INSTITUTE OF OCEANOGRAPHIC SCIENCES
DEACON LABORATORY

INTERNAL DOCUMENT No. 309

GPS and ship head recording system: technical reference for hardware interface and software

R W Pascal, K G Birch & A L Williams

1992

Wormley
Godalming
Surrey GU8 5UB
Tel 0428 684141
Telex 858833 OCEANS G
Telefax 0428 683066

1. SYSTEM OVERVIEW

The system described in this manual was developed in response to a requirement for navigational data on 'Vessels of Opportunity' where there are no data available or where it is not possible to gain access to the data.

The remit for the system is to acquire and record navigational data, with the secondary function to display the data for operational use. The system must by nature of its portability relies on the vessel for only its basic requirements ie stowage and mains power. Independence from ship support means that the system does not require operator intervention, unless so desired, for the duration of deployment.

The major components of the system are :-

IBM Personnel Computer or Clone

GPS Receiver

Fluxgate Compass

with the PC acting as controller and data recording medium for the data from the GPS Receiver and Fluxgate Compass.

2. SYSTEM COMPONENTS

The PC clone is primarily responsible for data collection and storage with a secondary function as a data display. The primary screen displays, in digital format, the Latitude, Longitude, Ships Head, together with a analog compass rose. Logging status and file house keeping information is also displayed

Interactive software allows the user (on user supported cruises) to control the logging functions and also enables either of the raw data strings to be viewed for diagnostic purposes.

The PC clone is upgraded with a Dual RS232 Serial data interface installed in the expansion bus, to allow simultaneous data collection of the two asynchronous data strings.

Data are stored on PC's two magnetic media ie Hard and Floppy Disk. Dual media storage provides data security in event of failure by either of the disc drives. The primary storage of time, position and heading is on the Hard Disc together with full satellite empheris data. The secondary data set are stored on the Floppy disc and contains only the time, position and heading data. The file error checking will ensure that in the event a single disk drive failure the data will continue to be logged to the alternate drive. In the event of both disc drives failing the program will continue to operate without data being recorded.

Primary data, stored on the Hard Disk, is written at '1' minute intervals in files containing 360 data values. The file names are Time and Data stamped derived from the PC clock, each '1' minute data string is also time stamped based on the time of the GPS Fix. The stored data string contains the '1' minute values together with quality control empheris parameters for the GPS position fixes, which can be used to further filter data at latter stage in the data processing.

Secondary data files are written at half the primary data rate, ie every '2' minutes

The formats of the file name and of the data string are of fixed length, a detailed description can be found in Sections 8 and 9

The data files are written in ASCII and can be ready directly by an editor or word processor ie WORD.

1. GPS PROGRAM DISCRIPTION

1.1. Introduction

The GPS software has been written in GWBASIC for an IBM PC or compatible, and requires the PC to have two serial COMS ports. Data is received at 4800 baud at a rate of 1 hz from both the MX4200 GPS Receiver and the KVH Fluxgate Compass, merged, and then written to disc. Both the GPS receiver and the Compass transmit there data under the NMEA 0183 interface standard. Each record has a five character address which labels it as a GPS message i.e. PMVXG=GPS and HCHDM=Compass. The GPS receiver outputs many different types of records, each record type containing different navigation parameters. The first data field after the record address is an index code which identifies the specific type and format of the record. The software identifies the index code and then processes the record according to the data format within that record. In this way the navigational data logged by the program is continuously updated as the various record types are detected. Once a minute the datafile is updated by the program with the current values for the navigational information. The file name for the datafile is determined by the date and time at which the file was originally opened, each file is then appended to for 6 hours where upon that file is closed and a new file opened.

1.2. Software operation

When the PC is powered up or re-booted, commands in the AUTOEXEC.BAT configure the PC and load the software in the required manor. The software requires the NUMLOCK, and CAPSLOCK keys to be set OFF on the keyboard, and for a number of GWBasic environment conditions to be set. On some PC the keyboard can be configured by commands in the AUTOEXEC.BAT such as :-

SETKBD NUMLOCK = OFF

SETKBD CAPSLOCK = OFF

When GWBasic is loaded by the PC certain switch parameters need to be set to enable the program to operate correctly. This is achieved by using the following command to load GWBasic and the program in the AUTOEXEC.BAT.

GWBASIC GPS12 /C:32767 /F:4

GPS12 loads the current version of the GPS program.

/C:32767 switch allocates 32767 bytes to the receive buffer and 128 bytes to the transmit buffer for the RS-232 communications.

/F:4 switch sets the maximum number of files that may be opened simultaneously during the execution of the program.

1.3. Program sections

Main Program

Communications

Display

Files

Error Handling

1.3.1. Main Program lines 10 - 530

When the program is first executed it runs through a number of routines and statements, which initialises and prepares the system for operation. These include, programming the control keys, opening the communication lines, intialising the GPS receiver, and opening files on both the hard and floppy disk's. When all these processes have been achieved, the program operates within a loop which reads in both the GPS and Compass data, and displays it on the screen. This loop can only be interupted in two ways, an interrupt driven timer, and control keys effected by an operator from the keyboard. The timer is set to produce an interrupt every 60 seconds. The interrupt redirects the program to the Write a Record routine, from which the current values for the GPS and Compass parameters are written to disk..

There are three control keys which may be active while the program is in the data input loop :-

ALT-F1	Open files for logging
ALT-F2	Stop logging
Alt-Esc	Switch to the command screen

ALT-F1 key and the ALT-F2 key are mutually exclusive as only one of these keys is active at a time. When the program first starts, data logging is automatically started and the ALT-F1 key is disabled. The ALT-ESC key transfers the program to the routine which displays additional commands and can display the contents of both comms ports.

line 20 CLEAR ,, 32767

Zeros all the variables, nulls the string variables, allocates 32767 bytes for stack space, and uses all available memory in the segment.

line 40 ON ERROR GOTO 20000

Sets the error trap and directs the program to the error trapping routine which starts on line 20000. This enables the program to resume execution after an error.

line 60 initialsee the string array RECFIELD$, which stores the data which is writern to disk., and some system parameters. HDRIVE$ is set to the harddisk drive letter (D:) and FDRIVE$ is set to the floppy disk drive letter (A:). Only these values need be changed to reconfigure the program for a different PC setup.

line 80 Branch to subroutine set up display screen. This produces the outline of the display on the screen.

lines 100 - 130 Assigns a KEY number between 15 - 20, to allow certain key combinations to be trapped when pressed.

KEY 15 ,CHR$(&H8)+CHR$(&H3B) ALT-F1

KEY 16 ,CHR$(&H8)+CHR$(&H3C) ALT-F2

KEY 17 ,CHR$(&H8)+CHR$(&H1) ALT-ESC

KEY 18 ,CHR$(&H8)+CHR$(&H1C) ALT-ENTER

LINES 140 - 170 Branch to a routine when a KEY code has been trapped.

KEY (15) Open file for logging

KEY (16) Stop logging data to disk

KEY (17) Display additional commands

KEY (18) Set up screen

lines 180 - 210 Turn trapping ON or OFF for the KEY codes between 15 -18.

lines 220 - 250 Branch to a routine when a KEY code has been trapped.

KEY (1) F1 - Display contents of COM1

KEY (2) F2 - Display contents of COM2

KEY (5) F5 - Restart GPS system

KEY (6) F6 - Initialise GPS parameters

line 260 - 290 Set KEYs 1,2,5,6 OFF.

lines 320-330 Open coms ports. COM1 is connected to the GPS and is required to receive and send data to the GPS. COM2 is connected to the Compass and just receives Compass data.

Open COM1 port to receive the GPS data at 4800 baud, no parity , 8 data bits, 1 stop bit, data set ready, file number 1.

Open COM2 port to receive Compass data at 4800 baud, no parity , 8 data bits, 1 stop bit, clear to send, data set ready, file number 2.

lines 350 - 370 initialise values used in the initialise GPS parameters routine. These are the date and time taken from the PC and modified to the format required by the GPS, and a default LAT and LON.

line 390 Branch to the flush buffers routine.

line 410- 440 These lines initialise the GPS with recommended paramaters which are different to the default paramenters set in the GPS unit. This is followed by a small delay and then the text "Initialising GPS navigation mode..." is removed by pronting a blank line. The GPS is iniatised by the Write #1 statement, #1 directs the output to the COM1 port. The $PMVXG,001,2,0,1.00,1.00,0010,0015,05,U,0 record is used to specify the various navigation parameters for the GPS. This is sent via the COM1 port, and is followed by a Carrage Return Line Feed.

Explanation of message fields

$PMVXG,001	Specifies the message type
2	2D navigation - minimum of 3 satellites required
0	unused
1.00	Horizontal acceleration factor (M/Sec**2)
1.00	Vertical acceleration factor (M/Sec**2)
0010	VDOP limit - used with 3D navigation
0015	HDOP limit - 2D navigation is suspended when HDOP exceeds this limit
05	Elevation limit - Satellite not used if elevation is below this limit
U	UTC time output selected
0	Loacl time offset (can be + or -) from GMT

lines 453 - 460 Open data files when program is first started. line 543 and 456 reads a GPS message untils the satellite time is obtained. This makes sure that the time varables used by the program have valid values before the program branches to the Open files routine on line 460.

lines 470 - 510 The y consist mainly of a WHILE WEND loop in which the program resides while aquiring data. This loop should never be ended, and the only process within the loop is a branch to Read GPS data.

1.3.2. Communications lines 1000 - 2400

In the Communicattions section there are a number of routines, and these primaraly deal with reading the COMMs ports and sorting out the messages after they have been read. The system uses two COMMs ports, one for two way communications to the GPS and the other to read the messages produced by the Compass. The two sets of data run asychrously, but both output data at a one second rate, although for the GPS this means a fix every second therefore it produces about 5 messages per second. The compass is read every time a code 003 message is recieved from the GPS, which occur approximaty every 5 Seconds. Both sets of information are kept in the string arry RECFIELD$, which is updated each time a message is recieved. The data in the arry is the written to hard disk every minute, giving spot reading of both compass and GPS data every minute. A subset of the information is written to floppy disk every two minutes to provide a backup data set, one disk containing about 45 days of data.

1.3.2.1. Read a GPS data record

lines 1030 - 1090 The first character of the GPS data must be checked to see that it is the start of a record . This is done by getting the character BYTE$ from the COM1 port, and checking it against the start of record character "$". When these are equal the start off

record has been reached, and the full message GPS$ can be received until BYTE$=CHR$(10), which signifies the end of a record.

line 1100 The program now branches to sortout which record has been received, and process the message acordingly.

line 1110 If the COM1 buffer is greater than 128 bytes then the program branches to the Flush Buffers routine.

1.3.2.2. Take a compass reading

line 1150 If the COMM2 buffer is empty i.e. no data then set the compass value to 999.9.

lines 1160 - 1230 The first character of the compass data must also be checked to see that it is the start of a record . This is done by getting the character COMPBYTE$ from the COM2 port, and checking it against the start of record character "$". When these are equal the start off record has been reached, and the full message HEADING$ can be received until COMPBYTE$=CHR$(10), which signifies the end of a record.

lines 1240 - 1280 The string varable HEADING$ which contains a record from the compass, is used to derive three different varables used in the program. A compass record , like the GPS data, is in the NMEA 0183 format and therefore contains a five character address which labels it as a GPS message i.e. HCHDM.

$HCHDM, 106.9, M, *0B Where 106.9 is the compass heading

This value is loaded into the numerical variable HEADING and is used in the mian display. The heading plus the character address is loaded into the string variable COMM2$, and is used when the COM2 ports data is to be displayed. The data stored in the array RECFIELD$ is just the heading information.

The string variable HEADING$ is then set to a null string.

line 1290 Branch to the Flush Buffers routine.

1.3.2.3. Draw Compass Needle

The routine starts by erasing the previous compass outline and unfilling, then the variable PICTURE$ is set to a null string. The variable TURN$ is generated from the characters "TA-" plus the current compass heading, this is then used to generate a new value for PICTURE$. PICTURE$ is a string of commands which are used to draw the compass needle shape, and the only value that changes within this is TURN$, which is included in the list of commands. The PSET command is used to specify that the coordinates given in PICTURE$ are relative to the last point drawn. The DRAW command is used to draw the figure of the compass needle on the display screen, based on the commands given in PICTURE$. The figure is then filled, and the outer circles are redrawn.

1.3.2.4. Flush buffers

Flush buffers tests for the number of characters in both of the COMMS port buffers, and then reads that many characters in from each buffer.

1.3.2.5. Sort out which message has been received

When a GPS message has been read in from the COM1 port the program branches to this routine so that the message can be processed. Within this routine there are subroutines for each message type recieved, so that, depending on the message code number, the program branches to the required routine. When the code is either 008 or 018 the flag FLAG is set to 0 or 1, as both codes use the same subroutine. If the display option is set to 1 the COM1 data is loaded in to the string COMM1$, ready to be displayed from the additional commands display.

Options within the routine :-

LOGGING If true logging enabled

DISPLAY If = 1 standard display. If = 0 additional commands display.

FLAG If =1 code 008. IF = 0 code 018

Status information Code 000. This message gives the status of the reciever, wheither navigating or tracking satellites etc. It also provides information on the number of satellites that should be visible, the number of satellites actually visible, and the time since the reciever was last navigating. These parameters are are read out of the GPS message and stored in variables so that they can be printed to the display, and stored in the data array RECFIELD$ if the logging is enabled.

