The initial loading procedure is very simple:
The bootstrap is loaded, either through the toggle switches or by pushing the IPL button on the control panel, then the Initial Program Loader (IPL) is loaded into memory, followed by the monitor.

LOADING BOOTSTRAP AND IPL

The bootstrap can be loaded in one of two ways, depending on whether the optional ROM bootstrap is included in the system or not:

If not, the procedure is as follows:

- switch on the CPU
- load the bootstrap into the first 64 memory locations manually, by means of the toggle switches and the Load Memory button on the control panel. The 64 bootstrap values can be found in Appendix E. Then check by reading these locations out.
- set up the device parameters on the toggle switches, as shown below, and load this value into the A15 register.
- put the disc containing IPL and monitor into the disc drive, push the START button on the drive and wait till the READY button lights
- push the MC button
- load 0 into the A0 register
- push the RUN button on the CPU control panel
- the IPL is now loaded into memory and it loads, in turn, the monitor, after which the monitor initialization phase is started with the typing out of the monitor identification.

If the ROM bootstrap is included in the CPU, which is highly recommended, the procedure is much simpler:

- switch on the CPU
- put the disc containing IPL and monitor into the disc drive, push the START button on the disc unit and wait for the READY button to light

- set up the device parameters on the toggle switches on the CPU, as shown below, and load this value into the A15 register
- push the IPL button on the control panel. This loads the bootstrap into memory, which immediately loads the IPL from disc. The IPL then loads the monitor and the monitor initialization phase is started with the typing out of the monitor identification.

The device parameters on the data switches must be set as follows:

0	1	2	3	4	7	8	9	10	15

where:

bit 0 = 0: character exchange on Programmed Channel
= 1: word exchange on Programmed Channel
bit 1 = 0: IPL not loaded from disc or loaded from CDC disc
= 1: IPL loaded from disc
bit 2 is used only if bit 1 = 1:
= 0: fixed head disc or flexible disc
= 1: moving head disc (X1215 or X1216)
bit 3 = 0: IPL input device connected to I/O Processor
= 1: IPL input device connected to Programmed Channel
bits 4 to 7 contain control information for the control unit:
TY = 0001 MT = 0010 FL = 0000
TK = 0111 PR = 0000 CDC = 0001 X1215/16 = 0011
bit 8 = 0: a single device control unit is involved
= 1: a multiple device control unit is involved
bit 9 = 1: X1215 or X1216 used for IPL
= 0: other device or disc type used
bits 10 to 15 contain the device address (See appendix on device addresses)

Initial Program Loader (IPL)

The disc IPL program is written onto the disc when the disc is premarked. It is written in absolute binary, so, to enable it to run anywhere in memory it does not contain any memory direct reference. When loaded, the IPL reads and loads into memory from the disc from which it has itself been loaded, starting from sector number /12 (the first two sectors of a file are reserved for the system). The first four words of sector /12 contain:

- start address of the load module
- number of sectors used by this module
  (This is the standard load format on disc; these words are generated by the disc linkage editor).

Note:
As long as it has been stored on the disc according to the IPL requirements, any stand-alone program, even if it does not use the disc, can be loaded into memory by the disc IPL.

Programs to be loaded by disc IPL
- must be in disc system load format (188 code words plus relocation per sector)
- must be catalogued on disc as a file starting at disc sector logical address /10
- must be built of consecutive granules.

After the monitor has been loaded from an input device, the system will output a question on the typewriter:

PARTITIONING?

The answer to this question must be Y or N.

If it is N, the sizes of the different memory partitions will remain as defined at system generation time.

If it is Y, the user is given the option of modifying these sizes. The following messages are output for this purpose:

DYN AREA LENGTH:

The user may type in 4 hexadecimal characters to define the length, in characters, of the Dynamic Allocation Area.

READ ONLY LENGTH:

The user may type in 4 hexadecimal characters to define the length of the Read Only Area. This must be at least /880 words.

PROGRAM SAVE AREA:

The user may type in 4 hexadecimal characters to define the length of the Program Save Area.

