M. BARLOW

1. General

The Disk Operating System is provided for program creation and debugging on P855-P860 Sagittaire family computers.

The minimum configuration is :

- 8k words of core
- P855/P860 C.P.U.
- ASR 33 typewriter for system operator
- One moving head disk (PHILIPS X1210 like) = 1.344 M words but a system based on fixed head disk will be provided as an enhancement .

Optional equipment will include :

- additional disk units
- real time clock
- high speed paper tape equipment
- ASCII code lineprinter
- cassette tapes
- card reader
- mag tapes

Operating procedures are mainly keyboard oriented, using a set of control statements for loading and starting the standard processors.

Disks are used to store the processors, their input and output files and also the user files.

A general catalog is present on each disk : entries in this catalog are related to users identification. For each user identification a special directory is provided in order to keep the names of permanent user files.

File protection is accomplished by means of user identification.

2. Functional Characteristics

2.1. General Concepts

2.1.1. Core Space Allocation

After the Monitor has been loaded, the unused space in core is allocated for user needs. This space is used to load user programs according to operator's or user's requests.

2.1.2. Disk Space Allocation

When using the system, only temporary files are created. Such a file is made of an integer number of disk allocation units which have the same size = one half track, i.e. eight consecutive sectors of 210 words each.
Addition of extra allocation units to the initial one is performed automatically by the Monitor according to user needs. These allocation units are taken from a pool (chain) of free units and given back to the pool when no longer required.
At a given time the address of the first unit free for allocation is stored in a system table. A file will often consist of non-consecutive half-tracks. In the first allocated unit the first sector is reserved for storing the addresses of allocated units.
Maximum file size is thus limited to 210 half tracks.

2.1.3. Operating Mode

A job under DOS control will consist of user identification transmission, possibly followed by a set of control commands for processor calls and always followed by the last control command used to leave the system.

A typical job may consist of :

- a Text Editor call for recording a new text on disk or updating a previously recorded source file.
- a Language Processor call, like an assembler or Fortran compiler, in order to produce object program.
- a linking step could be required to include external modules from object libraries.
- a debugging phase may consist of user program execution under DEBUG processor control, with breakpoint insert and dump output.
- a call for the utility package may be needed if the user wants to take a hard copy of any file, etc...

Each user will receive from the Monitor his own set of file codes only available during the current connection.

Except if explicitly required by the user, it is not possible to access other users'files. It is possible after any stage in the above job to save the state of the temporary files by making them permanent. This enables a restart to be made from the desired job stage in case of a subsequent stage proving erroneous. Later on, the user could assign file codes to these permanent files, retrieve his job and perhaps delete some of these permanent files.

2.2. Disk Organisation

2.2.1. Physical Organisation

On a X1210 disk unit, the available XMX 1416 cartridge is organised in 203 cylinders
of 2 tracks each /cyl.
of 16 sectors each /track
of 210 words each. /sector

There is one head for each track on the current cylinder. The average head positioning time is 125 ms and a half revolution requires 37,5 ms.

A seek operation between two consecutive cylinders requires 30 ms. Data transfer rate is about 50 K words/sec. Other disk types could be added later, assuming the sector length is equal to 210 words

2.2.2. Logical Organisation

On this small disk, the maximum record size will be used on sectors. The logical record length is variable without any upper limit. If needed, a long record will be written on several sectors. The Monitor will handle blocking the logical records in physical blocks without user intervention. To effect this, the Monitor will require a blocking buffer in the user area.

This organisation will comply with sequential processing, but random access will also be possible at physical sector level. In this latter case, the user must consider the file as a set of consecutive half tracks, even if not the case, in which the sectors are numbered from zero to n. As for sequential files, the first sector of the first allocated half track is reserved as an index. Random access to such a file will require the relative sector number (between 1 and n) to be given before the request is performed.

2.2.2.1. General Catalog

On any disk, the first cylinders are reserved for system purpose : standard volume label written in absolute sector # 0, general catalog of permanent files requires sectors from 1 to 7 inclusive, disk resident Monitor area and core image standard processors will be written on the following sectors.

