Attachment I
Final Report for
DIATION-HARD BREADBOARD STAR TRACKER
F85-03 September 1985
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
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>Star Tracker Control Program Listing</td>
<td>1-1</td>
</tr>
<tr>
<td>1-2</td>
<td>Flowchart of the Star Tracker Control Program</td>
<td>2-1</td>
</tr>
<tr>
<td>1-3</td>
<td>Data Input Program Listing</td>
<td>3-1</td>
</tr>
<tr>
<td>1-4</td>
<td>Command Output Program Listing</td>
<td>4-1</td>
</tr>
<tr>
<td>1-5</td>
<td>PLOTXY Program Listing</td>
<td>5-1</td>
</tr>
<tr>
<td>1-6</td>
<td>Error Messages</td>
<td>6-1</td>
</tr>
</tbody>
</table>
Section I-1
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.

DIM Clear$(2),Off_flag$(8),On_flag$(8),Disp_flag$(8),Auto_flag$(13)
DIM Auto_disp_flag$(13),Star_data(100,12),Read_string$(20)
DIM Send_msg$(35),File_string$(7),Disk_string$(14),Ctable(15)
DIM Err_msg$(30)

Initialize variables and flags *

For use w/ the interface box
0 is with, 1 is without
Self test ON/OFF to OFF
Star #1 enabled  
Star #2 enabled  
Star #3 enabled  
Current star  
Current X position  
Current Y position  
Adaptive Rate ON/OFF to OFF, 1 is ON  
Acquisition AUTO/MANUAL to AUTO  
0 is MANUAL, 1 is AUTO  
Acquisition type is FULL FOV,  
0 is OFF, 1 is FULL FOV, 2 is FULL EDGE  
3 is VECTORED EDGE  
Drop star criteria to 2 tries  
Take data ON/OFF to OFF  
Used for writing star data to file  
Maximum number of data sets per file  
Initialize file counter to zero  
Maximum number of data files  
For creating sequential data file names  
String used to create data file name  
Variable used to time data storage  
Terminal video OFF command  
Terminal inverse video command  
Set up priority for ON KEY commands  
Command to be sent to interface (0 No-op)  
Used to determine where errors come from  
Variable for control register  
Define CTRL$=0 (inverted)
560 Data_toggle$=CHR$(92)  \* Display character to show update interval
570 Clear$=CHR$(255)&CHR$(75)  \* Clear screen command
580 Auto_flag$="ON/OFF/"&CHR$(Inv_on)&"AUTO"&CHR$(Inv_off)
590 Off_flag$="ON/"&CHR$(Inv_on)&"OFF"&CHR$(Inv_off)
600 On_flag$=CHR$(Inv_on)&"ON"&CHR$(Inv_off)&"OFF"
610 GOSUB Set_up_table  \* Set up command values in CTBLE
620 OFF KEY
630
640 Menu_start:
650 Prior=Prior+1
660 ON KEY 0 LABEL "Redisplay",Prior GOSUB Re_display
670 ON KEY 1 LABEL "Self Test",Prior GOSUB Self_test
680 ON KEY 2 LABEL "Star Disable",Prior GOSUB Star_disable
690 ON KEY 3 LABEL "Star Enable",Prior GOSUB Star_enable
700 ON KEY 4 LABEL "Adaptive Rate",Prior GOSUB Adapt_rate
710 ON KEY 5 LABEL "Acquisition",Prior GOSUB Acquisition
720 ON KEY 6 LABEL "Take Data",Prior GOSUB Take_data
730 ON KEY 7 LABEL "Menu 2 Options",Prior GOSUB Menu2_start
740 ON KEY 8 LABEL "Tracker Status",Prior GOSUB Get_status
750 ON KEY 9 LABEL "Exit",Prior GOTO Shutdown
760 PRINTER 1 1
770 CONTROL 1:1,1
780 OUTPUT 2:Clear$;  \* Clear the terminal screen
790 GOSUB Chk_flgs_menu  \* Set up initial flag conditions
800
810 Menu:
820 Menu_flag=1
830 CONTROL 1:26,1
840 PRINT "TRACKER COMMAND MENU #1"
850 CONTROL 1:25,3
860 PRINT "1:0 Re-display Screen"
870 CONTROL 1:25,4
880 PRINT "1:1 Self Test"
890 CONTROL 1:25,5
900 PRINT "1:2 Star Disable"
910 CONTROL 1:25,6
920 PRINT "1:3 Star Enable"
930 CONTROL 1:25,7
940 PRINT "1:4 Adaptive Rate"
950 CONTROL 1:25,8
960 PRINT "1:5 Acquisition"
970 CONTROL 1:25,9
980 PRINT "1:6 Take Data"
990 CONTROL 1:25,10
1000 PRINT "1:7 Menu #2 Options"
1010 CONTROL 1:25,11
150 PRINT "k8 Get Tracker Status"
160 CONTROL 1:25,12
170 PRINT "k9 Exit Program"
180 CONTROL 1:11,14
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
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
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_flgs_menu ! Check the status of all flags, etc...
350 GOTO Menu1 ! Loop in Menu 1 options
360
370 ! -----------------------
380 ! Menu2_start
390 ! -----------------------
400 Menu2_start:
410 Menu_flag=2
420 CONTROL 1:26,1
430 PRINT "TRACKER COMMAND MENU #2"
440 CONTROL 1:25,3
450 PRINT "k0 Re-display Screen"
460 CONTROL 1:25,4
470 PRINT "k1 Track Current Star at X, Y"
480 CONTROL 1:25,5
490 PRINT "k2 Set Drop Star Criteria [";Num_times;"]
500 CONTROL 1:25,6
510 PRINT "k3 Set X Position [";X_posn;"]
520 CONTROL 1:25,7
530 PRINT "k4 Set Y Position [";Y_posn;"]
CONTROL 1:25.8
PRINT "k5 Set Star Number [";Curr_star;""
CONTROL 1:25.9
PRINT "k6 Take Data"
CONTROL 1:25.10
PRINT "k7 Menu #1 Options"
CONTROL 1:25.11
PRINT "k8 Get Tracker Status"
CONTROL 1:25.12
PRINT "k9 Exit Program"
CONTROL 1:1.14
PRINT 3(11X,10A.+)/"Star #1","Star #2","Star #3"
OUTPUT 1 USING 3(11X,10A.+)/"Star #1","Star #2","Star #3"

STATUS 12.5;Tkr_status
Tkr_command=BINAND(Tkr_status,3)+10
IF Test_flag=1 THEN Tkr_command=12
SELECT Tkr_command
CASE 10
GOSUB Get_data
GOSUB Trans_error
CASE 11
GOSUB Trans_error
CASE 12
GOSUB Get_data
END SELECT
GOSUB Check_data
GOTO Menu2
GOTO Menu2
STOP

----------- Program Subroutines -----------
----------- Re_display -----------

Redisplay the current screen

Re_display:
IF Menu_flag=1 THEN GOSUB Menu1_start
IF Menu_flag=2 THEN GOSUB Menu2_start
RETURN

----------- Chk_flag_men1 -----------
This routine will display all flags and conditions, for variables in Menu1.

Chk_flag_men1:
GOSUB Chr_self_tst Self Test Flag
GOSUB Check_stars Individual Star Enable/Disable conditions
GOSUB Check_adapt Adaptive Rate Flag
GOSUB Check_acqu Acquisition Flag
GOSUB Check_data Data taking Flag
RETURN

-----------------------------
This routine will check and display the STATUS of the SELF TEST FLAG.

Chk_self_tst:

CONTROL 1:40,4
SELECT Self_test_flag
CASE 0
   Disp_flag$=Off_flag$
   END SELECT
PRINT Disp_flag$
RETURN

This routine will check and display the STATUS (enable/disable) OF THE STARS.

Check_stars:

CONTROL 1:46,5
SELECT Star1_flag
CASE 0
   PRINT "1"
   CONTROL 1:46,6
   PRINT "1"
END SELECT
PRINT "1"
PRINT "1"
END SELECT
PRINT "1"
PRINT "1"
PRINT "1"
END SELECT
PRINT "1"
PRINT "1"
END SELECT
PRINT "1"
PRINT "1"
RETURN

Position cursor at SELF_TEST_FLAG location
Now determine self test condition
If Self test is OFF,
   then display OFF flag
If Self test is ON,
   then display ON flag

Display the flag

Position cursor for Star #1
Check Star #1 condition
If Star #1 is disabled,
   then display Star #1 as disabled
   and clear enable display for Star #1
If Star #1 is enabled,
   then clear disable display for Star #1
   and display Star #1 as enabled
End of Star #1 condition check...

Position cursor for Star #2
Check Star #2 condition
If Star #2 is disabled,
   then display Star #2 as disabled
   and clear enable display for Star #2
If Star #2 is enabled,
   then clear disable display for Star #2
   and display Star #2 as enabled
End of Star #2 condition check...

Position cursor for Star #3
Check Star #3 condition
If Star #3 is disabled,
   then display Star #3 as disabled
   and clear enable display for Star #3
If Star #3 is enabled,
   then clear disable display for Star #3
   and display Star #3 as enabled
End of Star #3 condition check...
DONE!!
This routine will check and display the STATUS of the ADAPTIVE RATE FLAG.

**Check_adapt**

0310 CHECK_ADAPT:
0320 CONTROL 1:44,7
0330 SELECT Adapt_rate_flag
0340 CASE 0
0350 DISP_FLAG$=OFF_FLAG$ ! Position cursor for ADAPT_RATE_FLAG
0360 CASE 1
0370 DISP_FLAG$=ON_FLAG$ ! Check condition of the adaptive rate flag
0380 END SELECT
0390 PRINT DISP_FLAG$
0400 RETURN

**Check_acqu**

This routine will check and display the STATUS of the ACQUISITION FLAG.

0500 CHECK_ACQU:
0510 CONTROL 1:42,8
0520 SELECT Acqu_flag
0530 CASE 0
0540 DISP_FLAG$="MANUAL" ! Position cursor for ACQU_FLAG
0550 IF Acqu_type=0 THEN DISP_FLAG2$="OFF" ! Check the status of the Acquisition Flag
0560 IF Acqu_type=1 THEN DISP_FLAG2$="FULL FOV"
0570 IF Acqu_type=2 THEN DISP_FLAG2$="FULL EDGE"
0580 IF Acqu_type=3 THEN DISP_FLAG2$="VEC EDGE"
0590 CASE 1
0600 DISP_FLAG$="AUTO" ! If Acquisition is OFF, then display the OFF condition
0610 IF Disp_flag2$="" THEN Disp_flag2$="OFF"
0620 IF Acqu_type=0 THEN Disp_flag2$="" ! If Acquisition is ON, then display the ON condition
0630 IF Acqu_type=1 THEN Disp_flag2$="FULL FOV"
0640 IF Acqu_type=2 THEN Disp_flag2$="FULL EDGE"
0650 IF Acqu_type=3 THEN Disp_flag2$="VEC EDGE"
0660 END SELECT
0670 PRINT DISP_FLAG$":"""";Disp_flag2$ ! Display the flags
0680 RETURN

**Check_data**

This routine will check and display the STATUS of the DATA TAKING FLAG.

