SUPERVISOR 214

Computer Graphics and Image Display System

Software Manual

Gresham Lion (PPL) Ltd.
Gresham House
Twickenham Road
Feltham
Middlesex

Tel : 01-894-5511 Telex 27419

A wholly owned subsidiary of Gresham Lion Limited

SUPERVISOR 214

SOFTWARE MANUAL

LIST OF CONTENTS

SUPERVISOR 214

LIST OF CONTENTS

SECTION 1

INTRODUCTION

LIST OF CONTENTS

SECTION 1

INTRODUCTION

1.0 Overview

Supervisor 214 is a modular display system that may be used for computer graphics or image processing. The hardware is offered with or without an LSI11 micro-computer as local intelligence.

Thus four categories of system exist:

On-line Graphics) On-line Imaging)	No LSI11
Off-line Graphics) Off-line Imaging)	LSI11 as local intelligence

Within each category variations of hardware and software are available. This section describes the various configurations, the facilities they offer, and the software which supports them.

1.1 Comparison of On-line and Off-line Systems

On-line systems offer the lowest cost and potentially the fastest response but the host computer has to do all the work. If the host is heavily loaded an Off-line system will usually prove faster despite the time taken by interprocessor communication. At present, on-line systems are only available for PDP11 and LSI11 hosts.

Off-line systems offer VDU emulation and local processing which are not available with on-line systems. VDU emulation allows the system to function like a standard computer terminal with a keyboard and alpha-numeric display. The VDU dialogue may be displayed simultaneously or alternately with the graphics or image display. Local processing allows the user to perform a variety of operations locally under keyboard controls without involving the host at all.

Off-line graphics systems are normally satisfactory with a 9600 Baud serial interface between the host and the LSI11. High activity, multichannel graphics may require a Direct Memory Access (DMA) or Programmed Word Transfer (PWT) interface.

Off-line imaging systems generally require a DMA interface to achieve sufficiently fast image transfer. When a DMA interface is fitted, the Serial Interface is retained for control purposes.

.2 Software Interface

On-line and off-line systems have compatible software interfaces so that programs written for one type of system may run on another.

In on-line systems a set of subroutines, coded in Macro 11, is provided to run in the host and perform directly all the graphics and imaging functions. These subroutines are called by standard Fortran or Macro statements.

In off-line systems,similar subroutines are provided for the host which are called in the same way but do not perform their function directly. Instead, they use the terminal driver of the host operating system to send a function code and arguments via the serial interface to the LSI11. The LSI11 is provided with a program to receive and execute the functions. This program may be held in ROM or may be down-line loaded.

1. 3 On-Line Graphics and Imaging Systems

Systems in which Supervisor 214 is interfaced directly to the host computer without local intelligence.

1.4 On-Line Interfaces

Interfaces exist, at present, for the following computers:

PDP11 Series 1 Quad card for SPC slot
LSI11 Series 1 Dual card for Q Bus slot

1.5 On-Line Software

A librar y of subroutines is available for PDP11 and LSI11 hosts. These are coded in Macro 11 and may be linked to and called by host programs which themselves are coded in Macro, Fortran or other high level languages. The functions available are given in Appendix A and the format of the call in Appendix B.

The full set of graphics subroutines occupies less than 4k words and the imaging and graphics subroutines combined less than 8k words. It is unlikely that a particular host program will use all the available subroutines and so in practice the storage required will be less than 4 or 8k words.

The subroutines are supplied in the form of object libraries. For graphics only, the library is called GLIB and for combined graphics and imaging, ILIB. These libraries may be kept separate or merged with the hosts' system library.

The subroutines detect various errors and these are reported on the console device by a common processing module, versions of which exist for the RT11 and RSX11M operating systems.

6 Address Utilisation - on-line

The registers of Supervisor 214 appear in the I/O page of the hosts' virtual address range. The number of addresses required varies from 16 to 64 depending on the options with which Supervisor 214 is fitted. The standard base address is 770000 but this may be adjusted on 64 word boundaries to avoid conflict with other devices in the users system.