CORE RESIDENT AREA LENGTH:

The user may type in 4 hexadecimal characters to define the length of the Memory Resident Area.

SWAP AREA LENGTH:

The user may type in 4 hexadecimal characters to define the length of the swap area.

Any remaining memory space will be used for background programs.

Bits 10 to 15 of the device parameter word, initially set up on the control panel data switches (see page 1-64), give the address of the system disc. If this is different from the one defined at system generation, file code /F0 will be assigned to this disc.

For example, at system generation time file code F0 was assigned to disc 02, and file code F1 to disc 12. If the contents of bits 10 - 15 = 02, this will remain so; if they are 12, file code F0 will be assigned to disc 12 and F1 to disc 02.

When the monitor has been loaded control is given to the INIMON module which takes the following actions:

- Checks the unit from which the monitor was loaded and possibly exchanges file code /F0 with that of the disc from which the monitor was loaded.
- Checks the operability of the disc units declared at SYSGEN. At this point the messages

  DKER or
  DISC UNIT <dev. addr.> UNKNOWN

  may be output in which case these units are flagged to be inoperable so that only a ready interrupt can allow their use. Fixed head discs must be ready before loading the system, for they do not generate ready interrupts.
- Reinitializes the memory layout parameters from the keyboard, if Y was answered to PARTITIONING? , determines the memory size and formats the memory. At this point the message

  BUY MORE CORE

  may be output to indicate that the memory is too small for the specified parameters.
- Determines the background area size.
- Sets the PCT pointers of the read only programs on their disc addresses (via the directory), reserves the save areas for the read only programs and activates the D:USV3 system program which creates the D:CI file and generates all system read only programs in core image format.
  At this point one of the following messages may be output:

  D:CI TOO BIG

  (not enough consecutive granules can be found on disc for the D:CI file)

D:CI TOO SMALL

(not enough room in this file for the system read only programs, or Read Only Area too small for the system read only programs, or read only program missing on disc).

At the end of this initialization by INIMON the message

DRTM <release number>

is output and a branch is made to the dispatcher.
Note: When a new supervisor is implemented (with an RSU command) with a D:CI file size declared during sysgen different from the size of the old one, it is strongly recommended to delete the old D:CI file before the loading phase, because the system will use this file instead of allocating granules to the new D:CI file.

System Initialization

When the monitor has been loaded by the Initial Program Loader (IPL), the user must load all memory resident programs, through LD control commands, and declare all Read Only programs, through RO or SW control commands. Then these programs must be connected to a hardware or software level and the system is ready to run. By means of external interrupts or by an ST control command various activities may be started.

Mounting a New Disc

When a system is running, the user may mount a new disc. A 'Ready' interrupt is sent to the system when the disc becomes operational, with the following distinctions for the different disc types:

a) If the interrupt comes from the system disc, nothing happens.
b) If it comes from the fixed disc in an X1215, there are two possibilities:
   - In case the disc has already been used, i.e. at least one file code has been assigned to a catalogued or temporary file on this unit, nothing happens.
   - In case the disc has not been used, i.e. no file code has yet been assigned to it, the system will read the volume label of the disc, print it on the typewriter and initialize the disc allocation table.

c) If the interrupt comes from a moving head disc or from the removable disc of the X1215, all file codes assigned to a disc on it are deleted, the volume label is typed out and the disc reallocation table reinitialized.

d) In case the disc volume label must be printed and there is no memory space available for the disc initialization program, the label will not be printed. The disc must then be mounted again.

SYSTEM MESSAGES

The following messages may be output by the monitor:

- during the initialization phase (see 1-66):

  PARTITIONING?
  DYN AREA LENGTH:
  READ ONLY LENGTH:
  PROGRAM SAVE AREA:
  CORE RESIDENT AREA LENGTH:
  DKER, if a disc cannot be initialized (see below).
  DISC <address> UNKNOWN, if the disc specified by <address> is unknown by the CPU (wiring-error or non-existant).