Position information Code 001. From this message the positional information is gathered, time of fix, lat, lon, and the type of navigation ie 2D. The time of fix is loaded into the variable HHMMSS$ with colan seperaters and is used for the display and updating the PC time. If logging is enabled then the Julian day is added to the time string and stored in RECFIELD$. The lat and lon values are read into variables with a space character seperating the degrees from the minutes to make the display clearer. The comma between tenths of seconds and the direction indicator (N or S, W or E) si removed when the values are stored in RECFIELD$.

DOPs information Code 003. The DPOs in this message correspond to the optimum constellation selected by the MX 4200, and not the actual satellites being tracked. The message provides, DOP east, DOP north, DOP vertical, and HDOP. These values are loaded into variables to be dispalyed on the screen and into RECFIELD$ when logging is enabled.

At the end of the routine, the program branches to read a compass message.

Speed and heading information Code 011. This provides course over the ground (COG$) and speed over the ground (SOG$), which are loaded into variables for diplay and stored in RECFIELD$ when logging.

Satellite 1-8 signal strength and location Code 008 and 018. Code 008 equals satellite signal strength and location for satellites 1 - 4, and code 018 is for satellites 5 - 8. This information is not displayed, therefore when logging is not enabled the program exits from the routine. The variable SATTRACK$, derived from message code 000, is used to determine which sections need to be read from the message, and there are two FOR NEXT loops which read the data depending on the message type. A third FOR NEXT loop pads out RECFIELD$ with zeros when less than 8 satellites are being tracked.

1.3.2.6. Julian day number

This routine is used to convert the date from the PC into a Julian day number, which is included in the data and used as part of the filename. The day and month are taken from the date string and converted into intergers. These are then included into an equation which generates the julian day as a string call JDAY$. This does not take a leap year into consideration. This number for the julian day is then converted into a three digit number by pading out with leading zeros when the number is less than three digits long. The final result is a string call JDAY$ which made up from a three digit number with leading zeros such as : 003, 033, 333.

1.3.3. Display line 5000 - 6600

The display for the GPS has two seperate modes, a screen to display navigational data, and a screen to display display additional information. The first mode is the standard display which shows the digital values of the navigation data from the GPS, and the compass heading which is displayed graphically as a compass needle. The second mode enables the user to examine raw data in real time from both comms ports, as well as sending specific commands to the GPS reciever. These commands will either restart the GPS, initialise GPS parameters, or assign new time, position and antenna height tothe GPS reciever.

1.3.3.1. Set up screen

This routine sets up the screen to display the navigation data by drawing all the outlines and printing the headings for the various navigation parameters. The control keys are set to there required values, key 17 is enabled so that the screen mode can be changed if nessisary. The program draws the border and then prints the tittle at the top of the screen.

Draw compass and annotate. The double compass circle is draw in the center of the screen, with 15 degree tick marks and then the four cardinal points are printed.

Draw boxes to hold other information. The routine Left box draws a box on the left hand side of the screen for the compass heading and the GPS latitude and longitude readings. The routine Right box draws a box on the right hand side of the screen for the satellite information, which includes, UTC time, Satellites tracked, Satellites visible, Course over the ground, Speed over the ground, and the N, E, V, H DOP values.

Inform user of keys for logging data, prints to the screen the command keys that are active and there functions, which are:-

ALT/F1 - start logging

ALT/F2 - sttop logging

ALT/ESC - command screen

If data is being logged then "logging data" is displayed with the filename of the data file. The drive specification is included in the filename, so that you can tell to which drive the main data is being written.

1.3.3.2. Display additional commands and the contents of both comms ports

This routine will either display data arriving at the comms ports, or allow the user to reset certain parameters in the GPS receiver. The routine starts by refreshing the screen and then enabling the command keys which are used for this screen display. A box is draw which will contain any data when display the contents of a comms port is selected. Above the box a list of the command keys and their functions is printed to inform the user of the option available within this screen. These are:-

ALT/ ENTER	Return to main display
F1	Display contents of COM1
F2	Display contents of COM2
F5	Restart GPS system
F6	Initialise GPS parameters

Display contents of COM1, prints within the boundary of the box the contents of the COM1 port. The data arriving at the COM1 port is normally the GPS data, and consists of many different messages arriving at a rate greater than one per second. As the data rate is high, the messages pass through the screen quickly, but this can be useful to help determine wether the GPS is functioning in the correct mode.

Display contents of COM2, displays the compass data within the box on the screen. The data arrives at exactly one message per second, and the messages are relatively short, so that there is little problem in viewing the data.

Restart GPS sends a message to the GPS receiver which causes the receiver to terminate operations and restart. The current values as entered by the user are preserved, so the effect is much the same as a power fail would be. When the message "$PMVXG,018,W" has been sent to the receiver " Initialising GPS " is printed on the screen, and there is a short delay before the program continues.

Initialise GPS parameters allows the user to enter values for Time, Date, Lat, Lon, and altitude of the antenna, and assign them to the GPS receiver. These are printed to the screen one at a time with default values, the user can either use these or enter in a new value. The routine is initated by pressing the F6 command key, this results in the heading "Initialising GPS Parameters" being printed to the screen along with the first parameter TIME. The default values are accepted by pressing carriage return, or a new value can be entered. The program steps througth each parameter in turn and then once all the values have been entered the user is promted to press "Y" or "N" to confirm that these values are correct. The default value for TIME is generated by using the PC TIME$ fuction and DATE$ for the date. The latitude, logitude and altitude have values specified at the start of the program on line 370.

Assign values to the GPS is the routine which sends the information entered by the user in the initialising GPS parameteres routine, to the GPS receiver. It is written to the COM1 port in the form of the message $PMVXG,000,date,time ,lat, NorS, lon,WorE, alt, with all the parameters amaliglimated into one string. When the message has been sent the box on the screnn is refreshed and a message " Assigning new parameters now...." is printed to

the screen. This is followed by a small delay while the GPS receiver processes the new information, after which the screen message is erased.

1.3.4. Files line 10000 - 11200

1.3.4.1. Open files for logging data

Disable Alt-F1 enable Alt F2

If new disk is inserted then obtain its details

Establish free space on disk

Write 6 hours worth of records (360)

Establish the filename and open

1.3.4.2. Write record to disk

First remove commas from fields 13 to 20

Write a single record

Update the record count and check if a new file is needed

Calculate percentage of file written

Calculate time to end of disk

1.3.4.3. Stop logging data to disk

Disable Alt F2 enable Alt F1

1.3.4.4. Check disk space

Disk might be empty, so create a file so as to get the number of bytes free when Using the FILES command

