JUN 29 1967

OAK RIDGE NATIONAL LABORATORY

operated by

UNION CARBIDE CORPORATION

NUCLEAR DIVISION

for the

U.S. ATOMIC ENERGY COMMISSION

ORNL - TM - 1878

CFSTI PRICES

COPY NO. - 257

H.C. \$ 3.00; MN. 65

DATE - June 1, 1967

SUPER-FLICKERS -

AN ON-LINE DATA ACQUISITION SYSTEM FOR THE PDP-8

B. W. Rust[†] and W. R. Burrus

ABSTRACT

SUPER-FLICKERS is a system of PDP-8 programs for accumulating and processing 4 parameter data from pulse height and time-of-flight analyzers. It writes all incoming data on magnetic tape for later analysis by a larger computer and provides oscilloscope displays of various 2-dimensional "flicker boxes" on an event by event basis. It also accumulates and plots certain one dimensional combination spectra which are of interest. Program options are controlled by key switches and plot scaling options are controlled by teletype keys.

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

Peg

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 DESCRIPTION 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 are written for a 256-word buffer beginning in core location 1000. 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 occasionally 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 old 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	<u>B</u> reak <u>e</u> n <u>A</u> BLE in Break control -- enables the data break and allows data to start coming in.
BDISAB	<u>B</u> reak <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> lear -- 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 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 program is 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 C, then D. It is possible for the particle to stop in B or C or D but not to get 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 D. Figure 1 is a schematic diagram of the 48 bits of input. For each 12 bit word the first bit (labeled OV) is an overflow bit. The bits labeled F1, F2, F3, etc are flag bits and the bits labeled 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 the C detectors,

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 for Each Event

- 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 operating procedure for a normal start is as follows:

1. Start in location 200 -- computer halts,
2. key in code word for ID record and hit continue -- computer will write the ID record and halt again to give the operator a pause to think things over or change his mind,
3. hit continue -- computer will clear the two spectrum buffers, initialize the overflow channel markers and halt again to give the operator one last moment of reflection before letting signals come in through the data break,
4. hit continue -- computer clears the data buffer and starts taking data, writing the data on tape in 100Q₂ word records and plotting or flickering data on the scope depending upon which option is specified in the key switches.

Other permissable starting locations for the FLICKERS system are:

- 372 a restart which gives the same effect as the execution of step three above,
- 373 a restart which flushes out the partially filled tape record buffer and then continues normal program operation (if no data is coming in then it simply keeps going through the anti-freeze loop),
- 377 a restart which gives the same effect as the execution of step 4 above,

- 120 an entry for independently plotting the contents of the spectrum buffers without data coming into the machine (the option switches should be set for options 1 or 2 since the other options would not be very interesting when no data is coming in),
- 130 an entry for continuously displaying a diagonal line across the scope for calibration purposes.

During execution the display option is determined by the contents of the key switches. The following options are available:

0. does nothing;
1. continuously plots the contents of the B + C buffer which contains the spectrum of the particles which are stopped in the C detector;
2. continuously plots the contents of the B + C + D buffer which contains the spectrum of the particles which are stopped in the D detector;
3. flickers for each event $2C$ vs $(C+D)/2$ with 1023_{10} dots full scale unless the event suffered one of the following disqualifications:
 - a) overflow in C and/or D,
 - b) $2C$ and/or $(C+D)/2$ exceeded 1023_{10} ,
 - c) flag 2 was 0, i.e. no BCD coincidence,
 - d) flag 4 was 1, i.e. pile-up flag was on;
4. flickers for each event $2B$ vs $(B+C)$ with 1023_{10} dots full scale unless the event suffered one of the following disqualifications:
 - a) overflow in B and/or C,
 - b) $2B$ or $(B+C)$ exceeded 1023_{10} ,
 - c) flag 1 was 0 or flag 2 was 1, i.e. No BC coincidence or BCD coincidence as well as BC coincidence,
 - d) flag 4 was 1, i.e. pile-up flag was on;
5. flickers for each event $2(B+C)$ vs $(B+C+D)/2$ with 1023_{10} dots full scale unless the event suffered one of the following disqualifications:

- a) overflow in B, C, or D,
 - b) $2(B+C)$ or $(B+C+D)/2$ exceeded 1023_{10} ,
 - c) flag 2 was 0, i.e. no BCD coincidence,
 - d) flag 4 was 1, i.e. pile-up flag was on;
6. flickers for each event $4T$ vs $2B$ with 1023_{10} dots full scale unless the event suffered one of the following disqualifications:
- a) overflow in T and/or B,
 - b) flag 2 was 1 or flag 1 was 1, i.e. either BCD or BC coincidence;
7. draws a box around the display area for scope calibration.

For the plotting options (1 and 2) the vertical scale is controlled by the typewriter. The operator should hit 0 to give full scale of 127_{10} , 1 to give full scale of 255_{10} , 2 to give full scale of 511_{10} , etc.

If an end of tape is encountered while the program is running the computer will stop after writing the current record. There will still be 14 feet of tape left and the operator should start at 200 and key in a 7777 ID code in step 2 above. When the computer again halts at the end of step 2 the operator can then remove the reel.

III. CORE STORAGE USED BY THE SUPER-FLICKERS SYSTEM

<u>Locations</u>	<u>Contents</u>
0010 -- 0012	Auto-index registers used by subroutine SAVED.
0020 -- 0062	Page 0 pointers and reference locations used for addressing across pages.
0070 -- 0076	Subroutine COMMAND.
0120 -- 0125	Independent plot routine (INDPLOT).
0130 -- 0135	Straight line scope calibrating routine (SCOPECAL).
0140 -- 0156	Subroutine BBIAS.
0170 -- 0176	Subroutine INIT.
0200 -- 0366	Main routine of SUPER-FLICKERS system (FLICKERS).
0372 -- 0377	Restart entry points and invariant return locations for FLICKERS.
0400 -- 0472	Subroutine ERRCK.
0500 -- 0513	Subroutine SAVED.

0550 -- 0567 Subroutine BFCLR.
 0600 -- 0760 The SYSPOP system of tape-popping routines (including
 subroutine WRITE).
 1000 -- 1377 Data Buffer.
 1400 -- 1553 Subroutine FLIC, options 1-4.
 1600 -- 1725 Subroutine FLIC, options 5-7.
 1736 -- 1777 Subroutine ANTFRZ (the anti-freeze subroutine).
 2000 -- 2174 Subroutine SPEC (including subroutines ADD1 and INCR).
 2200 -- 2272 Subroutine INTPLT.
 2376 -- 2575 Spectrum overflow buffer.
 2576 -- 3575 Tape record buffer.
 3576 -- 5576 B+C spectrum for particles that stop in the C detector.
 5577 -- 7577 B+C+D spectrum for the particles that stop in the D
 detector.

IV. PROGRAM DESCRIPTIONS

FLICKERS

FLICKERS is the main program of the SUPER-FLICKERS data acquisition system. It consists chiefly of a large loop which continually scans the data buffer for new data, transfers any new data that it finds to the tape record buffer, accumulates the spectrum of B+C for those particles which stop in the C detector and the spectrum of B+C+D for the particles which stop in the D detector, and jumps to the display routine (FLIC) to plot one of the spectrums or flicker the signals for the event depending upon the option specified in the key switches. In scanning the data buffer the program looks at every fourth location beginning with the first (location 1000) so that on a given pass it is looking to see if there is a T signal for a new event. If there is a new event then flag 3 (bit 3 of the T signal) will have been set to 1 by the logic hardware when the event was inputted. The program jumps to the anti-freeze subroutine (ANTFRZ) to check the flag. If ANTFRZ finds that flag 3 is 0 then it returns control back to the main program at a location that will send the main program back through the same procedure as before, looking at the same location in the data buffer, waiting for a new event to be inputted. If ANTFRZ finds that flag 3 is 1 it returns control to the main program in a location where

the program will process the new event. The main program will then transfer the four words for the event to the tape record buffer, accumulate the event in the spectrum buffers, jump off to subroutine FLIC to flicker the event if a flicker option is specified by the key switches or refresh the plot of the spectrum if a plot option is specified, zero out the T-signal in the data buffer, add 4 to the program parameter (EMARK) which advances the scan through the data buffer so that the next pass through the loop will look for the next event, and subtract 4 from the up-down counter. The program will then jump to the bias checking routine (BBIAS) which checks all the events which stop in detector B (i.e. all those events for which both flag 1 and flag 2 are zero). If either flag 1 or flag 2 is 1 or if they are both 0 but the B signal exceeds the bias then the return from BBIAS is to a location which adds 4 to the program parameter EMARK so that the next event to be inputted will be transferred to the next 4 locations in the tape record buffer. If both flag 1 and flag 2 are 0 and the B signal was less than the bias then the return to the main program is to a location at the beginning of the loop and EMARK is not advanced so that the next event will be written over the event which was just transferred to the tape record buffer. In the former case (the more normal of the two), after advancing EMARK the program then checks to see if the tape record buffer has been filled up. If it has not been filled the program then jumps back to the beginning of the loop to start the whole process again. If the tape record is full the program jumps to subroutine WRITE (one of the subroutines in the SYSPOP tape system) to write the record on tape. For every eighth record that it writes the program jumps to subroutine ERRCK which reads the record back and compares it with the contents of the tape record buffer in order to check the operation of the tape unit.

After writing the record on tape (and perhaps checking it) the program clears the tape record buffer, sets EMARK to zero, jumps to subroutine FLIC to flicker the last event processed or refresh the spectrum plot (depending on the option specified in the key switches), and checks to see if a new scaling constant for plotting has been typed into the keyboard. If no new scaling constant has been typed in, it returns to the beginning of the loop to start the process again. If

a new scaling constant has been typed in, the program jumps off to subroutine COMAND to read it and store it away in the plotting routine (INTPLT) before returning to the beginning of the loop.

There are several starting entries for FLICKERS which cause the program to carry out various tasks before entering the main loop described in the preceding paragraph. A description of these entries and what the program does when they are used is given in section II.

ANTFRZ

ANTFRZ is the subroutine which FLICKERS uses to test a T-signal location in the data buffer to see if it contains an event to be processed. ANTFRZ is called with the contents of the T-signal location in the AC. ANTFRZ checks the flag 3 position of this actual or potential T-signal. If it finds a 1 in this flag 3 bit then there is an event to be processed and ANTFRZ returns to a location in FLICKERS which starts the processing. If flag 3 is 0 then there is no event to be processed and ANTFRZ counts up a counter which overflows on a count of 40,000₈ and, if there is no overflow, returns to the main program at the beginning of the main loop, first jumping to FLIC if a plot option is specified in the switch register. If there is an overflow ANTFRZ rings the bell on the teletype, resets the counter to 0 and returns control to FLICKERS in a location which causes the program to clear the data buffer and the tape record buffer before going back to the beginning of the main loop.

