELTA (DR1, DR2, DR3) in AOR and sent to the Time Delay Generator.

At the beginning of the next interval the initial vernier is read into AVO by RVO. Then the time and projected range are moved from (DAY, HM, SEC, R1, R2, R3) to (DAY, HM1, SEC1, RA1, RA2, RA3) by MOVE. The final verniers are then read into AV1 and AV2. Then just before the end of the interval the Time Interval Meter is read into RB1 and RB2.

At the start of the next interval LOB stores the raw data in the tape output buffer. CRR then calculates the residual range and stores it in RB2 and RB3.. The time and residual range are then taken by OUT unpacked coded for the teletype and placed in the teletype output buffer ROUT. They are then typed out by TYPE.

Cleanup. At the end of the run there is still the data from the last shot to be processed. This is accomplished by calling LOB, CRR and OUT. If sense switch 3 is set, WOT (Write Output Tape) then writes the data from the output buffer onto magnetic tape. (See Appendix II) Once the tape record has been written, control is returned to the command interpreter.

UpDate Range. The function of UDR (see Fig. 4) is to establish two times and an associated projected range for the next shot. The times are T, the time of the next shot, and TO the firing time of the next shot which is 2 milliseconds earlier. The proper second of the next shot is established by reading the current second from the time-of-day clock and looking it up in a table of possible times. Each entry contains two times, one corresponding to shot times (SEC) and the other, one second less, corresponding to firing times (SECO). The first entry, for which SEC is greater than the second read from the clock, is taken as the next firing time. The two times are then stored in SEC and SECO. The days, hours, and minutes of T (DAY, HM) and TO (DAYO, HMO) are then set from the clock. If SEC equals 60, IM (Increment Minutes) is then called to increment T by one minute. If sense switch 2 is set the time associated with the current tape record (DAYT, HMT) is compared to T (DAY, HM) and tape records read until the time from the tape is equal to or greater than T. Otherwise T (DAY, HM) is substituted for (DAYT, HMT). If (DAYT, HMT) is greater than T, SYNC ERROR is typed and control is returned to the command interpreter unless a run is under way in which case UDR waits for the clock to catch up to the tape. Once the clock and the tape are synchronized, the range is located in the input buffer and stored in R1, R2 and R3. This is accomplished in two steps. First a pointer to the range is found by refrencing relative to the time found in the table of possible times. This pointer is then used to pick up the required range.

After this is accomplished, the clock is checked to make sure the projected firing time has not passed. If it has, the subroutine starts over from the beginning; otherwise it exits.

READ. Magnetic tape input is taken care of by READ which requires two arguments. The first is the location into which the record is to be read, while the second is the negative of the record length. Three 6 bit tape characters are packed in a 16 bit word by ignoring the two high order bits of the first character.

Three errors are detected. Short records and redundancy check errors cause a message to be typed and the next record to be read. In the case of long records, the required number of words are taken from the beginning of the record and the rest of the record is ignored.

Adjust and Output Range. If sense switch 2 is set AOR uses the one word BCD subtraction routine three times to subtract DELTA (DR1, DR2, DR3) from the projected range (R1, R2, R3); and sends the result to the Time-Delay-Generator as well as replacing the old value of the projected range. SUB subtracts two packed BCD words one digit at a time using the overflow indicator for the carry bit. Otherwise DELTA is substituted for (R1, R2, R3) and sent to the TDG.

Wait for T0. While waiting for T0, WT0 takes care of various background tasks by means of the subroutine JUNK. JUNK is called once each time the clock is checked except when waiting for milliseconds. In the current system JUNK calls TYPE to type messages on the teletype, RV12 to look for a final vernier. It also checks the teletype input flag returning control to the command interpreter on detection of an input character.

TYPE checks the teletype output flag and exits if the teletype is busy. Otherwise the next character to be typed is picked up indirectly through the pointer TTYP. A zero character indicates the end of a message and TYPE exits without incrementing TTYP. All other characters are sent to the teletype with TTYP incremented.

RV12 exits immediately if AV0 is zero. Otherwise it checks the flags for the two Sample-and-Hold Amplifiers associated with the final vernier Time-to-Pulse-Height Converters. If a flag is detected, that Sample-and-Hold Amplifier is connected to the Analog-to-Digital Converter. RADC is then called to read the result of the conversion. There is a several μ-second lag from the time the Executive Command is sent to start a conversion, until the converter busy flag comes up. For this reason RADC waits for the converter busy flag to come up and then go back down. The converted value is then read and stored in AV1 or AV2, and OPP2 set to display the return data.

Read Vernier 0. RV0 detects the initial vernier flag and reads the converted value with RADC as described above, storing it in AV0 with OPP2 set to NO RETURN. The loop which checks the flag also checks the clock and RV0 exits 0.01 sec after the second if no return is detected. This leaves OPP2 set to NO LASER.

Calculate Residual Range. The residual range is calculated in three stages in CRR (see Fig. 5), echoing the evolution of the program. The first stage finds the range to 50 nanoseconds by multiplying the output of the Time Interval Meter by five. This is accomplished by multiplying by ten and then dividing by two using the routine DB2 three times. DB2 divides a 4 digit BCD word by two by means of a series of shifts and logical operations using DC for the carry.

In the next stage of the calculation, the projected range is subtracted from this calculated range using the subroutine SUB. This gives a three word residual range good to 50 nanoseconds.

Finally, the verniers are added giving a range to 0.1 nanosecond. As read from the Analog-to-Digital Converter, the verniers consist of 12 bits of voltage information followed by a 4 bit channel number. The voltage information is in two's complement form with the exception of the high order bit which is inverted being 0 for negative and 1 for positive voltages. To convert the input word to a valid number,the high order bit is inverted and the whole thing shifted arithmetically 4 places to the right. This number can then be related to the time measured by the Time-to-Pulse-Height Converter with a polynomial transformation. In the present system used to monitor ranges on-line, only the linear term is used. It however, provides quite a good fit for the Time-to-Pulse-Height Converters as can be seen in Figure 6. Since the initial vernier is added and the final vernier subtracted, the constant term drops out. The vernier correction is then calculated by multiplying the difference of the initial and final verniers by the conversion factor 0.96818, which gives the correction to 0.1 nanosecond. The two low order words of the previously calculated course residual range are then converted to binary, multiplied by ten and added to the vernier correction. The result is then converted to BCD and stored in RB2 and RB3.

OUT. The time and residual range are unpacked and coded for teletype output by the routines PACK, PAK2, and PAK4. The results are stored in the teletype output buffer ROUT under control of a pointer in the X register. Punctuation is inserted by incrementing the X register, thus skipping over words in the output buffer which contain the necessary punctuation characters. OUT also pushes down the three output pointers. OPP1 goes to TTYP, OPP2 goes to OPP1, and then OPP2 is set to the null message NULM.

SECTION IV

TESTING AND DEVELOPMENT

As stated above, the original version of the program was developed by Mr. Don Day and the author. This version ran continuously and was only concerned with outputing ranges to the Time Delay Generator. The time and range were also typed out, giving a means of checking the program against the printout from the Univac 1108 program which produced the range tapes. This system was further checked by using the Time Delay Generator to start and stop the Time Interval Meter, thus giving an independent confirmation that the correct range was sent to the Time Delay Generator.

After it was moved from McDonald Observatory, the system was modified to read the Time Interval Meter and print the reading on the teletype. This change as well as the subsequent changes described below was made by means of octal patches entered directly into core by means of the Varian system debugging program AID. This was due to the inconvenience and time expenditure involved in the use of a paper tape based Symbolic Assembler.

Since the basic clock rate is 20 MHz, the printout of the Time Interval Meter was in units of 50 nanoseconds, which was not immediately useful without annoying calculation. To alleviate this, the divide by two routine DB2 was added with which proper scaling allowed output in nanosecond units (though only accurate to 50 nanoseconds).

To further simplify the interpretation of return data, it was decided to present the residual rather than the actual range. To assure a positive residual range, the range sent to the range gate, rather than that read from the tape, was subtracted from the measured range. It was this version of the program that was in use at the commencement of lunar ranging in July.

After a few weeks of operation, two changes suggested themselves. The first was a change from an essentially continuous operation which could be inhibited between runs to an intermittant operation with a fixed number of shots in each run. It also was desirable to simplify the changing of program parameters to ease the training of operating crews. To this end, an executive phase was added to the program to handle the changing of parameters and the initiation of runs. This modification has proved a very satisfactory and vastly simplified system operation.

The next modification of the system was the incorporation of the initial and final verniers giving a printout of the residual range to .1 nanosecond. This figure is unrealistic at present in terms of overall system accuracy but does represent a reasonable assessment of the potential accuracy of the timing electronics. (See fig.6)

Finally, two programs were written for calibration and testing of the system. The first, which is entered by the command K, reads the Analog-to-Digital converter. Each time a numeric key on the teletype is hit, the computer sends out EXC 060, 160 and 260 pulses and then waits for one of the Analog-to-Digital Converter flags to come up. When a flag is detected, the converted value is read in and stored along with the digit which initiated the cycle. This reading is also converted to decimal form and typed out. When a non-numeric key is hit the accumulated reading are written on tape and control returned to the command interpreter. This routine can be used to establish and check conversion curves for the Time-to-Pulse-Height Converters. Once this is done the TPHC's can be used to measure any short time intervals during calibration.

The second change modifies the system to allow a new mode of operation in which the value of DELTA stored in core rather than a range read from tape is used as the projected range. The transition between this and the normal mode is under control of sense switch 2 with sense switch 3 used to inhibit writing on the output tape. This provision for a constant operator-selectable range will greatly simplify the testing of program modifications and, in conjunction with a fixed precision delay, will allow for easy calibration checks.

Also desirable would be a program to summarize the return data from a run and present it in a useful form, thus facilitating decisions on changes for the next run.

