RECEIVED BY DTIC OCT 6 1967

MASTER

OAK RIDGE NATIONAL LABORATORY

operated by

UNION CARBIDE CORPORATION

NUCLEAR DIVISION

for the

U.S. ATOMIC ENERGY COMMISSION

ORNL - TM - 1879

COPY NO. - 220

DATE - Aug. 18, 1967

GEN. BRICES

H.C. 300, MN .65

Neutron Physics Division

FLEXI-KLUDGE -

A MULTI-PARAMETER NUCLEAR SPECTRUM SUMMING SYSTEM FOR THE PDP-8

B. W. Rust[†] and W. R. Burrus

ABSTRACT

FLEXI-KLUDGE is a system of PDP-8 programs for the on-line accumulation and summing of multi-parameter nuclear spectra. Pulse heights from four different detectors are brought into the computer together with flags which contain coincidence, pulser, and pile-up information. Upon initialization, the operator may impose conditions for the accumulation or summing of any of the four spectra. The result is a single dimensional 2048 channel pulse height distribution. A word and a half (18 bits) are used for the count storage for each channel. Routines are provided for typing out and integrating the spectrum between limits specified by the user and for displaying the spectrum on the oscilloscope. Routines are also provided for driving a Moseley X-Y plotter which is connected in parallel with the scope. The full 2048 channels are plotted on two 11 by 14 inch graphs in the form of an inked histogram in less than a minute. The primary purpose of the FLEXI-KLUDGE system is to aid in the routine testing and setting up of multi-parameter experiments.

This work Supported by the
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
Under Order R-104(1)

[†]Computing Technology Center, Oak Ridge Gaseous Diffusion Plant.

NOTICE This document contains information of a preliminary nature and was prepared primarily for internal use at the Oak Ridge National Laboratory. It is subject to revision or correction and therefore does not represent a final report.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

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

LEGAL NOTICE

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

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

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

PAGES 1 to 2
WERE INTENTIONALLY
LEFT BLANK

I. BRIEF DESCRIPTIONS OF THE DATA BREAK CONTROL

Data from the pulse-height analyzers enter the interface through the data break control which multiplexes 48 bits into four successive computer words. The "alert pulse" signals the computer when 48 bits are waiting to be inputted. Words are stacked in a buffer region of core whose origin and length are controlled by hardware switches. The buffer region can be either 256_{10} (400_8) or 512_{10} (1000_8) words long. If the length switches are set for a 256 word buffer then the origin switches can be set for any of the locations 0000, 0400, 1000, ..., 7400. For a 512 word buffer the possible origins are 0000, 1000, 2000, ..., 7000. The programs described in this report were originally written to use a 256 word buffer beginning in core location 1000 but they were later changed so that they use only the first 40_8 locations in this buffer leaving the remaining 340_8 locations for more programs. Switches are also available for controlling the number of words per event and these can be set for any of the numbers 1, 2, 3, or 4. These programs are designed for 4 words per event. The 4 words are referred to in the programs and in this report as words T, B, C, and D though sometimes word T may also be referred to as word G or occasionally as word A. As the 4 words for each new event come into the buffer, an address counter is counted up by 4 so that the next event will occupy the next 4 words in the buffer. This address counter works in a circular fashion so that it starts again at the beginning of the buffer when the end of it is reached. To prevent the data buffer from filling up and new events being written over all events which have not yet been processed the data break contains an up-down counter which counts up the new words coming in and can be counted down by the program as old words are taken out of the buffer and processed. If the data rate becomes fast enough that the data buffer fills up faster than the program can empty it out, then each time the buffer is completely filled with unprocessed data the up-down counter overflows and this condition stops new data from coming in until some of the old data has been removed and the up-down counter counted down by the program.

The following machine instructions are used for program control of the data break:

BADCLR	Break control <u>A</u> ddress counter <u>C</u> lear -- sets the address counter to the address of the first word in the buffer region so that the buffer will start at the position given by the hardware switches.
BABLE	Break <u>e</u> n <u>A</u> BLE in Break control -- enable the data break and allows data to start coming in.
BDISAB	Break <u>D</u> ISABLE in Break control -- disables the data break and stops the data from coming in.
UDCLR	<u>U</u> p- <u>D</u> own <u>C</u> lea <u>R</u> -- clears the up-down counter and issues an accepted pulse to the interface.
UDSOFL	<u>U</u> p- <u>D</u> own counter <u>S</u> kip on <u>O</u> ver <u>F</u> low -- causes the computer to skip the next instruction if the up-down counter is in the overflowed condition.
UDSUB1	<u>U</u> p- <u>D</u> own counter, <u>S</u> UB <u>1</u> -- subtracts 1 from the up-down counter.
UDSUB2	<u>U</u> p- <u>D</u> own counter, <u>S</u> UB <u>2</u> -- subtracts 2 from the up-down counter.
UDSUB3	<u>U</u> p- <u>D</u> own counter, <u>S</u> UB- <u>3</u> -- subtracts 3 from the up-down counter.

All of the circuits except for the address counter are reset by the POWER CLEAR which accompanies the start switch on the computer and by the clear signal generated by the UDCLR pulse.

The four signals that the programs are concerned with are denoted T, B, C, and D. The B, C, and D signals are the digitized amplitudes of the energy loss in a series of 3 detectors in a telescope arrangement. The incident particle first goes through B, then T, then D. It is possible for the signal to stop in B or C or D but not to go through them all. The electronics are set up so as to give a constant energy loss per channel in each detector. Thus the sum of energy loss in the telescope is $B + C + D$. If only the first two detectors are penetrated, $D = 0$ so that the total energy loss is just $B + C$. The T signal is derived from the time that it takes for the particle to go from the target to the B Detector. This information is of interest only if the particle stops in B so that signals are not available from C and/or D.

Figure 1 is a schematic diagram of the 48 bits of input. For each 12 bit word the first bit (labelled OV) is an overflow bit. The bits labelled F1, F2, F3, etc. are flag bits and the bits labelled D1, D2, D3, etc. are data bits. The flag bits serve the following purposes:

- F1 -- a coincidence flag which is set to one if the particle goes through the hole in the collimator and causes signals from both the B and C detectors,
- F2 -- a coincidence flag which is set to one if the particle goes through the hole in the collimator and causes signals from the C, and D detectors thus having passed through B and C and into D,
- F3 -- a flag which the logic hardware sets to one for every event so that the programs can test that bit location in the data buffer to tell whether or not an event has been inputted,
- F4 -- a pile-up indicator flag which is set to one whenever two or more events are recorded within 30 microseconds of each other and would cause an invalid signal,
- F5 -- an extra flag which does nothing at the present time but may be used for some other purpose later,
- F6 -- a flag which is set to one for pulser events which are fixed pulse-height pulses used for checking the stability of the system.

II. OPERATING INSTRUCTIONS

The normal starting address is location 200. The program clears all the buffers and starts taking data. The scope display combines groups of two channels (i.e. $2 \times 50 \text{ keV} = 100 \text{ keV}$). The vertical scale may be changed from the keyboard. Hit 0 to give full scale of 128, 1 to give full scale of 256, 2 to give full scale of 512, etc.

Other starting locations are:

- 0377 -- a restart location which causes the program to clear the input buffer but not the spectrum and then to continue taking data,

Word 0 (T) Time of Flight [sometimes called (G) Garbage]

OV	F6	F4	F3	D1	D2	D3	D4	D5	D6	D7	D8
----	----	----	----	----	----	----	----	----	----	----	----

Word 1 (B) B Detector

OV	F2	F5	D1	D2	D3	D4	D5	D6	D7	D8	D9
----	----	----	----	----	----	----	----	----	----	----	----

Word 2 (C) C Detector

OV	F1	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
----	----	----	----	----	----	----	----	----	----	----	-----

Word 3 (D) D Detector

OV	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11
----	----	----	----	----	----	----	----	----	----	-----	-----

Fig. 1. Schematic diagram of the four words of each event.

0177 -- an entry which enables the user to specify new flag and summing options for the program;

The options available are:

Sw. Reg. Bit 0	Proceed if Pile-up flag is 0	(U0)
Sw. Reg. Bit 1	Proceed if Pile-up flag is 1	(U1)
Sw. Reg. Bit 2	Proceed if Pulser flag is 0	(T0)
Sw. Reg. Bit 3	Proceed if Pulser flag is 1	(T1)
Sw. Reg. Bit 4	Proceed if Flag 1 is 0	(C0)
Sw. Reg. Bit 5	Proceed if Flag 1 is 1	(C1)
Sw. Reg. Bit 6	Proceed if Flag 2 is 0	(D0)
Sw. Reg. Bit 7	Proceed if Flag 2 is 1	(D1)
Sw. Reg. Bit 8	Sum Detector B	(B)
Sw. Reg. Bit 9	Sum Detector C	(C)
Sw. Reg. Bit 10	Sum Detector D	(D)

(The standard option is to ignore all flags and sum all three detectors. To change the option start the program at location 177. The program will halt. Set the key switches for the desired option and hit continue to clear all buffers and start taking data just as in a normal start.),

0100 -- an entry for printing between limits in core subject to a tolerance level (The computer halts 3 times -- for the first location to print, the last location to print, and the tolerance level to be keyed in. The program prints the contents of the memory between the specified limits in rows of ten numbers suppressing all those rows for which the sum of the numbers in the ten corresponding core locations is less than the specified tolerance. When the program is finished with the printing, the scope continuously plots the spectrum until it is stopped by the user.),

0120 -- an entry for continuously displaying the spectrum,

0460 -- an entry for summing between limits in core (The

computer stops twice for the limits to be keyed in and then prints the two limits and the sum of all the locations between them. It then stops with 7777 in the AC.),

0001 -- an entry for calibrating the Moseley plotter (The program reads the number in the switch register and positions the pen accordingly. When the number in the switch register is 1777 the pen goes to the point $X = 12.725$ in., $Y = 9.990$ in. To adjust the plotter, first set the switches to 1000 and use the zero controls to set X and Y at half scale. Then set the switches to 0000 and use the gain controls to set X and Y to zero. Repeat the above two steps until the procedure converges.),

1040 -- the entry for Moseley plotting. To use the Moseley plotter first make sure the plotter is calibrated. Then raise the pen and insert the graph paper. Start the program at 1040 and the computer will stop. Lower the pen, hit continue and the first 1048 channels will be printed out in groups of two channels with 40 groups/ inch. The computer will stop after the last channel is plotted. Raise the pen, remove the graph paper and put in a new sheet leaving the pen up. Hit continue and the computer will stop again. Lower the pen and hit continue to plot the last 1048 channels. The computer will stop after plotting the last channel. Raise the pen and remove the paper. Reload the paper for the next graph leaving the pen up.

III. CORE STORAGE USED BY THE FLEXI-KLUDGE SYSTEM

<u>Locations</u>	<u>Contents</u>
0000 -- 0010	Calibration program for the Moseley plotter.
0011 -- 0017	Option switches for subroutine DETEC.
0020 -- 0044	Page zero pointer used for addressing across pages.