INIT

INIT is a subroutine called by FLICKERS in the normal starting procedure, before entering the main loop, to initialize the teletype keyboard and the teleprinter and to set the scope to maximum brightness level.

COMAND

COMAND is a subroutine called by FLICKERS when it receives a signal typed in from the teletype keyboard. COMAND converts the signal from an ASCII code to an octal integer and stores the result in the

scaling constant location in the integrating plot routine (INTPLT). It then returns control to the main program.

BBIAS

BBIAS is a subroutine which is called by FLICKERS to set a threshold of acceptance for the events corresponding to particles which stop in the B detector. If either flag 1 or flag 2 is 1 then the particle penetrated to the C or D detector and BBIAS gives a normal return to FLICKERS. If both flag 1 and flag 2 are 0, the particle stopped in the B detector and BBIAS checks the B signal against a tolerance level which is set by a parameter in the program. If the B signal exceeds the tolerance then BBIAS gives a normal return to FLICKERS but if the signal does not exceed the bias then BBIAS gives a return which skips the section of FLICKERS which advances the tape record buffer marker EMARK so that the data for the next event will be written over the data for the present event thus effectively discarding the current event.

ERRCK

ERRCK is a subroutine called by FLICKERS to check every 8th record which it writes on tape. ERRCK is called immediately after the record is written. It backspaces the tape over the record and then reads the record comparing each word that it reads with the corresponding word in the tape record buffer. If it detects a discrepancy between the word on tape and the corresponding word in the buffer, or if it detects a size error (the tape record being more than or less than 1000 words), or if it detects a parity error, then ERRCK prints an "E" on the teleprinter before returning control to FLICKERS.

SAVED

SAVED is a subroutine called by FLICKERS to transfer the four words for each event to the tape record buffer and to mask off the flags from the four words and store the resulting "trimmed" signals in locations on page 0 that can be easily referenced by other programs in the system which need the trimmed signal (e.g. FLIC). SAVED uses the auto-indexing locations 0010, 0011, and 0012 and is called four

times successively by FLICKERS, once for each word of the event. It also uses locations 0047, 0050, 0051, ..., 0062 in the following manner:

<u>Loc.</u>	<u>Name</u>	<u>Use</u>
0047	TNAT	Location for storing the untrimmed T signal,
0050	TMASK	Mask for trimming the T signal (0377),
0051	TTR	Location for storing the trimmed T signal,
0052	BNAT	Location for storing the untrimmed B signal
0053	BMASK	Mask for trimming the B signal (0377),
0054	BTR	Location for storing the trimmed B signal,
0055	CNAT	Location for storing the untrimmed C signal,
0056	CMASK	Mask for trimming the C signal (1777),
0057	CTR	Location for storing the trimmed C signal,
0060	DNAT	Location for storing the untrimmed D signal,
0061	DMASK	Mask for trimming the D signal (3777),
0062	DTR	Location for storing the trimmed D signal.

The first time SAVED is called by FLICKERS, location 0010 should contain BUFFAD-1 where BUFFAD is the address in the data buffer of the T signal for the event being processed. Location 0011 should contain TRBUFP+EMARK-1 where TRBUFP is the address of the first location in the tape record buffer and EMARK is the program parameter which gives the location in the tape buffer for storing the T signal. Location 0012 should contain TNAT-1 (0046). SAVED uses location 0010 to pick up the T signal from the data buffer and location 0011 to store it away in the tape record buffer. In the process the auto-indexing feature of these two locations causes them to be incremented by 1 so that the next time SAVED is called they are set to transfer the B signal, and the next time after that they are set to transfer the C signal, and the last time they are set to transfer the D signal. Similarly, each time it is called, SAVED uses location 0012 to store the untrimmed signal, pick up the mask for trimming the signal, and store the trimmed signal and in the process the auto-indexing feature sets the location for the next signal. After SAVED has been called four times successively the event will have been transferred to the tape record buffer and locations 0051, 0054, 0057, and 0062 will contain the T, B, C, and D signals trimmed of all the flags for the event and ready to be used as such other programs in the SUPER-FLICKERS system.

BFCLR

BFCLR is a subroutine called several times by FLICKERS to clear various blocks of core storage (e.g. the data buffer). The calling sequence for BFCLR is:

```
JMS I BFCLRP
XXXX
YYYY
(CONTINUE)
```

where BFCLRP is a location containing the address of the first location of BFCLR (550), XXXX is the address of the first location in the block to be cleared, and YYYY is the number of words to clear. When it is called, BFCLR stores zeroes in the block specified in the calling sequence and then returns control to the calling program.

THE SYSPOP TAPE SYSTEM

SYSPOP is a system of 4 subroutines for reading, writing, rewinding, and spacing the magnetic tape. These 4 subroutines are called REWIND, RECSKP, WRITE, and READ. At the present time the SUPER-FLICKERS system uses only subroutine WRITE but all of them will be described here so they can be used in future data acquisition programs.

REWIND is a subroutine which can be called to rewind the tape. The calling sequence for REWIND is:

```
JMS I REWP
(CONTINUE)
```

where REWP is a location containing the address of the first location in REWIND (0600). When it is called, REWIND waits until the tape is ready, initiates the rewind, waits until the rewind is completed and then returns to the calling program.

RECSKP is a subroutine that can be called to skip over a specified number of records on the tape. It can skip either forward or backward depending on the calling sequence which is as follows:

```
CLL (STL)      /FOR SKIPPING FORWARD (BACKWARD)
JMS I RSKPP
XXXX
(CONTINUE)
```

where RSKPP is a location containing the address of the first location of RECSKP (0602), and XXXX is the number of records to skip. When it is called, RECSKP waits until the tape is ready, skips the specified number of records, stops the tape motion after the last record is skipped, and returns control to the calling program.

WRITE is a subroutine which is called to write a continuous block of storage as a record on tape. The calling sequence is:

```
JMS I WRITEP
XXXX
YYYY
(CONTINUE)
```

where WRITEP is a location containing the address of the first location of WRITE (0604), XXXX is the address of the first location to write and YYYY is the number of words to write (the length of the record to write). When it is called, WRITE waits until the tape is ready, writes the block specified in the calling sequence as a record on tape, stops the tape when the writing is finished, and checks to see if an End of Tape was detected while writing the record. If no End of Tape was encountered then it returns control to the calling program but if an End of Tape was encountered then WRITE does not return to the calling program but instead halts with 4444 in the accumulator.

READ is a subroutine which can be called to read a record from tape into a continuous block of storage. The calling sequence is:

```
JMS I READP
XXXX
YYYY
(CONTINUE)
```

where READP is a location containing the address of the first location of READ, XXXX is the address of the first location to read into, YYYY is the number of words in the record being read. When it is called, READ waits until the tape is ready, reads the record into the storage block specified in the calling sequence, checking for errors at the same time, and if it successfully reads the record without any errors it returns control to the calling program. If there is an error then READ will give one of the following error stops:

1. A stop with 7777 in the AC indicates a size error and means that the record on tape was either larger than or smaller than the number specified in the calling sequence,
2. A stop with 1111 in the AC means that the block of storage specified in the calling sequence overlapped the top of core (and among other things, this means that the RIM loader was probably destroyed),
3. A stop with 2222 in the AC indicates a parity error.

Potential users of READ may want to change some of the error stops described above so that the program tries again after failure.

All four of the subroutines described above use a subroutine called READY to wait until the tape is ready. When it is called, READY continually looks at the "Transport is Ready" bit of the Tape Status Register and when the ready condition is satisfied it returns control to the calling program.

FLIC

FLIC is a subroutine called by FLICKERS to flicker the event currently being processed or to plot one of the spectrum buffers. The present version of FLIC has 7 options corresponding to 7 different tasks which FLIC can perform. When it is called, FLIC picks up the number in the switch keys, masks it with the number 0007₈ and uses the result to determine which option it performs. If the result is 0000, FLIC simply returns to the calling program without doing anything. Options 1 through 7 are described below:

Option	Action
1	Each time it is called with option 1 specified, FLIC plots the next consecutive channel in the B+C spectrum of particles which stopped in the C detector. Since the derivative of the spectrum rather than the spectrum itself is what subroutine SPEC accumulates in the spectrum buffer, FLIC jumps off to an integrating plot routine (INTPLT) to compute and plot the channel. Since it only plots one channel each time it is called, FLIC must be called continually to maintain the plot of the spectrum.

- 2 Option 2 is a plot option just like option 1 except that it plots the B+C+D spectrum of particles which stopped in the D detector.
- 3 When option 3 is specified, FLIC flickers the event currently being processed by the main program provided Flag 2 is 1 (i.e. the particle penetrated to the D detector). It intensifies a spot on the oscilloscope screen whose X-coordinate is $(C+D)/2$ and whose Y-coordinate is $2C$. If the particle did not penetrate to the D detector, or if there is an overflow in C or D, or if either the X- or Y-coordinate exceeds 1023_{10} , FLIC returns control to the calling program without plotting anything.
- 4 For option 4 FLIC flickers the events which stop in the C detector. It intensifies a spot with X-coordinate $(B+C)$ and Y-coordinate $2B$ unless there is an overflow in B or C or unless $2B$ or $(B+C)$ exceeds 1023_{10} in which case it returns to the calling program without plotting anything. It also returns without plotting if Flag 1 is 0 (the particle did not reach the C detector), if Flag 2 is 1 (the particle penetrated to the D detector), or if the pile-up flag is on.
- 5 For option 5 FLIC flickers a spot with X-coordinate $(B+C+D)/2$ and Y-coordinate $2(B+C)$ provided the particle reached the D detector and provided there is no overflow in B, C, or D. If the particle failed to reach the D detector, or if there is an overflow in B, C, or D or if either the X- or Y-coordinate exceeds 1023_{10} , or if the pile-up flag is on, then FLIC return to the calling program without plotting anything.
- 6 Option 6 is an option for flickering the events for particles which stopped in the B detector. If both Flag 1 and Flag 2 are 0 then FLIC will intensify a point with X-coordinate $2B$ and Y-coordinate $4T$ provided there is no overflow in B or T. If this last condition is not satisfied then FLIC returns to the main program without plotting anything.

- 7 Option 7 causes FLIC to "draw" a box around the display area so that the user can calibrate the scope.