Wait 10 seconds to enter if space is ok else continue

1.3.5. Error handling line 20000 - 20160

1.3.5.1. Device I/O Error - Device Timeout

Communication buffer overflow

1.3.5.2. Disk and File errors

1.4. GWBasic Program Listing

```
10 REM *** GPS/Compass data receiving system ***
20  CLEAR,,4000
30 REM ** Set up error handling facility **
40 ON ERROR GOTO 20000
50 REM *** Initialise some values
60 DIM RECFIELD$(21) : RECPLT=0 : LOGGING=0 : NEWDISK=1 : IX=0 :HDRIVE$="D:":
FDRIVE$="A:":FLAGH=0:FLAGF=0
70 REM ** First set up display screen **
80 GOSUB 5000
90 REM ** Set up program control keys **
100 KEY 15,CHR$(&H8)+CHR$(&H3B)
110 KEY 16,CHR$(&H8)+CHR$(&H3C)
120 KEY 17,CHR$(&H8)+CHR$(&H1)
130 KEY 18,CHR$(&H8)+CHR$(&H1C)
140 ON KEY(15) GOSUB 10000
150 ON KEY(16) GOSUB 10650
160 ON KEY(17) GOSUB 5600
170 ON KEY(18) GOSUB 5000
180 KEY(15) ON
190 KEY(16) OFF
200 KEY(17) ON
210 KEY(18) OFF
220 ON KEY(1) GOSUB 5880
230 ON KEY(2) GOSUB 6020
240 ON KEY(5) GOSUB 6150
250 ON KEY(6) GOSUB 6210
260 KEY(1) OFF
270 KEY(2) OFF
280 KEY(5) OFF
290 KEY(6) OFF
300 REM ** Open COM lines for data input **
310 REM ** COM1=GPS data COM2=Compass data **
320 OPEN "COM1:4800,N,8,1,DS0,ASC" AS #1
330 OPEN "COM2:4800,N,8,1,CS0,DS0,ASC" AS #2
340 REM ** Initialise some values **
350 DDMMYY$=MID$(DATE$,4,2)+","+LEFT$(DATE$,2)+","+RIGHT$(DATE$,4)
360 GMT$=(LEFT$(TIME$,2))+(MID$(TIME$,4,2))+(RIGHT$(TIME$,2))
370  LAT$="5000.000,N":LON$="00008.000":ALT$="00100.0,W"
380 REM ** Flush comms buffers **
390 GOSUB 1410
400 REM ** Initialise GPS navigation mode **
410 LOCATE 23,6:PRINT"Initialising GPS navigation mode...."
420  WRITE#1,"$PMVXG,001,2,0,1.00,1.00,0010,0015,05,U,0"+CHR$(13)+CHR$(10)
430 FOR I=1 TO 10000 : NEXT I
440 LOCATE 23,6:PRINT"
450 REM ** open file on program bootup *****
453 gosub 1020
456 IF HHMMSS$="" THEN GOTO 453
460 GOSUB 10000
470 GPS$="":BYTE$="":HEADING$="":COMPBYTE$=""
480 WHILE NOT EOF(1)
490   REM ** First read GPS data **
500   GOSUB 1020
510 WEND
520 RESET:END
530 :
```

```
1000 REM *** Subroutines follow from here ***
1010 :
1020 REM ** Routine to read GPS data record **
1030 BYTE$=INPUT$(1,#1)
1040 IF BYTE$="$" THEN NEWMESSAGE=1 ELSE NEWMESSAGE=0
1050 WHILE NEWMESSAGE
1060  GPS$=GPS$+BYTE$
1070  BYTE$=INPUT$(1,#1)
1080  IF BYTE$=CHR$(10) THEN NEWMESSAGE=0 ELSE NEWMESSAGE=1
1090 WEND
1100 GOSUB 1460:LOCATE 23,6:GPS$=""
1110 IF (LOC(1)>128) THEN GOSUB 1410
1120 RETURN
1130 :
1140 REM ** Routine to take a compass reading **
1150 IF LOC(2)<1 THEN RECFIELD$(3)="999.9":RETURN
1160  COMPBYTE$=INPUT$(1,#2)
1170  IF COMPBYTE$="$" THEN NEWHEADING=1 ELSE NEWHEADING=0:GOTO 1160
1180  WHILE NEWHEADING
1190   HEADING$=HEADING$+COMPBYTE$
1200 IF LOC(2)<1 THEN RECFIELD$(3)="999.9":RETURN
1210   COMPBYTE$=INPUT$(1,#2)
1220   IF COMPBYTE$=CHR$(10) THEN NEWHEADING=0 ELSE NEWHEADING=1
1230  WEND
1240 HEADING=VAL(MID$(HEADING$,8,5))
1250 IF DISPLAY=1 THEN GOSUB 1320:LOCATE 13,16:PRINT USING"###.##";HEADING
1260 IF DISPLAY=0 THEN COMM2$=HEADING$
1270 IF LOGGING THEN RECFIELD$(3)=MID$(HEADING$,8,5)
1280 HEADING$=""
1290 GOSUB 1410
1300 RETURN
1310 :
1320 REM ** Routine to draw compass needle **
1330 DRAW PICTURE$:PAINT(324,132),0,7:PICTURE$=""
1340 TURN$="TA-"+STR$(CINT(HEADING))
1350 PICTURE$=TURN$+" BU40 M+10,40 M-10,40 M-10,-40 M+10,-40 BD15 L2 U5 R4 D4 L3
U3 R2 D2 L1"
1360 PSET(323,132)
1370 DRAW PICTURE$:PAINT(324,132),14,7
1380 CIRCLE (323,132),5,0 : CIRCLE (323,132),3,0
1390 RETURN
1400 :
1410 REM ** Routine to flush comms buffers **
1420 IF LOC(1)>0 THEN FLUSHCOM1$=INPUT$(LOC(1),#1)
1430 IF LOC(2)>0 THEN FLUSHCOM2$=INPUT$(LOC(2),#2)
1440 RETURN
1450 :
1460 REM ** Routine to sort out what message received **
1470 IF (MID$(GPS$,8,3))="000" THEN GOSUB 1560
1480 IF (MID$(GPS$,8,3))="001" THEN GOSUB 1740
1490 IF (MID$(GPS$,8,3))="003" THEN GOSUB 1860
1500 IF (MID$(GPS$,8,3))="011" THEN GOSUB 2030
1510 IF (MID$(GPS$,8,3))="008" THEN FLAG=0:GOSUB 2120
1520 IF (MID$(GPS$,8,3))="018" THEN FLAG=1:GOSUB 2120
1530 IF DISPLAY=0 THEN COMM1$=GPS$
1540 RETURN
1550 :
1560 REM ** Print status information (Code 000) **
1570 STATUS$=MID$(GPS$,12,3)
1580 IF STATUS$="ACQ" THEN MESSAGE$="Satellite re-acquisition"
1590 IF STATUS$="IAC" THEN MESSAGE$="Initial acquisition     "
1600 IF STATUS$="IDL" THEN MESSAGE$="No satellite visible    "
```

```
1610 IF STATUS$="NAV" THEN MESSAGE$="Navigating          "
1620 IF STATUS$="STS" THEN MESSAGE$="Searching the sky   "
1630 IF STATUS$="TRK" THEN MESSAGE$="Tracking satellites "
1640 IF DISPLAY=1 THEN LOCATE 18,17:PRINT MESSAGE$
1650 IF LOGGING THEN RECFIELD$(4)=STATUS$
1660 SATVIS$=MID$(GPS$,16,1)
1670 IF DISPLAY=1 THEN LOCATE 9,74:PRINT SATVIS$
1680 IF LOGGING THEN RECFIELD$(5)=SATVIS$
1690 SATTRACK$=MID$(GPS$,18,1)
1700 IF DISPLAY=1 THEN LOCATE 10,74:PRINT SATTRACK$
1710 IF LOGGING THEN RECFIELD$(6)=SATTRACK$
1720 RETURN
1730 :
1740 REM ** Print position information (Code 001) **
1750 HHMMSS$=MID$(GPS$,12,2)+":"+MID$(GPS$,14,2)+":"+MID$(GPS$,16,2)
1760 IF DISPLAY=1 THEN LOCATE 6,65:PRINT HHMMSS$
1770 IF LOGGING THEN GOSUB 2340:RECFIELD$(0)=JDAY$+MID$(GPS$,12,6)
1780 LAT$=MID$(GPS$,19,2)+" "+MID$(GPS$,21,8)
1790 IF DISPLAY=1 THEN LOCATE 6,12:PRINT LAT$
1800 IF LOGGING THEN RECFIELD$(1)=MID$(GPS$,19,8)+MID$(GPS$,28,1)
1810 LON$=MID$(GPS$,30,3)+" "+MID$(GPS$,33,8)
1820 IF DISPLAY=1 THEN LOCATE 9,12:PRINT LON$
1830 IF LOGGING THEN RECFIELD$(2)=MID$(GPS$,30,9)+MID$(GPS$,40,1)
1840 RETURN
1850 :
1860 REM ** Print DOPs information (Code 003) **
1870 EDOP$=MID$(GPS$,12,5)
1880 IF DISPLAY=1 THEN LOCATE 16,66:PRINT EDOP$
1890 IF LOGGING THEN RECFIELD$(7)=EDOP$
1900 NDOP$=MID$(GPS$,18,5)
1910 IF DISPLAY=1 THEN LOCATE 15,66:PRINT NDOP$
1920 IF LOGGING THEN RECFIELD$(8)=NDOP$
1930 VDOP$=MID$(GPS$,24,5)
1940 IF DISPLAY=1 THEN LOCATE 17,66:PRINT VDOP$
1950 IF LOGGING THEN RECFIELD$(9)=VDOP$
1960 HDOP$=MID$(GPS$,30,5)
1970 IF DISPLAY=1 THEN LOCATE 18,66:PRINT HDOP$
1980 IF LOGGING THEN RECFIELD$(10)=HDOP$
1990 REM ** Now take a compass reading **
2000 GOSUB 1140
2010 RETURN
2020 :
2030 REM ** Print speed & heading information (Code 011) **
2040 COG$=MID$(GPS$,12,5)
2050 IF DISPLAY=1 THEN LOCATE 12,65:PRINT COG$
2060 IF LOGGING THEN RECFIELD$(11)=COG$
2070 SOG$=MID$(GPS$,18,5)
2080 IF DISPLAY=1 THEN LOCATE 13,65:PRINT SOG$
2090 IF LOGGING THEN RECFIELD$(12)=SOG$
2100 RETURN
2110 :
2120 REM ** Satellite 1-8 signal strength and location (Code 008 & 018) **
2130 IF LOGGING=0 THEN GOTO 2320
2140 NOSATS=VAL(SATTRACK$)
2150 IF NOSATS=0 THEN GOTO 2290
2160 IF FLAG=0 THEN GOTO 2170 ELSE IF FLAG=1 THEN GOTO 2230
2170 IF NOSATS<=4 THEN LIMIT=NOSATS ELSE IF NOSATS>=5 THEN LIMIT=4
2180 STARTPOS=12
2190 FOR I=1 TO LIMIT
2200   RECFIELD$(12+I)=MID$(GPS$,STARTPOS,14)
2210   STARTPOS=STARTPOS+15
2220 NEXT I
2230 STARTPOS=12
```

```
2240 FOR I=5 TO NOSATS
2250   RECFIELD$(12+I)=MID$(GPS$,STARTPOS,14)
2260   STARTPOS=STARTPOS+15
2270 NEXT I
2280 IF NOSATS=8 THEN GOTO 2320
2290 FOR J=(NOSATS+1) TO 8
2300     RECFIELD$(12+J)="00,000,00,00.0"
2310 NEXT J
2320 RETURN
2330 :
2340 REM ******* calculate julian day no. *******
2350 DAY=VAL(MID$(DATE$,4,2)):MONTH=VAL(LEFT$(DATE$,2))
2360 JDAY$=STR$(VAL(MID$("000031059090120151181212243273304334",((MONTH-
1)*3)+1,3))+DAY)
2370 JDAY$=RIGHT$(JDAY$,(LEN(JDAY$)-1))
2380 JDAY$=RIGHT$("00",3-(LEN(JDAY$)))+JDAY$
2390 RETURN
2400 :
```

```
5000 REM ** Routine to set up screen **
5010 KEY OFF:KEY(18) OFF:KEY(17) ON:KEY(1) OFF:KEY(2) OFF:KEY(5) OFF:KEY(6) OFF
5020 SCREEN 9:COLOR 7,0:CLS:DISPLAY=1
5030  LINE(5,5)-(635,345),7,B
5040  LINE(7,7)-(633,343),7,B
5050 COLOR 15,0:LOCATE 3,11:PRINT " GPS Data Retreival and Display System ":COLOR 7,0
5060 REM ** Draw compass and anotate **
5070  LINE(323,187)-(323,177):LINE(323,77)-(323,87)
5080  LINE(246,132)-(256,132):LINE(400,132)-(390,132)
5090  CIRCLE  (323,132),70
5100  CIRCLE  (323,132),68
5110 FOR THETA = 0 TO 360 STEP 15
5120    ANGLE=THETA*(3.141592654#/180)
5130    X=(70*SIN(ANGLE))+323
5140    Y=470*(70*COS(ANGLE))/640+132
5150    P=(60*SIN(ANGLE))+323
5160    Q=470*(60*COS(ANGLE))/640+132
5170   PSET (P,Q):LINE-(X,Y)
5180 NEXT THETA
5190 COLOR 14,0
5200 LOCATE 5,41:PRINT "0"
5210 LOCATE 10,52:PRINT "90"
5220 LOCATE 15,40:PRINT "180"
5230 LOCATE 10,28:PRINT "270"
5240 REM ** Draw boxes to hold other info **
5250 REM ** Left box **
5260  LINE(32,51)-(435,218),7,B
5270  LINE(30,49)-(437,220),7,B
5280 LOCATE 6,6:PRINT "LAT : "
5290 LOCATE 9,6:PRINT "LON : "
5300 LOCATE 12,6:PRINT "Compass"
5310 LOCATE 13,6:PRINT "Heading :"
5320 REM ** Right box **
5330  LINE(470,51)-(607,315),7,B
5340  LINE(468,49)-(609,317),7,B
5350  LINE(470,98)-(607,98),7
5360  LINE(470,267)-(607,267),7
5370 LOCATE 6,61:PRINT "UTC"
5380 LOCATE 9,61:PRINT "Sats Visible "
5390 LOCATE 10,61:PRINT "Sats Tracked "
5400 LOCATE 12,61:PRINT "COG"
5410 LOCATE 13,61:PRINT "SOG"
5420 LOCATE 15,61:PRINT "NDOP"
5430 LOCATE 16,61:PRINT "EDOP"
5440 LOCATE 17,61:PRINT "VDOP"
5450 LOCATE 18,61:PRINT "HDOP"
5460 LOCATE 18,6:PRINT "GPS Mode : "
5470 REM ** Inform user of keys for logging data **
5480 COLOR 4,0
5490 LOCATE 20,8:PRINT "<Alt><F1>"
5500 LOCATE 21,8:PRINT "<Alt><F2>"
5510 LOCATE 20,40:PRINT "<Alt><Esc>"
5520  COLOR 7,0
5530 LOCATE 20,18:PRINT "- start logging"
5540 LOCATE 21,18:PRINT "- stop logging"
5550 LOCATE 21,38:PRINT "Command Screen"
5560 IF LOGGING THEN LOCATE 23,6:PRINT"Logging data....":LOCATE 23,22:PRINT FILE$
5570  PSET  (323,132)
5580  RETURN
5590  :
5600 REM ** Routine to display additional commands **
5610 REM ** aswell as contents of both COM ports **
```

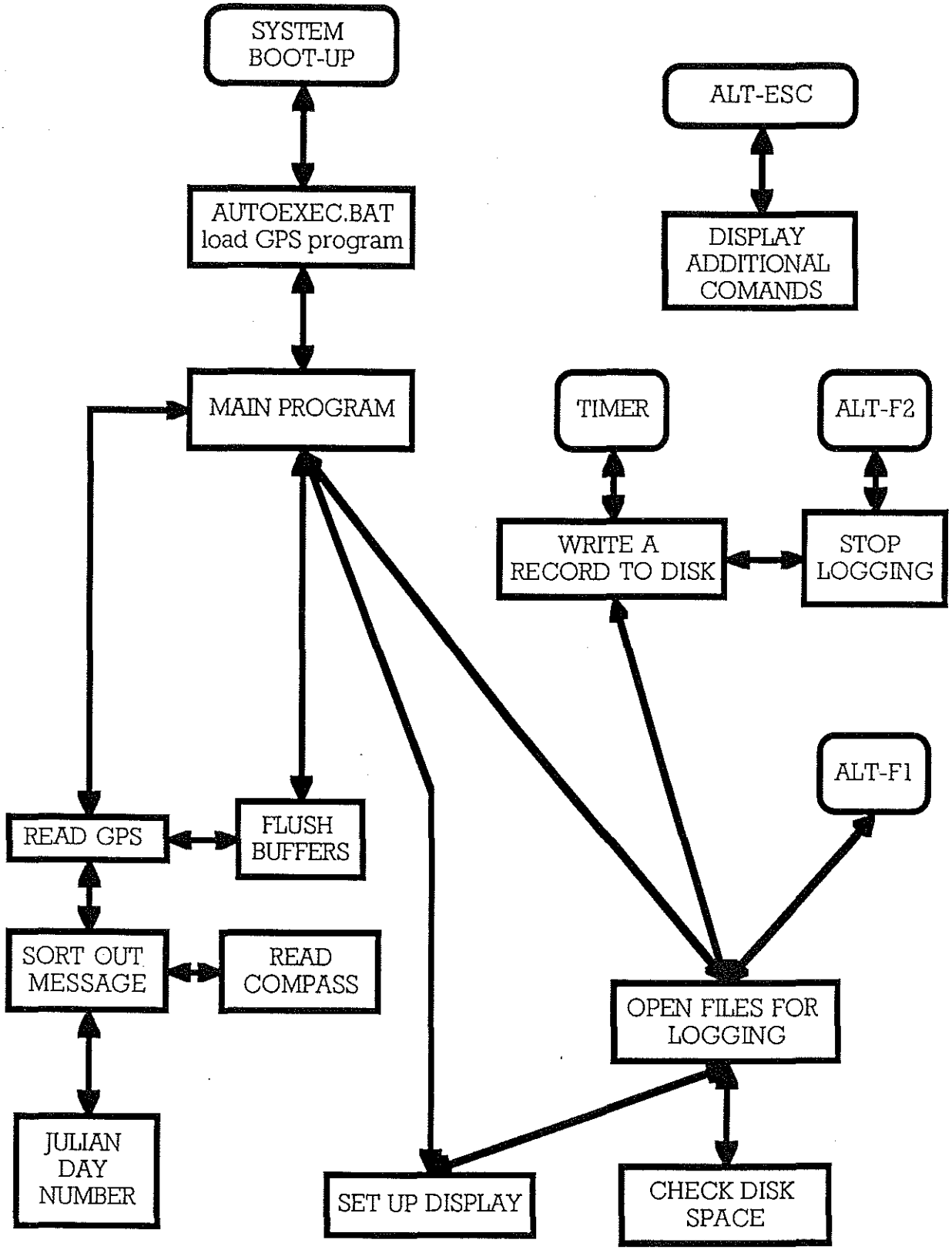
```
5620 SCREEN 9:COLOR 7,0:CLS:DISPLAY=0
5630 KEY(18) ON : KEY(17) OFF
5640 KEY(1) ON
5650 KEY(2) ON
5660 KEY(5) ON
5670 KEY(6) ON
5680 LINE(5,5)-(635,345),7,B
5690 LINE(7,7)-(633,343),7,B
5700 LINE(32,147)-(612,322),7,B
5710 LINE(30,145)-(614,324),7,B
5720 COLOR 14,0:LOCATE 3,5:PRINT"Control keys :"
5730 COLOR 4,0
5740 LOCATE 5,5:PRINT"<Alt><Enter>"
5750 LOCATE 7,6:PRINT"<F1>"
5760 LOCATE 8,6:PRINT"<F2>"
5770 LOCATE 7,42:PRINT"<F5>"
5780 LOCATE 8,42:PRINT"<F6>"
5790 COLOR 7,0
5800 LOCATE 5,18:PRINT"- Return to main display"
5810 LOCATE 7,11:PRINT"- Display contents of COM1"
5820 LOCATE 8,11:PRINT"- Display contents of COM2"
5830 LOCATE 7,47:PRINT"- Restart GPS system"
5840 LOCATE 8,47:PRINT"- Initialise GPS parameters"
5850 IF DISPLAY=1 THEN GOTO 5860  ELSE GOTO 5850
5860 RETURN
5870 :
5880 REM ** Routine to display contents of COM1 **
5890 LOCATE 11,20:PRINT" Data coming in on COM1 : "
5900 LINE(33,153)-(611,321),0,BF
5910 KEY(1) OFF:ROW=13:COL=7
5920 WHILE DISPLAY=0
5930   GOSUB 1020
5940   IF (ROW>=23) THEN ROW=13:LINE(33,153)-(611,321),0,BF
5950   LOCATE ROW,COL
5960   IF LEN(COMM1$)>68 THEN PRINT LEFT$(COMM1$,68):ROW=ROW+1 ELSE PRINT
COMM1$:ROW=ROW+1:GOTO 5990
5970   IF (ROW>=23) THEN ROW=13:LINE(33,153)-(611,321),0,BF
5980   LOCATE ROW,COL:PRINT RIGHT$(COMM1$,(LEN(COMM1$)-68)):ROW=ROW+1
5990   KEY(2) ON
6000 WEND
6010 RETURN
6020 REM ** Routine to display contents of COM2 **
6030 LOCATE 11,20:PRINT" Data coming in on COM2 : "
6040 LINE(33,153)-(611,321),0,BF
6050 KEY(2) OFF:ROW=13:COL=7
6060 WHILE DISPLAY=0
6070   GOSUB 1020
6080   LOCATE ROW,COL
6090   PRINT COMM2$:ROW=ROW+1
6100   IF (ROW>=23) THEN ROW=13:LINE(33,153)-(611,321),0,BF
6110   KEY(1) ON
6120 WEND
6130 RETURN
6140 :
6150 REM ** Routine to restart GPS **
6160 WRITE#1,"$PMVXG,018,W"+CHR$(13)+CHR$(10)
6170 LOCATE 14,10:PRINT "Initialising GPS....."
6180 FOR I=1 TO 20000 : NEXT I
6190 LOCATE 14,10:PRINT "                    "
6200 RETURN
6210 REM ** Routine to initialise GPS parameters **
6220 COLOR 14,0:LOCATE 11,20:PRINT" Initialise GPS Parameters "
6230 LINE(33,153)-(611,321),0,BF
```