  When the system is ready to be started, the following message will br printed on the typewriter:
  * * DRTM * * X X * *
  Where X X indicates the release number.
  The operator may now press the control panel interrupt button and start the system.

- while the system is running:

  ER if an operator message is not recognized by the system, or cannot be executed.

  PU␣<device name> <device address>, < status> ,RY
  An error or abnormal condition has occurred on the device with address specified. The hardware status is printed. If RY is typed as well, the operator may attempt to correct the error and retry the last I/O operation by typing the operator message RY.

  DKER␣<disc address>␣<sector number>␣<status>
  This message indicates that an irrecoverable error has occurred on a disc:
  <disc address> is the physical address of the disc unit on which the error has occurred.
  <sector number> is the address of the sector on which the current operation should be performed. It is specified as a 16-bit value:
  - for fixed head disc, it gives the absolute sector address;
  - for the X1215 discs:
    bits 1 to 10 give the cylinder number
    bit 11 the head number
    bits 12 to 15 the sector number where the error occurred.

DSK INIT ERR

When a disc unit is switched on and the monitor cannot read its volume label, it will print this message on the typewriter to tell the operator that this disc cannot be used. He will have to mount another disc or try and fix the current one.

Fatal Errors

There will be no message output in case of a fatal error, but the system will execute a loop instruction (RB *) and put a code in the A1 register to indicate the cause of the fatal error:

0: unknown interrupt
1: power failure (if no routine is available)

3: stack overflow
4: non-wired instruction
5: scheduled label for a level equal to or below 48
6: overflow on T:EVT tables (event count; see Part 2)
7: overflow on the scheduled label table
8: overflow on the dynamic allocation area
9: request to M:DML (dynamic allocation) to release an unknown block
A: exit from a level equal to or below 48
B: overflow on the program save area.

Note: The address of the RB * instruction can be found in the SYSLINK MAP output (see System Generation): it is the address of location D:ERDG+2, inside the module named HALT.

SYSTEM COMMAND LANGUAGE (SCL)

The DRTM has not been designed to handle batch processing. Therefore, the System Command Language must not be considered as a job control language, but more as a system initialization program. It allows the operator to load or declare various programs written independently from the monitor.

The System Command Language module is activated when the operator types in CC after having pressed the INT button on the control panel. It consists of read only programs usually connected to level 61 and can therefore interrupt the execution of lower priority user programs. This is not much of a problem, because:

- it performs many I/O operations, often on typewriter (a slow device) on a conversational basis, so that it does not consume a great deal of processor time;
- it is used mainly to initialize the system, not to supervise processing.

Each command consists of a mnemonic of two characters, followed by a space and parameters, if any.
Parameters are positional and separated by commas.
The syntax description for the commands where DAD or disc specification is required, is as follows:

<das> ::= <dfc>|<din>⌈:<dadname>⌉
<dfc> ::= <DAD file code>(user assigned)
<din> ::= <disc file code>(/F0 - /FF)

DADname may be omitted, in which case the first DAD on the disc will be taken.

When input via the typewriter, a command must be terminated by CR LF. System commands are always read from a device with system file code /E0, which is normally assigned to the typewriter. It can easily be assigned to another device.
Numerical characters used in the command parameters can be specified in hexadecimal format (preceded by a slash) or simply in decimal.

When an error occurs, the command cannot be processed and an error code will be output on the typewriter through file code EF:

ER 00: command unknown
ER 01: syntax error
ER 02: disc not operational
ER 03: file code unknown or: not compatible with the requested operation
ER 11: I/O error
ER 15: parameter error

Other error codes are specific to each command and are therefore described with the commands in the following paragraphs.