SPEC

SPEC is the subroutine which is called by FLICKERS to accumulate in the two spectrum buffers the derivatives of the two spectrums. The derivatives rather than the spectrums themselves are accumulated since a 12-bit word is not large enough to accomodate the number of counts that will be accumulated in many of the channels during the course of a run (the maximum number which can be expressed by 12 bits is 4095_{10}). In accumulating the derivative of the spectrum it is necessary to store in each channel location the difference between the number of counts in that channel and the number of counts in the preceding channel. Since this difference can be negative the numbers which are accumulated must be thought of as signed integers. With a 12-bit word this would normally allow the difference to lie anywhere between -2047_{10} and $+2047_{10}$, but since in the peaks of the spectrum the difference between two adjacent channels may exceed 2047_8 in absolute value some provision had to be made for channels which overflow in this manner. This was done by reserving one bit in the word as an overflow indicator and this left 11 bits for expressing the difference. Figure 2a is a schematic diagram of the typical channel location which has not suffered an overflow. Bits 0-10 constitute the two's complement representation of a number in the range between -1023_{10} and 1023_{10} inclusive. Bit 11 is the overflow indicator bit and is equal to 0 for a channel difference which has not suffered an overflow. To subtract one from the channel difference one simply adds 7776 and to add one to the difference one adds 0002. In both cases the overflow bit is left unchanged. If the result of such a subtraction or addition ever produces 4000 as a result then the subtraction or addition has caused an overflow and the channel location becomes a overflow location. Figure 2b illustrates such an overflow location. Bit 11 is set to one to indicate the overflow condition and bits 0-10 contain the address in the overflow buffer of the low order word of a double precision pair of words in which the overflowed channel difference is stored. The overflow buffer is located in locations 2376-2575 and any number in this range is

$$2376 \leq 2XYZ \leq 2575$$

Fig. 2. Schematic Diagram of Typical Channel Locations in the Spectrum Buffers when (a) the channel difference has not overflowed and (b) when the channel difference has overflowed.

expressible in the 11 bits allotted. When SPEC is called it accumulates the event currently being processed in the B+C+D buffer if Flag 2 is 1 and in the B+C buffer if Flag 2 is 0 and Flag 1 is 1. If both Flag 1 and Flag 2 are 0 but the pulser flag is on then SPEC accumulates the event in the B+C+D buffer. If both Flag 1 and Flag 2 are 0 and the pulser flag is 0 also then SPEC returns control to FLICKERS without accumulating the event. SPEC also returns without accumulating if the pile-up flag is on, if there is an overflow in B, C, or D or, in the case where Flag 2 is 1, if $(B+C+D)/2$ exceeds 1777_8 . To do the actual accumulating SPEC calls subroutine ADD1 which is described below.

ADD1

ADD1 is the subroutine called by SPEC to accumulate an event in one of the spectrum derivative buffers. If the event is to be accumulated in the B+C buffer then SPEC is called with the AC containing the address in the B+C buffer of the channel location corresponding to the channel being incremented. ADD1 then adds 1 to that channel location since the addition of a count in that channel will increase its difference from the preceding channel by one. ADD1 also subtracts 1 from the succeeding channel location since adding a count to any channel decreases by one the difference between the succeeding channel and the channel in question. If the event is to be accumulated in the B+C+D buffer then ADD1 is called with the AC containing the channel location in the B+C+D buffer of the channel being incremented. ADD1 calls subroutine INCR to do the actual adding and subtracting. It should be noted that each of the two buffers contains 2001_8 locations and can accomodate 2000_8 channel locations (the extra location in each case is a dummy location at the end of the buffer to give the program a location to subtract from for counts which go into the highest channel). But the quantities B+C and B+C+D can exceed 2000_8 . Therefore the sums B+C and B+C+D are divided by 2 to determine the actual channel numbers used in the spectrum buffers.

INCR

INCR is the subroutine called by ADD1 to do the actual adding to or subtracting from a given channel location. INCR is called with a

program parameter (TEMP) containing the address in the spectrum buffer of the channel location to be incremented or decremented. It is called twice by ADD1 -- the first time to add 1 to the channel location corresponding to the channel receiving the new count and the second time to subtract one from the succeeding channel location. The calling sequence for adding 1 is:

```
JMS INCR
0002
(CONTINUE),
```

and the calling sequence for subtracting 1 is:

```
JMS INCR
7776
(CONTINUE).
```

When it is called INCR checks to overflow bit of the channel location and if it is not a overflow channel adds the increment (0002 or 7776) and then if that did not cause an overflow it returns control to ADD1. If adding the increment does cause an overflow then INCR stores the overflowed result as a double precision channel location in the next 2 available words in the overflow buffer, stores the address of the low order word of that double precision pair in bits 0-10 of the original channel location, sets the overflow bit (bit 11) of the original location to 1 and returns control to ADD1. If the channel location was already an overflow location when INCR was called then INCR adds the increment to the double precision locations in the overflow buffer whose low order address is specified in bits 0-10 of the channel location and then returns control to ADD1. Users of the FLICKERS system should keep in mind that the overflow buffer contains 200_8 locations and hence can accomodate only 100_8 overflows. There is no provision in INCR for checking the number of overflows and it is conceivable that if a spectrum were especially jagged and if the experimental run were long enough then the overflows might overflow the overflow buffer and INCR might start using locations in the tape record buffer (which follows the overflow buffer in core) as overflow locations. The guiding philosophy in writing INCR was that such an occurrence was very unlikely in a normal experimental run but a dubious user might want to build some protection for the tape buffer into INCR. This might possibly be done by just keeping a count of the overflows and stopping the program when the count reaches 100_8 .

INTPLT

INTPLT is a subroutine called by FLIC to integrate the derivative of the spectrum stored in one of the spectrum buffers and to plot the spectrum thus obtained on the oscilloscope. INTPLT must be called once for each channel that it plots. Therefore it must be called continuously to maintain a plot of a spectrum. INTPLT should be called with the AC containing the beginning address of the buffer being plotted. INTPLT adds to this address a program parameter COUNT to get the address of the channel to be plotted. Each time it is called INTPLT uses COUNT to determine the X-coordinate of the plot (the channel number) and the location in the spectrum buffer of the channel location corresponding to that channel. It also increases COUNT by 1 so that the next time it is called it will plot the next channel. Whenever COUNT reaches 2000_{10} (1024_{16}) INTPLT sets COUNT back to 0 to begin the next sweep through the buffer and across the scope. As it sweeps through the buffer INTPLT maintains a double precision sum of all the channel differences that went before and hence has the contents of the channel currently being plotted. Thus any time INTPLT is called this double precision sum will contain the number of counts in the last channel plotted and INTPLT will add to this the channel difference corresponding to the next channel in the buffer. The result is just the number of counts in the channel currently under consideration and INTPLT uses this, subject to scaling by a scaling constant, as the Y-coordinate of the plot. When the end of a sweep is reached INTPLT sets the double precision sum to 0 for the next sweep.

INDPLOT

INDPLOT is an independent program which can be used to plot a spectrum when nothing else is happening. It consists simply of a loop which continually calls FLIC and jumps to COMAND to pick up any new scaling constant that might be typed in by the user.

SCOPECAL

SCOPECAL is an independent program which continually plots a diagonal line which can be used in calibrating the scope.

ORNL DWG 67-6407

Fig. 3. Abbreviated Flow Diagram of Main Loop in FLICKERS (does not show the overflow branch from ANTFRZ).

Fig. 5. Flow Diagrams for COMAND, BBIAS, and INIT.

Fig. 6. Flow Diagram for ERRCK

ORNL DWG 67-6411

NOTE: SAVED IS CALLED 4 TIMES SUCCESSIVELY FOR EACH EVENT. THE FIRST TIME IT IS CALLED, THE AUTO-INDEXING LOCATIONS 10, 11, 12 WILL HAVE CONTENTS AS FOLLOWS:

C(Loc.10) = BUFFAD-1
where BUFFAD is the address in the data buffer of the first (T) signal for the event.

C(Loc.11) = TRBUFP+EMARK-1
where TRBUFP+EMARK is the address in the tape record buffer for the signal.

C(Loc.12) = TNAT-1
where TNAT is the address on page zero of the first location in the following block (Locs. 47-62)

47, TNAT = Loc. for storing untrimmed T-signal

50, TMASK = Loc. of mask for trimming T-signalk

51, TTR = Loc. for storing trimmed T-signal

52, BNAT = etc.,

53, BMASK = etc.,

54, BTR = etc. for B-signal,

55, CNAT = and

56, CMASK = for

57, CTR = C-signal,

60, DNAT = and

61, DMASK = for

62, DTR = D-signal..

Fig. 7. Flow Diagram for SAVED

Fig. 8. Flow Diagram for BFCLR

Fig. 9. Flow Diagrams for the SYSPOP System Subroutines REWIND, RECSKP, and WRITE

Fig. 11. Flow Diagram for FLIC

Fig. 13. Flow Diagrams for SPEC and ADD1

Fig. 15. Flow Diagram for INTPLT

ORNL DWG 67-6420

Fig. 16. Flow Diagram for INDPLOT and SCOPECAL

/ FLICKERS -- WRITES TAPE, FLICKERS, AND DERIVATIVE STORAGE			
/ PERMANENT SYMBOLS			
/			
*20			
20	1000	DBUFP, 1000	/ 1ST LOC. OF DATA BUFFER (400*8 WORDS)
21	3576	DSPECP, 3576	/ 1ST LOC. OF B+C SPECTRUM FOR PARTICLES THAT STOP IN C.
22	5577	ESPECP, 5577	/ 1ST LOC. OF B+C+D SPECTRUM FOR PARTICLES STOPPING IN D.
23	2576	TRBUFP, 2576	/ 1ST LOC. OF TAPE RECORD BUFFER (1000*8 WORDS)
24	500	SAVED, 500	/ 1ST LOC. OF DATA TRANSFER ROUTINE
25	600	REWP, 600	/ 1ST LOC. OF REWIND ROUTINE
26	602	RSKPP, 602	/ 1ST LOC. OF RECORD SKIPPING ROUTINE
27	604	WRITEP, 604	/ 1ST LOC. OF RECORD WRITING ROUTINE
30	2376	OVRFLP, 2376	/ 1ST LOC. IN OVERFLOW BUFFER
31	606	READP, 606	/ 1ST LOC. OF RECORD READING ROUTINE
32	0	DUMMY1, 0000	
33	0	DUMMY2, 0000	
34	0	NEXTHI, 0000	/ ADDR. TO BE HI-ORDER WORD OF NEXT OVERFLOW CHANNEL
35	0	NEXTLO, 0000	/ ADDR. TO BE LO-ORDER WORD OF NEXT OVERFLOW CHANNEL
36	550	BFCLRP, 1550	/ 1ST LOC. IN BUFFER CLEARING ROUTINE
37	1400	PL0TP, 1400	/ 1ST LOC. IN PLOT PACKAGE
40	0	DUMMY7, 0000	
41	400	ERRCKP, 0400	/ 1ST LOC. OF ROUTINE TO CHECK FOR TAPE ERROR
42	0	BMARK, 0000	/ INDEXING PARAMETER FOR DATA BUFFER
43	2235	SCALEP, 2235	/ SCALE FACTOR FOR INTEGRATING PLOT ROUTINE
44	70	COMMAND, 70	/ 1ST LOC. OF COMMAND ROUTINE
45	2000	SPECP, 2000	/ 1ST LOC. IN SPECTRUM ROUTINE
46	2200	INTPLT, 2200	/ 1ST LOC. IN INTEGRATING PLOT ROUTINE
47	0	TNAT, 0	/ LOC. FOR UNTRIMMED T-SIGNAL
50	377	TMASK, 0377	/ MASK FOR TRIMMING T-SIGNAL
51	0	TTR, 0	/ LOC. FOR TRIMMED T-SIGNAL
52	0	BNAT, 0	/ ETC.,
53	377	BMASK, 0377	/ ETC.,
54	0	BTR, 0	/ ETC. FOR B-SIGNAL
55	0	CNAT, 0	/
56	1777	CMASK, 1777	/
57	0	CTR, 0	/ AND FOR C-SIGNAL
60	0	DNAT, 0	/
61	3777	DMASK, 3777	/
62	0	DTR, 0	/ AND FOR D-SIGNAL