0700 CHECK_DATA:
0710 CONTROL 1:40,9
0720 SELECT Tak_data_flag
0730 CASE 0
0740 DISP_FLAG$=OFF_FLAG$ ! Position cursor for TAK_DATA_FLAG
0750 PRINT DISP_FLAG$;
0760 CASE 1
0770 DISP_FLAG$=ON_FLAG$ ! Check status of the data taking flag
0780 PRINT DISP_FLAG$;
0790 END SELECT
0800 RETURN
This routine will TOGGLE the SELF TEST FLAG and set up the variables needed to send commands to the tracker interface.

Self_test:

DISABLE

Self_test_flag=1-Self_test_flag

GOSUB Chk_self_tst

Disable all interrupts from ON KEY commands

Toggle the SELF TEST FLAG

Display the NEW condition

Set up command for SELF TEST and send message

Send_msg$="Self Test "&Disp_flag$&" Command"

GOSUB Cmd_interface

Enable interrupts for ON KEY commands

Return

This routine will DISABLE the CURRENT STAR, and change the star flag to show such status.

Star_disable:

DISABLE

IF Curr_star=1 THEN Star1_flag=0

IF Curr_star=2 THEN Star2_flag=0

IF Curr_star=3 THEN Star3_flag=0

GOSUB Check_stars

Send_msg$="Star Disable Command"

GOSUB Cmd_interface

Output command to interface box

Set up command for disable star and send message

Display NEW condition

Set up command to reflect star # and send message

Output star # to interface

Enable all interrupts for ON KEY commands

Return

This routine will ENABLE the CURRENT STAR, and change the star flag to show such status.
Star_enable:  ! Disable all interrupts from the ON KEY commands
   DISABLE
   IF Curr_star=1 THEN Star1_flag=1
   IF Curr_star=2 THEN Star2_flag=1
   IF Curr_star=3 THEN Star3_flag=1
   GOSUB Check_stars
   Interface=120
   Send_msg$="Star Enable Command"
   GOSUB Cmd_interface
   Interface=120 ! Set up command for star enable
   Send_msg$="Star #&VAL$(Curr_star)"
   GOSUB Cmd_interface
   ENABLE
   RETURN

Adapt_rate:  ! Disable all interrupts from ON KEY commands
   DISABLE
   Adapt_rate_flag=1-Adapt_rate_flag
   GOSUB Check_adapt
   Interface=180
   GOSUB Cmd_interface
   ENABLE
   RETURN

Acquisition:  ! Disable all interrupts from ON KEY commands
   DISABLE
   Acqu_flag=1-Acqu_flag
   GOSUB Check_acqu
   IF Acqu_flag=0 THEN
      BEEP
      INPUT "Please input Acquisition type: 0=OFF, 1=FULL FOV, 2=FULL EDGE, 3=VSC EDGE",Acqu_type
      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 Interface=204
4810 GOSUB Cmd_interface ! output command to interface
4820 ENABLE ! Enable all interrupts for ON KEY commands
4830 RETURN
4840
4850 |
4860 |
4870 |
4880 |
4890 |
4900 |
4910 |
4920 |
4930 |
4940 |
4950 |
4960 |
4970 |
4980 |
4990 |
5000 |
5010 |
5020 |
5030 |
5040 |
5050 |
5060 |
5070 |
5080 |
5090 |
5100 |
5110 |
5120 |
5130 |
5140 |
5150 |
5160 |
5170 |
5180 |
5190 |
5200 |
5210 |
5220 |
5230 |
5240 |
5250 |
5260 |
5270 |
5280 |
5290 |
5300 |
5310 |

This routine will TOGGLE the DATA TAKING FLAG, between ON/OFF, no command is sent to the interface but some variables associated with the process must be set.

Take_data: ! Disable all interrupts from ON KEY commands
4950 Disable
4960 Take_data_flag=1-Take_data_flag ! Toggle the data taking flag
4970 GOSUB Check_data ! Display the NEW condition
4980 IF Take_data_flag=0 THEN Take_data_end ! Else re-set the data set counter
5000 Data_count=0
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 INPUT "Please input initial data file # ",Nfiles
5070 IF Nfiles>=0 AND Nfiles<=99 THEN ! Check input for validity
5080 !
5090 !
5100 !
5110 !
5120 !
5130 !
5140 !
5150 !
5160 !
5170 !
5180 !
5190 !
5200 Take_data_end: ! Write residual data to the file
5210 IF Data_count>0 THEN GOSUB Write_data ! Enable all interrupts for the ON KEY commands
5220 ENABLE
5230 RETURN
5240
5250 !
5260 !
5270 !
5280 !
5290 !
5300 !
5310 !
This routine will set up variables for the COMMANDED TO TRACK POSITION, it will use the CURRENT STAR #, X POSITION, Y POSITION for information sent to the tracker interface.

Track_it:

DISABLE
Inter_face=84
Set up commanded to track position command
Send_msg$="Track at X. Y Command"
GOSUB Cmd_interface
Output Track command
Send_msg$="Star #"&VAL$(Curr_star)
GOSUB Cmd_interface
Set up star # and send message
Send_msg$="X Position ["&VAL$(X_posn)&"]"
GOSUB Cmd_interface
Output X position to interface (must output this puppy twice...)
Send_msg$="Y Position ["&VAL$(Y_posn)&"]"
GOSUB Cmd_interface
Output Y position to interface (must output this puppy twice)
ENABLE
RETURN

Drop_criteria:

DISABLE
DISP
BEEP
INPUT "Please input the new drop criteria >", Num_times
IF Num_times>0 AND Num_times<=15 THEN End_criteria
BEEP
DISP "INVALID drop criteria "; Num_times; ", press <CONT> to try again!" PAUSE
GOTO Drop_criteria
End_criteria:
Send_msg$="Drop Criteria Command"
Inter_face=44
GOSUB Cmd_interface
Set up send message and Drop criteria command
GOSUB Cmd_interface
Output command to interface
Set up send message
.interface=Ctble(Num_times) ! and drop criteria data
Send_msg$="Drop Criteria Data ["&VAL$(Num_times)&"]"
GOSUB Cmd_interface ! Output command to interface
ENABLE ! Enable all interrupts for ON KEY commands
RETURN

This routine will query the user for the new X POSITION.

Set_x:
DISP ! Clear the display line
BEEP
INPUT "Please input the new X position",X_posn
IF X_posn>-1 AND X_posn<257 THEN End_set_x
BEEP
DISP "INVALID X position ";X_posn;" press <CONT> to try again!"
PAUSE
GOTO Set_x
End_set_x:
RETURN

This routine will query the user for the new Y POSITION.

Set_y:
DISP ! Clear the display line
BEEP
INPUT "Please input the new Y position",Y_posn
IF Y_posn>-1 AND Y_posn<257 THEN End_set_y
BEEP
DISP "INVALID Y position ";Y_posn;" press <CONT> to try again!"
PAUSE
GOTO Set_y
End_set_y:
RETURN

This routine will query the user for the NEW STAR NUMBER.

Set_star:
DISP ! Clear the display line
BEEP
INPUT "Please input the new star number",Curr_star
IF Curr_star>0 AND Curr_star<4 THEN End_set_star
BEEP
DISP "INVALID star number ":Curr_star;" press <CONT> to try again!
PAUSE
GOTO Set_star ! Let them try again...
End_set_star:
RETURN

*************
Get_status
*************

This routine will output the GET STATUS command to the interface, and will then check the each of the individual condition flags that will be retrieved from the interface.

Get_status:
DISABLE ! Disable all interrupts from ON KEY commands
Set uip the send message
and Get status command
Send_msg$="Get Status Command"

GOSUB Cmd_interface ! Output command to the interface
IF test_flag=0 THEN ! If we ARE using the interface box then...
Ctrl=BINAND(1,Ctrl)
CONTROL 12,2;Ctrl ! Get the status data from the interface
WAIT .1
ENTER 12 USING "3(#,W)";Status1,Status2,Status3
ENTER 12 USING "3(#,W)";Status4,Status5,Status6
ENTER 12 USING "3(#,W)";Status7,Status8,Status9
IF (Status1=Status4) AND (Status4=Status7) THEN ! Make sure they are of the same value
IF Status1<0 THEN Status1=65536+Status1
Status1=Status1/256 ! Convert these values to decimal
CONTROL 1:22,14
PRINT USING "*.10A.K";" Status* ";:Status1
PAUSE
Star1_flag=BIT(Status1,15)
Star2_flag=BIT(Status1,14)
Star3_flag=BIT(Status1,13)
Track1_flag=BIT(Status1,12)
Track2_flag=BIT(Status1,11)
Track3_flag=BIT(Status1,10)
Pt1=BIT(Status1,9)
Pt2=BIT(Status1,8)
Acou_flag=Pt1+Pt2
Seif_test_flag=BIT(Status1,7)
Pt1=BIT(Status1,4)
Pt2=BIT(Status1,3)
Pt3=BIT(Status1,2)
Pt4=BIT(Status1,1)
N pur times=(Pt1*8)+(Pt2*4)+(Pt3*2)+Pt1
END IF
END IF
Enable all interrupts for the ON KEY commands
RETURN

*************
Cmd_interface
*************
This routine will output a specific command stored in variable INTERFACE, to the interface box. While it is "waiting" for the acknowledge signal, it will concurrently collect data from the tracker.

```
180 Cmd_interface:!!
190 Menu_flag=3
200 Ctrl=2 ! Set dir=OUT CTL1=1
210 CONTROL 12.2;Ctrl ! Command ready CTL0=1
220 WAIT .1 ! Wait for uP to recover from reset
230 Ctrl=BINIOR(1,Ctrl) ! CTL0 pulsed down, set low when in normal operation
240 CONTROL 12.2;Ctrl
250 DISP
260 DISP "Sending ":Send_msg$:" to tracker..."
270 ! DISP "Sending ":Send_msg$:" to tracker...PRESS STEP, CONT, ETC"
280 ! PAUSE
290 IF Test_flag=1 THEN
300 ! WAIT .5
310 ! DISP
320 ! RETURN
330 END IF
340 OUTPUT 12 USING ":,B":INTERFACE ! Output command to interface
350 Time_out=(TIMEDATE) MOD 86400 ! Must wait for 7 seconds for
360 Time_out=Time_out+7 ! interface, collect data while
370 Time_out=Time_out+1 MOD 60 ! waiting...
380 Time_flag=0 ! It isn't time yet...
390 Do_it_again:
400 Time_yet=TIMEDATE MOD 86400 ! Check current time
410 IF (Time_out>Time_out1) THEN
420 IF (Time_yet>Time_out) OR (Time_yet<Time_out1) THEN Time_flag=1
430 ELSE
440 IF (Time_yet>Time_out) AND (Time_yet<Time_out1) THEN Time_flag=1
450 END IF
460 ! Check status:
470 ! DISP "Time_yet":Time_yet;" Time_out ":;Time_out1;" Time_flag ":Time_flag ! FOR DEBUGGING PURPOSES
480 ! WAIT 1.0
490 STATUS 12.5:Tkr_status
500 Tkr_command=BINAND(Tkr_status,3)+10
510 SELECT Tkr_command
520 CASE 10
530 ! GOSUB Get_data
540 ! Ctrl=BINAND(2,Ctrl)
550 ! CONTROL 12.2;Ctrl
560 ! Ctrl=BINIOR(1,Ctrl)
570 ! CONTROL 12.2;Ctrl
580 ! GOSUB Trans_error
590 CASE 12
600 ! GOSUB Trans_error
610 ! CASE 12
620 ! GOSUB Get_data
630 END SELECT
640 IF Time_flag=2 THEN
650 ! DISP
660 ! RETURN
670 END IF
680 IF Time_flag=0 THEN GOTO Do_it_again ! Not time yet...
690 IF Time_flag=1 THEN Time_flag=2 ! Check status one more time...
```
This routine will RECEIVE and DISPLAY STAR DATA from the interface box.

