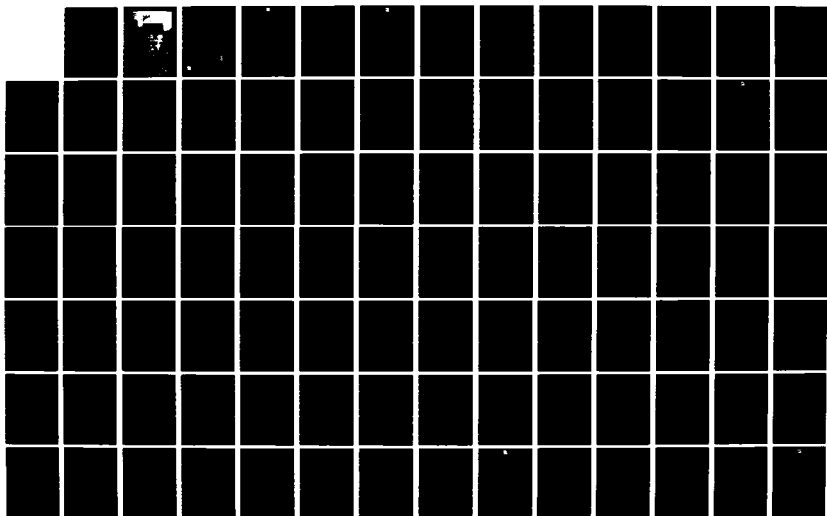
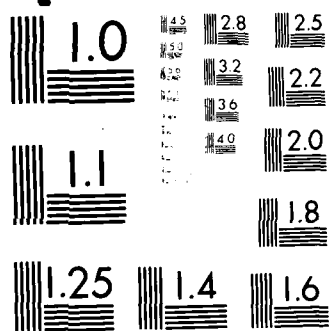


AD-A162 778 RADIATION-HARD BREADBOARD STAR TRACKER ATTACHMENT 1(U) 1/2
BALL AEROSPACE SYSTEMS DIV BOULDER CO
M W HUBBARD ET AL SEP 85 BASD/F85-03-1
UNCLASSIFIED N00014-82-6(C)-2488 F/G 9/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

85 9 19 111

AD-A162 778

DTIC FILE COPY

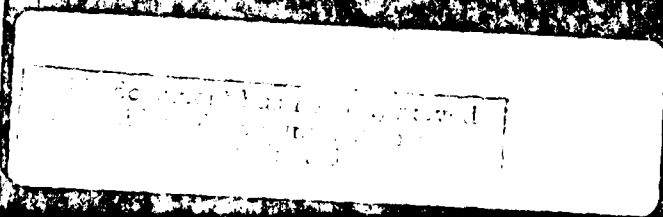


Attachment I
Final Report for

DIATION-HARD BREADBOARD STAR TRACKER

F85-03

September 1985



12

Attachment I
Final Report for

RADIATION-HARD BREADBOARD STAR TRACKER

F85-03

September 1985

Prepared for

Naval Research Laboratory
Washington, D.C.

Contract N00014-82-6(C)-2488

DTIC
ELECTE
S DEC 30 1985
A



This document has been approved
for public release and sale; its
distribution is unlimited.

BOULDER, COLORADO 80306

Section I-1



F85-03

Section I-1
STAR TRACKER CONTROL PROGRAM LISTING

PROGRAM "NEWNRL"

Written by kris Parrish
April 11, 1985

This is a new program to be used with the
NRL tracker and Phil McCollum's interface box.

```

100 DIM Clear$(2),Off_flag$(8),On_flag$(8),Disp_flag$(8),Auto_flag$(13)
110 DIM Auto_disp_flag$(13),Star_data(100,12),Read_string$(20)
120 DIM Send_msg$(35),File_string$(7),Disk_string$(14),Ctble(15)
130 DIM Err_msg$(30)
140
150 *****
160 *           Initialize variables and flags          *
170 *****
180
190 Test_flag=0                ! For use w/ the interface box
200                          ! 0 is with, 1 is without
210 Self_test_flag=0          ! Self test ON/OFF to OFF
220 Star1_flag=1              ! Star #1 enabled
230 Star2_flag=1              ! Star #2 enabled
240 Star3_flag=1              ! Star #3 enabled
250                          ! 0 is disabled...
260 Curr_star=3               ! Current star
270 X_posn=128                ! Current X position
280 Y_posn=128                ! Current Y position
290 Adapt_rate_flag=0        ! Adaptive Rate ON/OFF to OFF, 1 is ON
300 Acqu_flag=1               ! Acquisition AUTO/MANUAL to AUTO
310                          ! 0 is MANUAL, 1 is AUTO
320 Acqu_type=1               ! Acquisition type is FULL FOV,
330                          ! 0 is OFF, 1 is FULL FOV, 2 is FULL EDGE
340                          ! 3 is VECTORED EDGE
350 Num_times=2               ! Drop star criteria to 2 tries
360 Tak_dat_flag=0            ! Take data ON/OFF to OFF
370 Data_count=0              ! Used for writing star data to file
380 Max_dat=100               ! Maximum number of data sets per file
390 Max_num_files=0           ! Initialize file counter to zero
400 Max_files=5               ! Maximum number of data files
410 Nfiles=0                  ! For creating sequential data file names
420 File_string$="FILE_"      ! String used to create data file name
430 Disk_string$=":HP82901,700,1" ! Defines right disk drive for data storage
440 Hour=0                    ! Variable used to time data
450 Minute=0
460 Second=0
470 Time_flag=0
480 Inv_off=128               ! Terminal video OFF command
490 Inv_on=129                ! Terminal inverse video command
500 Prior=0                   ! Set up priority for ON KEY commands
510 Inter_face=0              ! Command to be sent to interface (0 No-op)
520 Menu_flag=0               ! Used to determine where errors come from
530 Ctrl=3                    ! Variable for control register
540 CONTROL 12.2:Ctrl         ! Define CTRLQ=0 (inverted)
550

```

```

560                                     ! CTRL1=0 (inverted)
570 Data_toggle$=CHR$(92)             ! Display character to show update interval
580 Clear$=CHR$(255)&CHR$(75)         ! Clear screen command
590                                     !
600 Auto_flag$="ON/OFF/"&CHR$(Inv_on)&"AUTO"&CHR$(Inv_off)
610 Off_flag$="ON/"&CHR$(Inv_on)&"OFF"&CHR$(Inv_off)
620 On_flag$=CHR$(Inv_on)&"ON"&CHR$(Inv_off)&"OFF"
630                                     !
640 GOSUB Set_up_table                 ! Set up command values in CTBLE
650 OFF KEY
660
670 *****
680 *      Menu1_start      *
690 *****
700
710 Set up ON KEY, and screen stuff...
720
730 Menu1_start:                      !
740 Prior=Prior+1
750 ON KEY 0 LABEL "Redisplay",Prior GOSUB Re_display
760 ON KEY 1 LABEL "Self Test",Prior GOSUB Self_test
770 ON KEY 2 LABEL "Star Disable",Prior GOSUB Star_disable
780 ON KEY 3 LABEL "Star Enable",Prior GOSUB Star_enable
790 ON KEY 4 LABEL "Adaptive Rate",Prior GOSUB Adapt_rate
800 ON KEY 5 LABEL "Acquisition",Prior GOSUB Acquisition
810 ON KEY 6 LABEL "Take Data",Prior GOSUB Take_data
820 ON KEY 7 LABEL "Menu 2 Options",Prior GOSUB Menu2_start
830 ON KEY 8 LABEL "Tracker Status",Prior GOSUB Get_status
840 ON KEY 9 LABEL "Exit",Prior GOTO Shutdown
850 PRINTER IS 1
860 CONTROL 1:1,1
870 OUTPUT 2:Clear$;                 ! Clear the terminal screen
880 GOSUB Chk_flg$men1               ! Set up initial flag conditions
890
900 *****
910 *      Menu1      *
920 *****
930
940 Menu1:                            !
950 Menu_flag=1
960 CONTROL 1:26,1
970 PRINT "TRACKER COMMAND MENU #1"
980 CONTROL 1:25,3
990 PRINT "k0 Re-display Screen"
1000 CONTROL 1:25,4
1010 PRINT "k1 Self Test"
1020 CONTROL 1:25,5
1030 PRINT "k2 Star Disable"
1040 CONTROL 1:25,6
1050 PRINT "k3 Star Enable "
1060 CONTROL 1:25,7
1070 PRINT "k4 Adaptive Rate"
1080 CONTROL 1:25,8
1090 PRINT "k5 Acquisition"
1100 CONTROL 1:25,9
1110 PRINT "k6 Take Data"
1120 CONTROL 1:25,10
1130 PRINT "k7 Menu #2 Options"
1140 CONTROL 1:25,11

```

```

150 PRINT "k8   Get Tracker Status"
160 CONTROL 1;25,12
170 PRINT "k9   Exit Program"
180 CONTROL 1;1,14
190 OUTPUT 1 USING "3(11X,10A,/)";"Star #1   ","Star #2   ","Star #3   "
200 CONTROL 1;12,18
210 PRINT "Current Star # ";Curr_star;"   X= ";X_posn;"   Y= ";Y_posn;"   Star
    Acqu tries":Num_times
220 STATUS 12,5:Tkr_status           ! Get status from tracker
230 Tkr_command=BINAND(Tkr_status,3)+10
240 IF Test_flag=1 THEN Tkr_command=12 ! If we aren't using the interface
250 SELECT Tkr_command               ! else, determine what to do now
260 CASE 10
270     GOSUB Get_data
280     GOSUB Trans_error
290 CASE 11
300     GOSUB Trans_error
310 CASE 12
320     GOSUB Get_data
330 END SELECT
340 GOSUB Chk_flg_men1               ! Check the status of all flags, etc...
350 GOTO Menu1                       ! Loop in Menu 1 options
360 !
370 !
380 !   *****
390 !   *   Menu2_start   *
400 !   *****
410 Menu2_start:                     !
420 Prior=Prior+1
430 ON KEY 0 LABEL "Redisplay",Prior GOSUB Re_display
440 ON KEY 1 LABEL "Track @ X,Y",Prior GOSUB Track_it
450 ON KEY 2 LABEL "Drop Criteria",Prior GOSUB Drop_criteria
460 ON KEY 3 LABEL "X Position",Prior GOSUB Set_x
470 ON KEY 4 LABEL "Y Position",Prior GOSUB Set_y
480 ON KEY 5 LABEL "Current Star",Prior GOSUB Set_star
490 ON KEY 6 LABEL "Take Data",Prior GOSUB Take_data
500 ON KEY 7 LABEL "Menu 1 Options",Prior GOSUB Menu1_start
510 ON KEY 8 LABEL "Tracker Status",Prior GOSUB Get_status
520 ON KEY 9 LABEL "Exit Program",Prior GOTO Shutdown
530 CONTROL 1;1,1
540 OUTPUT 2:Clear$:
550 !
560 !   *****
570 !   *   Menu2         *
580 !   *****
590 !
600 Menu2:                           !
610 Menu_flag=2
620 CONTROL 1;26,1
630 PRINT "TRACKER COMMAND MENU #2"
640 CONTROL 1;25,3
650 PRINT "k0   Re-display Screen"
660 CONTROL 1;25,4
670 PRINT "k1   Track Current Star at X, Y"
680 CONTROL 1;25,5
690 PRINT "k2   Set Drop Star Criteria   [":Num_times:"]   "
700 CONTROL 1;25,6
710 PRINT "k3   Set X Position           [":X_posn:"]       "
720 CONTROL 1;25,7
730 PRINT "k4   Set Y Position           [":Y_posn:"]       "

```

```

1740 CONTROL 1:25.8
1750 PRINT "k5 Set Star Number [";Curr_star;" "
1760 CONTROL 1:25.9
1770 PRINT "k6 Take Data"
1780 CONTROL 1:25.10
1790 PRINT "k7 Menu #1 Options"
1800 CONTROL 1:25.11
1810 PRINT "k8 Get Tracker Status"
1820 CONTROL 1:25.12
1830 PRINT "k9 Exit Program"
1840 CONTROL 1:1.14
1850 OUTPUT 1 USING "3(11X,10A./)";"Star #1 ","Star #2 ","Star #3 "
1860
1870 STATUS 12.5:Tkr_status ! Get tracker status
1880 Tkr_command=BINAND(Tkr_status.3)+10 ! Form command
1890 IF Test_flag=1 THEN Tkr_command=12 ! If we aren't using the Interface
1900 SELECT Tkr_command ! else decide what to do now
1910 CASE 10
1920 GOSUB Get_data
1930 GOSUB Trans_error
1940 CASE 11
1950 GOSUB Trans_error
1960 CASE 12
1970 GOSUB Get_data
1980 END SELECT
1990 GOSUB Check_data ! Check the condition of data taking flag
2000 GOTO Menu2 ! Loop in Menu 2 Options
2010
2020 STOP
2030 !*****
2040 !***** Program Subroutines *****
2050 !*****
2060
2070 !*****
2080 * Re_display *
2090 !*****
2100
2110 ! Redisplay the current screen
2120
2130 Re_display:
2140 IF Menu_flag=1 THEN GOSUB Menu1_start ! We were in Menu 1
2150 IF Menu_flag=2 THEN GOSUB Menu2_start ! We were in Menu 2
2160 RETURN
2170
2180 !*****
2190 * Chk_flg_men1 *
2200 !*****
2210
2220 ! This routine will display all flags
2230 ! and conditions. for variables in Menu1.
2240
2250 Chk_flg_men1: ! Check the status of...
2260 GOSUB Chk_self_tst ! Self Test Flag
2270 GOSUB Check_stars ! Individual Star Enable/Disable conditions
2280 GOSUB Check_adapt ! Adaptive Rate Flag
2290 GOSUB Check_acqu ! Acquisition Flag
2300 GOSUB Check_data ! Data taking Flag
2310 RETURN
2320
2330 !*****

```

```

2340      *      Chk_self_tst      *
2350      *****
2360
2370      This routine will check and display the
2380      STATUS of the SELF TEST FLAG.
2390
2400  Chk_self_tst:
2410      CONTROL 1:40.4      ! Position cursor at SELF_TEST_FLAG location
2420      SELECT Self_test_flag ! Now determine self test condition
2430      CASE 0              ! If Self test is OFF,
2440          Disp_flag$=Off_flag$ ! then display OFF flag
2450      CASE 1              ! If Self test is ON,
2460          Disp_flag$=On_flag$ ! then display ON flag
2470      END SELECT
2480      PRINT Disp_flag$      ! Display the flag
2490      RETURN
2500
2510      *****
2520      *      Check_stars      *
2530      *****
2540
2550      This routine will check and display
2560      the STATUS (enable/disable) OF THE STARS.
2570
2580  Check_stars:
2590      CONTROL 1:46.5      ! Position cursor for Star #1
2600      SELECT Star1_flag   ! Check Star #1 condition
2610      CASE 0              ! If Star #1 is disabled,
2620          PRINT "1 "      ! then display Star #1 as disabled
2630          CONTROL 1:46.6  ! and clear enable display for Star #1
2640          PRINT " "
2650      CASE 1              ! If Star #1 is enabled,
2660          PRINT " "        ! then clear disable display for Star #1
2670          CONTROL 1:46.6  ! and display Star #1 as enabled
2680          PRINT "1 "
2690      END SELECT          ! End of Star #1 condition check...
2700      CONTROL 1:48.5      ! Position cursor for Star #2
2710      SELECT Star2_flag   ! Check Star #2 condition
2720      CASE 0              ! If Star #2 is disabled,
2730          PRINT "2 "      ! then display Star #2 as disabled
2740          CONTROL 1:48.6  ! and clear enable display for Star #2
2750          PRINT " "
2760      CASE 1              ! If Star #2 is enabled,
2770          PRINT " "        ! then clear disable display for Star #2
2780          CONTROL 1:48.6  ! and display Star #2 as enabled
2790          PRINT "2 "
2800      END SELECT          ! End of Star #2 condition check...
2810      CONTROL 1:50.5      ! Position cursor for Star #3
2820      SELECT Star3_flag   ! Check Star #3 condition
2830      CASE 0              ! If Star #3 is disabled,
2840          PRINT "3 "      ! then display Star #3 as disabled
2850          CONTROL 1:50.6  ! and clear enable display for Star #3
2860          PRINT " "
2870      CASE 1              ! If Star #3 is enabled,
2880          PRINT " "        ! then clear disable display for Star #3
2890          CONTROL 1:50.6  ! and display Star #3 as enabled
2900          PRINT "3 "
2910      END SELECT          ! End of Star #3 condition check...
2920      RETURN              ! DONE !!
2930      !

```

```

2940 | *****
2950 | *      Check_adapt      *
2960 | *****
2970 |
2980 | This routine will check and display
2990 | the STATUS of the ADAPTIVE RATE FLAG.
3000 |
3010 | Check_adapt:
3020 | CONTROL 1:44.7          ! Position cursor for ADAPT_RATE_FLAG
3030 | SELECT Adapt_rate_flag ! Check condition of the adaptive rate flag
3040 | CASE 0                  ! If Adaptive rate is OFF,
3050 |     Disp_flag$=Off_flag$ ! then display the OFF condition
3060 | CASE 1                  ! If Adaptive rate is ON,
3070 |     Disp_flag$=On_flag$  ! then display the ON condition
3080 | END SELECT
3090 | PRINT Disp_flag$
3100 | RETURN
3110 |
3120 | *****
3130 | *      Check_acqu      *
3140 | *****
3150 |
3160 | This routine will check and display
3170 | the STATUS of the ACQUISITION FLAG.
3180 | 0 is OFF, 1 is ON, 2 is AUTO.
3190 |
3200 | Check_acqu:
3210 | CONTROL 1:42.8          ! Position cursor for ACQU_FLAG
3220 | SELECT Acqu_flag        ! Check the status of the Acquisition Flag
3230 | CASE 0                  ! If Acquisition is OFF,
3240 |     Disp_flag$="MANUAL" ! then display the OFF condition
3250 |     IF Acqu_type=0 THEN Disp_flag2$="OFF"
3260 |     IF Acqu_type=1 THEN Disp_flag2$="FULL FOV "
3270 |     IF Acqu_type=2 THEN Disp_flag2$="FULL EDGE"
3280 |     IF Acqu_type=3 THEN Disp_flag2$="VEC EDGE "
3290 | CASE 1                  ! If Acquisition is ON
3300 |     Disp_flag$="AUTO"    ! then display the ON condition
3310 |     Disp_flag2$=" "
3320 | END SELECT
3330 | PRINT Disp_flag$;" ":Disp_flag2$ ! Display the flags
3340 | RETURN
3350 |
3360 | *****
3370 | *      Check_data      *
3380 | *****
3390 |
3400 | This routine will check and display
3410 | the STATUS of the DATA TAKING FLAG.
3420 |
3430 | Check_data:
3440 | CONTROL 1:40.9          ! Position cursor for TAK_DATA_FLAG
3450 | SELECT Tak_data_flag    ! Check status of the data taking flag
3460 | CASE 0                  ! If Data taking is OFF,
3470 |     Disp_flag$=Off_flag$ ! then display the OFF condition
3480 |     PRINT Disp_flag$;" " ! Clear file number area
3490 | CASE 1                  ! If Data taking is ON
3500 |     Disp_flag$=On_flag$  ! then display the ON condition
3510 |     PRINT Disp_flag$
3520 | END SELECT
3530 | RETURN

```

```

3540
3550
3560      *      Menu1 Options      *
3570      *      *      *
3580
3590
3600      *      Self_test      *
3610      *      *      *
3620
3630      This routine will TOGGLE the SELF TEST
3640      FLAG and set up the variables needed to send
3650      commands to the tracker interface.
3660
3670 Self_test:
3680     DISABLE      ! Disable all interrupts from ON KEY commands
3690                  ! Toggle the SELF TEST FLAG
3700     Self_test_flag=1-Self_test_flag
3710     GOSUB Chk_self_tst      ! Display the NEW condition
3720
3730     Inter_face=224      ! Set up command for SELF TEST
3740                        ! and send message
3750
3760     Send_msg$="Self Test "&Disp_flag$&" Command"
3770
3780
3790     GOSUB Cmd_interface      ! Output command to interface box
3800     ENABLE      ! Enable interrupts for ON KEY commands
3810     RETURN
3820
3830
3840      *      Star_disable      *
3850      *      *      *
3860
3870      This routine will DISABLE the CURRENT STAR,
3880      and change the star flag to show such status.
3890
3900 Star_disable:
3910     DISABLE      ! Disable all interrupts from the ON KEY commands
3920                  ! Is current star ...
3930     IF Curr_star=1 THEN Star1_flag=0      ! #1 ??
3940     IF Curr_star=2 THEN Star2_flag=0      ! #2 ??
3950     IF Curr_star=3 THEN Star3_flag=0      ! #3 ??
3960     GOSUB Check_stars      ! Display NEW condition
3970
3980     Inter_face=152      ! Set up command for disable star
3990     Send_msg$="Star Disable Command"      ! and send message
4000     GOSUB Cmd_interface      ! Output command to interface box
4010     Inter_face=Ctble(Curr_star)      ! Set up command to reflect star #
4020     Send_msg$="Star #"&VAL$(Curr_star)      ! and send message
4030     GOSUB Cmd_interface      ! Output star # to interface
4040     ENABLE      ! Enable all interrupts for ON KEY commands
4050     RETURN
4060
4070
4080      *      Star_enable      *
4090      *      *      *
4100
4110      This routine will ENABLE the CURRENT STAR,
4120      and change the star flag to show such status.
4130

```

```

4140 Star_enable:
4150 DISABLE ! Disable all interrupts from the ON KEY commands
4160 ! Is current star ...
4170 IF Curr_star=1 THEN Star1_flag=1 ! #1 ??
4180 IF Curr_star=2 THEN Star2_flag=1 ! #2 ??
4190 IF Curr_star=3 THEN Star3_flag=1 ! #3 ??
4200 GOSUB Check_stars ! Display NEW condition
4210 !
4220 Inter_face=120 ! Set up command for star enable
4230 Send_msg$="Star Enable Command" ! and send message
4240 GOSUB Cmd_interface ! Output command to interface
4250 Inter_face=Ctbl(Curr_star) ! Set up command to reflect star #
4260 Send_msg$="Star #"&VAL$(Curr_star) ! and send message
4270 GOSUB Cmd_interface ! Output star # to interface
4280 ENABLE ! Enable all interrupts for ON KEY commands
4290 RETURN
4300 !
4310 ! *****
4320 ! * Adapt_rate *
4330 ! *****
4340 !
4350 ! This routine will TOGGLE the ADAPTIVE RATE FLAG,
4360 ! between ON/OFF, and also set up the correct command
4370 ! to be sent to the tracker interface.
4380 !
4390 Adapt_rate:
4400 DISABLE ! Disable all interrupts from ON KEY commands
4410 Adapt_rate_flag=1-Adapt_rate_flag ! Toggle the adaptive rate flag
4420 GOSUB Check_adapt ! Display NEW condition
4430 ! Set up send message
4440 ! and Adaptive Rate command
4450 !
4460 Send_msg$="Adaptive Rate "&Disp_flag$&" Command"
4470 Inter_face=180 !
4480 !
4490 GOSUB Cmd_interface ! Output command to interface
4500 ENABLE ! Enable all interrupts for ON KEY commands
4510 RETURN
4520 !
4530 ! *****
4540 ! * Acquisition *
4550 ! *****
4560 !
4570 ! This routine will TOGGLE the ACQUISITION FLAG
4580 ! between ON/OFF/AUTO, and also set up the correct command
4590 ! to be sent to the tracker interface. 0 is OFF, 1 is ON,
4600 ! 2 is AUTO.
4610 !
4620 Acquisition:
4630 DISABLE ! Disable all interrupts from ON KEY commands
4640 Acqu_flag=1-Acqu_flag ! Change the acquisition flag
4650 GOSUB Check_acqu ! Display the NEW condition
4660 ! Set up send message
4670 ! and command for Acquisition
4680 !
4690 IF Acqu_flag=0 THEN
4700 BEEP
4710 INPUT "Please input Acquisition type: 0=OFF, 1=FULL FOV, 2=FULL EDGE, 3=
VEC EDGE".Acqu_type
4720 IF Acqu_type>-1 AND Acqu_type<4 THEN Send_acqu

```

```

4730      BEEP
4740      DISP "INVALID acquisition type ";Acqu_type;" press <CONT> to try
again"
4750      PAUSE
4760      GOTO 4710
4770  END IF
4780  Send_acqu: !
4790  Send_msg$="Acquisition "&Auto_disp_flag$&" Command"
4800  Inter_face=204
4810  GOSUB Cmd_interface      ! Output command to interface
4820  ENABLE                  ! Enable all interrupts for ON KEY commands
4830  RETURN
4840  !
4850  ! *****
4860  !      Take_data      *
4870  ! *****
4880  !
4890  ! This routine will TOGGLE the DATA TAKING FLAG,
4900  ! between ON/OFF, no command is sent to the interface
4910  ! but some variables associated with the process
4920  ! must be set.
4930  !
4940  Take_data:
4950  DISABLE                  ! Disable all interrupts from ON KEY commands
4960  Tak_data_flag=1-Tak_data_flag      ! Toggle the data taking flag
4970  GOSUB Check_data          ! Display the NEW condition
4980  ! If we aren't taking data then DONE
4990  IF Tak_data_flag=0 THEN Take_data_end
5000  Data_count=0            ! Else re-set the data set counter
5010  Max_num_files=Nfiles+Max_files      ! Increment the max number of files
5020  IF Nfiles<=Max_num_files THEN      ! If we haven't exceeded our limit
5030  PRINTER IS 1
5040  IF Nfiles=0 THEN      ! If we haven't input a starting file #
5050  BEEP
5060  DISP
5070  INPUT "Please input initial data file # >",Nfiles
5080  !
5090  IF Nfiles>=0 AND Nfiles<=99 THEN      ! Check input for validity
5100  Max_num_files=Max_files+Nfiles
5110  ELSE
5120  BEEP
5130  DISP
5140  DISP "INVALID file # ";Nfiles;" press <CONT> to try again!"
5150  PAUSE
5160  DISP
5170  GOTO 5070      ! Input file number again...
5180  END IF
5190  END IF
5200  Take_data_end:
5210  ! Write residual data to the file
5220  IF Data_count>0 THEN GOSUB Write_data
5230  !
5240  ENABLE                  ! Enable all interrupts for the ON KEY commands
5250  RETURN
5260  !
5270  ! *****
5280  !      Menu2 Options      *
5290  ! *****
5300  !
5310  ! *****

```

```

5320 !      *      Track_it      *
5330 !      *****
5340 !
5350 !      This routine will set up variables for the
5360 !      COMMANDED TO TRACK POSITION, it will use the
5370 !      CURRENT STAR #, X POSITION, Y POSITION for
5380 !      information sent to the tracker interface.
5390 !
5400 Track_it:
5410     DISABLE ! Disable all interrupts from ON KEY commands
5420     Inter_face=84 ! Set up commanded to track position command
5430 ! and send message
5440     Send_msg$="Track at X, Y Command"
5450     GOSUB Cmd_interface ! Output Track command
5460 !
5470     Inter_face=Ctble(Curr_star) ! Set up star #
5480     Send_msg$="Star #"&VAL$(Curr_star) ! and send message
5490     GOSUB Cmd_interface ! Output star # to track to interface
5500 !
5510     Inter_face=X_posn ! Set up the X position
5520 ! and send message
5530     Send_msg$="X Position ["&VAL$(X_posn)&"]"
5540     GOSUB Cmd_interface ! Output X position to interface
5550     GOSUB Cmd_interface ! (must output this puppy twice...)
5560 !
5570     Inter_face=Y_posn ! Set up the Y position
5580 ! and send message
5590     Send_msg$="Y Position ["&VAL$(Y_posn)&"]"
5600     GOSUB Cmd_interface ! Output Y position to interface
5610     GOSUB Cmd_interface ! (must output this puppy twice)
5620 !
5630     ENABLE ! Enable all interrupts for ON KEY commands
5640     RETURN
5650 !
5660 !      *****
5670 !      *      Drop_criteria      *
5680 !      *****
5690 !
5700 !      This routine will query the user for the DROP
5710 !      CRITERIA which is currently defines as the number of times
5720 !      the tracker will try and track the star before it is dropped,
5730 !      it will also set up the correct command to be sent to
5740 !      the tracker interface.
5750 !
5760 Drop_criteria:
5770     DISABLE ! Disable all interrupts from ON KEY commands
5780     DISP ! Clear the display line
5790     BEEP
5800     INPUT "Please input the new drop criteria >".Num_times
5810     IF Num_times>0 AND Num_times<=15 THEN End_criteria
5820     BEEP
5830     DISP "INVALID drop criteria ";Num_times;" , press <CONT> to try again!"
5840     PAUSE
5850     GOTO Drop_criteria ! Let the user try again...
5860 End_criteria:
5870     Send_msg$="Drop Criteria Command" ! Set up send message
5880     Inter_face=44 ! and Drop criteria command
5890     GOSUB Cmd_interface ! Output command to interface
5900 !
5910 ! Set up send message