Imaging systems can support Direct Memory Access (DMA) transfers from disc, tape or Digivisor to the Supervisor 214 display memory. For this a Data Page is required in the hosts virtual address space. The standard data page is 512 words from 762000 to 763776. The size of the data page may be adjusted by factors of 2 from 512 words to 4096 words and its base address may be altered to avoid conflict with other devices or memory. The base address may be anywhere in the virtual address range and is not fixed in the I/O page. However, it moves on boundaries of 512 to 4096 words according to its size.

1.7 On-Line Graphics, Standard Functions

Program definition of graphics display channels
Multiple display channels*
Multiple users - (on RSX11M systems)
Hardware erasure
Alphanumerics - standard and magnified
Vectors, absolute & chained
Boxes
Rectangles
Circles
Area Fill
Single Pixel read/write
Monochrome & colour graphics with overlays*
Hardware cursor
Software generated cursors*
Area flash*
Watch-dog timer
Vertical roll and multiple display pages (Single user or single channel only)
Zoom

* Subject to sufficient pixel planes being available.

1.8 On-Line Imaging, Standard Functions

All graphics functions listed under 1.7 plus:

Program definition of image display channels
Write image line
Read image line
Unpack integers into Pixels
Pack pixels into integers
Zoom

Page copy
Intensity/colour mapping with graphics/cursor overlay*
4 way output channel selection*

*Subject to sufficient pixel planes being available.

1.9 On-Line Optional Functions

Programmable Display format
Hardware cursor
Joystick or trackerball control unit
Graphics colour mapping output card
Imaging pseudo-colour output card
ASC11 generator card, 80 x 24 characters
Image capture from Digivisor.

1.10 Off-line Graphics & Imaging

Systems in which Supervisor 214 has an LSI11 as local intelligence and a serial interface to a host.

1.11 Off-line Interfaces

Systems requiring only a serial interface may be connectd to virtually any host computer using either the 20 milliamp loop, RS232, RS422 or RS423 or equivalent transmission.

DMA interfaces for PDP11 and LSI computers are provided as standard using the following hardware:

PDP11 Series Able Buslink Univ/Q 1 Hex SPC card
LSI11 Series Able Buslink Q/Q 1 Quad Q Bus card

DMA interfaces for other computers may be provided to special order as follows:

GEC 4080
Prime
Interdata/Perkin Elmer
Hewlett Packard
Floating Point Systems Array Processors

12 Off-line Software

A module of subroutines (HLIB) is available for PDP11 and LSI11 hosts which is coded in Macro 11. This may be linked to & called by host programs which, themselves, are coded in Macro, Fortran or other high level language. For graphics systems the module requires about 1100 words of storage independent of the number of subroutines actually called. For combined imaging and graphics the module occupies approximately 1500 words.

For other host computers, a library of subroutines FLIB is available, coded in Fortran, which may be linked to host Fortran programs.

In both cases the subroutines generate ASC11 messages and transmit them to the Supervisor 214 via the host computers standard terminal driver. As an alternative to use of either of the subroutine libraries, the user may write his own code to generate the simple messages required. Details of the message format are given in Appendix C.

Appendix A describes the functions which are available.

1.13 Address Utilisation Off-line

Off-line PDP11 and LSI11 systems require address space in the I/O page for the serial interface. This is dependent on the type of serial interface hardware employed. If a DL11 type is fitted four words of address space will be needed plus a floating vector. If a channel of a serial multiplexer is used the address allocation may already exist.

The DMA interface, if fitted, requires four words in the I/O page. The default base address is 772410 but this may be moved on 4 word boundaries if necessary to avoid conflict. The default vector address is 300 and this also may be moved if necessary.

1.14 Off-line Systems - Standard Functions

Off-line systems offer the same standard functions as on-line systems and these are listed in sections 1.7 and 1.8.

1.15 Off-line Systems - Optional Functions

ASC11 Keyboard & VDU generator card providing:

VDU emulation with separate storage for VDU text & local processing under keyboard control
Error message display

ROM memory for LSI11 program

All optional functions listed in section 1.9.