FIGURE CAPTIONS

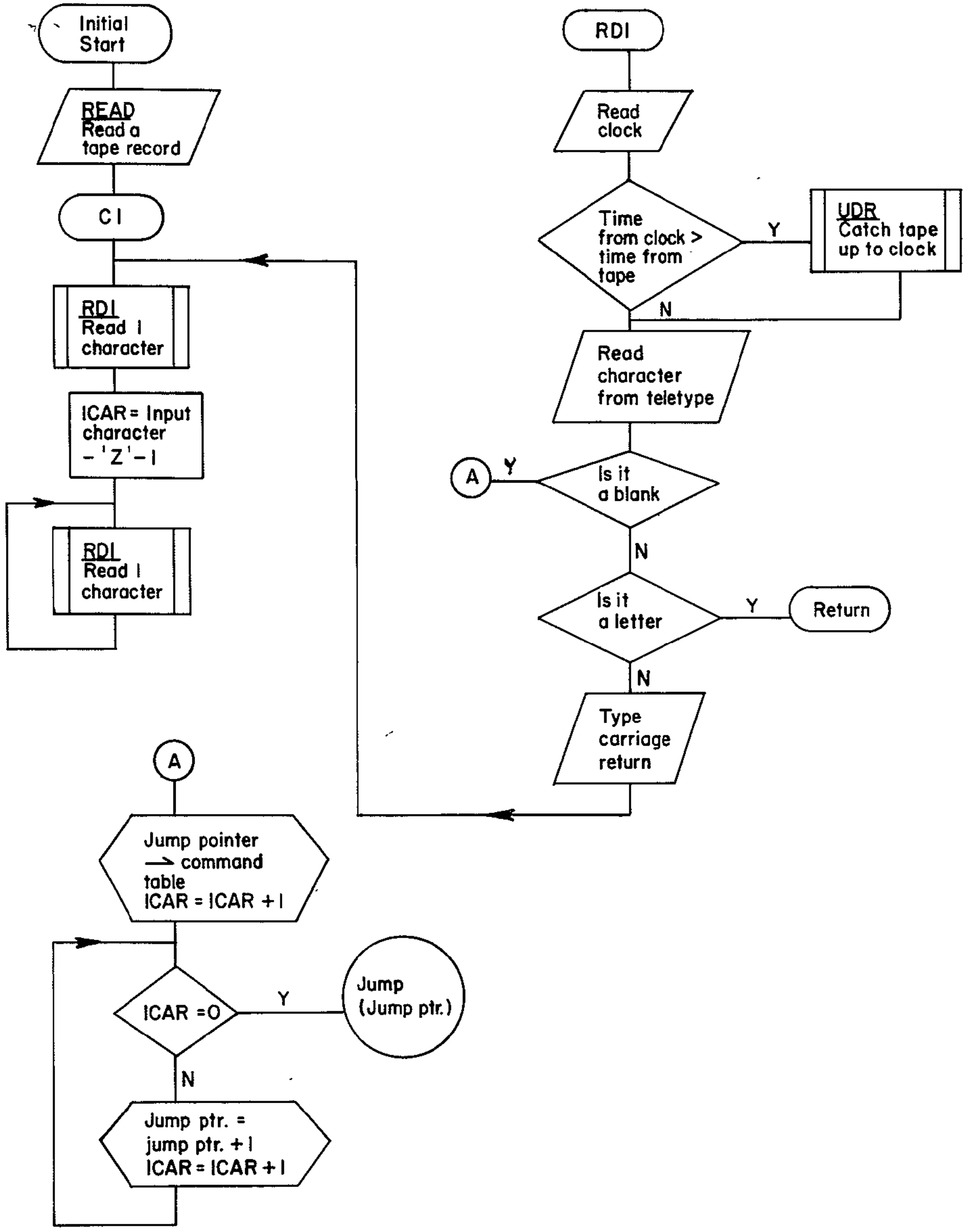
Initial Start
READ Read a tape record
CI
RDI Read I character
ICAR = Input character − 'Z' − I
RDI Read I character
A
Jump pointer → command table ICAR = ICAR + I
ICAR = O
Y
Jump (Jump ptr.)
N
Jump ptr. = jump ptr. + I ICAR = ICAR + I
RDI
Read clock
Time from clock > time from tape
Y
UDR Catch tape up to clock
N
Read character from teletype
A
Y
Is it a blank
N
Is it a letter
Y
Return
N
Type carriage return

Fig. I COMMAND INTERPERTER

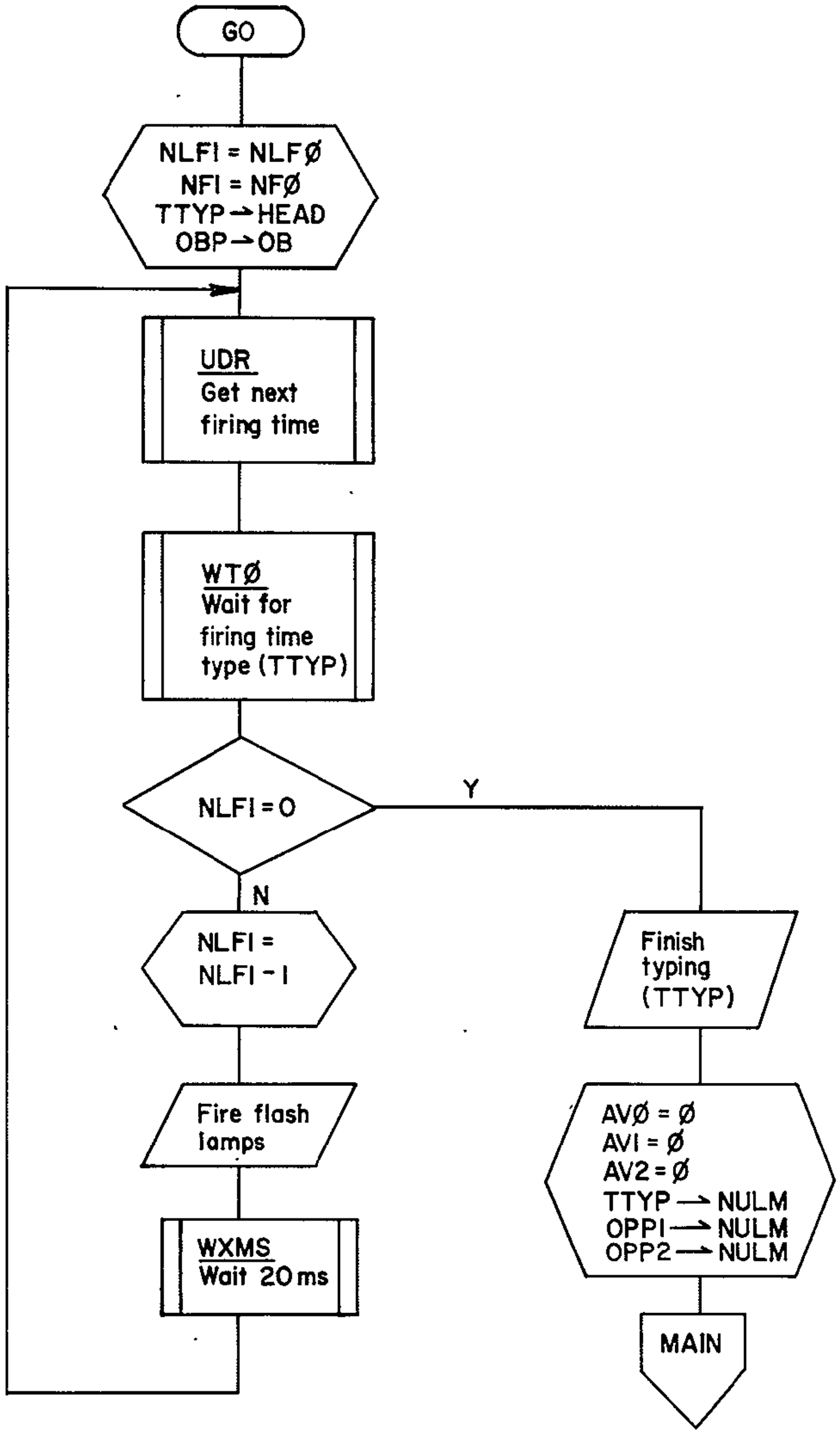

Fig. 2 MAIN PROGRAM - INITIALIZATION

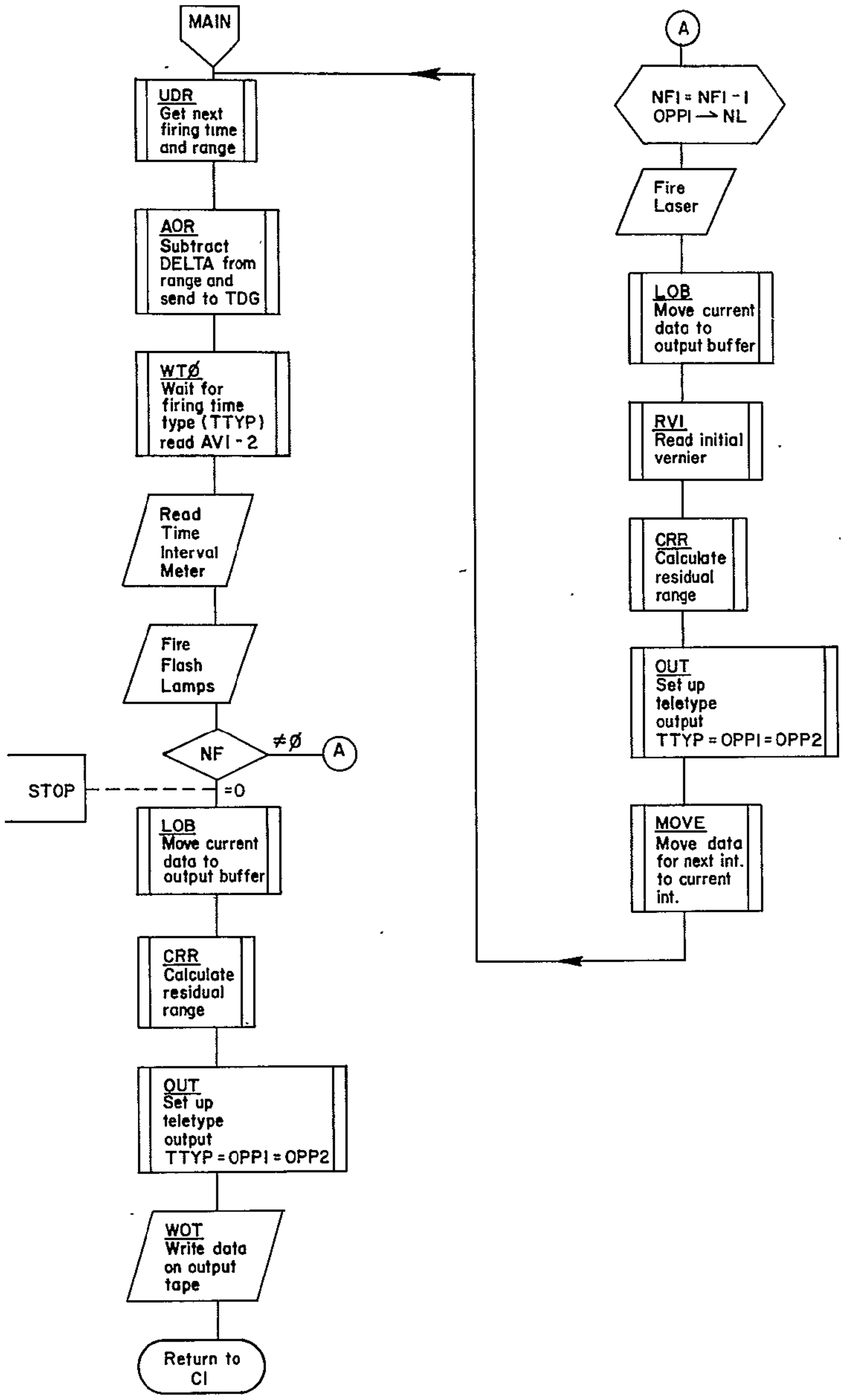
MAIN
UDR Get next firing time and range
AOR Subtract DELTA from range and send to TDG
WTØ Wait for firing time type (TTYP) read AVI - 2
Read Time Interval Meter
Fire Flash Lamps
NF
≠Ø
A
=0
STOP
LOB Move current data to output buffer
CRR Calculate residual range
OUT Set up teletype output TTYP = OPP1 = OPP2
WOT Write data on output tape
Return to CI
A
NF1 = NF1 - 1
OPP1 → NL
Fire Laser
LOB Move current data to output buffer
RVI Read initial vernier
CRR Calculate residual range
OUT Set up teletype output TTYP = OPP1 = OPP2
MOVE Move data for next int. to current int.