```

```

5920 Inter_face=Ctble(Num_times)          ! and drop criteria data
5930 Send_msg$="Drop Criteria Data  ["&VAL$(Num_times)&"]"
5940                                     !
5950 GOSUB Cmd_interface                   ! Output command to interface
5960 ENABLE                               ! Enable all interrupts for ON KEY commands
5970 RETURN
5980 !
5990 ! *****
6000 ! *      Set_X      *
6010 ! *****
6020 !
6030 ! This routine will query the user for the new X
6040 ! POSITION.
6050 !
6060 Set_x:                                !
6070 DISP                                  ! Clear the display line
6080 BEEP
6090 INPUT "Please input the new X position",X_posn
6100                                     ! Check input for validity
6110 IF X_posn>-1 AND X_posn<257 THEN End_set_x
6120 BEEP
6130 DISP "INVALID X position ";X_posn;" press <CONT> to try again!"
6140 PAUSE
6150 GOTO Set_x                           ! Let the user try again
6160 End_set_x:                            !
6170 RETURN
6180 !
6190 ! *****
6200 ! *      Set_Y      *
6210 ! *****
6220 !
6230 ! This routine will query the user for the new Y
6240 ! POSITION.
6250 !
6260 Set_y:!!
6270 DISP                                  ! Clear the display line
6280 BEEP
6290 INPUT "Please input the new Y position".Y_posn
6300                                     ! Check input for validity
6310 IF Y_posn>-1 AND Y_posn<257 THEN End_set_y
6320 BEEP
6330 DISP "INVALID Y position ";Y_posn;" press <CONT> to try again!"
6340 PAUSE
6350 GOTO Set_y                           ! Let the user try again
6360 End_set_y:                            !
6370 RETURN
6380 !
6390 ! *****
6400 ! *      Set_star   *
6410 ! *****
6420 !
6430 ! This routine will query the user for the NEW
6440 ! STAR NUMBER.
6450 !
6460 Set_star:!!
6470 DISP                                  ! Clear the display line
6480 BEEP
6490 INPUT "Please input the new star number",Curr_star
6500                                     ! Check input for validity
6510 IF Curr_star>0 AND Curr_star<4 THEN End_set_star

```

```

6520      BEEP
6530      DISP "INVALID star number ";Curr_star;" press <CONT> to try again!"
6540      PAUSE
6550      GOTO Set_star                      ! Let them try again...
6560 End_set_star:
6570      RETURN
6580      !
6590      ! *****
6600      ! *      Get_status      *
6610      ! *****
6620      !
6630      ! This routine will output the GET STATUS
6640      ! command to the interface, and will then
6650      ! check the each of the individual condition
6660      ! flags that will be recieved from the interface.
6670      !
6680 Get_status:
6690      DISABLE                      ! Disable all interrupts from ON KEY commands
6700                                ! Set up the send message
6710      Inter_face=210              ! and Get status command
6720      Send_msg$="Get Status Command"
6730      !
6740      GOSUB Cmd_interface          ! Output command to the interface
6750      IF Test_flag=0 THEN          ! If we ARE using the interface box then...
6760          Ctrl=BINAND(1,Ctrl)
6770          CONTROL 12.2;Ctrl        ! Get the status data from the interface
6780          WAIT .1
6790          ENTER 12 USING "3(#.W)":Status1,Status2,Status3
6800          ENTER 12 USING "3(#.W)":Status4,Status5,Status6
6810          ENTER 12 USING "3(#.W)":Status7,Status8,Status9
6820          IF (Status1=Status4) AND (Status4=Status7) THEN ! Make sure they are of
6830              IF STATUS1=0 THEN 6900
6840              ! IF Status1<0 THEN Status1=65536+Status1
6850              ! Status1=Status1/256          ! Convert these values to decimal
6860              CONTROL 1;22.14
6870              PRINT USING "#.10A.K": " Status= ";Status1
6880              PAUSE
6890              Star1_flag=BIT(Status1,15)
6900              Star2_flag=BIT(Status1,14)
6910              Star3_flag=BIT(Status1,13)
6920              Track1_flag=BIT(Status1,12)
6930              Track2_flag=BIT(Status1,11)
6940              Track3_flag=BIT(Status1,10)
6950              Pt1=BIT(Status1,9)
6960              Pt2=BIT(Status1,8)
6970              Acqu_flag=Pt1+Pt2
6980              Seif_test_flag=BIT(Status1,7)
6990              Pt1=BIT(Status1,4)
7000              Pt2=BIT(Status1,3)
7010              Pt3=BIT(Status1,2)
7020              Pt4=BIT(Status1,1)
7030              Num_times=(Pt1*8)+(Pt2*4)+(Pt3*2)+Pt1
7040      END IF
7050      ENABLE                      ! Enable all interrupts for the ON KEY commands
7060      RETURN
7070      !
7080      !
7090      ! *****
7100      ! *      Cmd_interface    *

```

```

7110 ! *****
7120 !
7130 ! This routine will output a specific command
7140 ! stored in variable INTER_FACE, to the interface box.
7150 ! while it is "waiting" for the acknowledge signal
7160 ! it will concurrently collect data from the tracker.
7170 !
7180 Cmd_interface:
7190   Menu_flag=3
7200   Ctrl=2           ! Set dir=OUT CTL1=1
7210   CONTROL 12.2:Ctrl ! Command ready CTLO=1
7220   WAIT .1         ! Wait for uP to recover from reset
7230   Ctrl=BINIOR(1,Ctrl) ! CTLO pulsed down, set low when in normal operation
7240   CONTROL 12.2:Ctrl
7250   DISP
7260   DISP "Sending ";Send_msg$;" to tracker..."
7270 ! DISP "Sending ";Send_msg$;" to tracker...PRESS STEP, CONT,ETC"
7280 ! PAUSE
7290   IF Test_flag=1 THEN
7300     WAIT .5
7310     DISP
7320     RETURN
7330   END IF
7340   OUTPUT 12 USING "#,B";Inter_face ! Output command to interface
7350   Time_out=(TIMEDATE) MOD 86400      ! Must wait for 7 seconds for
7360   Time_out=Time_out+7                ! interface, collect data while
7370   Time_out1=Time_out+1 MOD 60        ! waiting...
7380   Time_flag=0                       ! It isn't time yet...
7390 Do_it_again:
7400   Time_yet=TIMEDATE MOD 86400       ! Check current time
7410   IF (Time_out>Time_out1) THEN
7420     IF (Time_yet>Time_out) OR (Time_yet<Time_out1) THEN Time_flag=1
7430   ELSE
7440     IF (Time_yet>Time_out) AND (Time_yet<Time_out1) THEN Time_flag=1
7450   END IF
7460 Check_tracker:
7470 ! DISP "Time_yet";Time_yet;" Time_out ";Time_out;" Time_out1 ";Time_out1;"
   Time_flag";Time_flag ! FOR DEBUGGING PURPOSES
7480   WAIT 1.0
7490   STATUS 12.5:Tkr_status
7500   Tkr_command=BINAND(Tkr_status,3)+10
7510   SELECT Tkr_command
7520   CASE 10
7530     GOSUB Get_data
7540     Ctrl=BINAND(2,Ctrl)
7550     CONTROL 12.2:Ctrl
7560     Ctrl=BINIOR(1,Ctrl)
7570     CONTROL 12.2:Ctrl
7580     GOSUB Trans_error
7590   CASE 11
7600     GOSUB Trans_error
7610   CASE 12
7620     GOSUB Get_data
7630   END SELECT
7640   IF Time_flag=2 THEN
7650     DISP
7660     RETURN
7670   END IF
7680   IF Time_flag=0 THEN GOTO Do_it_again ! Not time yet...
7690   IF Time_flag=1 THEN Time_flag=2    ! Check status one more time...

```

```

7700 GOTO Check_tracker
7710 !
7720 *****
7730 *      Get_data      *
7740 *****
7750 !
7760 This routine will RECIEVE and DISPLAY
7770 STAR DATA from the interface box.
7780 !
7790 Get_data: !
7800 DISABLE ! Disable all interrupts from the ON KEY commands
7810 IF Test_flag=0 THEN ! If we ARE using the interface box then...
7820 Ctrl=BINAND(1,Ctrl) ! Get star data from the tracker
7830 CONTROL 12,2:Ctrl
7840 ENTER 12 USING "3( #,W)";Star1_x,Star1_y,Star1_m
7850 ENTER 12 USING "3( #,W)";Star2_x,Star2_y,Star2_m
7860 ENTER 12 USING "3( #,W)";Star3_x,Star3_y,Star3_m
7870 !
7880 ! Now check these numbers for negative values
7890 !
7900 IF Star1_x<0 THEN Star1_x=65536+Star1_x
7910 IF Star1_y<0 THEN Star1_y=65536+Star1_y
7920 IF Star1_m<0 THEN Star1_m=65536+Star1_m
7930 IF Star2_x<0 THEN Star2_x=65536+Star2_x
7940 IF Star2_y<0 THEN Star2_y=65536+Star2_y
7950 IF Star2_m<0 THEN Star2_m=65536+Star2_m
7960 IF Star3_x<0 THEN Star3_x=65536+Star3_x
7970 IF Star3_y<0 THEN Star3_y=65536+Star3_y
7980 IF Star3_m<0 THEN Star3_m=65536+Star3_m
7990 !
8000 Star1_x=Star1_x/256 ! Convert these values to decimal
8010 Star1_y=Star1_y/256
8020 Star1_m=Star1_m/256
8030 Star2_x=Star2_x/256
8040 Star2_y=Star2_y/256
8050 Star2_m=Star2_m/256
8060 Star3_x=Star3_x/256
8070 Star3_y=Star3_y/256
8080 Star3_m=Star3_m/256
8090 ELSE
8100 Star1_x=11+Data_count ! NOT using interface so set up some
8110 Star1_y=12+Data_count ! bogus numbers for star values
8120 Star1_m=13+Data_count
8130 Star2_x=14+Data_count
8140 Star2_y=15+Data_count
8150 Star2_m=16+Data_count
8160 Star3_x=17+Data_count
8170 Star3_y=18+Data_count
8180 Star3_m=19+Data_count
8190 END IF
8200 ! Calculate time variables
8210 ! to be written to the data file
8220 Datat=(3600*Hour+60*Minute+Second-TIMEDATE)+ABS(Datat)
8230 Data_time=(TIMEDATE+Datat) MOD 86400
8240 Data_time=INT(Data_time*100)/100
8250 Hour=Data_time DIV 3600
8260 Minute=Data_time MOD 3600 DIV 60
8270 Second=Data_time MOD 60
8280 CONTROL 1:1.10 ! Display the update flag
8290 PRINT USING "#.1A";Data_toggle$

```

```

8300 Data_toggle$=CHR$(139-NUM(Data_toggle$))
8310
8320 ! Display the star values being recieved
8330 ! from the tracker...
8340 CONTROL 1;22,14
8350 PRINT USING "#,4A,4D.4D": " X=";Star1_x
8360 PRINT USING "#,4A,4D.4D": " Y=";Star1_y
8370 PRINT USING "#,4A,4D.4D": " M=";Star1_m
8380 CONTROL 1;22,15
8390 PRINT USING "#,4A,4D.4D": " X=";Star2_x
8400 PRINT USING "#,4A,4D.4D": " Y=";Star2_y
8410 PRINT USING "#,4A,4D.4D": " M=";Star2_m
8420 CONTROL 1;22,16
8430 PRINT USING "#,4A,4D.4D": " X=";Star3_x
8440 PRINT USING "#,4A,4D.4D": " Y=";Star3_y
8450 PRINT USING "#,4A,4D.4D": " M=";Star3_m
8460
8470 IF Tak_data_flag=0 THEN End_get_data ! If NOT taking data then DONE
8480 Data_count=Data_count+1 ! ELSE,
8490 IF Data_count<=Max_dat THEN
8500 Star_data(Data_count,1)=Star1_x ! Load star data into array
8510 Star_data(Data_count,2)=Star1_y ! for writing data to file...
8520 Star_data(Data_count,3)=Star1_m
8530 Star_data(Data_count,4)=Star2_x
8540 Star_data(Data_count,5)=Star2_y
8550 Star_data(Data_count,6)=Star2_m
8560 Star_data(Data_count,7)=Star3_x
8570 Star_data(Data_count,8)=Star3_y
8580 Star_data(Data_count,9)=Star3_m
8590 Star_data(Data_count,10)=Hour
8600 Star_data(Data_count,11)=Minute
8610 Star_data(Data_count,12)=Second
8620 CONTROL 1;48,9
8630 IF Data_count>0 THEN PRINT Data_count ! Display data count
8640 IF Data_count=0 THEN PRINT " " ! Else clear count area
8650 END IF
8660 END IF
8670 IF Data_count=Max_dat THEN GOSUB Write_data ! If array is full
8680 ! then write it out
8690 End_get_data:
8700 ENABLE ! Enable all interrupts for the ON KEY commands
8710 RETURN
8720
8730 *****
8740 * Write_data *
8750 *****
8760
8770 ! This routine will write data to the output file.
8780
8790 Write_data:
8800 ON ERROR GOTO Error_tst ! Trap errors that occur when writing data
8810 Number$=VAL$(Nfiles)
8820 File_string$[6]=Number$[1] ! Create a file name to be used
8830 DISP ! Clear display line
8840 ! Let the user know what file we are using
8850 DISP "Writing data to >";File_string$
8860 ! Designate the right disk drive for data
8870 MASS STORAGE IS ":HP82901,700,1"
8880 CREATE BDAT File_string$,Max_dat ! Open the data file
8890 ASSIGN @Path_1 TO File_string$
8900 OUTPUT @Path_1:Star_data(*) ! Write data to the file

```

```

8900 ASSIGN @Path_1 TO >          ! Close the data file
8910 Nfiles=Nfiles+1              ! Increment file name variable
8920 Data_count=0                 ! Initialize data set counter for another round !!
8930 OFF ERROR
8940 DISP                          ! Clear the display line
8950 RETURN
8960 !
8970 ! *****
8980 !      Error_tst      *
8990 ! *****
9000 !
9010 ! This routine will trap any error's encountered
9020 ! in the data taking process.
9030 !
9040 Error_tst:                    !
9050 BEEP                          ! Beep 'em ...
9060 OFF ERROR
9070 IF ERRN=54 THEN              ! Duplicate file name error code
9080 DISP
9090 DISP "This file already exists, press <CONT> to try again!"
9100 PAUSE
9110 DISP
9120 INPUT "Please input initial file # >",Nfiles  ! try again...
9130                                ! Check input for validity
9140 IF Nfiles>0 AND Nfiles<99 THEN
9150     Max_num_files=Max_files+Nfiles
9160     GOTO Write_data          ! Input was OK, now write the data
9170 END IF
9180 ELSE
9190 IF ERRN=64 THEN              ! Mass storage overflow error code
9200 DISP
9210 DISP "The data disc is full, Please exchange it with a new one..."
9220 WAIT .5
9230 DISP "Press <CONT> after replacing the disc to continue!"
9240 PAUSE
9250 DISP
9260 GOTO Write_data            ! Hopefully the data disc was replaced correctly
9270 ELSE
9280 IF ERRN=80 THEN              ! Disc NOT changed or not IN right drive...
9290 DISP
9300 DISP "There is not a correct disc in the right disk drive..."
9310 WAIT .5
9320 DISP "Press <CONT> after placing disc in the right hand drive"
9330 PAUSE
9340 DISP
9350 GOTO Write_data
9360 ELSE
9370                                ! There was a different error, must look up and correct
9380                                ! the problem before continuing...
9390 DISP
9400 DISP "Unexpected error (":ERRN:") press <CONT> after correcting pr
oblem"
9410 PAUSE
9420 DISP
9430 END IF
9440 END IF
9450 END IF
9460 RETURN
9470 !
9480 ! *****

```

```

9490 ! *      Trans_error      *
9500 ! *****
9510 !
9520 ! This routine will display a message, and beep at you
9530 ! if there was a transmit error. By using the Menu_flag
9540 ! to determine where the error came from, a message will
9550 ! be displayed.
9560 !
9570 Trans_error: !
9580 IF Menu_flag=0 THEN Err_msg$="" ""
9590 IF Menu_flag=1 THEN Err_msg$=" Menu1 Xmit Error" ""
9600 IF Menu_flag=2 THEN Err_msg$=" Menu2 Xmit Error" ""
9610 IF Menu_flag=3 THEN Err_msg$=" Command Interface Xmit Error"
9620 DISP
9630 DISP Err_msg$ ! Let the user know where the error came from...
9640 !
9650 Ctrl=BINAND(2,Ctrl)
9660 CTRLDL 12.2:Ctrl
9670 Ctrl=BINIOR(1,Ctrl)
9680 CTRLDL 12.2:Ctrl
9690 Freq1=4000
9700 Freq2=1000
9710 Bnumber=2
9720 Btime=.2
9730 Dfreq=(Freq2-Freq1)/Bnumber-1
9740 FOR F=0 TO Bnumber-1
9750     Bfreq=Freq1+(Dfreq*F)
9760     BEEP Bfreq,Btime ! Beep 'em...
9770     Btime=Btime*5
9780 NEXT F
9790 DISP
9800 RETURN
9810 !
9820 ! *****
9830 ! *      Set_up_table      *
9840 ! *****
9850 !
9860 ! This routine will set up values for
9870 ! a command table.
9880 !
9890 Set_up_table: !
9900 Ctbl(1)=224 ! Self test ON/OFF
9910 Ctbl(2)=152 ! Disable current star
9920 Ctbl(3)=120 ! Enable current star
9930 Ctbl(4)=84 ! Track current star @ X, Y
9940 Ctbl(5)=180 ! Adaptive Rate ON/OFF
9950 Ctbl(6)=204 ! Acquisition ON/OFF/AUTO
9960 Ctbl(7)=44 ! Set drop star criteria
9970 Ctbl(8)=210 ! Get Tracker status
9980 Ctbl(9)=50 ! UNDEFINED as of yet
9990 Ctbl(10)=74 ! UNDEFINED as of yet
10000 Ctbl(11)=172 ! UNDEFINED as of yet
10010 Ctbl(12)=136 ! UNDEFINED as of yet
10020 Ctbl(13)=102 ! UNDEFINED as of yet
10030 Ctbl(14)=30 ! UNDEFINED as of yet
10040 Ctbl(15)=254 ! UNDEFINED as of yet
10050 RETURN
10060 !
10070 ! *****
10080 ! *      No_op      *

```

```

0090 ! *****
0100 !
0110 ! This routine will be used as a no operation type
0120 ! subroutine, if the user inputs an invalid option, they
0130 ! will be kicked into here, and be advised of the input error.
0140 !
0150 No_op: !
0160 BEEP
0170 Err_msg$=" "
0180 IF Menu_flag=1 THEN Err_msg$="from Menu #1 "
0190 IF Menu_flag=2 THEN Err_msg$="from Menu #2 "
0200 DISP
0210 DISP "INVALID COMMAND ENTRY..."
0220 WAIT .5
0230 DISP Err_msg$
0240 WAIT .5
0250 DISP
0260 RETURN
0270 !
0280 ! *****
0290 ! * Shutdown *
0300 ! *****
0310 !
0320 ! This is the routine which allows the user
0330 ! to EXIT from the program.
0340 !
0350 Shutdown: !
0360 OFF KEY
0370 PRINTER IS 1
0380 OUTPUT 2:Clear$;
0390 PRINT
0400 PRINT "PROGRAM TERMINATED..."
0410 END

```

Section I-2



F85-Q3

Section I-2
FLOWCHART OF THE STAR TRACKER CONTROL PROGRAM

✓

v

k1
GOSUB Self_test
(PAGE 17)

.

k2
GOSUB Star_disable
(PAGE 18)

.

k3
GOSUB Star_enable
(PAGE 20)

.

k4
GOSUB Adapt_rate
(PAGE 23)

.

k5
GOSUB Acquisition
(PAGE 24)

.

.

.

.

.

.

.

v

v

```
+-----+
| k6     |
+-----+
| GOSUB  |
| Take_data |
+-----+
| (PAGE 26) |
+-----+
```

.

```
+-----+
| k7     |
+-----+
| GOSUB  |
| Menu2_start |
+-----+
| (PAGE 5) |
+-----+
```

.

```
+-----+
| k8     |
+-----+
| GOSUB  |
| Get_status |
+-----+
| (PAGE 36) |
+-----+
```

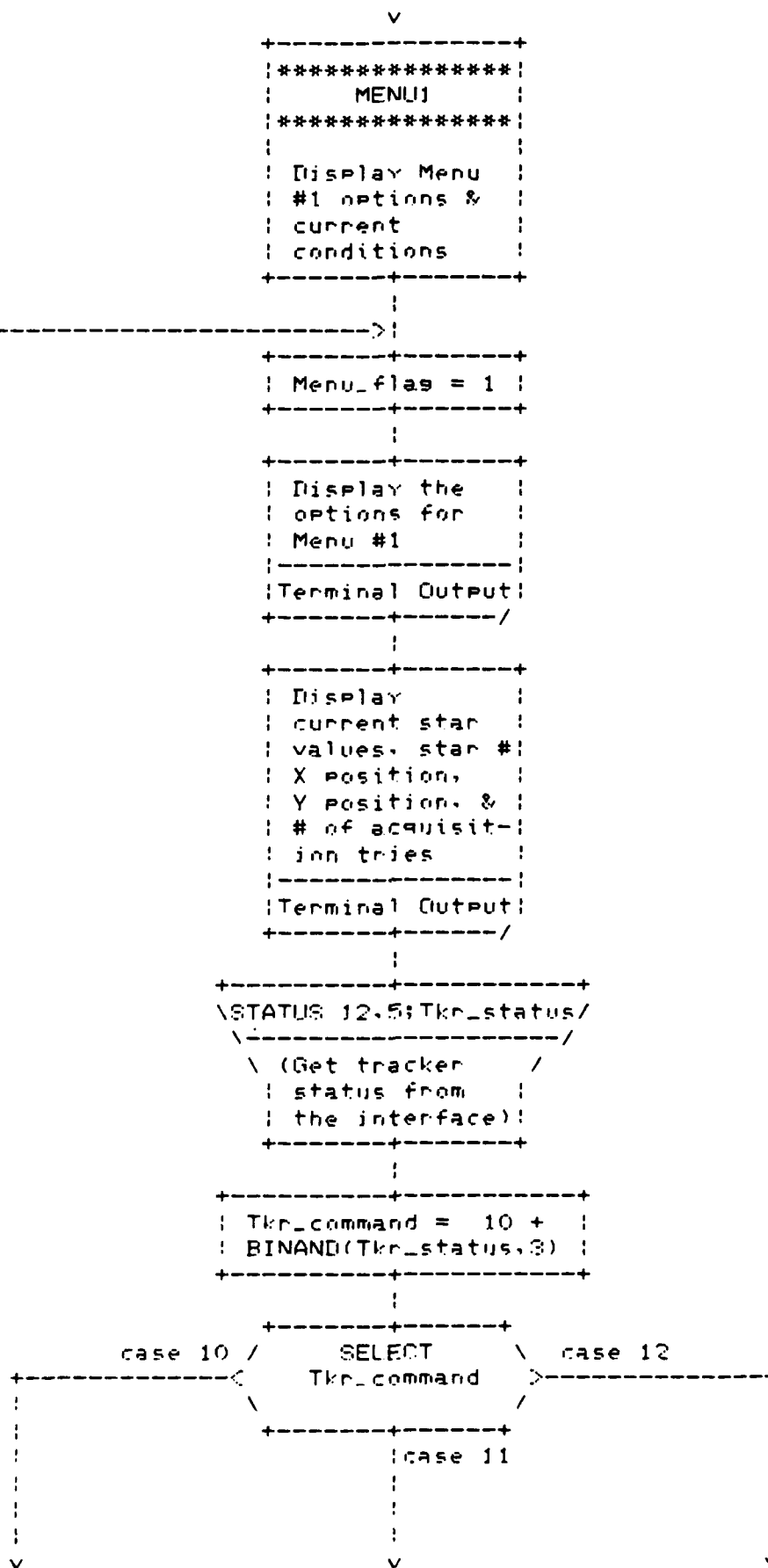
.

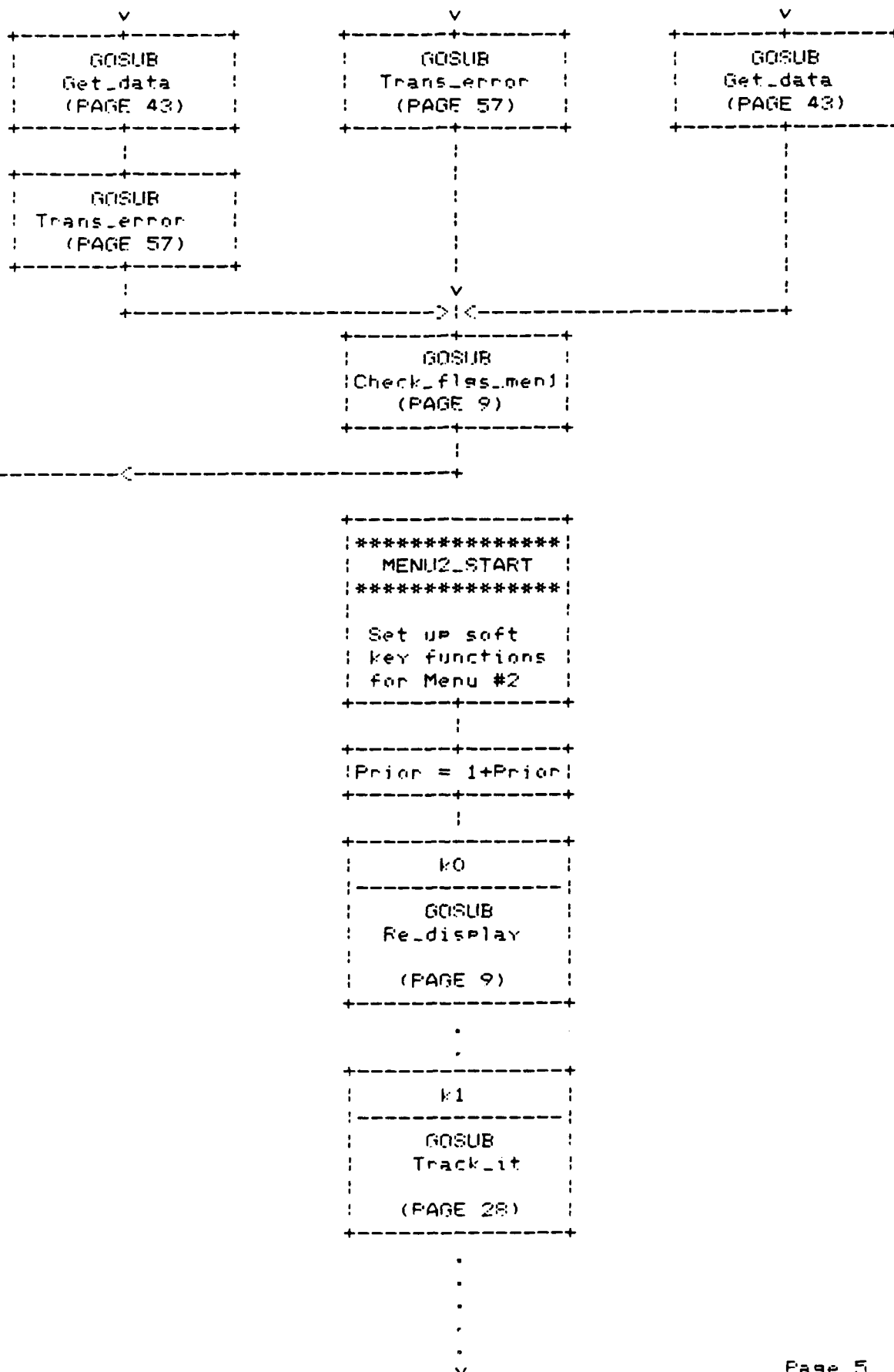
```
+-----+
| k9     |
+-----+
| GOTO   |
| Shutdown |
+-----+
| (PAGE 64) |
+-----+
```

```
+-----+
| Clear the |
| screen    |
+-----+
| Terminal Output |
+-----+
| /          |
+-----+
```

```
+-----+
| GOSUB   |
| Check_flash_men |
+-----+
| (PAGE 9) |
+-----+
```

v





v	
k2	
GOSUB Drop_criteria	
(PAGE 31)	
:	
k3	
GOSUB Set_Lx	
(PAGE 34)	
:	
k4	
GOSUB Set_Ly	
(PAGE 35)	
:	
k5	
GOSUB Set_star	
(PAGE 30)	
:	
k6	
GOSUB Take_data	
(PAGE 26)	
:	
k7	
GOSUB Menu1_start	
(PAGE 1)	
v	

v

```
+-----+
| k8     |
+-----+
| GOSUB  |
| Get_status |
+-----+
| (PAGE 36) |
+-----+
```

.

```
+-----+
| k9     |
+-----+
| GOTO   |
| Shutdown |
+-----+
| (PAGE 64) |
+-----+
```

|

```
+-----+
| Clear the |
| screen    |
+-----+
| Terminal Output |
+-----+
| /          |
+-----+
```

|

```
+-----+
| ***** |
| MENU2    |
| ***** |
+-----+
| Display Menu |
| #2 options & |
| current      |
| conditions   |
+-----+
```

|

```
+-----+
| > |
+-----+
| Menu_flag = 2 |
+-----+
```

|

```
+-----+
| Display the |
| options for |
| Menu #2     |
+-----+
| Terminal Output |
+-----+
| /          |
+-----+
```

|

v

```

V
+-----+
| Display |
| current star |
| values, star # |
| X position, |
| Y position, & |
| # of acquisit- |
| ion tries |
+-----+
| Terminal Output |
+-----+