SECTION 2

CODING OF USER PROGRAMS

LIST OF CONTENTS

SECTION 2

CODING OF USER PROGRAMS

2.0 On-line and Off-line Systems

The coding of user programs for on-line and off-line systems is identical and a given program will run on either type of system provided that both have adequate hardware.

2.1 Initialisation

The first step in all programs is to call subroutine SET to initialise the hardware and software to a particular display resolution. This is followed by one or more calls to subroutine DEF to define display channels. Then SEL will be called to select a display channel.

After this the various graphics and imaging subroutines may be called as required.

On-line systems have their hardware initialised by the CPU Bus Init. signal on power-up to the fixed resolution determined by the Format PCB or to 384 x 293 if the Programmable Format Option is fitted. However, the software requires that subroutine SET be called both to initialise the software and to ensure that a data area in SET is present for use by other subroutines.

Off-line systems contain subroutine SET in their local program and this is called automatically during the power-up procedure or after down-line loading. In theory, therefore, SET need not be called by the user program provided the default resolution is correct. In practice SET should always be called because there is no guarantee that the initial setting will not be changed by another program or by operator action in 'Local'.

SET erases all memory planes in the system and so RUB is not usually needed at the initialisation stage.

In on-line multi-channel, multi-user systems a special version of SET is available which does not erase any memory planes. This allows a user to initialise his program without erasing another users memory planes. With this version, RUB must be used by each user to clear his channel during initialisation, All users must specify the same display resolution and avoid contention problems by the use of Event Flags or similar techniques.

2.2 Graphics and Imaging Subroutines - General

After initialisation,the imaging and graphics subroutines may be used freely in accordance with the rules of the language in which the program is being written. For example, they may be included in loops and in high level subroutines.

The detailed operation of each subroutine is described in the header of the sub-routine listing. The subroutines and their allowable arguments are summarised in Appendix A.

2.3 Co-ordinate Chaining

Where appropriate,the subroutines allow chaining of co-ordinates so that connected shapes may be drawn without the need to specify new co-ordinates.

After each subroutine call involving X and Y co-ordinates current X and Y values are updated. Subsequent calls may use these as default values. For example, the vector subroutine revalues XC and YC to X2 and Y2. A subsequent call to VEC can omit X1 and Y1 and the new vector will be drawn from the end of the previous vector. Similarly, the alpha-numeric routine revalues XC and YC to X & (Y + 12). Thus, a subsequent call to ALP omitting X and Y will generate the character string one row below at the same horizontal position.

2.4 Arrays as Arguments

When subroutines use arrays as arguments the arrays must be pre-defined in accordance with the rules of the language in which the program is being written.

2.5 Interactive Subroutines

Subroutines such as ROL, CUR and ZOO return one or more values to the calling program. The programmer must ensure that arguments are pre-defined to receive these values.

In off-line systems the subroutines make the necessary I/O calls via the Terminal Driver to receive the returned values. They are then converted from their ASC11 format to a binary integer and placed in the correct storage location. They may then be referenced in the normal way by subsequent statements in the program.

The subroutines do not exit until the return values are received.

2.6 Overlays, Cursors & Area Flashing

There are two methods available for logically combining overlay, cursor and area flash information with the main picture. One uses logic contained on the memory cards and the other uses additional graphics or imaging output cards which embody mapping tables.

Each individual system is supplied with a configuration drawing which shows the method employed. The drawing is located at the rear of the Hardware Manual and should be consulted for specific programming detail.

The following sections describe the general principles of programming for both methods of output.

2.6.1 Memory Card Output Logic

Each memory card embodies a circuit for mixing the cards own picture with an overlay picture received from another card. The card providing the overlay picture may be a pixel memory card or a cursor card. The visibility of the overlay is controlled not by the card doing the mixing, but by the card on which the overlay picture originates. This will be explained more fully later.

Each memory card also embodies a circuit whereby the card's own picture may be inhibited by an external signal. The card also has a Flag which the programmer can use to allow the inhibiting signal to inhibit or to cause it to be ignored. This flag is called the Output Flag.