<u>Locations</u>	<u>Contents</u>
0045 -- 0076	Subroutine ANTFRZ (the anti-freeze subroutine).
0100 -- 0122	CALIP, the main program for calling the channel printing routine (CHPRNT).
0124 -- 0152	Subroutine INIT.
0153 -- 0176	Subroutine SET1.
0177 -- 0361	KLUDGE, the main program for data acquisition.
0362 -- 0372	Subroutine COMAND.
0374 -- 0377	Invariant points for KLUDGE.
0400 -- 0456	Subroutine SCOPE.
0460 -- 0510	SUM, the main routine for integrating over specified channels.
0511 -- 0574	Subroutine SUMM.
0575 -- 0577	Subroutine SUMP.
0600 -- 0776	Subroutine CHPRNT.
1000 -- 1037	Data buffer.
1040 -- 1046	MOE, main routine for Moseley plotting.
1047 -- 1160	Subroutine EANIE.
1161 -- 1177	Subroutine DELAY.
1200 -- 1267	Subroutine GRAPH.
1270 -- 1356	Subroutine DETEC.
1357 -- 1377	Subroutine TEST.
1400 -- 1540	Dan's CONVERT and DECODE routines, i.e. Subroutine PRINT.
1545 -- 1556	Subroutine LINEF.
1560 -- 1575	Subroutine TYPEO.
1600 -- 7577	Pulse height spectrum buffer.

Note: When the program is first loaded, locations 1620 -- 1625 contain SCOPECAL, a straight line scope calibrating routine, but when KLUDGE is started the spectrum buffer is cleared and SCOPECAL is lost.

IV. PROGRAM DESCRIPTIONS

A. KLUDGE and ANTFRZ

KLUDGE is the main data acquisition program of the FLEXI-KLUDGE

system. When it is first started it jumps off to subroutine INIT which allows the user to specify the treatment of the various flags and to specify the detectors whose signals are to be summed in the pulse-height spectrum. After returning from INIT the program clears the 40_8 word data buffer and the spectrum buffer and initializes the hardware. The up-down counter is initially set to 1340 so that 8 events will cause an overflow in the 256 bit. The program then enters the data acquisition loop which constitutes most of the KLUDGE routine. In this main loop the program continually jumps off to subroutine ANTFRZ to check for new data in the data buffer. If ANTFRZ finds a new event it returns control to a location in KLUDGE which loads the AC with the address in the data buffer of the first word of the new event and then jumps off to subroutine DETEC which returns to KLUDGE with the AC containing the number of the channel to be incremented in the pulse-height spectrum. The pulse-height spectrum is stored in a 6000_8 word buffer beginning in location 1600 (see Fig. 2). Each channel is represented by a word and a half (18 bits) in the buffer giving 2048_{10} channels in all. Upon receiving the number of the channel to be incremented from DETEC, KLUDGE increments the corresponding 18-bit number in the buffer using the proper incrementing procedure according as the channel number is odd or even. KLUDGE then checks to see if it has reached the end of the 40-word data buffer. If it has it disables the data break, counts down the up-down counter by 40 (the up-down counter will be in the overflow condition when the program gets to the end of the data buffer and this will have stopped the data from coming in) and clears the address counter. It then enables the data break and jumps off to SCOPE to plot a channel on the oscilloscope. If the program has not reached the end of the data buffer it skips the re-initialization of the hardware and jumps to SCOPE. Whenever ANTFRZ fails to find a new event it returns control to KLUDGE in this same location where it jumps off to SCOPE. Upon returning from SCOPE, KLUDGE jumps off to COMAND to pick up any new scaling constant that might have been typed in by the user and then returns to the beginning of the loop to start the process again. If ANTFRZ ever fails to find new data in $20,000_8$ successive passes through the loop it types out

Fig. 2. Schematic Diagram of the Pulse-Height Spectrum Buffer.

an "F", rings the bell and jumps outside the loop to reinitialize everything except the spectrum buffer and then re-enters the loop.

B. INIT and SET1

INIT is a subroutine which is called by KLUDGE to enable the user to specify the flag options and to specify which of the detector signals are to be included in the sum whose spectrum is accumulated. Immediately after being called INIT stops for the user to set the switches for the desired options. The switch settings are described in the section on operating instruction. When the user sets the switches and hits continue INIT picks up the contents of the switch register and uses this to set a number of internal switches. These internal switches are:

<u>Switch</u>	<u>Use</u>
SWPILE	the pile-up flag switch,
SWPUL	the pulser flag switch,
SWFL1	the Flag 1 switch,
SWFL2	the Flag 2 switch,
SWB	the B detector switch,
SWC	the C detector switch,
SWD	the D detector switch.

INIT calls subroutine SET1 to set the first 4 of the above switches. Each of the four correspond to two keys in the switch register setting and SET1 unpacks those two keys and sets the internal switch to do whatever was specified by the key settings. If the first of the two keys is up and the second one is down the corresponding internal switch is set to +1, if the first key is down and the second key is up the switch is set to -1, and if both keys are up or if both are down the switch is set to 0. Switches SWB, SWC, and SWD are each set to 0 or +1 depending on whether the corresponding key switch is down or up. After setting all the switches described above INIT returns control to KLUDGE.

C. DETEC and TEST

DETEC is a subroutine called by KLUDGE each time a new event is found in the data buffer. Its purpose is to determine which channel

in the pulse-height spectrum should be incremented and to return this channel number to KLUDGE in the accumulator. It checks the status of the pulser flag, the pile-up flag, Flag 1, and Flag 2 for the event and takes the action specified by the flag switches which were set in subroutine INIT. To do this DETEC calls subroutine TEST four times, once for each of the above-mentioned flags. In each case, if the flag in question is 0 and the corresponding switch is -1 or if the flag is 1 and the corresponding switch is +1, TEST returns control to DETEC in a location which causes DETEC to return the highest possible channel number (2047_{10}) to KLUDGE. If all of the four flags are consistent with the switch settings specified by INIT then TEST gives a normal return to DETEC in each case and DETEC computes the channel number to return to KLUDGE by forming a sum of the signals from those detectors (B,C,D) whose corresponding switches (SWB,SWC,SWD) were set to +1 in INIT, not summing the detectors whose corresponding switches were set to 0. If there is an overflow in any of the detectors being summed or if the sum exceeds 2047_{10} (3777_8) DETEC returns 2047_{10} to KLUDGE which cause it to increment the highest possible channel.

D. SCOPE and COMAND

SCOPE is a subroutine for plotting the pulse-height spectrum. It is called periodically by KLUDGE, CALLP, and in the wait loop of TYPEO to plot a channel of the spectrum. The pulse-height spectrum contains 2048_{10} channels but the scope can plot only 1024_{10} distinct abscissas, so each time it is called SCOPE adds the next two adjacent channels and plots the sum as a single point. When the plot reaches the right hand limit of the scope it starts again at the beginning of the plot. The ordinates (counts accumulated in each pair of adjacent channels) are scaled by a scaling factor which can be changed by typing in a new one from the keyboard. All of the routines which use SCOPE also periodically call subroutine COMAND which checks to see if a new scaling constant has been typed in and which reads any such new scaling constant and stores it away in the proper location of SCOPE.

E. CALLP and CHPRNT

CALLP is the main routine for the channel print-out system. When the user starts the program at the beginning of CALLP, it disables the data break, initializes the printer, and halts for the user to key in the number of the first channel to print. When the first channel has been keyed in it halts again for the last channel number to be keyed in and then again for the tolerance level for printing to be keyed in. After picking up the two channel numbers and the tolerance level it jumps off to subroutine CHPRNT to do the printing. The calling sequence for CHPRNT is:

```
JMS I CHPRNT
YPOINT
CHN1
CHN2
TOLER
(CONTINUE)
```

where YPOINT is the address of the first location in the spectrum buffer, CHN1 is the number of the first channel to be printed, CHN2 is the number of the last channel to be printed, and TOLER is the tolerance level for printing. CHPRNT prints the channels between the specified limits in the following format:

```
DCNO CHO CH1 CH2 CH3 CH4 CH5 CH6 CH7 CH8 CH9 CH10
```

where DCNO is a channel number from the sequence 0, 10, 20, 30, ... and begins in the first column of the page, and CHO, CH1, ..., CH10 are the contents of channels DCNO through DCNO+9 and are printed as 6 digit numbers whose leading zeros are suppressed. If the sum of the channels in any decade is below the tolerance level then printing is suppressed for that decade. Before printing anything CHPRNT calls subroutine LINEF twice to space to a new line. CHPRNT calls subroutine SUMP which in turn calls subroutine SUMM to do the summing for each decade. For each decade whose channel sum exceeds the tolerance it calls subroutine PRINT 11 times -- once to print out the initial channel number and ten times to print the contents of the ten channels. After each decade has been

printed CHPRNT calls LINEF to space to a new line and after the last decade has been printed CHPRNT returns control to CALLP with the teleprinter spaced to a new line. When control is returned to it CALLP goes into a loop which continuously calls SCOPE and COMAND thus plotting the pulse-height spectrum until the user can decide what to do next.

F. PRINT and DECODE

PRINT is a subroutine called by CHPRNT and SUM to print out the decimal equivalents of various single and double precision machine numbers in various integer formats. If the number to be printed out is a double precision number PRINT should be called with the low order part of the number in the MQ and the high order part of the number in the AC. If the number is a single precision number it should go in the MQ and the AC should be cleared before calling PRINT. PRINT has 8 different entries, ENTRY1, ENTRY2, ENTRY3, ... , ENTRY8, each one corresponding to a different format for printing. ENTRY1 corresponds to a single digit format, ENTRY2 corresponds to a 2 digit integer format, ENTRY3 a 3 digit integer format, etc. The calling sequence for PRINT is:

```
JMS I PRINT
JMP I ENTRYn
(CONTINUE)
```

where ENTRYn is one of the eight possible entries.

Subroutine PRINT calls subroutine DECODE to convert the machine number to ASCII decimal digits and DECODE call subroutine TYPEO to print out each of the digits. DECODE operates on the principle that powers of ten in octal notation can be subtracted from an octal number to convert it to its decimal equivalent. When PRINT is called with ENTRYn in the calling sequence, the user is specifying an n-digit integer format and PRINT calls DECODE n times successively with the following calling sequences:

```
ENTRYn,    JMS      DECODE
           NNNN
           NNNN
```

```
ENTRYn-1, JMS      DECODE
```

```
      MMMM
```

```
      MMMM
```

```
      .
```

```
      .
```

```
      .
```

```
ENTRY1, JMS      DECODE
```

```
      7777
```

```
      7777
```

```
      (CONTINUE)
```

where NNNNNNNN is the double precision complement of 10 raised to the nth power, MMMMMMMM is the double precision complement of 10 raised to the (n-1)th power, ... , and 77777777 is of course the double precision complement of 10 raised to the zeroth power. Each time that DECODE is called from this sequence it prints out the digit of the number corresponding to that power of ten, suppressing the printing of leading zeros. If the user chooses a format that is not large enough to contain the number he wants printed, the program types out a slash (/).