```

```

+-----+
| \STATUS 12,5:Tkr_status/ |
| \-----/ |
| \ (Get tracker / |
| | status from | |
| | the interface) |
+-----+

```

```

+-----+
| Tkr_command = 10 + |
| BINAND(Tkr_status,3) |
+-----+

```

```

+-----+
| case 10 / SELECT \ case 12 |
| Tkr_command |
+-----+

```

```

+-----+
| case 11 |
+-----+

```

```

+-----+
| GOSUB |
| Get_data |
| (PAGE 43) |
+-----+

```

```

+-----+
| GOSUB |
| Trans_Lenon |
| (PAGE 57) |
+-----+

```

```

+-----+
| GOSUB |
| Get_data |
| (PAGE 43) |
+-----+

```

```

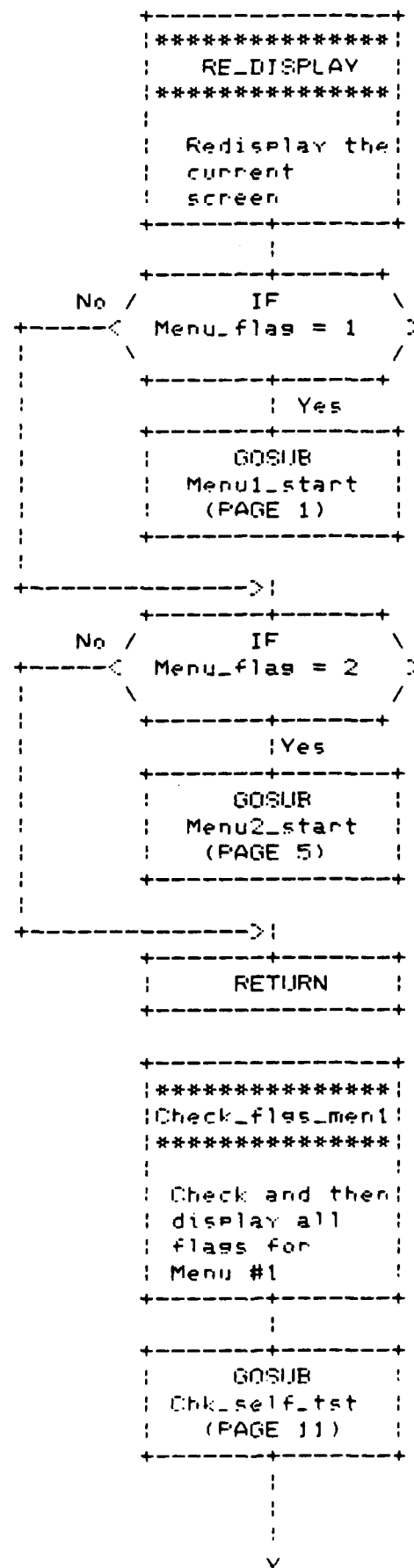
+-----+
| GOSUB |
| Trans_Lenon |
| (PAGE 57) |
+-----+

```

```

+-----+
| GOSUB |
| Check_data |
| (PAGE 15) |
+-----+

```



v

```
+-----+
| GOSUB |
| Check_start |
| (PAGE 11) |
+-----+
```

|

```
+-----+
| GOSUB |
| Check_adapt |
| (PAGE 14) |
+-----+
```

|

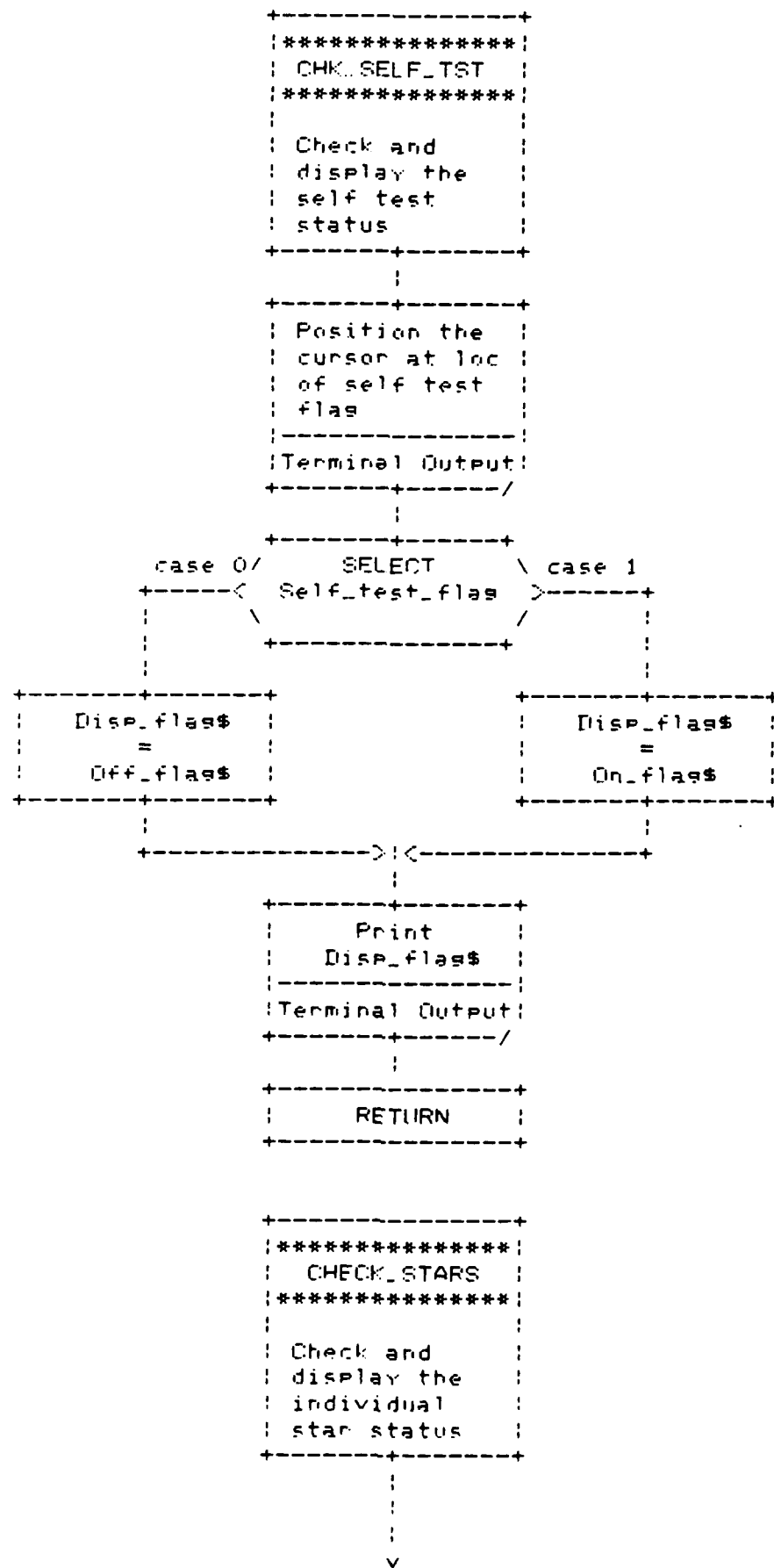
```
+-----+
| GOSUB |
| Check_acqu |
| (PAGE 16) |
+-----+
```

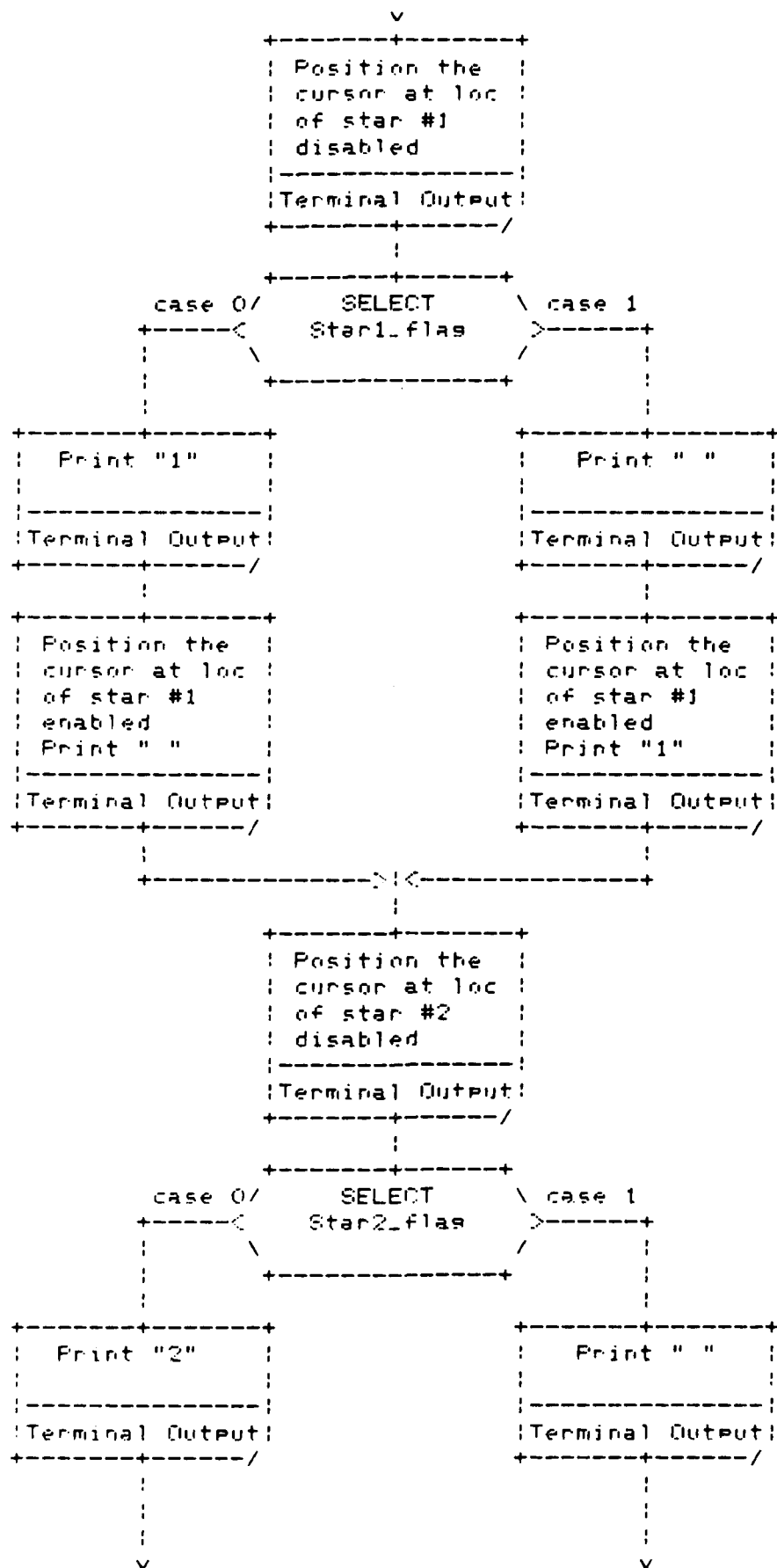
|

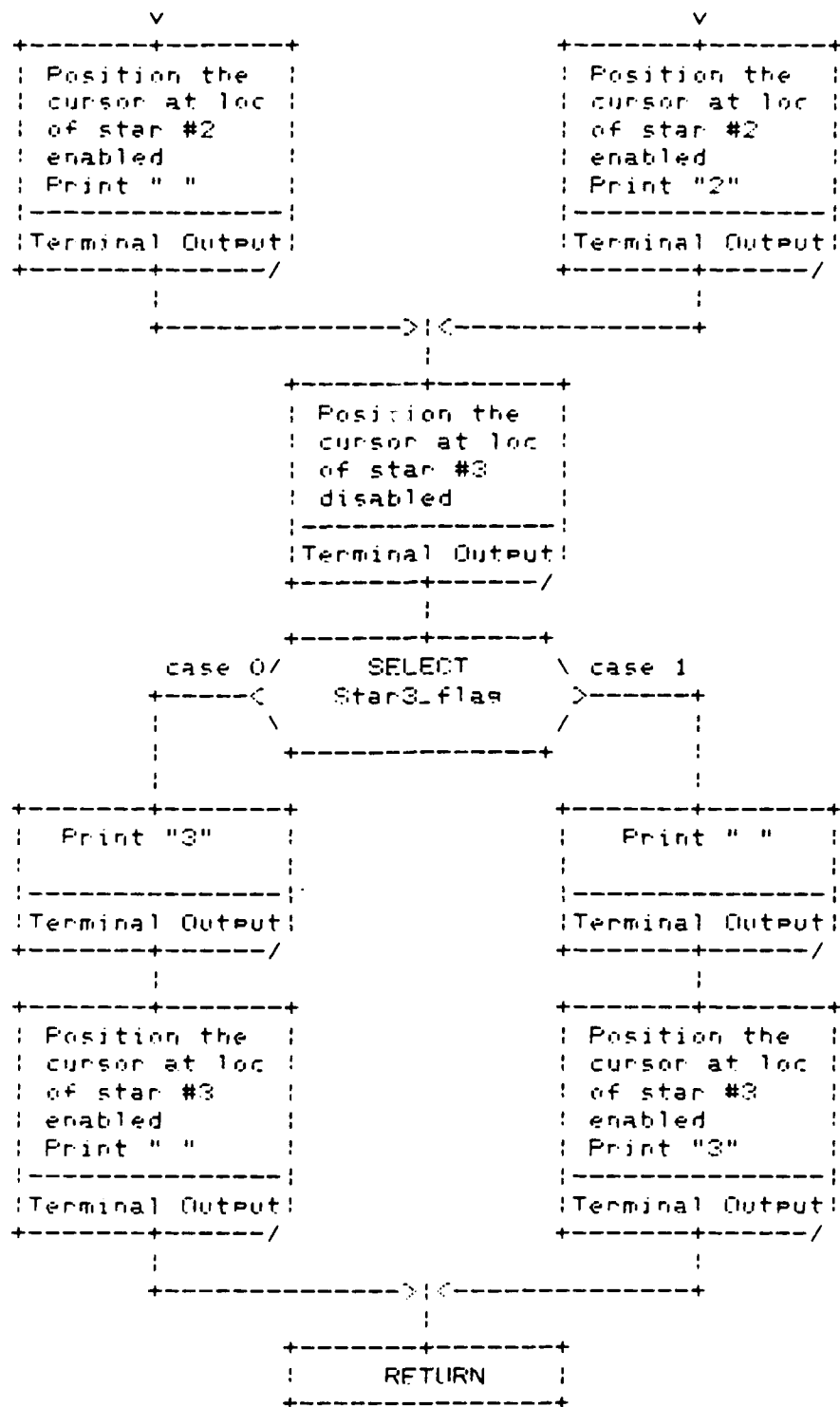
```
+-----+
| GOSUB |
| Check_data |
| (PAGE 15) |
+-----+
```