TABLE OF SCL CONTROL COMMANDS

Command	Meaning	Page
AS	Assign a file code	1-96
BF	Space file backwards	1-99
BR	Space record backwards	1-99
CN	Connect a program to a software level	1-91
CT	Connect a program to a timer	1-92
DF	Delete a file	1-98
DL	Delete a file code	1-97
DN	Disconnect a program from a level	1-93
DT	Disconnect a program from a timer	1-93
EN	End of commands	1-99
FF	Space file forward	1-98
FR	Space record forward	1-98
KF	Keep file	1-95
LD	Load a memory resident program	1-88
Magnetic Tape Control Commands		1-98
RO	Declare a read only program	1-89
RW	Rewind tape	1-98
SC	Set clock	1-94
SD	Set date	1-94
ST	Start a program	1-94
SW	Declare a swappable program	1-90
TS	Define time slice	1-91
UN	Unload tape	1-98
WF	Write End-Of-File mark	1-99
WS	Write End-Of-Segment mark	1-99
WV	Write End-Of-Volume mark	1-99

LOAD A MEMORY RESIDENT PROGRAM

Syntax

LD␣<name>,<das>[,<level>][.SL,<number>]

Use

This command is used to load a program into memory from disc.
<name> is the name of the program which must be loaded. It consists of a character string of up to 6 characters.
<das> : see page 1-85
<level> must be specified if the program is an interrupt routine. If <level> is not specified, the program to be loaded is not an interrupt routine. In this case a 16-word save area will be reserved in front of the program.
SL, <number> indicates that the program uses scheduled labels; <number> is the maximum number of labels to be scheduled at the same time. A scheduled label save area is also reserved in front of the program.

Note: The granules of a load module need not be consecutive on the disc. This command must be used at initialization time, before starting the memory resident program.

Error Codes

One of the following error messages may be output for this command:
ER 04: no Program Control Table available
ER 06: memory resident area overflow
ER 07: level error
ER 08: level already connected
ER 09: program unknown
ER 10: too many scheduled labels
ER 12: program already declared
ER14 : DADname unknown
ER 21: DAD not found
LDER <program name>: I/O error during loading operation.

DECLARE A READ ONLY PROGRAM

Syntax

RO␣<name>,<das>⌈,SL,<number>⌉

Use

Before starting any program activity, the user must declare all the programs which are used in the system at execution time, by means of this command. The monitor is then able to build the Program Control Table used by this program.

<name>: is the name of the read only program (up to 6 ASCII characters).

<das> : see page 1-85.

SL,<number> indicates that the program uses scheduled labels and specifies the number of scheduled labels for which a save area must be reserved.

Note: All read only programs must be declared before use, otherwise they will be considered as background programs.
The system will make a core image copy of the program and store it in the D:CI file. This file does not have to be catalogued by the user, for it will be created implicitly by the system when the monitor is loaded for the first time.

Error Codes

One of the following error messages may be output for this command:

ER 04: no Program Control Table available
ER 05: read only save area overflow
ER 09: program unknown
ER 10: too many scheduled labels
ER 12: program already declared previously
ER 13: program too long (overflow into read only save area).
ER 14: DADname unknown
ER 21: DAD not found
ER 26: D:CI file overflow.
ER 27: A read-only program cannot be segmented.

DECLARE A SWAPPABLE PROGRAM

Syntax

SW␣<name>,<das>[<time slice>|S],[I|E][,SL,<number>]

Use

The user must declare all swappable programs before they are activated, otherwise they will be considered as background programs. By means of this command a program is declared as swappable, causing the monitor to build an entry for it in the Program Control Table, allocate its save area and produce the core image of the swappable program in the D:CI file.

<name> is the name of the program.

<das> : see page 1-85.

SL,<number> defines the number of scheduled labels for which space must be reserved in this program's save area.

<time slice> is the value of the time slice which must be used for this program. It must be a multiple of 100 milliseconds.

If S is specified instead, the default value for the time slice, as defined at System Generation time, must be used.

If I is specified, the program can be swapped immediately when its time slice has elapsed, regardless of the number of I/O operations taking place on non-disc devices.

If E is specified, the program will be swapped only after all current I/O operations have been terminated.

If I is specified, the buffer and the ECB required for the I/O operations for non-disc devices must be outside the swap area, i.e. in the memory-resident area or in the dynamic buffer area. When E is specified, there is no such restriction.