The inhibiting signal can be of three types. Firstly, a steady inhibiting voltage derived from the backplane wiring. In this case, setting the Output Flag will inhibit the picture and clearing the Flag will enable the picture. Secondly, an alternating inhibit/enable voltage derived from the backplane wiring. In this case, the output picture flashes when the Flag is set and is steady when the Flag is clear. Thirdly, the signal may be derived from another pixel plane in which case the inhibiting signal has a spatial component. Thus, when the Flag is set the picture is inhibited in particular areas as defined by the contents of the inhibiting plane. Finally, the inhibit circuit on the inhibiting plane may itself be connected so that the inhibiting signal flashes. The result is spatial and temporal control of inhibiting. Now defined areas of the original picture may be made to flash.

In cases one and two the programmer action is to set or clear the Output Flag on the original memory plane. In case three the action is set or clear the Output Flag on both the original plane and the inhibiting plane.

Returning now to the control of an overlay signal, it will be seen that the inhibit circuit on the overlay plane can be used for this purpose. It may be wired for a steady overlay which is turned on or off by clearing or setting the overlay plane Ouput Flag respectively.

Or, it may be wired for a steady or flashing overlay obtained by clearing or setting the Output Flag.

In either case, the action for the programmer is to set or clear the Output Flags on the Overlay Card.

The software means to control the Output Flag is the WRI subroutine. This allows any Supervisor 214 register to be written.

The output Flags are in Register 8 for planes 0 to 15 and Register 12 for planes 16-31.

For example, to enable area flashing where the original picture is in planes 0, 1 and 2 and the flashing area is defined by plane 3:-

CALL WRI(8, "17).

The octal pattern "17 sets the output flags in all four planes.

2.6.2 Image and Graphics Output Cards

These embody a hardware mapping table(s) connected so that each pixel memory plane or cursor, in effect, provides an address bit for the table(s). The contents of the table drive the display to required intensity and/or colour. The programmers task is thus, to load the table(s) so that each combination of pixel and cursor input produces an appropriate output. Full details of the Image, Pseudo-colour and Graphics output card are given in Appendixes F, G and H.

For example, with a Graphics Output Card planes 0, 1 and 2 could provide address bits 0, 1 and 2 of the table and contain the Red, Green and Blue components of a picture. Plane 3 could be a hardware cursor driving address bit 3. In the absence of a cursor (address bit 3 = 0), locations 0 to 7 of the table will be addressed and should be loaded as follows to map the RGB components correctly:-

Plane Output				Colour	Table Address	Table Contents		
Cursor	Blue	Green	Red			Blue Output	Green Output	Red Output
0	0	0	0	Black	0	0	0	0
0	0	0	1	Red	1	0	0	15.
0	0	1	0	Green	2	0	15.	0
0	0	1	1	Yellow	3	0	15.	15.
0	1	0	0	Blue	4	15.	0	0
0	1	0	1	Magenta	5	15.	0	15.
0	1	1	0	Cyan	6	15.	15.	0
0	1	1	1	White	7	15.	15.	15.
1	-	-	-	White	8 - 15	15.	15.	15.

- = Don't Care

When the cursor is present, locations 8-15 of the table will be addressed. Assuming that the cursor is to appear in white, locations 8-15 all loaded with white, i.e. intensity 15. in the Red, Green and Blue tables. (In practice the ability of the Graphics Output Card to produce more subtle colours would require values other than 0 to 15 in the tables).

When Output cards are used for output combination, the memory plane circuits are not used and the Output Flags can remain Clear, which their default state. An exception to this is when flashing is required. In this case, the plane whose output is to flash e.g. the cursor, or an area flash plane, has its inhibit input wired to the appropriate backplane signal and its Output Flag must be set as described in section 2.6.1.

The programmer has two methods of loading the mapping tables. One is to use subroutine LUT to load linearly changing or constant data. The other is to use WMP to copy an array into the table(s). The array may be filled from a file or set up by the program.