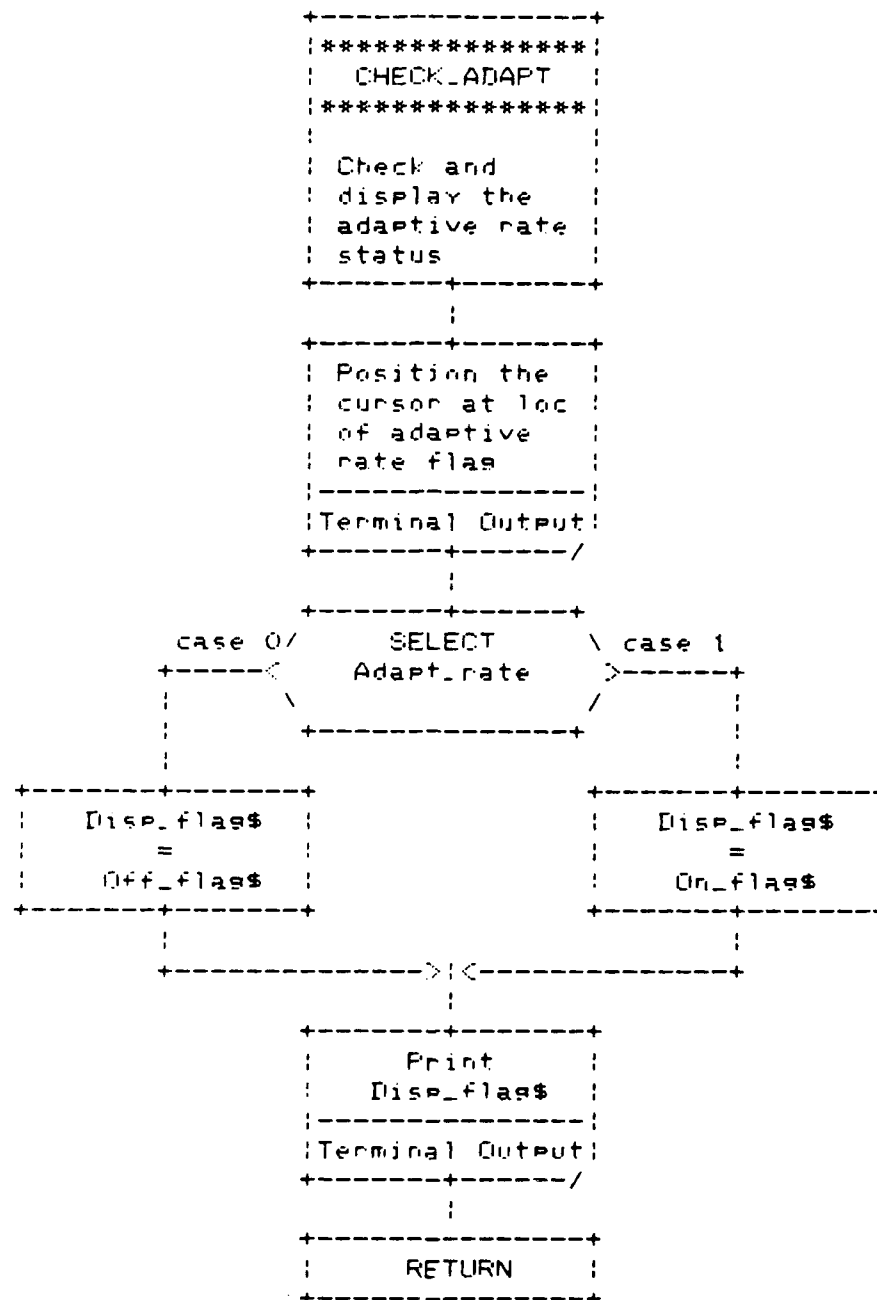
|

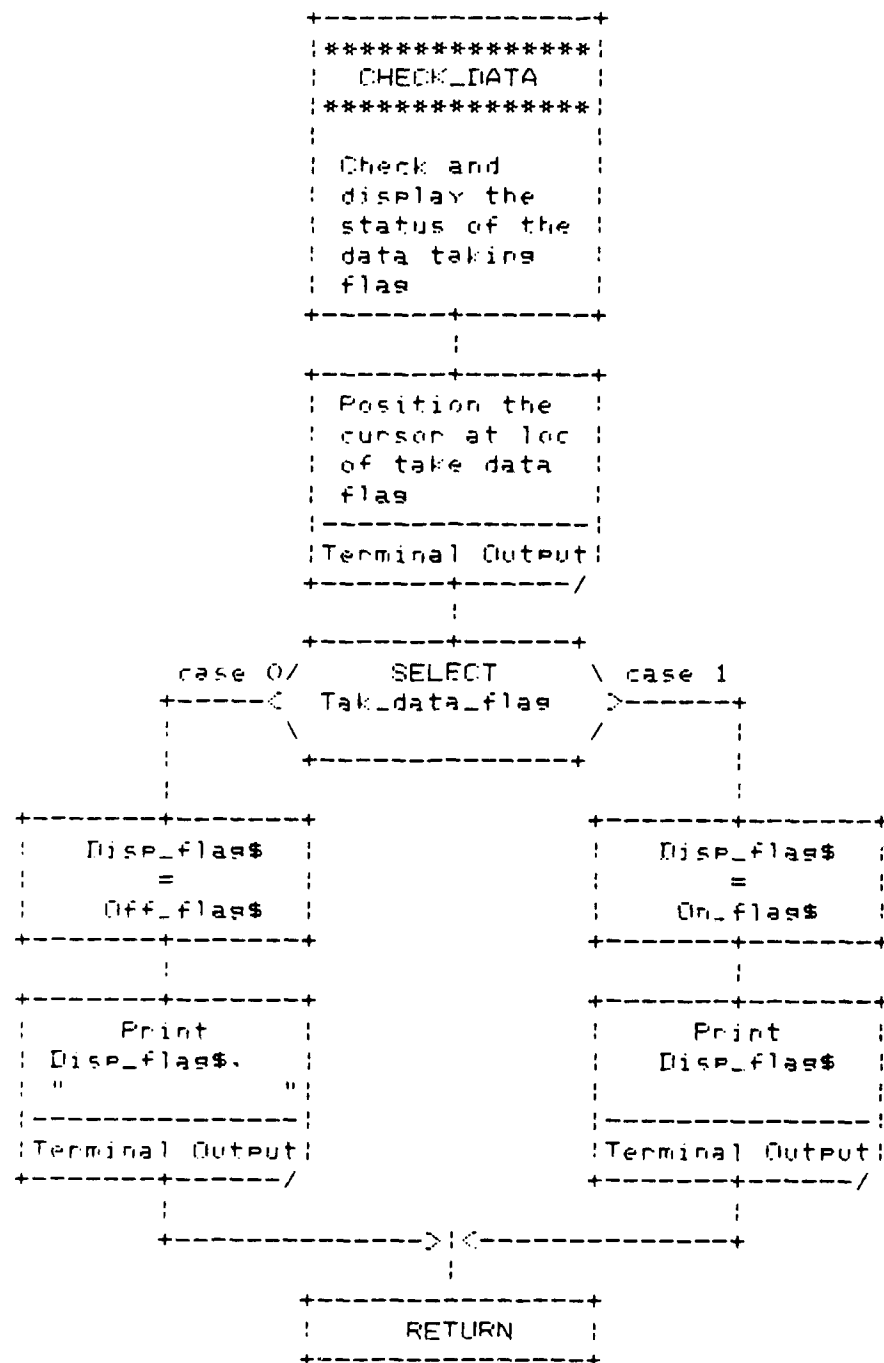
```
+-----+
| RETURN |
+-----+
```

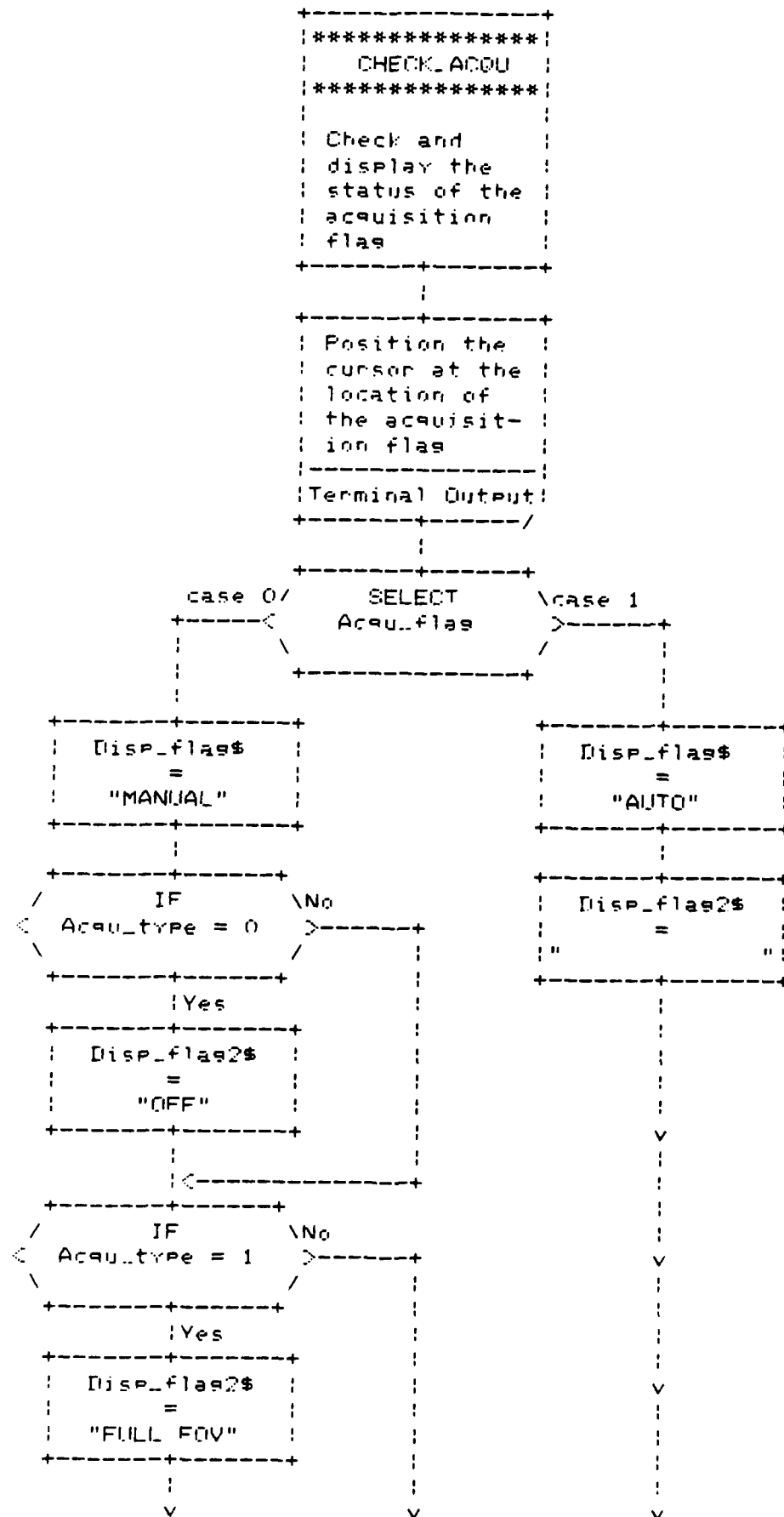


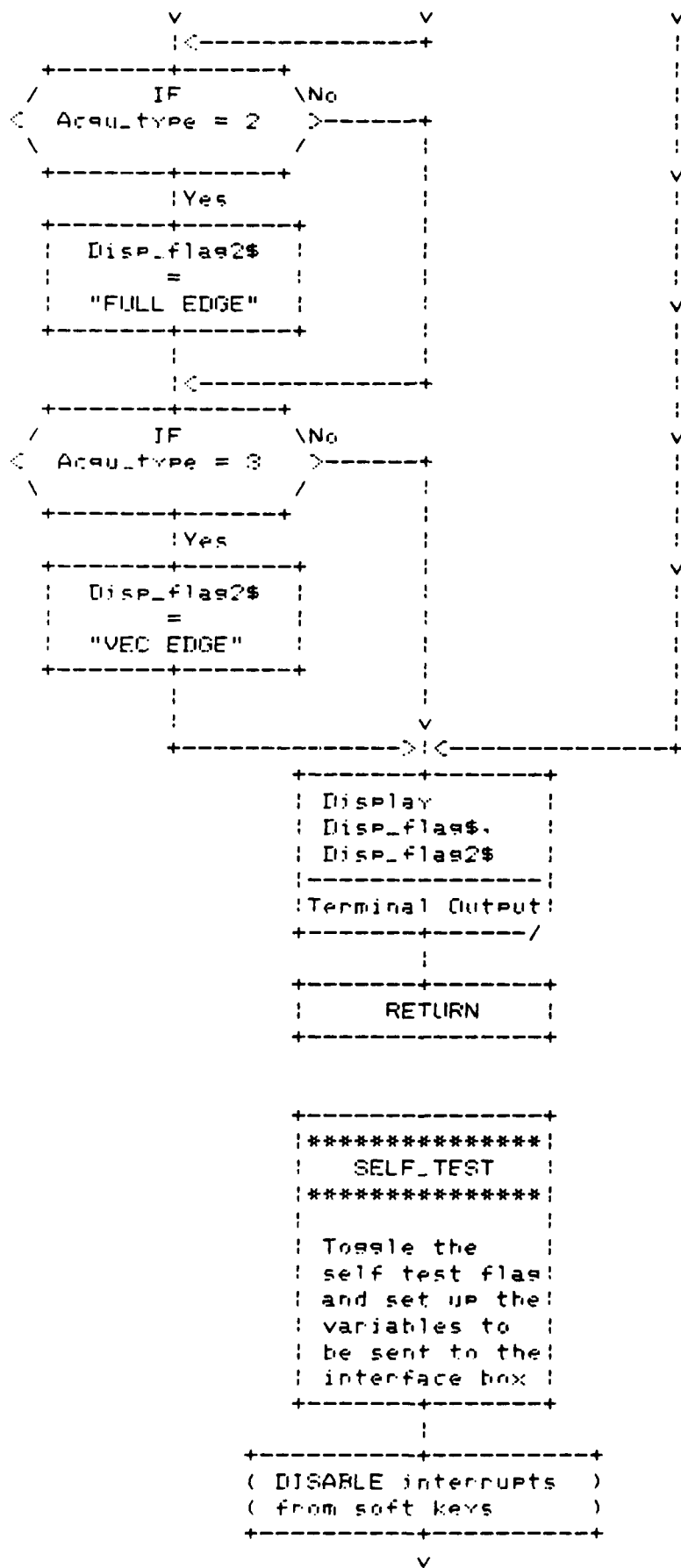








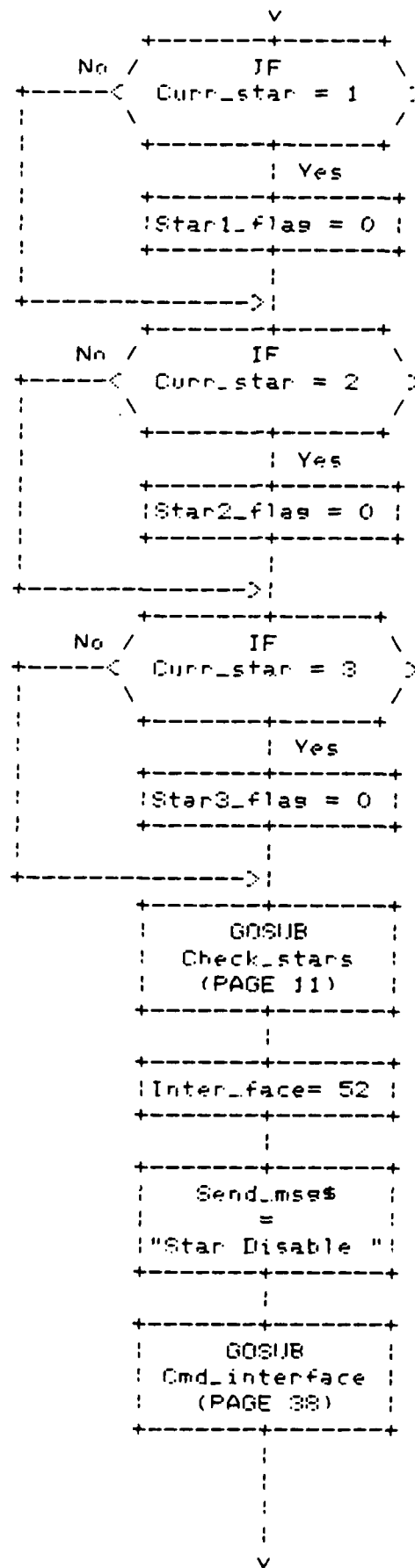


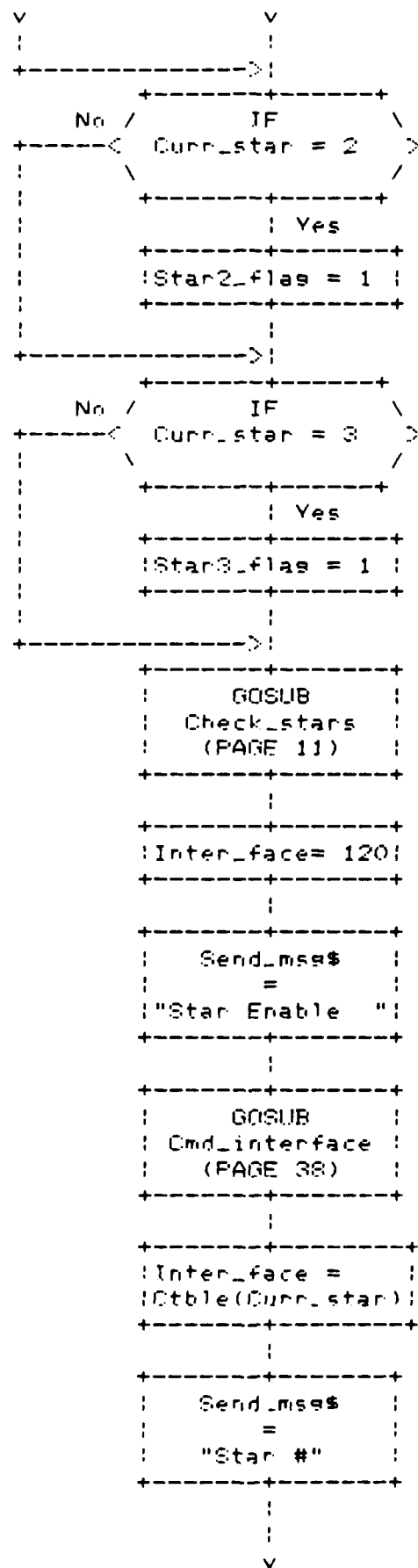


```

|
+-----+
| Self_test_flag |
|   = 1 -       |
| Self_test_flag |
+-----+
|
+-----+
|   GOSUB        |
|   Chk_self_tst |
|   (PAGE 11)    |
+-----+
|
+-----+
| Inter_face= 224|
+-----+
|
+-----+
|   Send_mss$    |
|   =            |
| "Self Test Cmd"|
+-----+
|
+-----+
|   GOSUB        |
|   Cmd_interface|
|   (PAGE 38)    |
+-----+
|
+-----+
| ( ENABLE interrupts ) |
| ( from soft keys   ) |
+-----+
|
+-----+
|   RETURN       |
+-----+
|
+-----+
| *****       |
| STAR_DISABLE   |
| *****       |
|
| Disable the    |
| current star   |
| and change the |
| star flag to   |
| show such      |
| status         |
+-----+
|
+-----+
| ( DISABLE interrupts ) |
| ( from soft keys   ) |
+-----+
|
|
V

```





v

```
+-----+  
| GOSUB |  
| Cmd_Interface |  
| (PAGE 38) |  
+-----+
```

|

```
+-----+  
| ( ENABLE interrupts ) |  
| ( from soft keys ) |  
+-----+
```

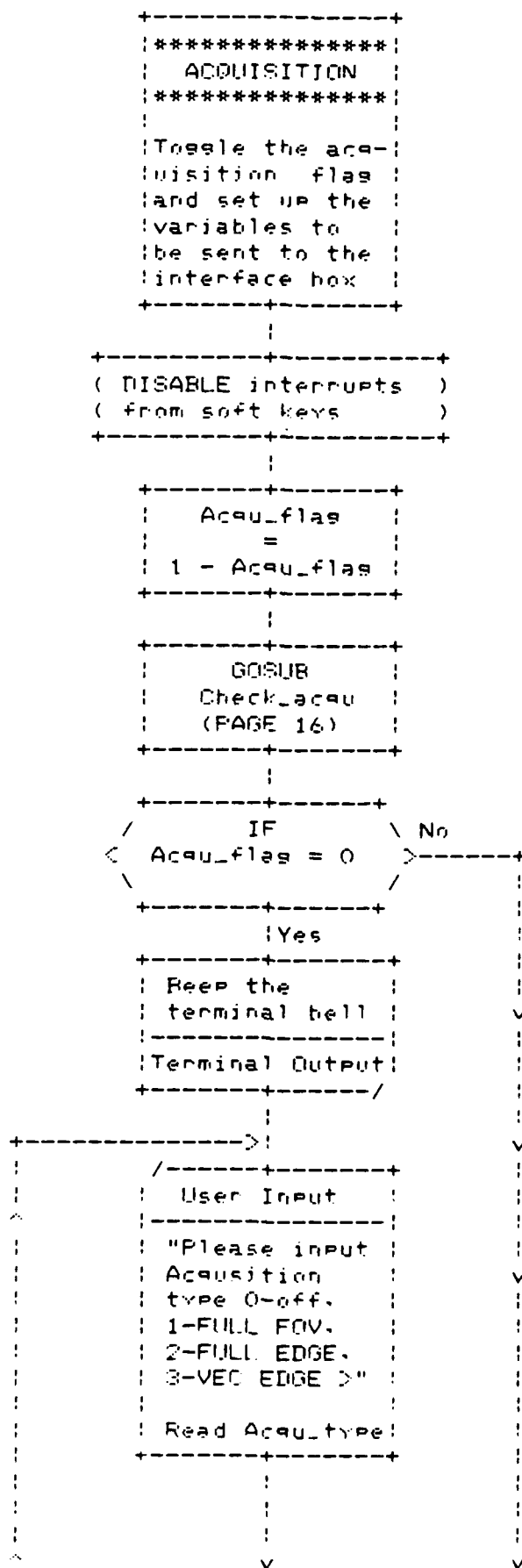
|

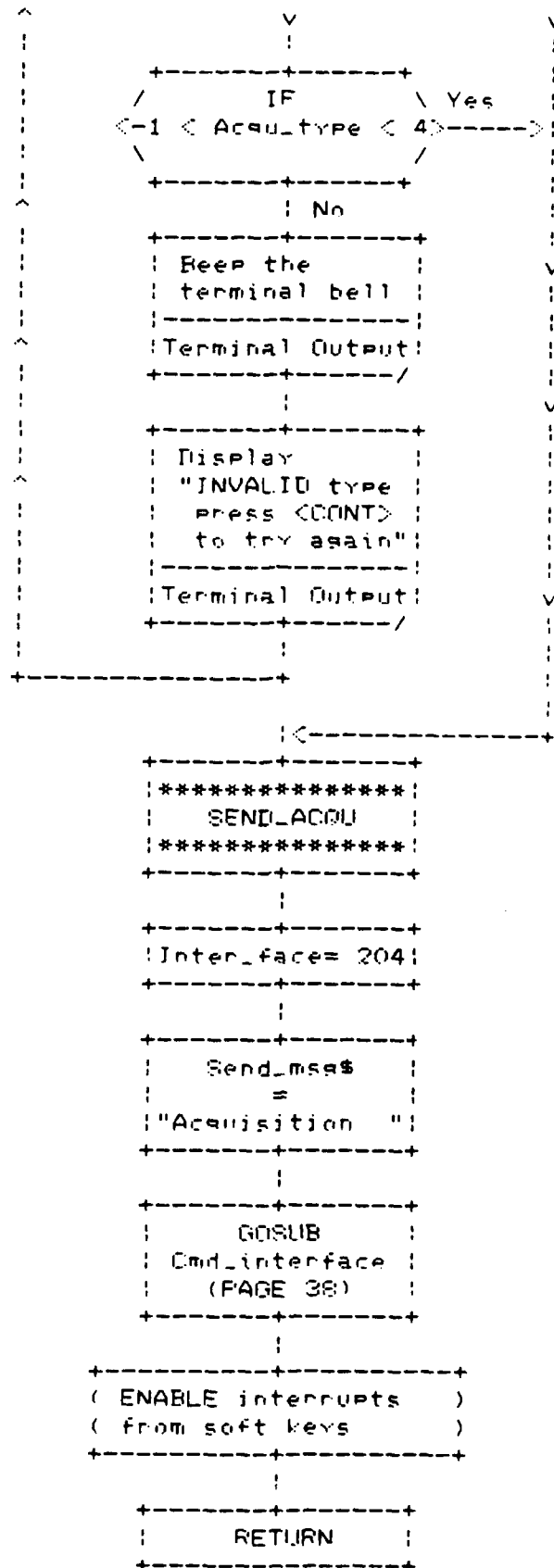
```
+-----+  
| RETURN |  
+-----+
```

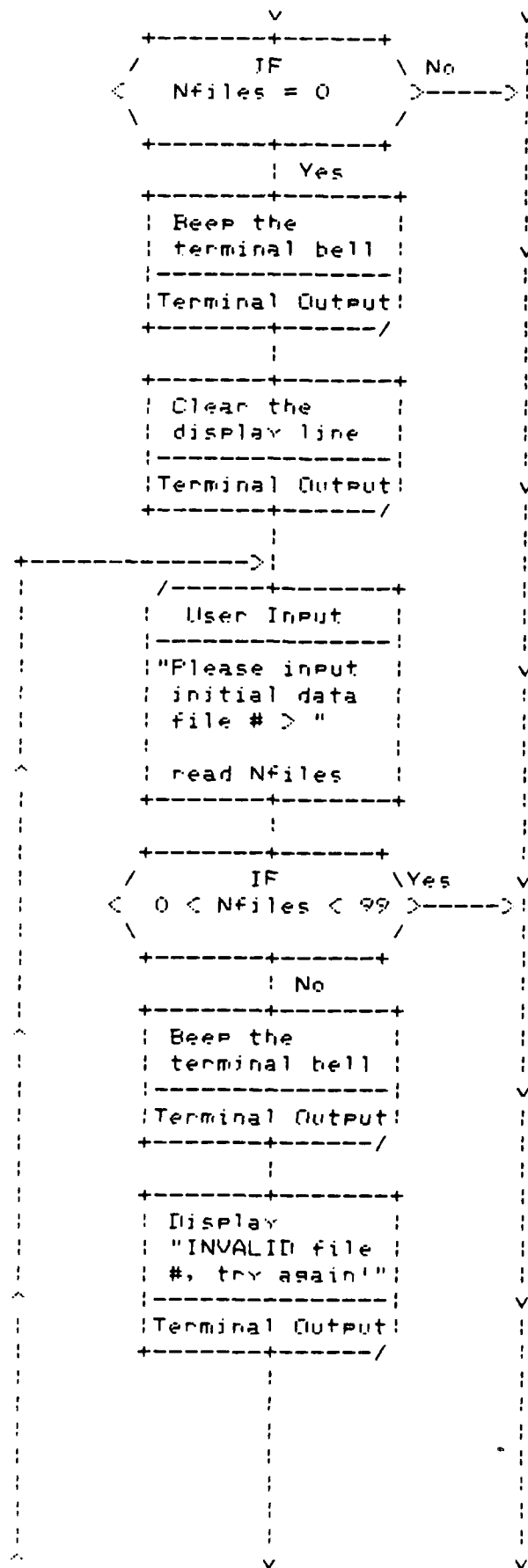
```

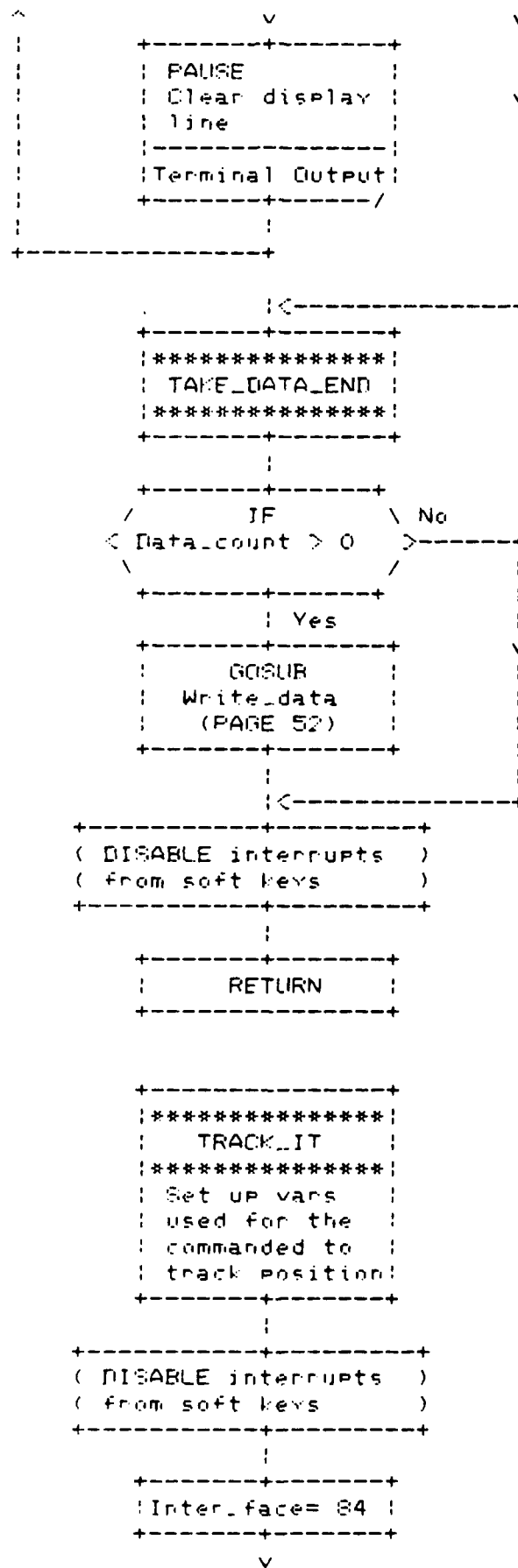
+-----+
|*****|
| ADAPT_RATE |
|*****|
|
|Toggle the ada-|
|ptive rate flas|
|and set up the |
|variables to    |
|be sent to the  |
|interface box   |
|-----+
|
|
+-----+
| ( DISABLE interrupts ) |
| ( from soft keys      ) |
+-----+
|
|
+-----+
|Adapt_rate_flg|
|   = 1 -   |
|Adapt_rate_flg|
+-----+
|
|
+-----+
| GOSUB |
| Check_adapt |
| (PAGE 14) |
+-----+
|
|
+-----+
|Inter_face= 180|
+-----+
|
|
+-----+
| Send_msgs |
|   =       |
| "Adaptive Rate"|
+-----+
|
|
+-----+
| GOSUB |
| Cmd_interface |
| (PAGE 38) |
+-----+
|
|
+-----+
| ( ENABLE interrupts ) |
| ( from soft keys      ) |
+-----+
|
|
+-----+
| RETURN |
+-----+

```





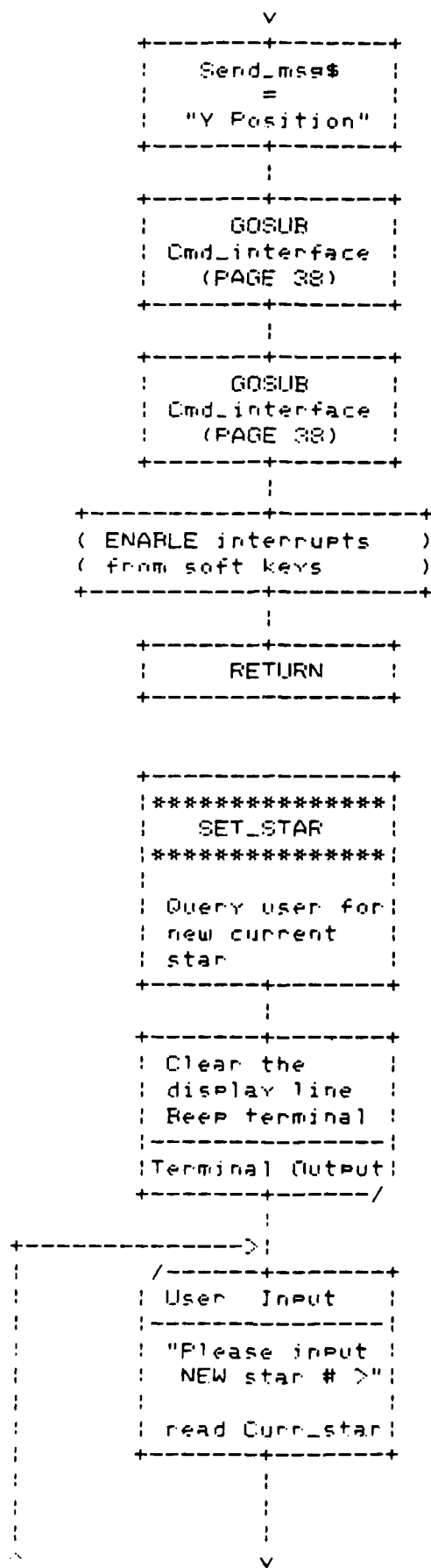


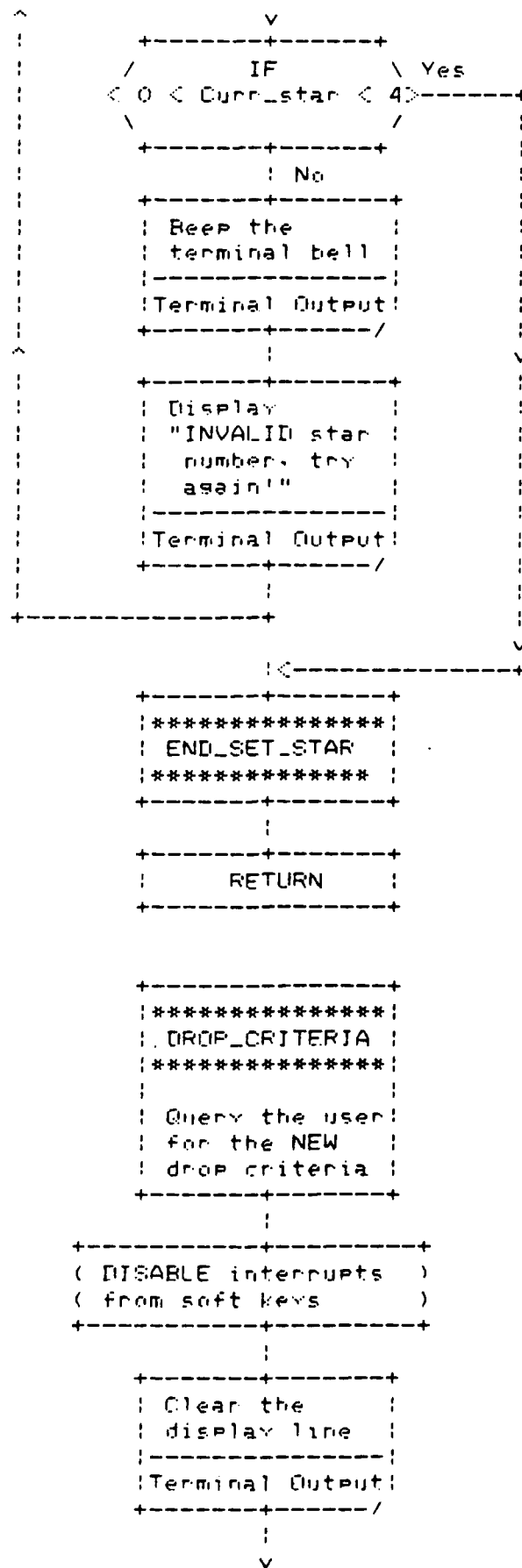


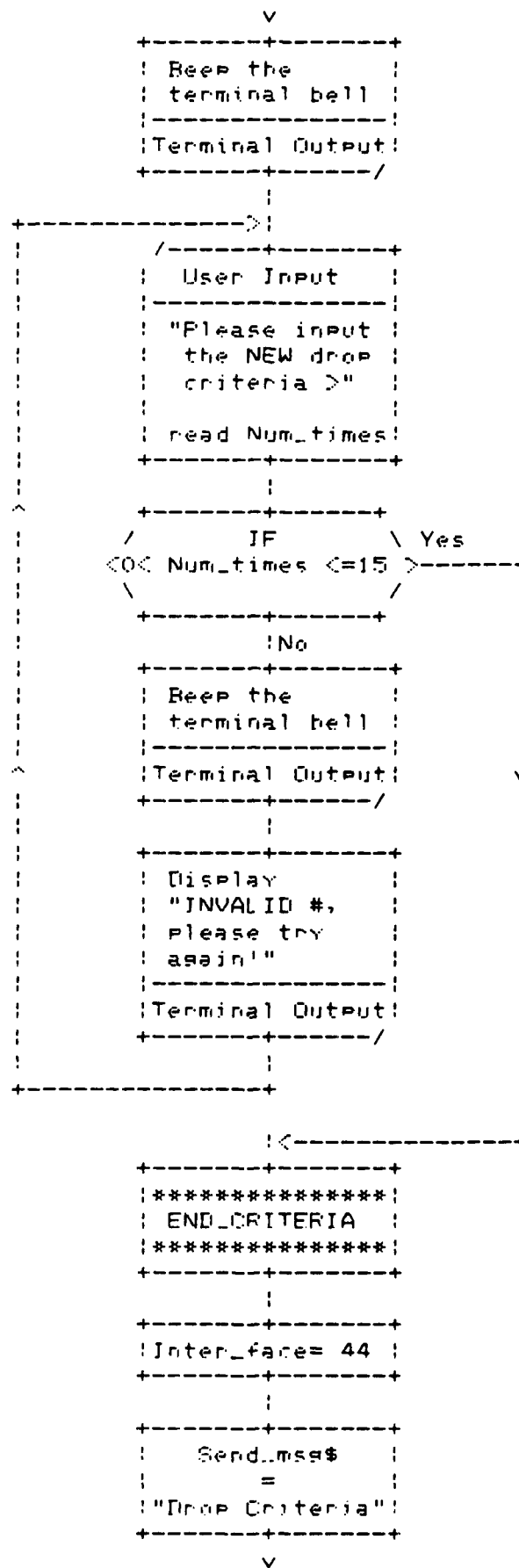
```

      V
      |
      +-----+
      | Send_msg$ |
      |   =       |
      | "Track @ X,Y " |
      +-----+
      |
      +-----+
      | GOSUB      |
      | Cmd_interface |
      | (PAGE 38)   |
      +-----+
      |
      +-----+
      | Inter_face = |
      | Ctblr(Curr_star) |
      +-----+
      |
      +-----+
      | Send_msg$   |
      |   =         |
      | "Star #"    |
      +-----+
      |
      +-----+
      | GOSUB      |
      | Cmd_interface |
      | (PAGE 38)   |
      +-----+
      |
      +-----+
      | Inter_face=X_posn |
      +-----+
      |
      +-----+
      | Send_msg$   |
      |   =         |
      | "X Position" |
      +-----+
      |
      +-----+
      | GOSUB      |
      | Cmd_interface |
      | (PAGE 38)   |
      +-----+
      |
      +-----+
      | GOSUB      |
      | Cmd_interface |
      | (PAGE 38)   |
      +-----+
      |
      +-----+
      | Inter_face=Y_posn |
      +-----+
      |
      V