Get_data:

DISABLE

IF Test_flag=0 THEN

Get star data from the tracker

CONTROL 12,2:Ctrl

ENTER 12 USING "3(#,W)";Star1_x,Star1_y,Star1_m

ENTER 12 USING "3(#,W)";Star2_x,Star2_y,Star2_m

ENTER 12 USING "3(#,W)";Star3_x,Star3_y,Star3_m

Now check these numbers for negative values

IF Star1_x<0 THEN Star1_x=65536+Star1_x

IF Star1_y<0 THEN Star1_y=65536+Star1_y

IF Star1_m<0 THEN Star1_m=65536+Star1_m

IF Star2_x<0 THEN Star2_x=65536+Star2_x

IF Star2_y<0 THEN Star2_y=65536+Star2_y

IF Star2_m<0 THEN Star2_m=65536+Star2_m

IF Star3_x<0 THEN Star3_x=65536+Star3_x

IF Star3_y<0 THEN Star3_y=65536+Star3_y

IF Star3_m<0 THEN Star3_m=65536+Star3_m

Star1_x=Star1_x/256

Star1_y=Star1_y/256

Star1_m=Star1_m/256

Star2_x=Star2_x/256

Star2_y=Star2_y/256

Star2_m=Star2_m/256

Star3_x=Star3_x/256

Star3_y=Star3_y/256

Star3_m=Star3_m/256

ELSE

Star1_x=11+Data_count

Star1_y=12+Data_count

Star1_m=13+Data_count

Star2_x=14+Data_count

Star2_y=15+Data_count

Star2_m=16+Data_count

Star3_x=17+Data_count

Star3_y=18+Data_count

Star3_m=19+Data_count

END IF

Calculate time variables to be written to the data file

Datat=(3600*Hour+60*Minute+Second-TIMEDATE)+ABS(Datat)

Data_time=(TIMEDATE+Datat) MOD 86400

Data_time=INT(Data_time*100)/100

Hour=Data_time DIV 3600

Minute=Data_time MOD 3600 DIV 60

Second=Data_time MOD 60

CONTROL 1:1,10

Display the update flag

PRINT USING "#.1A";Data_toggle$
Data_toggle$=CHR$(139-NUM(Data_toggle$))

CTRL 1:22.14
! Display the star values being received
from the tracker...

PRINT USING "#.4A,4D,4D"; X=":Star1_x
PRINT USING "#.4A,4D,4D"; Y=":Star1_y
PRINT USING "#.4A,4D,4D"; M=":Star1_m

CTRL 1:22.15
PRINT USING "#.4A,4D,4D"; X=":Star2_x
PRINT USING "#.4A,4D,4D"; Y=":Star2_y
PRINT USING "#.4A,4D,4D"; M=":Star2_m

CTRL 1:22.16
PRINT USING "#.4A,4D,4D"; X=":Star3_x
PRINT USING "#.4A,4D,4D"; Y=":Star3_y
PRINT USING "#.4A,4D,4D"; M=":Star3_m

IF Tak_data_flag=0 THEN End_get_data

Data_count=Data_count+1

IF Data_count<Max_dat THEN
    Star_data(Data_count,1)=Star1_x
    Star_data(Data_count,2)=Star1_y
    Star_data(Data_count,3)=Star1_m
    Star_data(Data_count,4)=Star2_x
    Star_data(Data_count,5)=Star2_y
    Star_data(Data_count,6)=Star2_m
    Star_data(Data_count,7)=Star3_x
    Star_data(Data_count,8)=Star3_y
    Star_data(Data_count,9)=Star3_m
    Star_data(Data_count,10)=Hour
    Star_data(Data_count,11)=Minute
    Star_data(Data_count,12)=Second

CTRL 1:48.9
IF Data_count>0 THEN PRINT Data_count
IF Data_count=0 THEN PRINT " "

END IF

END IF

IF Data_count=Max_dat THEN GOSUB Write_data

End_get_data:

ENABLE
RETURN

------------------------
 Write_data
------------------------

This routine will write data to the output file.

ON ERROR GOTO Error_tst

Number$=VAL$(Number$)


DISP "Writing data to >";File_string$

DISP "Designate the right disk drive for data

MASS STORAGE IS ";HP82901,700,1"

CREATE BDAT File_string$.Max_dat

ASSIGN @Path_1 TO File_string$

OUTPUT @Path_1:Star_data(*)
ASSIGN @Path_1 TO *  
INC Nfiles=Nfiles+1  
DATA_count=0  
OFF ERROR  
DISP  
RETURN  

**Error_tst**  

This routine will trap any errors encountered in the data taking process.

```plaintext
4040 Error_tst:  
4050 BEEP  
4060 OFF ERROR  
4070 IF ERRN=54 THEN  
4080 DISP "This file already exists, press <CONT> to try again!"  
4090 PAUSE  
4100 DISP "Please input initial file # >","Nfiles " try again...  
4110 IF Nfiles>0 AND Nfiles<99 THEN  
4120 INPUT "Please input initial file # >",Nfiles " try again...  
4130 ELSE  
4140 IF ERRN=64 THEN  
4150 DISP "The data disc is full, Please exchange it with a new one..."  
4160 WAIT .5  
4170 DISP "Press <CONT> after replacing the disc to continue!"  
4180 PAUSE  
4190 DISP  
4200 GOTO Write_data  
4210 END IF  
4220 ELSE  
4230 IF ERRN=80 THEN  
4240 DISP "This file already exists, press <CONT> to try again!"  
4250 PAUSE  
4260 END IF  
4270 END IF  
4280 IF ERRN=54 THEN  
4290 DISP "This file already exists, press <CONT> to try again!"  
4300 PAUSE  
4310 END IF  
4320 ELSE  
4330 IF ERRN=64 THEN  
4340 DISP "The data disc is full, Please exchange it with a new one..."  
4350 WAIT .5  
4360 DISP "Press <CONT> after replacing the disc to continue!"  
4370 PAUSE  
4380 END IF  
4390 END IF  
4400 END IF  
4410 END IF  
4420 END IF  
4430 END IF  
4440 END IF  
4450 RETURN  
4460 RETURN  
4470 !  
4480 !
```
This routine will display a message, and beep at you if there was a transmit error. By using the Menu_flag to determine where the error came from, a message will be displayed.

9570 Trans_error: !
9580 IF Menu_flag=0 THEN Err_msg$="" ! Let the user know where the error came from...
9590 IF Menu_flag=1 THEN Err_msg$=" 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$
9640 Ctrl=BNAND(2,Ctrl)
9650 CONTROL 12.2:Ctrl
9660 Ctrl=BNOR(1,Ctrl)
9670 CONTROL 12.2:Ctrl
9680 Freq=4000
9690 Freq2=1000
9700 Bnumber=2
9710 Btime=.2
9720 Dfreq=(Freq2-Freq)
9730 FOR F=0 TO Bnumber-1
9740 Bfreq=Freq+(Dfreq*F)
9750 BEEP Bfreq,Btime ! Beep 'em...
9760 Btime=Btime*5
9770 NEXT F
9780 DISP
9790 RETURN
9800 !
9810 !
9820 !
9830 !
9840 !
9850 !
9860 !
9870 !
9880 !
9890 !
9900 Ctabl(1)=224
9910 Ctabl(2)=152
9920 Ctabl(3)=120
9930 Ctabl(4)=84
9940 Ctabl(5)=180
9950 Ctabl(6)=204
9960 Ctabl(7)=44
9970 Ctabl(8)=210
9980 Ctabl(9)=50
9990 Ctabl(10)=74
10000 Ctabl(11)=172
10010 Ctabl(12)=136
10020 Ctabl(13)=102
10030 Ctabl(14)=30
10040 Ctabl(15)=254
10050 RETURN
10060 !
10070 !
10080 !
This routine will be used as a no operation type subroutine, if the user inputs an invalid option, they will be advised of the input error.

No_op:

BEEP

Err_msg$=""

IF Menu_flag=1 THEN Err_msg$="from Menu #1"

IF Menu_flag=2 THEN Err_msg$="from Menu #2"

DISP "INVALID COMMAND ENTRY..."

WAIT .5

DISP Err_msg$

WAIT .5

DISP

RETURN

-------------

Shutdown:

FF KEY

PRINT "PROGRAM TERMINATED..."