2.7 Example Fortran Program

The example below illustrates the method of coding a simple graphics picture. Appendix B describes the Fortran interface in detail.

```
C *** TEST ALL GRAPHICS SHAPES, FLASH, CURSOR AND ROL FUNCTIONS
      CALL SET(3)              !   Set resolution to 384 x 292
      CALL DEF(0,1,3,0)        !   Define RGB colour graphics channel
      CALL DEF(1,1,1,3)        !   Define monochrome flashing area channel
      CALL SEL(0,7,0)          !   Select graphics channel white foreground, black background
      CALL BOX(383,292,0,0)    !   Box around the edge of picture
      CALL BOX(25,25)          !   Chained co-ordinates
      CALL VEC(383,0,0,292)    !   Diagonal vector
      CALL VEC(10,200,50,150)  !   Draw a triangle
      CALL VEC(100,200)
      CALL VEC(10,200)
      CALL FIL(50,280)         !   and fill it
      CALL MAG(125,10,3, 'SUPERVISOR)
      CALL ALP(140,50, 'GRESHAM LION')
      CALL ALP('PPL LTD.')
      CALL ALP(140,100,8, 'FEB 1981234')
      CALL ALP(9, 'MAR 1981234')
      CALL SEL(0,6,0)          !   Graphics channel, cyan on black
      CALL CIR(75,75,50,"170)  !   Partial circle
      CALL CIR(75,75,40)       !   Full circle
      CALL SEL(0,1,0)          !   Single red pixel
      CALL BIT(75,75)
      IC=6                     !   Use a DO loop to draw some vectors in colour
      IX=54
      IY=41
      DO 100 I=1,6
      CALL SEL(0,IC,0)
      CALL VEC(IX,292,383,IY)
      IX-IX+54
      IY=IY+41
      IC=IC-1
 100  CONTINUE
      IX=100                   !   Get cross wire cursor values & type them
      IY=100
      CALL CUR(IX,IY)
      TYPE 200,IX,IY
      CALL SEL(1,1,0)          !   Draw area flash back
      CALL BLK(140,50,224,60)
      CALL WRI(8,"17)          !   Enable flashing logic*
      IX=75                    !   Get box cursor values and type them
      IY=75
      CALL CUR(IX,IY,50,50)
      TYPE 200,IX,IY
      IX=0
      CALL ROL(IX)             !   Roll the picture and type top line no.
      TYPE 200,IX
200   FORMAT(1X,2I5)
      CALL EXIT
      END
```

* The precise method of enabling overlays and area flashing etc. is dependant on the configuration of the particular system. All systems are supplied with a configuration drawing which shows the logic interconnections of the memory cards and how they are controlled.

2.8 Coding Macro-Programs on PDP11 & LSI11 Hosts

The same principles apply as for Fortran programs with the CALL statement replaced by JSR PC, MNE. Each subroutine call must be preceded by instructions to set up the argument table and its pointer as described in Appendix B3.

The subroutine names must be declared as global symbols.

SECTION 3

OPERATING PROCEDURES

LIST OF CONTENTS

SECTION 3

OPERATING PROCEDURES

3.0 General

This section is written with specific reference to PDP11 or LSI11 hosts running under the RT11 or RSX11 operating systems. Similar principles apply to other host machines.

3.1 Compiling or Assembling User Programs

This is done in the normal way appropriate to the language in use.

3.2 Linking or Task Building User Programs

The procedure is straightforward for both on-line and off-line systems under RT11 and is best illustrated by the following typical link commands:-

On-line system:-

```
    .R LINK
*   DEMO,LP : = DEMO, GLIBRT
*   ↑C
```

Off-line system:-

```
    .R LINK
*   DEMO,LP : = DEMO, HLIBRT
*   ↑C
```

where DEMO is the name of the user object module
GLIBRT is the on-line graphics library
HLIBRT is the RT11 off-line graphics module

Under RSX11M, Task Builder options must be specified as follows:-

On-line systems:-

```
      >TKB
TKB>   DEMO,LP   = DEMO, [G,U] GLIBRX/LB
TKB>   /
       ENTER OPTIONS :
TKB>  COMMON = COM214 : RW
TKB>  //
```

Off-line system :-

```
>TKB
  TKB>  DEMO,LP = DEMO, [G,U] HLIBRX
  TKB>  /
        ENTER OPTIONS :
  TKB>  ASG = TTn : 1
  TKB>  //
```

where DEMO is the name of the user object module
GLIBRX is the on-line RSX11M graphics library
[G,U] is the group and user under which the library is stored
HLIBRX is the off-line RSX11M graphics module
COM214 is the Device Common established for access to the S214 registers (see section4)
TTn is the terminal to which Supervisor is connected
1 is the LUN used by HLIBRX to access the terminal driver

3.3 Running User Programs

No special actions are necessary to run user programs.