```



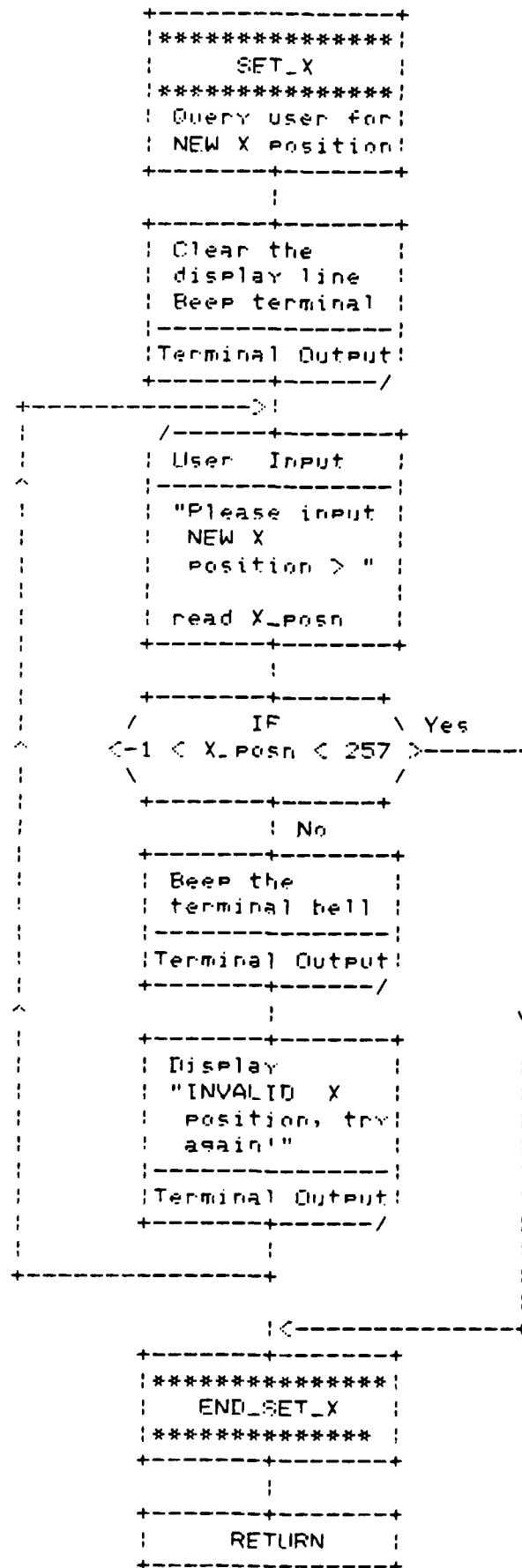


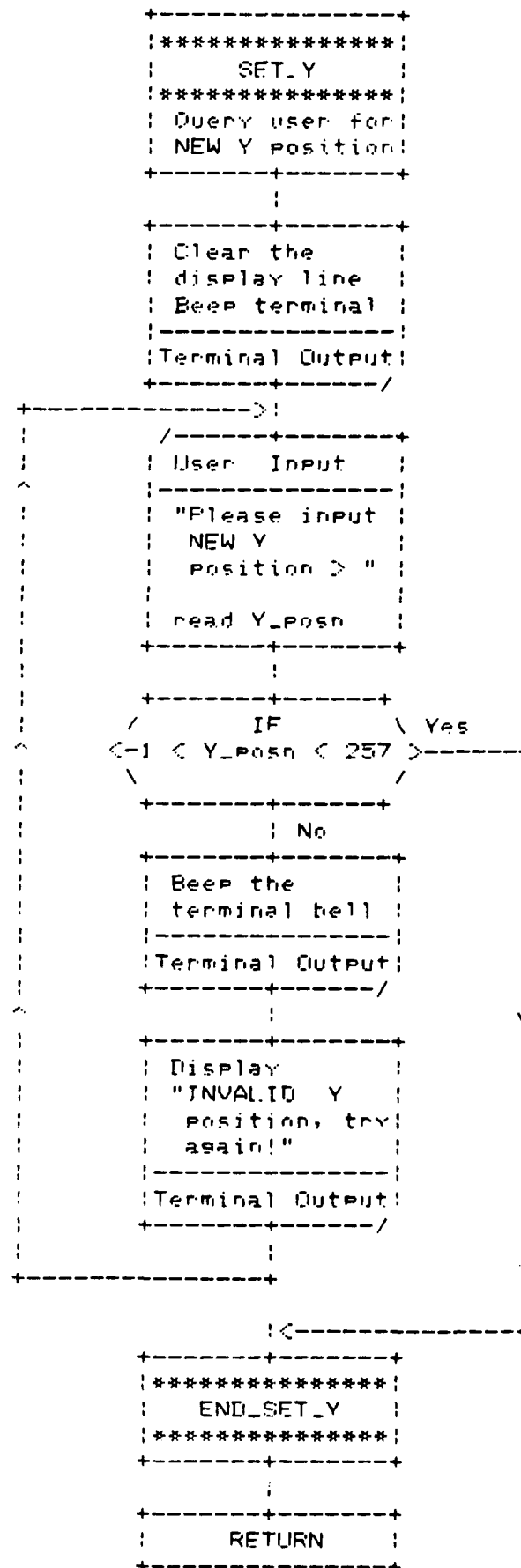


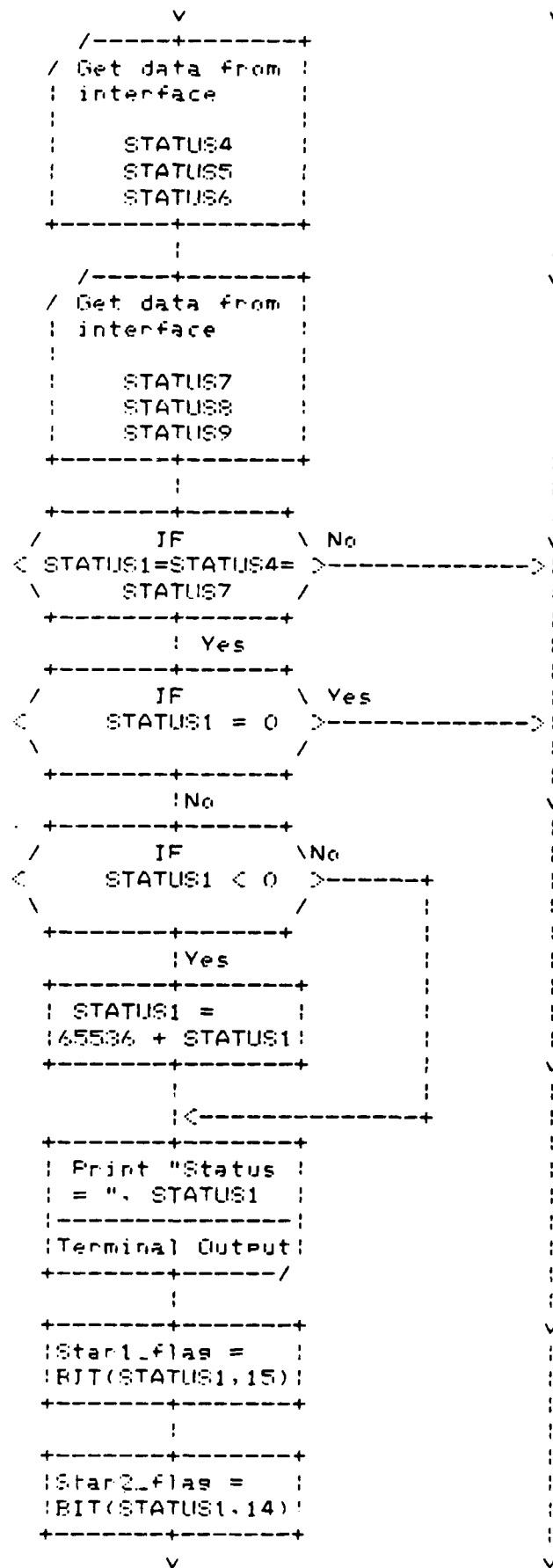
```

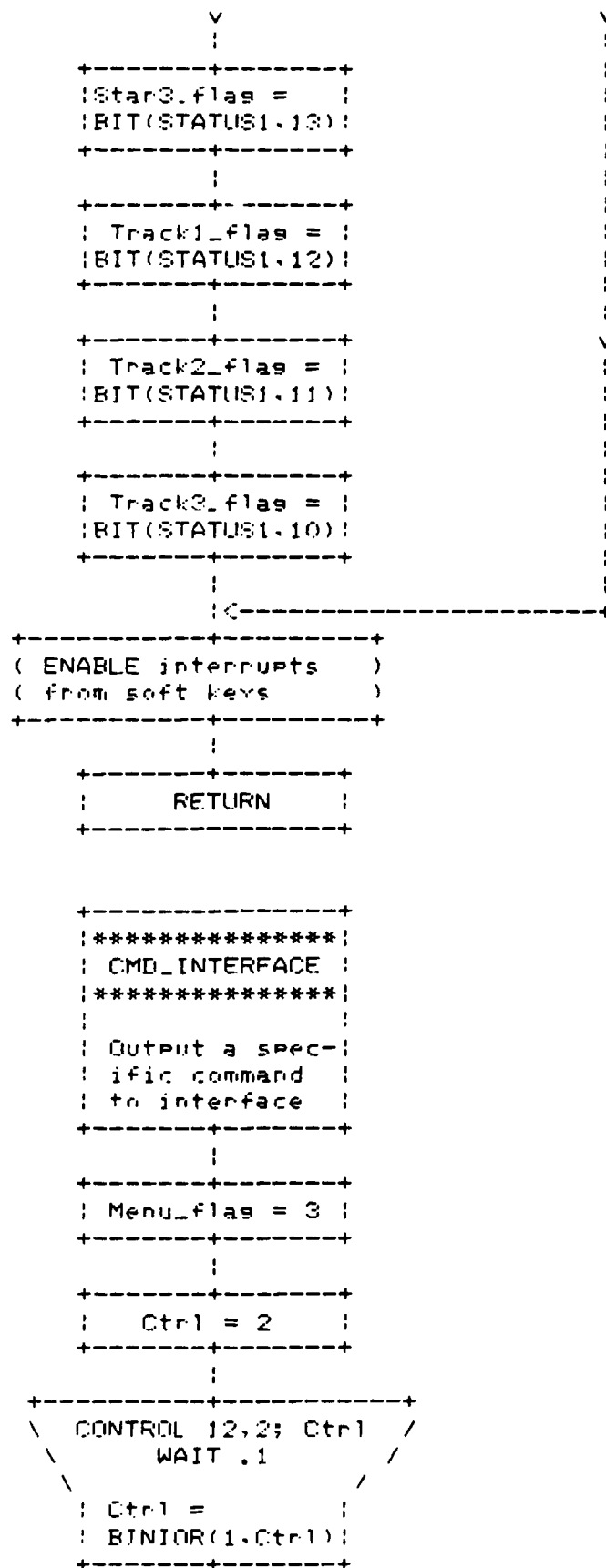
      V
      |
      +-----+
      | GOSUB   |
      | Cmd_interface |
      | (PAGE 38) |
      +-----+
      |
      +-----+
      | Inter_face = |
      | Ctblr(Num_times) |
      +-----+
      |
      +-----+
      | Send_mse$    |
      | =            |
      | "Drop data"  |
      +-----+
      |
      +-----+
      | GOSUB   |
      | Cmd_interface |
      | (PAGE 38) |
      +-----+
      |
      +-----+
      | ( ENABLE interrupts ) |
      | ( from soft keys   ) |
      +-----+
      |
      +-----+
      | RETURN      |
      +-----+

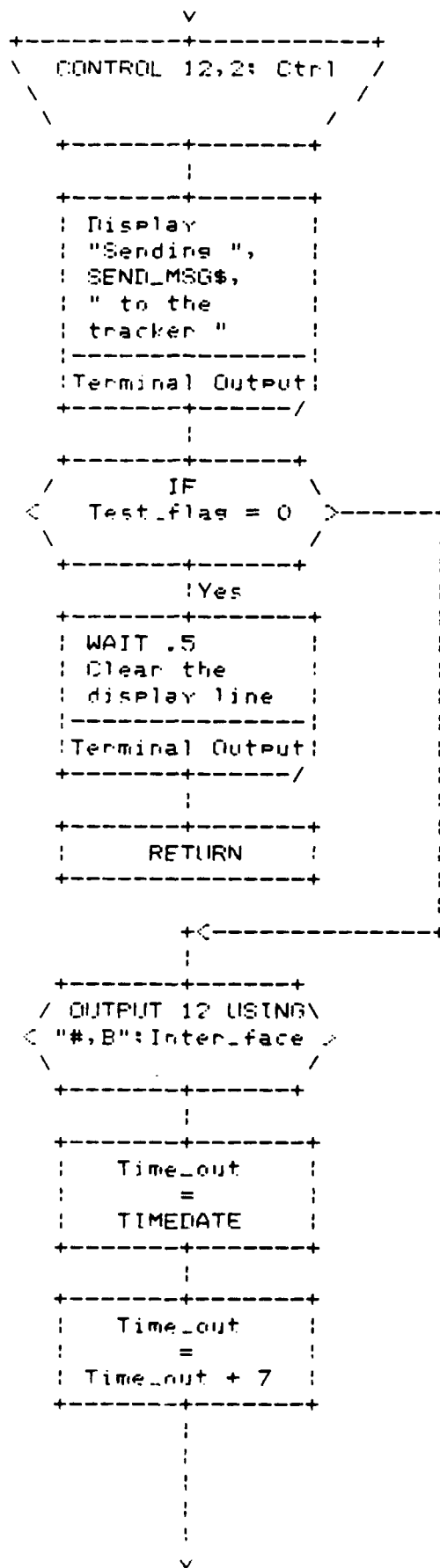
```