Fig. 3 MAIN PROGRAM - BODY

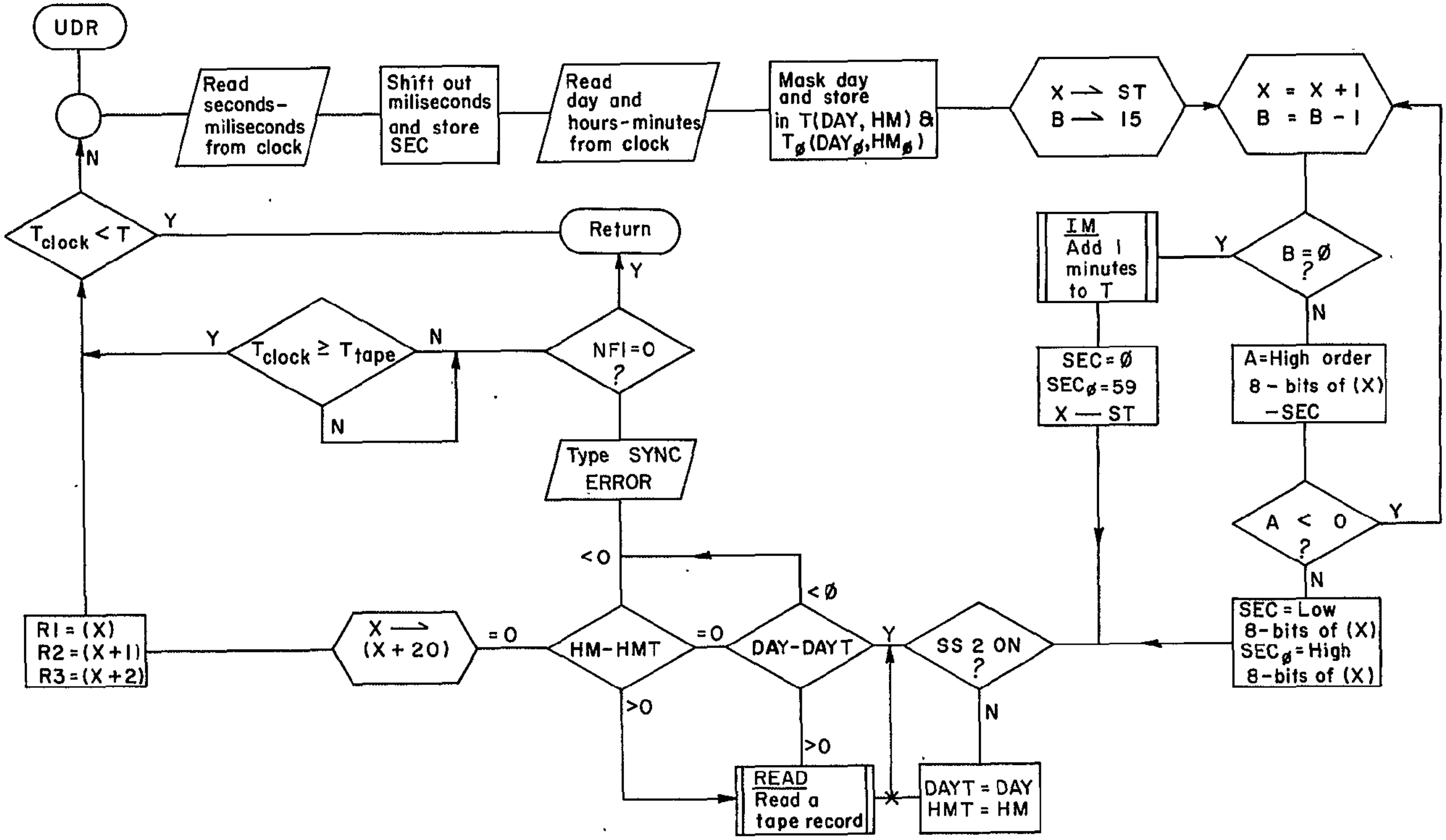

Fig. 4 UPDATE RANGE

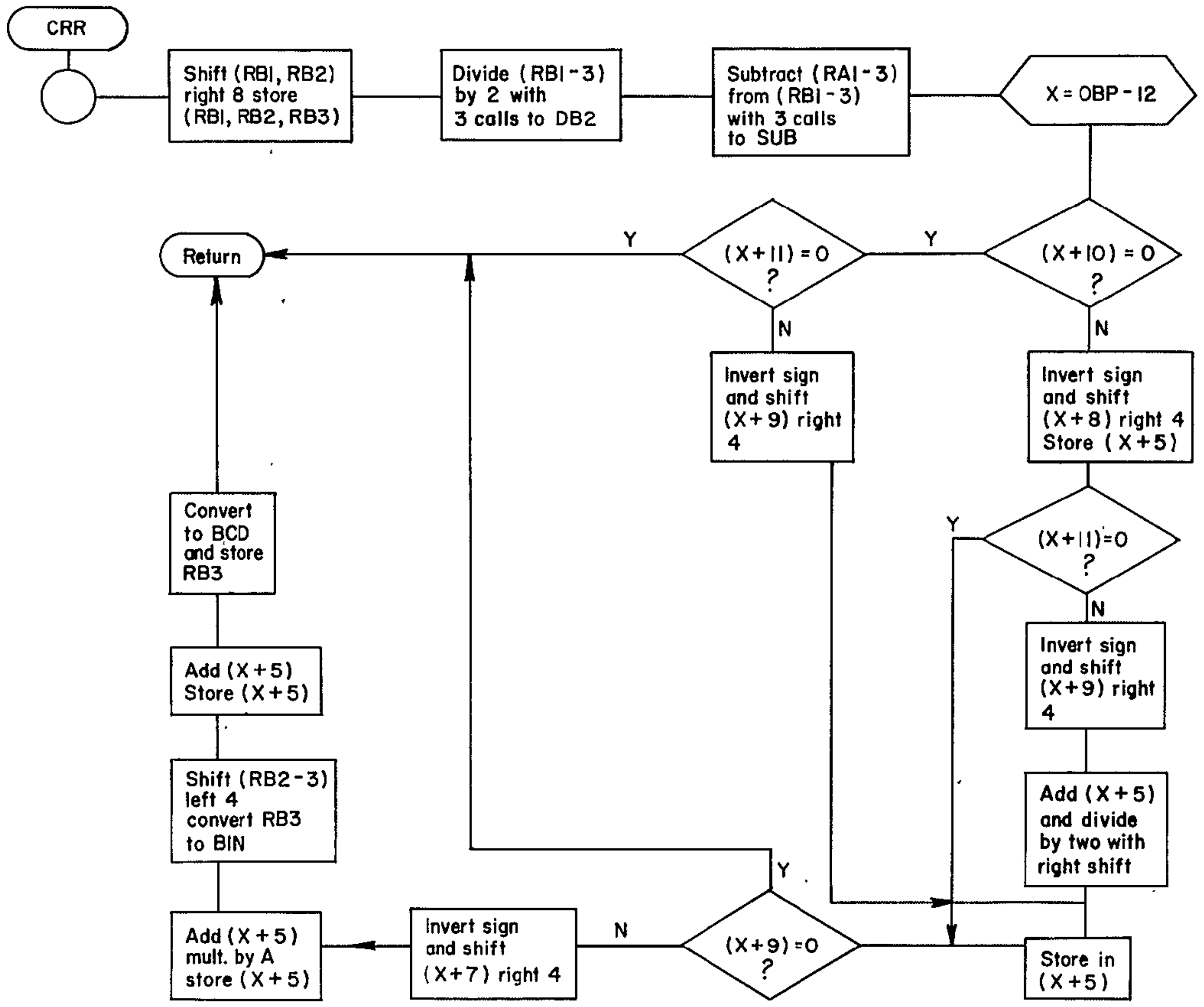
CRR
Shift (RB1, RB2) right 8 store (RB1, RB2, RB3)
Divide (RB1-3) by 2 with 3 calls to DB2
Subtract (RA1-3) from (RB1-3) with 3 calls to SUB
X = OBP-12
(X+10) = 0 ?
Y
N
(X+11) = 0 ?
Y
N
Return
Invert sign and shift (X+8) right 4 Store (X+5)
Invert sign and shift (X+9) right 4
(X+11)=0 ?
Y
N
Invert sign and shift (X+9) right 4
Add (X+5) and divide by two with right shift
Store in (X+5)
(X+9) = 0 ?
Y
N
Invert sign and shift (X+7) right 4
Add (X+5) mult. by A store (X+5)
Shift (RB2-3) left 4 convert RB3 to BIN
Add (X+5) Store (X+5)
Convert to BCD and store RB3