SYMBOL TABLE

BFCLRP	36
BMARK	42
BMASK	53
BNAT	52
BTR	54
CMASK	56
CNAT	55
COMMAND	44
CTR	57
DBUFP	20
DMASK	61
DNAT	60
DSPECP	21
DTR	62
DUMMY1	32
DUMMY2	33
DUMMY7	40
ERRCKP	41

ESPECP	22
INIELT	46
NEXTHI	34
NEXTLM	35
OVHFLP	30
PLGTP	37
READP	31
REWP	25
RSKPP	26
SAVED	24
SCALE0	43
SPLCP	45
TMASK	50
TNAT	47
TRGUPP	23
ITR	51
WRITEP	27

- DUPLICATE TAGS
 NONE

UNDEFINED SYMBOLS
 NONE

/ COMMAND ROUTINE FOR SCALE CHANGING

/ *70

/ SCALE=43

70 00000000 COMMAND, 0000

71 0036 KRB / READ SCALE CONSTANT FROM KEYBOARD

72 7041 CIA

73 1076 1AD P276

74 3443 DCA I SCALED / DEPOSIT IN SCALE CONST. LOC.

75 5470 JMP I COMMAND

76 276 P276, 276

SYMBOL TABLE

COMMAND 70

P276 76

SCALE 43

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/INDPLOT

/THIS PROGRAM. ALLOWS INDEPENDENT PLOTTING WITH SCALE CHANGE

FLICP=37

COMAND=44

*120

120	6077	USB*3
121	4437	JMS I FLICP
122	6031	KSF
123	5125	JMP ,+2
124	4444	JMS I COMAND
125	5121	JMP ,+4

SYMBOL TABLE

COMAND	44
FLICP	37

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/SCOPECAL

/STRAIGHT LINE SCOPE CALIBRATION PROGRAM

/SET DIAGONAL LINE TO RUN FROM (1,1) TO (10,8) ON SCOPE GRATICULE

/FOR FLICKERS PICTURES

*130

130	6076	DSH+2
131	7001	IAC
132	6053	DXL
133	6063	DYL
134	6054	DIX
135	5136	JMP

*5

SYMBOL TABLE

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

*140

/BBIAS ROUTINE FOR ELIMINATING TAPE FOR T-O-F EVENTS UNDER FIXED BIAS

B=52

C=55

BTRIM=54

140 0 BBIAS, 0

141 1025 TAD C /PICK UP C FOR FLAG 2

142 7004 RAL

143 7710 SPA CLA /TEST FLAG 2

144 5154 JMP OK

145 1052 TAD B /PICK UP B FOR FLAG 1

146 7004 RAL

147 7710 SPA CLA /TEST FLAG 1

150 5154 JMP OK

151 1054 TAD BTRIM

152 1156 TAD MBIAS /MINUS BIAS IN CHANNELS

153 7700 SMA CLA

154 2140 OK, ISZ BBIAS

155 5540 JMP I BBIAS

156 0 MBIAS, 0 /SET TO ZERO FOR NO BIAS, *14(8) FOR 600 KEV

SYMBOL TABLE

B 52

BBIAS 140

BTRIM 54

C 55

MBIAS 156

OK 154

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

```

/FLICKERS
/
/ SUPER DUPER DATA ACQUISITION PROGRAM WITH POPPING TAPES
/
DRUFF=20
DSPECP=21
ESPECP=22
TRBUFP=23
SAVED=24
WRITER=27
TNAT=47
BFCLRP=36
FLICK=37
ERRCKP=41
BMARK=42
SPECP=45
COMMAND=44
ANIFR2=1736
GVRELP=38
NEXTHI=34
NEXTLO=35
HBIAS=140
/
/ INITIALIZATION PATCH
/
*0170
/
170      0      INIT,      0000
171      6032      KCC
172      6046      TLS
173      6041      TSP
174      5173      JMP      *-1
175      6077      DSB*3
176      5570      JMP      I      INIT
/
*200
/
200      7402      BEGIN,      HLT      /STOP FOR CODE WORD TO BE KEYED IN
201      7200      CLA
202      1023      TAD      TRBUFP
203      3354      DCA      ADDR2
204      1357      TAD      H0400
205      3353      DCA      COUNT
206      7604      LP,      LRS
207      3754      DCA      I      ADDR2
208      2354      ISZ      ADDR2
209      2353      ISZ      COUNT
210      5206      JMP      LP
211      1023      TAD      TRBUFP
212      3216      DCA      *-2
213      4427      JMS      I      WRITER      /WRITE THE IDENT. REC.
214      7000      GPR
215      1000      L000
216      7402      HLT
217      7200      COUNT,      CLA
218      1021      TAD      DSPECP
219      3225      DCA      *-2
220      4436      JMS      I      BFCLRP      /CLEAR THE TWO SPECTRUM BUFFERS
221      7000      GPR

```

226	4002	4002		
227	1030	TAD	OVRFLP	
230	3034	DCA	NEXTHI	/INITIALIZE OVERFLOW CHANNEL MARKERS
231	1034	TAD	NEXTHI	
232	7001	IAC		
233	3035	DCA	NEXTLO	
234	7402	HLT		/HIT CONTINUE TO TAKE DATA
235	4170	RESTRT. JMS	INIT	
236	7200	CLA		
237	1020	TAD	DBUFF	/INITIALIZE DATA BUFFER TO ALL 0000
240	3242	DCA	..+2	
241	4436	JMS	1 BFCLRP	/AC LEFT CLEARED BY BFCLRP
242	7000	OPR		
243	400	C400		
244	3042	DCA	BMARK	/INITIALIZE BMARK TO STEP THRU DATA BUFFER
245	6311	BADCLR		/CLEAR HARDWARE
246	6324	UDCLR		
247	5334	JMP	1BFCLR	
250	6712	1.3HP. HALL		/START TAKING DATA
251	7200	CLA		
252	1020	TAD	DBUFF	
253	1042	TAD	BMARK	
254	3364	DCA	BUFFAD	
255	1764	TAD	1 BUFFAD	/EXAMINE CONTENTS OF LOC, BMARK OF DATA BUFFER
256	5657	JMP	1 ..+1	/ JUMP TO THE ANTIFREEZE PATCH
257	1736	ANTFRZ		
260	7240	DATA. CLA	CMA	/USE AUTO-INDEX REGISTER 10,11,12 TO TRANSFER
261	1364	TAD	1 BUFFAD	/ THE DATA FOR THE EVENT TO THE TAPE RECORD
262	3010	DCA	10	/ BUFFER AND THE T,B,C,D WORDS ON PAGE ZERO
263	7240	CLA	CMA	/ WHICH ARE USED TO TRANSMIT THE DATA TO THE
264	1023	TAD	TRBUFF	/ SPECTRUM ACCUMULATING ROUTINE.
265	1365	TAD	EMARK	
266	3011	DCA	11	
267	1366	TAD	LISTP	
270	3012	DCA	12	
271	4424	JMS	1 SAVED	/TRANSFER T-SIGNAL
272	4424	JMS	1 SAVED	/ B-SIGNAL
273	4424	JMS	1 SAVED	/ C-SIGNAL
274	4424	JMS	1 SAVED	/ D-SIGNAL
275	4445	JMS	1 SPEC	
276	4437	JMS	1 FLICP	
277	7200	CLA		
300	3764	DCA	1 BUFFAD	
301	1042	TAD	EMARK	
302	1356	TAD	LOC04	
303	363	AND	MASK2	
304	3042	DCA	BMARK	
305	6314	BDISAR		
306	6333	UDSUB2		
307	6333	UDSUB2		
310	4140	JMS	HBIA	/FOR BIAS TEST TO CONSERVE TAPE FOR T-B-F
311	5250	JMP	LOOP	/UBIAS SKIPS THIS IF BIAS IS OK
				/**CHANGE TAD EMARK TO JMP LOOP FOR NO TAPE
312	1365	TAD	EMARK	
313	1356	TAD	LOC04	
314	3365	DCA	EMARK	
315	2353	ISZ	COUNT	/SKIP WHEN TAPE BUFFER IS FULL
316	5250	JMP	LOOP	
317	7200	WRITE. CLA		/IF TAPE BUFFER FULL WRITE A REC. ON TAPE
320	1023	TAD	TRBUFF	

```

321 3323      DCA      .+2
322 4427      JMS      I WRITEP
323 7000      GPR
324 1000      1000
325 7201      CLA      1AC      /CHECK EVERY 8TH RECORD FOR AN ERROR
326 1355      TAD      CKCNT
327 362       AND      MASK1
330 3355      DCA      CKCNT
331 1355      TAD      CKCNT
332 7450      SNA
333 4441      JMS      I ERRCKP
334 7200      TRFCLR, CLA      /CLEAR TAPE BUFFER AND INITIALIZE EMARK AND COUNT
335 1023      TAD      TRBUF / TO STEP THRU IT
336 3340      DCA      .+2
337 4436      JMS      I BFCLRP /AC LEFT CLEARED
340 7000      GPR
341 1000      1000
342 3365      DCA      EMARK
343 1361      TAD      M0200
344 3353      DCA      COUNT
345 4437      JMS      I FLICP
346 6031      KSF
347 5351      JMP
350 4444      JMS      I COMMAND
351 5250      JMP      LOOP
352 0         ADDR, 0000
353 0         COUNT, 0000
354 0         ADDR2, 0000
355 0         CKCNT, 0000
356 4         L0004, 0004
357 7400      M0400, 7400
360 2         L0002, 0002
361 7600      M0200, 7600
362 7         MASK1, 0007
363 377      MASK2, 0377
364 0         BUFFAD, 0000
365 0         EMARK, 0000
366 46      LISTP, TNAT=1
/
/
*372
/
372 5221      JMP      COUNT
373 5317      FLUSH,   JMP      WRITE
374 5250      JMP      LOOP
375 5260      JMP      DATA
376 5345      JMP      PLOT
377 5235      JMP      RESTRT
/

```

SYMBOL TABLE

```

ADDR      352
ADDR2     354
ANTFRZ    1736
BIAS      140
BEGIN     200
BFCLRP    36
EMARK     42
BUFFAD    364
CKCNT     355