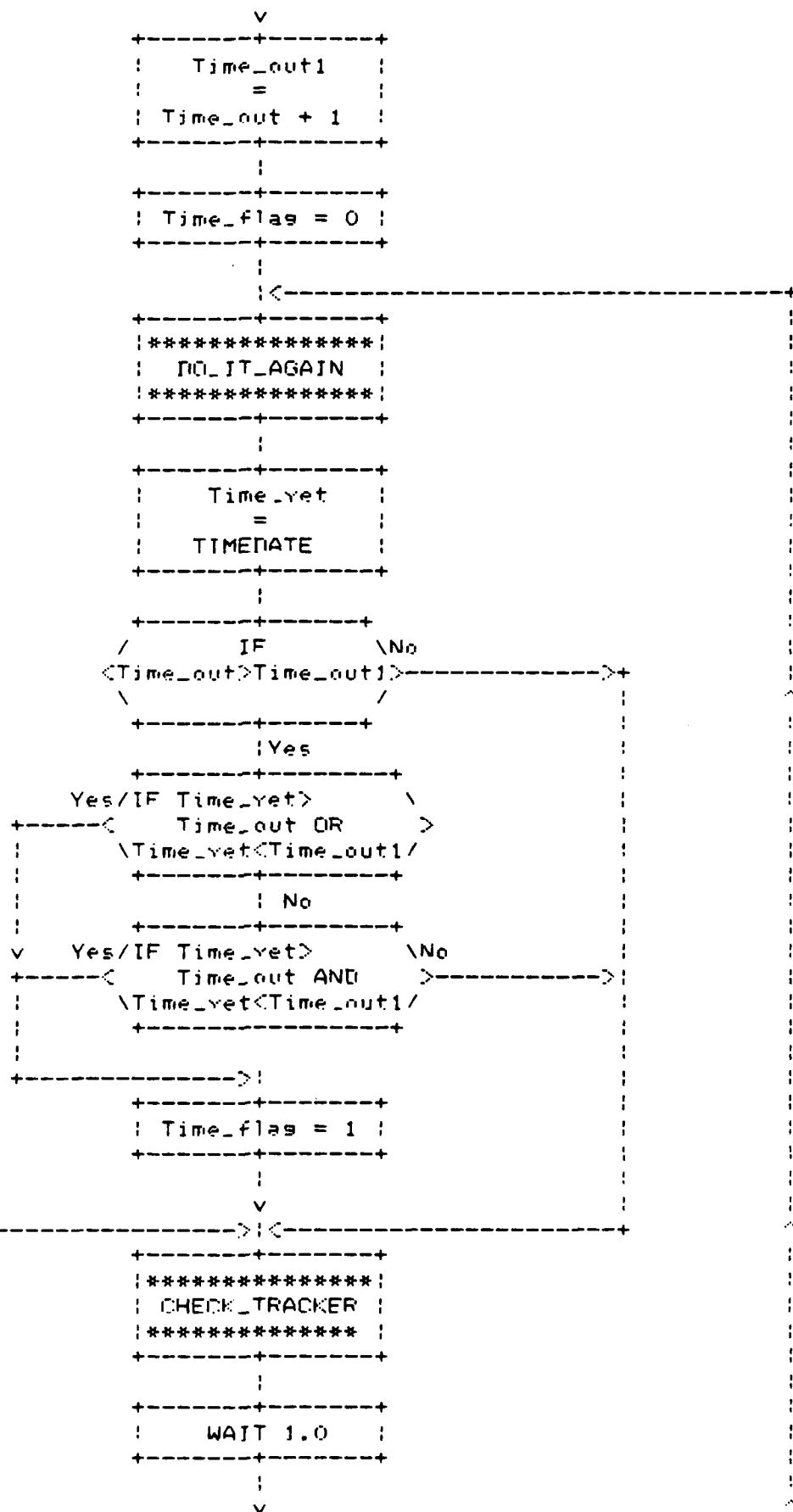


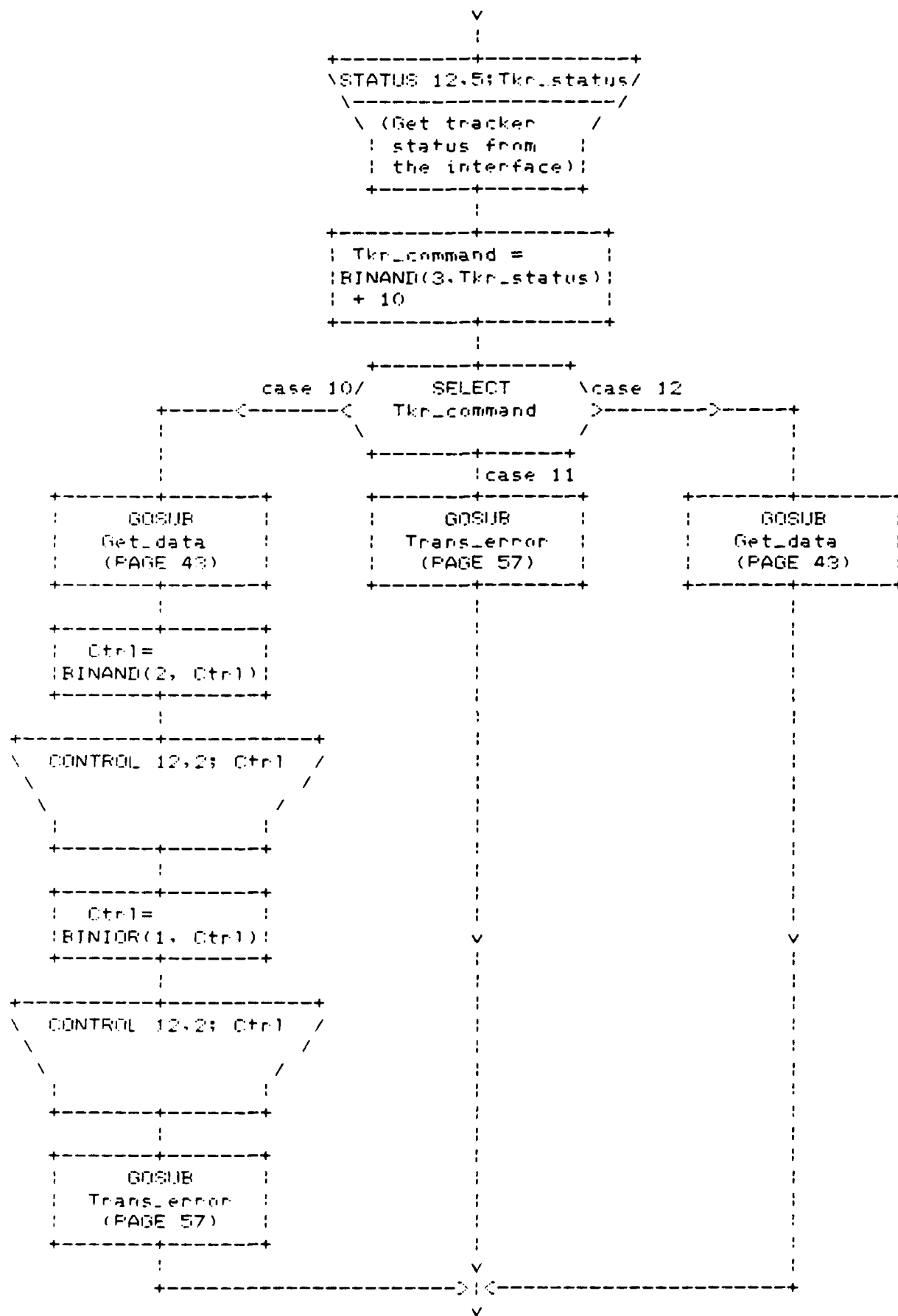


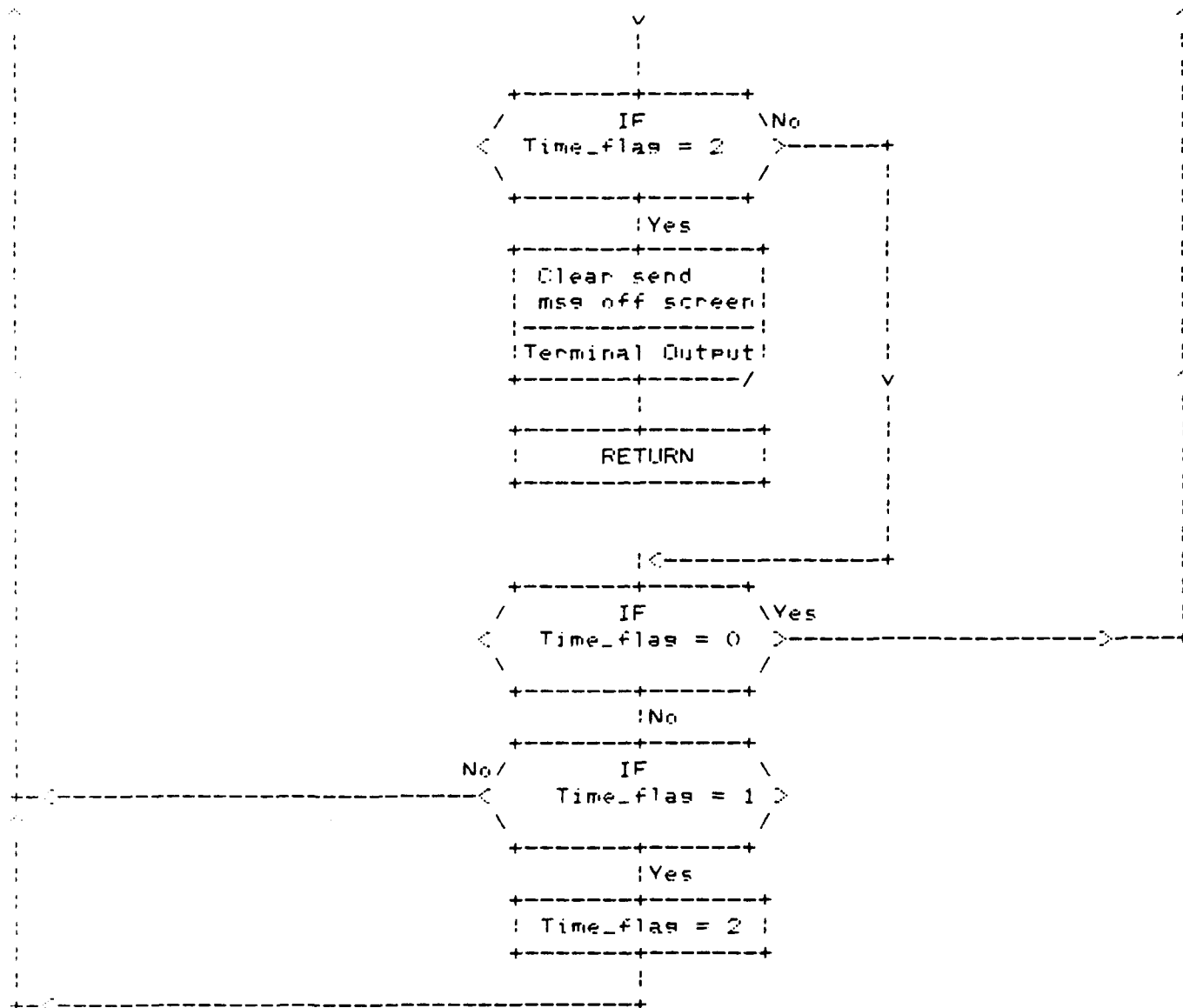


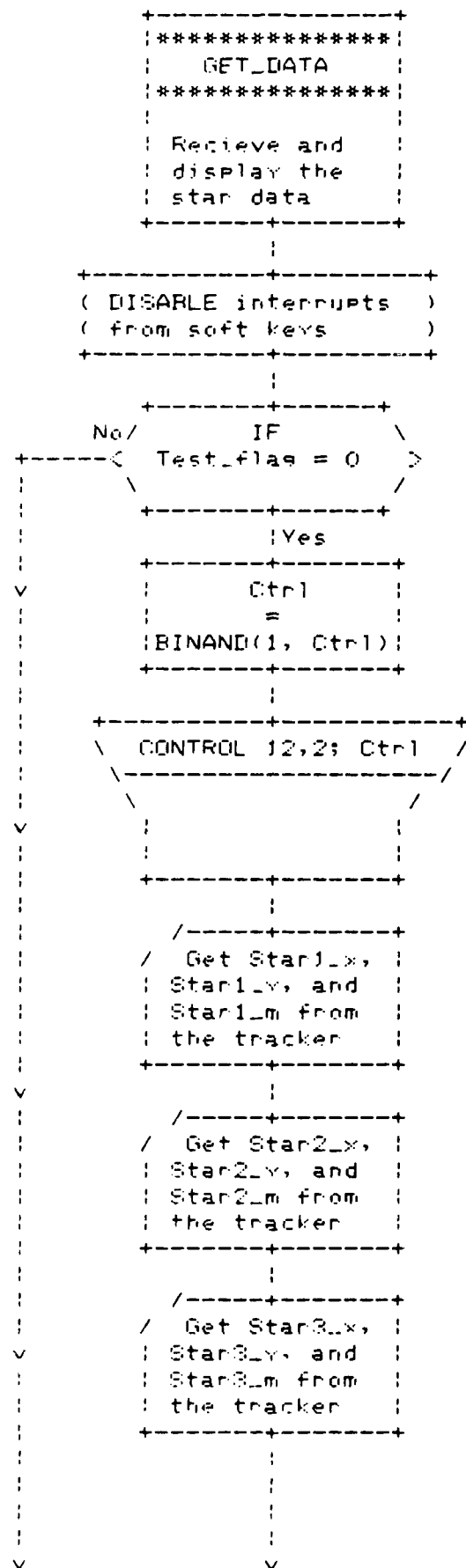


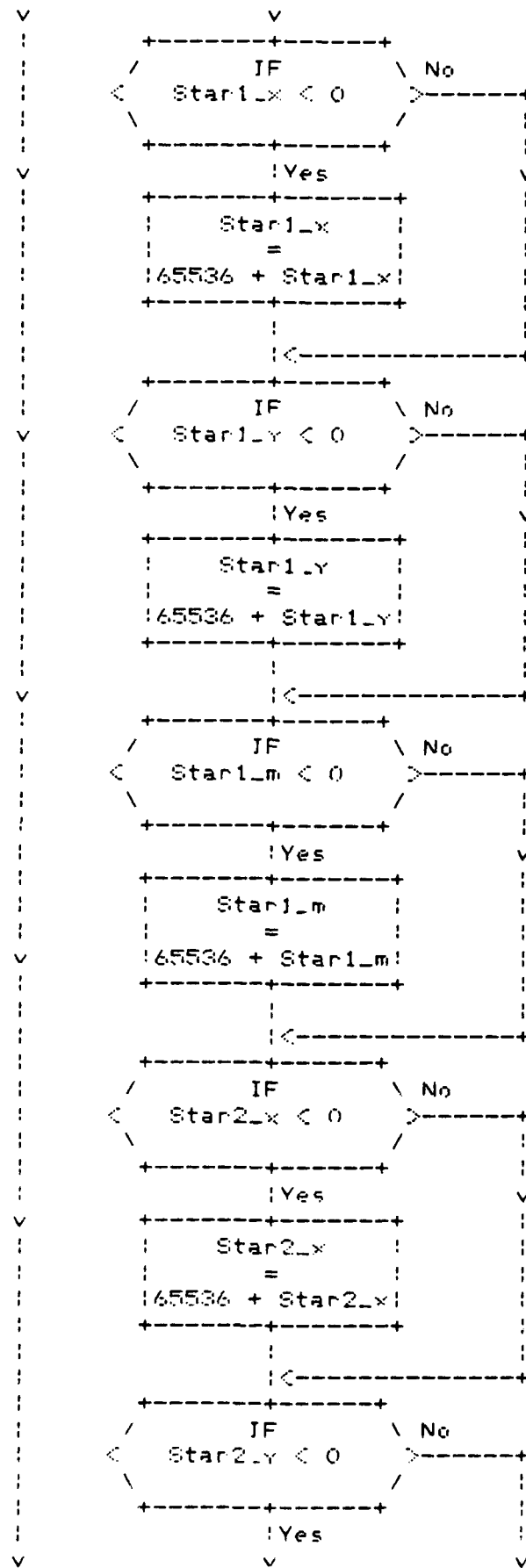


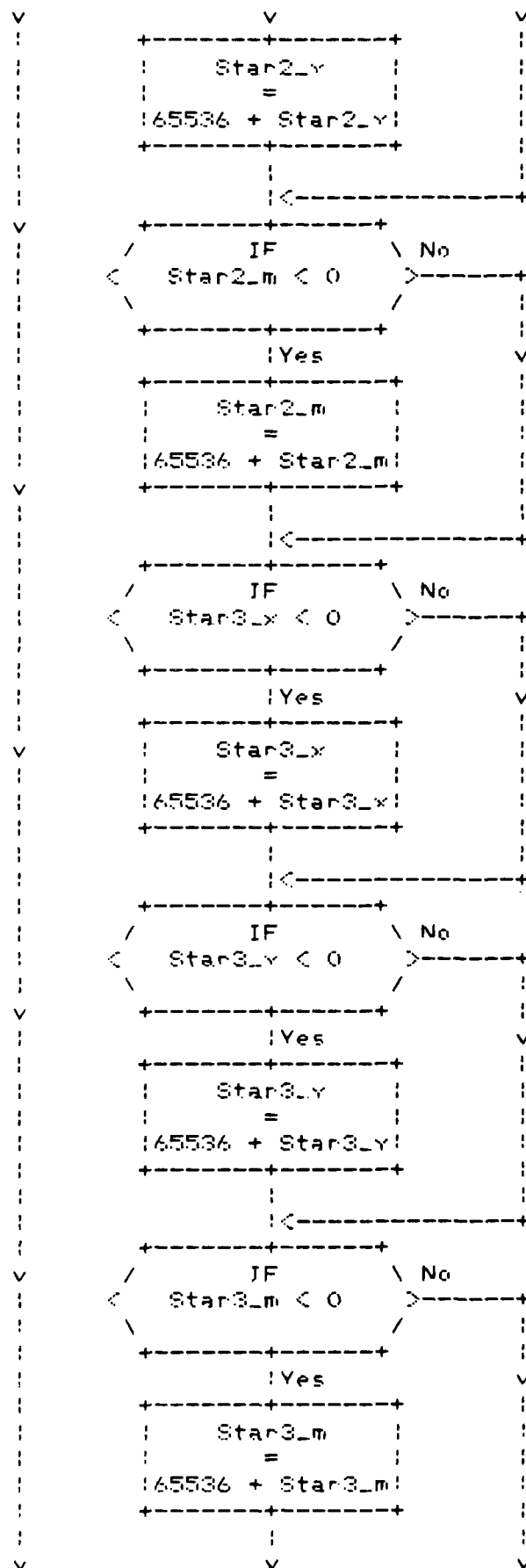


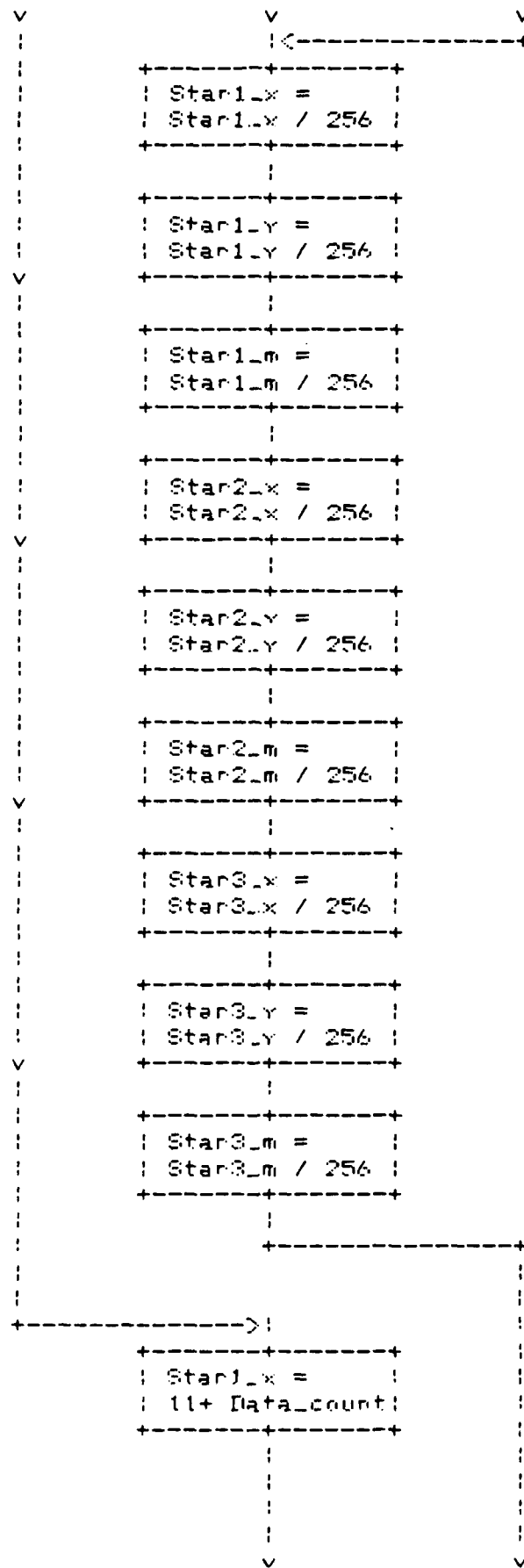


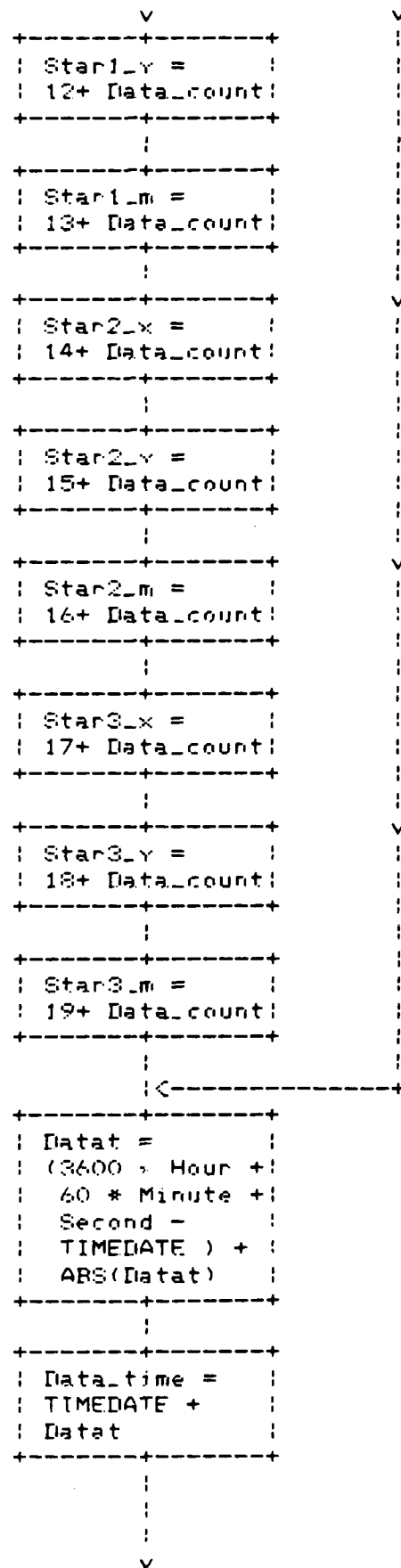








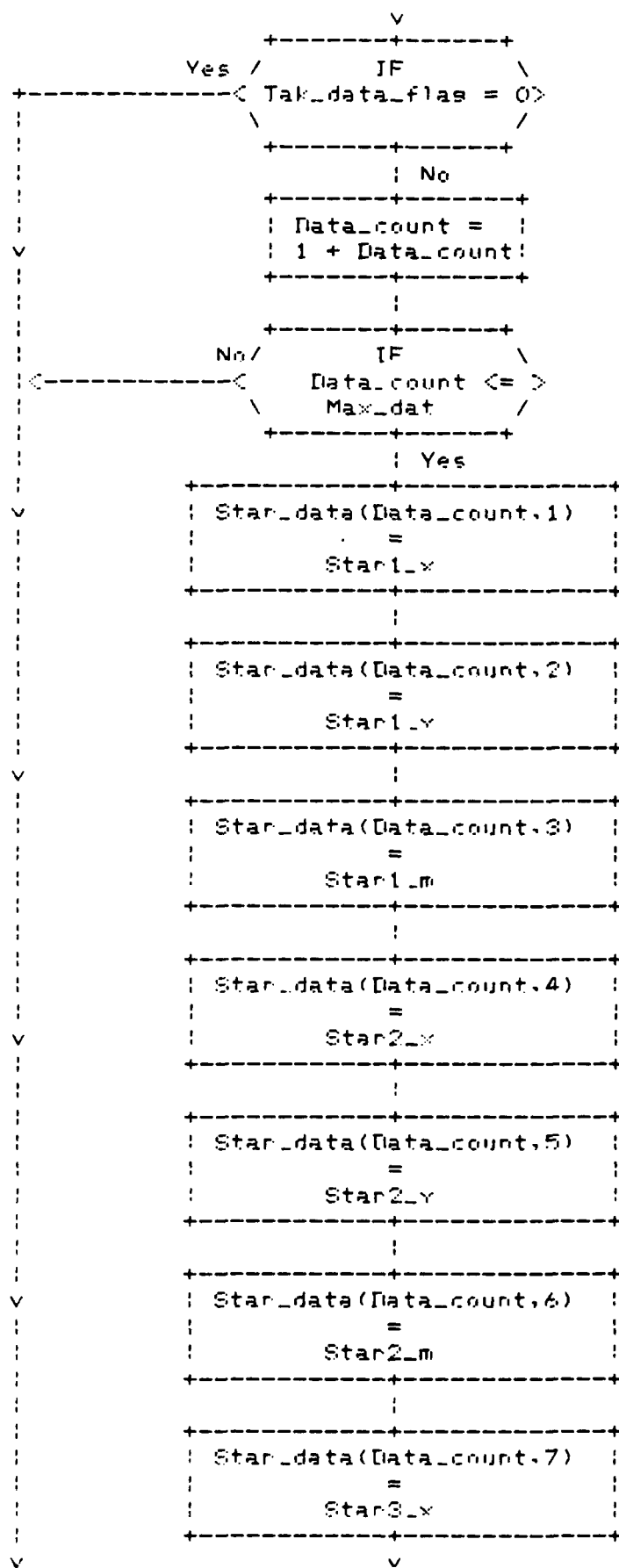


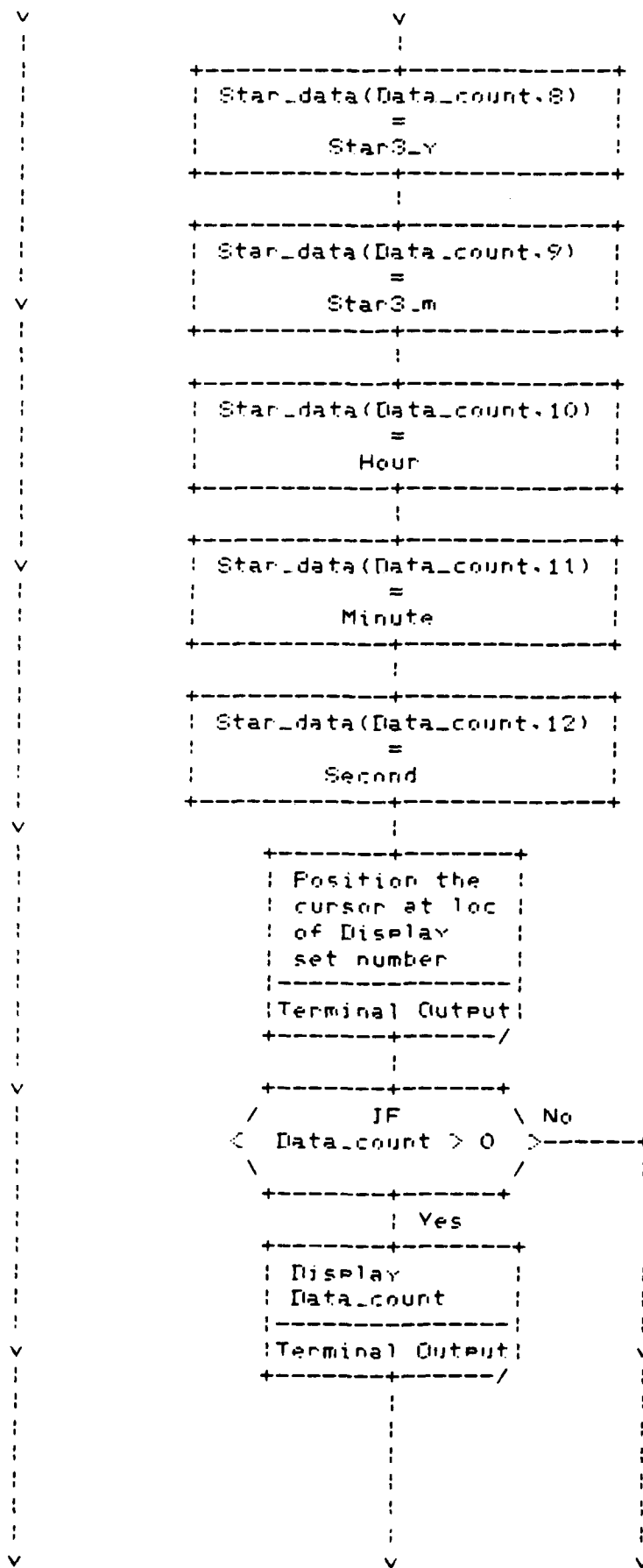


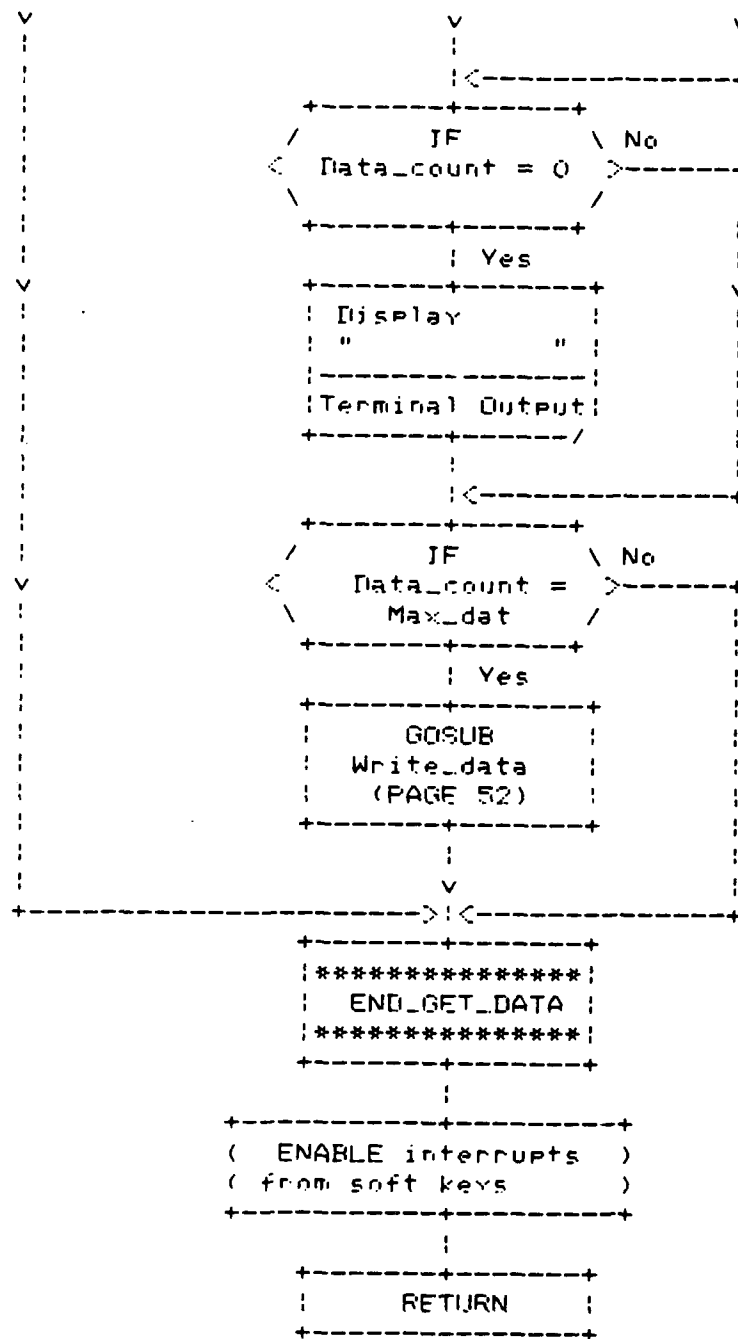
```

v
+-----+
| Data_time = |
| INT( Data_time |
| * 100) / 100 |
+-----+
|
+-----+
| Hour = |
| Data_time |
| DIV 3600 |
+-----+
|
+-----+
| Minute = |
| Data_time |
| MOD 3600 DIV 60 |
+-----+
|
+-----+
| Second = |
| Data_time |
| MOD 60 |
+-----+
|
+-----+
| Display the |
| update toggle |
+-----+
| Terminal Output |
+-----/
|
+-----+
| Display |
| Star1_x |
| Star1_y |
| Star1_m |
+-----+
| Terminal Output |
+-----/
|
+-----+
| Display |
| Star2_x |
| Star2_y |
| Star2_m |
+-----+
| Terminal Output |
+-----/
|
+-----+
| Display |
| Star3_x |
| Star3_y |
| Star3_m |
+-----+
| Terminal Output |
+-----/
|
v

```







```
+-----+
|*****|
|WRITE_DATA|
|*****|
```

```
| Write the star|
| data to the   |
| output file   |
```

```
+-----+
|ON ERROR|
|GOTO    |
|Error_tst|
```

```
| (trap any    |
| errors that  |
| may occur in |
| this routine)|
```

```
+-----+
|Number$|
|=|
|VAL$(Nfiles)|
```

```
+-----+
|File_string$(6)|
|= Number$(1)|
```

```
+-----+
|Display|
|"Writing data|
| to file ">|
| Nfiles|
```

```
| Terminal Output|
+-----/
```

```
+-----+
|MASS STORAGE|
|IS...|
```

```
| assign the   |
| right disc   |
| drive for data|
```

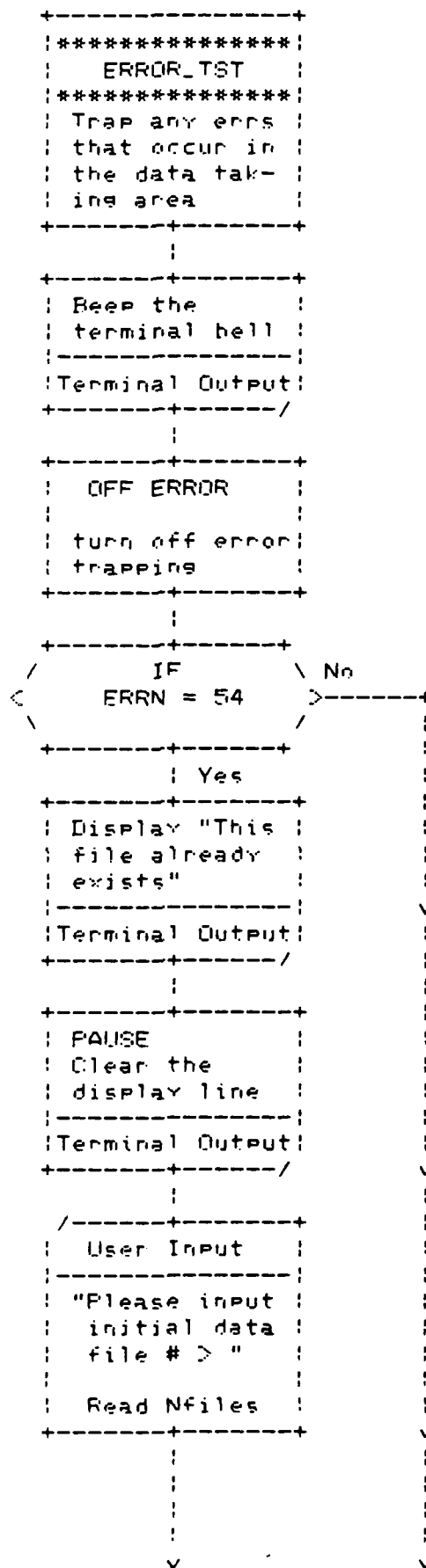
```
+-----+
|CREATE BDAT|
| (Open the data|
| file)|
```

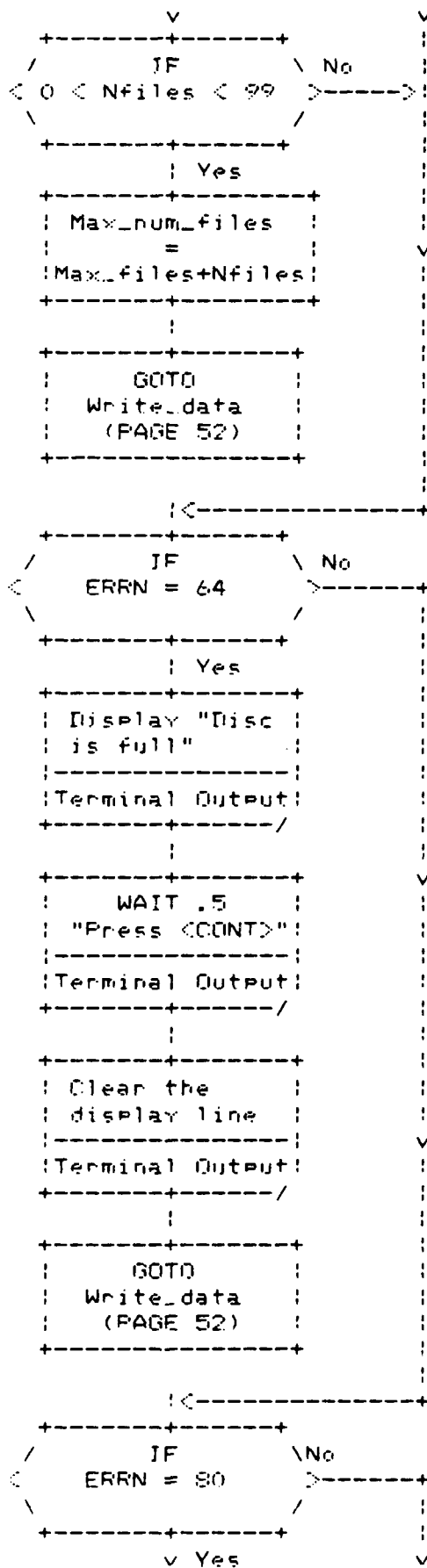
v

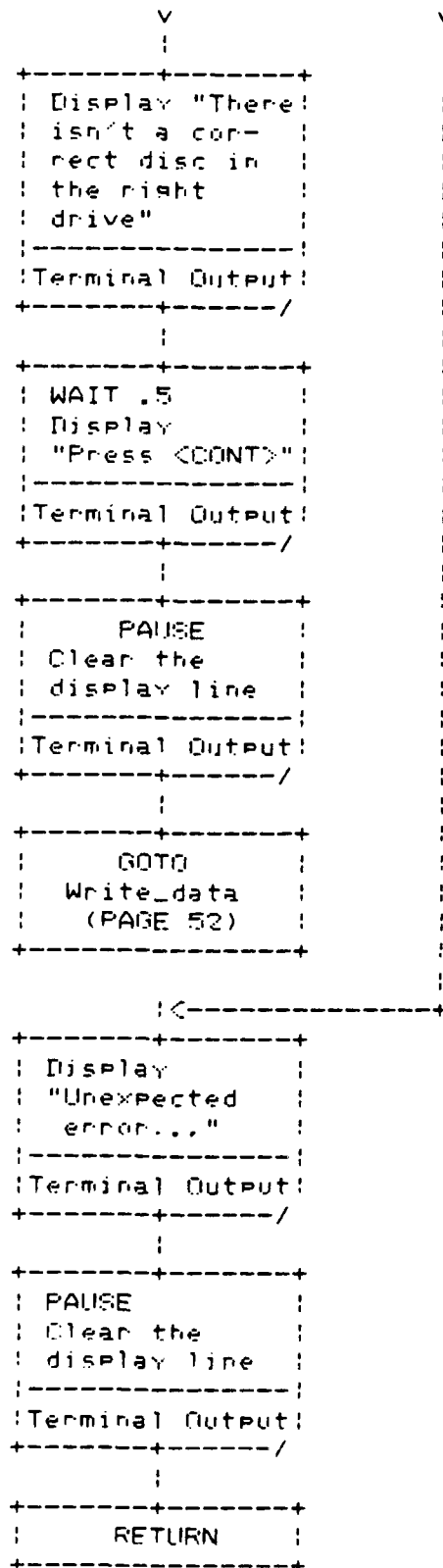
```

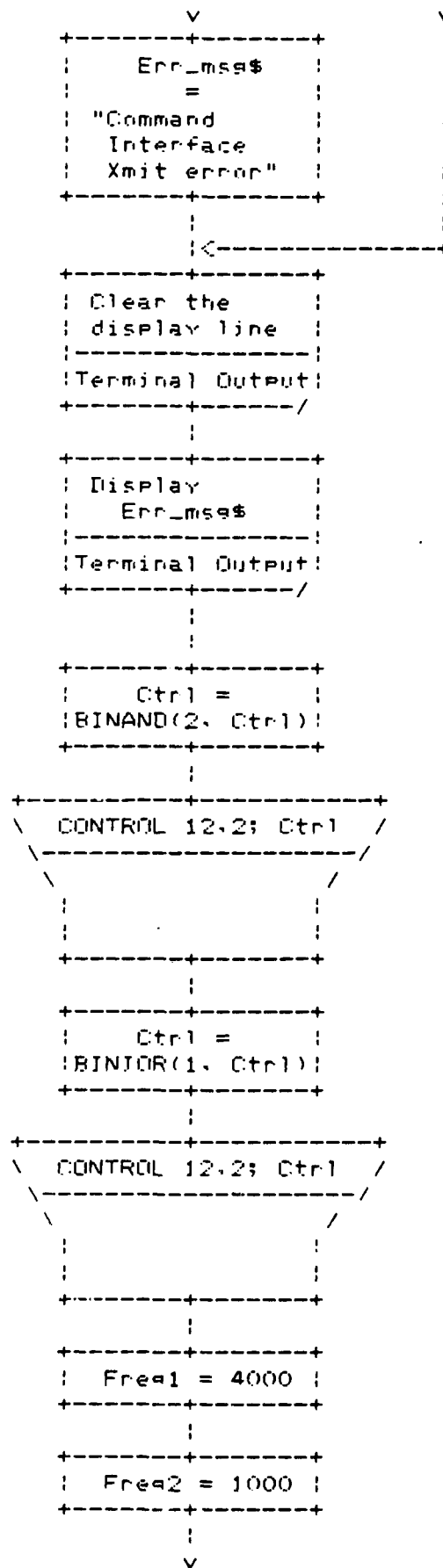
      V
+-----+
| ASSIGN an |
| output path |
| name to file |
| on disk |
+-----+
|
+-----+ \
| WRITE      | \
| Star_data(*) |
| to the file |
+-----+
|
+-----+
| CLOSE the |
| data file |
+-----+
|
+-----+
| Nfiles    |
| =         |
| 1 + Nfiles |
+-----+
|
+-----+
| Data_count = 0 |
+-----+
|
+-----+
| OFF ERROR |
| turn off the |
| error trapping |
+-----+
|
+-----+
| Clear the |
| display line |
+-----+
| Terminal Output |
+-----+ /
|
+-----+
| RETURN |
+-----+

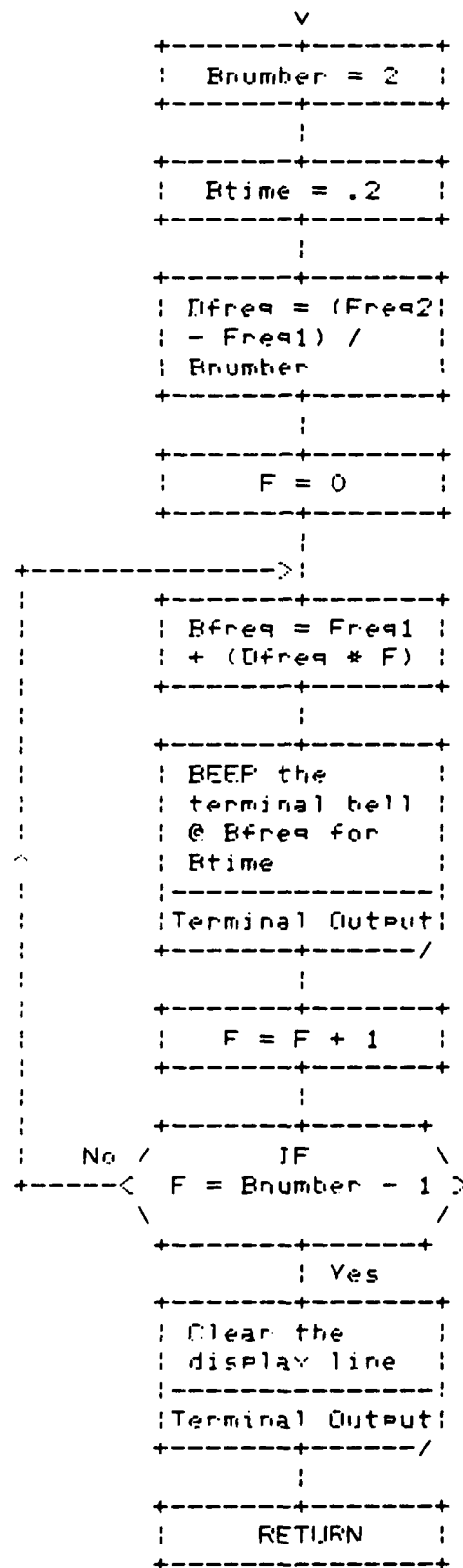
```











```
+-----+
|*****|
|SET_UP_TABLE|
|*****|
|
|Set up the
|values for a
|command table
|
+-----+
```

```
+-----+
|Ctble(1)|
|   =    |
|  224   |
|
+-----+
```

```
+-----+
|Ctble(2)|
|   =    |
|  152   |
|
+-----+
```

```
+-----+
|Ctble(3)|
|   =    |
|  120   |
|
+-----+
```

```
+-----+
|Ctble(4)|
|   =    |
|   84   |
|
+-----+
```

```
+-----+
|Ctble(5)|
|   =    |
|  180   |
|
+-----+
```

```
+-----+
|Ctble(6)|
|   =    |
|  204   |
|
+-----+
```

```
+-----+
|Ctble(7)|
|   =    |
|   44   |
|
+-----+
```