Fig. 5 CALCULATE RESIDUAL RANGE

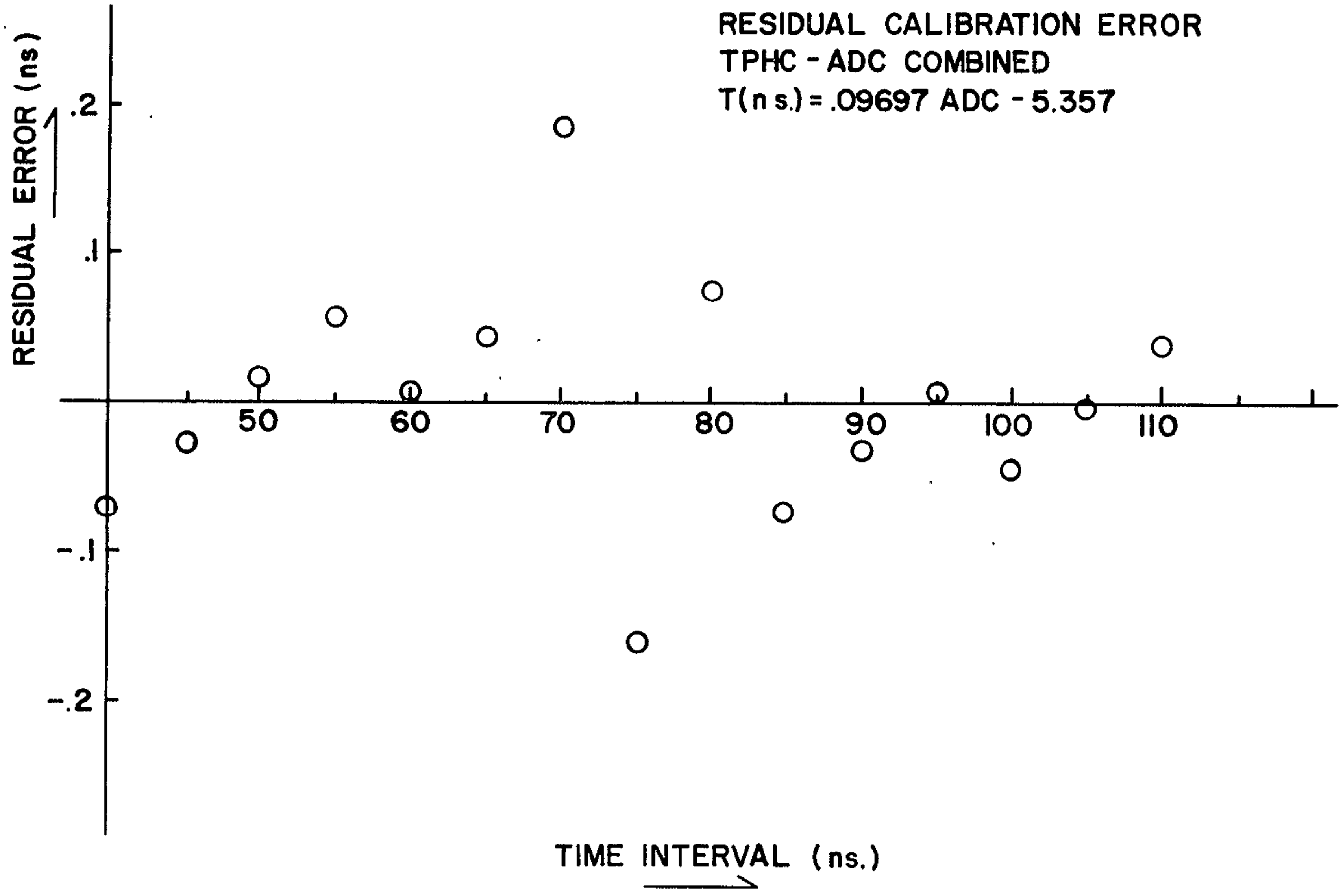
RESIDUAL CALIBRATION ERROR
TPHC - ADC COMBINED
T(n s.) = .09697 ADC - 5.357
RESIDUAL ERROR (ns)
.2
.1
-.1
-.2
50
60
70
80
90
100
110
TIME INTERVAL (ns.)

Fig. 6

APPENDIX I

RANGE TAPE FORMAT

The input range tape is a 7 track 556 character 1 inch IBM compatable magnetic tape. A 16 bit Varian data word is made up of three 6-bit characters with the two high order bits of the first character dropped.

There is one record for each minute of predictions. The first two words are the day, hour and minute corresponding to the predictions. Following this are the predicted ranges - three words per range. Records with an odd number of words then have one word of zeroes for compatability with the 36 bit word size of the 1108. All this data is in the form of four 4-bit BCD digits per word.

Word	Contents
1	DAY
2	Hour, Minute
3	Range 1 1
4	Range 1 2
5	Range 1 3
6	Range 2 1
7	Range 2 2
8	Range 2 3
3N	Range N 1
3N + 1	Range N 2
3N + 2	Range N 3

APPENDIX II

DATA TAPE FORMAT

The output data tape is a 7 track 556 character / inch IBM compatible magnetic tape. A 16 bit Varian data word is stored in three 6-bit characters with the two high order bits of the first character unused.

The results of each run are stored in one variable length record. The first three words of the record hold the current value of DELTA. Following that is the raw data, one 12 word block per shot.

Run Record

Word	Contents	Remarks
1	DELTA 1	BCD
2	DELTA 2	BCD
3	DELTA 3	BCD
4	DAY	BCD
5	HM	Hour, minute BCD
6	SEC	BCD
7	RB 1	Time Interval Meter 1 BCD
8	RB 2	Time Interval Meter 2 BCD
9	RB 3	Calculated Range BIN
10	RA 1	Projected Range 1 BCD
11	RA 2	Projected Range 2 BCD
12	RA 3	Projected Range 3 BCD
13	AV 0	Initial Vernier BIN

14	AV 1	Final Vernier 1 BIN
15	AV 2	Final Vernier 2 BIN
16	DAY	Beginning of next shot
	etc	

Calibration Record

1	177777_8	Flag
2	177777_8	Flag
3	1	Identification
4	CODE	Measurement type code
5	ADC	ADC reading (unconverted)
6	CODE	
7	ADC	
	etc	As many additional sets as desired

APPENDIX III.