PRINT and DECODE were written by Dan Putzulu (a Co-op student and a very clever fellow) and are the most subtle programs in the FLEXI-KLUDGE system. The user is cautioned against heavy-handed attempts to change these programs.

G. TYPEO and LINEF

TYPEO is a subroutine which is called by several of the programs in the FLEXI-KLUDGE system to print out characters on the teletype. It should be called with the character to be printed in the AC and it returns control to the calling program with the AC cleared. It also preserves the status of the link. TYPEO calls SCOPE in the wait loop for the teletype and also calls COMAND after emerging from the wait loop. LINEF is a subroutine which is called by various programs in the system to produce carriage returns and a line feeds on the teletype. It calls TYPEO to do the actual teletype commands. It returns to the calling program with a cleared AC and the link is preserved.

H. SUM and SUMM

SUM is the main program for integrating over specified channels. When it is first started it stops for the lower limit to be keyed in and then stops again for the upper limit. These two channel numbers are stored in invariant locations 0573 and 0574 and are picked up from these locations whenever the program needs to use them. These invariant locations are used by subroutine SUMM, the actual summing routine, and must contain the limits for summing whenever SUMM is called by programs other than SUM. The advantage of having these locations invariant is that it enables the user to change SUM and SUMM without having to change the programs on other pages which also use SUMM. After storing away the channel limits, SUM jumps off to LINEF and then jumps off to PRINT twice to print out the two limits. It then jumps to SUMM which computes the double precision sum of the contents of all the channels between the two specified in locations C1 and C2. SUMM returns control to the calling program (in this case SUM) with the low order part of the sum in the MQ and the high order part in the AC. When control is returned to SUM it jumps off to PRINT to print out the double precision sum in a 7 digit format and then stops with 7777 in the AC.

I. SUMP

SUMP is a dummy subroutine which is called by programs that want to use SUMM but which are located on other pages. Its locations are invariant (0575 - 0577) and SUMM can be changed without changing the programs on other pages which use it. Such a program must set the limits for summing in locations C1 and C2 (0573 and 0574) and then JMS to SUMP which in turn will JMS to SUMM.

J. EANIE, MOE, GRAPH, and DELAY

EAНИЕ, MOE, GRAPH, and DELAY are a package of 4 routines for making histogram plots on a standard Moseley X-Y plotter. The Moseley plotter is connected in parallel with the standard Type 34 Oscilloscope Display through suitable isolating attenuators (with an attenuation factor of about 20 to 1). In order to plot on the Moseley plotter, the "spot" is

slowly moved across the scope. The pen on the Moseley plotter then moves in step with the "spot", except that the inertial reaction time is much greater than the deflection time of the scope. Therefore, a suitable delay is inserted after each change of Y coordinate and after each change of X coordinate so that the Moseley pen will come to rest before the next step.

In order to obtain adequate separation between channels, the full 2048 channel spectrum is plotted on two 10.00 by 13.3 inch pages of graph paper giving a total X axis length of 26.6 inches. This would result in 80 channels per inch, a spacing which would be too close to resolve visually, so that groups of two channels at a time are plotted with 40 groups per inch. This reduces the overall plotting time and is compatible with the way in which the normal oscilloscope display works so that it can use the same program steps to combine groups of two channels.

The MOE routine is the main routine of the sequence and consists of a call to EANIE for plotting the first page of graph, followed by another call to EANIE for plotting the second page.

EANIE makes a histogram of 512 bars in groups of two channels per bar. It picks up the origin of the 1024 channels to be plotted from the calling sequence in MOE and picks up the scale constant that is used in the standard oscilloscope display so that the scale of the graph will be the same as that displayed with the normal oscilloscope display. EANIE first positions the pen at the X-Y origin and halts for the paper to be inserted and the pen to be lowered. It then transfers control to GRAPH each time a step is to be added to the histogram. At the completion of the graph, a halt allows the paper to be removed and another piece inserted. If a prospective bar on the histogram is off-scale, EANIE truncates it by dropping off the high order bits so that a peak would "roll around" off the top and back onto the bottom of the graph. This tends to mess up the appearance of a graph when several plots are made on one page with different scale factors.

The GRAPH routine is responsible for moving the Moseley pen in the Y direction, then waiting for the Y motion of the pen to be completed,

then moving in the X direction, and finally waiting for the X motion to be completed. After the Y and X movements are finished, return is made to the calling sequence. In order to make the plot as rapidly as possible, a decision as to which of two delay strategies to use is made for each step. For small Y steps (when the curve is nearly flat), a fixed delay of about 15 ms is made following the Y motion and about 6 ms following the X motion. The tolerance for a small step and the Y and X delays are controlled by separate constants so that they can be easily adjusted to match the kinetics of a particular plotter. In the event of a large step which exceeds the tolerance, a variable time delay for the Y step is used depending upon the magnitude of the step. The delay length for this case is a fixed 16 ms plus an additional amount (up to about 500 ms) proportional to the Y step size. After the X step, a fixed delay of about 100 ms is introduced. The effect of this strategy is to "zip" along without much Y or X detail if the graph is changing very slowly, but to plot more deliberately in both the Y and X direction for rapidly varying plots. Two problems have been encountered. The first is that the Moseley plotter used tends to overshoot about 5% when making a step of several inches. The other is that the pen "slews" so fast on long steps that the ink does not flow uniformly. A solution to both these problems would be to generate many mini-steps which uniformly move the pen rather than a single step. The GRAPH routine was made a separate module so that this eventual refinement could be easily made.

The DELAY routine is a minimum length routine for inserting program delays into a computation. When DELAY is called, a sufficient number of steps are "looped" to kill the required time. The calling sequence is:

```

CLA
TAD N
JMS DELAY
M

```

The value of M determines the number of repetitions of an inner loop. M = 1 produces the minimum delay and M = 7777 produces the maximum inner loop delay of about 18 ms (4095 times around). M = 0 produces 4096 times around! The number of repetitions of the inner loop is N-1. If

$N = 0$, the routine immediately returns with only a few cycles delay. The maximum delay that can be obtained is about a minute.

K. MOECAL

MOECAL is a simple routine which picks up the contents of the accumulator switches and loads both the X and Y buffers of the scope display system. The primary purpose is for calibration of the Moseley plotter which is connected in parallel with the scope. A time of about 18 ms is introduced between loading the buffers and looping around to read the switches again so that switching transients in the digital-to-analog converters will not occur sufficiently often to confuse the plotter. Instructions for the setting of the gain and zero controls are given in the program comments accompanying MOECAL. These instructions assume that the Moseley plotter is connected in parallel with the scope. The conventional manner of driving the scope is to use a differential input with the deflection on one side and with a "half-full-scale" signal on the other. It would simplify the calibration procedure specified for the Moseley plotter if the input were "single-ended" with the other side of the Moseley input grounded.

L. SCOPECAL

SCOPECAL is a minimum length routine for making a diagonal line on the Standard Type-34 Oscilloscope Display. First the brightness is set at the minimum level, and then the accumulator is loaded into the X and Y display buffers and the point intensified. Then the accumulator is kicked up one, and the buffers loaded again. Since the display ignores the two high order bits, the effect is to move the dot diagonally across the scope 4 times for every time the accumulator gets kicked completely around (4096 times).

ORNL DWG 67-8655

Fig. 3. Interrelations of the programs in the FLEXI-KLUDGE system.

Fig. 4. Flow diagram for KLUDGE and ANTIFRZ.

Fig. 5. Flow diagrams of INIT and SET 1.

ORNL DWG 67-8659

Fig 7. Flow diagram of SCOPE.

ORNL DWG 67-8660

Fig. 8. Flow diagrams of CALLP and COMMAND.

ORNL DWG 67-8663

Fig. 11. Flow diagrams of LINEF and TYPE0.

Fig. 12. Flow diagrams of SUN, SUMM, and SUMP

ORNL DWG 67-8665

MAIN ROUTINE FOR MOSELEY PLOTTER

Fig. 13. Flow diagrams of MOE and DELAY.

ORNL DWG 67-8666

EANIE PLOTTING SUBROUTINE

Fig. 14. Flow diagram of EANIE.

ORNL DWG 67-8667

PEN MOVING SUBROUTINE

Fig. 15. Flow diagram of GRAPH.

ORNL DWG 67-8696

Fig. 16. Flow diagrams of MPECAL and SCPECAL.

```

/ MOECAL == CALIBRATION PROGRAM FOR THE MOSELEY PLOTTER
/ PROGRAM READS N FROM SWITCHES AND POSITIONS PEN
/ WHEN N .EG. 1777, THE PEN GOES TO X = 12.725 IN., Y = 9.990 IN.
/ TO ADJUST PLOTTER, FIRST SWITCH IN N=1000(8) AND
/ USE ZERO CONTROLS TO SET X AND Y AT HALF SCALE,
/
/ THEN SWITCH IN N = 0, AND USE
/ GAIN CONTROLS TO SET X AND Y ZERO.
/ REPEAT THIS PROCEEDURE UNTIL IT CONVERGES.

```

```

*0000
0 0 ROLL0, 0000
1 7604 MOECAL, LAS
2 6053 DXL
3 7410 SKP
4 0 0000 /SACRED TO DDT
5 6063 DYL
6 2000 ISZ ROLL0
7 5006 JMP .+1
10 5001 JMP MOECAL

```

SYMBOL TABLE

```

MOECAL 1
ROLL0 0

```

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/ OPTION SWITCHES FOR SUBROUTINE DETEC

/ THE VALUES OF THE SWITCHES CAN BE CHANGED BY THE SUBROUTINE
 / INIT IN THE MAIN PROGRAM

11	1	SWB,	0001	/ UNLESS THE SWITCHES ARE CHANGED BY INIT
12	1	SWC,	0001	/ THEY WILL BE SET TO ACCUMULATE AND
13	1	SWD,	0001	/ PLOT (B+C+D) WITHOUT REGARD TO WHETHER
14	0	SWPIL,	0000	/ THE FLAGS ARE 0 OR 1
15	0	SWPUL,	0000	
16	0	SWFL1,	0000	
17	0	SWFL2,	0000	