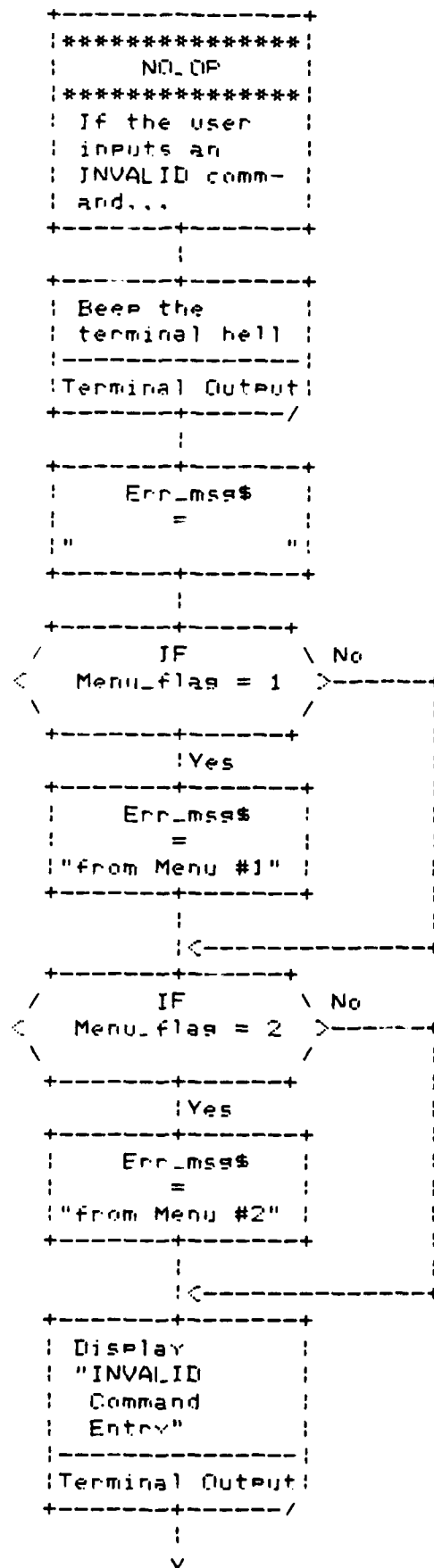
```
+-----+
|Ctble(8)|
|   =    |
|  210   |
|
+-----+
```

v

```

      v
+-----+
| Ctble(9) |
|   =      |
|   50     |
+-----+
|   |
+-----+
| Ctble(10) |
|   =      |
|   74     |
+-----+
|   |
+-----+
| Ctble(11) |
|   =      |
|  172     |
+-----+
|   |
+-----+
| Ctble(12) |
|   =      |
|  136     |
+-----+
|   |
+-----+
| Ctble(13) |
|   =      |
|  102     |
+-----+
|   |
+-----+
| Ctble(14) |
|   =      |
|   30     |
+-----+
|   |
+-----+
| Ctble(15) |
|   =      |
|  254     |
+-----+
|   |
+-----+
| RETURN    |
+-----+

```



v

```
+-----+
| WAIT .5 |
| Display |
| Err_mse$|
+-----+
| Terminal Output |
+-----+
|
+-----+
| WAIT .5 |
| Clear the |
| display line |
+-----+
| Terminal Output |
+-----+
|
+-----+
| RETURN |
+-----+
```

```

+-----+
|*****|
|SHUTDOWN|
|*****|
|This routine|
|allows the|
|user a clean|
|exit !!|
+-----+
|
+-----+
|OFF KEY|
|
|turn off all|
|assigned func-|
|tion keys|
+-----+
|
+-----+
|Printer is 1|
+-----+
|
+-----+
|Clear the|
|screen|
+-----+
|Terminal Output|
+-----/
|
+-----+
|Display|
|"PROGRAM|
|TERMINATED"|
+-----+
|Terminal Output|
+-----/
|
+-----+
|END|
+-----+

```

Section I-3



F85-Q3

Section I-3
DATA INPUT PROGRAM LISTING

LOCATION OBJECT CODE LINE SOURCE LINE

```

1 "6805"LIST
2 *      NRL Tracker Interface
3 *      Data Input to 6805 Program
4 *      April 2, 1984
5
6 *      Revised Sept 9, 1984      3:10 PM
7
8 PORTA <0000> EQU 0000H      LSByte output port
9 PORTB <0001> EQU 0001H      MSByte output port
10 PORTC <0002> EQU 0002H      Control port
11 PORTD <0003> EQU 0003H      Data input port
12 LIMIT <0011> EQU 0011H      Number of data bytes
13 TABLE <0010> EQU 0010H      Beginning of data table
14 IDATA <0008> EQU 0008H      Timer data register
15 TCONT <0009> EQU 0009H      Timer Control register
16
17 NAME "NRL_DATA"
18
19 ORG 0100H
20
21 INIT
22     LDA #0FFH      Initialize
23     STA DDRA        DDRA
24     LDA #005H      DDRA
25     STA DDRB        DDRA
26     LDA #006H      DDRC
27     CLR PORTA        Put zero's
28     CLR PORTB        in Ports
29     CLR PORTC
30     BSET 0,PORTC      Set PELG hi
31     BSET 2,PORTC      Set not_DAT_RDY hi
32     CLRX             Clear TABLE offset
33     CLI              Enable interrupts
34
35     NOP              Loop till interrupt
36     BRA WAIT
37
38 * Receive data from tracker and store in table.
39 * Interrupts are disabled by IRQ.
40
41 REC 0119 1F 02      BCLR      7,PORTC
42 READ 011B B6 03      LDA       PORTD
43      011D E7 10      STA       TABLE,X
44      011F 5C 44      INX
45      0120 1E 02      BSET      7,PORTC
46      0122 B6 03      LDA       PORTD
47      0124 E7 10      STA       TABLE,X
48      0126 5C 48      INX
49      0127 1F 02      BCLR      7,PORTC
50      0129 A3 11      CPX       #LIMIT
51      012B 22 04      BHI       TRANSFER
52      012D BF 7D      STX       07DH
53      012F 9A 53      CLI
54      0130 80 54      RTI
55
56 * Transmit data from the data table to the HP9816 computer
57

```

Mon, 25 Mar 1985, 16:10

LOCATION OBJECT CODE LINE SOURCE LINE

0131 15 02	58 TRANSFER	BCLR	2, PORTC	Set not_Data_ready LO
0133 5F	59	CLRX		
0134 0202 FD	60 LOOP1	BRSET	1, PORTC, LOOP1	Loop till PCTL=1
0137 E6 10	61 LOOP2	LDA	TABLE, X	Get LSbyte for out
0139 B7 00	62	STA	PORTA	
013B 5C	63	INX		Increment to next byte
013C E6 10	64	LDA	TABLE, X	Get MSbyte for out
013E B7 01	65	STA	PORTB	
0140 5C	66	INX		Increment to next byte
0141 11 02	67	BCLR	0, PORTC	Set PFLAG low
0143 0302 FD	68 LOOP3	BCLR	1, PORTC, LOOP3	Loop till PCTL=0
0146 14 02	69	BSET	2, PORTC	Set not_Data_ready H1
0148 10 02	70	BSET	0, PORTC	Set PFLAG high
014A 0202 FD	71 LOOP4	BRSET	1, PORTC, LOOP4	Loop till PCTL1
014D A3 11	72	CMPI	#LIMIT	
014F 23 E6	73	BLS	LOOP2	Loop back if more data
0151 11 02	74	BCLR	0, PORTC	Set PFLAG low
0153 A6 DF	75	LDA	#0DFH	Delay for 9816
0155 4A	76 LOOP5	DECA		
0156 26 FD	77	BNE	LOOP5	
0158 10 02	78	BSET	0, PORTC	Set PFLAG back high
015A 9A	79	CLI		Enable interrupts
015B 80	80	RTI		
	81			
0F38 00	82	ORG	0F38H	
	83	FCB	0000	
	84			
0FF8 01000119	85	ORG	0FF8H	
	86	FDB	INIT, REC, INIT, INIT	
	87			
	88	END		

Errors= 0

LINE#	SYMBOL	TYPE	REFERENCES
21	INIT	A	86,86,86
12	LIMIT	A	50,72
60	LOOP1	A	60
61	LOOP2	A	73
68	LOOP3	A	68
71	LOOP4	A	71
76	LOOP5	A	77
8	PORTA	A	26,62
9	PORTB	A	27,65
10	PORTC	A	28,29,30,41,45,49,58,60,67,68,69,70,71,74,78
11	PORTD	A	42,46
42	READ	A	
41	REC	A	86
13	TABLE	A	43,47,61,64
15	TCONT	A	
14	TDATA	A	
58	TRANSFER	A	51
34	WAIT	A	35

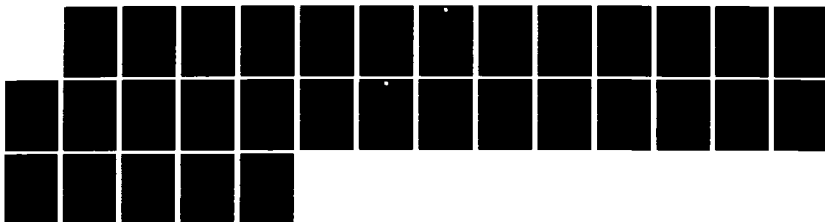
Section I-4

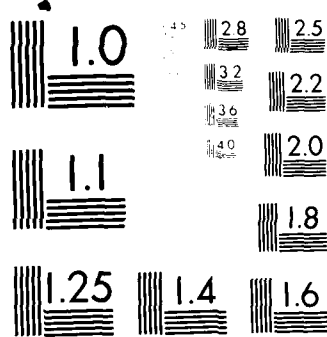


F85-03

Section I-4
COMMAND OUTPUT PROGRAM LISTING

AD-A162 778 RADIATION-HARD BREADBOARD STAR TRACKER ATTACHMENT 1(U) 2/2
BALL AEROSPACE SYSTEMS DIV BOULDER CO
M W HUBBARD ET AL SEP 85 BASD/F85-83-1
UNCLASSIFIED N00014-82-6(C)-2488 F/G 9/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

LOCATION OBJECT CODE LINE SOURCE LINE

```
1 "6805",LIST
2 *
3 *      NRL Tracker Interface
4 *      Command Output To Tracker
5      April 30, 1984
6 *
7      Revised Sept 10, 1984  3:00 PM
8
9      EQU      0000H      Control Port
10     EQU      0001H      Serial Output (bit 0)
11     EQU      0002H      LSByte Input Port
12     EQU      0003H      MSByte Input Port
13     EQU      0008H      Timer Data Register
14     EQU      0009H      Timer Control Register
15     EQU      0010H      Temp Reg For Command
16     EQU      0011H      Bit Counter
17     EQU      0012H
18     EQU      0013H
19     EQU      0014H
20
21     NAME      "NRL_CMD"
22
23     ORG      0100H
24
25     LDA      00D9H      Initialize
26     STA      0004H      DDRA
27
28     LDA      00FFH
29     CLR      0005H      DDRB
30     CLR      0006H      DDRC
31     CLR      PORTA      Put zero's
32     CLR      PORTB      in ports
33     CLR      PORTC
34     BSET     0,PORTA    Set PFLG2 hi
35     LDA      0077H      Disable TIMER IRQ's
36     STA      TCONT     and start timer clk
37
38     BRCLR    5,PORTA,START  Loop till CTL0-1
39                        9816 has a command
40
41     BRSET    1,PORTA,LOOP1  Loop till PCTL-1
42                        valid command
43
44     BCLR     0,PORTA      Set PFLG2 low
45                        command acknowledge
46
47     BRCLR    1,PORTA,PCTL1  Loop till PCTL-0
48                        command present
49
50     LDA      PORTC      Get command
51     STA      TREG      Save command
52     BSET     0,PORTA    Set PFLG hi
53                        data acknowledge
54
55     BRSET    1,PORTA,PCTL0  Loop till PCTL-1
56                        9816 acknowledges data ack
57
58     JSR      DELAY      Wait ~80 us
59     BCLR     0,PORTA    Set PFLG low
```

LOCATION OBJECT CODE LINE SOURCE LINE

```
58 ;
59
60 012F A6 05          LDA
61 0131 AE FF          LDX
62 0133 CD 01A3        JSR
63 0136 4A             DECA
64 0137 26 F8          BNE
65
66 0139 10 00          BSET
67 013B CD 0166        JSR
68 013E CD 0154        JSR
69 0141 CD 0166        JSR
70
71
72 0144 0500 FD        2,PORTA,LOOP6
73 0147 A6 C8          BRCLR
74 0149 B7 08          LDA
75 014B 0500 3E        STA
76 014E B6 08          BRCLR
77 0150 26 F9          LDA
78 0152 20 C2          BNE
79                      BRA
80 0154 0500 FD        2,PORTA,LOOP9
81 0157 A6 C8          BRCLR
82 0159 B7 08          LDA
83 015B 0500 04        STA
84 015E B6 08          BRCLR
85 0160 26 F9          LDA
86 0162 0500 FD        2,PORTA,LOOP11
87 0165 81             BNE
88
89 * Output subroutine
90
91 0166 A6 08          LDA
92 0168 B7 11          STA
93 016A B6 10          LDA
94 016C B7 01          STA
95 016E 16 00          BSET
96 0170 18 00          BSET
97 0172 AD 2D          BSR
98 0174 19 00          BCLR
99 0176 AD 29          BSR
100 0178 38 01          LSL
101 017A 3A 11          DEC
102 017C 26 F2          BNE
103
104 017E 17 00          BCLR
105 0180 A6 C8          LDA
106 0182 B7 08          STA
107 0184 0400 01        BRSET
108 0187 81             RTS
109
110 0188 B6 08          LDA
111 018A 26 F8          BNE
112
113 018C 1C 00          BSET
114 018E 0800 FB        BRCLR

remove data ack
Loop delay 4 times
Setup delay
for ~10 ms
Loop till A=0
Set PFLG h1
Go to OUTPUT 1
Ignore first mismatch
Go to OUTPUT 2

Wait till ACK h1
Preset timer counter
to 200d.
Test for error ACK=0
Test for timeout
Still waiting
Command OK, start over

Wait till ACK h1
Preset timer counter
to 200d.
Test for error ACK=0
Test for timeout
Still waiting
wait for it to go away

Get max bit count
Retrieve command
Put CMD in output port
Set CMD FLAG = 1
Set CMD CLK = 1
Delay
Set CMD CLK = 0
Delay
Move next bit into d7
Count shift
Not done - loop back

Set CMD FLAG = 0
Preset timer counter
to 200d.
Test for ACK
Acknowledge received

Test time out
More time - LOOPS

Set error bit = 1
Wait for 9816 to get
```

LOCATION OBJECT CODE LINE SOURCE LINE

```
0191 ID 00 115 error flg & ACK.  
0193 9C 116 Reset SP to $7F  
0194 A6 05 117  
0196 AE FF 118 Loop delay 4 times  
0198 CD 01A3 119 LOOP8 #05H  
019B 4A 120 LDX #0FFH  
019C 26 F8 121 JSR DELAY2  
019E CC 0100 122 DECA  
123 BNE  
124 * JMP LOOP8  
125 * INIT  
126  
127 * Delay overhead 16 cycles; delay loop time 8 cycles.  
128  
01A1 AE 08 129 DELAY #008H Start delay loop  
01A3 SA 130 DELAY2  
01A4 26 FD 131 BNE Loop till X=0  
01A6 81 132 RTS  
133  
01A7 AE 02 134 DEL #002H  
01A9 BF 14 135 LDX DCOUNT3  
01AB 5F 136 CLRX  
01AC BF 13 137 STX  
01AE BF 12 138 STX DCOUNT2  
01B0 3A 12 139 DEL1 DCOUNT1  
01B2 26 FC 140 DEC DEL1  
01B4 3A 13 141 DEC DCOUNT2  
01B6 26 F8 142 BNE DEL1  
01B8 3A 14 143 DEC DCOUNT3  
01BA 26 F4 144 BNE DEL1  
01BC 81 145 RTS  
146  
0F38 37 147 ORG Setup Mask Option  
148 FCB Register  
149  
0FF8 01000100 150 ORG Initialize Vectors  
151 INIT,INIT,INIT,INIT  
152  
153 END
```

Errors= 0

LINE# SYMBOL TYPE REFERENCES

15	COUNT	A	92,101
16	DCOUNT1	A	138,139
17	DCOUNT2	A	137,141
18	DCOUNT3	A	135,143
134	DEL	A	
139	DEL1	A	140,142,144
129	DELAY	A	56,97,99
130	DELAY2	A	62,120,131
113	ERROR	A	75,114
24	INIT	A	123,151,151,151,151
39	LOOP1	A	39
83	LOOP10	A	85
86	LOOP11	A	83,86
75	LOOP2	A	77
96	LOOP4	A	102
107	LOOP5	A	111
72	LOOP6	A	72
61	LOOP7	A	64
119	LOOP8	A	122
80	LOOP9	A	68,80
91	OUTPUT	A	67,69
53	PCTL0	A	53
45	PCTL1	A	45
8	PORTA	A	29,32,36,39,42,45,50,53,57,66,72,75,80,83,86,95,96,98,104,107,113,114,115
9	PORTB	A	30,94,100
10	PORTC	A	31,48
11	PORTD	A	
36	START	A	36,78
13	TCOUNT	A	34
12	TDATA	A	74,76,82,84,106,110
110	TIME	A	107
14	TREG	A	49,93

Section I-5



F85-03

Section I-5
PLOTXY PROGRAM LISTING

```

10! -----
20! - PROGRAM PLOTXY -
30! -----
40!
50! Written by Kris Parrish
60!
70 DIM Sd(100,12),Read_string$(20),Clear$(2),Disp_flag$(8),Disp_msg$(60)
80 DIM File_string$(7),Disk_string$(14),Off_flag$(8),On_flag$(8),Titles$(35)
90 DIM Plot_x(100),Plot_y(100),Test_data(100)
100 !
110 INTEGER I ! For loop counter
120 Data_count=0 ! Used for writing star data to file
130 Max_dat=20 ! Maximum # of data sets for a single
140 ! write to a data file.
150 Max_num_files=5 ! Maximum number of data files allowed
160 Nfiles=0 ! String designator for concatenation
170 ! of data file name, to be used for mult
180 ! data files
190 File_string$="FILE_"
200 Disk_string$=":HP82901,700,1"
210 Read_string$="FILE_0:HP89201,700,1"
220 Clear$=CHR$(255)&CHR$(75)
230 Inv_off=128 ! Code for video attributes OFF
240 Inv_on=129 ! Code for inverse video ON
250 !
260 Off_flag$="ON/"&CHR$(Inv_on)&"OFF"&CHR$(Inv_off)
270 On_flag$=CHR$(Inv_on)&"ON"&CHR$(Inv_off)&"/OFF"
280 !
290 Star1_flag=0 ! Plot star #1 OFF
300 Star2_flag=0 ! Plot star #2 OFF
310 Star3_flag=0 ! Plot star #3 OFF
320 Data_read=0 ! Show that a data file hasn't been read yet
330 Print_graph=0 ! Send current graph to printer OFF
340 X_minimum=9999 ! Initialize Min and Max values
350 X_maximum=-9999
360 Y_minimum=9999
370 Y_maximum=-9999
380 DEG ! Initialize graphics stuff...
390 GINIT
400 GRAPHICS ON
410 PRINTER IS 1
420 !
430 ! -----
440 ! Main_menu
450 ! -----
460 !
470 ! Define and display soft key functions
480 !
490 Main_menu: !
500 ON KEY 0 LABEL "Re-display",3 GOTO Redisp_screen
510 ON KEY 1 LABEL "Star #1",3 GOSUB Star_1
520 ON KEY 2 LABEL "Star #2",3 GOSUB Star_2
530 ON KEY 3 LABEL "Star #3",3 GOSUB Star_3
540 ON KEY 4 LABEL "Data file #",3 GOSUB Data_file
550 ON KEY 5 LABEL "X vs Y",3 GOSUB X_vs_y
560 ON KEY 6 LABEL "X vs Time",3 GOSUB X_vs_time
570 ON KEY 7 LABEL "Y vs Time",3 GOSUB Y_vs_time
580 ON KEY 8 LABEL "Dump Graph",3 GOSUB Print_graphics
590 ON KEY 9 LABEL "EXIT",3 GOTO Shutdown
600 OUTPUT 2;Clear$;

```

```

610 :
620 : *****
630 : *      Menu_loop      *
640 : *****
650 :
660 : Display menu options on the screen
670 :
680 Menu_loop: !
690 CONTROL 1:31.1
700 PRINT "PLOT PROGRAM MENU"
710 CONTROL 1:30.3
720 PRINT "k0 Redisplay Screen (Clear Graphics)"
730 CONTROL 1:30.4
740 PRINT "k1 Star #1 "
750 CONTROL 1:30.5
760 PRINT "k2 Star #2 "
770 CONTROL 1:30.6
780 PRINT "k3 Star #3 "
790 CONTROL 1:30.7
800 PRINT "k4 Data File # "
810 CONTROL 1:30.8
820 PRINT "k5 X vs Y "
830 CONTROL 1:30.9
840 PRINT "k6 X vs Time "
850 CONTROL 1:30.10
860 PRINT "k7 Y vs Time "
870 CONTROL 1:30.11
880 PRINT "k8 Dump Screen Graphics"
890 CONTROL 1:30.12
900 PRINT " to Printer"
910 CONTROL 1:30.13
920 PRINT "k9 Exit Program"
930 GOSUB Check_flags
940 GOTO Menu_loop
950 :
960 : *****
970 : *      Check_flags      *
980 : *****
990 :
1000: Check the status of the flags displayed
1010: on the screen.
1020:
1030 Check_flags: !
1040 CONTROL 1:42.4 ! Star #1
1050 SELECT Star1_flag
1060 CASE 0
1070 Disp_flag$=Off_flag$
1080 CASE 1
1090 Disp_flag$=On_flag$
1100 END SELECT
1110 PRINT Disp_flag$
1120 CONTROL 1:42.5 ! Star #2
1130 SELECT Star2_flag
1140 CASE 0
1150 Disp_flag$=Off_flag$
1160 CASE 1
1170 Disp_flag$=On_flag$
1180 END SELECT
1190 PRINT Disp_flag$
1200 CONTROL 1:42.6 ! Star #3

```

```

210 SELECT Star3_flag
220     CASE 0
230         Disp_flag$=Off_flag$
240     CASE 1
250         Disp_flag$=On_flag$
260 END SELECT
270 PRINT Disp_flag$
280
290 DATA FILE # ( Enabled-File read, Disabled- File not read)
300
310 CONTROL 1:46.7
320 IF Data_read=1 THEN
330     PRINT "[:Nfiles;"]
340 ELSE
350     PRINT "    Disabled- file not read"
360 END IF
370 RETURN
380
390 *****
400     >      Data_file      *
410 *****
420
430     Get user input as to which data file
440     to use.
450
460 Data_file: !
470     BEEP
480     INPUT "Please input file # > ".Nfiles
490     IF Nfiles>=0 AND Nfiles<=99 THEN Input_ok
500         BEEP
510         Disp_msg$="Invalid file # "&VAL$(Nfiles)&" Please, try again !"
520         GOSUB Out_msg
530         GOTO Data_file
540
550 *****
560     >      Input_ok      *
570 *****
580
590     Open and read data file, filling
600     the array with the star values.
610
620 Input_ok: !
630     ON ERROR GOTO Error_tst
640
650     Number$=VAL$(Nfiles)
660     File_string$[6]=Number$[1]
670     MASS STORAGE IS ":HP82901,700,1"
680     ASSIGN @Path_1 TO File_string$
690     DISP "Opening file > ":File_string$
700     Read_string$=File_string$&Disk_string$
710     ASSIGN @F_1 TO Read_string$
720     ENTER @F_1;Sd(*)
730     DISP "File read completed"
740     Data_read=1
750     ASSIGN @F_1 TO *
760     WAIT 1
770     DISP
780     OFF ERROR
790     RETURN
800

```

```

! Any file errors are taken
! care of in this routine...
! Set up for concat. of filename
! Create file name
! Designate rt drive for data
!
! Let user know what's going on
!
! Read in the data
! Let user know it is done
! Show Data has been read...
! Close the data file
! Delay so user can read message
! Clear display line
! Turn off error trapping
! Done!

```

```

1810! *****
1820! -      Error_tst      *
1830! *****
1840!
1850! This is used for testing of data file
1860! errors.
1870!
1880 Error_tst: !
1890     OFF ERROR
1900     BEEP
1910     IF ERRN=56 THEN
1920         DISP "File # ";Nfiles;" does not exist, press <CONT> to continue"
1930         PAUSE
1940     ELSE
1950         IF ERRN=80 THEN
1960             DISP "Disc not changed or NOT located in the right hand drive..."
1970             WAIT .5
1980             DISP "Press <CONT> after placing correct disc in right drive"
1990             PAUSE
2000         ELSE
2010             CONTROL 1;1.24
2020             PRINT "Unexpected error (";ERRN;") consult list of errors"
2030             PRINT "and correct problem... press <CONT> to try again!"
2040             PAUSE
2050         END IF
2060     END IF
2070     GOTO Data_file
2080!
2090! *****
2100! -      X_vs_y      *
2110! *****
2120!
2130! Plot X position vs Y position
2140!
2150 X_vs_y: !
2160     IF Data_read=0 THEN                ! A data file hasn't been read in yet
2170         Disp_msg$="You must first read in a data file before you can plot..."
2180         GOSUB Out_msg
2190         RETURN
2200     END IF
2210     OUTPUT 2:Clear$:
2220     FRAME
2230     WINDOW -5.270,-5.270
2240     AXES 2.2,-5,-5.5,5.5
2250     Npts=0
2260     FOR I=1 TO 100
2270         IF Sd(I,10)=0 AND Sd(I,11)=0 AND Sd(I,12)=0 THEN 2300
2280             Npts=Npts+1
2290     NEXT I
2300     IF Npts=0 THEN
2310         Npts=1
2320     END IF
2330
2340     LABEL THE X AXIS
2350
2360     X_maximum=260
2370     Step_size=20
2380     X_offset=10
2390     GOSUB Label_x
2400

```

```

2410 ! LABEL THE Y AXIS
2420 !
2430 Y_maximum=260
2440 Step_size=20
2450 Y_offset=12
2460 GOSUB Label_y
2470 !
2480 ! LABEL THE PLUT
2490 !
2500 Title$="X vs Y from "&File_string$
2510 Location_x=137
2520 Location_y=255
2530 GOSUB Label_plot
2540 !
2550 IF Star1_flag=0 THEN Plot_star_2
2560     FOR I=1 TO Npts
2570         Plot_x(I)=Sd(I,1)
2580         Plot_y(I)=Sd(I,2)
2590     NEXT I
2600     Line_type=1
2610     Line_label$="STAR #1"
2620     GOSUB Plot_star
2630 !
2640 ! *****
2650 ! *      Plot_star_2      *
2660 ! *****
2670 !
2680 Plot_star_2:
2690     IF Star2_flag=0 THEN Plot_star_3
2700     FOR I=1 TO Npts
2710         Plot_x(I)=Sd(I,4)
2720         Plot_y(I)=Sd(I,5)
2730     NEXT I
2740     Line_type=5
2750     Line_label$="STAR #2"
2760     GOSUB Plot_star
2770 !
2780 ! *****
2790 ! *      Plot_star_3      *
2800 ! *****
2810 !
2820 Plot_star_3:
2830     IF Star3_flag=0 THEN RETURN
2840     FOR I=1 TO Npts
2850         Plot_x(I)=Sd(I,7)
2860         Plot_y(I)=Sd(I,8)
2870     NEXT I
2880     Line_type=3
2890     Line_label$="STAR #3"
2900     GOSUB Plot_star
2910     RETURN
2920 !
2930 ! *****
2940 ! *      Star_1      *
2950 ! *****
2960 !
2970 ! Toggle star #1 flag
2980 !
2990 Star_1:
3000     Star1_flag=1-Star1_flag

```

```

3010 RETURN
3020
3030 *****
3040 *          Star_2          *
3050 *****
3060
3070 Toggle star #2 flag
3080
3090 Star_2:1
1100 Star2_flag=1-Star2_flag
3110 RETURN
3120
3130 *****
3140 *          Star_3          *
3150 *****
3160
3170 Toggle star #3 flag
3180
3190 Star_3:1
1200 Star3_flag=1-Star3_flag
1210 RETURN
1220
1230 *****
1240 *          X_vs_time       *
1250 *****
1260
1270 Plot X position vs time
1280
1290 X_vs_time:1
1300 IF Data_read=0 THEN
1310 Disp_msg$="You must first read in a data file before you can plot..."
1320 GOSUB Out_msg
1330 RETURN
1340 END IF
1350 INPUT "2:Clear$:"
1360 FRAME
1370 CONTROL 1:20,18
1380 PRINT CHR$(Inv_on)&"Determining MINIMUM and MAXIMUM values"&CHR$(Inv_off)
1390
1400 X_maximum=-9999 ! Initialize Maximum value
1410 X_minimum=9999 ! Initialize Minimum value
1420 Npts=0
1430 FOR I=1 TO 100
1440 IF Sd(I,10)=0 AND Sd(I,11)=0 AND Sd(I,12)=0 THEN 3490
1450 Npts=Npts+1
1460 Test_data(I)=Sd(I,12)+(60*Sd(I,11))
1470 IF X_maximum<Test_data(I) THEN X_maximum=Test_data(I)
1480 IF X_minimum>Test_data(I) THEN X_minimum=Test_data(I)
1490 NEXT I
1500 IF Npts=0 THEN
1510 Npts=1
1520 X_maximum=60
1530 X_minimum=0
1540 END IF
1550 X_maximum=X_maximum+20
1560 WINDOW X_minimum,X_maximum,-5,270
1570 AXES 1,2,X_minimum,-5,5,5,5
1580 Label the X axis
1590

```

```

3600 Step_size=10
3610 X_offset=15
3620 GOSUB Label_x
3630 !
3640 ! Label the Y axis
3650 !
3660 Y_maximum=260
3670 Step_size=20
3680 Y_offset=X_minimum+15
3690 GOSUB Label_y
3700 !
3710 ! Label the plot
3720 !
3730 Title$="X vs TIME from "&File_string$
3740 Location_x=X_minimum+125
3750 Location_y=255
3760 GOSUB Label_plot
3770 !
3780 ! START PLOTTING...
3790 !
3800 CONTROL 1:20.18
3810 PRINT "
3820 IF Star1_flag=0 THEN Plot_x_star2
3830 FOR I=1 TO Npts
3840 Plot_x(I)=Test_data(I)
3850 Plot_y(I)=Sd(I,1)
3860 NEXT I
3870 Line_type=1
3880 Line_label$="STAR #1"
3890 GOSUB Plot_star
3900 !
3910 ! *****
3920 ! * Plot_x_star2 *
3930 ! *****
3940 !
3950 Plot_x_star2:!
3960 IF Star2_flag=0 THEN Plot_x_star3
3970 FOR I=1 TO Npts
3980 Plot_x(I)=Test_data(I)
3990 Plot_y(I)=Sd(I,4)
4000 NEXT I
4010 Line_type=5
4020 Line_label$="STAR #2"
4030 GOSUB Plot_star
4040 !
4050 ! *****
4060 ! * Plot_x_star3 *
4070 ! *****
4080 !
4090 Plot_x_star3:!
4100 IF Star3_flag=0 THEN RETURN
4110 FOR I=1 TO Npts
4120 Plot_x(I)=Test_data(I)
4130 Plot_y(I)=Sd(I,7)
4140 NEXT I
4150 Line_type=3
4160 Line_label$="STAR #3"
4170 GOSUB Plot_star
4180 RETURN
4190 !