Program Listing

```
*SYMBOLS
*
*TELETYPE
TTY ,EQU,01
TTO ,EQU,0101
TTI ,EQU,0201
*TAPE DECK 1
DSR1,EQU,010
WPS1,EQU,010
EOT1,EQU,0110
RBR1,EQU,0210
RPE1,EQU,0310
EOF1,EQU,0410
EOR1,EQU,0510
IRG1,EQU,010
WFZ1,EQU,0110
RR1 ,EQU,0210
BSR1,EQU,0310
*TAPE DECK 2
DSR2,EQU,011
VBR2,EQU,0610
IRG2,EQU,0410
*TIME DELAY GENERATOR
TDG1,EQU,062
TDG2,EQU,061
TDG3,EQU,060
LTDG,EQU,060
*TIME INTERVAL METER
TIM1,EQU,064
TIM2,EQU,063
*CLOCK
DCB ,EQU,060
DB  ,EQU,062
HMB ,EQU,061
SMB ,EQU,060
*ANALOG TO DIGITAL CONVERTER
ADC ,EQU,065
ADCF,EQU,0465
ADC0,EQU,065
ADC1,EQU,0165
ADC2,EQU,0265
*LASER
FFT ,EQU,0160
EPC ,EQU,0260
    ,EJEC,
    ,MORE,
*LOW CORE
*INITIALIZATION AND POINTERS
PNTR,BEGI,0
    ,USE ,PNTR
OB  ,EQU ,010000
    ,HLT ,
    ,CALL,READ,TIB,-47
    ,LDX ,=(SYS)
    ,CALL,TYP1
    ,JMP ,CI
TTYP,PZE,
OPP1,PZE,
OPP2,PZE,
OBP ,PZE,
DPTR,PZE,
*INTERRUPT SERVICE
INT ,BEGI,040
    ,USE ,INT
    ,HLT,
    ,HLT,
    ,JMP,CI
*COMMON STORAGE
    ,USE,COMN
DAY ,PZE,
DAYO,PZE,
HM  ,PZE,
HMO ,PZE,
SEC ,PZE,
SECO,PZE,
R1  ,PZE,
R2  ,PZE,
R3  ,PZE,
DR1 ,PZE,
DR2 ,PZE,
DR3 ,PZE,
NLFO,PZE,
NLF1,PZE,
NFO ,PZE,
NF1 ,PZE,
ICAR,PZE,
INUM,PZE,
ATNS,DATA,075755
DAY1,PZE,
HM1 ,PZE,
SEC1,PZE,
RB1 ,PZE,
RB2 ,PZE,
RB3 ,PZE,
RA1 ,PZE,
RA2 ,PZE,
RA3 ,PZE,
AVO  ,PZE,
AV1  ,PZE,
AV2  ,PZE,
*          SECONDS TABLE 4 SEC   BCD
ST   ,DATA,01404,03410,010422,012426,014440,021444
     ,DATA,023450,030462,032466,034500,041504
     ,DATA,043510,050522,52526,0,0,0,0,0

*POINTERS TO RANGE TABLE
RT   ,DATA,RT1,RT1+3,RT1+2*3,RT1+3*3,RT1+4*3,RT1+5*3
     ,DATA,RT1+6*3,RT1+7*3,RT1+8*3,RT1+9*3,RT1+10*3,RI1+11*3
     ,DATA,RT1+12*3,RT1+13*3,RT1+14*3,RT1+15*3,RT1+16*3
     ,DATA,RT1+17*3,RT1+18*3,RT1+19*3
TIB  ,NULL,
DAYT,PZE,
HMT  ,PZE
RT1  ,BSS,60
     ,EJEC,
     ,MORE,
*CONTROL TABLE
CT,PZE,CERR Z
   ,PZE,CERR Y
   ,PZE,CERR X
   ,PZE,RDW  W
   ,PZE,CERR V
   ,PZE,CERR U
   ,PZE,TIME T
   ,PZE,STOP S
   ,PZE,CERR R
   ,PZE,CERR Q
   ,PZE,POLT P
   ,PZE,CERR O
   ,PZE,CERR N
   ,PZE,CERR M
   ,PZE,CERR L
   ,PZE,KAL  K
   ,PZE,CERR J
   ,PZE,CERR I
   ,PZE,CERR H
   ,PZE,GO   G
   ,PZE,RDF  F
   ,PZE,CERR E
   ,PZE,RDD  D
   ,PZE,FLNT C
   ,PZE,CERR B
   ,PZE,015000
   ,EJEC,
    ,MORE,
*MESSAGES
MESG,BEGI,03000
    ,USE ,MESG
SEM ,DATA,' S Y N C   E R R O R'
CRLF,DATA,0215,0212
NULM,PZE ,
TE1 ,DATA,' S H O R T   R E C .'
    ,DATA,0215,0212,0
TE2 ,DATA,' R E D U N D A N C Y'
    ,DATA,0215,0212,0
HEAD,DATA,' D E L T A ='
DOUT,DATA,0,0,0,0,0,0,0,0,0,0,' T I M E ='
TOUT,DATA,' D D D : H H : M M',0215,0212,0
KOUT,DATA,'   D D D D',0
ROUT,DATA,' : S S  '
    ,DATA,' R R R R R R R R R R  '
    ,DATA,' D D D D . D',0215,0212,0
NL  ,DATA,' N O   L A S E R',0215,0212,0
NR  ,DATA,' N O   R E T U R N',0215,0212,0
SYS ,DATA,' L U N A R   R A N G I N G  '
    ,DATA,' C O N T R O L   P R O G R A M  '
    ,DATA,'   - -   V E R S I O N   9 . 1  '
    ,DATA,' 1 2 / 1 5 / 6 9',0215,0212,0
    ,EJEC,
    ,MORE,
*COMMAND INTERPERTER
      ,USE,
CI   ,CALL,RD1
     ,STA ,ICAR
RDCL,CALL,RD1
     ,JMP ,RDCL
CLOK,LDA ,ICAR
     ,LDX ,=(CT-1)*
     ,STX ,CJMP+3
CJMP,IAR ,
     ,INR ,CJMP+3
     ,JAP ,CERR
     ,JMP ,CJMP
*READ AND CHECK CONTROLE CHARACTERS
RD1,ENTR,
     ,CALL,RD2
     ,SUB ,=' '
     ,JAZ ,CLOK
     ,SUB ,='A'-' '
     ,JAN ,CERR
     ,SUB ,='Z'-'A'+1
     ,JAP ,CERR
     ,JMP*,RD1
*CHECK CLOCK FOR TAPE SYNC
WFT ,ENTR ,
     ,SEN,DCB,*
     ,CIA,DB
     ,ANA,=07777
     ,SUB,DAYT
     ,JAZ ,WFM
     ,JAPM,UDR
     ,JMP*,WFT
WFM,SEN ,DCB,*
     ,IME ,HMB,WFMT
     ,LDA ,HMT
     ,SUB ,WFMT
     ,JAP*,WFT
     ,CALL,UDR
     ,JMP*,WFT
WFMT,PZE ,
CERR,LDX ,=(CRLF)
     ,CALL,TYP1
     ,JMP ,CI
     ,EJEC,
*READ A NUMBER AND CONVERT TO BINARY
RDNB,ENTR,
    ,TZA ,
    ,STA ,INUM
    ,CALL,RD2
    ,SUB ,='9'+1
    ,JAP ,ENDN
    ,ADD ,=10
    ,JAN ,ENDN
    ,STA ,ICAR
    ,LDA ,INUM
    ,LRLA,2
    ,ADD ,INUM
    ,LRLA,1
    ,ADD ,ICAR
    ,JMP ,RDNB+2
ENDN,LDX ,=CRLF
    ,CALL,TYP1
    ,LDA ,INUM
    ,JMP*,RDNB
*READ A CHARACTER
RD2 ,ENTR,
    ,CALL,WFT
    ,SEN ,TTI,*+4
    ,JMP ,RD2+1
    ,CIA ,TTY
    ,OAR ,TTY
    ,JMP*,RD2
*READ NO. OF WARMUPS
RDW ,CALL,RDNB
    ,STA ,NLFO
    ,JMP ,CERR
*READ NO. OF FINRINGS
RDF ,CALL,RDNB
    ,STA ,NFO
    ,JMP ,CERR
```

```
*READ DELTA (BCD-3 WORDS)
RDD ,LDA ,=DSTK
    ,STA ,DPTR
    ,LDA ,=(CERR)
    ,STA ,RDDR+1
    ,CALL,RD2
    ,SUB ,='+'
    ,JAZ ,RDD1
    ,SUB ,='-'-'+'
    ,JAZ  ,RDDM
    ,SUB ,='9'-'-'+1
    ,JMP ,RDD1+3
RDDM,LDA ,=(RDDN)
    ,STA ,RDDR+1
RDD1,CALL,RD2
    ,SUB ,='9'+1
    ,JAP ,ENDD
    ,ADD ,=10
    ,JAN ,ENDD
    ,STA*,DPTR
    ,INR ,DPTR
    ,JMP ,RDD1
ENDD,LDA ,DPTR
    ,SUB ,=10
    ,TAX ,
    ,TZA ,
    ,CALL,PKD
    ,CALL,PKD
    ,STA ,DR1
    ,CALL,PKD4
    ,STA ,DR2
    ,CALL,PKD4
    ,STA ,DR3
RDDR,JMP ,CERR
RDDN,LDA ,DR1
    ,STA ,DSTK
    ,LDA ,DR2
    ,STA ,DSTK+1
    ,LDA ,DR3
    ,STA ,DSTK+2
    ,LDX ,=(DSTK-1)
    ,CALL,SUB
    ,STA ,DR3
    ,CALL,SUB
    ,STA ,DR2
    ,CALL,SUB
    ,STA ,DR1
    ,JMP ,CERR
    ,DUP ,10
    ,PZE ,
DSTK,BSS ,10
```

```
PKD ,ENTR,
    ,LRLA,4
    ,ORA ,0,1
    ,IXR ,
    ,JMP*,PKD
PKD4,ENTR,
    ,TZA ,
    ,CALL,PKD
    ,CALL,PKD
    ,CALL,PKD
    ,CALL,PKD
    ,JMP*,PKD4
    ,EJEC,
    ,MORE,
*SET UP RUN
GO   ,LDA ,NLFO
     ,STA ,NLF1
     ,LDA ,NFO
     ,STA ,NF1