SYMBOL TABLE

SWB	11
SWC	12
SWD	13
SWFL1	16
SWFL2	17
SWPIL	14
SWPUL	15

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/ SUPER KLUDGE SINGLE PARAMETER 18-BIT ROUTINE

/ PERMANENT SYMBOLS

*0020

20	400	SCOPEP, 0400	/1ST LOC. IN SCOPE ROUTINE
21	1600	YPOINT, 1600	/LOC. OF 1ST CHANNEL
22	1000	BUFFP, 1000	/1ST LOC. IN BUFFER
23	600	CHPRNT, 0600	/1ST LOC. IN CHAN. PRINT ROUTINE
24	1400	PRINTP, 1400	/1ST LOC. OF DAN S PRINT ROUTINE
25	200	BEGIN, 0200	/1ST LOC. IN KLUDGE ROUTINE
26	377	RESTRT, 0377	/RESTART LOC. FOR KLUDGE ROUTINE
27	1403	ENTRY8, 1403	
30	1406	ENTRY7, 1406	
31	1411	ENTRY6, 1411	
32	1414	ENTRY5, 1414	
33	1417	ENTRY4, 1417	
34	1422	ENTRY3, 1422	
35	1425	ENTRY2, 1425	
36	1431	ENTRY1, 1431	
37	1545	LINEF, 1545	
40	1560	TYPE0, 1560	
41	1270	DETEC, 1270	
42	1161	DELAY, 1161	
43	436	SCALE, 0436	
44	362	COMMAND, 362	

SYMBOL TABLE

BEGIN	25
BUFFP	22
CHPRNT	23
COMMAND	44
DELAY	42
DETEC	41
ENTRY1	36
ENTRY2	35
ENTRY3	34
ENTRY4	33
ENTRY5	32
ENTRY6	31
ENTRY7	30
ENTRY8	27
LINEF	37
PRINTP	24
RESTRT	26
SCALE	43
SCOPEP	20
TYPE0	40
YPOINT	21

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/ ANTFRZ = THE ANTIFREEZE SUBROUTINE

/
TYPE0=40

*0045

/

45	7650	ANTFRZ, SNA CLA	/SKIP IF NEW DATA HAS ARRIVED
46	5053	JMP WAIT	
47	1067	TAD M0002	
50	3075	DCA COUNT	
51	3076	DCA R0LL0	
52	5472	JMP I DATAP	
53	2076	WAIT, ISZ R0LL0	
54	5473	JMP I PLOTP	
55	2075	ISZ COUNT	
56	5473	JMP I PLOTP	
57	1067	TAD M0002	
60	3075	DCA COUNT	
61	3076	DCA R0LL0	
62	1070	TAD BELL	
63	4440	JMS I TYPE0	
64	1071	TAD F	
65	4440	JMS I TYPE0	
66	5474	JMP I RESTRP	
67	7776	M0002, 7776	
70	207	BELL, 0207	/ASCII CODE FOR BELL
71	306	F, 0306	/ASCII CODE FOR F
72	374	DATAP, 0374	
73	375	PLOTP, 0375	
74	377	RESTRP, 0377	
75	0	COUNT, 0	/INITIAL COUNT DETERMINES BEGINNING ANTIFREEZE / TIME
76	0	R0LL0, 0	

SYMBOL TABLE

ANTFRZ	45
BELL	70
COUNT	75
DATAP	72
F	71
M0002	67
PLOTP	73
RESTRP	74
R0LL0	76
TYPE0	40
WAIT	53

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

```

/ CALLP -- A PROGRAM TO CALL THE CHANNEL PRINTING ROUTINE
/
/ IMMEDIATELY AFTER BEING STARTED PROGRAM HALTS FOR
/ OPERATOR TO KEY IN NO. OF 1ST CHANNEL DESIRED AND
/ THEN HALTS AGAIN FOR NO. OF LAST FILE DESIRED
/ AND THEN HALTS AGAIN FOR TOLERANCE FOR PRINTING
/
/
CHPRNT=0023
YPOINT=0021
SCOPEP=20
COMAND=44
*0100
100 6314 BEGIN, BDISAB
101 6046 TLS
102 7402 HLT /LOAD NO. OF 1ST CHANNEL DESIRED IN
103 7604 LAS / SWITCHES AND HIT CONTINUE
104 3115 DCA CHN1
105 7402 HLT /LOAD NO. OF LAST CHAN. DESIRED
106 7604 LAS
107 3116 DCA CHN2
110 7402 HLT /LOAD TOLERANCE FOR PRINTING
111 7604 LAS
112 3117 DCA CHN3
113 4423 JMS I CHPRNT
114 21 YPOINT
115 0 CHN1, 0000
116 0 CHN2, 0000
117 0 CHN3, 0000
120 4420 JMS I SCOPEP
121 4444 JMS I COMAND
122 5120 JMP ,=2

```

SYMBOL TABLE

BEGIN	100
CHN1	115
CHN2	116
CHN3	117
CHPRNT	23
COMAND	44
SCOPEP	20
YPOINT	21

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/ INIT == THE SWITCH INITIALIZATION SUBROUTINE

/

•124

SWB=11

SWC=12

SWD=13

SWPILE=14

SWPUL=15

SWFL1=16

SWFL2=17

/SET UP PROGRAM OPTIONS FROM SWITCHES

/SW0 PROCEED IF PILE UP FLAG = 0

/ 1 = 1

/ 2 PULSER FLAG = 0

/ 3 = 1

/ 4 FLAG1 = 0

/ 5 = 1

/ 6 FLAG2 = 0

/ 7 = 1

/ 8 SUM DETECTOR B

/ 9 C

/ 10 D

124	0	INIT,	0000
125	7602	SETUP,	HLT CLA
126	3011	DCA	SWB
127	3012	DCA	SWC
130	3013	DCA	SWD
131	7604	LAS	
132	4153	JMS	SETI
133	14	SWPILE	
134	4153	JMS	SETI
135	15	SWPUL	
136	4153	JMS	SETI
137	16	SWFL1	
140	4153	JMS	SETI
141	17	SWFL2	
142	7510	SPA	
143	2011	ISZ	SWB
144	7004	RAL	
145	7510	SPA	
146	2012	ISZ	SWC
147	7004	RAL	
150	7510	SPA	
151	2013	ISZ	SWD
152	5524	JMP	I INIT

/

/SETI ROUTINE FOR UNPACKING TWO KEYS. TAD KEYS

/ JMS SETI

/ASSUMES KEYS ARE IN BITS 0 AND 1 SW

/ (RETURN)

/KEY TABLE

/ SWITCH

/ KEY 1 UP .GT. 0

/ KEY 2 UP .LT. 0

```

/ BOTH UP .EQ. 0
/
153 0 SETI, 00
154 3175 DCA SAVE
155 1553 TAD I SETI /GET ADDRESS OF THE SWITCH AND STORE IT AWAY
156 3176 DCA SWADDR
157 7240 CLA CMA
160 3576 DCA I SWADDR /SET SW = -1
161 1175 TAD SAVE
162 7004 RAL
163 7500 SMA
164 2576 ISZ I SWADDR /SET SW = 0 IF KEY 2 DOWN
165 7000 OPR
166 7630 SZL CLA
167 2576 ISZ I SWADDR /ADD 1 TO SW IF KEY 1 UP
170 7000 OPR
171 2153 ISZ SETI
172 1175 TAD SAVE
173 7006 RTL /SET UP NEXT TWO BITS
174 5553 JMP I SETI
175 0 SAVE, 0000
176 0 SWADDR, 0000
/
/
/ KLUDGE ROUTINE -- FOR SINGLE DETECTOR SPECTRA
/
/ THE BUFFER IS 40 (8) WORDS LONG. ITS STARTING ADDRESS IS STORED IN
/ LOC. BUFP ON PAGE ZERO. (BUFFER OCCUPIES 1000-1037 IN CORE)
/
/ THERE ARE 2048 (10) = 4000 (8) DATA CHANNELS,
/ EACH CHANNEL OCCUPIES 18 BITS AND ALL THE
/ CHANNELS TOGETHER OCCUPY A BLOCK OF
/ 3072 (10) = 6000 (8) CONSECUTIVE WORDS,
/ (LOC. 1600-7577) AND THE 1ST ADDR. OF THE
/ BLOCK IS STORED IN LOC. YPOINT ON PAGE ZERO.
/
/
SCOPEP=20
YPOINT=21
BUFP=22
ANTFRZ=45
DETEC=41
SCALEB=43
COMMAND=44
/
*177
177 4124 BEGIN, JMS INIT
200 7200 CLRSTR, CLA
201 1342 TAD M6000
202 3353 DCA COUNT
203 1021 TAD YPOINT
204 3354 DCA ADDR
205 3754 DCA I ADDR /CLEAR THE DATA CHANNELS
206 2354 ISZ ADDR
207 2353 ISZ COUNT
210 5205 JMP .-3
/
211 6032 RSTRT, KCC
212 7200 CLA
213 6046 TLS

```

214	6077		DSB+3		
215	1343		TAD	M40	
216	3353		DCA	COUNT	
217	1022		TAD	BUFFP	
220	3354		DCA	ADDR	
221	3754		DCA	I ADDR	/CLEAR DATA BUFFER TO 0000
222	2354		ISZ	ADDR	
223	2353		ISZ	COUNT	
224	5221		JMP	I-3	
225	3352		DCA	BMARK	/INITIALIZE BMARK TO SCAN DATA BUFFER
226	6311		BADCLR		/CLEAR THE HARDWARE
227	6324		UDCLR		
230	1350		TAD	M440	/SET UP-DOWN COUNTER TO 1340 SO
231	3353		DCA	COUNT	/ 7 EVENTS (34 WORDS) WILL CAUSE AN
232	6331		UDSUBI		/ OVERFLOW IN THE 256 BLT.
233	2353		ISZ	COUNT	
234	5232		JMP	I-2	
235	6312	LOOP,	BABLE		/START TAKING DATA
236	7200		CLA		
237	1352		TAD	BMARK	/PICK UP BUFFER(I)
240	1022		TAD	BUFFP	/ I=BMARK
241	3351		DCA	BUFADD	
242	1751		TAD	I BUFADD	/EXAMINE CONTENTS OF BUFFER(I)
243	5644		JMP	I I+1	/JUMP TO ANTIFREEZE ROUTINE
244	45		ANTFRZ		
245	7200	DATA,	CLA		
246	1351		TAD	BUFADD	
247	4441		JMS	I DETEC	/JUMP TO DETECTOR SUBR, WITH LOC. OF G-SIGNAL
	/				/ IN AC AND RETURN WITH CHAN. NO. TO
	/				/ INCREMENT IN AC
250	3355	CHNST0,	DCA	CHANNO	
251	7100		CLL		
252	1355		TAD	CHANNO	/ DIVIDE CHAN. NO. BY 2 AND
253	7010		RAR		/ DISCARD REMAINDER FOR ODD
254	1355		TAD	CHANNO	
255	1021		TAD	YPOINT	
256	3354		DCA	ADDR	
257	1354		TAD	ADDR	
260	7001		IAC		
261	3357		DCA	ADDR2	
262	7430		SZL		
263	5276		JMP	HALFP	
264	7300		CLA	CLL	
265	1757		TAD	I ADDR2	
266	1344		TAD	L0100	
267	3757		DCA	I ADDR2	
270	7420		SNL		
271	5314		JMP	C0NT	
272	1754		TAD	I ADDR	
273	7001		IAC		
274	3754		DCA	I ADDR	
275	5314		JMP	C0NT	
276	7300	HALFP,	CLA	CLL	
277	1757		TAD	I ADDR2	
300	7001		IAC		
301	3757		DCA	I ADDR2	
302	7420		SNL		
303	5314		JMP	C0NT	
304	1754		TAD	I ADDR	
305	7001		IAC		

```

306 341      AND      L0077
307 3356     DCA      TEMP
310 1754     TAD      I      ADDR
311 345      AND      L7700
312 1356     TAD      TEMP
313 3754     DCA      I      ADDR
314 7200     CONT,    CLA
315 3751     DCA      I      BUFADD /RESTORE BUFFER(I) TO 0000
316 1352     TAD      BMARK / AND INCREMENT BMARK TO PICK
317 1347     TAD      L0004 / UP BUFFER(I+4)
320 360      AND      L0037
321 3352     DCA      BMARK
322 1352     TAD      BMARK
323 7440     SZA
324 5336     JMP      PLOT
325 6314     BDISAB
326 1361     TAD      M10
327 3353     DCA      COUNT
330 6333     UDSUB2
331 6333     UDSUB2
332 2353     ISZ      COUNT
333 5330     JMP      =3
334 6311     BADCLR
335 6312     BABLE
336 4420     PLOT,    JMS      I      SCOPEP
337 4444     JMS      I      COMMAND
340 5235     JMP      LOOP

/
341 77      L0077, 0077
342 2000     M6000, 2000
343 7740     M40, =40
344 100      L0100, 0100
345 7700     L7700, 7700
346 7774     M4, =4
347 4        L0004, 0004
350 7340     M440, =440
351 0        BUFADD, 0
352 0        BMARK, 0
353 0        COUNT, 0
354 0        ADDR, 0
355 0        CHANN0, 0
356 0        TEMP, 0
357 0        ADDR2, 0
360 37       L0037, 0037
361 7770     M10, =10

/
/
/
/
/ COMMAND = A COMMAND ROUTINE FOR CHANGING THE SCALE FACTOR FOR PLOTTING
/
*362
362 0        COMM, 0 /COMMAND ROUTINE FOR CHANGEING SCALE FACTOR
363 6031     KSF /SKIP IF A NEW SCALE CONST. HAS BEEN TYPED IN
364 5762     JMP      I      COMM
365 6036     KRB / READ THE NEW SCALE CONST.
366 7041     CIA
367 1372     TAD      P276
370 3443     DCA      I      SCALE0
371 5762     JMP      I      COMM