Error Codes

ER01: syntax error

ER04: no free Program Control Table available

ER05: Program Save Area overflow

ER11: I/O error

ER12: the program has already been declared

ER15: parameter error

ER 14: DADname unknown

ER 21: DAD not found

ER26: overflow on the D:CI file.

ER 27: a swappable program cannot be segmented

DEFINE TIME SLICE

Syntax

TS␣<number>

Use

This command is used to define the time slice used for the swappable program, if it has not already been specified in the SW command. If the time slice has already been defined at System Generation time, it may be redefined by means of this command.
<number> is a value, specifying the length of the time slice in tenths of seconds, with a maximum of 256.

CONNECT A PROGRAM TO A SOFTWARE LEVEL

Syntax

CN␣<name>,<level>

Use

By means of this command a program, which has already been declared to the monitor as either a memory resident or read only program, can be connected to a software level. Interrupt routines cannot be connected to a priority level with this command, as no Program Control Table is allocated for them.
It is also possible to connect a program to a software level later on, by means of a monitor request.
<name> is the name of the program (up to 6 ASCII characters).
<level> is the software level to which the program must be connected.

Note: Background programs need not be connected explicitly when they are activated, because they don't have to be declared to the system at initialization time.
However, at system generation time, the user must reserve enough memory space to handle the Program Control Tables of all background programs used. These tables are allocated and released dynamically for the background programs.

Error Codes

ER07: <level> error

ER08: program already connected

ER09: program unknown.

CONNECT A PROGRAM TO THE CLOCK OR A TIMER

Syntax

CT␣<name>,<NTIM>,<PR>,[<NC>|<HH>,<MM>,<SS>]

Use

When a program has been connected to a software level, it may, through this command, also be connected to the real time clock or to a timer. This may also be done by monitor request.
For timer numbering, see part 2, page 2-51.
<name> is the name of the program which must be connected (up to 6 ASCII characters).
<NTIM> is the timer number. If the program must be connected to the real time clock, 0 must be specified. For timer numbering, see under monitor request LKM10.
<PR> gives the pulse rate, a number from 0 to 127. When 0 is specified, only one program activation is made, after which it is automatically disconnected from the clock or timer.
<NC> indicates the number of timer cycles before the first program activation (a hexadecimal value from 0 to /7FFF).
<HH>,<MM>,<SS> gives the time in hours, minutes and seconds when the program must be activated.

Error Codes

ER16: the specified timer has not been defined.

ER14: the program has not been connected to a level.

DISCONNECT A PROGRAM FROM A LEVEL

Syntax

DN␣<name>,<level>

Use

This message is used to disconnect a program from a software level.

Error Code

ER01: syntax error

ER07: level error.

ER20: disconnection impossible: program busy

DISCONNECT A PROGRAM FROM A TIMER

Syntax

DT␣<name>,<NTIM>

Use

This command is used to disconnect the program specified with its <name> from the timer specified to which it has been connected.

Error Code

ER16: the program had not been connected to a timer or wrong timer number.

SET DATE

Syntax

SD␣<DD>,<MM>,<YY>

Use

This command is used to set the current date in the machine. <DD>,<MM>,<YY> denote day, month and year, 2 characters each. They will be stored in memory in ASCII format and returned to user programs issuing the monitor request Get Time.
The system does not check the validity of the date.

SET CLOCK

Syntax

SC␣[<HH>[,<MM>[,<SS>]]]

Use

This message is used to initialize the system clock. The current time is entered in a specific internal system table and the real time clock is started.
<HH>,<MM>,<SS> specify the time in hours, minutes and seconds, two characters for each. Default value is 0.

START A PROGRAM

Syntax

ST␣<name>[,<das>]

Use

This command can be used for starting memory resident, read only, swappable or background programs.

<das> : see page 1-85.

When <name> identifies a memory resident, swappable or read only program declared previously, the program must have been connected to a software level.

Note: A segmented program may run as a core resident or as a background program.