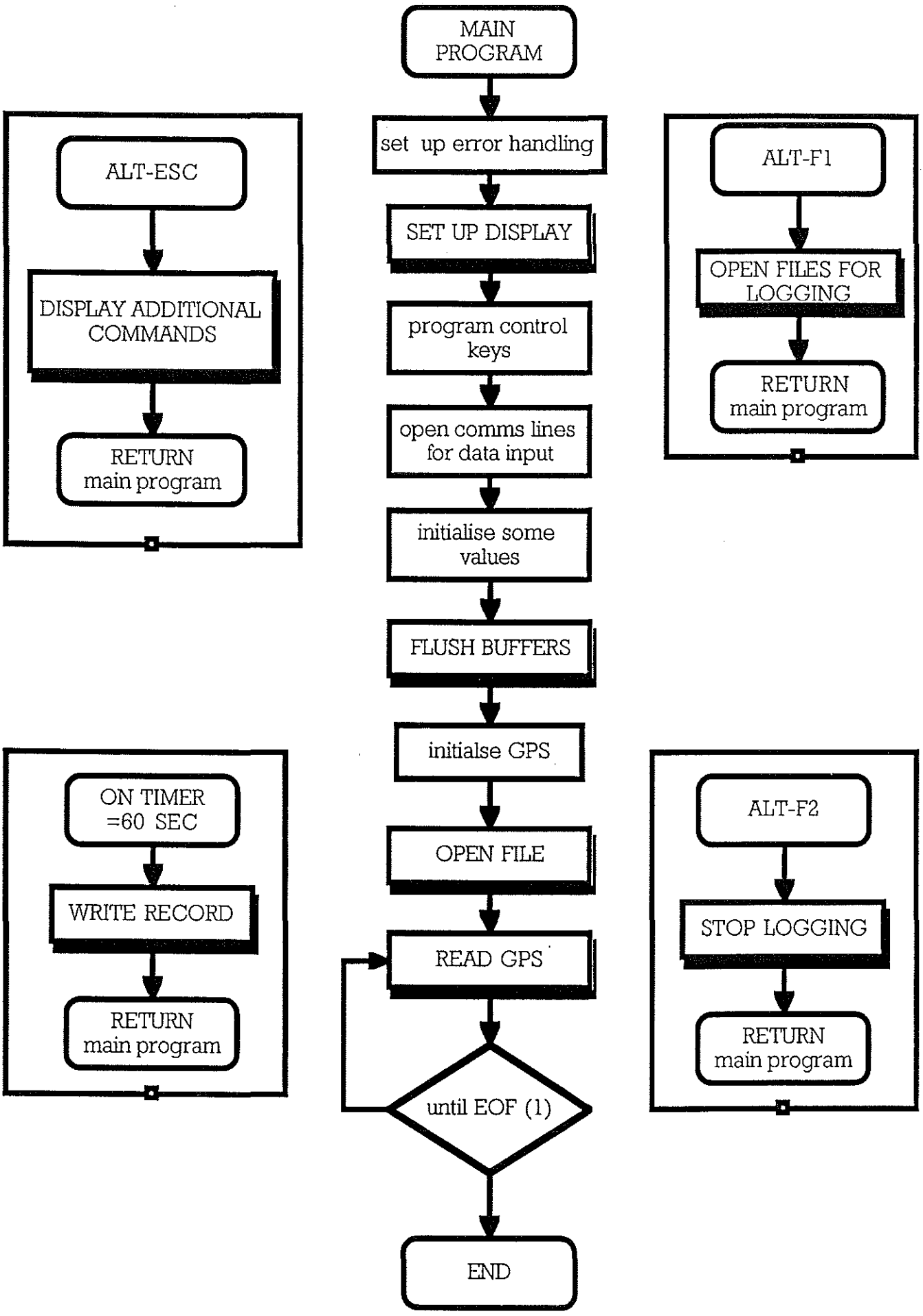
```
6240 KEY(6) OFF
6250 LOCATE 13,7:PRINT"Time (UTC) [";TIME$;"] "
6260 COLOR 7,0:LOCATE 13,30:INPUT GMT$
6270 IF VAL(GMT$)=VAL(CHR$(13)) THEN GMT$=TIME$:LOCATE 13,33:PRINT GMT$
6280 COLOR 14,0:LOCATE 15,7:PRINT"Date [";DATE$;"] "
6290 COLOR 7,0:LOCATE 15,26:INPUT DDMMYY$
6300 IF VAL(DDMMYY$)=VAL(CHR$(13)) THEN DDMMYY$=DATE$:LOCATE 15,29:PRINT
DDMMYY$
6310 COLOR 14,0:LOCATE 17,7:PRINT"Approx latitude [";LAT$;"] "
6320 COLOR 7,0:TEMP1$=LAT$:LOCATE 17,37:INPUT LAT$
6330 IF VAL(LAT$)=VAL(CHR$(13)) THEN LAT$=TEMP1$:LOCATE 17,40:PRINT LAT$:GOTO
6350 ELSE COLOR 14,0:LOCATE 17,56:PRINT"N or S [N] ":COLOR 7,0:LOCATE 17,67:INPUT
NORS$
6340 IF VAL(NORS$)=VAL(CHR$(13)) THEN NORS$="N":LOCATE 17,69:PRINT NORS$
6350 COLOR 14,0:LOCATE 19,7:PRINT"Approx longitude [";LON$;"] "
6360 COLOR 7,0:TEMP2$=LON$:LOCATE 19,39:INPUT LON$
6370 IF VAL(LON$)=VAL(CHR$(13)) THEN LON$=TEMP2$:LOCATE 19,42:PRINT LON$:GOTO
6390 ELSE COLOR 14,0:LOCATE 19,56:PRINT"W or E [W] ":COLOR 7,0:LOCATE 19,67:INPUT
WORE$
6380 IF VAL(WORE$)=VAL(CHR$(13)) THEN WORE$="W":LOCATE 19,69:PRINT WORE$
6390 COLOR 14,0:LOCATE 21,7:PRINT"Approx antenna height [";ALT$;"] "
6400 COLOR 7,0:TEMP3$=ALT$:LOCATE 21,40:INPUT ALT$
6410 IF VAL(ALT$)=VAL(CHR$(13)) THEN ALT$=TEMP3$:LOCATE 21,43:PRINT ALT$
6420 COLOR 7,0:LOCATE 13,52:PRINT"Are these values OK ?? ":COLOR 4,0:LOCATE
14,60:PRINT"(y/n)":COLOR 7,0
6430 LOCATE 14,66:INPUT K$
6440 IF (K$="Y" OR K$="y") THEN GOSUB 6470 :KEY(6) ON:RETURN ELSE IF (K$="N" OR
K$="n") THEN GOTO 6210  ELSE BEEP:GOTO 6420
6450 RETURN
6460 :
6470 REM ** Routine to assign values to GPS **
6480 FDATE$=MID$(DDMMYY$,4,2)+","+LEFT$(DDMMYY$,2)+","+RIGHT$(DDMMYY$,4)
6490 FGMT$=(LEFT$(GMT$,2))+(MID$(GMT$,4,2))+(RIGHT$(GMT$,2))
6500
PARAMS$="$PMVXG,000,"+FDATE$+","+FGMT$+","+LAT$+","+NORS$+","+LON$+","+WORE$+"
,"+ALT$+","+CHR$(13)+CHR$(10)
6510 WRITE#1,PARAMS$
6520 LOCATE 11,20:PRINT"                    "
6530  LINE  (32,147)-(612,322),7,B
6540  LINE  (30,145)-(614,324),7,B
6550  LINE  (33,153)-(611,321),0,BF
6560 LOCATE 14,10:PRINT"Assigning new parameters now...."
6570 FOR I=1 TO 10000 : NEXT I
6580 LOCATE 14,10:PRINT"                    "
6590  RETURN
6600 :
```

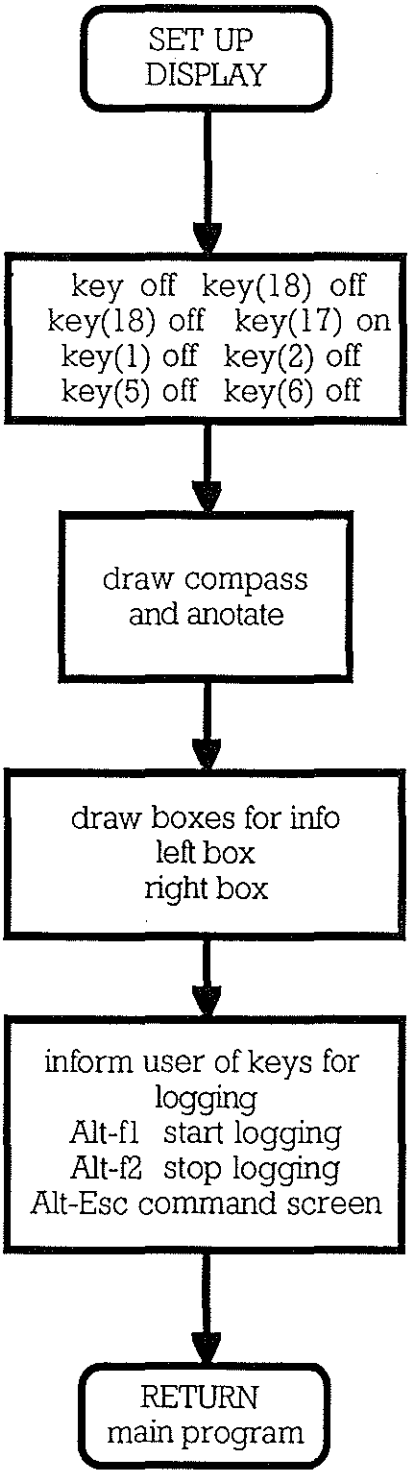
```
10000 REM ** Routine to open file for logging data **
10010 REM * First DISABLE <Alt><F1> and ENABLE <Alt><F2> *
10020  FLAGH=0:FLAGF=0
10030 KEY(15) OFF : KEY(16) ON
10035 TIME$=HHMMSS$
10040 REM * If a new disk is inserted then obtain its details *
10050 IF NEWDISK THEN DRIVE$=HDRIVE$:GOSUB 10760
10060 IF NEWDISK THEN DRIVE$=FDRIVE$:GOSUB 10760
10070 REM * Establish free space on disk *
10080 LOCATE 23,6:PRINT"Please wait....establishing file size..."
10090 IF FLAGH=0 THEN CLOSE #3:OPEN HDRIVE$+"\dummy.dat" FOR OUTPUT AS #3
10100 IF FLAGF=0 THEN CLOSE #4:OPEN FDRIVE$+"\dummy.dat" FOR OUTPUT AS #4
10110 REM * write 6 hours worth of records (360) *
10120
DUMMYREC$="xxxxxxxxx1xxxxxxxxx1xxxxxxxxx1xxxxxxxxx1xxxxxxxxx1xxxxxxxxx1xxxxxxxx
x1xxxxxxxxx1xxxxxxxxx1xxxxxxxxx1xxxxxxxxx1xxxxxxxxx1xxxxxxxxx1xxxxxxxxx1
xxxxxx"+CHR$(13)+CHR$(10)
10130 FOR I=1 TO 360
10140    DRIVE$=HDRIVE$:IF FLAGH=0 THEN WRITE#3,DUMMYREC$
10150    DRIVE$=FDRIVE$:IF FLAGF=0 THEN WRITE#4,DUMMYREC$
10160    NUMBYTES=NUMBYTES+158
10170 NEXT I
10180 IF FLAGH=0 THEN CLOSE #3:KILL HDRIVE$+"\dummy.dat"
10190 IF FLAGF=0 THEN CLOSE #4:KILL FDRIVE$+"\dummy.dat"
10200 LOCATE 23,6:PRINT "Logging data....                    ":LOGGING=1
10210 LOGGING=1
10220 ON TIMER(60) GOSUB 10360
10230 TIMER OFF
10240 REM * Establish the filename, and open *
10250  GOSUB 2340
10260 HOUR$=(LEFT$(TIME$,2)):MINUTE$=(MID$(TIME$,4,2))
10270 FILE$=HDRIVE$+"\N"+JDAY$+HOUR$+MINUTE$+".DAT"
10280 FILE2$=FDRIVE$+"\G"+JDAY$+HOUR$+MINUTE$+".DAT"
10290 IF FLAGH=1 THEN LOCATE 23,22:PRINT FILE2$ ELSE LOCATE 23,22:PRINT FILE$
10300 IF FLAGH=1 AND FLAGF=1 THEN LOCATE 23,22:PRINT "NO FILES OPENED"
10310 IF FLAGH=0 THEN OPEN FILE$ FOR OUTPUT AS #3
10320 IF FLAGF=0 THEN OPEN FILE2$ FOR OUTPUT AS #4
10330 TIMER ON : NUMRECS=0
10340 RETURN
10350 :
10360 REM ** Routine to write a record to disk **
10370 REM * First remove commas from fields 13 to 20 *
10380 FOR FLDNUM=13 TO 20
10390   TEMP$=""
10400   FOR BYTENUM=1 TO 14
10410     FIELDBYTE$=MID$(RECFIELD$(FLDNUM),BYTENUM,1)
10420     IF FIELDBYTE$="," THEN GOTO 10440
10430     TEMP$=TEMP$+FIELDBYTE$
10440   NEXT BYTENUM
10450   RECFIELD$(FLDNUM)=TEMP$
10460 NEXT FLDNUM
10470 REM * Now write a single record *
10480 FOR FLDNUM=0 TO 20
10490 IF FLDNUM<5 AND RECPLT=0 AND FLAGF=0 THEN PRINT #4,RECFIELD$(FLDNUM);
10500 IF FLAGH=0 THEN  PRINT #3,RECFIELD$(FLDNUM);
10510 NEXT FLDNUM
10520 IF FLAGH=0 THEN PRINT #3,CHR$(13)+CHR$(10)
10530 IF RECPLT=0 AND FLAGF=0 THEN PRINT #4,CHR$(13)+CHR$(10):RECPLT=1 ELSE
RECPLT=0
10540 REM * Update record count & check if new file needed *
10550 NUMRECS=NUMRECS+1
10560 IF NUMRECS=360 THEN GOSUB 10650 : GOSUB 10000
```