```

CONAND	44
CONT	221
COUNT	353
DATA	260
DBUFP	20
DSFECF	21
EMARK	365
ERRCKP	41
ESFECF	22
FLICP	37
FLUSH	373
INIT	170
LO002	360
LO004	356
LISTP	366
LOOP	250
LP	206
M02U0	361
M04U0	357
MASK1	362
MASK2	363
NEXTHI	34
NEXTLO	35
OVRFLP	30
PLOT	345
RESTR	235
SAVED	24
SPECF	45
TBFCLR	334
INAT	47
TRUFP	23
WRITE	317
WRITEP	27
DUPLICATE TAGS	
NONE	
UNDEFINED SYMBOLS	
NONE	

ERRCK -- A ROUTINE TO CHECK FOR TAPE ERRORS

```

/
*0400
/
RSKPP=26
READP=31
TRBUFP=23
ERRCK, C
400 C JMS READY
401 4264 TAD L0510 /BACKSPACE OVER ONE RECORD
402 1256 TIFM
403 6707 TCPI
404 6732 JMP
405 5204 TSST
406 6724 JMS READY
407 4264 TAD TRBUFP
410 1023 DCA RDADD
411 3257 TAD L1000
412 1260 CMA
413 7040 DCA RDCNT
414 3261 TAD L2570 /CODE FOR READING
415 1262 TIFM
416 6707 TSDF
417 6721 JMP
420 5217 JMP RDWORD
421 5226
422 6722 RDL00P, TSSR
423 5222 JMP
424 6721 TSDF
425 5241 JMP ENDREC
426 6715 RDWORD, TSRD
427 2261 ISZ RDCNT
430 5232 JMP ,+2
431 5250 JMP ERROR /SIZE ERROR
432 7041 CIA
433 1657 TAD 1 RDADD /COMPARE WORD READ FROM TAPE WITH CORRESPONDING
434 7440 SZA / WORD IN CORE
435 5250 JMP ERROR / IF THE TWO WORDS DO NOT AGREE
436 2257 ISZ RDADD
437 5222 JMP RDL00P
440 5250 JMP ERROR /INITIAL ADDRESS ERROR
441 6734 ENDREC, TSRS
442 7500 SMA
443 5245 JMP SZCHK
444 5250 JMP ERROR /PARITY ERROR
445 2261 SZCHK, ISZ RDCNT
446 5250 JMP ERROR /SIZE ERROR
447 5600 JMP 1 ERRCK
450 7200 ERROR, CLA
451 1263 TAD L0305
452 6041 TSF
453 5252 JMP
454 6046 TLS
455 5600 JMP 1 ERRCK
/
456 510 L0510, C510
457 0 RDADD, C
460 1000 L1000, 1000
461 0 RDCNT, 0
462 2570 L2570, 2570

```

```

463 305 L0305, 0305 /ASCII CODE FOR E
464 0 READY, 0
465 7208 CLA
466 6734 TSRS
467 7086 RTL
470 7700 SMA CLA
471 5266 JMP .-3
472 5664 JMP .I READY

```

SYMBOL TABLE

```

ENDREC 441
ERRCK 400
ERROR 450
LUS02 463
L0510 456
L1000 460
L2570 462
RDADD 457
RDUNT 461
RDLEOP 422
RDWORD 426
READP 31
READY 464
RSKPP 26
SZCHK 445
TREUP 23

```

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

```

/ SAVED -- A SUBROUTINE WHICH USES THE AUTO-INDEX REGISTERS 10,11,12 TO
/          TRANSFER THE DATA FOR AN EVENT FROM DATA BUFFER TO TAPE
/          RECORD BUFFER AND TO FOUR WORDS ON PAGE ZERO (47,52,55,60)
/          USED BY THE SPECTRUM ACCUMULATING ROUTINE.
/          THE PROGRAM ALSO TRIMS THE WORDS (47,52,55,60) WITH THE
/          MASKS IN (50,53,56,61) AND STORES THE TRIMMED WORDS IN
/          LOCATIONS (51,54,57,62)

```

```

      *500
      SAVED, 0000
500      TAD I 10      /GET THE SIGNAL
501      LCA TEMP
502      TAD TEMP
503      TAD TEMP
504      LCA I 11      / STORE IT IN TAPE RECORD BUFFER
505      TAD TEMP
506      LCA I 12      / STORE IT ON PAGE ZERO
507      TAD TEMP
508      AND I 12      / MASK THE SIGNAL
509      LCA I 12      / AND STORE TRIMMED RESULT ON PAGE ZERO ALSO
510      JMP I SAVED
511
512      TEMP, 0000

```

SYMBOL TABLE

SAVED 500
TEMP 513

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

BFCLR -- A ROUTINE FOR CLEARING BUFFERS

CALLING SEQUENCE JMS I BFCLRP
 XXXX /1ST LOC IN BUFFER
 XXXX /NO. WORDS IN BUFFER
 (CONTINUE)

550

550	0	BFCLR, 0000	
551	7200	CLA	
552	1750	TAD I	BFCLR
553	3366	DCA	ADDR
554	2350	ISZ	BFCLR
555	1750	TAD I	BFCLR
556	7041	CIA	
557	3367	DCA	COUNT
560	2350	ISZ	BFCLR
561	3766	LOOP, DCA I	ADDR
562	2366	ISZ	ADDR
563	2367	ISZ	COUNT
564	5361	JMP	LOOP
565	5750	JMP I	BFCLR
566	0	ADDR, 0000	
567	0	COUNT, 0000	

NOTE -- THE AC IS LEFT CLEARED.

SYMBOL TABLE

ADDR	506
BFCLR	550
COUNT	567
LOOP	561

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

```

/SYSRMP -- A SYSTEM OF SUBROUTINES FOR POPPING THE TAPE
/
/      ALL ENTRIES GIVEN FIRST WITH JUMPS TO CORRESPONDING SUBROUTINES
/
/      THE CALLING SEQUENCE FOR EACH ENTRY IS DESCRIBED JUST BEFORE
/      SUBROUTINE CORRESPONDING TO THAT ENTRY.
/
*500
/
600      0 REWIND, 0000      /ENTRY FOR TAPE REWIND
601      5210      JMP      BGNREW
602      0 RECSKP, 0000      /ENTRY FOR SKIPPING RECORDS
603      5216      JMP      BGNSKP
604      0 WRITE, 0000      /ENTRY FOR WRITING A RECORD
605      5240      JMP      BGNWRT
606      0 READ, 0000      /ENTRY FOR READING A RECORD
607      5277      JMP      BGNRD
/
/
/REWIND -- A ROUTINE FOR REWINDING THE TAPE
/
/      CALLING SEQUENCE      JMS I REWP /REWP CONTAINS ADDR. OF REWIND ENTRY
/                               (CONTINUE) /CONTROL RETURNED HERE
/
610      4352 BGNREW, JMS      READY /WAIT UNTIL UNIT IS READY
611      1215      TAD      L0002 /LOAD COMMAND CODE FOR REWIND
612      6707      TIFM      / AND START TAPE MOVING
613      4352      JMS      READY /WAIT UNTIL REWIND COMPLETED
614      5600      JMP      I REWIND / AND THEN RETURN
615      2 L0002, 0002
/
/
/RECSKP -- A ROUTINE FOR SKIPPING RECORDS ON TAPE
/
/      CALLING SEQUENCE      CIL (STL) /FOR SKIPPING FORWARD (BACKWARD)
/                               JMS I RSKPP /RSKPP CONTAINS ADDR. OF RECSKP ENTRY
/                               XXXX /NO. OF RECORDS TO SKIP
/                               (CONTINUE) /CONTROL RETURNED HERE
/
616      4352 BGNSKP, JMS      READY /WAIT UNTIL TAPE UNIT READY
617      1602      TAD      I RECSKP
618      7041      CIA      / NEG. OF NO. RECS. TO SKIP
619      3235      DCA      R0CNT
620      2202      ISZ      RECSKP / RECSKP NOW CONTAINS RETURN ADDRESS
/
621      1236      TAD      L0530 /LOAD COMMAND CODE FOR SKIPPING
622      7430      SZL
623      1237      TAD      M0020 / FOR SKIPPING BACKWARDS
624      6707      TIFM      / AND START TAPE MOVING
625      6732 SKLOOP, TCPI /SKIP AFTER EACH RECORD
626      5227      JMP      --1
627      2235      ISZ      R0CNT / SKIP AFTER LAST REC.
628      5227      JMP      SKLOOP
629      6724      TSS1 /STOP TAPE MOTION
630      5602      JMP      I RECSKP
631      0 R0CNT, 0000
632      530 L0530, 0530

```

657 7760 M0020, 7760

/ WRITE -- A ROUTINE FOR WRITING A BLOCK OF STORAGE AS A RECORD ON TAPE

/ CALLING SEQUENCE JMS I WRITEP /WRITEP CONTAINS LOC OF WRITE ENTRY
 XXXX /ADDR. OF 1ST LOC. OF BLOCK TO WRITE
 XXXX /NO. OF WORDS TO WRITE
 (CONTINUE) /CONTROL RETURNED HERE

640 4352 BSNWRT, JMS READY
 641 1604 TAD I WRITE
 642 3274 DCA WRADD /ADDR. OF 1ST LOC IN BLOCK
 643 2204 ISZ WRITE
 644 1604 TAD I WRITE
 645 7041 CIA /NEG. OF NO. WORDS
 646 3275 DCA WRCNT
 647 4304 ISZ WRITE /RETURN ADDR. LEFT IN WRITE

650 1273 TAD L2730 /LOAD COMMAND CODE FOR WRITING
 651 6707 TIFM / AND START TAPE MOVING
 652 7200 WRLOOP, CLA
 653 1674 TAD I WRADD /PICK UP A WORD
 654 6716 TSWR / AND WRITE IT
 655 6721 TSDF / SKIP WHEN READY FOR NEXT WORD
 656 5255 JMP --1
 657 2274 ISZ WRADD / COMPUTE ADDR. OF NEXT WORD
 660 2275 ISZ WRCNT / SKIP IF LAST WORD HAS BEEN WRITTEN
 661 5252 JMP WRLOOP
 662 6724 TSST /STOP THE TAPE

/ AFTER WRITING THE RECORD THE PROGRAM CHECKS FOR THE END OF TAPE
 IF THE E. O. T. HAS BEEN ENCOUNTERED, PROG. STOPS WITH 4444 IN AC

663 7200 CLA
 664 6734 TSRS /TAPE STATUS REGISTER INTO AC(0-5)
 665 7006 RTL /END OF TAPE
 666 7006 RTL / BIT INTO AC(0)
 667 7700 SMA CLA /SKIP IF END OF TAPE
 670 5604 JMP I WRITE / OTHERWISE RETURN
 671 1276 TAD L4444
 672 7402 HLT
 673 2730 L2730, 2730
 674 0 WRADD, 0000
 675 0 WRCNT, 0000
 676 4444 L4444, 4444