Error Code

ER 04: PCT pool overflow
ER 05: Save Area pool overflow
ER 14: program has not been connected
ER 18: program does not exist on disc
ER 19: overflow on dynamic area (no possibility to build an activate block to activate background)
ER 20: program too large for background area
ER 21: DAD not found
ER 51: disc I/O error.

KEEP FILE

Syntax

KF␣<file code>,< file name>

Use

This command is used to make a temporary file permanent by storing it in the library of the disc on which it is located.
<file code> identifies the file which must be catalogued.
<file name> is a string of up to 6 characters to identify the file which must be catalogued. The type of the file is implicitly UF (user data file). If <file name> already exists in the directory, this one replaces the old one, but the granules are not recovered.

Error Codes

ER22: non-disc file
ER23: file has already been catalogued
ER24: no entry available in the library directory.

ASSIGN A FILE CODE

Syntax

AS␣ <fc1>,[<fc2>|<dnda>| DK<das>,<n>|DK<das>,<file name>
| EP<das>,<file name>|<das >]

Use

This message is used to assign or re-assign a file code either to a physical device or to a disc file.

<file code 1>: file code which must be assigned.

<file code 2>: file code to which <file code 1> must be assigned.

DN[DA] is the device name (2 characters) and device address (2 hexadecimal digits) of the device to which <file code 1> must be assigned.

<fc1> is assigned to:

- another file: <fc2>
- a peripheral device: <dnda>
- a temporary file on disc: DK<das>,<n>
- a permanent file on disc: DK<das>,<file name>
- an extended file on a disc: EP<das>,<file name>
- a disc or DAD: <das>

When <n> is specified, the system will allocate a number of consecutive granules to the file. Such a file is temporary and must be used with random access.

<das> : see page 1-85.

Note: the system file codes must all be assigned at sysgen time, because when an AS command is given the previous assign ment is deleted before the new one is assigned. If an error occurs at that point, a system deadlock may occur.

Note: When this command is used for re-assigning a file code, the monitor will delete the old file codes, before assigning the new one. Therefore, if an error occurs, the file code remains unassigned. For some system file codes this may cause system deadlock, so the user must not re-assign system file codes, but declare them all at system generation time.

Also, as it is useful to change the assignment of /EO, SCL assigns this file code to /EF when it finds it unassigned.
The following device names are supported by the system:

MT: magnectic tape
TK: cassette tape
CR: card reader
PR: punched tape reader
PP: tape punch
TY: operator's typewriter
TR: ASR punched tape reader
TP: ASR tape punch
LP: line printer
NO: no device.

Error Codes

ER 14: DADname unknown
ER51: I/O error on disc
ER52: no spare entry available in file code table
ER53: no disc file description table free
ER54: device unknown or disc file code unknown
ER55: disc overflow or too many granules requested
ER56: file unknown
ER57: <file code 2> unknown
ER58: more than 7 file codes assigned to the same disc file.

DELETE A FILE CODE

Syntax

DL␣<file code>

Use

This message is used to delete a previously assigned file code.
<file code> is the file code which must be deleted.
If it was a file code assigned to a temporary disc area, all granules allocated to this file are released for use by other files.

Note: In case of disc files, the file description table is released. If the file had not been closed, however, the current contents of the buffer are not written on the disc, but they are lost and the space occupied by the blocking buffer is not released.

DELETE A FILE

Syntax

DF␣ <das>,<file name>

Use

This command is used to delete a catalogued file on a disc. <file name> is the name (up to 6 characters) of the file which must be deleted.

<das> : see page 1-85.

The granules occupied by this file are released, but they are not used before the disc allocation table has been loaded again, i.e. at system reloading time for the System Disc and when the disc unit is restarted for user discs. This is done because there may be several file codes assigned to the deleted file.

Error Code

ER21: unknown <file name>

MAGNETIC AND CASSETTE TAPE CONTROL FUNCTIONS

A number of commands can be used only for magnetic or cassette tape, to perform a number of control functions:

- RW␣<file code> is used to rewind the tape of which the <file code> is specified.
- UN␣<file code> is used to unload the tape specified.
- FF␣<file code>[,<number>|ALL] is used to space the specified tape forward over a <number> of files or to skip all

files (ALL), in which case the tape will stop as soon as two consecutive tape marks are encountered.