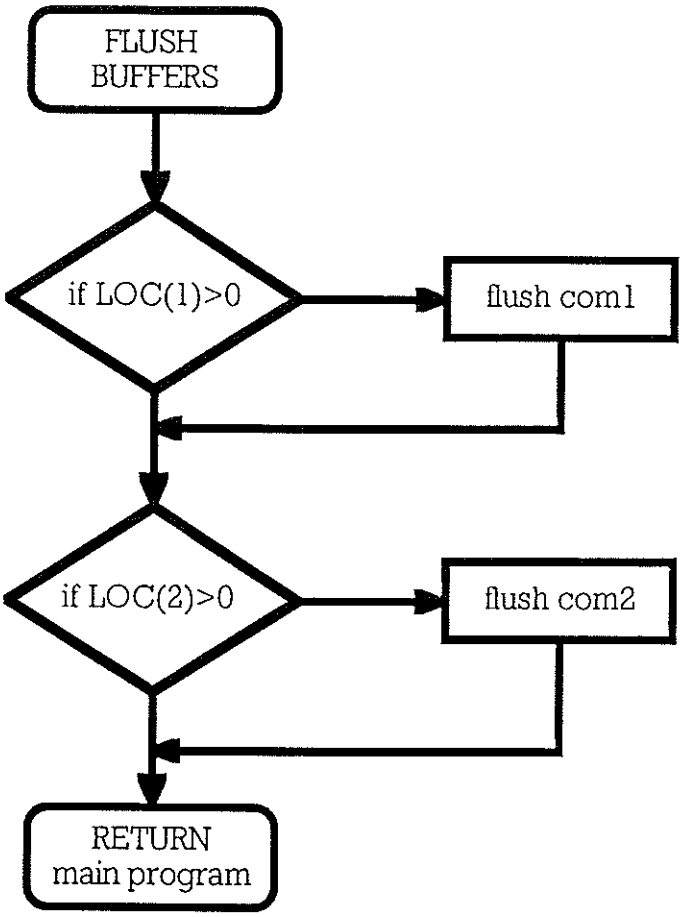
```
10570 REM * Calculate percentage of file written *
10580 PCENTAGE=(NUMRECS/360)*100
10590 REM * Calculate time to end of disk *
10600 IF TIMEMIN>0 THEN TIMEMIN=TIMEMIN-1 ELSE IF TIMEMIN=0 THEN TIMEMIN=59:IF
TIMEHRS>0 THEN TIMEHRS=TIMEHRS-1 ELSE IF TIMEHRS=0 THEN TIMEHRS=23:IF
TIMEDAY>0 THEN TIMEDAY=TIMEDAY-1
10610 LOCATE 21,60:PRINT USING "###.#_%";PCENTAGE;:COLOR 14,0:PRINT" WTF ":COLOR
7,0
10620 LOCATE 22,60:PRINT USING "##";TIMEDAY;:PRINT"d ";:PRINT USING
"##";TIMEHRS;:PRINT":";:PRINT USING"##";TIMEMIN;:COLOR 14,0:PRINT" - EOD":COLOR 7,0
10630 RETURN
10640 :
10650 REM ** Routine to stop logging data to disk **
10660 REM * DISABLE <Alt><F2> and ENABLE <Alt><F1> *
10670 KEY(16) OFF : KEY(15) ON
10680 LOCATE 23,6:PRINT "Closing data file.....                "
10690 LOGGING=0 : TIMER OFF : CLOSE #3 : CLOSE #4
10700 LOCATE 23,6:PRINT "                                      "
10710 LOCATE 21,60:PRINT"          "
10720 LOCATE 22,60:PRINT"            "
10730 NEWDISK=1
10740 RETURN
10750 :
10760 REM ** Routine to check amount of space available on disk **
10770 CLS : I=1 : Y$="" : TOTAL$=""
10780 IF DRIVE$=FDRIVE$ THEN NBYTES=20 ELSE NBYTES =160
10790 SCREEN 0 : VIEW PRINT
10800 LOCATE 10,10:PRINT"Please wait.....examining disk...."
10810 COLOR 0,0
10820 REM ** Disk might be empty, so create a file so as to get **
10830 REM ** the number of bytes free when using the FILES command **
10840 OPEN DRIVE$+"\NEWFILE.DAT" FOR OUTPUT AS #3
10850 IF FLAGF=1 AND DRIVE$=FDRIVE$ THEN CLOSE#3:GOSUB 5000:RETURN
10860 IF FLAGH=1 AND DRIVE$=HDRIVE$ THEN CLOSE#3:GOSUB 5000:RETURN
10870 VIEW PRINT 21 TO 23
10880 FILES DRIVE$
10890 VIEW PRINT
10900 WHILE Y$<>"B"
10910   Y$=CHR$(SCREEN(21,I))
10920   TOTAL$=TOTAL$+Y$ : I=I+1
10930 WEND
10940 CLOSE #3 : KILL DRIVE$+"\NEWFILE.DAT"
10950 SCREEN 9:COLOR 7,0
10960 AMNT=LEN(TOTAL$)-2
10970 DISKSPACE$=LEFT$(TOTAL$,AMNT)
10980 TIMEFILL=INT(VAL(DISKSPACE$)/NBYTES)
10990 TIMEDAY=INT((TIMEFILL/60)/24)
11000 TIMEHRS=INT((((TIMEFILL/60)/24)-TIMEDAY)*24)
11010 TIMEMIN=INT((((((TIMEFILL/60)/24)-TIMEDAY)*24)-TIMEHRS)*60)
11020 LINE (100,50)-(540,300),7,B
11030 LINE (102,52)-(538,298),7,B
11040 COLOR 14,0:LOCATE 7,20:PRINT "Disk information (Drive "+DRIVE$+")":COLOR 7,0
11050 LOCATE 9,16:PRINT "The disk you have inserted has";DISKSPACE$;" bytes free"
11060 IF DRIVE$=HDRIVE$ THEN LOCATE 11,16:PRINT "Writing a record at the rate of 1 per
minute,"
11070 IF DRIVE$=FDRIVE$ THEN LOCATE 11,16:PRINT "Writing a record at the rate of 1
every 2 minutes,"
11080 LOCATE 12,16:PRINT "implies that the disk will be full in :"
11090 COLOR 14,0:LOCATE 14,19:PRINT TIMEDAY;"days ";TIMEHRS;"hrs
";TIMEMIN;"mins":COLOR 7,0
11100 LOCATE 16,16:PRINT "Is this suitable ? (y/n)"
11110 REM ** wait 10 sec to enter if space is ok else continue ****
11120 FOR I=0 TO 10000:NEXT:IX=IX+1
```

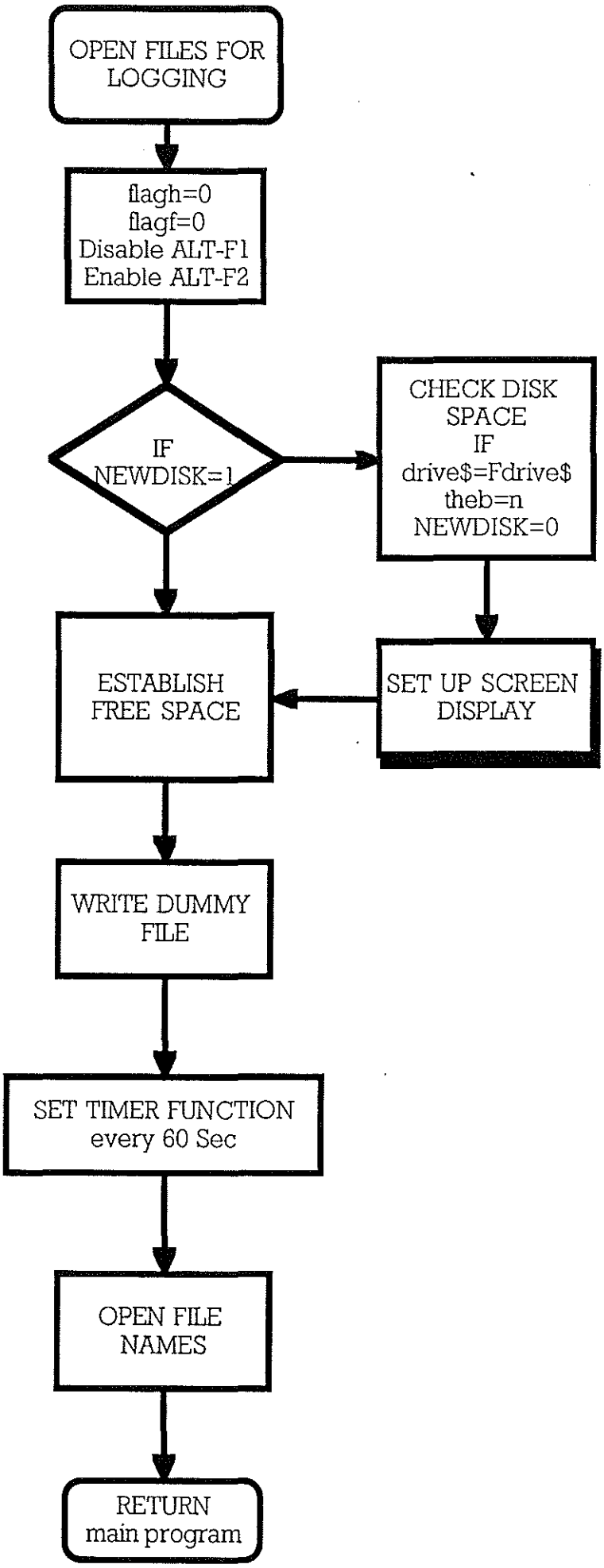
```
11130 IF IX > 10 GOTO 11160
11140 LOCATE 16,41:K$=INKEY$
11150 IF (K$="Y" OR K$="y") THEN GOTO 11160 ELSE IF (K$="N" OR K$="n") THEN LOCATE
18,16:PRINT"Please insert another disk and press <Enter>":WHILE
INKEY$<>CHR$(13):WEND:GOTO 10760 ELSE BEEP:GOTO 11100
11160 IF DRIVE$=FDRIVE$ THEN NEWDISK=0
11170 IX=0
11180 GOSUB 5000
11190 RETURN
11200 :
```