/ READ -- A ROUTINE FOR READING A RECORD FROM TAPE INTO A BLOCK OF STORAGE

/ CALLING SEQUENCE JMS I READP /READP CONTAINS ADDR. OF READ ENTRY
 XXXX /ADDR OF 1ST LOC TO READ INTO
 XXXX /NO. OF WORDS IN RECORD
 (CONTINUE) /CONTROL RETURNED HERE

/ ERROR STOPS IIII IN AC MEANS INITIAL ADDR. TOO HIGH FOR RECORD TO
 FIT WITHOUT OVERFLOWING THE TOP OF CORE.

```

/
/ THE RIM LOADER WAS PROBABLY DESTROYED
/ 2222 IN AC MEANS PARITY ERROR
/ 7777 IN AC MEANS SIZE ERROR
/
677 4352 BGNRD, JMS READY
700 1606 TAD I READ
701 3350 DCA RDADD /ADDR OF 1ST LOC. TO READ INTO
702 2206 ISZ READ
703 1606 TAD I READ
704 7040 CMA
705 3351 DCA RDCNT /COMP. OF NO. WORDS IN RECORD
706 2206 ISZ READ /RETURN ADDR. NOW IN READ
/
707 1345 TAD L2570 /LOAD COMMAND CODE FOR READING
710 6707 TIFM / AND START TAPE MOVING.
711 6721 TSDF /SKIP WHEN READY TO READ FIRST WORD
712 5311 JMP *-1
713 5320 JMP RDWORD
/
714 6722 RDLRDP, TSSR /SKIP ON NEW WORD OR END OF RECORD
715 5314 JMP *-1
716 6721 TSDF / SKIP IF IT IS NEW WORD
717 5332 JMP ENDREC
720 6715 RDWORD, TSRD /READ A WORD
721 2351 ISZ RDCNT / SKIP IF NO MORE WORDS WERE EXPECTED
722 5324 JMP *-2 / (IE. REC. ON TAPE IS TOO BIG)
723 5343 JMP SZERR
724 3750 DCA I RDADD / OTHERWISE STORE THE WORD
725 2350 ISZ RDADD / AND COMPUTE ADDR. FOR NEXT ONE
726 5314 JMP RDLRDP
727 6701 TIFM=6 /EMERGENCY STOP (10T+0701) -- IF CONTROL
730 1346 TAD L1111 / REACHES THIS POINT THE RECORD HAS
731 7402 HLT / OVERFLOWED THROUGH TOP OF CORE
/
732 6734 ENDREC, TSRS /END OF RECORD HAS BEEN DETECTED.
733 7500 SMA / SKIP ON PARITY ERROR
734 5340 JMP SZCHK / OTHERWISE CHECK FOR SIZE ERROR
735 7209 CLA
736 1347 TAD L2222
737 7402 HLT
740 2351 SZCHK, ISZ RDCNT /IF NO SKIP HERE THEN PROGRAM EXPECTS MORE WORDS
741 5343 JMP SZERR / (IE. THE RECORD ON TAPE WAS TOO SHORT)
742 5606 JMP I READ
743 7240 SZERR, CLA CMA
744 7402 HLT
745 2570 L2570, 2570 /COMMAND CODE FOR READING
746 1111 L1111, 1111
747 2222 L2222, 2222
750 0 RDADD, 0000
751 0 RDCNT, 0000
/
/
/READY -- A ROUTINE TO WAIT UNTIL TAPE UNIT IS READY
/
/ THE AC IS CLEARED BEFORE RETURNING TO CALLING ROUTINE
/
752 0 READY, 0000
753 7200 BEGIN, CLA
754 6734 L90P, TSRS /TAPE STATUS REGISTER INTO AC(0-5)
755 7056 RTL / TRANSPORT IS READY BIT INTO AC(0)

```

756	7700	SMA CLA	/SKIP IF TRANSPORT IS READY
757	5354	JMP	LOOP
760	5752	JMP I	READY

SYMBOL TABLE

BEGIN	753
BGNRD	677
BGNREN	610
BGNSKP	616
BGNWRT	640
ENDREC	732
LOU2	615
LD530	636
L1111	746
L2222	747
L2570	745
L2730	673
L4444	676
LOOP	754
MOD20	637
RCCNT	635
RDADD	750
RDCNT	751
RDL00P	714
RWORD	720
READ	606
READY	752
RECSKP	602
REWIND	600
SKLOOP	627
SZCHK	740
SZERK	743
WRADD	674
WRCNT	675
WRITE	604
WRLOOP	652

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

/FLIC -- A FLICKER BOX AND PLOTTING PROGRAM

/ ASSUMES ALERTS ARE .NOT.A H OR .NOT.A B C

/ THIS PROGRAM IS CALLED ONLY ONCE PER EVENT IN OPTIONS 3 OR ABOVE,
/ BUT AS OFTEN AS POSSIBLE IN ALL OTHER OPTIONS.

/ OPTION 0 DOES NOTHING (RETURNS)

/ 1 DISPLAY B+C WITH 2000(8) F.S.

/ 2 DISPLAY B+C+D WITH 2000(8) F.S.

/ OPTION 3 FLICKERS 2C VS. (C+D)/2 WITH 2000(8) DOTS FULL SCALE
/ THROW OUT WITHOUT FLICKERING IF ANY OVERFLOW FLAGS ON OR
/ IF FLAG2 = 0 OR IF 2C .EG. 1024 OR IF (C+D)/2 .EG. 1024
/ OR IF FLAG4=1 (PILE-UP FLAG IS ON)/ 4 FLICKER 2B VS. (B+C) WITH 2000(8) DOTS F.S.
/ THROW OUT IF ANY OVERFLOWS

/ IF BC = 0

/ IF BCD = 1

/ IF 2B .GE. 2000(8)

/ IF (B+C) .GE. 2000(8)

/ ALSO THROW THE EVENT OUT IF FLAG4 = 1 (PILE UP FLAG ON)

/ NOTE OPTIONS 5 AND ABOVE ARE ON THE
/ PAGE BEGINNING AT LOC. 1600.

/ 5 FLICKER 2(B+C) VS. (B+C+D)/2

/ THROW OUT IF ANY OVERFLOWS

/ IF BCD = 0

/ IF 2(B+C) .EG. 2000

/ IF (B+C+D)/2 .EG. 2000

/ ALSO THROW THE EVENT OUT IF FLAG4 = 1 (PILE UP FLAG ON)

/ 6 FRED BERTRAND TIME-OF-FLIGHT OPTION

/ 7 DRAWS A BOX AROUND THE DISPLAY AREA FOR SCOPE CALIBRATION

/ DBUFP=20

/ BMARKP=42

/ FLAG2S=L2000

/ DSPEC=21

/ ESPEC=22

/ INTPLT=46

/ T=47

/ TTR=51

/ B=52

/ BTR=54

/ C=55

/ CTR=57

/ D=60

/ DTR=62

/ RETURN=5600

/ *1400

1400 0

FLIC, 0

1401 7604

LAS

1402 347

AND

L0007

1403 1346

TAD

L5206

/5206 IS RUST NONRELOCATABLE OCTAL FOR JMP .+3

1404 3205

DCA

.+1

1405	0	00		/JMP 606,607,....615
1406	5620	JMP	1 FLIC	/0 (RETURN)
1407	5221	JMP	0P1	/1
1410	5225	JMP	0P2	/2
1411	5231	JMP	0P3	/3
1412	5307	JMP	0P4	/4
1413	4616	JMS	1 0P5P	
1414	4617	JMS	1 0P6P	
1415	4626	JMS	1 0P7P	
/				
1416	1600	0P5P,	1600	/OPTIONS 5 AND ABOVE ARE TO BE FOUND
1417	1602	0P6P,	1602	/ ON THE PAGE BEGINNING WITH
1420	1604	0P7P,	1604	/ LOCATION 1600
/				
1421	7200	0P1,	CLA	/GO TO INTEGRATING
1422	1021	TAD	DSPEC	/PLOT FOR DELTA E PLOT
1423	4446	JMC	1 INTPLT	
1424	5600	JMP	1 FLIC	
/				
1425	7200	0P2,	CLA	/GO TO INTEGRATING
1426	1022	TAD	ESPEC	/PLOT FOR TOTAL ENERGY
1427	4446	JMS	1 INTPLT	/PLOT
1430	5600	JMP	1 FLIC	
/				
1431	7200	0P3,	CLA	/ *** 2C VS. (C+D)/2
1432	1020	TAD	DRUFP	
1433	1042	TAD	BMARKP	
1434	3302	DCA	TEMP1	
1435	1702	TAD	1 TEMP1	/PICK UP THE A-SIGNAL
1436	306	AND	L1000	/SIEVE FLAG4 ONLY
1437	7510	SPA		
1440	5600	JMP	1 FLIC	/IF FLAG4 IS ON (PILE UP)
1441	2302	ISZ	TEMP1	/IF FLAG4 NOT ON (NO PILE UP)
1442	1702	TAD	1 TEMP1	/ PICK UP FLAG2
1443	7004	RAL		
1444	7500	SMA		
1445	5600	JMP	1 FLIC	/ RETURN IF FLAG2 OFF
1446	7200	CLA		
1447	2302	ISZ	TEMP1	
1450	1702	TAD	1 TEMP1	/ PICKS UP C
1451	304	AND	L5000	/CHECK FOR OVERFLOW OR IF 2C .GE. 1024
1452	7440	SZA		
1453	5600	JMP	1 FLIC	/RETURN IF OVERFLOW OR IF 2C .GE. 1024
1454	1702	TAD	1 TEMP1	/ PICK UP C AGAIN
1455	7104	CLL	RAL	/ MULTIPLY BY 2
1456	6063	DYL		
1457	7200	CLA		
1460	1302	TAD	TEMP1	
1461	7001	IAC		
1462	3303	DCA	TEMP2	
1463	1703	TAD	1 TEMP2	/ PICK UP D
1464	305	AND	L4000	
1465	7440	SZA		
1466	5600	JMP	1 FLIC	/ RETURN IF OVERFLOW
1467	1702	TAD	1 TEMP1	/ PICK UP C
1470	1703	TAD	1 TEMP2	/ ADD D
1471	7110	CLL	RAR	/ DIVIDE SUM (C+D) BY 2
1472	3303	DCA	TEMP2	
1473	1303	TAD	TEMP2	
1474	344	AND	L6000	/CHECK IF (C+D)/2 .GE. 1024

```

1475 7440 SZA
1476 5600 JMP I FLIC /RETURN IF (C+D)/2 .GE. 1024
1477 1333 TAD TEMP2
1500 6057 DXS
1501 5600 JMP I FLIC
1502 0 TEMP1, 2000
1503 0 TEMP2, 0
1504 5000 L5000, 5000
1505 4000 L4000, 4000
1506 1000 L1000, 1000
/
1507 1047 OP4, TAD T / *** 2B VS. (B+C)
1510 351 AND SIEVE
1511 7440 SZA
1512 5600 RETURN /PULSER OR PILEUP ACCORDING TO SIEVE
1513 1052 TAD B
1514 7510 SPA
1515 5600 RETURN /B OVERFLO
1516 7104 CLL RAL
1517 7510 SPA
1520 5600 RETURN /FLAG2 = 1
1521 7004 RAL
1522 7510 SPA
1523 5600 RETURN /2B OVERFLO
1524 7010 RAR
1525 6063 DYL /2B TO Y AXIS
1526 7200 CLA
1527 1055 TAD C
1530 7510 SPA
1531 5600 RETURN /C OVERFLO
1532 7006 RTL
1533 7420 SNL
1534 5600 RETURN /FLAG1 = 0
1535 7112 CLL RTR /TRIMS FLAG1
1536 1052 TAD B
1537 6053 DXL /((B+C) TO X AXIS
1540 344 AND L6000
1541 7450 SNA /SKIP IF (B+C) .GT. 1777
1542 6054 DIX
1543 5600 RETURN
/
/ OPTIONS 5 AND ABOVE ARE ON PAGE BEGINNING AT 1600
1544 6000 L6000, 6000
1545 2000 L2000, 2000
1546 5206 L5206, 5206
1547 7 L0007, 7
1550 0 POINT, 0
1551 1000 SIEVE, 1000
1552 0 TEMP, 0
1553 1777 L1777, 1777