Off-line Supervisor 214 systems emulate VDUs and so are able to function as a conventional VDU. Supervisor can thus be the console device under RT11 or a terminal under RSX11M. In either case the Supervisor keyboard can be used to type the command to run the user program. Alternatively, under RSX11M, the user programs can be run from any other terminal.

3.4 Filing User Program Output

Off-line systems are controlled by a stream of commands which consist of ASC11 characters. It is possible to redirect this stream to a disc file instead of to the Supervisor 214 terminal. The Peripheral Interchange Utility PIP may then be used to transfer this file to Supervisor.

This indirect method has the advantage of requiring the user program, which may be time consuming, only to be run once. The file may be edited to correct minor errors or printed to assist in de-bugging the user program.

3.5 Power -Up Procedures for Off-line Systems

If the off-line system has its programs in ROM it will power-up and behave immediately as a fully intelligent graphics or image terminal.

If the program is held in volatile RAM the system will power-up in a bootstrap program and wait to be down-line loaded. Loading may be initiated manually or from the host start-up indirect command file.

3.6 Down-line Loading Off-line Systems

The procedure is to run the loader program which is called SLALRT.SAV or SLALRX. TSK under RT11 or RSX11M respectively. The programs prompt for a file name and when this is entered, loading begins. The bell is sounded after every 512 words of program have been loaded. Upon completion, the loader returns control to the monitor and the Supervisor system executes its initialisation procedure and is ready for use. If a file is not entered but CR is typed instead, a default file name of VDUCON.SAV or VDUCON.TSK is used.

If it is required to re-load the Supervisor system it must first be forced into the bootstrap program by powering down & up using the DC powerswitch on the Supervisor box.

SECTION 4

SOFTWARE INSTALLATION

LIST OF CONTENTS

SECTION 4

SOFTWARE INSTALLATION

4.0 General

This section refers specifically to installation on PDP11 or LSI11 computers running under the RT11 or RSX11M operating systems. Similar principles apply to installation on other computer systems.

Installation mainly consists of copying the software from the distribution medium to appropriate places in the host computer's backing storage. In the case of on-line RSX11M systems it is also necessary to create a Device Common for access to the memory I/O page.

4.1 RT11 with On-line Supervisor 214

Copy the Graphics subroutine library GLIBRT . OBJ or the Imaging subroutine library ILIBRT . OBJ from the distribution medium to a convenient storage device, normally the system disc.

Alternatively, the library may be merged with the System Library using Librarian. In this case, it is not necessary to name GLIBRT or ILIBRT as an input when linking.

4.2 RT11 with Off-line Supervisor 214

Copy the host module HLIBRT . OBJ from the distribution medium to a convenient storage device, normally the system disc. Alternatively, merge HLIBRT . OBJ with SYSLIB . OBJ as described in section 4.1

If the Supervisor 214 system is down-line loadable, copy the loader program SLALRT.SAV and the off-line program VDUCON.SAV to a convenient device such as the system disc where they can be accessed after powering-up the computer.

Finally, copy the Sources and Object modules of all the components of the off-line program to host storage if it is intended to create a user-tailored version of the off-line program.

4.3 RSX11M with On-line Supervisor 214

Copy the Graphics or Imaging subroutine libraries GLIBRX.OLB or ILIBRX.OLB to a convenient storage device, normally the system disc. Alternatively, merge the libraries with the System Library SYSLIB.OLB.