END
Section 1-2
Section I-2

FLOWCHART OF THE STAR TRACKER CONTROL PROGRAM
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)
Clear the screen
Terminal Output

GOSUB
[Check_flags_menu]
(PAGE 9)
```
00SUB
Get_status
(PAGE 36)

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''

: Display
: current star
: values, star #
: X position,
: Y position &
: # of acquisition
: 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
\*------------------*

\*------------------*
**GOSUB**
\*Get_data*
\*(PAGE 43)*
\*------------------*

**GOSUB**
\*Trans_error*
\*(PAGE 57)*

**GOSUB**
\*Get_data*
\*(PAGE 43)*

\*------------------*
**GOSUB**
\*Trans_error*
\*(PAGE 57)*

\*------------------*
**GOSUB**
\*Check_data*
\*(PAGE 15)*

\*------------------*
Redisplay the current screen.

No /

IF

< Menu_flag = 1 >

< Menu_flag = 2 >

RETURN

Check and then display all flags for Menu #1.

Go to Menu #1 start (PAGE 11).
**CHK_SELT_TST**

Check and display the self test status

**Position the cursor at loc of self test flag**

**Terminal Output**

```plaintext
case 0/1
SELECT Self_test_flag
```

```
Disp_flag$ = Off_flag$
```

```
Disp_flag$ = On_flag$
```

**Print**

```
Disp_flag$
```

**Terminal Output**

**RETURN**

**CHECK_STARS**

Check and display the individual star status
Position the cursor at loc of star #1
disabled
Terminal Output

\begin{figure}
\centering
\includegraphics[width=\textwidth]{diagram}
\caption{Diagram of cursor positioning and print statements.}
\end{figure}

\begin{itemize}
\item Case 0:
\begin{itemize}
\item SELECT Star1_flag
\end{itemize}
\item Case 1:
\begin{itemize}
\item Print "1"
\end{itemize}
\item Terminal Output
\end{itemize}

\begin{itemize}
\item Position the cursor at loc of star #1
\item enabled
\item Print "1"
\end{itemize}

\item Terminal Output

\begin{itemize}
\item Print "1"
\end{itemize}

\item Terminal Output

\begin{itemize}
\item Position the cursor at loc of star #2
\item disabled
\item Terminal Output
\end{itemize}

\begin{itemize}
\item Case 0:
\begin{itemize}
\item SELECT Star2_flag
\end{itemize}
\item Case 1:
\begin{itemize}
\item Print "2"
\end{itemize}
\item Terminal Output
\end{itemize}

\item Terminal Output

\item Terminal Output
CHECK_ADAPT

Check and display the adaptive rate status

Position the cursor at loc of adaptive rate flag

Terminal Output

---

case 0/ SELECT \\ case 1

----------

Adapt_rate

----------

Disp_flag$ = Off_flag$

----------

Disp_flag$ = On_flag$

----------

Print Disp_flag$

---

Terminal Output

---

RETURN
CHECK_DATA

Check and display the status of the data taking flag

Position the cursor at loc of take data flag

Terminal Output

CASE 0/

SELECT \ case 1

\ case 1

Tak_data_flag

\ case 1

disp_flag$ = off_flag$

Print disp_flag$

"";

Terminal Output

+----------------------+

RETURN

+----------------------+
IF Acnu_type = 2

Yes

Disp_flag2$ = "FULL EDGE"

IF Acnu_type = 3

Yes

Disp_flag2$ = "VEC EDGE"

Display

Disp_flag2$

Disp_flag2$

Terminal Output:

RETURN

***************

SELF_TEST

***************

Toggle the self test flag and set up the variables to be sent to the interface box!

(DISABLE interrupts)

(from soft keys)
**STAR_DISABLE**

Enable the current star and change the star flag to show such status.

(ENABLE interrupts)
(from soft keys)

---

**DISABLE interrupts**
(from soft keys)

---

**Self_test_flag** = 1 - Self_test_flag

GOSUB Chk_self_tst (PAGE 11)

Interface = 224:

Send_msg$ = "Self Test Cmd"

GOSUB Cmd_interface (PAGE 38)

(ENABLE interrupts)
(from soft keys)

RETURN

---

Page 18
No / IF

Curr_star = 1

Yes

!Star1_flag = 0:

No / IF

Curr_star = 2

Yes

!Star2_flag = 0:

No / IF

Curr_star = 3

Yes

!Star3_flag = 0:

GOSUB Check_stars (PAGE 11)

!Inter_face= 52

Send_mess$ = "Star Disable"

GOSUB Cmd_interface (PAGE 38)
Interface = Ctable(Curr_star)

Send_msg$ = "Star #"

GOSUB Cmd_interface
(PAGE 38)

ENABLE interrupts
(from soft keys)

RETURN

***************
STAR_ENABLE
***************
Enable the current star and change the star flag to show such status

DISABLE interrupts
(from soft keys)

IF Curr_star = 1

Yes

Star1_flag = 1
GOSUB Cmd_interface (PAGE 38)

( ENABLE interrupts )
( from soft keys )

RETURN
ADAPT-RATE

**Togqle** the adaptive rate flag and set up the variables to be sent to the interface box.

( DISABLE interrupts )
(from soft keys )

---

(Adapt_rate_flag) = 1
(Adapt_rate_flag)

---

GO SUB
Check adapt
(PAGE 14)

---

Interface = 180

---

Send_msg$ = "Adaptive Rate"

---

GO SUB
Cmd interface
(PAGE 38)

---

( ENABLE interrupts )
(from soft keys )

---

RETURN
*******
ACQUISITION
*******

Toggle the acq-
quisition flag
and set up the
variables to
be sent to the
interface box

(DISABLE interrupts)
(from soft keys)

Acq_flag
= 1 - Acq_flag

GOSUB
Check_acq
(PAGE 16)

IF
< Acq_flag = 0
\ No
/ Yes
/
Yes
Keep the
terminal bell

Terminal Output

User Input

"Please input
Acquisition
type 0-off,
1-FULL FOV,
2-FULL EDGE,
3-VEC EDGE >"

Read Acq_type

Page 24
IF (-1 < Acq_type < 4) THEN
    Yes

    Beep the terminal bell

    Terminal Output

    Display "INVALID type
    press <CONT> to try again"

    Terminal Output

ENDIF

;**************************************************************
SEND_ACOU
;**************************************************************

;**************************************************************
!Interface = 204;
;**************************************************************

Send_message = "Acquisition"

SUB Cmd_interface
(PAGE 38)

;**************************************************************

( ENABLE interrupts )
( from soft keys )

RETURN
TAKE_DATA

Toggle the take data flag, and set up variables associated with the current status

(DISABLE interrupts)
(from soft keys)

TAKE_flag = 1 - TAKE_flag

GOSUB Check_data (PAGE 15)

IF (TAKE_data_flag = 0)

No

Data_count = 0

Max_num_files = Nfiles + Max_files

IF (Nfiles <= Max_files)

Yes

Printer is 1
IF Nfiles = 0

\  \ No
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
/
**Terminal Output:**

```plaintext
\* Clear display
\* line
\* Terminal Output:
```

```
\*------------------------
\* ************************
\* TAKE_DATA_END
\* ************************
\*------------------------
\* IF \ No
\< Data_count > 0 >------------------------
\*------------------------
\* Yes
\*------------------------
\* GOSUB
\* Write_data
\* (PAGE 52)
```

```
\*------------------------
\* ( DISABLE interrupts )
\* ( from soft keys )
```

```
\*------------------------
\* RETURN
```

```
\*------------------------
\* ************************
\* TRACK_IT
```

```
\*------------------------
\* Set up vars
\* used for the command to track position
```

```
\*------------------------
\* ( DISABLE interrupts )
\* ( from soft keys )
```

```
\*------------------------
\* Interface= 84
```

---

Page 28
Send_msg$ = "Track @ X,Y"

GOSUB Cmd_interface (PAGE 38)

Interface = Cthie(Curr_star)

Send_msg$ = "Star #"

GOSUB Cmd_interface (PAGE 38)

Interface=X_posn!

Send_msg$ = "X Position"

GOSUB Cmd_interface (PAGE 38)

GOSUB Cmd_interface (PAGE 38)

Interface=Y_posn!
Sends message
"Y Position"

GOSUB
Cmd_interface
(PAGE 38)

GOSUB
Cmd_interface
(PAGE 38)

( ENABLE interrupts )
( from soft keys )

RETURN

***************
SET_STAR
***************

Query user for new current star

Clear the display line
Beep terminal
Terminal Output

User Input
"Please input NEW star # >"
read Curr_star

Page 30
IF 0 < Curr_star < 4:

Yes

/ No

: Enter the terminal bell!

: Terminal Output

/ Display

"INVALID star number, try again!"

: Terminal Output

/:

<

END_SET_STAR

RETURN

DROP_CRITERIA

Query the user for the NEW drop criteria

DISABLE interrupts

from soft keys

Clear the display line

: Terminal Output

/
```plaintext
GOSUB Cmd_interface (PAGE 38)

Interface = Ctable(Num_times)

Send_msg$ = "Drop data"

GOSUB Cmd_interface (PAGE 38)

( ENABLE interrupts )
( from soft keys )