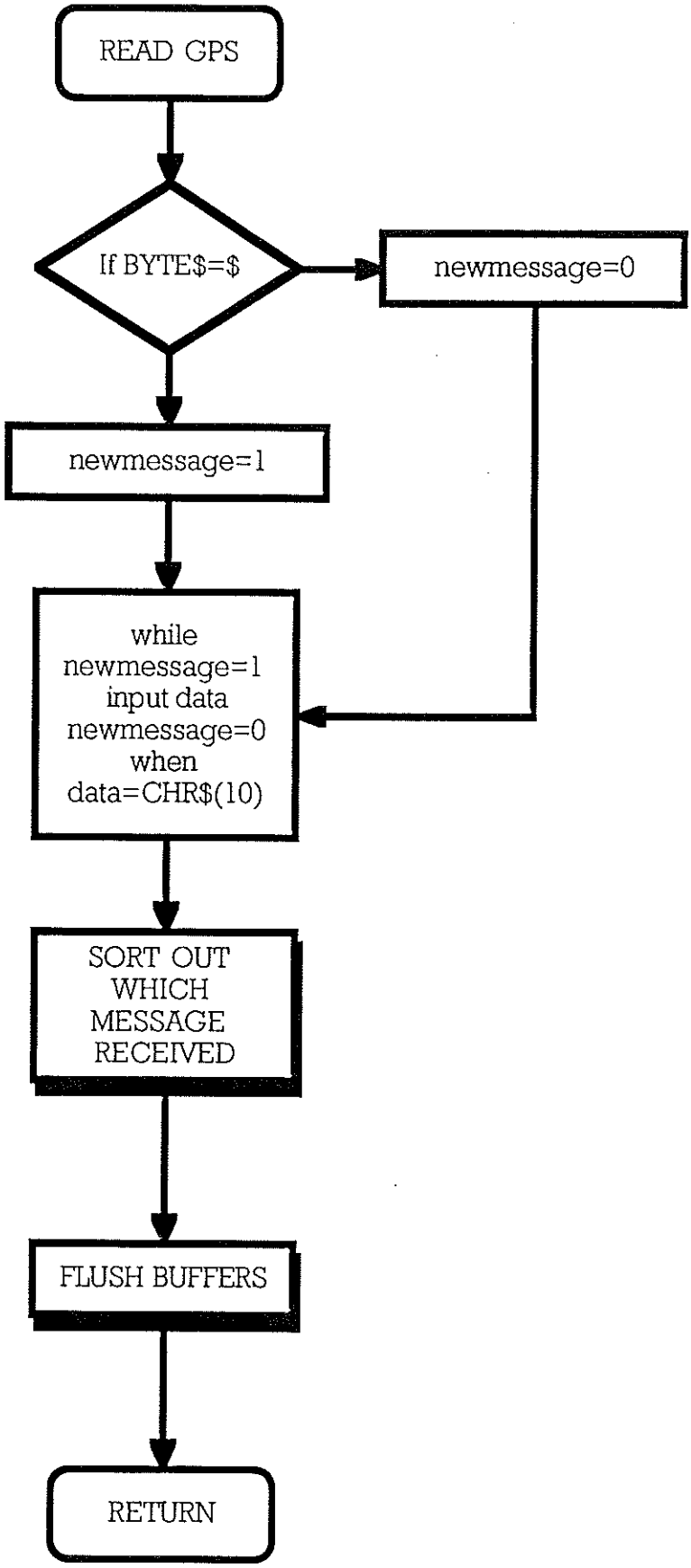

SYSTEM BOOT-UP
AUTOEXEC.BAT load GPS program
MAIN PROGRAM
ALT-ESC
DISPLAY ADDITIONAL COMANDS
TIMER
ALT-F2
WRITE A RECORD TO DISK
STOP LOGGING
ALT-F1
READ GPS
FLUSH BUFFERS
SORT OUT MESSAGE
READ COMPASS
JULIAN DAY NUMBER
OPEN FILES FOR LOGGING
SET UP DISPLAY
CHECK DISK SPACE

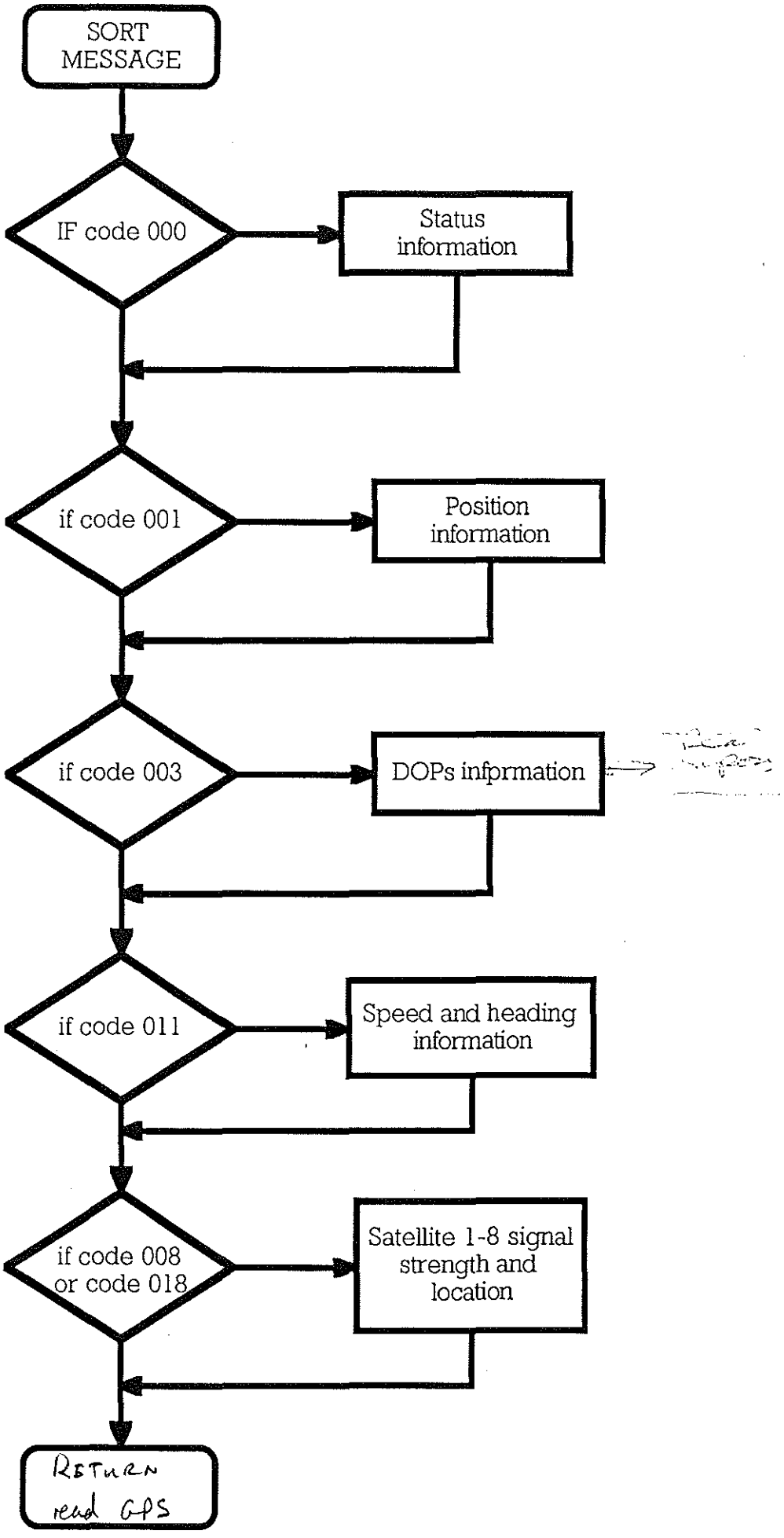

MAIN PROGRAM
set up error handling
SET UP DISPLAY
program control keys
open comms lines for data input
initialise some values
FLUSH BUFFERS
initialse GPS
OPEN FILE
READ GPS
until EOF (1)
END
ALT-ESC
DISPLAY ADDITIONAL COMMANDS
RETURN main program
ALT-F1
OPEN FILES FOR LOGGING
RETURN main program
ON TIMER =60 SEC
WRITE RECORD
RETURN main program
ALT-F2
STOP LOGGING
RETURN main program

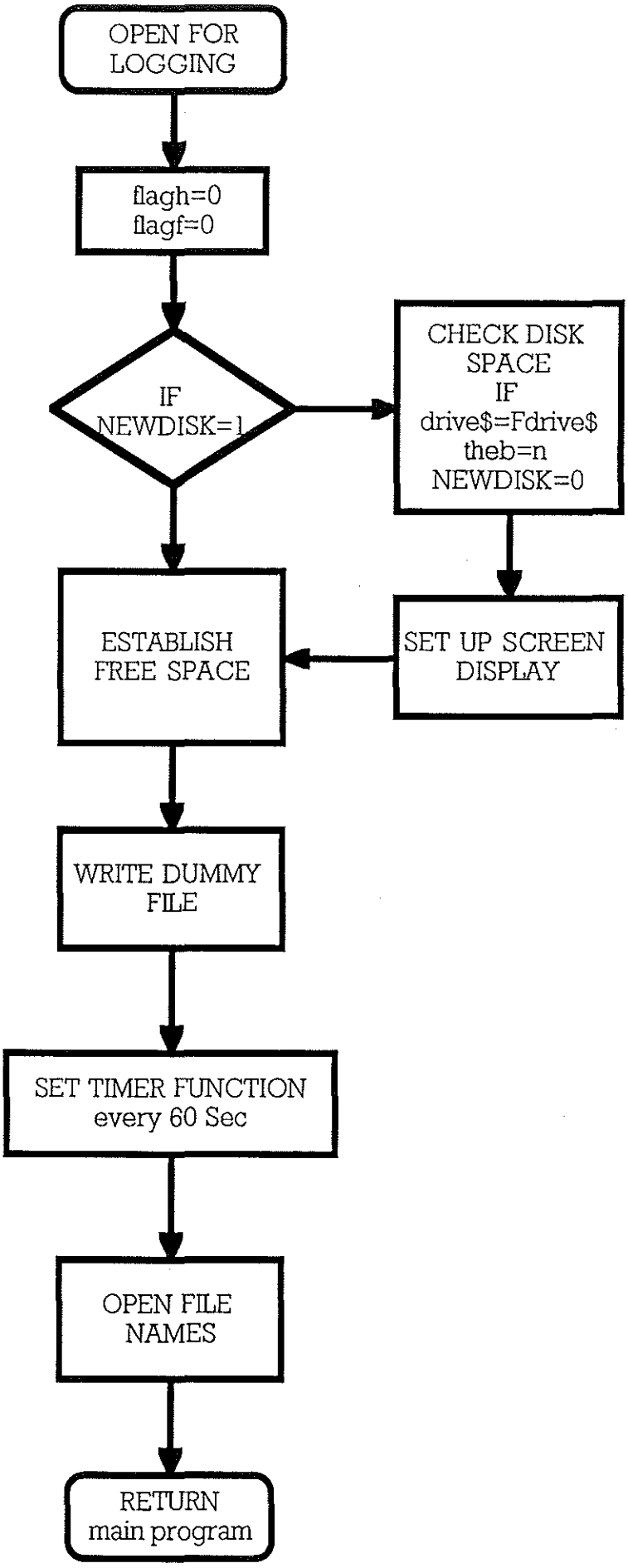

SET UP
DISPLAY
key off key(18) off
key(18) off key(17) on
key(1) off key(2) off
key(5) off key(6) off
draw compass
and anotate
draw boxes for info
left box
right box
inform user of keys for
logging
Alt-f1 start logging
Alt-f2 stop logging
Alt-Esc command screen
RETURN
main program