Create a memory partition using VMR or MCR for the device common as follows:

```
MCR>   SET/MAIN = COM214 : 7700 : 2 : DEV
```

Copy the device common object module COM214.OBJ to a convenient storage device and task build the device common as follows:-

```
TKB>  LB : [1,1] COM214/-HD/PI,LP,LB:[1,1] COM214 = COM214
TKB>  /
ENTER OPTIONS:
TKB>  STACK = 0
TKB>  PAR = COM214 :0 :200
TKB>  //
```

Finally, install COM214 using MCR or VMR as follows:

```
MCR> INS   LB : [1,1] COM214.
```

Refer to the program listing for COM214 if the Supervisor 214 is to be installed with a device address different from 770000.

4.4 RSX11M with off-line Supervisor 214

Use the same procedure as for an off-line RT11 system as described in section 4.2.

The appropriate module names are:

Host Module	HLIBRX.OBJ
Loader	SLALRX.TSK
Off-line Program	VDUCON.TSK

SECTION 5

MODIFYING AND CREATING FUNCTIONAL SUBROUTINES

LIST OF CONTENTS

SECTION 5

MODIFYING AND CREATING FUNCTIONAL SUBROUTINES

5.1 Modifying Functional Subroutines

To modify an existing subroutine, refer to the listing in Appendix Z and to the Supervisor 214 Hardware Manual to gain an understanding of how it works.

5.2 Creating Functional Subroutines

New subroutines must conform to certain requirements as follows:

1. The function of the subroutine should be fully explained in the header text of the source code. Specific hardware requirements should also be listed and error messages detailed.

2. Read-only instructions and constants should be put in a program section defined as follows:

   .PSECT USER $I,RW,I,LCL,REL,CON.

   Read/write variables and working storage should be put in a program section defined as follows:

   .PSECT USER $D,RW,D,LCL,REL,CON.

3. The entry point must be labelled and defined as global. Global symbols referencing locations in other subroutines should be declared global.

4. The subroutine must conform to the calling formats defined in Appendix B.

5. As a first step, the subroutine must check that sufficient arguments have been supplied.

6. Judgement is needed as to the extent of further argument checking. A balance is required between bulk and speed of execution on the one hand and the need to give diagnostic messages to the programmer on the other. In many cases an error will be diagnosable from the resulting picture.

7. Only one ASC11 string or Hollerith is allowed and it must be the last argument.

   All other arguments must be 16 bit integers.

8. When error messages are required, subroutine GERRPR must be called. This provides centralised error reporting and allows the error action to be easily changed to suit different user needs and different operating systems.

Before calling GERRPR, the error word, GERRWD must be set up. The lower byte contains a code indicating the desired error message. Refer to the listing of GERRPR for the available messages and their code numbers. Add new messages to GERRPR if required.

The upper byte of GERRWD contains a code to indicate the subroutine name which is to be printed as part of the message. Include the appropriate name in GERRPR.

Bit 15 of GERRWD must be set to 1 if the error is fatal. This causes the subroutine to be aborted. The precise action depends on whether the system is on-line or off-line. In the on-line case, a trap instruction is executed after printing the error message on the console terminal. Fortran intercepts the Trap and adds further diagnostic messages before returning control to the Monitor. The off-line error processor displays the error message in the VDU plane before returning control to the off-line monitor. In both cases, the subroutine nesting level is assumed be 2 before the TRAP instruction, i.e. the functional subroutine is level 1 and the error processor level 2.

In certain environments it is not appropriate for the Error Processor to TRAP. To cater for this, calls to GERRPR which are fatal should be followed by RTS, PC. Thus, the subroutine will abort itself if the Error Processor does not.

9. In general, the subroutines should leave the following undisturbed on return:

The Processor Stack

Supervisor 214	Command & Status Register
Supervisor 214	Access Flags
Supervisor 214	Output Flags
Supervisor 214	Foreground and Background Flags

Registers R0 to R5 need not be preserved.

5.3 Modifying the Functional Subroutine Library

When a subroutine is modified or created it is assembled in the normal way. It must then be replaced or inserted in the library GLIBRX, GLIBRT, ILIBRX or ILIBRT as appropriate. This is done using the Librarian utility, again in the normal way described in DEC literature.

The modified library is then ready for use to build on-line user programs or to build the off-line program as described in sections 3 and 5 respectively.