RETURN
```
Qur-y User 4or NEW XPositi
Clear the display line Beep terminal
Terminal Output

User Input
"Please input NEW x position > "
read X_posn

IF \ Yes
<-1 < X_posn < 257 >---
\ No

Beep the terminal bell
Terminal Output

Display "INVALID X position, try again!"
Terminal Output

RETURN
SET_Y

Clear the display line
Beep terminal
Terminal Output:

User Input

"Please input NEW Y position > "
read Y_posn

IF \ Yes
<-1 < Y_posn < 257 >

Beep the terminal bell
Terminal Output:

Display
"INVALID Y position, try again!"
Terminal Output:

<-

END_SET_Y

RETURN
**GET_STATUS**

Output set status cmd and then set flags by rec. status!

---

(DISABLE interrupts )  
(from soft keys )

---

!Interface = 2101

---

Send_msg$ = "Get Status"

---

**ROSUB**

Cmd_interface (PAGE 38)

---

IF Test_flag = 0

---

!Yes

\Ctrl = RINAND(1, Ctrl)/ \CONTROL 12, 21 Ctrl /

/ WAIT .1

---

/ Get data from interface

STATUS1

STATUS2

STATUS3
/
/ Get data from interface
/
STATUS4
STATUS5
STATUS6
/
/
/ Get data from interface
/
STATUS7
STATUS8
STATUS9
/
/
/ IF \ No < STATUS1=STATUS4= >------------------>
/ \ STATUS7 /
/ \ Yes +------------------+
/ \ Yes
/ \ IF \ Yes < STATUS1 = 0 >------------------>
/ \ No +------------------+
/ \ IF \ No < STATUS1 < 0 >------------------>
/ \ Yes +------------------+
/ \ Yes
/ \ STATUS1 =
/ \ 165536 + STATUS1:
/ \ Print "Status 1 = ", STATUS1:
/ \ Terminal Output:
/ \ Star1_flag =
/ \ BIT(STATUS1,15):
/ \ Star2_flag =
/ \ BIT(STATUS1,14):
CONTROL 12,2; Ctrl

Display "Sending ", SEND.MSG$, " to the tracker ">

Terminal Output:

IF < Test_flag = 0 >

Yes

WAIT .5
Clear the display line

Terminal Output:

RETURN

OUTPUT 12 USING < "#,B": Interface >

Timing_out = TIMEDATE

Timing_out = Timing_out + 7
v

+------------------------+
| Time_out1 = 1 |
| +------------------------+

+------------------------+
| Time_flag = 0 |
| +------------------------+

+------------------------+
| Time yet = TIMENATE |
| +------------------------+

+------------------------+
/ IF \No
\<Time_out> Time_out1>--------->
/ |
+------------------------+
| Yes |
+------------------------+

+------------------------+ 
| Yes/IF Time_yet> \No |
+------------------------+ 
| \Time_out OR > |
| | \Time_yet<Time_out1/ |
+------------------------+ 
| : No |
+------------------------+ 

+------------------------+ 
| Yes/IF Time_yet> \No |
+------------------------+ 
| \Time_out AND > |
| | \Time_yet<Time_out1/ |
+------------------------+ 
| ! |
+------------------------+ 

+------------------------+ 
| Time_flag = 1 |
+------------------------+ 

+------------------------+ 
| CHECK_TRACKER |
| +------------------------+

+------------------------+ 
| WAIT 1.0 |
+------------------------+ 

v
\( \text{STATUS 12.5: Tkr\_status} \)
\( \text{Get tracker status from the interface} \)

\( \text{Tkr\_command} = \text{BINAND(3, Tkr\_status)} + 10 \)

\( \text{case 10/ SELECT Tkr\_command case 12} \)

\( \text{GOSUB Get\_data (PAGE 43)} \)

\( \text{CONTROL 12, 2; Ctrl} \)

\( \text{GOSUB Trans\_error (PAGE 57)} \)

\( \text{CONTROL 12, 2; Ctrl} \)

\( \text{GOSUB Get\_data (PAGE 43)} \)

\( \text{CONTROL 12, 2; Ctrl} \)

\( \text{GOSUB Trans\_error (PAGE 57)} \)
**GET_DATA**

Retrieve and display the star data

---

(DISABLE interrupts)
(from soft keys)

---

No/IF

< Test_flag = 0 >

---

Yes

---

\CONTROL 12.2; Ctrl 

---

/ Get Star1_x,
| Star1_y, and
| Star1_m from
| the tracker

---

/ Get Star2_x,
| Star2_y, and
| Star2_m from
| the tracker

---

/ Get Star3_x,
| Star3_y, and
| Star3_m from
| the tracker

---

Page 43
I Star.

Sitarl_y / 256

I Star1_m =
I Star1_m / 256

I Star2_x =
I Star2_x / 256

I Star2_y =
I Star2_y / 256

I Star2_m =
I Star2_m / 256

I Star3_x =
I Star3_x / 256

I Star3_y =
I Star3_y / 256

I Star3_m =
I Star3_m / 256

I Star1_x =
I Star1_x / 256

1 + Data_count

I Star1_x =
I Star1_x / 256
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:
Star_data(Data_count.8) = Star3_y

Star_data(Data_count.9) = Star3_m

Star_data(Data_count.10) = Hour

Star_data(Data_count.11) = Minute

Star_data(Data_count.12) = Second

Position the cursor at loc of Display set number

IF Data_count > 0

Yes

Display Data_count

Terminal Output

IF Data_count > 0

Yes

Display Data_count

Terminal Output
IF Data_count = 0
  THEN
    No
    Display
    Terminal Output
    Yes
  ENDGETDATA
ENDGETDATA

IF Data_count = Max_data
  THEN
    Yes
    Gosub Write_data (PAGE 52)
    RETURN
  ENDGETDATA
ENDGETDATA

( ENABLE interrupts )
( from soft keys )

RETURN
WRITE_DATA

Write the star data to the output file

ON ERROR GOTO Error_tst
    (trap any errors that may occur in this routine)

Numbers$ = VAL$(Nfiles)

FILE_strings$(6) = Numbers$(1)

DISPLAY 
    "Writing data to file >", Nfiles

Terminal Output:

MASS STORAGE IS...

assign the right disc drive for data

CREATE BDATA
    (Open the data file)

ASSIGN an output path name to file on disk

WRITE \nStar_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
ERROR_TST

Trap any errors that occur in the data taking area.

Keep the terminal bell.

Terminal Output

OFF ERROR

Turn off error trapping.

IF No
< ERNN = 54

Yes

Display "This file already exists".

Terminal Output

PAUSE

Clear the display line.

Terminal Output

User Input

"Please input initial data file # > "

Read Nfiles
IF \( 0 < Nfiles < 99 \) ---->

\[ \text{Max_num_files} = \text{Max_files} + Nfiles! \]

GOTO Write_data (PAGE 52)

IF \( \text{ERRN} = 64 \) ---->

\[ \text{Display} \ "\text{Disc is full}" \text{Terminal Output!} \]

WAIT .5
"Press <CONT>"
\text{Terminal Output!}

Clear the display line
\text{Terminal Output!}

GOTO Write_data (PAGE 52)

IF \( \text{ERRN} = 80 \) ---->

v Yes v
Display "There isn't a correct disc in the right drive"

Terminal Output

WAIT .5
Display "Press <CONT>"

Terminal Output

PAUSE
Clear the display line

Terminal Output

GOTO
Write_data
(PAGE 52)

Display "Unexpected error..."

Terminal Output

PAUSE
Clear the display line

Terminal Output

RETURN
Err_msg$ = "Command Interface Xmit error"

Clear the display line

Terminal Output!

Display
Err_msg$

Terminal Output!

Ctrl1 = BINAND(2, Ctrl)

\ CONTROL 12.2; Ctrl1 /

\ CONTROL 12.2; Ctrl1 /

Ctrl1 = BINITOR(1, Ctrl)

\ CONTROL 12.2; Ctrl1 /

Frlen1 = 4000

Frlen2 = 1000
Brumber = 2

Btime = .2

Dfref = (Fref2 - Fref1) / Brumber

F = 0

Bfreq = Fref1 + (Dfref * F)

BEEP the terminal bell @ Bfreq for Btime

Terminal Output

F = F + 1

No / IF

< F = Brumber - 1 >

Yes

Clear the display line

Terminal Output

RETURN
**SET_UP_TABLE**

Set up the values for a command table.

<table>
<thead>
<tr>
<th>Table</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>224</td>
</tr>
<tr>
<td>2</td>
<td>152</td>
</tr>
<tr>
<td>3</td>
<td>120</td>
</tr>
<tr>
<td>4</td>
<td>84</td>
</tr>
<tr>
<td>5</td>
<td>180</td>
</tr>
<tr>
<td>6</td>
<td>204</td>
</tr>
<tr>
<td>7</td>
<td>44</td>
</tr>
<tr>
<td>8</td>
<td>210</td>
</tr>
</tbody>
</table>
\begin{verbatim}
  V
  +----------+
  | Ctable(9) |
  | = 50      |
  +----------+

  +----------+
  | Ctable(10)|
  | = 74      |
  +----------+

  +----------+
  | Ctable(11)|
  | = 172     |
  +----------+

  +----------+
  | Ctable(12)|
  | = 136     |
  +----------+

  +----------+
  | Ctable(13)|
  | = 102     |
  +----------+

  +----------+
  | Ctable(14)|
  | = 30      |
  +----------+

  +----------+
  | Ctable(15)|
  | = 254     |
  +----------+

  RETURN
\end{verbatim}
If the user inputs an INVALID command...

Beep the terminal hell!

*Terminal Output*