```

372 276 P276, 0276

/

*374

374	5245	JMP	DATA	/POINTERS
375	5336	JMP	PLOT	
376	5200	JMP	CLRSTR	
377	5211	JMP	RSTRY	

/

SYMBOL TABLE

ADDR	354
ADDR2	357
ANTFRZ	45
BEGIN	177
BMARK	352
BUFADD	351
BUFFP	22
CHANNO	355
CHNST0	250
CLRSTR	200
COMMAND	44
COMM	362
CONT	314
COUNT	353
DATA	245
DETEC	41
HALFP	276
INIT	124
L0004	347
L0037	360
L0077	341
L0100	344
L7700	345
L80P	235
M10	361
M40	343
M4	346
M440	350
M6000	342
P276	372
PLOT	336
RSTRY	211
SAVE	175
SCALE0	43
SCOPEP	20
SETI	153
SETUP	125
SWADDR	176
SWB	11
SWC	12
SWD	13
SWFL1	16
SWFL2	17
SWFILE	14
SWPUL	15
TEMP	356
YP0INT	21

DUPLICATE TAGS

NONE

```

/SCOPE -- A ROUTINE FOR DISPLAYING ON SCOPE THE CONTENTS
/      OF 2048 (10) 18-BIT CHANNELS,
/      EACH OF THE 1024 (10) POSSIBLE X-VALUES
/      WILL CORRESPOND TO TWO ADJACENT
/      CHANNELS. EACH PAIR OF ADJACENT
/      CHANNELS WILL BE SUMMED AND THIS 2BLE-
/      PRECISION SUM NORMALIZED TO 10 BITS SO
/      THAT IT DOES NOT EXCEED 1024 (10). THIS
/      VALUE IS PLOTTED ON THE Y-AXIS
/
/      THE ADDRESS OF THE 1ST WORD OF THE 1ST CHANNEL
/      SHOULD BE CONTAINED IN LOC. YPOINT ON PAGE 0.
/
SCOPEP=20
YPOINT=21
*0400
SCOPE, 0000
400      0
401 7200      CLA
402 1247      TAD      COUNT
403 7104      CLL RAL      /MULTIPLY COUNT BY 3 AND ADD
404 1247      TAD      COUNT / YPOINT TO GET ADDRESS OF 1ST
405 1021      YAD YPOINT / WORD OF THE 3 WORDS CONTAINING
406 3250      DCA ADDR1 / NEXT 2 CHANNELS
407 1250      TAD ADDR1
410 7001      IAC
411 3251      DCA ADDR2 / ADDRESS OF 2ND WORD OF THE 3
412 1251      TAD ADDR2
413 7001      IAC
414 3252      DCA ADDR3 / ADDRESS OF 3RD WORD OF THE 3
/
/      /PICK UP THE TWO ADJACENT CHANNELS
415 1651      TAD I ADDR2 / AND SUM THEIR CONTENTS
416 255      AND MASK1
417 3254      DCA HI2 / HI-ORDER PART OF 2ND CHANNEL
420 1651      TAD I ADDR2
421 7421      MQL
422 1650      TAD I ADDR1
423 7417      LSR      / (LINK IS CLEARED HERE)
424 5      0005
425 3253      DCA HI1 / HI-ORDER PART OF 1ST CHANNEL
426 7501      MQA
427 1652      TAD I ADDR3 / SUM THE TWO LO-ORDER PARTS
430 7421      MQL / (AC IS CLEARED)
431 7430      SZL / SKIP IF NO OVERFLOW
432 7001      IAC
433 1253      TAD HI1 / ADD THE HI-ORDER PARTS
434 1254      TAD HI2
435 7413      SHL
436 16      SCALE, 16
437 6063      DYL / WE ASSUME THE SUM NEVER
440 7200      CLA / EXCEEDS 17 BITS,
441 1247      TAD COUNT
442 6057      DXS
443 7001      IAC
444 256      AND MASK2 / COUNT MUST NOT EXCEED 1024 (10)
445 3247      DCA COUNT
446 5600      JMP I SCOPE
447 0      COUNT, 0000
450 0      ADDR1, 0000

```


451	0	ADDR2,	0000
452	0	ADDR3,	0000
453	0	H11,	0000
454	0	H12,	0000
455	77	MASK1,	0077
456	1777	MASK2,	1777

SYMBOL TABLE

ADDR1	450
ADDR2	451
ADDR3	452
COUNT	447
H11	453
H12	454
MASK1	455
MASK2	456
SCALE	436
SCOPE	400
SCOPEP	20
YPOINT	21

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

```

/ SUM -- A ROUTINE FOR INTEGRATING BETWEEN SPECIFIED CHANNELS
/
/   STARTING ADDRESS = 460
/
/   ROUTINE HALTS TWICE FOR OPERATOR TO KEY IN THE LIMITS OF
/   THE INTEGRATION. THE LIMITS AND THE VALUE OF THE SUM ARE
/   PRINTED OUT.
/
/   NORMAL STOP HAS 7777 IN AC.
YP0INT=21
PRINTP=24
ENTRY6=31
ENTRY7=30
LINEF=37
*460
460 7402 BEGIN, HLT
461 6046 TLS
462 7604 LAS /KEY IN LOWER LIMIT AND HIT CONTINUE
463 3373 DCA C1
464 7402 HLT
465 7604 LAS
466 3374 DCA C2 /KEY IN UPPER LIMIT
467 4437 JMS I LINEF /RETURN AND SKIP TO NEW LINE
470 1373 TAD C1
471 7421 MQL
472 4424 JMS I PRINTP / PRINT LOWER LIMIT
473 5431 JMP I ENTRY6
474 1374 TAD C2
475 7421 MQL
476 4424 JMS I PRINTP / PRINT UPPER LIMIT
477 5431 JMP I ENTRY6
/
500 4311 JMS SUMM
501 4424 JMS I PRINTP / PRINT VALUE OF INTEGRAL
502 5430 JMP I ENTRY7
503 7240 CLA CMA
504 7402 HLT / NORMAL STOP
505 77 MA0077, 0077
506 0 TEMP, 0000
507 0 ADDR1, 0000
510 0 ADDR2, 0000
/
/ SUMM -- SUBROUTINE FOR PERFORMING THE INTEGRATION BETWEEN 2 CHANNELS
/
/ CHANNEL NOS. OF LOWER AND UPPER LIMITS MUST BE STORED IN C1,C2
/ THE DOUBLE PRECISION SUM IS RETURNED IN AC,MO
/
511 0 SUMM, 0000
512 7200 CLA
513 1374 TAD C2
514 7040 CMA
515 1373 TAD C1
516 3370 DCA COUNT
517 3371 DCA H
520 3372 DCA L
521 7300 LOOP, CLA CLL /LOOP TO COMPUTE THE INTEGRAL. C1 ALWAYS
522 1373 TAD C1 / CONTAINS NO. OF NEXT CHANNEL TO BE ADDED.
523 7010 RAR / SHIFTS A I INTO LINK IF CHAN. NO. IS ODD
524 1373 TAD C1

```

525	1021		TAD	YPOINT	
526	3307		DCA	ADDR1	/ ADDRESS OF 1ST WORD OF THE CHANNEL
527	1307		TAD	ADDR1	
530	7001		IAC		
531	3310		DCA	ADDR2	/ ADDRESS OF 2ND WORD OF THE CHANNEL
532	7430		SZL		/ SKIP FOR EVEN=NO, CHANNEL
533	5342		JMP	ODD	
534	1710		TAD	I ADDR2	
535	7421		MQL		
536	1707		TAD	I ADDR1	
537	7417		LSR		
540	5		0005		
541	5346		JMP	SUM	
542	1710	ODD,	TAD	I ADDR2	
543	7421		MQL		
544	1707		TAD	I ADDR1	
545	305		AND	MA0077	
546	7100	SUM,	CLL		
547	3306		DCA	TEMP	/ HI-ORDER PART OF NEXT CHAN.
550	7501		MQA		
551	1372		TAD	L	/ ADD THE LO-ORDER PARTS
552	3372		DCA	L	
553	7430		SZL		/ SKIP ON NO OVERFLOW
554	7001		IAC		
555	1306		TAD	TEMP	
556	1371		TAD	H	
557	3371		DCA	H	
560	2373		ISZ	CI	
561	7000		NOP		
562	2370		ISZ	COUNT	/ SKIP AFTER LAST CHANNEL ADDED TO INTEGRAL
563	5321		JMP	LOOP	
564	1372		TAD	L	
565	7421		MQL		
566	1371		TAD	H	
567	5711		JMP	I SUMM	
570	0	COUNT,	0000		
571	0	H,	0000		
572	0	L,	0000		
/					
*573					
/ INVARIANT ADDRESSES FOR C1 AND C2					
/					
573	0	C1,	0000		
574	0	C2,	0000		
/					
/ SUMP == A DUMMY ROUTINE POINTING TO SUMM					
/					
575	0	SUMP,	0000		
576	4311		JMS	SUMM	
577	5775		JMP	I SUMP	