     ,LDA ,=(OB)
     ,STA ,OBP
     ,LDX ,=(DOUT)
     ,LDB ,DR1
     ,STB*,OBP
     ,INR ,OBP
     ,LRLB,8
     ,CALL,PAK2
     ,LDB ,DR2
     ,STB*,OBP
     ,INR ,OBP
     ,CALL,PAK4
     ,LDB ,DR3
     ,STB*,OBP
     ,INR ,OBP
     ,CALL,PAK4
     ,LDA ,=(HEAD)
     ,STA ,TTYP
     ,LDX ,=(TOUT)
     ,LDB ,DAYO
     ,LLRL,4
     ,CALL,PAK2
     ,CALL,PACK
     ,IXR ,
     ,LDB ,HMO
     ,CALL,PAK2
     ,IXR ,
     ,CALL,PAK2
     ,IXR ,
*FIRE WARM UP SHOTS
FLNT,LDA ,NLF1
     ,JAZ ,SET
     ,IAR ,
     ,STA ,NLF1
     ,CALL,UDR
     ,CALL,WTO
     ,JS1M,EFT
     ,CALL,WXMS
     ,JMP ,FLNT
SET  ,LDX ,TTYP
     ,CALL,TYP1
     ,STX ,TTYP
     ,STX ,OPP1
     ,STX ,OPP2
     ,TZA ,
     ,STA ,AVO
     ,EJEC,
*MAIN LOOP
MAIN,CALL,UDR
    ,CALL,AOR
    ,CALL,WTO
    ,IME ,TIM1,RB1
    ,IME ,TIM2,RB2
    ,JS1M,EFT
    ,JS1M,FIRE
    ,CALL,LOB
    ,CALL,RVO
    ,CALL,CRR
    ,CALL,OUT
    ,CALL,MOVE
    ,JMP ,MAIN
    ,EJEC,
**ENABLE FLASH TUBE
EFT ,ENTR,
    ,EXC ,FFT
    ,JMP*,EFT
*FIRE LASER
FIRE,ENTR,
    ,LDA ,NF1
    ,JAZ ,STOP
    ,DAR ,
    ,STA ,NF1
    ,EXC ,LTDG
    ,EXC ,EPC
    ,LDA ,=(NL)
    ,STA ,OPP2
    ,JMP*,FIRE
*CLOSE OFF RUN
STOP,CALL,LOB
    ,CALL,CRR
    ,CALL,OUT
    ,LDX ,TTYP
    ,CALL,TYP1
    ,LDA ,OBP
    ,SUB ,=(OB)
    ,LDX ,=(OB)
    ,JS3M,WOT
    ,JMP ,CERR
*LOAD TAPE OUTPUT BUFFER
LOB ,ENTR,
    ,LDX ,=(DAY1)
    ,LDA ,10,1
    ,ADD ,11,1
    ,JAZ*,LOB
    ,LDA ,=-12
LOBL,LDB ,0,1
    ,STB*,OBP
    ,IXR ,
    ,IAR ,
    ,INR ,OBP
    ,JAN ,LOBL
    ,JMP*,LOB
    ,EJEC,
    ,MORE,
*UPDATE RANGE SUBROUTINE
UDR ,ENTR,
    ,SEN ,DCB,*             READ IN TIME
    ,CIA ,DB
    ,ANA ,=07777
    ,STA ,DAY
    ,STA ,DAYO
    ,SEN ,DCB,*
    ,CIA ,HMB
    ,STA ,HM
    ,STA ,HMO
    ,SEN ,DCB,*
    ,CIA ,SMB
    ,LSRA,8
    ,STA ,SECO
    ,LDX ,=(ST-1)
    ,LDB ,=-15
UDR1,IXR ,
    ,IBR ,       INCREMENT B+X
    ,JBZ ,NM
    ,LDA ,0,1
    ,ANA ,=0377
    ,SUB ,SECO
    ,JAN ,UDR1
    ,LDA ,0,1
    ,TAB ,
    ,ANA ,=0377
    ,STA ,SEC
    ,LSRB,8
    ,STB ,SECO
    ,JMP ,
NM  ,CALL,IM
    ,JSS2,UDR2
    ,LDA ,DAY
    ,STA ,DAYT
    ,LDA ,HM
    ,STA ,HMT
UDR2,LDA ,DAY
    ,SUB ,DAYT
    ,JAZ ,UDR3
    ,JAP ,NXTR
     ,JMP ,SE
UDR3,LDA ,HM
    ,SUB ,HMT
    ,JAZ ,SR
    ,JAP ,NXTR
S
```

```
SE   ,LDX ,=(SEM)
   ,CALL,TYP1
   ,LDA ,NF1
    ,JAZ*,UDR
WFD ,SEN ,DCB,*
    ,CIA ,DB
    ,ANA ,=07777
    ,SUB ,DAYT
    ,JAN ,WFD
WHM ,SEN ,DCB,*
    ,CIA ,HMB
    ,SUB ,HMT
    ,JAN ,WHM
SR   ,LDX ,20,1
    ,LDA ,0,1
    ,STA ,R1
    ,LDA ,1,1
    ,STA ,R2
    ,LDA ,2,1
    ,STA ,R3
UCHK,SEN ,DCB,*
    ,CIA ,DB
    ,ANA ,=07777
    ,SUB ,DAYO
    ,JAZ ,CHM
    ,JMP ,UDR+1
CHM ,SEN,DCB,*
    ,CIA ,HMB
    ,SUB ,HMO
    ,JAZ ,CSEC
    ,JMP ,UDR+1
CSEC,SEN ,DCB,*
    ,CIA ,SMB
    ,LSRA,8
    ,SUB ,SECO
    ,JAP ,UDR+1
    ,JMP*,UDR
NXTR,CALL,READ,TIB,-47
    ,JMP ,UDR2
    ,EJEC,
    ,MORE,
*ADJUST AND OUTPUT RANGE
AOR ,ENTR,
    ,JSS2,AOR1
    ,LDA ,DR1
    ,STA ,R1
    ,OAR ,TDG1
    ,LDA ,DR2
    ,STA ,R2
    ,OAR ,TDG2
    ,LDA ,DR3
    ,STA ,R3
    ,OAR ,TDG3
    ,JMP*,AOR
AOR1,LDX ,=(R3)
    ,ROF ,
    ,CALL,SUB
    ,STA ,RB3
    ,OAR ,TDG3                  , SEND LOW ORDER BCD TO TDG
    ,CALL,SUB
    ,STA ,RB2
    ,OAR ,TDG2                    TDG2
    ,CALL,SUB
    ,STA ,RB1
    ,OAR ,TDG1                    HIGH ORDER TO TDG
    ,JMP*,AOR
*ONE WORD BCD SUBTRACTION
SUB ,ENTR,
    ,LDA ,0,1
    ,STA ,A
    ,LDA ,3,1
    ,STA ,B
    ,TZA ,
    ,STA ,C
    ,CALL,SUB1
    ,CALL,SUB1
    ,CALL,SUB1
    ,CALL,SUB1
    ,DXR ,
    ,JMP*,SUB
```

```
*ONE DIGIT BCD SUBTRACTION
SUB1,ENTR,
    ,LDA ,B
    ,TAB ,
    ,ANA ,=017
    ,STA ,B1
    ,LSRB,4
    ,STB ,B
    ,LDA ,A
    ,TAB ,
    ,ANA ,=017
    ,LSRB,4
    ,STB ,A
    ,SUB ,B1
    ,SOFA,
    ,ROF ,
    ,JAP ,SUB2
    ,ADD ,=10
    ,SOF ,
SUB2,ORA ,C
    ,LRLA,12
    ,STA ,C
    ,JMP*,SUB1
A   ,PZE ,0
B   ,PZE ,0
B1  ,PZE ,0
C   ,PZE ,0
    ,EJEC
    ,MORE,
*READ INITAL VERNIER
RVO ,ENTR,
    ,TZA ,
    ,STA ,AVO
    ,STA ,AV1
    ,STA ,AV2
    ,SEN ,DCB,*
    ,CIA ,SMB
    ,ANA ,=0377
    ,SUB ,=020
    ,JAZ*,RVO
    ,SEN ,ADCO,*+4
    ,JMP ,RVO+3
    ,EXC ,ADCO
    ,CALL,RADC
    ,STA ,AVO
    ,LDA ,=(NR)
    ,STA ,OPP2
    ,JMP*,RVO
*READ FINAL VERNIERS
RV12,ENTR,
    ,LDA ,AVO
    ,JAZ*,RV12
    ,SEN ,ADC1,RV1
    ,SEN ,ADC2,RV2
    ,JMP*,RV12
RV1 ,EXC ,ADC1
    ,CALL,RADC
    ,STA ,AV1
    ,JMP ,RV2+4
RV2 ,EXC ,ADC2
    ,CALL,RADC
    ,STA ,AV2
    ,LDA ,=(ROUT)
    ,STA ,OPP1
    ,JMP*,RV12
*READ ANALOG TO DIGITAL CONVERTER
RADC,ENTR,
    ,SEN ,ADCF,*+4
    ,JMP ,*+4
    ,SEN ,ADCF,*
    ,SEN ,ADCF,*+4
    ,JMP ,*-2
    ,CIA ,ADC
    ,JMP*,RADC
    ,EJEC,
    ,MORE,
*MOVE DATA TO INTERMEDIATE STORE
MOVE,ENTR,
    ,LDA ,DAY
    ,STA ,DAY1
    ,LDA ,HM
    ,STA ,HM1
    ,LDA ,SEC
    ,STA ,SEC1
    ,LDA ,R1
    ,STA ,RA1
    ,LDA ,R2
    ,STA ,RA2
    ,LDA ,R3
    ,STA ,RA3
    ,JMP*,MOVE
```

```
*CALCULATE RESIDUAL RANGE
CRR ,ENTR,
*GET RANGE TO 50 NS.
    ,TZB ,
    ,LDA ,RB1
    ,CPA ,
    ,LLSR,8
    ,STA ,RB1
    ,LDA ,RB2
    ,CPA ,
    ,STB ,RB2
    ,TZB ,
    ,LLSR,8
    ,ORA ,RB2
    ,STA ,RB2
    ,STB ,RB3
*DIVIDE TIM BY 2
    ,TZB ,
    ,STB ,DC
    ,LDA ,RB1
    ,CALL,DB2
    ,STA ,RB1
    ,LDA ,RB2
    ,CALL,DB2
    ,STA ,RB2
    ,LDA ,RB3