```
Err_msg$ = ""
```

```
IF \ No Menu_flag = 1 \-------------------
  \ Yes

Err_msg$ = "from Menu #1"
```

```
IF \ No Menu_flag = 2 \-------------------
  \ Yes

Err_msg$ = "from Menu #2"
```

Display
"INVALID Command Entry"

*Terminal Output*
v

---

WAIT .5
Display
Err_msg
---

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 function keys

Printer is 1

Clear the screen

Terminal Output

Display "PROGRAM TERMINATED"

Terminal Output

END
Section 1–3
Section 1-3
DATA INPUT PROGRAM LISTING
* "6805" LIST

* NRL Tracker Interface

* Data Input to 6805 Program

* April 2, 1984

* Revised Sept 9, 1984 3:10 PM

0000 PORTA EQU 0000H LSByte output port

0001 PORTB EQU 0001H MSByte output port

0002 PORTC EQU 0002H Control port

0003 PORTD EQU 0003H Data input port

0011 LIMIT EQU 0011H Number of data bytes

0010 TABLE EQU 0010H Beginning of data table

0008 TUMTA EQU 0008H Timer data register

0009 TCONT EQU 0009H Timer Control register

0010 ORG 0100H

0100 A6 FF 21 INIT LDA #$0FFH Initialize

0102 B7 04 STA 004H DDRA

0104 B7 05 STA 005H DDRB

0106 A6 05 LDA #$085H

0108 B7 06 STA 006H DDRC

010A 3F 00 26 CLR PORTA Put zero's in Ports

010C 3F 01 27 CLR PORTB

010E 3F 02 28 CLR PORTC

0110 10 02 29 BSF 0,PORTC Set PFLG hi

0112 14 02 30 BSF 2,PORTC Set not_DAT_RDY hi

0114 5F 31 CLRX Clear TABLE offset

0115 9A 32 CLI Enable interrupts

0116 9D 34 WAIT NOP Loop till interrupt

0117 2D 35 BRA WAIT

0119 1F 02 41 REC BCLR 7,PORTC Set Byte-control=0

011B B6 03 42 READ LDA PORTD Transfer byte 1 to

011D E7 10 STA TABLE,X data table.

011F 5C 44 INX Increment to next byte

0120 1E 02 45 BSF 7,PORTC Set Byte-control=1

0122 B6 03 46 LDA PORTD Transfer byte 2 to

0124 E7 10 STA TABLE,X data table.

0126 5C 48 INX Increment to next byte

0127 1F 02 49 BCLR 7,PORTC Set Byte-control=0

0129 A3 11 50 CPX *LIMIT

012B 22 04 51 BMI TRANSFER Branch if last data

012D BF 70 52 STX 07DH Save count for return

012F 9A 53 CLI Enable interrupts

0130 80 54 RTI Go wait for next word.

* Transmit data from the data table to the HP9816 computer
<table>
<thead>
<tr>
<th>LOCATION</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
<th>OBJECT CODE</th>
<th>LINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0131 15 02</td>
<td>58</td>
<td>TRANSFER</td>
<td>BCLR</td>
<td>2, PORTC</td>
<td>Set not_Data_ready LO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0133 5F</td>
<td>59</td>
<td></td>
<td>CLKX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0134 0202 FD</td>
<td>60</td>
<td>LOOP1</td>
<td>BRSET</td>
<td>1, PORTC, LOOP1</td>
<td>Loop till PCTL=1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0137 E6 10</td>
<td>61</td>
<td>LOOP2</td>
<td>LDA</td>
<td>TABLE, X</td>
<td>Set LSbyte for out</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0139 B7 00</td>
<td>62</td>
<td>STAH</td>
<td>PORTA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>013B 5C</td>
<td>63</td>
<td>INCX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>013C E6 10</td>
<td>64</td>
<td>LDAH</td>
<td>TABLE, X</td>
<td>Increment to next byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>013E B7 01</td>
<td>65</td>
<td>STA</td>
<td>PORTB</td>
<td>Get MSbyte for out</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0140 5C</td>
<td>66</td>
<td>INCX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0141 11 02</td>
<td>67</td>
<td>BCLR</td>
<td>0, PORTC</td>
<td>Increment to next byte</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0143 0302 FD</td>
<td>68</td>
<td>LOOP3</td>
<td>BRCLR</td>
<td>1, PORTC, LOOP3</td>
<td>Set PFLAG low</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0146 14 02</td>
<td>69</td>
<td>BSET</td>
<td>2, PORTC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>014B 10 02</td>
<td>70</td>
<td>BSET</td>
<td>0, PORTC</td>
<td>Loop till PCTL0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>014A 0202 FD</td>
<td>71</td>
<td>LOOP4</td>
<td>BRSET</td>
<td>1, PORTC, LOOP4</td>
<td>Set not_Data_ready HI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>014D A3 11</td>
<td>72</td>
<td>CMPX</td>
<td>LIMIT</td>
<td>Loop till PCTL1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>014F 23 E6</td>
<td>73</td>
<td>BLS</td>
<td>LOOP2</td>
<td>Loop back if more data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0151 11 02</td>
<td>74</td>
<td>BCLR</td>
<td>0, PORTC</td>
<td>Set PFLAG low</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0153 A6 DF</td>
<td>75</td>
<td>LDA</td>
<td>0DFH</td>
<td>Delay for 9816</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0155 4A</td>
<td>76</td>
<td>LOOP5</td>
<td>DECA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0156 26 FD</td>
<td>77</td>
<td>BNE</td>
<td>LOOP5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0158 10 02</td>
<td>78</td>
<td>BSET</td>
<td>0, PORTC</td>
<td>Set PFLAG back high</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>015A 9A</td>
<td>79</td>
<td>CLI</td>
<td></td>
<td>Enable interrupts</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>015B 80</td>
<td>80</td>
<td>RTI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0F38 00</td>
<td>81</td>
<td>ORG</td>
<td>0F38H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>FCB</td>
<td>0000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>84</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>ORG</td>
<td>0FF8H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>86</td>
<td>FDB</td>
<td>INIT, REC, INIT, INIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>87</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>88</td>
<td>END</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Errors* 0
<table>
<thead>
<tr>
<th>LINE</th>
<th>SYMBOL</th>
<th>TYPE</th>
<th>REFERENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>INIT</td>
<td>A</td>
<td>86, 86, 86</td>
</tr>
<tr>
<td>12</td>
<td>LIMIT</td>
<td>A</td>
<td>50, 72</td>
</tr>
<tr>
<td>60</td>
<td>LOOP1</td>
<td>A</td>
<td>60</td>
</tr>
<tr>
<td>61</td>
<td>LOOP2</td>
<td>A</td>
<td>73</td>
</tr>
<tr>
<td>68</td>
<td>LOOP3</td>
<td>A</td>
<td>60</td>
</tr>
<tr>
<td>71</td>
<td>LOOP4</td>
<td>A</td>
<td>71</td>
</tr>
<tr>
<td>76</td>
<td>LOOP5</td>
<td>A</td>
<td>77</td>
</tr>
<tr>
<td>8</td>
<td>PORTA</td>
<td>A</td>
<td>26, 62</td>
</tr>
<tr>
<td>9</td>
<td>PORTB</td>
<td>A</td>
<td>27, 65</td>
</tr>
<tr>
<td>10</td>
<td>PORTC</td>
<td>A</td>
<td>28, 29, 30, 41, 45, 49, 50, 60, 67, 68, 69, 70, 71, 74, 78</td>
</tr>
<tr>
<td>11</td>
<td>PORTD</td>
<td>A</td>
<td>42, 46</td>
</tr>
<tr>
<td>42</td>
<td>READ</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>REC</td>
<td>A</td>
<td>86</td>
</tr>
<tr>
<td>13</td>
<td>TABLE</td>
<td>A</td>
<td>43, 47, 61, 64</td>
</tr>
<tr>
<td>15</td>
<td>TCONT</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>TDATA</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>TRANSFER</td>
<td>A</td>
<td>51</td>
</tr>
<tr>
<td>34</td>
<td>WAIT</td>
<td>A</td>
<td>35</td>
</tr>
</tbody>
</table>
Section 1–4
Section 1-4
COMMAND OUTPUT PROGRAM LISTING
FILE: NRL_CMD.NRL805  HEWLETT-PACKARD: 6805 Assembler
Mon, 25 Mar 1985, 16:56  PAGE 1

LOCATION OBJECT CODE LINE  SOURCE LINE
1 "6805".LIST
2 *  NRL Tracker Interface
3 *  Command Output To Tracker
4 *  April 30, 1984
5 *
6 *  Revised Sept 10, 1984  3:00 PM
7
8  PORTA EQU 0000H  Control Port
9  PORTB EQU 0001H  Serial Output (bit 0)
10  PORTC EQU 0002H  LSBYTE Input Port
11  PORTD EQU 0003H  MSBYTE Input Port
12  DATA EQU 0008H  Timer Data Register
13  TCONT EQU 0009H  Timer Control Register
14  TREG EQU 0010H  Temp Reg For Command
15  COUNT EQU 0011H  Bit Counter
16  DCOUNT1 EQU 0012H
17  DCOUNT2 EQU 0013H
18  DCOUNT3 EQU 0014H
19
20  NAME "NRL_CMD"
21
22  ORG 0100H
23
24  INIT LDA $0D9H  Initialize
25  STA 0004H  DDRA
26  STA $0FFH  DDRB
27  STA 0005H  DDRC
28  STA 0006H
29  CLR PORTA  Put zero's
30  CLR PORTB  in ports
31  CLR PORTC
32  BSER 0,PORTA  Set PFLG2 hi
33  LDA $07FH  Disable TIMER IRQ's
34  STA TCONT  and start timer clk
35
36  START BRCLR 5,PORTA,START  Loop till CTL0=1
37 ;
38  LOOP1 BRSET 1,PORTA,LOOP1  Loop till PCTL=1
39 ;
40  Valid command
41
42  BCLR 0,PORTA  Set PFLG2 low
43 ;
44  command acknowledge
45  PCTL1 BRCLR 1,PORTA,PCTL1  Loop till PCTL=0
46 ;
47  Command present
48
49  PCTL0 BRSET 1,PORTA,PCTL0  Loop till PCTL=1
50 ;
51  9816 acknowledges data ack
52
53  CD 01A1 JSR DELAY  Wait ~80 us
54  0 PORTA  Set PFLG low
FILE: NRL_CSD:NRL805  HEWLETT-PACKARD: 6805 Assembler
LOCATION OBJECT CODE LINE  SOURCE LINE

58 ;
59
012F A6 05  60  LDA  #05H
0.31 HE FF  61  LOOP7  LDX  #OFFH
0.33 CD 01A3  62  JSR  DELAY2
0.36 4A  63  DECA
0.37 26 F0  64  BNE  LOOP7
0.39 10 00  65  BSET  0,PORTA
0.3B CD 0186  66  JSR  OUTPUT
0.3E CD 0154  67  JSR  LOOP9
0.41 CD 0186  68  JSR  OUTPUT
70
71
0.44 0500 FD  72  LOOP6  BCLR  2,PORTA,LOOP6
0.47 A6 C8  73  LDA  #0CH
0.49 B7 08  74  STA  TDATA
0.4B 0500 3E  75  LOOP2  BCLR  2,PORTA,ERROR
0.4C E6 08  76  LDA  TDATA
0.4D 26 F9  77  BNE  LOOP2
0.4E 20 C2  78  BRA  START
79
0.54 0500 FD  80  LOOP9  BCLR  2,PORTA,LOOP9
0.57 A6 C8  81  LDA  #0CH
0.59 B7 08  82  STA  TDATA
0.5B 0500 04  83  LOOP10  BCLR  2,PORTA,LOOP11
0.5C B6 08  84  LDA  TDATA
0.5D 26 F9  85  BNE  LOOP10
86
0.62 0500 FD  86  LOOP11  BCLR  2,PORTA,LOOP11
wait for it to go away
87
88 * Output subroutine
90
0.66 A6 08  91  OUTPUT  LDA  #00H
0.68 B7 11  92  STA  COUNT
0.6A B6 10  93  LDA  TREG
0.6C B7 01  94  STA  PORTB
0.6E 16 00  95  BSET  3,PORTA
0.70 18 00  96  BSET  4,PORTA
0.71 0D 2D  97  BSR  DELAY
0.74 19 00  98  BCLR  4,PORTA
0.76 AD 29  99  BSR  DELAY
0.78 38 01 100  LSL  PORTB
0.7A 3A 11 101  DEC  COUNT
0.7C 26 F2 102  BNE  LOOP4
103
0.7E 17 00 104  BCLR  3,PORTA
0.80 A6 C8 105  LDA  #0CH
0.82 B7 08 106  STA  TDATA
0.84 0400 01 107  LOOPS  BRESET  2,PORTA,TIME
0.87 81 108  RTS
109
0.88 B6 08 110  TIME  LDA  TDATA
0.8A 26 F8 111  BNE  LOOPS
112
0.8C 1C 00 113  ERROR  BSET  6,PORTA
0.91 E000 F8 114  BCLR  5,PORTA ERROR
wait for 9118 to get
remove data ack
Loop delay 4 times
Setup delay
for ~10 ms
Loop till A=0
Set PFLG hi
Go to OUTPUT 1
Ignore first mismatch
Go to OUTPUT 2
Wait till ACK hi
Preset timer counter
to 200d.
Test for error ACK=0
Test for timeout
Still waiting
Command OK, start over
Wait till ACK hi
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 CHD in output port
Set CMD FLAG = 1
Set CMD CLK = 1
Delay
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 9118 to get
<table>
<thead>
<tr>
<th>LOCATION OBJECT CODE LINE</th>
<th>SOURCE LINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0191 0D 00</td>
<td>115</td>
</tr>
<tr>
<td>0193 3C</td>
<td>116</td>
</tr>
<tr>
<td></td>
<td>117</td>
</tr>
<tr>
<td>0194 AF 05</td>
<td>118</td>
</tr>
<tr>
<td>0196 AE FF</td>
<td>119_LOOP8</td>
</tr>
<tr>
<td>0198 CD 01H3</td>
<td>120</td>
</tr>
<tr>
<td>019B 4A</td>
<td>121</td>
</tr>
<tr>
<td>019C 76 F8</td>
<td>122</td>
</tr>
<tr>
<td>019E CC 0100</td>
<td>123</td>
</tr>
</tbody>
</table>

| 124                       |                      |
| 125                       |                      |
| 126                       |                      |
| 127                       | Delay overhead 16 cycles; delay loop time 0 cycles. |
| 128                       |                      |
| 01A1 AE 08               | 129_DELAY           |
| 01A3 5A                 | 130_DELAY2          |
| 01A4 26 FD               | 131                 |
| 01A6 81                 | 132                 |
|                         | 133                 |
| 01A7 AE 02               | 134_DEL             |
| 01A9 BF 14               | 135                 |
| 01AB 5F                 | 136                 |
| 01AC BF 13               | 137                 |
| 01AE BF 12               | 138                 |
| 01B0 3A 12               | 139_DEL1            |
| 01B2 26 FC               | 140                 |
| 01B4 3A 13               | 141                 |
| 01B6 26 F8               | 142                 |
| 01B8 3A 14               | 143                 |
| 01BA 26 F4               | 144                 |
| 01BC 81                 | 145                 |
|                         | 146                 |
| 0F30 37                 | 147                 |
|                         | 148                 |
|                         | 149                 |
|                         | 150                 |
| 0FF8 01000100            | 151                 |
|                         | 152                 |
|                         | 153                 |

Errors = 0

- BULR 6, PORTA
- RSP
- Loop delay 4 times
- Setup delay
- Delay for ~10 ms
- Abnormal exit from subroutine to CMD
- Delay loop
- Start delay loop
- Loop till X=0
- Setup Mask Option Register
- Initialize Vectors
- INIT, INIT, INIT, INIT
<table>
<thead>
<tr>
<th>Line</th>
<th>Symbol</th>
<th>Type</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>COUNT</td>
<td>A</td>
<td>92,101</td>
</tr>
<tr>
<td>16</td>
<td>DCOUNT1</td>
<td>A</td>
<td>138,139</td>
</tr>
<tr>
<td>17</td>
<td>DCOUNT2</td>
<td>A</td>
<td>137,141</td>
</tr>
<tr>
<td>18</td>
<td>DCOUNT3</td>
<td>A</td>
<td>135,143</td>
</tr>
<tr>
<td>134</td>
<td>DEL</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>139</td>
<td>DEL1</td>
<td>A</td>
<td>140,142,144</td>
</tr>
<tr>
<td>129</td>
<td>DELAY</td>
<td>A</td>
<td>56,97,99</td>
</tr>
<tr>
<td>130</td>
<td>DELAY2</td>
<td>A</td>
<td>62,120,131</td>
</tr>
<tr>
<td>113</td>
<td>ERROR</td>
<td>A</td>
<td>75,114</td>
</tr>
<tr>
<td>24</td>
<td>INIT</td>
<td>A</td>
<td>123,151,151,151,151</td>
</tr>
<tr>
<td>39</td>
<td>LOOP1</td>
<td>A</td>
<td>39</td>
</tr>
<tr>
<td>83</td>
<td>LOOP10</td>
<td>A</td>
<td>85</td>
</tr>
<tr>
<td>86</td>
<td>LOOP11</td>
<td>A</td>
<td>83,86</td>
</tr>
<tr>
<td>75</td>
<td>LOOP2</td>
<td>A</td>
<td>77</td>
</tr>
<tr>
<td>96</td>
<td>LOOP4</td>
<td>A</td>
<td>102</td>
</tr>
<tr>
<td>107</td>
<td>LOOPS</td>
<td>A</td>
<td>111</td>
</tr>
<tr>
<td>72</td>
<td>LOOP5</td>
<td>A</td>
<td>72</td>
</tr>
<tr>
<td>61</td>
<td>LOOP7</td>
<td>A</td>
<td>64</td>
</tr>
<tr>
<td>119</td>
<td>LOOP8</td>
<td>A</td>
<td>122</td>
</tr>
<tr>
<td>80</td>
<td>LOOP9</td>
<td>A</td>
<td>68,80</td>
</tr>
<tr>
<td>91</td>
<td>OUTPUT</td>
<td>A</td>
<td>67,69</td>
</tr>
<tr>
<td>53</td>
<td>PCTRL</td>
<td>A</td>
<td>53</td>
</tr>
<tr>
<td>45</td>
<td>PCTRL1</td>
<td>A</td>
<td>45</td>
</tr>
<tr>
<td>8</td>
<td>PORTA</td>
<td>A</td>
<td>29,32,36,39,42,45,50,53,57,66,72,75,80,83,86,95,96,98,104,107,113,114,115</td>
</tr>
<tr>
<td>9</td>
<td>PORTB</td>
<td>A</td>
<td>30,94,100</td>
</tr>
<tr>
<td>10</td>
<td>PORTC</td>
<td>A</td>
<td>31,48</td>
</tr>
<tr>
<td>11</td>
<td>PORTD</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>START</td>
<td>A</td>
<td>36,78</td>
</tr>
<tr>
<td>13</td>
<td>TCOUNT</td>
<td>A</td>
<td>34</td>
</tr>
<tr>
<td>12</td>
<td>TDATA</td>
<td>A</td>
<td>74,76,82,84,106,110</td>
</tr>
<tr>
<td>110</td>
<td>TIME</td>
<td>A</td>
<td>107</td>
</tr>
<tr>
<td>14</td>
<td>TREG</td>
<td>A</td>
<td>49,93</td>
</tr>
</tbody>
</table>
Section 1-5
Section I-5

PLOTXY PROGRAM LISTING
*PROGRAM PLOTXY*

written by Kris Parrish