```

SYMBOL TABLE

B	52
UMARKP	42
BTR	54
C	55
CTR	57
D	60
DBUFP	20

DSPEC	21
DTF	62
ESPEC	22
FLAG2S	0
FLIC	1400
INTPLT	45
L0007	1547
L1000	1506
L1777	1553
L2000	1545
L4000	1505
L5000	1504
L5206	1546
L6000	1544
OP1	1421
OP2	1425
OP3	1431
OP4	1507
OP5P	1416
OP6P	1417
OP7P	1420
POINT	1550
RETURN	2600
STEVE	1551
T	47
TEMP	1552
TEMP1	1502
TEMP2	1503
TTR	51

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

```

/ OPTIONS 5,6,7 FOR FLIC
/
T=47
TTR=51
H=52
BTR=54
C=55
CTR=57
D=60
DTR=62
PL0TP=37
/
*1600
/
1600 0 0P5, 0000
1601 5206 JMP BGNBP5
1602 0 0P6, 0000
1603 5254 JMP BGNBP6
1604 0 0P7, 0000
1605 5310 JMP BGNBP7
/
1606 1047 BGNBP5, TAD T *** 2(B+C) VS. (B+C+D)/2
1607 250 AND SIEVE
1610 7640 SZA CLA
1611 5321 JMP RETURN /RETURN (PULSER OR PILEUP ACCORDING TO SIEVE)
1612 1052 TAD B
1613 7510 SPA
1614 5321 JMP RETURN /RETURN (B OVERFLOW)
1615 7106 CLL RTL
1616 7420 SNL
1617 5321 JMP RETURN /RETURN (FLAG2 = 0)
1620 7112 CLL RTR /TRIMS FLAG2
1621 3251 DCA TEMP /SAVE B
1622 1055 TAD C
1623 7510 SPA
1624 5321 JMP RETURN /RETURN (C OVERFLOW)
1625 252 AND L1777 /C TRIMMED
1626 1251 TAD TEMP
1627 7106 CLL RTL
1630 7510 SPA
1631 5321 JMP RETURN /RETURN (2(B+C) OVERFLOW)
1632 7010 RAR
1633 6063 DYL /2(B+C) TO Y AXIS
1634 7010 RAR
1635 3251 DCA TEMP /SAVE (B+C)
1636 1060 TAD D
1637 7510 SPA
1640 5321 JMP RETURN /RETURN (D OVERFLOW)
1641 1251 TAD TEMP
1642 7110 CLL RAR
1643 6053 DXL /((B+C+D)/2 TO X AXIS
1644 253 AND L6000
1645 7450 SNA /SKIP IF (B+C+D) .GT. 1777
1646 6064 DYL
1647 5321 JMP RETURN
/
1650 1050 STEVE, 1000
1651 0 TEMP, 0000

```

1652	1777	L1777,	1777	
1653	6000	L6000,	6000	
/				
1654	7200	BGNOP6,	CLA	/IF FLAG1 AND FLAG2 ARE 0 THEN FLICKER 4T VS. 2B
1655	1052		TAD	B
1656	7004		RAL	
1657	7510		SPA	/TEST FLAG2 (BCD)
1660	5321		JMP	RETURN
1661	7200		CLA	
1662	1055		TAD	C
1663	7004		RAL	
1664	7510		SPA	/TEST FLAG1 (BCD)
1665	5321		JMP	RETURN
1666	7300		CLA	CLL
1667	1047		TAD	T
1670	7510		SPA	/PICK UP T
1671	5321		JMP	RETURN
1672	305		AND	L0377
1673	7006		RTL	
1674	6063		DYL	
1675	7300		CLA	CLL
1676	1052		TAD	B
1677	7510		SPA	
1700	5321		JMP	RETURN
1701	306		AND	L0777
1702	7004		RAL	
1703	6057		DXS	
1704	5321		JMP	RETURN
1705	377	L0377,	0377	
1706	777	L0777,	0777	
1707	5321		JMP	RETURN
1710	7240	BGNOP7,	CLA	CMA
1711	6054		DIX	/MAKES A ...BOX... FOR ADJUSTING SCOPE
1712	6053		DYL	/INTENSIFIES THE BOX, ASSUMES LONG DSB
1713	6063		DYL	
1714	7200		CLA	
1715	6053		DXL	
1716	6063		DYL	
1717	5321		JMP	RETURN
1720	5321		JMP	RETURN
/				
1721	7200	RETURN,	CLA	
1722	1437		TAD	I
1723	3325		DCA	RTNP
1724	5725		JMP	I
1725	0	RTNP,	0000	

SYMBOL TABLE

B	52
BGNOP5	1606
BGNOP6	1654
BGNOP7	1710
BTR	54
C	55
CTR	57
D	60
DTR	62
L0377	1705
L0777	1706

L1777	1652
L6000	1653

OP5	1600
OP6	1602
OP7	1604
PLOTH	37
RETURN	1721
RTNP	1725

SIEVE	1650
T	47
TEMP	1651
TTK	51

DUPLICATE TAGS	
NONE	

UNDEFINED SYMBOLS	
NONE	

/ ANTIFREEZE SUBROUTINE

/ *1736

```

1736 7650 ANTFRZ, SNA CLA /SKIP IF NEW DATA HAS ARRIVED
1737 5344 JMP WAIT
1740 1360 TAD M4
1741 3365 DCA COUNT
1742 3366 DCA ROLLO
1743 5762 JMP I DATAP
1744 2366 WAIT, ISZ ROLLO
1745 5367 JMP CHECK
1746 2365 ISZ COUNT
1747 5367 JMP CHECK
1750 1360 TAD M4
1751 3365 DCA COUNT
1752 3366 DCA ROLLO
1753 1361 TAD BELL
1754 6041 TSF
1755 5354 JMP -1
1756 6046 TIS
1757 5764 JMP I RESTRP

```

```

1760 7774 M4, -4
1761 207 BELL, 0207
1762 375 DATAP, 0375
1763 376 PLOTP, 0376
1764 377 RESTRP, 0377
1765 0 COUNT, 0
1766 0 ROLLO, 0

```

```

1767 7604 CHECK, LAS /FOR OPT .GE. 3 GO BACK THRU LOOP W/O PLOTTING
1770 377 AND L0007
1771 1376 TAD M0002 / FOR OPTIONS 0.1,2 PLOT BEFORE RETURNING TO
1772 7550 SPA SNA / LOOP
1773 5763 JMP I PLOTP
1774 5775 JMP I L00PP
1775 374 L00PP, 0374
1776 7776 M0002, 7776
1777 7 L0007, 0007

```

SYMBOL TABLE

```

ANTFRZ 1736
BELL 1761
CHECK 1767
COUNT 1765
DATAP 1762
L0007 1777
L00PP 1775
M0002 1776
M4 1760
PLOTP 1763
RESTRP 1764
ROLLO 1766
WAIT 1744

```

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS
NONE

----- /SPEC=30 SPECTRUM ACCUMULATING ROUTINES

/ THIS VERSION STORES DERIVATIVES OF B+C AND B+C+D

/ THE BUFFERS ARE 2001(R) WORDS LONG

/ B+C BUFFER HAS 2 CHAN/WORD (INCREMENTED IF F1=1 AND F2=0,

I.E. IF PARTICLE STOPPED IN C)

/ B+C+D BUFFER HAS 2 CHAN/WORD (INCREMENTED IF F2=1, I.E. IF PARTICLE STOPPED IN D)