The general catalog is a separate file made of this first disk allocation unit possibly chained with others, in which entries have a fixed length (8 words per entry) and refer to the user identification codes. For each user present in this general catalog a first half track is allocated and may be used to record the names and type of his permanent files.

2.2.2.2. Library Directories

The user files may be grouped in libraries, that means the type of such a file becomes "library" and the file itself is made of two separate files.
The directory part of a library file will contain control information about name and location of each library component.
The body of the library itself is made of consecutive library components, each of them starting at a sector boundary. When creating a library file, two different allocation units are allocated to this file : the first will be the head of the directory entries chain, the second being the head of the library body.
Adding a new library component will consist of

- first adding a new entry in the directory part
- then copying the library component in the library body part, starting at a sector boundary.

2.2.2.3. File Names

The user can specify either a file name or a library element name.
In the first case the name will consist of one to six ASCII characters.
In the latter case the name will include a period character in order to separate the library name from the component name.

Example : OBJ345 is a file name

OBJECT.MØD123 is the name of the "MØD123" component in the "OBJECT" library.

3. External Information

3.1. Standard Files and Assignments

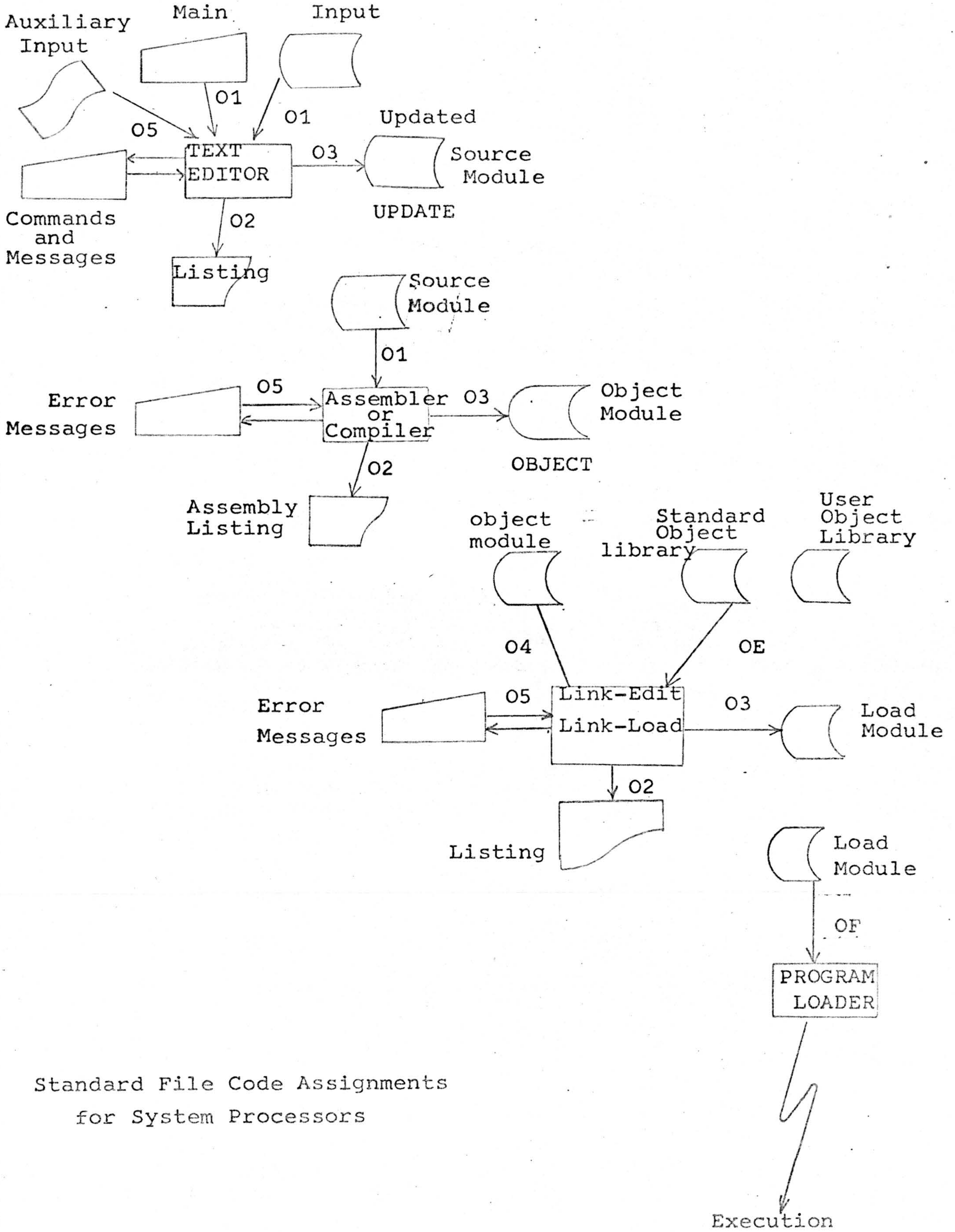
The standard processors perform input and output operations using standard file codes usually assigned to temporary disk files. In any generated system some file codes will always be defined and may not be destroyed. They will be assigned to physical devices or temporary disk files according to the system configuration.