```
DIM Sd(100,12),Read_string$(20),Clear$(2),Disp_flag$(8),Disp_msg$(60)
DIM File_string$(7),Disk_string$(14),Off_flag$(8),On_flag$(8),Title$(35)

INTEGER I
Data_count=0
Max_dat=20
Max_num_files=5
Files=0

File_string$="FILE_"
Disk_string$=":HP82901,700,1"
Read_string$="FILE_0:HP89201,700,1"

Inv_off=128
Inv_on=129

Off_flag$="ON/"&CHR$(Inv_on)&"OFF"&CHR$(Inv_off)
On_flag$=CHR$(Inv_on)&"ON"&CHR$(Inv_off)

Start_flag=0
Star2_flag=0
Star3_flag=0
Data_read=0
Print_graph=0

X_minimum=9999
X_maximum=-9999
Y_minimum=9999
Y_maximum=-9999

DEG
GINIT
GRAPHICS ON
PRINTER IS 1
```

```
> Main_menu

Define and display soft key functions
```

```
Main_menu:
ON KEY 0 LABEL "Re-display",3 GOTO Redisp_screen
ON KEY 1 LABEL "Star #1",3 GOSUB Star_1
ON KEY 2 LABEL "Star #2",3 GOSUB Star_2
ON KEY 3 LABEL "Star #3",3 GOSUB Star_3
ON KEY 4 LABEL "Data file ",3 GOSUB Data_file
ON KEY 5 LABEL "X vs Y",3 GOSUB X_vs_y
ON KEY 6 LABEL "X vs Time",3 GOSUB X_vs_time
ON KEY 7 LABEL "Y vs Time",3 GOSUB Y_vs_time
ON KEY 8 LABEL "Dump Graph",3 GOSUB Print_graphics
ON KEY 9 LABEL "EXIT",3 GOTO Shutdown
```