/ THERE IS NO ACCUMULATION IF OVERFLOWS

/

*2000

DSPEC=21

/DELTA E

ESPEC=22

/TOTAL ENERGY

NEXTHI=34

NEXTLO=35

T=47

TTR=51

B=52

BTK=54

C=55

CTR=57

D=60

DTR=62

RETURN=5600

2000 U

SPEC, 00

2001 1047

TAD

T

2002 274

AND

SIEVE

2003 7640

SZA CLA

2004 5600

RETURN

/PULSER OR PILEUP

2005 1052

TAD

H

2006 7510

SPA

2007 5600

RETURN

/H OVERFLO

2010 7104

CLL RAL

2011 7710

SPA CLA

2012 5235

JMP

F2ONE

2013 1055

F2ZERO,

TAD

C

2014 7004

RAL

2015 7710

SPA CLA

2016 5224

JMP

.*6

2017 1047

TAD

T

2020 7004

RAL

2021 7700

SMA CLA

2022 5600

RETURN

2023 5235

JMP

F2ONE

2024 1055

TAD

C

/*****INCREMENT B+C

2025 7510

SPA

2026 5600

RETURN

/C OVERFLO

2027 274

AND

L1777

2030 1052

TAD

H

2031 7110

CLL RAL

2032 1021

TAD

DSPEC

2033 4264

JHS

ADD1

2034 5600

RETURN

2035 1055

F2ONE,

TAD

C

/*****INCREMENT B+C+D

2036 7510

SPA

2037 5600

RETURN

/C OVERFLO

2040 274

AND

L1777

2041 3277

DCA

TEMP

2042 1060

TAD

H

2043 7510

SPA

2044	5600	RETURN		/D OVERFLOW
2045	1277	TAD	TEMP	
2046	3277	DCA	TEMP	/SAVE (C+D) TRIMMED
2047	1052	TAD	0	
2050	276	AND	L1777	
2051	1277	TAD	TEMP	
2052	7110	CLL	RAR	
2053	3277	DCA	TEMP	
2054	1277	TAD	TEMP	
2055	276	AND	L6000	
2056	7640	SZA	CLA	
2057	5600	RETURN		/B+C+D OVERFLOW
2060	1277	TAD	TEMP	
2061	1022	TAD	ESPEC	
2062	4264	JMS	ADD1	
2063	5600	RETURN		
2064	0	ADD1,	0000	
2065	3300	DCA	TEMP2	/PUT AWAY THE CHANNEL NO. (WD.)
2066	4301	JMS	INCR	/ADD 1 TO WD.
2067	2	0002		/ 0002 TELLS INCREMENT ROUTINE TO ADD 1.
2070	2300	ISZ	TEMP2	/WD=WD+1
2071	4301	JMS	INCR	/SUBTRACT 1 FROM WD+1
2072	7776	7776		/ 7776 TELLS INCREMENT ROUTINE TO SUBTRACT 1
2073	5664	JMP	1 ADD1	
2074	1000	SIEVE,	1000	
2075	6000	L6000,	6000	
2076	1777	L1777,	1777	
2077	0	TEMP,	0000	
2100	0	TEMP2,	0	
/				
/CHANNEL INCREMENTING ROUTINE				
/				
/ LOC. (TEMP2) SHOULD CONTAIN ADDR. OF CHANNEL TO BE INCREMENTED.				
/				
/ CALLING SEQUENCES				
/				
/ FOR ADDING 1 TO CHANNEL JMS INCR				
/ 0002				
/				
/ (RETURN)				
/				
/ FOR SUBTRACTING 1 JMS INCR				
/ 7776				
/ (RETURN)				
/				
2101	0	INCR,	0000	
2102	7200	CLA		
2103	1701	TAD	1 INCR	
2104	3367	DCA	LINCR	/0002 (7776) FOR ADDING (SUBTRACTING) 1.
2105	2301	ISZ	INCR	/SET UP RETURN ADDRESS
2106	1700	TAD	1 TEMP2	
2107	7110	CLL	RAR	/OVERFLOW BIT INTO LINK
2110	7430	SZL		/SKIP IF NOT ALREADY AN OVERFLOW CHANNEL
2111	5344	JMP	PREV0V	
2112	7004	RAL		
2113	1367	TAD	LINCR	/A POS. OVERFLOW (3776 P 0002 = 4000) WILL
				/ LEAVE THE LINK = 0, BUT A NEG. OVERFLOW
				/ (4002 P 7776 = 4000) WILL SET LINK = 1.

2114	3700	DCA	I	TEMP2	
2115	1700	TAD	I	TEMP2	
2116	3700	AND		L3777	
2117	7440	SZA			/SKIP IF CHAN=4000 (FOR OVERFLOW OR UNDERFLOW) / OR IF CHAN=0000
2120	5701	JMP	I	INCR	
2121	1700	TAD	I	TEMP2	
2122	3774	AND		L4000	
2123	7400	SNA			/SKIP IF CHAN WAS 4000
2124	5701	JMP	I	INCR	/ RETURN IF CHAN WAS 0000
2125	3435	DCA	I	NEXTLO	
2126	7430	SZL			/SKIP IF OVERFLOW WAS A POS. OVERFLOW
2127	7040	CMA			/ IF OVERFLOW WAS NEGATIVE
2130	3434	DCA	I	NEXTHI	(FOR POS.(NEG.) OVERFLOW STORE 0000 (7777) IN / HI-ORDER WORD OF NEXT OVERFLOW CHANNEL
2131	1035	TAD		NEXTLO	
2132	7104	CLL	HAL		/SET ADDR. OF NEW OVERFLOW CHANNEL (LO-ORDER)
2133	7001	IAC			/ AND OVERFLOW MARKER IN OLD CHANNEL
2134	3700	DCA	I	TEMP2	/ LOCATION.
2135	1034	TAD		NEXTHI	/ADVANCE THE OVERFLOW CHANNEL MARKERS
2136	1371	TAD		L0002	
2137	3034	DCA		NEXTHI	
2140	1035	TAD		NEXTLO	
2141	1371	TAD		L0002	
2142	3035	DCA		NEXTLO	
2143	5701	JMP	I	INCR	
2144	3372	PREV00,	DCA	ADDRLO	/ADDR. OF LO-ORDER WORD OF OVERFLOW CHANNEL
2145	7040	CMA			
2146	1372	TAD		ADDRLO	
2147	3373	DCA		ADDRHI	/ ADDR. OF HI-ORDER WORD
2150	1367	TAD		LINCR	/PICK UP INCREMENT (0002 OR 7776)
2151	7500	SNA			/SKIP IF IT IS NEG
2152	5357	JMP		INCRLO	
2153	7240	CLA	CMA		
2154	1773	TAD	I	ADDRHI	
2155	3773	DCA	I	ADDRHI	
2156	1367	TAD		LINCR	
2157	7100	INCRLO,	CLL		
2160	1772	TAD	I	ADDRLO	
2161	3772	DCA	I	ADDRLO	
2162	7420	SNL			/SKIP IF THERE WAS A CARRY TO HI-ORDER WORD.
2163	5701	JMP	I	INCR	
2164	2773	ISZ	I	ADDRHI	/ INCREMENT HIGH ORDER WORD
2165	7000	OPR			
2166	5701	JMP	I	INCR	
2167	0	LINCR,		0000	
2170	3777	L3777,		3777	
2171	2	L0002,		0002	
2172	0	ADDRLO,		0000	
2173	0	ADDRHI,		0000	
2174	4000	L4000,		4000	

SYMBOL TABLE

ADDRLO	2064
ADDRHI	2173
ADDRLO	2172
B	52
BTR	54

C	55
CTR	57
D	60
DSPEC	21
DTR	62
ESPEC	22
F2ONE	2035
F2ZERO	2013
INCR	2101
INCRLO	2157
L0002	2171
L1777	2076
L3777	2170
L4000	2174
L6000	2075
LINCR	2167
NEXTHI	34
NEXTLO	35
PREVGV	2144
RETURN	5600
SIEVE	2074
SPEC	2000
T	47
TEMP	2077
TEMP2	2100
ITR	51
DUPLICATE TAGS	
NONE	
UNDEFINED SYMBOLS	
NONE	

```

/ INTEGRATING PLOT ROUTINE
/ ASSUMES DERIVATIVE OF SPECTRUM IS STORED AND ORIGIN
/ OF SPECTRUM + 1 IS PASSED ALONG AS AN ARGUMENT IN THE
/ ACCUMULATOR.
/ IT PLOTS FROM LEFT TO RIGHT INTEGRATING AS IT GOES
/
SCALE=43
*2200
2200 0 INTPLT, 0000
2201 1263 TAD COUNT /PICK UP COUNT AND ADD TO ARGUMENT TO
2202 3264 DCA ADDR / GET ADDR. OF WORD TO BE INTEGRATED
2203 1664 TAD I ADDR /PERFORM INTEGRATION IN 2BLE PRECISION
2204 7110 CLL RAR /OVERFLOW BIT TO LINK
2205 7430 SZL /SKIP IF NOT AN OVERFLOW CHANNEL
2206 5252 JMP OVRFL0
2207 7004 RAL
2210 7500 SNA
2211 5216 JMP OK
2212 7240 CLA CMA
2213 1270 TAD SUMH
2214 3270 DCA SUMH
2215 1664 TAD I ADDR
2216 7100 CLL OK,
2217 1267 TAD SUML
2220 3267 DCA SUML
2221 1267 TAD SUML
2222 7421 MCL
2223 1270 TAD SUMH
2224 7430 SZL
2225 7001 IAC
2226 3270 DCA SUMH
2227 7501 MDA
2230 1272 TAD P2
2231 7650 SNA CLA /SKIP UNLESS SUML WAS 7777 (I.E. =1)
2232 5236 JMP L0ADY
2233 1270 TAD SUMH
2234 7413 SHL
2235 16 L016 /THIS SCALING CONSTANT CAN BE ALTERED BY COMAND
2236 6063 L0ADY, DYL /LOAD Y
2237 7200 CLA
2240 1263 TAD COUNT
2241 6057 DXS /LOAD X AND PLOT
2242 7001 IAC
2243 266 AND L1777
2244 7440 SZA
2245 5250 JMP .+3
2246 3267 DCA SUML
2247 3270 DCA SUMH
2250 3263 DCA COUNT
2251 5600 JMP I INTPLT
/
2252 3264 OVRFL0, DCA ADDR /ADDR. OF LO-ORDER WORD OF OVERFLOW CHANNEL
2253 7040 CMA
2254 1264 TAD ADDR
2255 3271 DCA ADDRHI /HI-ORDER WORD
2256 1671 TAD I ADDRHI
2257 1270 TAD SUMH /ADD HI-ORDER PART OF OVERFLOW CHAN. TO SUMH
2260 3270 DCA SUMH
2261 1664 TAD I ADDR

```

2262	5216	JMP	OK
2263	0	COUNT,	0
2264	0	ADDR,	0
2265	7777	MI,	7777
2266	1777	LI777,	1777
2267	0	SJML,	0
2270	0	SJMH,	0
2271	0	ADDRHI,	3000
2272	2	P2,	2

SYMBOL TABLE

ADDR	2264
ADDRHI	2271
COUNT	2263
INTPLT	2200
LI777	2266
LOADY	2236
MI	2265
OK	2216
OVRFLO	2252
P2	2272
SCALE0	43
SJMH	2270
SJML	2267

DUPLICATE TAGS

NONE

UNDEFINED SYMBOLS

NONE

7 MINUTES, 18 SECONDS.

1495 LINES.

END J98. AAO.

Internal Distribution

1-3. L. S. Abbott	22. B. W. Rust
4. C. L. Allen (CTC)	23. D. K. Trubey
5. F. E. Bertrand	24. L. W. Weston
6. A. A. Brooks (CTC)	25. G. Dessauer (consultant)
7-9. W. R. Burrus	26. B. C. Diven (consultant)
10. C. E. Clifford	27. M. H. Kalos (consultant)
11. F. E. Funderlic (CTC)	28. L. V. Spencer (consultant)
12. T. A. Love	29-30. Central Research Library
13. F. Maddon	31. Document Reference Section
14. F. C. Maienschein	32-252. Laboratory Records Department
15. E. McDaniel	253. Document Reference Section
16-20. R. W. Peelle	254. ORNL Patent Office
21. F. G. Perey	

External Distribution

- 255. P. B. Hemmig, Division of Reactor Development and Technology, U. S. Atomic Energy Commission, Washington, D. C. 20545
- 256. I. F. Zartman, Division of Reactor Development, U. S. Atomic Energy Commission, Washington, D. C. 20545
- 257-271. Division of Technical Information Extension (DTIE)
- 272. Division of Research and Development (ORO)