These standard file codes are the following :

X'01' : Standard Source Input
X'02' : Standard Listing Output
X'03' : Standard Punch Output
X'04' : Standard Object Input
X'05' : Standard Operator Keyboard (Input+Output)
X'0E' : Standard Library Input
X'0F' : Standard Program Input

The processors, such as TEXT EDITOR, ASSEMBLER, FORTRAN COMPILER, LINKING LOADER, will use these file code values in their Event Control Block when requesting any I/O operation (LKM DATA 1).

- User file codes must be in the range X'10' to X'7F' since file code values between X'01' and X'0F' are exclusively reserved for system use.



Standard File Code Assignments
for System Processors

3.2. Operator Control Statements

The system operator is able to communicate with the Monitor using first the control panel interrupt then the system teleprinter to type in the control statement. Control statements can only be transmitted to the Monitor after the interrupt button has been depressed and the "M:" message output on the teletype. The syntax is the following :

<statement name>⊔<param1>,<param2>,......,<paramn>

where statement name is written as a two letter name. <parami> is either an ASCII string or a hexadecimal value.

3.2.1. Assign

The statement is used to assign an I/O file code to a physical non disk unit or a permanent area on disk.

Format : AS⊔<file code>,<device physical address>

or AS⊔<file code>,<disk unit logical address>, <file name>

The disk unit logical address is a reserved system file code whose value could be X'OA',X'OB',X'OC' or X'OD'

3.2.2. Pause

This statement puts the running program in Pause State pending operator action

Format : PS

The running program will be restarted by an RS statement

3.2.3. Restart

This statement allows the program

- which has been put in Pause State by a PS operator statement
- or which requested a PAUSE system macro

to be restarted.

Format : RS[␣<new A7 hexadecimal contents>]

The optional parameter could be used to transmit the operator's answer.

3.3. MONITOR REQUESTS

Programs can request the Monitor to perform system functions using a link to monitor instruction (LKM).

The calling sequence consists of

```
   } loading certain registers
 LKM
 DATA  <parameter>(value depending upon the requested
            function).
```

These requests are available during program execution and program development phases.

At run time the user program may require other system functions. In this case the program will be aborted, unless it is running under control of the Debug Package. This Debug Package will only output an error message on the user's typewriter, indicating the requested function is not available and allow the user to decide whether the abortion or continuation of the run is required.

3.3.1. INPUT/OUTPUT REQUEST

Before issuing an I/O request, the program must load two registers with the I/O parameters.

Calling sequence : A7 = I/O ORDER IN RIGHT HAND BYTE

A8 = ADDRESS OF AN EVENT CONTROL BLOCK

LKM

DATA 1

The I/O order is made of a six-bit function code (bits 10 to 15 in A7) and a wait flag in bit#8.

This flag must be set when the program wishes to cease execution up to the point when the I/O operation is completed.

If this flag is not set when the request is performed, the program will regain control as soon as the I/O operation is initiated and must perform a WAIT request in order to achieve Synchronization of the current I/O operation with the current processing (see 3.3.2.).