SYMBOL TABLE

ADDR1	507
ADDR2	510
BEGIN	460
C1	573
C2	574
COUNT	570
ENTRY6	31
ENTRY7	30

H	571
L	572
LINEF	37
LOOP	521
MA0077	505
ODD	542
PRINTP	24
SUM	546
SUMM	511
SUMP	575
TEMP	506
YPOINT	21

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/ CHPRNT *- A ROUTINE FOR PRINTING 18-BIT CHANNELS

/ CALLING SEQUENCE JMS I PRINTP /PAGE 0 POINTER TO THIS ROUTINE
 / YPOINT /PAGE 0 POINTER TO 1ST CHANNEL
 / XXXX /CHAN. NO. OF 1ST CHAN. DESIRED
 / XXXX /CHAN. NO. OF 2ND CHAN. DESIRED
 / XXXX /TOLERANCE FOR PRINTING
 / (CONTINUE) /CONTROL RETURNED HERE

/ THE CHANNELS ARE PRINTED IN DECADES IN THE FOLLOWING FORMAT

/ YYYX XXXXX XXXXX XXXXX XXXXX XXXXX XXXXX XXXXX XXXXX XXXXX

/ WHERE YYYX IS A CHANNEL NO. FROM THE SEQUENCE 0,10,20,30,...
 / AND BEGINS IN THE 1ST COL. OF THE PAGE, AND THE XXXXX ARE
 / THE CONTENTS OF CHANNELS YYYX THRU YYYX+9. EACH XXXXX
 / REPRESENTS A NUMBER OF UP TO 6 DIGITS WHOSE LEADING
 / ZEROS ARE NOT PRINTED.

/ IF SUM OF CHANNELS IN A DECADE IS BELOW TOLERANCE THEN PRINTING
 / IS SUPPRESSED FOR THAT DECADE.

/ LINEF=0037
 / TYPE0=0040
 / PRINTP=0024
 / ENTRY6=0031
 / ENTRY4=0033
 / *0600

600	0	CHPRNT,	0000	
601	7200	CLA		
602	1600	TAD	I	CHPRNT
603	3352	DCA		ORIGIN /ADDR. OF 1ST WORD OF 1ST CHANNEL
604	2200	ISZ		CHPRNT
605	1600	TAD	I	CHPRNT
606	3353	DCA		CHAN1 /NO. OF 1ST CHANNEL DESIRED
607	2200	ISZ		CHPRNT
610	1600	TAD	I	CHPRNT
611	3354	DCA		CHAN2 /NO. OF LAST CHANNEL DESIRED
612	2200	ISZ		CHPRNT
613	1600	TAD	I	CHPRNT /TOLERANCE FOR PRINTING
614	373	AND		L3777 /TOLERANCE MUST BE BELOW 2048(10)
615	7041	CIA		
616	3371	DCA		MTQIER
617	2200	ISZ		CHPRNT /RETURN ADDRESS
620	7200	CLA		
621	3355	DCA		CHNNO / 1 INTO CHNNO
622	1353	TAD		CHAN1
623	3356	DCA		TEMP
624	1356	LOOP1,	TAD	TEMP /LOOP TO COMPUTE INITIAL CHANNEL
625	1364	TAD		M10 / OF 1ST DECADE TO PRINT.
626	7510	SPA		
627	5235	JMP		CNT
630	3356	DCA		TEMP
631	1355	TAD		CHNNO
632	1363	TAD		TEN
633	3355	DCA		CHNNO
634	5224	JMP		LOOP1

635	7200	CONT,	CLA		/UPON EMERGING FROM LOOP1 CHNN0 CONTAINS
636	1355		TAD	CHNN0	/ NO. OF 1ST CHANNEL TO PRINT
637	3356		DCA	TEMP	/NOW COMPUTE ADDR. OF 1ST WORD OF
640	1356		TAD	TEMP	/ 1ST CHAN. TO PRINT
641	7110		CLL	RAR	
642	1356		TAD	TEMP	
643	1752		TAD	i ORIGIN	
644	3357		DCA	ADDR1	/ ADDR. OF 1ST WORD
645	1354		TAD	CHAN2	
646	7041		CIA		
647	3356		DCA	TEMP	/ NEG. OF LAST CHAN. DESIRED
650	4437		JMS	i LINEF	/ SKIP TO A NEW LINE
651	4437		JMS	i LINEF	/ AC LEFT CLEARED
652	1355	LOOP2,	TAD	CHNN0	
653	3774		DCA	i C1ADDR	/1ST CHANNEL IN DECADE
654	1355		TAD	CHNN0	
655	1372		TAD	NINE	
656	3775		DCA	i C2ADDR	/LAST CHANNEL IN DECADE
657	4776		JMS	i SUMP	/GO TO INTEGRATING ROUTINE
660	7440		SZA		
661	5274		JMP	PRNT	/PRINT IF HIGH ORDER PART OF SUM IS NON ZERO
662	7501		MOA		/LOAD LOW ORDER PART OF SUM INTO AC
663	7510		SPA		
664	5274		JMP	PRNT	/ IF IT EXCEEDS 2047(10) PRINT THE DECADE
665	1371		TAD	MTOLER	/ OTHERWISE CHECK IT AGAINST TOLERANCE
666	7700		SMA	CLA	
667	5274		JMP	PRNT	/PRINT IF GREATER THAN TOLERANCE
670	1357		TAD	ADDR1	/ IF SUM OF CHANNELS IN DECADE BELOW
671	1365		TAD	L17	/ TOLERANCE LEVEL
672	3357		DCA	ADDR1	
673	5340		JMP	TEST	
674	7200	PRNT,	CLA		
675	1364		TAD	M10	
676	3361		DCA	COUNT1	
677	1355		TAD	CHNN0	
700	7421		MQL		
701	4424		JMS	i PRINTP	/ PRINT CHAN. NO. OF 1ST CHAN. IN
702	5433		JMP	i ENTRY4	/ DECADE FOLLOWED BY 2 SPACES
703	7200		CLA		
704	1367		TAD	SPACE	
705	4440		JMS	i TYPE0	
706	1367		TAD	SPACE	
707	4440		JMS	i TYPE0	/ LEAVES AC CLEARED
710	7100		CLL		
711	1357	INLP2,	TAD	ADDR1	/ LOOP FOR PRINTING THE 10
712	7001		IAC		/ CHANNELS IN THE DECADE
713	3360		DCA	ADDR2	
714	1760		TAD	i ADDR2	
715	7421		MQL		/ LO-ORDER PART OF CHANNEL INTO MO
716	1757		TAD	i ADDR1	/ HI-ORDER PART INTO AC
717	7430		SZL		/ ZERO LINK MEANS CHAN. NO. IS ODD
720	5327		JMP	EVEN	
721	7417		LSR		
722	5		0005		
723	4424		JMS	i PRINTP	
724	5431		JMP	i ENTRY6	
725	7120		STL		
726	5334		JMP	INDEX	

727	370	EVEN,	AND	MASK1	
730	4424		JMS	1	PRINTP
731	5431		JMP	1	ENTRY6
732	7100		CLL		
733	2357		ISZ	ADDR1	
734	2357	INDEX,	ISZ	ADDR1	
735	2361		ISZ	COUNT1	/ SKIP AFTER PRINTING 10 CHANNELS
736	5311		JMP	INLP2	
737	4437		JMS	1	LINEF / CARRIAGE RETURN AND LINE FEED
					/ AC IS CLEARED ON RETURN
740	1355	TEST,	TAD	CHNN0	/ NOW CHECK TO SEE IF ALL DECADES
741	1356		TAD	TEMP	/ HAVE BEEN PRINTED
742	1363		TAD	TEN	
743	7540		SMA	SZA	
744	5600		JMP	1	CHPRNT / WHEN ALL DECADES HAVE BEEN PRINTED
745	7200		CLA		
746	1355		TAD	CHNN0	
747	1363		TAD	TEN	
750	3355		DCA	CHNN0	
751	5252		JMP	LOOP2	
752	0	ORIGIN,	0000		
753	0	CHAN1,	0000		
754	0	CHAN2,	0000		
755	0	CHNN0,	0000		
756	0	TEMP,	0000		
757	0	ADDR1,	0000		
760	0	ADDR2,	0000		
761	0	COUNT1,	0000		
762	0	COUNT2,	0000		
763	12	TEN,	0012		
764	7766	M10,	7766		
765	17	L17,	0017		
766	7761	M15,	7761		
767	240	SPACE,	0240		/ASCII CODE FOR SPACING
770	77	MASK1,	0077		
771	0	MTOLER,	0000		
772	11	NINE,	0011		
773	3777	L3777,	3777		
774	573	CIADDR,	573		
775	574	C2ADDR,	574		
776	575	SUMP,	575		

SYMBOL TABLE

ADDR1	757
ADDR2	760
CIADDR	774
C2ADDR	775
CHAN1	753
CHAN2	754
CHNN0	755
CHPRNT	600
C0NT	635
C0UNT1	761
C0UNT2	762
ENTRY4	33
ENTRY6	31
EVEN	727
INDEX	734
INLP2	711
L17	765

L3777	773
LINEF	37
LOOP1	624
LOOP2	652
M10	764
M15	766
MASK1	770
MTOLER	771
NINE	772
ORIGIN	752
PRINTP	24
PRNT	674
SPACE	767
SUMP	776
TEMP	756
TEN	763
TESTY	740
TYPE0	40

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/ MOE -- MAIN ROUTINE FOR MOSELEY PLOTTING

BUFFP=22

EANIE=1047

COMAND=44

SCOPEP=20

*1040

1040	4247	MOE,	JMS	EANIE
1041	1600		1600	
1042	4247		JMS	EANIE
1043	4600		4600	
1044	4444		JMS I	COMAND
1045	4420		JMS I	SCOPEP
1046	5244		JMP	.+2