    ,CALL,DB2
    ,STA ,RB3
*SUBTRACT PROJECTED RANGE
    ,LDX ,=(RB3)
    ,ROF ,
    ,CALL,SUB
    ,STA ,RB3
    ,CALL,SUB
    ,STA ,RB2
    ,CALL,SUB
    ,STA ,RB1
*CONVERT VERNIER TO NS.
    ,LDA ,OBP
    ,SUB ,=12
    ,TAX ,
    ,TZA ,
    ,STA ,5,1
    ,LDA ,10,1
    ,JAZ ,CAV2
    ,ERA ,=0100000
    ,ASRA,4
    ,STA ,5,1
    ,LDA ,11,1
    ,JAZ ,CAV0
    ,ERA ,=0100000
    ,ASRA,4
    ,ADD ,5,1
    ,ASRA,1
    ,STA ,5,1
    ,JMP ,CAV0
```

```
CAV2,LDA ,11,1
    ,JAZ*,CRR
    ,ERA ,=0100000
    ,ASRA,4
    ,STA ,5,1
CAV0,LDA ,9,1
    ,ERA ,=0100000
    ,ASRA,4
    ,SUB ,5,1
    ,TAB ,
    ,TZA ,
    ,CALL,XMUL,ATNS
    ,STA ,5,1
*ADD TIM DATA*10
    ,LDA ,RB2
    ,LDB ,RB3
    ,LLRL,4
    ,STA ,RB2
    ,TBA ,
    ,CALL,XDTB
    ,TBA ,
    ,ADD ,5,1
    ,CALL,XBTD
    ,STB ,RB3
    ,LDA ,RB2
    ,LLRL,12
    ,CALL,XDTB
    ,STB ,5,1
    ,JMP* ,CRR
*BCD DIVIDE BY 2
DB2 ,ENTR,
    ,TAB ,
    ,LSRA,1
    ,ANA ,=073567
    ,STA ,DB2T
    ,TZA ,
    ,LLRL,15
    ,ANA ,=04210
    ,ORA ,DC
    ,STB ,DC
    ,LSRA,1
    ,TAB ,
    ,LSRB,2
    ,MERG,031
    ,ADD ,DB2T
    ,JMP*,DB2
DC  ,PZE ,
DB2T,PZE ,
    ,EJEC,
    ,MORE,
*SET UP TELETYPE OUTPUT
OUT ,ENTR,
*PUSH POINTERS
    ,LDA ,OPP1
    ,STA ,TTYP
    ,LDA ,OPP2
    ,STA ,OPP1
    ,LDA ,=(NULM)
    ,STA ,OPP2
    ,LDX ,=(ROUT)

    ,LDB ,SEC1
    ,LLRL,8
    ,CALL,PAK2
    ,IXR ,
    ,LDB ,RA1
    ,LLRL,8
    ,CALL,PAK2
    ,LDB ,RA2
    ,CALL,PAK4
    ,LDB ,RA3
    ,CALL,PAK4
    ,IXR ,
    ,LDA ,RB2
    ,LLSR,4
    ,CALL,PACK
    ,LDB ,RB3
    ,CALL,PAK2
    ,CALL,PACK
    ,IXR ,
    ,CALL,PACK
    ,JMP*,OUT
PACK,ENTR,
    ,TZA ,
    ,LLRL,4
    ,ORA ,=0260
    ,STA ,0,1
    ,IXR ,
    ,JMP*,PACK
PAK2,ENTR,
    ,CALL,PACK
    ,CALL,PACK
    ,JMP*,PAK2
PAK4,ENTR,
    ,CALL,PAK2
    ,CALL,PAK2
    ,JMP*,PAK4
    ,EJEC,
    ,MORE,
*WAIT FOR TO
WTO ,ENTR,
WCK1,SEN ,DCB,WCK1
    ,CIA ,DB                    CHECK DAY
    ,ANA ,=07777
    ,SUB ,DAYO
    ,JAZ ,WCK2
    ,JMP ,MTO
WCK2,SEN ,DCB,WCK1
    ,CIA ,HMB                   CHECK HOURS MINUTES
    ,SUB ,HMO
    ,JAZ ,WCK3
    ,JMP ,MTO
WCK3,SEN ,DCB,WCK3
    ,CIA ,SMB                   CHECK SECONDS
    ,LSRA,8
    ,SUB ,SECO
    ,JAZ ,WCK4
    ,JAP ,MTO
    ,CALL,JUNK
    ,JMP ,WCK3
WCK4,SEN ,DCB,WCK4
    ,CIA ,SMB                   CHECK MS.,TENS AND HUNDREDS
    ,ANA ,=0377
    ,SUB ,=0231
    ,JAZ ,WCK5
    ,JAP ,MTO
    ,CALL,JUNK
    ,JMP ,WCK4
WCK5,SEN ,DCB,WCK5
    ,CIA ,DB
    ,ANA ,=0170000              CHECK MS.,UNITS
    ,SUB ,=0100000              CHECK FOR 10 MS.
    ,JAZ*,WTO                   BEFORE T ZERO
    ,JMP ,WCK5
*MISSED TO
MTO ,LDA ,NLF1
    ,JAZ ,*+4
    ,JMP ,FLNT
    ,CALL,WXMS
    ,CALL,CRR
    ,CALL,OUT
    ,JMP ,MAIN
JUNK,ENTR,
    ,CALL,TYPE
    ,CALL,RV12
    ,SEN ,TTI,CERR
    ,JMP*,JUNK
    ,EJEC-

    ,MORE,
*TYPE OUT ROUTINE
TYPE,ENTR,
     ,SEN ,TTO,*+4
     ,JMP*,TYPE
     ,LDA*,TTYP
     ,JAZ*,TYPE
     ,OAR ,TTY
     ,INR ,TTYP
     ,JMP*,TYPE
*MESSAGE TYPING PROGRAM
TYP1,ENTR,
     ,SEN ,TTO,*+4
     ,JMP ,*-2
     ,LDA ,0,1
     ,JAZ*,TYP1
     ,OAR ,TTY
     ,IXR ,
     ,JMP ,TYP1+1
*WASTE 20 MS
WXMS,ENTR,
     ,STA ,XMS1
     ,STB ,XMS2
     ,STX ,XMS3
     ,LDA ,=-1851
WT   ,IAR ,
     ,LRLB,16
     ,NOP ,
     ,JAN ,WT
     ,LDA ,XMS1
     ,LDB ,XMS2
     ,LDX ,XMS3
     ,JMP*,WXMS
XMS1,PZE ,0
XMS2,PZE ,0
XMS3,PZE ,0
     ,EJEC,
     ,MORE,
*TAPE READING PROGRAM
*     CALL,READ,LOC,-NWRDS
TR1  ,LDA ,READ
     ,TAB ,
     ,ADD ,=2
     ,STA ,RXIT+1
     ,LDX ,0,2
     ,LDB ,1,2
     ,SEN ,EOR1,TR2
     ,JMP ,*-2
TR2  ,CALL,WXMS
     ,EXC ,RR1
     ,CALL,WTR
RC1  ,CIA ,DSR1
     ,LRLA,6
     ,CALL,WTR
RC2  ,INA ,DSR1
     ,LRLA,6
     ,CALL,WTR
RC3  ,INA ,DSR1
     ,STA ,0,1
     ,IBR ,
     ,IXR ,
     ,JBZ ,TR3
     ,JMP ,TR2+3
TR3  ,CALL,WTR
     ,JMP ,TR3
RTE  ,SEN ,RPE1,TERR
     ,LDX ,TRX
RXIT,JBZ*,READ
     ,LDX ,=(TE1)
     ,CALL,TYP1
     ,JMP ,TR1
READ,ENTR,
     ,STX ,TRX
     ,JMP ,TR1
*WAIT FOR TAPE READY
WTR  ,ENTR,
     ,SEN ,EOR1,RTE
     ,SEN ,RBR1,(WTR)*
     ,JMP ,WTR+1
TERR,LDX ,=(TE2)
     ,CALL,TYP1
     ,JMP ,TR1
TRX  ,PZE ,
     ,EJEC,
     ,MORE,
*WRITE OUTPUT TAPE
WOT ,ENTR,
    ,JAZ ,WOTE
    ,DAR ,
    ,STA ,WOTC
    ,LDB ,0,1
    ,TZA ,
    ,LLRL,4
    ,CALL,W1C
    ,LLRL,6
    ,CALL,W1C
    ,LLRL,6
    ,CALL,W1C
    ,IXR ,
    ,LDA ,WOTC
    ,JMP ,WOT+1
WOTE,SEN ,WBR2,*+4
    ,JMP ,WOTE
    ,EXC ,IRG2
    ,JMP*,WOT
*WRITE 1 CHARACTER
W1C ,ENTR,
    ,SEN,WBR2,*+4
    ,JMP ,*-2
    ,OAR ,DSR2
    ,JMP*,W1C
WOTC,PZE ,
    ,EJEC,
    ,MORE,.
```

```
*INCREMENT MINUTES SUBROUTINE
IM   ,ENTR,
     ,TZA ,
     ,LDX ,=(ST-1)
     ,STA ,SEC
     ,LDA ,=0131
     ,STA ,SECO
     ,LDA ,HM
     ,TAB ,
     ,ANA ,=0377
     ,SUB ,=0131
     ,JAZ ,IM2
     ,TBA ,
     ,ANA ,=017
     ,SUB ,=9
     ,JAZ ,IM1
     ,IBR ,
     ,STB ,HM
     ,JMP*,IM
IM1  ,TBA ,
     ,ADD ,=07
     ,STA ,HM
     ,JMP*,IM
IM2  ,TBA ,
     ,ANA ,=0177400
     ,SUB ,=021400
     ,JAZ ,IM4
     ,TBA ,
     ,ANA ,=0007400
     ,SUB ,=0004400
     ,JAZ ,IM3
     ,TBA ,
     ,ADD ,=0247
     ,STA ,HM
     ,JMP*,IM
IM3  ,TBA ,
     ,ADD ,=003247
     ,STA ,HM
     ,JMP*,IM
IM4  ,STA ,HM
     ,LDA ,DAY
     ,TAB ,
     ,ANA ,=017
     ,SUB ,=9
     ,JAZ ,IM5
     ,IBR ,
     ,STB ,DAY
     ,JMP*,IM
```