Orders: Bit 8=1 with implicit wait function	Orders: Bit 8=0 with explicit wait request	Meaning	Returned Status in ECB4
X'81'/X'C1'	X'01'/X'41'	BASIC READ	Hardware
X'82'	X'02'	STANDARD READ	Software
X'85'/X'C5'	X'05'/X'45'	BASIC WRITE	Hardware
X'86'	X'06'	STANDARD WRITE (ASCII)	Software
X'87'	X'07'	STANDARD WRITE (4.4.4.4. Object)	
X'88'	X'08'	STANDARD WRITE (8.8. Object)	
X'94'	X'14'	SKIP UP TO EOS mark	
X'96'	X'16'	-----------EOF-----	
X'A2'	X'22'	WRITE EOS mark	
X'A6'	X'26'	WRITE EOF mark	
X'BO'	X'30'	Get information about a file code	

The Event Control Block is a six word block with the following format

ECB			
ECB	0	Event Byte	File Code
	1	Beginning Buffer Address	
	2	Requested Length in char	
	3	Returned Effective Length in char	
	4	Returned Status	
	5	Tabulation Table Address or Relative Sector Nbr	

ECB0 : The Event byte is used by the Monitor to store system information : bit 0 is set to one when the I/O is completed. The file code byte must be filled before requesting the I/O with a value in the range X'10' to X'7F', it is the logical number of the file and may be defined using an ASSIGN macro or AS statement.

ECB1 : Contains the beginning address of the logical record to be exchanged.

ECB2 : Contains the length in characters of the logical record to be exchanged.

ECB3 : This word is filled by the Monitor after completion of the I/O operation. It gives the actual number of characters which have been exchanged.

ECB4 : After I/O completion contains the hardware or software status depending upon the type of request. (See I/O orders above).

EBC5 : Is only filled by the user to store :

- the tabulation table address in the case of a standard read operation on ASR or paper tape equipment.
- or the relative sector number to be exchanged in the case of a random access disk file.

Returned status (in ECB4)

a) When Basic orders are used, ECB4 will be filled with the hardware status returned by the control unit unless the Monitor detects an error in the calling sequence. In this case the Monitor will fill this word with a software status, in which bit 0 is set to one. Standard error recovery procedures are used when bit 9 is reset in basic orders.

b) With all other orders a software status will be returned, as follows :

bit 0 is reset if the exchange has been correctly performed. In this case the Monitor could set one of the rightmost bits in the status depending one of the following events:

bit 15 = 1 means an EOF mark has been read
bit 14 = 1 means an EOS mark has been read
bit 13 = 1 means a character code is illegal
bit 12 = 1 means the requested length is incorrect
bit 11 = 1 means end of tape/end of media
bit 10 = 1 means beginning of tape

bit 0 is set to one and bit 1 is reset when the operation is not correctly performed and retry processing has been unsuccessful.

In this case bits 2 to 15 indicate the hardware status.

bit 0 is set to one, and bit 1 also set to one when the Monitor has detected an error in the calling sequence. In this case, bits 11 to 15 have the following meaning :

bit 15 = 1 if illegal file code has been used
14 = 1 if device is attached to another program
13 = 1 if ECB address is illegal
12 = 1 if buffer size or address is illegal
11 = 1 if function is unknown or not compatible with the device.

3.3.2. WAIT FOR AN EVENT REQUEST

The program can use the WAIT request to synchronize central processing with I/O completion. For this purpose the requesting program must reload A8 with the same ECB address as used for the associated I/O request.

Calling Sequence :

A8 = ECB ADDRESS
LKM
DATA 2

The program will be restarted as soon as bit ≠ 0 of the event byte becomes one (event occurred).

3.3.3. PAUSE REQUEST

This system function allows the program to be put in a Pause State pending an operator action.

A message must be provided by the program in order to communicate to the operator the desired action : changing a tape, mounting a new disk ...

When the operator restarts the program, he can indicate his answer (up to four hexadecimal digits) in the A7 register.

Calling Sequence :

A7 = address of the message block to be printed on the operator's typewriter

LKM
DATA 16

on return :

A7 = unchanged or containing the operator's answer.