SYMBOL TABLE

BUFFP	22
COMAND	44
EANIE	1047
MOE	1040
SCOPEP	20

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

```

/ EANIE   GRAPHS ONE 512 SECTION OF THE 2048 WORD KLUDGE BUFFER,
/ TAKING TWO CHANNELS AT A TIME.
/ CALLING SEQUENCE..... JMS EANIE
/
SCALE0=43
SHIFT=1005 /MOSELEY JAM SUBROUTINE TO AVOID ROLL-OVER
*1047
1047 0 EANIE, 0
1050 1647 TAD I EANIE
1051 3346 DCA YPOINT
1052 6063 DYL
1053 6053 DXL
1054 7402 HLT /LOWER PEN AND CONTINUE
1055 7200 CLA
1056 1443 TAD I SCALE0
1057 3325 DCA SCALE
1060 3334 DCA OLDX /OLDX = 0 AND OLDY = 0
1061 3332 DCA OLDY
1062 3350 DCA COUNT
1063 1345 TAD P512 /SET UP LOOP
1064 7041 CIA
1065 3347 DCA SKIP
1066 1334 LOOP, TAD OLDX / GET NEWX = OLDX + 2
1067 1344 TAD P2
1070 3333 DCA NEWX
1071 1350 TAD COUNT
1072 7104 CLL RAL /MULTIPLY COUNT BY 3 AND ADD
1073 1350 TAD COUNT / YPOINT TO GET ADDRESS OF 1ST
1074 1346 TAD YPOINT / WORD OF THE 3 WORDS CONTAINING
1075 3351 DCA ADDR1 / NEXT 2 CHANNELS
1076 1351 TAD ADDR1
1077 7001 IAC
1100 3352 DCA ADDR2 / ADDRESS OF 2ND WORD OF THE 3
1101 1352 TAD ADDR2
1102 7001 IAC
1103 3353 DCA ADDR3 / ADDRESS OF 3RD WORD OF THE 3
/
/
1104 1752 TAD I ADDR2 /PICK UP THE TWO ADJACENT CHANNELS
1105 354 AND MASK1 / AND SUM THEIR CONTENTS
1106 3356 DCA HI2 / HI-ORDER PART OF 2ND CHANNEL
1107 1752 TAD I ADDR2
1110 7421 MOL
1111 1751 TAD I ADDR1
1112 7417 LSR / (LINK IS CLEARED HERE)
1113 5 D005
1114 3355 DCA HI1 / HI-ORDER PART OF 1ST CHANNEL
1115 7501 MOA
1116 1753 TAD I ADDR3 / SUM THE TWO LO-ORDER PARTS
1117 7421 MOL / (AC IS CLEARED)
1120 7430 SZL / SKIP IF NO OVERFLOW
1121 7001 IAC
1122 1355 TAD HI1 / ADD THE HI-ORDER PARTS
1123 1356 TAD HI2
1124 7413 SHL
1125 16 SCALE, 16
1126 360 AND L1777
1127 3331 DCA NEWY
1130 4757 JMS I GRAPH

```

1131	0	NEWY,	00	
1132	0	OLDY,	00	
1133	0	NEWX,	00	
1134	0	OLDX,	00	
1135	2350		ISZ	COUNT
1136	2347		ISZ	SKIP
1137	5266		JMP	LOOP
1140	7200		CLA	
1141	7402		HLT	/RAISE PEN AND CONTINUE
1142	2247		ISZ	EANIE
1143	5647		JMP	I EANIE
1144	2	P2,	2	
1145	777	P512,	777	
1146	0	YP0INT,	0	
1147	0	SKIP,	0	
1150	0	COUNT,	0	
1151	0	ADDR1,	0	
1152	0	ADDR2,	0	
1153	0	ADDR3,	0	
1154	77	MASK1,	0077	
1155	0	HI1,	0	
1156	0	HI2,	0	
1157	1200	GRAPH,	1200	
1160	1777	L1777,	1777	

SYMBOL TABLE

ADDR1	1151
ADDR2	1152
ADDR3	1153
COUNT	1150
EANIE	1047
GRAPH	1157
HI1	1155
HI2	1156
L1777	1160
LOOP	1066
MASK1	1154
NEWX	1133
NEWY	1131
OLDX	1134
OLDY	1132
P2	1144
P512	1145
SCALE	1125
SCALE0	43
SHIFT	1005
SKIP	1147
YP0INT	1146

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

	/DELAY		CLA	
	/		TAD	N
	/		JMS	DELAY
	/		M	
	/		(RETURN)	
	/			
	•1161			
1161	0	DELAY,	00	
1162	7040		CMA	
1163	3377		DCA	N
1164	2377	LOOP,	ISZ	N
1165	5370		JMP	+3
1166	2361		ISZ	DELAY
1167	5761		JMP	I DELAY /RETURN WITH (AC) = 0
1170	1761		TAD	I DELAY
1171	7041		CIA	
1172	3376		DCA	ROLL0
1173	2376		ISZ	ROLL0
1174	5373		JMP	+1
1175	5364		JMP	LOOP
1176	0	ROLL0,	0000	
1177	0	N,	0000	

SYMBOL TABLE

DELAY	1161
LOOP	1164
N	1177
ROLL0	1176

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

```

/ GRAPH -- A POINT MOVING ROUTINE FOR THE MOSELEY PLOTTER
/ CALLING SEQUENCE... JMS GRAPH
/ NEWY
/ OLDY
/ NEWX
/ OLDX
/
/ MOVES PEN FROM OLD Y TO NEW Y. THEN OLDY = NEWY.
/ NEXT MOVES PEN FROM OLD X TO NEW X. THEN OLDX = NEWX.
/ FULL SCALE = 1024(10)
/
DELAY=42
*1200
GRAPH, 0
1200 0 CLA
1201 7200 CLA
1202 1600 TAD I GRAPH
1203 3265 DCA NEWY
1204 1265 TAD NEWY
1205 7041 CIA
1206 2200 ISZ GRAPH
1207 1600 TAD I GRAPH /OLDX = NEWX
1210 7510 SPA
1211 7041 CIA
1212 3266 DCA DIF
1213 1265 TAD NEWY
1214 3600 DCA I GRAPH /OLDY = NEWY
1215 2200 ISZ GRAPH
1216 1600 TAD I GRAPH
1217 3267 DCA NEWX
1220 2200 ISZ GRAPH
1221 1267 TAD NEWX
1222 3600 DCA I GRAPH /OLDX = NEWX
1223 2200 ISZ GRAPH /SET UP RETURN
1224 1266 TAD DIF /C0 PARE DIF AGAINST C1, SAY 10
1225 7041 CIA
1226 1262 TAD CI
1227 7710 SPA CLA
1230 5244 JMP GT
1231 1265 LT, TAD NEWY
1232 6063 DYL
1233 7201 CLA IAC
1234 4442 JMS I DELAY
1235 7000 C2, 7000 /MINIMUM Y DELAY FOR SMALL DIF, SAY 14
1236 1267 TAD NEWX
1237 6053 DXL
1240 7201 CLA IAC
1241 4442 JMS I DELAY
1242 3000 C3, 3000 /MINIMUM X DELAY FOR SMALL DIF, SAY 14
1243 5261 JMP BACK
1244 1265 GT, TAD NEWY
1245 6063 DYL
1246 7200 CLA
1247 1263 TAD C4 /FIXED Y DELAY FOR GREAT DIF, SAY 10
1250 1266 TAD DIF
1251 4442 JMS I DELAY
1252 300 300 /DELAY CONSTANT FOR HORIZONTAL MOTION
1253 1267 TAD NEWX
1254 6053 DXL
1255 7200 CLA

```

1256	1264	TAD	C5	/NOMINAL X DELAY FOR GREAT DIF, SAY 30
1257	4442	JMS I	DELAY	
1260	300		300	
1261	5600	BACK,	JMP I	GRAPH
1262	10	C1,	10	
1263	20	C4,	20	
1264	100	C5,	100	
1265	0	NEWY,	0	
1266	0	DIF,	0	
1267	0	NEWX,	0	

SYMBOL TABLE

BACK	1261
C1	1262
C2	1239
C3	1242
C4	1263
C5	1264
DELAY	42
DIF	1266
GRAPH	1200
GT	1244
LT	1231
NEWX	1267
NEWY	1265

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/ DETEC == DETECTOR SUMMING ROUTINE FOR KLUDGE
/ RETURNS WITH CHANNEL NUMBER IN THE AC.

*1270
SWB=11
SWC=12
SWD=13
SWPILE=14
SWPUL=15
SWFL1=16
SWFL2=17

1270	0	DETEC,	0000		
1271	3353	DCA	ADDR	/ADDR OF WD=0	
1272	3356	DCA	SUM		
1273	1753	TAD	I ADDR		
1274	7004	RAL		/GET PULSER FLAG	
1275	4357	JMS	TEST		
1276	15	SWPUL		/PULSER SWITCH =	
1277	1753	TAD	I ADDR		
1300	7006	RTL		/GET PILE UP FLAG	
1301	4357	JMS	TEST		
1302	14	SWPILE		/PILE UP SWITCH=	
1303	2353	GETB,	ISZ	ADDR	
1304	1753	TAD	I ADDR		
1305	7004	RAL		/GET FLAG2	
1306	4357	JMS	TEST		
1307	17	SWFL2		/FLAG2 SWITCH =	
1310	1011	TAD	SWB		
1311	7650	SNA	CLA		
1312	5320	JMP	GETC	/SKIP OVER B	
1313	1753	TAD	I ADDR		
1314	7510	SPA			
1315	5350	JMP	JAM	/B OVERFLO	
1316	354	AND	L0777		
1317	3356	DCA	SUM		
1320	2353	GETC,	ISZ	ADDR	
1321	1753	TAD	I ADDR		
1322	7004	RAL		/GET FLAG1	
1323	4357	JMS	TEST		
1324	16	SWFL1		/FLAG1 SWITCH	
1325	1012	TAD	SWC		
1326	7650	SNA	CLA		
1327	5336	JMP	GETD		
1330	1753	TAD	I ADDR		
1331	7510	SPA			
1332	5350	JMP	JAM	/C OVERFLO	
1333	355	AND	L1777		
1334	1356	TAD	SUM		
1335	3356	DCA	SUM		
1336	2353	GETD,	ISZ	ADDR	
1337	1013	TAD	SWD		
1340	7650	SNA	CLA		
1341	5345	JMP	DONE		
1342	1753	TAD	I ADDR		
1343	7510	SPA			
1344	5350	JMP	JAM	/D OVERFLO	
1345	1356	DONE,	TAD	SUM	
1346	7500	SMA		/SKIP IN CASE SUM EXCEEDS 3777(8)	
1347	5670	JMP	I DETEC		
1350	7340	JAM,	CLA CMA CLL	/IN CASE THERE WAS AN OVERFLOW	

```

1351 7010      RAR      / RETURN WITH 3777(8) IN THE AC.
1352 5670      JMP I DETEC
1353      0 ADDR, 0000
1354 777 L0777, 0777
1355 1777 L1777, 1777
1356      0 SUM, 0000

```

```

/
/
/SUBROUTINE TEST
/CALLED WITH FLAG BIT IN SIGN POSITION

```

```

/ACTION TABLE      FLAG = 0      FLAG = 1

```

```

/      SW .EQ. 0      PROCEED      PROCEED
/      .LT. 0      JAM      PROCEED
/      .GT. 0      PROCEED      JAM