```
OUTPUT 2:Clear$:
```
Display menu options on the screen.

Menu_loop: !

CONTROL 1:31,1
PRINT "PLOT PROGRAM MENU"
CONTROL 1:30,3
PRINT "k0 Redisplay Screen (Clear Graphics)"
CONTROL 1:30,4
PRINT "k1 Star #1"
CONTROL 1:30,5
PRINT "k2 Star #2"
CONTROL 1:30,6
PRINT "k3 Star #3"
CONTROL 1:30,7
PRINT "k4 Data File #"
CONTROL 1:30,8
PRINT "k5 X vs Y"
CONTROL 1:30,9
PRINT "k6 X vs Time"
CONTROL 1:30,10
PRINT "k7 Y vs Time"
CONTROL 1:30,11
PRINT "k8 Dump Screen Graphics"
CONTROL 1:30,12
PRINT "to Printer"
CONTROL 1:30,13
PRINT "k9 Exit Program"
GOSUB Check_flags
GOTO Menu_loop

Check the status of the flags displayed on the screen.

Check_flags: !
CONTROL 1:42,4 ! Star #1
SELECT Star1_flag
CASE 0
Disp_flag$=Off_flag$
CASE 1
Disp_flag$=On_flag$
END SELECT
PRINT Disp_flag$
CONTROL 1:42,5 ! Star #2
SELECT Star2_flag
CASE 0
Disp_flag$=Off_flag$
CASE 1
Disp_flag$=On_flag$
END SELECT
PRINT Disp_flag$
CONTROL 1:42,6 ! Star #3
310 SELECT Star3_flag
320   CASE 0
330     Disp_flag$=Off_flag$
340   CASE 1
350     Disp_flag$=On_flag$
360   END SELECT
370 PRINT Disp_flag$
380!
390 DATA FILE # (Enabled-File read, Disabled- File not read)
400!
410 CONTROL 1:46,7
420 IF Data_read=1 THEN
430   PRINT "[";Nfiles;"]"
440 ELSE
450   PRINT " Disabled- file not read"
460 END IF
470 RETURN
480!
490 ! -----------------------------
500 ! - Data_file -
510 ! -----------------------------
520!
530 Get user input as to which data file
540 to use.
550!
560 Data_file: !
570 BEEP
580 INPUT "Please input file # > ",Nfiles
590 IF Nfiles>=0 AND Nfiles<=99 THEN Input_ok
600 BEEP
610 Disp_msg$="Invalid file # ",&VALS(Nfiles)&" Please, try again !"
620 GOSUB Out_msg
630 GOTO Data_file
640!
650 ! -----------------------------
660 ! - Input_ok -
670 ! -----------------------------
680!
690 Open and read data file, filling
700 the array with the star values.
710!
720 Input_ok: !
730 ON ERROR GOTO Error_tst
740!
750 Numbers=VALS(Nfiles)
770 MASS STORAGE IS "H:SP2901.700.1"
780 ASSIGN @Path_1 TO File_string$
790 DISP "Opening file > ":File_string$
800 Read_string$=File_string$&Disk_string$
810 ! ASSIGN #F_1 TO Read_string$
820 ENTER #F_1;Sd(*)
830 DISP "File read completed"
840 Data_read=1
850 ASSIGN #F_1 TO *
860 WAIT 1
870 DISP
880 OFF ERROR
890 RETURN
900!
910!
920 ! Any file errors are taken
930 ! care of in this routine...
940 ! Set up for concat. of filename
950 ! Create file name
960 ! Designate rt drive for data
970 ! Let user know what's going on
980 ! Read in the data
990 ! Let user know it is done
100 ! Show Data has been read...
101 ! Close the data file
102 ! Delay so user can read message
103 ! Clear display line
104 ! Turn off error trapping
105 ! Done!
This is used for testing of data file errors.

Error_test: !
OFF ERROR
BEEP
IF ERRN=56 THEN
DISP "File # ".Nfiles: " does not exist, press <CONT> to continue"
PAUSE
ELSE
IF ERRN=80 THEN
DISP "Disc not changed or NOT located in the right hand drive..."
WAIT .5
DISP "Press <CONT> after placing correct disc in right drive"
PAUSE
ELSE
CONTROL 1:1,24
PRINT "Unexpected error (";ERRN;") consult list of errors"
PRINT "and correct problem... press <CONT> to try again!"
PAUSE
END IF
END IF
GOTO Data_file

X_vs_y:
Plot X position vs Y position

IF Data_read=0 THEN
Disp_msg$="You must first read in a data file before you can plot..."
GOSUB Out_msg
RETURN
END IF
OUTPUT 2:Clear$:
FRAME
WINDOW -5.270,-5.270
AXES 2.2,-5.5,-5.5
Npts=0
FOR I=1 TO 100
IF Sd(I,10)=0 AND Sd(I,11)=0 AND Sd(I,12)=0 THEN 2300
Npts=Npts+1
NEXT I
IF Npts=0 THEN
Npts*=
END IF

LABEL THE X AXIS
X_maximum=260
Step_size=20
X_offset=0
GOSUB Label_x
! 2400 ! LABEL THE Y AXIS
2410  Y_maximum=260
2420  Step_size=20
2430  Y_offset=12
2440  GOSUB Label_y
2450 ! LABEL THE PLOT
2460  title="X vs Y from "&File_string$
2470  Location_x=137
2480  Location_y=255
2490  GOSUB Label_plot
2500  IF Star1_flag=0 THEN Plot_star_2
2510    FOR I=1 TO Npts
2520      Plot_x(I)=Sd(I,1)
2530      Plot_y(I)=Sd(I,2)
2540    NEXT I
2550    Line_type=1
2560    _line_label$="STAR #1"
2570    GOSUB Plot_star
2580  * ------------------------------- *
2590  +  Plot_star_2  +
2600  * ------------------------------- *
2610  Plot_star_2:
2620  ! IF Star2_flag=0 THEN Plot_star_3
2630    FOR I=1 TO Npts
2640      Plot_x(I)=Sd(I,4)
2650      Plot_y(I)=Sd(I,5)
2660    NEXT I
2670    Line_type=5
2680    _line_label$="STAR #2"
2690    GOSUB Plot_star
2700  * ------------------------------- *
2710  +  Plot_star_3  +
2720  * ------------------------------- *
2730  Plot_star_3:
2740  ! IF Star3_flag=0 THEN RETURN
2750    FOR I=1 TO Npts
2760      Plot_x(I)=Sd(I,7)
2770      Plot_y(I)=Sd(I,8)
2780    NEXT I
2790    Line_type=3
2800    _line_label$="STAR #3"
2810    GOSUB Plot_star
2820    RETURN
2830  * ------------------------------- *
2840  +  Star_1  +
2850  * ------------------------------- *
2860  ! Toggle star #1 flag
2870  Star_1:
2880  ! Star_flag=1-Star_flag
RETURN

- Star_2 -

Toggle star #2 flag

Star_2!: Star_flag=1-Star2_flag

RETURN

- Star_3 -

Toggle star #3 flag

Star_3!: Star3_flag=1-Star3_flag

RETURN

X_vs_time

Plot X position vs time

IF data_read=0 THEN
  GOSUB Out_msg
  RETURN
END IF

FRAME
CTRL "Clear$":
"Determination MINIMUM and MAXIMUM values" & CHR$(Inv_on)
X_max,m=9999
! Initialize Maximum value
X_min,m=9999
! Initialize Minimum value
Note=0
J=1: TC 100
IF Sd(I, 0)+0 AND Sd(I, 11)+0 AND Sd(I, 12)+0 THEN 3490
Note=Note+1
TEST_data(I)=Sd(I, 12)+(60*Sd(I, 11))
IF X_max,m<TEST_data(I) THEN X_max,m=TEST_data(I)
IF X_min,m>TEST_data(I) THEN X_min,m=TEST_data(I)
END IF
IF Note=0 THEN
  Note=
  X_max,m=60
  X_min,m=0
END IF
X_max,m=X_max,m+20
WINDOW X_min,m, X_max,m, -5, 270
-5 5 5 5 5

Label the X axis
Step_size=10
X_offset=15
GOSUB Label_x
Label the Y axis
Y_max=260
Step_size=20
Y_offset=X_min+15
GOSUB Label_y
Label the plot
Title="X vs TIME from "&File_string$
Location_x=X_min+125
Location_y=255
GOSUB Label_plot
START PLOTTING...
CONTROL 1:20.18
PRINT 
IF Star1_flag=0 THEN Plot_x_star2
FOR I=1 TO Npts
   Plot_x(I)=Test_data(I)
   Plot_y(I)=Sd(I,1)
NEXT I
Line_type=1
Line_labels="STAR #1"
GOSUB Plot_star
***************
- Plot_x_star2 -
***************
Plot_x_star2:
IF Star2_flag=0 THEN Plot_x_star3
FOR I=1 TO Npts
   Plot_x(I)=Test_data(I)
   Plot_y(I)=Sd(I,4)
NEXT I
Line_type=5
Line_labels="STAR #2"
GOSUB Plot_star
***************
- Plot_x_star3 -
***************
Plot_x_star3:
IF Star3_flag=0 THEN RETURN
FOR I=1 TO Npts
   Plot_x(I)=Test_data(I)
   Plot_y(I)=Sd(I,7)
NEXT I
Line_type=3
Line_labels="STAR #3"
GOSUB Plot_star
RETURN
Plot Y position vs time

Y_vs_time!

---

Y_vs_time!!

IF Data_read=0 THEN
Disp_msg$="You must first read in a data file before you can plot..."
GOSUB Out_msg
END IF
OUTPUT 2;Clear$;
FRAME
CONTROL 1;20,18
PRINT CHR$(Inv_on)"Determining MINIMUM and MAXIMUM values"&CHR$(Inv_off)
X_maximum=-9999
X_minimum=9999
Npts=0
FOR I=1 TO 100
IF Sd(I,10)=0 AND Sd(I,11)=0 AND Sd(I,12)=0 THEN 4450
Npts=Npts+1
Test_data(I)=Sd(I,12)+(60*Sd(I,11))
IF X_maximum<Test_data(I) THEN X_maximum=Test_data(I)
IF X_minimum>Test_data(I) THEN X_minimum=Test_data(I)
NEXT I
IF Npts=0 THEN 4470
Npts=1
X_maximum=60
X_minimum=0
END IF
X_maximum=X_maximum+20
WINDOW X_minimum, X_maximum,-5,270
AXES 2,2,X_minimum,-5,5,5
Label the X axis
Step_size=10
X_offset=15
GOSUB Label_x
Label the Y axis
Y_maximum=260
Step_size=20
Y_offset=Y_minimum+15
GOSUB Label_y
Label the plot
Title$="Y vs TIME from 
Location_x=X_minimum+125
Location_y=255
GOSUB Label_plot
START PLOTTING Y vs TIME...
CONTROL 1;20,18
PRINT 
IF Star1_flag=0 THEN Plot_y_star2
FOR \( I=1 \) TO Npts
   \( \text{Plot}_x(I)=\text{Test}_\text{data}(I) \)
   \( \text{Plot}_y(I)=\text{Sd}(I,2) \)
NEXT \( I \)

\( \text{Line}_\text{type}=1 \)
\( \text{Line}_\text{label}="\text{STAR} \#1" \)
GOSUB Plot_star

\( \text{Plot}_y\_\text{star}2 \)

\( \text{Plot}_y\_\text{star}3 \)
IF \( \text{Star}_3 \_\text{flag}=0 \) THEN RETURN
FOR \( I=1 \) TO Npts
   \( \text{Plot}_x(I)=\text{Test}_\text{data}(I) \)
   \( \text{Plot}_y(I)=\text{Sd}(I,5) \)
NEXT \( I \)
\( \text{Line}_\text{type}=5 \)
\( \text{Line}_\text{label}="\text{STAR} \#2" \)
GOSUB Plot_star

\( \text{Plot}_y\_\text{star}3 \)
IF \( \text{Star}_3 \_\text{flag}=0 \) THEN RETURN
FOR \( I=1 \) TO Npts
   \( \text{Plot}_x(I)=\text{Test}_\text{data}(I) \)
   \( \text{Plot}_y(I)=\text{Sd}(I,8) \)
NEXT \( I \)
\( \text{Line}_\text{type}=3 \)
\( \text{Line}_\text{label}="\text{STAR} \#3" \)
GOSUB Plot_star
RETURN

\( \text{Print}_\text{graphics} \)

\( \text{Label}_x \)
\( \text{LORG} 6 \)
\( \text{CSIZE} 3,7 \)
\( \text{LDIR} 90 \)
FOR \( I=0 \) TO X maximum STEP Step_size
   MOVE \( I,X\_\text{offset} \)
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 1–6
Section 1-6
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 or 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 RPTS. 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.
103 Item has no corresponding IMAGE. The image specifier has no fields that are used for item processing. Specifiers such as \* / are not used to process the data for the item list. Item-processing specifiers include things like \* 0 8 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 CHRS(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 READIOWRITEIO to nonexistent memory location.

136 REAL underflow. The input or result is closer to zero than $10^{-20}$ (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.
Duplicate line label in context. There cannot be two lines with the same line label in one context.

Illegal interface select code or device selector. Value out of range.

Panty error

Insufficient data for ENTER. A statement terminator was received before the variable list was satisfied.

String greater than 32,767 bytes in ENTER.

Improper interface register number. Value out of range or negative.

Illegal expression type in list. For example, trying to ENTER into a constant.

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.

Improper image specifier or nesting images more than 8 deep. The characters used for an image specifier are improper or in an improper order.

Numeric data not received. When entering characters for a numeric field, an item terminator was encountered before any "numeric" characters were received.

Attempt to enter more than 32,767 digits into one number.

Interface not present. The intended interface is not present, set to a different select code, or is malfunctioning.

Illegal BYTE, WORD operation. Attempt to ASSIGN with the WORD attribute to a non-word device.

Image specifier greater than dimensioned string length.

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.

Device timeout occurred and the ON TIMEOUT branch could not be taken.

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.

Illegal I/O addressing sequence. The secondary addressing in a device selector is improper or primary address too large for specified device.

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.

Active or system controller required. The HP-IB is not active controller and needs to be for the specified operation.

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.

Undefined I/O path name. Attempting to use an I/O path name that is not assigned to a device or file.

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 MOVELINES 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 \( P2 < = P1 \) 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.
Programmable only: cannot be executed from the keyboard.

Command only: cannot be stored as a program line.

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.

Too many symbols in this context. Symbols include variable names, I/O path names, COM block names, subprogram names, and line identifiers.

Too many subscripts: maximum of six dimensions allowed.

Wrong type or number of parameters. An improper parameter list for a machine-resident function.

Invalid quoted string.

Invalid line number: must be a whole number 1 thru 32 766.