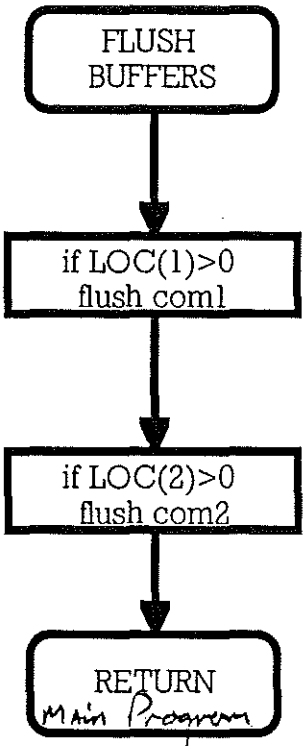

FLUSH
BUFFERS
if LOC(1)>0
flush com1
if LOC(2)>0
flush com2
RETURN
main program

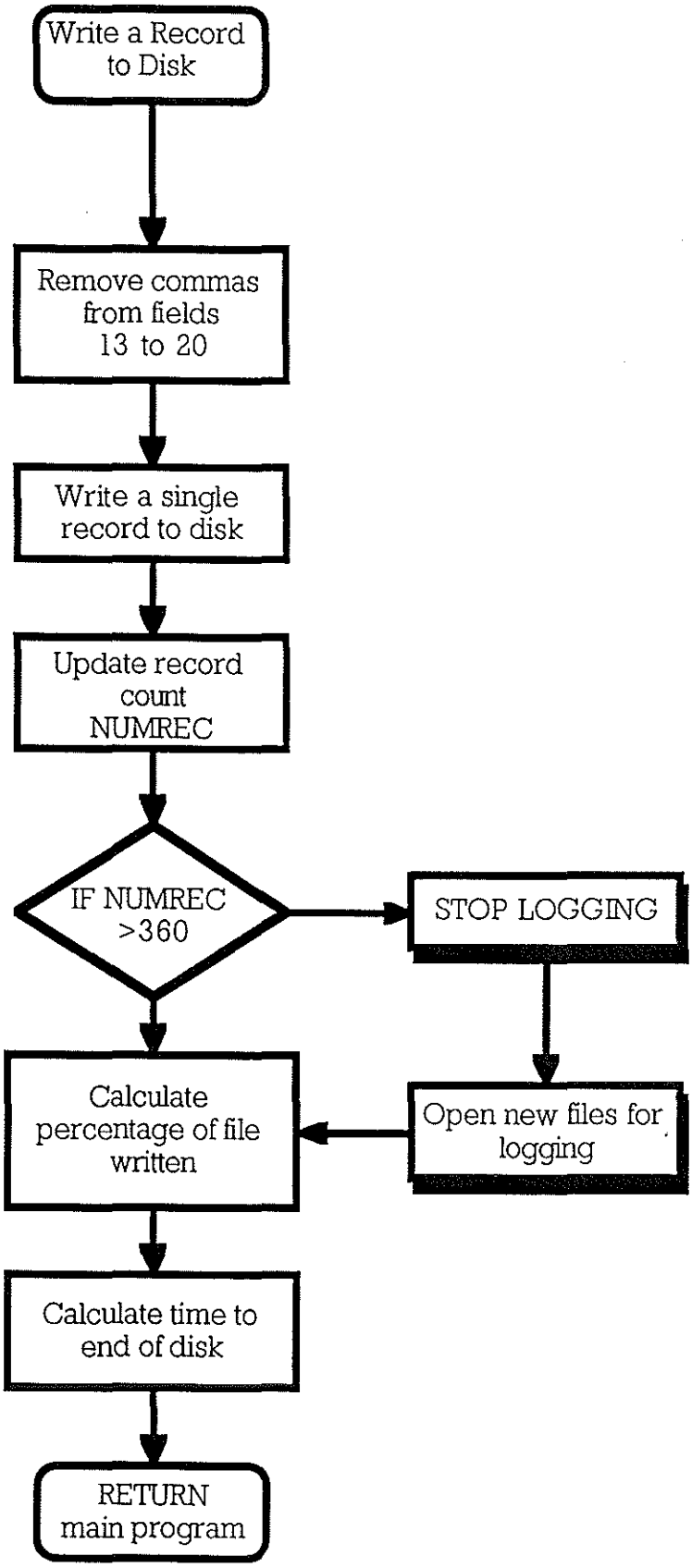

OPEN FILES FOR LOGGING
flagh=0
flagf=0
Disable ALT-F1
Enable ALT-F2
IF NEWDISK=1
CHECK DISK SPACE
IF drive$=Fdrive$
theb=n
NEWDISK=0
ESTABLISH FREE SPACE
SET UP SCREEN DISPLAY
WRITE DUMMY FILE
SET TIMER FUNCTION every 60 Sec
OPEN FILE NAMES
RETURN main program

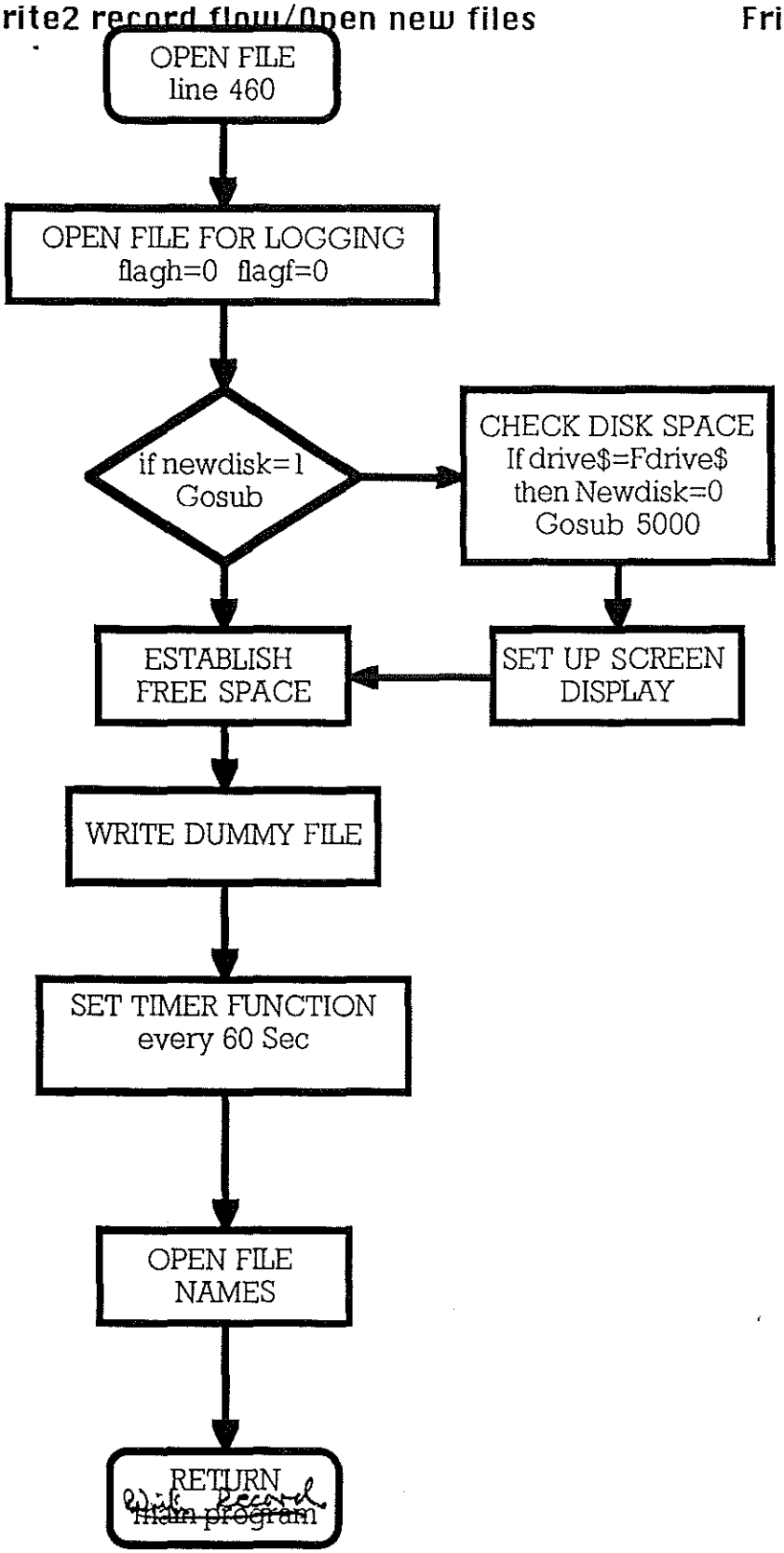

READ GPS
If BYTE$=$
newmessage=0
newmessage=1
while
newmessage=1
input data
newmessage=0
when
data=CHR$(10)
SORT OUT
WHICH
MESSAGE
RECEIVED
FLUSH BUFFERS
RETURN

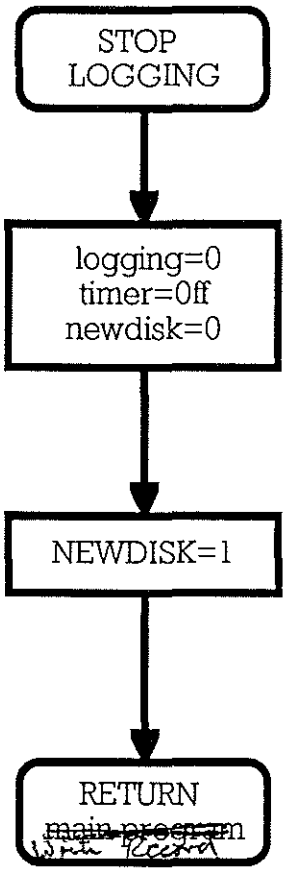

SORT MESSAGE
IF code 000
Status information
if code 001
Position information
if code 003
DOPs infprmation
if code 011
Speed and heading information
if code 008 or code 018
Satellite 1-8 signal strength and location
RETURN read GPS

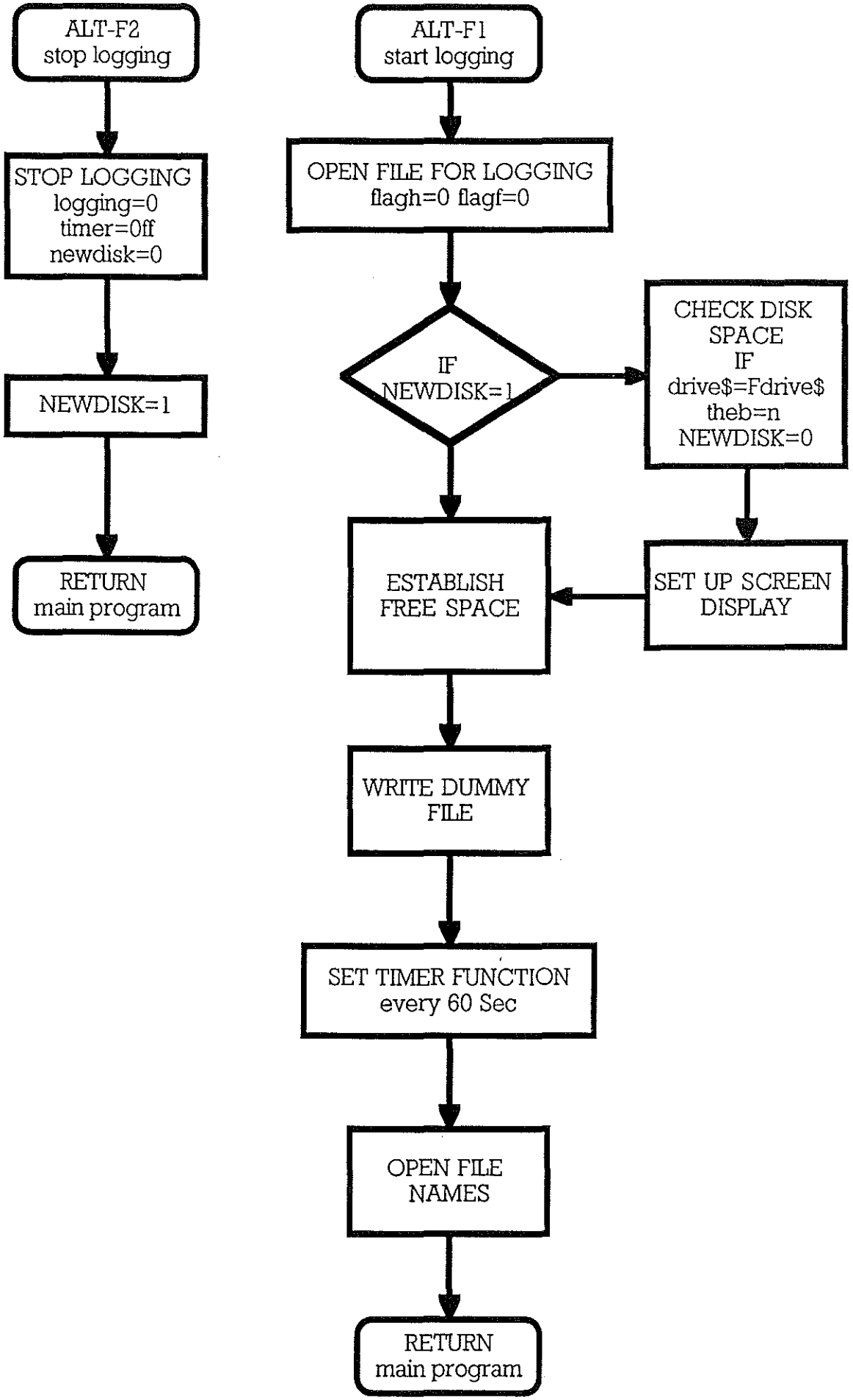

OPEN FOR LOGGING
flagh=0
flagf=0
IF NEWDISK=1
CHECK DISK SPACE
IF drive$=Fdrive$
theb=n
NEWDISK=0
ESTABLISH FREE SPACE
SET UP SCREEN DISPLAY
WRITE DUMMY FILE
SET TIMER FUNCTION every 60 Sec
OPEN FILE NAMES
RETURN main program