2

```
IM5 ,TBA ,
    ,ANA ,=0360
    ,SUB ,=0220
    ,JAZ ,IM6
    ,TBA ,
    ,ADD ,=07
    ,STA ,DAY
    ,JMP*,IM
IM6 ,TBA ,
    ,ADD ,=0147
    ,STA ,DAY
    ,JMP*,IM
    ,EJEC,
    ,MORE,
*MULTI-CHANEL ANALIZER SYMULATOR
POLT,CALL,RDNB
    ,JAZ ,CERR
    ,TAB.,
    ,SUB ,=2000
    ,JAP ,CERR
    ,STB ,NF1
    ,LDX ,=(OB)
    ,LDA ,=0177777
    ,STA ,0,1
    ,IXR ,
    ,STA ,0,1
    ,IXR ,
    ,ADD ,=3
    ,STA ,0,1
    ,IXR ,
PSAC,SEN ,ADC0,PRA0
    ,SEN ,ADC1,PRA1
    ,SEN ,ADC2,PRA2
    ,JMP ,PSAC
PRA0,EXC ,ADC0
    ,JMP ,PRAC
PRA1,EXC ,ADC1
    ,JMP ,PRAC
PRA2,EXC ,ADC2
PRAC,CALL,RADC
    ,STA ,0,1
    ,IXR ,
    ,DBR ,
    ,JBZ ,*+4
    ,JMP ,PSAC
    ,LDA ,=(OB)
    ,ADD ,=3
    ,STA ,OBP
PL1 ,LDX ,=(CRLF)
    ,CALL,TYP1
    ,LDA ,=9
    ,STA ,NLF1
PL2 ,LDA ,NF1
    ,DAR ,
    ,JAN ,STOP+11
    ,STA ,NF1
    ,LDA*,OBP
    ,INR ,OBP
    ,SUB ,=0177760
    ,JAP ,PL2
    ,ADD ,=077760
    ,ASRA,4
    ,CALL XBTD
    ,LDX ,=(KOUT)
    ,CALL,TYP1
    ,LDA ,NLF1
    ,DAR ,
    ,STA ,NLF1
    ,JAP ,PL2
    ,JMP ,PL1
    ,EJEC,
    ,MORE,
*CALIBRATION PROGRAM
KAL ,LDX ,=(OB)
    ,STX ,OBP
    ,LDA ,=0177777
    ,STA*,OBP
    ,INR ,OBP
    ,STA*,OBP
    ,INR ,OBP
    ,ADD ,=2
      ,STA*,OBP
    ,INR ,OBP
NXTK,SEN ,TTI,RDCD
    ,JMP ,NXTK
CLK1,SEN ,DCB,*
    ,CIA ,SMB
    ,AND ,=0377
    ,SUB ,=0231
    ,JAN ,CLK1
CLK2,SEN ,DCB,*
    ,CIA ,DB
    ,LSRA,12
    ,SUB ,=8
    ,JAN ,CLK2
    ,EXC ,060
    ,EXC ,0160
    ,EXC ,0260
SADC,SEN ,ADC0,RA0
    ,SEN ,ADC1,RA1
    ,SEN ,ADC2,RA2
    ,JMP ,SADC
RA0 ,EXC ,ADC0
    ,JMP ,RAC
RA1 ,EXC ,ADC1
    ,JMP ,RAC
RA2 ,EXC ,ADC2
RAC ,CALL,RADC
    ,STA*,OBP
    ,INR ,OBP
    ,ERA ,=0100000
    ,ASRA,4
    ,LDX ,=(KOUT)
    ,IXR ,
    ,CALL,XBTD
    ,CALL,PAK4
    ,LDX ,=(KOUT)
    ,CALL,TYP1
    ,LDX ,=(CRLF)
    ,CALL,TYP1
    ,JMP ,NXTK
```

```
RDCD,CIA ,TTY
    ,OAR ,TTY
    ,SUB ,='9'+1
    ,JAP ,STOP+11
    ,ADD ,=10
    ,JAN ,STOP+11
    ,STA*,OBP
    ,INR ,OBP
    ,JMP ,CLK1
    ,EJEC,
    ,MORE,
```

```
*OUTPUT CURRENT TAPE TIME
TIME,LDX ,=(TOUT)
    ,LDB ,DAYT
    ,LRLB,4
    ,CALL,PAK2
    ,CALL,PACK
    ,IXR ,
    ,LDB ,HMT
    ,CALL,PAK2
    ,IXR ,
    ,CALL,PAK2
    ,LDX ,=(TOUT)
    ,CALL,TYP1
    ,JMP ,CERR
    ,EJEC,
    ,MORE,
*       FIXED POINT MATH       16 BIT, NO MUL/DIV       VERSION 02.0
*
*   XMUL                       STANDARD SOFTWARE MULTIPLY
*
*              A,B>[B*PAR]<A
*              X IS UNCHANGED 40 WORDS
NBIT,EQU ,16
*
ADD    ,DATA   ,0124025        ADD MCND
ERA    ,DATA   ,0134023        ERA SIGN
SOF    ,DATA   ,0124013        ADD SIGN
SUBX   ,DATA   ,0144007        SUB CND
BGN    ,ROF    ,               RESET OF
       ,STX    ,XMXR           SAVE XR
       ,LDX    ,XMUL           GET ADDRASS OF CALL SEQ
       ,LDX    ,0,1            GET ADDR OF MCND
       ,LDX    ,0,1            GET MCND
       ,STX    ,MCND           AND SAVE
       ,LDX    ,K15            SET BIT COUNT
RPT    ,LLRL   ,31             A SIGN> LSB OF MPLR
       ,ADD    ,XSIG           SET OF IF LSB>1
       ,LLRL   ,1              ALIGN PARTIAL PRODUCT
       ,XOF    ,ADD            ADD MCND IF LSB>1
       ,LASR   ,1              AND SHIFT RIGHT
       ,XOF    ,ERA            INVERT SIGN IF OF
       ,DXR    ,               COUNT BITS DDEVELOPED
       ,JXZ    ,*+4            JMP IF DONE
       ,JMP    ,RPT            ELSE REPEAT
       ,LLRL   ,NBIT           A SIGN>MPLR SIGN
       ,XAN    ,SOF            SET OF IF NEG MPLR
       ,LLRL   ,NBIT           RESET PRODUCT
       ,XOF    ,SUBX           SUB MCND IF NEG MPLR
       ,INR    ,XMUL           SET RETURN
       ,LDX    ,XMXR           RESTORE XR
       ,JMP*   ,XMUL           A,B>B*M<A
XMUL   ,BES    ,0              ENTRY
       ,JMP    ,BGN
XMXR   ,BSS    ,1              TEMPORARY STORAGE
MCND   ,BSS    ,1
XSIG   ,DATA   ,0100000        CONSTANTS
K15    ,DATA   ,15
       ,MORE   ,
       ,END    ,
*       FIXED POINT MATH       16 BIT, NO MUL/DIV        VERSION 02.0
*
*   XDIV                       SOTWARE DIVIDE
*
*                 A,B/MB [QUOTIENT] < A [REMINDER]
*                 A REG MEMORY X IS UNCHANGED
*                 QUOTIENT IS ALWAYS TRUE
*                 REMAINDER IS SIGN OF DIVIDEND [UNLESS R>0]
*
TOP     ,STX     ,XR                SAVE XR
        ,DECR    ,4                 SET SIGN INDICATOR
        ,JAP     ,POSU              SET DIVIDEND POS
        ,CPX     ,                  SET DSIN>NEG
        ,CPB     ,                  LO ORDER TWO,S COMPL
        ,IBR     ,
        ,LRLB    ,1                 SIGN>0
        ,LSRB    ,1
        ,CPA     ,                  HI ORDER TWO,S COMPL
        ,JBZ     ,*+4
        ,JMP     ,*+3
        ,IAR     ,
POSU    ,STX     ,DSIN              SAVE DIVDN SIGN
        ,STA     ,DVDN              SAVE DIVDN
        ,LDX     ,XDIV              GET ADDR OF CALL SEQ
        ,LDX     ,0,1               GET ADDR OF PARAM
        ,LDA     ,0,1               GET DIVISOR
        ,LDX     ,DSIN              GET DIVDN SIGN
        ,JAP     ,*+5               SET DIVISOR POS
        ,CPX     ,                 SET QUOTIENT SIGN
        ,CPA     ,                 TWO,S COMPL
        ,IAR     ,
        ,STA     ,DVSR              SAVE DIVISOR
        ,LDA     ,DVDN              GET DIVDN
        ,STX     ,QSIN              SAVE QUOT SIGN
        ,LDX     ,K14               SET CYCLE COUNT
        ,LRLB    ,1                 ADJUST LO ORDER [DELETE SIGN]
        ,SUB     ,DVSR              SUB DIVSOR
        ,SOF     ,
        ,JAP     ,XERR              JMP IF OVERFLOW ERROR
        ,ROF     ,
NEGU    ,LLRL    ,1                 DEVELOP 14 QUOTIENT BITS
        ,ADD     ,DVSR              [NON RESTORING ALGORITHM]
TEST    ,JXZ     ,ADJ               JMP IF COMPLETE
        ,DXR     ,                  COUNT BITS
        ,JAN     ,NEGU              JUMP IF NEG REMAINDER
        ,LLRL    ,1                 SHIFT QUOTOREM
        ,SUB     ,DVSR              SUBTRACT DIVSR
        ,JMP     ,TEST              GO TEST
```

```
ADJ   ,LRLB   ,1            GET LAST QUOTIENT BIT
      ,JAP    ,*+4          JMP IF OR
      ,IBR    ,             ELSE SET LOB
      ,ADD    ,DVSR         RESTORE REMAINDER
      ,LDX    ,QSIN         GET TRUE QUOTIENT
      ,JXZ    ,*+4          JMP IF NEGATIVE QUOT
      ,CPB    ,             ELSE SET POSITIVE
      ,DBR    ,
      ,IBR    ,
      ,LDX    ,DSIN         GET TRUE REMAINDER
      ,JXZ    ,*+4          JMP IF REMAINDER NEG
      ,JMP    ,*+4          ELSE LEAVE POS
      ,CPA    ,
      ,IAR    ,
XERR  ,INR    ,XDIV         SET RETURN
      ,LDX    ,XR
      ,JMP*   ,XDIV         A,BOM- B>QUOT A>REM
XDIV  ,BES    ,0            ENTRY
      ,JMP    ,TOP
K14   ,DATA   ,14
XR    ,BSS    ,1            TEMP STORAGE
DVSR  ,BSS    ,1
DVDN  ,BSS    ,1
DSIN  ,BSS    ,1
QSIN  ,BSS    ,1
      ,MORE   ,
      ,END    ;
```

```
*      FIXED POINT MATH      16 BIT, NO MUL/DIV      VERSION 02.0
*
*  XDTB                    FIXED POINT INTEGER DEC TO BIN CONVERSION
*
XDTB  ,ENTR    ,
      ,STA     ,AA
      ,STX     ,AA+1
      ,TAB     ,
      ,LDXI    ,3              INITIALIZE COUNT
      ,TZA     ,
      ,STA     ,AA+2
      ,LLRL    ,4              GET NEXT DIGIT
      ,STB     ,AA+3
      ,LDB     ,AA+2
      ,CALL    ,XMUL,B10
      ,JXZ     ,*+7            JUMP IF COMPLETE
      ,DXR     ,               ELSE COUNT DIGITS
      ,STB     ,*+11           SAVE PARTIAL PRODUCT
      ,LDB     ,*+11           GET REMAINING DIGITS
      ,JMP     ,*-11           GO GET NEXT PRODUCT
      ,LDA     ,*+5            RESTORE AR
      ,LDX     ,*+5            RESTORE XR
      ,JMP*    ,XDTB           RETURN
B10   ,DATA    ,10             CONSTANT
AA    ,DATA    ,,0,0,0,0        TEMP STORAGE
*      FIXED POINT MATH      16 BIT, NO MUL/DIV      VERSION 02.0
*
*  XBTD                    FIXED POINT INTEGER BIN TO DEC CONVERSION
*
XBTD  ,ENTR    ,
      ,STA     ,BX
      ,STX     ,BX+1
      ,JAP     ,*+4            JUMP IF POSITIVE
      ,CPA     ,               ELSE COMPLEMTNT
      ,IAR     ,               AND ADD ONE
      ,TAB     ,
      ,LDXI    ,3              INITIALIZE COUNT
      ,TZA     ,
      ,CALL    ,XDIV,BB
      ,STB     ,*+16           SAVE BIN VALUE
      ,LDB     ,*+16           GET PREVIOUS DIGITS
      ,LLSR    ,4              ATTACH DIGIT TO RESULT
      ,JXZ     ,*+7            JUMP IF COMPLETE
      ,DXR     ,               ELSE COUNT DIGITS
      ,STB     ,*+11           SAVE DIGITS ASSEMBLED
      ,LDB     ,*+9            GET BIN VALUE
      ,JMP     ,*-11           GO GET NEXT DIGIT
      ,LDA     ,*+4            RESTORE AR
      ,LDX     ,*+4            RESTORE XR
      ,JMP*    ,XBTD           RETURN
BX     ,DATA    ,0,0,0,0        TEMP STORAGE
BB    ,DATA    ,10             CONSTANT
      ,MORE    ,
      ,END     ,
```