```

```

/ASSUME SW IS DETERMINED FROM TWO KEYS

```

```

/KEY 1 UP MEANS PROCEED IF FLAG = 0
/      2      = 1

```

```

/      (ONE OR BOTH SWITCHES SHOULD BE UP)

```

```

/KEY TABLE

```

```

/      SWITCH
/      KEY 1 UP      .GT.0
/      KEY 2 UP      .LT.0
/      BOTH UP      .EQ.0

```

```

1357      0 TEST, 00
1360 7710      SPA CLA
1361 5370      JMP FONE
1362 1757 FZERO, TAD I TEST /FLAG = 0
1363 3377      DCA SWADDR
1364 1777      TAD I SWADDR
1365 7710      SPA CLA /PROCEED IF .GE. 0
1366 5350      JMP JAM
1367 5375      JMP BACK
1370 1757 FONE, TAD I TEST /FLAG = 1
1371 3377      DCA SWADDR
1372 1777      TAD I SWADDR
1373 7740      SMA SZA CLA /PROCEED IF .LE. 0
1374 5350      JMP JAM
1375 2357 BACK, ISZ TEST
1376 5757      JMP I TEST

```

```

1377      0 SWADDR, 0000

```

SYMBOL TABLE

```

ADDR      1353
BACK      1375
DETEC     1270
DONE      1345
FONE      1370
FZERO     1362
GETB      1303
GETC      1320
GETD      1336
JAM       1350

```


L0777	1354
L1777	1355
SUM	1356
SWADDR	1377
SWB	11
SWC	12
SWD	13
SWFL1	16
SWFL2	17
SWPILE	14
SWPUL	15
TEST	1357

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

```

/ PRINT AND DECODE
/ (WRITTEN BY DAN PUTZOLU)
/
TYPE0=40
*1400
1400 0 PRINT, 0000
1401 7100 CLL
1402 5600 JMP I PRINT
1403 4236 ENTRY8, JMS DECODE
1404 3166 3166
1405 4600 4600
1406 4236 ENTRY7, JMS DECODE
1407 7413 7413
1410 6700 6700
1411 4236 ENTRY6, JMS DECODE
1412 7747 7747
1413 4540 4540
1414 4236 ENTRY5, JMS DECODE
1415 7775 7775
1416 4360 4360
1417 4236 ENTRY4, JMS DECODE
1420 7777 7777
1421 6030 6030
1422 4236 ENTRY3, JMS DECODE
1423 7777 7777
1424 7634 7634
1425 4236 ENTRY2, JMS DECODE
1426 7777 7777
1427 7766 7766
1430 7120 STL
1431 4236 ENTRY1, JMS DECODE
1432 7777 7777
1433 7777 7777
1434 2200 ISZ PRINT
1435 5600 JMP I PRINT
/
/
/ *****DECODE*****
/ DECODE IS AN ALGORITHM FOR CONVERSION OF OCTAL NUMBERS TO
/ THEIR DECIMAL EQUIVALENT. THE PROGRAM OPERATES ON THE PRIN-
/ CIPAL THAT POWERS OF TEN IN OCTAL NOTATION CAN BE SUBTRACTED
/ FROM AN OCTAL NUMBER THUS CONVERTING IT TO ITS DECIMAL EQUI-
/ VALENT. CONNECTING TWO TWELVE BIT WORDS BY MEANS OF THE
/ LINK BIT, DECODE CAN CONVERT POSITIVE NUMBERS IN THE RANGE
/ *00000000.GT.N(8).LT.*77777777. DECODE PRINTS ONE DECIMAL
/ DIGIT AT A TIME ALLOWING THE USER A CHOICE OF FORMATS. I.E.
/ 11 TO 18. IF THE FORMAT THE USER CHOOSES CANNOT CONTAIN
/ THE NUMBER, A / IS PRINTED. THE ROUTINE RUNS CONCURRENT-
/ LY WITH A SECOND SUBROUTINE (CONVRT) WHICH YIELDS THE
/ COMPLEMENTS OF POWERS OF TEN IN OCTAL AND SUPPRESSES LEADING
/ ZEROS,
/
1436 0 DECODE, 0
1437 3325 DCA H
1440 7501 7501 /MQA
1441 3326 DCA L
1442 7420 SNL
1443 1336 TAD M20 /LINK=0, SPACE=*20
1444 3333 DCA SPACE /LINK=1, SPACE=0
1445 3335 DCA COUNT

```

1446	1636		TAD	I	DECODE	
1447	3327		DCA		CH	
1450	2236		ISZ		DECODE	
1451	1636		TAD	I	DECODE	
1452	3330		DCA		CL	
1453	2236		ISZ		DECODE	
1454	7300	START,	CLA	CLL		
1455	1326		TAD		L	
1456	1330		TAD		CL	
1457	3331		DCA		L2	
1460	7420		SNL			
1461	5264		JMP	A1	/NO CARRY FROM WORD ONE ADDITION.	
1462	7201		CLA	IAC	/CARRY FROM ADDITION.	
1463	3334		DCA		J	
1464	7300	A1,	CLA	CLL		
1465	1325		TAD		H	
1466	1327		TAD		CH	
1467	1334		TAD		J	
1470	3332		DCA		H2	
1471	3334		DCA		J	
1472	7430		SZL			
1473	5315		JMP	A3	/CONTINUE SUBTRACTING POWER OF TEN.	
1474	7100		CLL		/WRITE OUT DECIMAL DIGIT	
1475	1335		TAD		COUNT	
1476	1337		TAD		M12	
1477	7710		SPA	CLA		
1500	5303		JMP		OKPRT	
1501	7340		CLA	CMA	CLL	
1502	5307		JMP		TYPE	
1503	1335	OKPRT,	TAD		COUNT	
1504	7450		SNA			
1505	1333		TAD		SPACE	
1506	7120		STL			
1507	1340	TYPE,	TAD		P260	
1510	4440		JMS	I	TYPE0	
1511	1326		TAD		L	
1512	7421		7421		/MQL	
1513	1325		TAD		H	
1514	5636		JMP	I	DECODE	
1515	1335	A3,	TAD		COUNT	
1516	7001		IAC			
1517	3335		DCA		COUNT	
1520	1332		TAD		H2	
1521	3325		DCA		H	
1522	1331		TAD		L2	
1523	3326		DCA		L	
1524	5204		JMP		START	
1525	0	H,	0			
1526	0	L,	0			
1527	0	CH,	0			
1530	0	CL,	0			
1531	0	L2,	0			
1532	0	H2,	0			
1533	0	SPACE,	0			
1534	0	J,	0			
1535	0	COUNT,	0			
1536	7760	M20,	7760			
1537	7766	M12,	7766			
1540	260	P260,	260			

SYMBOL TABLE

A1	1464
A3	1515
CH	1527
CL	1530
COUNT	1535
DECODE	1436
ENTRY1	1431
ENTRY2	1425
ENTRY3	1422
ENTRY4	1417
ENTRY5	1414
ENTRY6	1411
ENTRY7	1406
ENTRY8	1403
H	1525
H2	1532
J	1534
L	1526
L2	1531
M12	1537
M20	1536
OKPRT	1503
P260	1540
PRINT	1400
SPACE	1533
START	1454
TYPE	1507
TYPE0	40

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/ LINEF -- CARRIAGE RETURN AND LINE FEED ROUTINE

/ USE TYPE0 TO DO THE ACTUAL TYPING

TYPE0=40

*1545

1545	0	LINEF,	0000	
1546	7200	CLA		
1547	1355	TAD	CGRTN	
1550	4440	JMS	TYPE0	
1551	4440	JMS	TYPE0	/PROVIDES EXTRA DELAY FOR WEAK TELEPRINTER
1552	1356	TAD	LNFEED	
1553	4440	JMS	TYPE0	
1554	5745	JMP	LINEF	
1555	0215	CGRTN,	0215	/ASCII CODE FOR CARRIAGE RETURN
1556	0212	LNFEED,	0212	/ASCII CODE FOR LINE FEED

SYMBOL TABLE

CGRTN	1555
LINEF	1545
LNFEED	1556
TYPE0	40

DUPLICATE TAGS:

NONE

UNDEFINED SYMBOLS

NONE

```

/ TYPE0 -- SUBROUTINE FOR TYPING OUT THE CONTENTS OF THE AC.
/ THE LINK IS SAVED AND RESTORED AFTER TYPING
/

```

```

COMMAND=44

```

```

*1560

```

```

SCOPEP=20

```

```

1560      0  TYPE0, 0
1561      6046      TLS
1562      7010      RAR
1563      3375      DCA      LINK      /SAVE THE LINK
1564      4420      JMS      I  SCOPEP
1565      6041      TSF
1566      5364      JMP      .+2
1567      4444      JMS      I  COMMAND
1570      7200      CLA
1571      1375      YAD      LINK
1572      7104      CLL RAL      /RESTORE THE LINK
1573      7200      CLA
1574      5760      JMP      I  TYPE0
1575      0  LINK, 0000

```

SYMBOL TABLE

```

COMMAND      44

```

```

LINK          1575

```

```

SCOPEP        20

```

```

TYPE0         1560

```

DUPLICATE TAGS:

```

NONE

```

UNDEFINED SYMBOLS

```

NONE

```

*1620

/ SCOPECAL ** A STRAIGHT LINE SCOPE CALIBRATION PROGRAM

1620	6075	SCAL,	DSB+1
1621	7001	IAC	
1622	6053	DXL	
1623	6063	DYL	
1624	6054	DIX	
1625	5221	JMP	SCAL+1

SYMBOL TABLE

SCAL 1620

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

INTERNAL DISTRIBUTION

1-3.	L. S. Abbott	25.	F. G. Perey
4.	C. L. Allen (CTC)	26.	B. W. Rust
5.	R. G. Alsmiller, Jr.	27.	E. G. Silver
6.	F. E. Bertrand	28.	J. W. Wachter
7.	A. A. Brooks	29.	L. W. Weston
8-10.	W. R. Burrus	30.	G. Dessauer (consultant)
11.	C. E. Clifford	31.	B. C. Diven (consultant)
12.	G. deSaussure	32.	W. N. Hess (consultant)
13.	R. E. Funderlic (CTC)	33.	M. H. Kalos (consultant)
14.	W. A. Gibson	34.	L. V. Spencer (consultant)
15.	T. A. Love	35-36.	Central Research Library
16.	F. Madden	37.	Document Reference Section
17-18.	F. C. Maienschein	38-215.	Laboratory Records Department
19.	E. McDaniel	216.	Laboratory Records ORNL RC
20-24.	R. W. Peelle	217.	ORNL Patent Office

EXTERNAL DISTRIBUTION

- 218. P. B. Hemmig, Division of Reactor Development and Technology,
U. S. Atomic Energy Commission, Washington, D. C. 20545
- 219. I. F. Zartman, Reactor and Development, U. S. Atomic Energy
Commission, Washington, D. C. 20545
- 220-234. Division of Technical Information Extension (DTIE)
- 235. Division of Reactor Development (ORO)
- 236. Research and Development Div., ORO