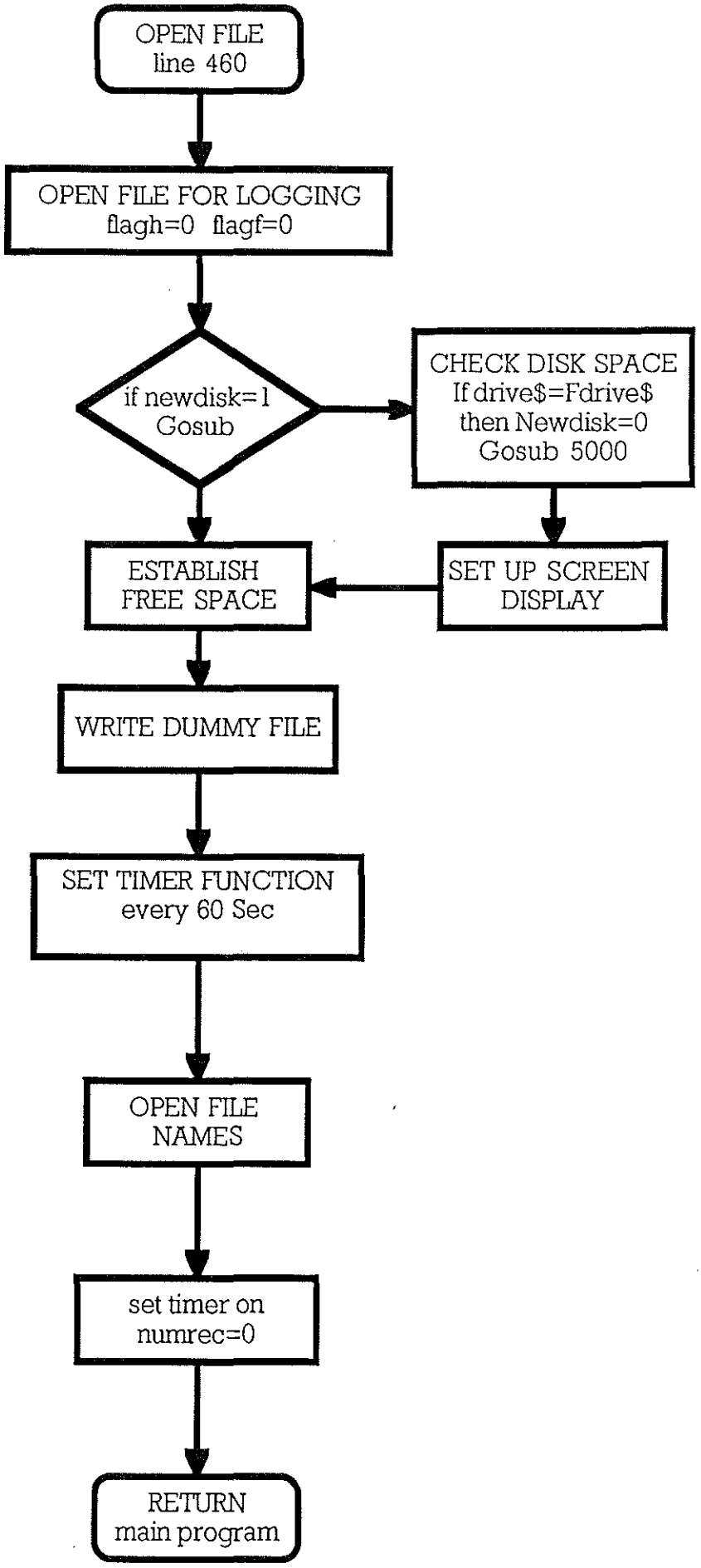

FLUSH
BUFFERS
if LOC(1)>0
flush com1
if LOC(2)>0
flush com2
RETURN
Main Program

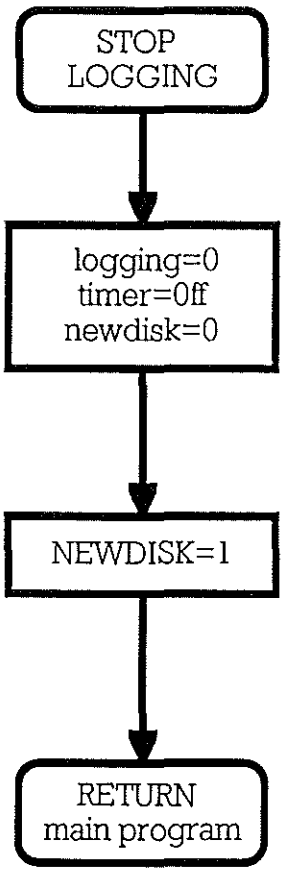

Write a Record to Disk
Remove commas from fields 13 to 20
Write a single record to disk
Update record count NUMREC
IF NUMREC >360
STOP LOGGING
Open new files for logging
Calculate percentage of file written
Calculate time to end of disk
RETURN main program


OPEN FILE
line 460
OPEN FILE FOR LOGGING
flagh=0 flagf=0
if newdisk=1
Gosub
CHECK DISK SPACE
If drive$=Fdrive$
then Newdisk=0
Gosub 5000
ESTABLISH
FREE SPACE
SET UP SCREEN
DISPLAY
WRITE DUMMY FILE
SET TIMER FUNCTION
every 60 Sec
OPEN FILE
NAMES
RETURN
Write record
Main program


STOP
LOGGING
logging=0
timer=0ff
newdisk=0
NEWDISK=1
RETURN
main program
Write Record


ALT-F2
stop logging
STOP LOGGING
logging=0
timer=0ff
newdisk=0
NEWDISK=1
RETURN
main program
ALT-F1
start logging
OPEN FILE FOR LOGGING
flagh=0 flagf=0
IF
NEWDISK=1
CHECK DISK
SPACE
IF
drive$=Fdrive$
theb=n
NEWDISK=0
ESTABLISH
FREE SPACE
SET UP SCREEN
DISPLAY
WRITE DUMMY
FILE
SET TIMER FUNCTION
every 60 Sec
OPEN FILE
NAMES
RETURN
main program


OPEN FILE
line 460
OPEN FILE FOR LOGGING
flagh=0 flagf=0
if newdisk=1
Gosub
CHECK DISK SPACE
If drive$=Fdrive$
then Newdisk=0
Gosub 5000
ESTABLISH
FREE SPACE
SET UP SCREEN
DISPLAY
WRITE DUMMY FILE
SET TIMER FUNCTION
every 60 Sec
OPEN FILE
NAMES
set timer on
numrec=0
RETURN
main program


STOP
LOGGING
logging=0
timer=0ff
newdisk=0
NEWDISK=1
RETURN
main program

APPENDIX A

A.1. GPS Specification

A.1.1. Operating Modes

-Dimensional Navigation

Altitude hold(3 satellites visible)

Altitude hold(less than 3 satellites visible)

Altitude aided(NMEA input)

Dead reckoning (NMEA input)

Automatic cold start (no time or position input required)

A.1.2. Position Update Rate

Once per second(unless set by an external controller ie a PC)

A.1.3. Time to first Fix

minute(typical)

A.1.4. Accuracy

Position (with SA implemented -- 100 meters, 2 drms

Position (with no SA implemented -- 30 meters, 2 drms

Velocity - 0.1 knot

Time Recovery(optional) - 1pps 100 nanoseconds

A.1.5. Self-Test Coverage

Antenna,antenna cable digital sections of the circuit card assemblies, and analog section of RF/power supply circuit card assembly

A.1.6. Remote Control Sources

IBM-PC with control software or Magnavox MX4102GPS unit.

A.1.7. Power input requirement

Vdc, 5 Watts typical

Fuse Rating and type 1 Amp, 3AG3 fast blow

Chassis isolated from electrical ground

A.1.8. Physical Data

RECEIVER

Height	mm(1.8 inches)
Width	mm(5.5 inches)
Depth	mm(6.9 inches)

ANTENNA

Height	mm(7.5 inches)
Diameter	mm(3.25 inches)
Weight (including 15M cable)	Kg(2.5 pounds)

STATUS INDICATOR ANALYSIS

STATUS	Red	Yellow	Green
NORMAL SEQUENCE			
Power OFF or processor CCA failure see Table KKK	OFF	OFF	OFF
Power first turned ON	ON	OFF	OFF
Initial receiver test OK	ON	ON	OFF
Communicating with controlling (attached) device established, but not tracking	OFF	ON	OFF
First satellite being tracked but not navigating	OFF	OFF	FLASHING
Navigating and communicating with attached device	OFF	OFF	ON
ERROR STATUS			
Receiver memory failure	ON	ON	ON
Communication with attached device failed, not tracking satellites	OFF	FLASHING	OFF
Communications with attached device failed, tracking satellites	OFF	FLASHING	FLASHING
RF Self-Test Failed	FLASHING	OFF	OFF
Antenna cable Self-Test Failed	FLASHING	FLASHING	OFF
Antenna Self-Test Failed	FLASHING	OFF	FLASHING
Oscillator SELF-TEST Failed	FLASHING	FLASHING	FLASHING

MX4200 TROUBLESHOOTING TABLE

STATUS INDICATORS			POSSIBLE CAUSE	RECOMMENDED ACTION
RED	YELLOW	GREEN		
OFF	OFF	OFF	Fuse Blown	Replace fuse
ON	ON	ON	Receiver memory failure	Replace CCA
OFF	FLASHING	OFF	Receiver Processor/navigation processor failure	Replace CCA
FLASHING	OFF	OFF	RF self-test failed	Replace RF/power supply CCA
FLASHING	FLASHING	OFF	Antenna cable problem	1.Tighten cable connections 2.Replace antenna
FLASHING	OFF	FLASHING	Antenna problem	Replace antenna
FLASHING	FLASHING	FLASHING	Reference oscillator problem	Replace RF/power supply CCA

APPENDIX D

D.1. Compass Wiring.

The compass as used in GAFERS is not wired as the handbook. A Universal Interface Card is fitted to the unit providing RS232,NMEA0183 and Sine/Cosine Analog outputs . The wiring is detailed in table HHHHH.

Output Cable Wire Colour	Function
Red	+12V;power input , 12Vdc
White /Red	N=1;output line for N+1 signal
Violet	NMEA 0183 OUT (+); sensor output
White/Violet	NMEA 0183 OUT (-); reference ground for output
Black	PGround; power ground input from ship's ground
Black/White	TxD; RS232 transmit data output (no handshaking)
Green	NMEA 0183 In (+); data input stream
White/Green	NMEA In(-); data input stream
Blue	Sin 2; sine output for 2nd sin/cos channel
White/Blue	Cos 2; cosine output for 2nd sin/cos channel
Yellow	Sin 1(+); sine output for 1st sin/cos channel
Yellow/Yellow	Sin 1(+); inverted sine output for 1st sin/cos channel
White	Vref 1; voltage reference output for 2nd sin/cos channel
White/Black	Vref 2; voltage reference output for 2nd sin/cos channel
Brown	Cos 1(+); cosine output for 1st sin/cos channel
White/Brown	Cos 1(+); inverted cosine output for 1st sin/cos channel
Orange	Damp Out 0; control input; Ground to reduce damping
White/Orange	Damp Out 1; control input; Ground to reduce damping

Damping Level	Update	Connect to ground
Damping 1	seconds	Damp Out 0 & Damp Out 1
Damping 2	seconds	Damp Out 1 only
Damping.3	seconds	Damp Out 0 only
Damping 4	Variable **	Do not ground either wire
** same level as selected (0-9) on Azimuth 314AC display		

APPENDIX E

E.1. Circuit of Interface/Power Unit

Electronic Component Description					Identification		
Circuit Diagram Symbol	**Name**	**Rating**	**Grade**	**Type**	**Trade or Suppliers Name**	**Reference Number**	**Alternatives or Remarks**
Con 1	Connector Cir			6 Pin Free Plug	RS Components	RS 466-618	
Con 2	Connector Cir			3 Pin Free Plug Connector Shell	RS Components RS Components	RS 475-707 RS 475-937	
Con 3	Connector 'D'			25 Pin Free Skt	RS Components	RS 466-220	
Con 4	Connector 'D'			9 Pin Free Plug	RS Components	RS 472-437	
Con 5	Connector 'D'			9 Pin Free Skt	RS Components	RS 472-843	
Con 6	Connector IEC			3 Pin Free Skt	RS Components	RS 488-208	
Con 7	Connector 'D'			9 Pin Free Plug	RS Components	RS 472-437	
Con 8	Connector 'D'			9 Pin Free Plug	RS Components	RS 472-437	
Con 9	Connector			2 Pin Free Plug	Magnavox		
Con 10	Connector Coax			RG-6/U F-Type Plg	RS Components	RS 483-001	
Con 11	Connector 'D'			25 Pin Free Skt	RS Components	RS 466-220	

Remarks	Issue	Date	Remarks	Issue	Date	Remarks	Issue	Date
First Draft	1	12.2.92						

GPS and Fluxgate Recording System(GAFERS) - Cable Connectors

Institute Of Oceanographic Sciences

Electronics Comps Drg No. I.O.S./

Compiled By : K.Birch

Sheet Number : 2

Date

Electronic Component Description					Identification		
Circuit Diagram Symbol	**Name**	**Rating**	**Grade**	**Type**	**Trade or Suppliers Name**	**Reference Number**	**Alternatives or Remarks**
IC 1	Intergrated Circuit			RS422A	RS Components	RS 633-098	
IC 2	Intergrated Circuit			RS422A	RS Components	RS 633-098	
PSU 1	Power Regulator				RS Components	RS 591-809	
PSU 2	Power Regulator				RS Components	RS 591-310	
Case					RS Components	RS 501-238	
Con 1	Connector Cir			6 Pin Fixed Skt	RS Components	RS 466-646	
Con 2	Connector Cir			3 Pin Fixed Skt	RS Components	RS 475-707	
Con 3	Connector 'D'			25 Pin Fixed Plug	RS Components	RS 466-191	
Con 4	Connector 'D'			9 Pin Fixed Skt	RS Components	RS 472-843	
Con 5	Connector 'D'			9 Pin Fixed Plug	RS Components	RS 472-437	
Con 6	Connector IEC			3 Pin Fixed Plug	RS Components	RS 488-191	
Boot 1	Saftey Boot			IEC Saftey Boot	RS Components	RS 544-112	5 per Packet

Remarks	Issue	Date	Remarks	Issue	Date	Remarks	Issue	Date
Case and Components	1	11.2.92						

GPS and Fluxgate Interface Unit

Institute Of Oceanographic Sciences

Electronics Comps Drg No. I.O.S./

Compiled By : K.Birch

Sheet Number : 1

Date