3.3.4. EXIT REQUEST

This allows the program to indicate its termination and asks the Monitor to notify the occurrence of this event to the user.

The DOS will output the "NOW ?" message thus allowing a new control command to be typed in by the user.

Calling Sequence :

LKM
DATA 3

3.3.5. ASSIGN REQUEST

This function allows the program to define a file code and assign it to a permanent disk file or a non-disk physical device.

The file code must be in the range X'10' to X'7F'.

In the case of a multi disk configuration, the user must specify the logical number of the required disk instead of the physical device address.

3.3.6. LOAD/LINK REQUEST

A running program can request the Monitor to load a specified program. The loaded program can overlay, the requesting one depending upon whether the specified relative loading address is in the requesting program' area. Using the link option, the requesting program implicitly performs its exit and asks for execution of the loaded program. If only the load function is required the user must be careful of possible overlay, since when loading is completed control returns to the point where the request was issued.

Calling Sequences :

LOAD	LINK
A = address of load parameter block	A = address of the link block
LKM	LKM
DATA	DATA
Next instruction to be executed after loading	No return to calling program

3.4. USER CONTROL COMMANDS

The user can type control commands in order to call standard processors or perform file processing functions.

3.4.1. System Processor Calls

By typing one of the following statements on the keyboard a user can effect the loading and entering of system processors.

ASM (< source module name >) }
FORT (< source module name >) } to be processed
EDIT (< source module name >) }
LINK (< user object library name >) to be scanned during linking
UTIL (< parameters >)
DBUG (< parameters >)

3.4.2. Auxiliary Commands

3.4.2.1. Assign

Assign an I/O file code to a specified disk file.

Format : ASSIGN (<user file code>,< disk file name> [,<disk logical address>])

3.4.2.3. Include

Include a specified object module in the input file of the Linking Loader.

Format : INCLUDE (<object module name >)

3.4.2.4. Keep

Create permanent entry in directory for a specified disk file.

Format : KEEP ((UPDATE | OBJECT | <file code>) , <file name> [, < file type>]) [,< disk logical address>]

3.4.2.5. Start

Load and start the program in the Linkage Editor output file or any specified core image program.

Format : START [(< program name >)]

3.4.2.6. Access

This is a request from a user to access a file belonging to another user in the system.

Format : ACCESS (<user identification> , <file name>)

3.4.2.7. Delete

Delete a specified user file or all temporary files before end of session.

Format = DELETE [(<user file name>)]

3.4.2.8. Load

Load contents of the Linkage Editor output file.

Format : LOAD

3.4.2.9. Save

Save core image program onto a specified disk file.

Format : SAVE (<name of core image>
[,<disk logical address>])

For core image files this command performs the same function as the KEEP statement does for source, object or data files.

3.5. System Messages

To be supplied

4. Design Restrictions

4.1. Equipment Configuration

The Disk Operating System is available with a minimun configuration including :

- a P855/P860 central processing unit
- 8 k words of core
- ASR 33 teleprinter
- one disk unit moving head or fixed head

Extra devices may be added, assuming the added peripheral drivers do not cause core overflow.

4.2. Software Interface

This system being program production and testing oriented, will provide supervisor services for some system macros. It is not possible to include all kinds of system macros available in any monitor (real time functions of BRTM for instance). Consequently requesting a non existing system function will produce abortion of the requesting program, unless this system is running under control of the debugging package in which case control will be given to the user keyboard after a warning message output.

4.3. Source Language

The Disk Operating System will be written in Sagittaire Assembly Language available on P855/P860 computers.

5. Performance Requirements

5.1. Maintainability

The disk operating system will be designed in a modular way in order to allow easy modifications or addition of new components.

5.2. Compatibility

This system cannot be compatible with any other monitor because it is mainly program production oriented. (See point 4.2.). It is not intended to be a runtime Monitor.

5.3. Throughput Efficiency

Only limited by the user keyboard response time.

5.4. Test Requirements

Testing must be performed in respect with the following steps :

- Monitor requests
- Operator control statements
- Disk physical I/O
- Disk logical I/O
- User control commands
- Running system

First steps can be performed under simulator'control.