```

```

4200!      *****
4210!      *      Y_vs_time      *
4220!      *****
4230!
4240!      Plot Y position vs time
4250!
4260 Y_vs_time:!
4270      IF Data_read=0 THEN
4280          Disp_msg$="You must first read in a data file before you can plot..."
4290          GOSUB Out_msg
4300      END IF
4310      OUTPUT 2;Clear$:
4320      FRAME
4330      CONTROL 1;20,18
4340      PRINT CHR$(Inv_on)&"Determining MINIMUM and MAXIMUM values"&CHR$(Inv_off
)
4350      X_maximum=-9999
4360      X_minimum=9999
4370      Npts=0
4380      FOR I=1 TO 100
4390          IF Sd(I,10)=0 AND Sd(I,11)=0 AND Sd(I,12)=0 THEN 4450
4400              Npts=Npts+1
4410              Test_data(I)=Sd(I,12)+(60*Sd(I,11))
4420              IF X_maximum<Test_data(I) THEN X_maximum=Test_data(I)
4430              IF X_minimum>Test_data(I) THEN X_minimum=Test_data(I)
4440          NEXT I
4450      IF Npts=0 THEN
4460          Npts=1
4470          X_maximum=60
4480          X_minimum=0
4490      END IF
4500      X_maximum=X_maximum+20
4510      WINDOW X_minimum,X_maximum,-5,270
4520      AXES 2,2,X_minimum,-5,5,5,5
4530 !
4540 ! Label the X axis
4550 !
4560      Step_size=10
4570      X_offset=15
4580      GOSUB Label_x
4590 !
4600 ! Label the Y axis
4610 !
4620      Y_maximum=260
4630      Step_size=20
4640      Y_offset=X_minimum+15
4650      GOSUB Label_y
4660 !
4670 ! Label the plot
4680 !
4690      Title$="Y vs TIME from "&File_string$
4700      Location_x=X_minimum+125
4710      Location_y=255
4720      GOSUB Label_plot
4730 !
4740 ! START PLOTTING Y vs TIME...
4750 !
4760      CONTROL 1;20,18
4770      PRINT "
4780      IF Star1_flag=0 THEN Plot_y_star2

```

```

4790     FOR I=1 TO Npts
4800         Plot_x(I)=Test_data(I)
4810         Plot_y(I)=Sd(I,2)
4820     NEXT I
4830     Line_type=1
4840     Line_label$="STAR #1"
4850     GOSUB Plot_star
4860!
4870! *****
4880! *      Plot_y_star2      *
4890! *****
4900!
4910 Plot_y_star2:~
4920     IF Star2_flag=0 THEN Plot_y_star3
4930     FOR I=1 TO Npts
4940         Plot_x(I)=Test_data(I)
4950         Plot_y(I)=Sd(I,5)
4960     NEXT I
4970     Line_type=5
4980     Line_label$="STAR #2"
4990     GOSUB Plot_star
5000!
5010! *****
5020! *      Plot_y_star3      *
5030! *****
5040!
5050 Plot_y_star3:~
5060     IF Star3_flag=0 THEN RETURN
5070     FOR I=1 TO Npts
5080         Plot_x(I)=Test_data(I)
5090         Plot_y(I)=Sd(I,8)
5100     NEXT I
5110     Line_type=3
5120     Line_label$="STAR #3"
5130     GOSUB Plot_star
5140     RETURN
5150!
5160! *****
5170! *      Print_graphics     *
5180! *****
5190!
5200 Print_graphics:~
5210     DUMP GRAPHICS
5220     PRINTER IS 701
5230     PRINT
5240     PRINT
5250     PRINTER IS 1
5260     GCLEAR
5270     RETURN
5290!
5290! *****
5300! *      Label_x            *
5310! *****
5320!
5330 Label_x:~
5340     LDRG 6
5350     CSIZE 3..7
5360     LDIR 90
5370     FOR I=0 TO X_maximum STEP Step_size
5380         MOVE I,X_offset

```

```

5390      LABEL I
5400      NEXT I
5410      RETURN
5420!
5430!      *****
5440!      *          Label_y          *
5450!      *****
5460!
5470 Label_y:~
5480      LDIR 0
5490      CSIZE 2.6,.6
5500      LORG 8
5510      FOR I=0 TO Y_maximum STEP Step_size
5520          MOVE Y_offset,I
5530          LABEL I
5540      NEXT I
5550      RETURN
5560!
5570!      *****
5580!      *          Label_plot        *
5590!      *****
5600!
5610 Label_plot:~
5620      CSIZE 7,.6
5630      LORG 5
5640      LDIR 0
5650      MOVE Location_x,Location_y
5660      LABEL Title$
5670      RETURN
5680!
5690!      *****
5700!      *          Plot_star         *
5710!      *****
5720!
5730 Plot_star:~
5740      CSIZE 2.5,1.5
5750      LINE TYPE Line_type
5760      MOVE Plot_x(1),Plot_y(1)
5770      FOR I=1 TO Npts
5780          DRAW Plot_x(I),Plot_y(I)
5790      NEXT I
5800      LINE TYPE 1
5810      IF Plot_y(100)=0 THEN Plot_y(100)=25
5820      IF Plot_x(100)=0 THEN Plot_x(100)=35
5830      MOVE Plot_x(100)-5,Plot_y(100)+5
5840      LABEL Line_label$
5850      MOVE 0,0
5860      RETURN
5870!
5880!      *****
5890!      *          Redisp_screen     *
5900!      *****
5910!
5920 Redisp_screen:~
5930      OUTPUT 2:Clear$;
5940      GCLEAR
5950      GOTO Menu_loop
5960!
5970!      *****
5980!      *          Out_msg           *

```

```
5990! *****
6000!
6010 Out_msg:~
6020 BEEP
6030 DISP Disp_msg$
6040 WAIT 2
6050 DISP
6060 RETURN
6070!
6080! *****
6090! * Shutdown *
6100! *****
6110!
6120 Shutdown:~
6130 OFF KEY
6140 OUTPUT 2:Clear$;
6150 GCLEAR
6160 PRINT "PLOT PROGRAM TERMINATED..."
6170 END
```

Section I-6



F85-03

Section I-6
ERROR MESSAGES

Error Messages

- 1 Missing ROM or configuration error. Loading a program or binary file that is not compatible with the language system. For example, trying to load the 1.0 PHYREC Binary into a 2.0 system, or loading a program containing 2.0 keywords into a 1.0 system.
- 2 Memory overflow. If you get this error while loading a file, the program is too large for the computer's memory. If the program loads, but you get this error when you press RUN, then the overflow was caused by the variable declarations. Either way, you need to modify the program or add more read/write memory.
- 3 Line not found in current context. Could be a GOTO or GOSUB that references a non-existent (or deleted) line, or an EDIT command that refers to a non-existent line label.
- 4 Improper RETURN. Executing a RETURN statement without previously executing an appropriate GOSUB or function call. Also, a RETURN statement in a user-defined function with no value specified.
- 5 Improper context terminator. You forgot to put an END statement in the program. Also applies to SUBEND and FNEND.
- 6 Improper FOR...NEXT matching. Executing a NEXT statement without previously executing the matching FOR statement. Indicates improper nesting or overlapping of the loops.
- 7 Undefined function or subprogram. Attempt to call a SUB or user-defined function that is not in memory. Look out for program lines that assumed an optional CALL.
- 8 Improper parameter matching. A type mismatch between a pass parameter and a formal parameter of a subprogram.
- 9 Improper number of parameters. Passing either too few or too many parameters to a subprogram. Applies only to non-optional parameters.
- 10 String type required. Attempting to return a numeric from a user-defined string function.
- 11 Numeric type required. Attempting to return a string from a user-defined numeric function.
- 12 Attempt to redeclare variable. Including the same variable name twice in declarative statements such as DIM or INTEGER.
- 13 Array dimensions not specified. Using the (*) symbol after a variable name when that variable has never been declared as an array.
- 14 OPTION BASE not allowed here. The OPTION BASE statement must appear before any declarative statements such as DIM or INTEGER. Only one OPTION BASE statement is allowed in one context.

- 15 Invalid bounds. Attempt to declare an array with more than 32 767 elements or with upper bound less than lower bound.
- 16 Improper or inconsistent dimensions. Using the wrong number of subscripts when referencing an array element.
- 17 Subscript out of range. A subscript in an array reference is outside the current bounds of the array.
- 18 String overflow or substring error. String overflow is an attempt to put too many characters into a string (exceeding dimensioned length). This can happen in an assignment, an ENTER, an INPUT, or a READ. A substring error is an attempted violation of the rules for substrings (see Chapter 5). Watch out for null strings where you weren't expecting them.
- 19 Improper value or out of range. A value is too large or too small. Applies to items found in a variety of statements. Often occurs when the number builder overflows (or underflows) during an I/O operation.
- 20 INTEGER overflow. An assignment or result exceeds the range allowed for INTEGER variables. Must be -32 768 thru 32 767.
- 22 REAL overflow. An assignment or result exceeds the range allowed for REAL variables. (See Chapter 4.)
- 24 Trig argument too large for accurate evaluation. Out-of-range argument for a function such as TAN or LDIR.
- 25 Magnitude of ASN or ACS argument is greater than 1. Arguments to these functions must be in the range -1 thru +1.
- 26 Zero to non-positive power. Exponentiation error.
- 27 Negative base to non-integer power. Exponentiation error.
- 28 LOG or LGT of a non-positive number.
- 29 Illegal floating point number. Does not occur as a result of any calculations, but is possible when a FORMAT OFF I/O operation fills a REAL variable with something other than a REAL number.
- 30 SQR of a negative number.
- 31 Division (or MOD) by zero.
- 32 String does not represent a valid number. Attempt to use "non-numeric" characters as an argument for VAL, data for a READ, or in response to an INPUT statement requesting a number.
- 33 Improper argument for NUM or RPT\$. Null string not allowed.
- 34 Referenced line not an IMAGE statement. A USING clause contains a line identifier, and the line referred to is not an IMAGE statement.
- 35 Improper image. See IMAGE or the appropriate keyword in the *BASIC Language Reference*.
- 36 Out of data in READ. A READ statement is expecting more data than is available in the referenced DATA statements. Check for deleted lines, proper OPTION BASE, proper use of RESTORE, or typing errors.

- 38 TAB or TABXY not allowed here. The tab functions are not allowed in statements that contain a USING clause. TABXY is allowed only in a PRINT statement.
- 40 Improper REN, COPYLINES, or MOVELINES command. Line numbers must be whole numbers from 1 to 32 766. This may also result from a COPYLINES or MOVELINES statement whose destination line numbers lie within the source range.
- 41 First line number greater than second line number. Parameters out of order in a statement like SAVE, LIST, or DEL.
- 43 Matrix must be square. The MAT functions: IDN, INV, and DET require the array to have equal numbers of rows and columns.
- 44 Result cannot be an operand. Attempt to use a matrix as both result and argument in a MAT TRN or matrix multiplication.
- 46 Attempting a SAVE when there is no program in memory or a STORE BIN when there are no binary programs in memory.
- 47 COM declarations are inconsistent or incorrect. Includes such things as mismatched dimensions, unspecified dimensions, and blank COM occurring for the first time in a subprogram.
- 49 Branch destination not found. A statement such as ON ERROR or ON KEY refers to a line that does not exist. Branch destinations must be in the same context as the ON...statement.
- 51 File not currently assigned. Attempting an ON/OFF END statement with an unassigned I/O path name.
- 52 Improper mass storage unit specifier. The characters used for a msus do not form a valid specifier. This could be a missing colon, too many parameters, illegal characters, etc.
- 53 Improper file name. File names are limited to 10 characters. Foreign characters are allowed, but punctuation is not.
- 54 Duplicate file name. The specified file name already exists in directory. It is illegal to have two files with the same name on one volume.
- 55 Directory overflow. Although there may be room on the media for the file, there is no room in the directory for another file name. Discs initialized by BASIC have room for over 100 entries in the directory, but other systems may make a directory of a different size.
- 56 File name is undefined. The specified file name does not exist in the directory. Check the contents of the disc with a CAT command.
- 58 Improper file type. Many mass storage operations are limited to certain file types. For example, LOAD is limited to PROG files and ASSIGN is limited to ASCII and BDAT files.
- 59 End of file or buffer found. For files: No data left when reading a file, or no space left when writing a file. For buffers: No data left for an ENTER, or no buffer space left for an OUTPUT. Also, WORD-mode TRANSFER terminated with odd number of bytes.
- 60 End of record found in random mode. Attempt to ENTER a field that is larger than a defined record.
- 62 Protect code violation. Failure to specify the protect code of a protected file, or attempting to protect a file of the wrong type.
- 64 Mass storage media overflow. There is not enough contiguous free space for the specified file size. The disc is full.

- 66 INITIALIZE failed. Too many bad tracks found. The disc is defective, damaged, or dirty.
- 67 Illegal mass storage parameter. A mass storage statement contains a parameter that is out of range, such as a negative record number or an out of range number of records.
- 68 Syntax error occurred during GET. One or more lines in the file could not be stored as valid program lines. The offending lines are usually listed on the system printer. Also occurs if the first line in the file does not start with a valid line number.
- 72 Disc controller not found or bad controller address. The msus contains an improper device selector, or no external disc is connected.
- 73 Improper device type in mass storage unit specifier. The msus has the correct general form, but the characters used for a device type are not recognized.
- 76 Incorrect unit number in mass storage unit specifier. The msus contains a unit number that does not exist on the specified device.
- 77 Attempt to purge an open file. The specified file is assigned to an I/O path name which has not been closed.
- 78 Invalid mass storage volume label. Usually indicates that the media has not been initialized on a compatible system. Could also be a bad disc.
- 79 File open on target device. Attempt to copy an entire volume with a file open on the destination disc.
- 80 Disc changed or not in drive. Either there is no disc in the drive or the drive door was opened while a file was assigned.
- 81 Mass storage hardware failure. Also occurs when the disc is pinched and not turning. Try reinserting the disc.
- 82 Mass storage unit not present. Hardware problem or an attempt to access a left-hand drive on the Model 26.
- 83 Write protected. Attempting to write to a write_protected disc. This includes many operations such as PURGE, INITIALIZE, CREATE, SAVE, OUTPUT, etc.
- 84 Record not found. Usually indicates that the media has not been initialized.
- 85 Media not initialized. (Usually not produced by the internal drive.)
- 87 Record address error. Usually indicates a problem with the media.
- 88 Read data error. The media is physically or magnetically damaged, and the data cannot be read.
- 89 Checkread error. An error was detected when reading the data just written. The media is probably damaged.
- 90 Mass storage system error. Usually a problem with the hardware or the media.
- 100 Numeric IMAGE for string item.
- 101 String IMAGE for numeric item.
- 102 Numeric field specifier is too large. Specifying more than 160 characters in a numeric field

- 103 Item has no corresponding IMAGE. The image specifier has no fields that are used for item processing. Specifiers such as * X / are not used to process the data for the item list. Item-processing specifiers include things like K D B A.
- 105 Numeric IMAGE field too small. Not enough characters are specified to represent the number
- 106 IMAGE exponent field too small. Not enough exponent characters are specified to represent the number.
- 107 IMAGE sign specifier missing. Not enough characters are specified to represent the number. Number would fit except for the minus sign.
- 117 Too many nested structures. The nesting level is too deep for such structures as FOR, SELECT, IF, LOOP, etc.
- 118 Too many structures in context. Refers to such structures as FOR NEXT, IF THEN ELSE, SELECT, CASE, WHILE, etc.
- 120 Not allowed while program running. The program must be stopped before you can execute this command.
- 121 Line not in main program. The run line specified in a LOAD or GET is not in the main context
- 122 Program is not continuable. The program is in the stopped state, not the paused state. CONT is allowed only in the paused state.
- 126 Quote mark in unquoted string. Quote marks must be used in pairs.
- 127 Statements which affect the knob mode are out of order.
- 128 Line too long during GET.
- 131 Unrecognized non-ASCII keycode. An output to the keyboard contained a CHR\$(255) followed by an illegal byte.
- 132 Keycode buffer overflow. Trying to send too many characters to the keyboard buffer with an OUTPUT 2 statement.
- 133 DELSUB of non-existent or busy subprogram.
- 134 Improper SCRATCH statement
- 135 READIO/WRITEIO to nonexistent memory location.
- 136 REAL underflow. The input or result is closer to zero than 10^{-308} (approximately).
- 140 Too many symbols in the program. Symbols are variable names, I/O path names, COM block names, subprogram names, and line identifiers.
- 141 Variable cannot be allocated. It is already allocated.
- 142 Variable not allocated. Attempt to DEALLOCATE a variable that was not allocated
- 143 Reference to missing OPTIONAL parameter. The subprogram is trying to use an optional parameter that didn't have any value passed to it. Use NPAR to check the number of passed parameters.
- 145 May not build COM at this time. Attempt to add or change COM when a program is running. For example, a program does a LOADSUB and the COM in the new subprogram does not match existing COM.

- 146 Duplicate line label in context. There cannot be two lines with the same line label in one context.
- 150 Illegal interface select code or device selector. Value out of range.
- 152 Parity error
- 153 Insufficient data for ENTER. A statement terminator was received before the variable list was satisfied.
- 154 String greater than 32 767 bytes in ENTER.
- 155 Improper interface register number. Value out of range or negative.
- 156 Illegal expression type in list. For example, trying to ENTER into a constant.
- 157 No ENTER terminator found. The variable list has been satisfied, but no statement terminator was received in the next 256 characters. The * specifier allows the statement to terminate when the last item is satisfied.
- 158 Improper image specifier or nesting images more than 8 deep. The characters used for an image specifier are improper or in an improper order
- 159 Numeric data not received. When entering characters for a numeric field, an item terminator was encountered before any "numeric" characters were received.
- 160 Attempt to enter more than 32 767 digits into one number.
- 163 Interface not present. The intended interface is not present, set to a different select code, or is malfunctioning.
- 164 Illegal BYTE WORD operation. Attempt to ASSIGN with the WORD attribute to a non-word device.
- 165 Image specifier greater than dimensioned string length.
- 167 Interface status error. Exact meaning depends upon the interface type. With HP-IB, this can happen when a non-controller operation by the computer is aborted by the bus.
- 168 Device timeout occurred and the ON TIMEOUT branch could not be taken.
- 170 I/O operation not allowed. The I/O statement has the proper form, but its operation is not defined for the specified device. For example, using an HP-IB statement on a non-HP-IB interface or directing a LIST to the keyboard.
- 171 Illegal I/O addressing sequence. The secondary addressing in a device selector is improper or primary address too large for specified device.
- 172 Peripheral error. PSTS line is false. If used, this means that the peripheral device is down. If PSTS is not being used, this error can be suppressed by using control register 2 of the GPIO
- 173 Active or system controller required. The HP-IB is not active controller and needs to be for the specified operation
- 174 Nested I/O prohibited. An I/O statement contains a user-defined function. Both the original statement and the function are trying to access the same file or device
- 177 Undefined I/O path name. Attempting to use an I/O path name that is not assigned to a device or file.
- 178 Trailing punctuation in ENTER. The trailing comma or semicolon that is sometimes used at the end of OUTPUT statements is not allowed at the end of ENTER statements

- 301 Cannot do while connected.
- 303 Not allowed when trace active.
- 304 Too many characters without terminator.
- 306 Interface card failure. The datacomm card has failed self-test.
- 310 Not connected.
- 313 USART receive buffer overflow. Overrun error detected. Interface card is unable to keep up with incoming data rate. Data has been lost.
- 314 Receive buffer overflow. Program is not accepting data fast enough to keep up with incoming data rate. Data has been lost.
- 315 Missing data transmit clock. A transmit timeout has occurred because a missing data clock prevented the card from transmitting. The card has disconnected from the line.
- 316 CTS false too long. The interface card was unable to transmit for a predetermined period of time because Clear-To-Send was false on a half-duplex line. The card has disconnected from the line.
- 317 Lost carrier disconnect. Data Set Ready (DSR) or Data Carrier Detect (if full duplex) went inactive for too long.
- 318 No activity disconnect. The card has disconnected from the line because no data was transmitted or received for a predetermined length of time.
- 319 Connection not established. Data Set Ready or Data Carrier Detect (if full duplex) did not become active within a predetermined length of time.
- 324 Card trace buffer overflow.
- 325 Illegal databits parity combination. Attempting to program 8 bits-per-character and a parity of "1" or "0".
- 326 Register address out of range. A control or status register access was attempted to a non-existent register.
- 327 Register value out of range. Attempting to place an illegal value in a control register.
- 328 USART Transmit underrun.
- 330 User-defined LEXICAL ORDER IS table size exceeds array size.
- 331 Repeated value in pointer. A MAT REORDER vector has repeated subscripts. This error is not always caught.
- 332 Non-existent dimension given. Attempt to specify a non-existent dimension in a MAT REORDER operation.
- 333 Improper subscript in pointer. A MAT REORDER vector specifies a non-existent subscript.
- 334 Pointer size is not equal to the number of records. A MAT REORDER vector has a different number of elements than the specified dimension of the array.
- 335 Pointer is not a vector. Only single-dimension arrays (vectors) can be used as the pointer in a MAT REORDER or a MAT SORT statement.
- 337 Substring key is out-of-range. The specified substring range of the sort key exceeds the dimensioned length of the elements in the array.

- 338 Key subscript out-of-range. Attempt to specify a subscript in a sort key outside the current bounds of the array.
- 340 Mode table too long. User-defined LEXICAL ORDER IS mode table contains more than 63 entries.
- 341 Improper mode indicator. User-defined LEXICAL ORDER IS table contains an illegal combination of mode type and mode pointer.
- 342 Not a single-dimension integer array. User-defined LEXICAL ORDER IS mode table must be a single-dimension array of type INTEGER.
- 343 Mode pointer is out of range. User-defined LEXICAL ORDER IS table has a mode pointer greater than the existing mode table size.
- 344 1 for 2 list empty or too long. A user-defined LEXICAL ORDER IS table contains an entry indicating an improper number of 1 for 2 secondaries.
- 345 CASE expression type mismatch. The SELECT statement and its CASE statements must refer to the same general type, numeric or string.
- 346 INDENT parameter out-of-range. The parameters must be in the range: 0 thru eight characters less than the screen width.
- 347 Structures improperly matched. There is not a corresponding number of structure beginnings and endings. Usually means that you forgot a statement such as END IF, NEXT, END SELECT, etc.
- 349 CSUB has been modified. A contiguous block of compiled subroutines has been modified since it was loaded. A single module that shows as multiple CSUB statements has been altered because program lines were inserted or deleted.
- 353 Data link failure.
- 401 Bad system function argument. An invalid argument was given to a time, date, base conversion, or SYSTEM\$ function.
- 403 Copy failed: program modification incomplete. An error occurred during a COPYLINES or MOVE LINES resulting in an incomplete operation. Some lines may not have been copied or moved.
- 427 Priority may not be lowered.
- 471 TRANSFER not supported by the interface.
- 488 DMA hardware required. HP 9885 disc drive requires a DMA card or is malfunctioning.
- 511 The result array in a MAT INV must be of type REAL.
- 600 Attribute cannot be modified. The WORD/BYTE mode cannot be changed after assigning the I/O path name.
- 601 Improper CONVERT lifetime. When the CONVERT attribute is included in the assignment of an I/O path name, the name of a string variable containing the conversion is also specified. The conversion string must exist as long as the I/O path name is valid.
- 602 Improper BUFFER lifetime. The variable designated as a buffer during an I/O path name assignment must exist as long as the I/O path name is valid.

- 603 Variable was not declared as a BUFFER. Attempt to assign a variable as a buffer without first declaring the variable as a BUFFER.
- 604 Bad source or destination for a TRANSFER statement. Transfers are not allowed to the CRT, keyboard, or tape backup on CS80 drives. Buffer to buffer or device to device transfers are not allowed.
- 605 BDAT file type required. Only BDAT files can be used in a TRANSFER operation.
- 606 Improper TRANSFER parameters. Conflicting or invalid TRANSFER parameters were specified, such as RECORDS without and EOR clause, or DELIM with an outbound TRANSFER.
- 607 Inconsistent attributes. Such as CONVERT or PARITY with FORMAT OFF.
- 609 IVAL or DVAL result too large. Attempt to convert a binary, octal, decimal, or hexadecimal string into a value outside the range of the function.
- 612 BUFFER pointers in use. Attempt to change one or more buffer pointers while a TRANSFER is in progress.
- 700 Improper plotter specifier. The characters used as a plotter specifier are not recognized. May be misspelled or contain illegal characters.
- 702 CRT graphics hardware missing. Hardware problem.
- 704 Upper bound not greater than lower bound. Applies to $P2 \leq P1$ or VIEWPORT upper bound and CLIP limits.
- 705 VIEWPORT or CLIP beyond hard clip limits.
- 708 Device not initialized.
- 900 Undefined typing aid key.
- 901 Typing aid memory overflow.
- 902 Must delete entire context. Attempt to delete a SUB or DEF FN statement without deleting its entire context. Easiest way to delete is with DELSUB.
- 903 No room to renumber. While EDIT mode was renumbering during an insert, all available line numbers were used between insert location and end of program.
- 904 Null FIND or CHANGE string.
- 905 CHANGE would produce a line too long for the system. Maximum line length is 100 characters for the Model 26 and 160 characters for the Models 16 and 36.
- 906 SUB or DEF FN not allowed here. Attempt to insert a SUB or DEF FN statement into the middle of a context. Subprograms must be appended at the end.
- 909 May not replace SUB or DEF FN. Similar to deleting a SUB or DEF FN.
- 910 Identifier not found in this context. The keyboard-specified variable does not already exist in the program. Variables cannot be created from the keyboard; they must be created by running a program.
- 911 Improper I/O list
- 920 Numeric constant not allowed.

- 921 Numeric identifier not allowed.
- 922 Numeric array element not allowed.
- 923 Numeric expression not allowed.
- 924 Quoted string not allowed.
- 925 String identifier not allowed.
- 926 String array element not allowed.
- 927 Substring not allowed.
- 928 String expression not allowed.
- 929 I/O path name not allowed.
- 930 Numeric array not allowed.
- 931 String array not allowed.
- 932 Excess keys specified. A sort key was specified following a key which specified the entire record.
- 935 Identifier is too long: 15 characters maximum.
- 936 Unrecognized character. Attempt to store a program line containing an improper name or illegal character.
- 937 Invalid OPTION BASE. Only 0 and 1 are allowed.
- 939 OPTIONAL appears twice. A parameter list may have only one OPTIONAL keyword. All parameters listed before it are required, all listed after it are optional.
- 940 Duplicate formal parameter name.
- 942 Invalid I/O path name. The characters after the @ are not a valid name. Names must start with a letter.
- 943 Invalid function name. The characters after the FN are not a valid name. Names must start with a letter.
- 946 Dimensions are inconsistent with previous declaration. The references to an array contain a different number of subscripts at different places in the program.
- 947 Invalid array bounds. Value out of range, or more than 32 767 elements specified.
- 948 Multiple assignment prohibited. You cannot assign the same value to multiple variables by stating X=Y=Z=0. A separate assignment must be made for each variable.
- 949 This symbol not allowed here. This is the general "syntax error" message. The statement you typed contains elements that don't belong together, are in the wrong order, or are misspelled.
- 950 Must be a positive integer.
- 951 Incomplete statement. This keyword must be followed by other items to make a valid statement.
- 961 CASE expression type mismatch. The CASE line contains items that are not the same general type, numeric or string.

- 962 Programmable only: cannot be executed from the keyboard.
- 963 Command only: cannot be stored as a program line.
- 977 Statement is too complex. Contains too many operators and functions. Break the expression down so that it is performed by two or more program lines.
- 980 Too many symbols in this context. Symbols include variable names, I/O path names, COM block names, subprogram names, and line identifiers.
- 982 Too many subscripts: maximum of six dimensions allowed.
- 983 Wrong type or number of parameters. An improper parameter list for a machine-resident function.
- 985 Invalid quoted string.
- 987 Invalid line number: must be a whole number 1 thru 32 766.

END

FILMED

1-86

DTIC