- BF␣<file code>[,<number>] is used to space the specified tape backwards one file or, if specified, a <number> of files. The file is positioned after the tape mark.
- FR␣<file code>[,<number>] is used to skip forward in the defined file over 1 or, if specified, a <number> of records.
- BR␣<file code>[,<number>] is used to skip backwards in the defined file over 1 or, if specified, a <number> of records.
- WF␣<file code> is used to write an end-of-file mark after the specified file.
- WS␣<file code> is used to write an end-of-segment mark.
- WV␣<file code> is used to write an end-of-volume mark.

Error Codes

ER01: syntax error

ER11: I/O error.

END OF COMMANDS

Syntax

EN

Use

This command is used to indicate the end of a series of system commands. The SCL package will perform an exit and thus release the dynamic area.

10 Operator Control Messages

By means of the operator commands the user may have limited control over the running system. This is initiated by pressing the interrupt button (INT) on the control panel, to start the operator control package. As soon as this module becomes the highest priority active program in the system, it will output the message

M:

on the operator's typewriter to request input from the operator.
Any of the commands described below may then by typed in.
All commands are terminated by LF CR.
The characters ↑ and ← may be used to correct the typed-in message, if necessary.

Message	Meaning	Page
CC	Request SCL	1-102
CR	Correction	1-102
DD	Dump Disc	1-102
DM	Dump Memory	1-102
HD	Halt Dump	1-103
HT	Stop CPU	1-103
RD	Release Device	1-103
RY	Retry I/O Operation	1-103
WM	Write Memory	1-104

Request SCL:

CC

This command is given to activate the SCL (System Command Language) program, so that commands may be given to initialize the system or give control to the running programs. Upon this command, SCL will type out C: and wait for the operator to type in an SCL command.

Correction:

CR␣<file code>
By means of this command correction records can be entered.
<file code> is the file code of the device from which the corrections are input.
The correction records must have the following format (up to 72 characters each):
<address>,<value 1>⌈,<value 2>.....,<value n>⌉
where <address> is the address of the first (or only) location to be modified and <value 1> specify the values which must be stored in <address> and the locations following it.

Dump Disc:

DD␣<das>,<sector1>⌈,<sector2>⌉
As a result of this command, a hexadecimal dump will be made of the contents of the disc specified by <das> (see page 1-85).

The dump will be made of <sector 1> or, if specified, from <sector 1> to <sector 2> inclusive. The dump will be output on the device with file code 02.

Dump Memory:

DM␣<address 1>⌈,<address 2>⌉
A hexadecimal memory dump will be made of <address 1> or of the memory locations from <address 1> to<address 2> inclusive, if specified. The dump will be output on the device with file code 02.

Halt Dump:

HD
This command is used to stop a current dump operation, which was initiated by a DD or DM command.

Stop CPU

HT
This command is used to stop all CPU activity. However, all current physical I/O operations are terminated (without checking). Then the CPU will enter the idle loop:

```
INH
RB   *-2
```

after which the operator can switch off the system.
This command is not normally used, but in some situations it might be useful: if, for example, a program continues writing onto a magnetic device for some reason, and the operator wants to stop the machine, he may do so with the HT command and still allow the current write operation to be terminated.

Retry/Release and I/O Operation:

[RY|RD]␣<device address>
When an I/O error occurs which requires operator intervention, the system prints out the message:

PU␣<device name><device addres>,<status>, RY

where RY denotes that the operator may retry the I/O operation. If the error can be corrected, the operator must give the RY command to allow the program to continue. Otherwise he must type in the command RD to release the I/O operation from the device specified. In both cases, the system will give a CIO Start Instruction to retry the last erroneous operation.

Write Memory:

WM␣<address>,<value 1>[,<value 2>,....,<value n>]

This command is used to modify one or more memory locations.
<address> is the first location to be modified (an even address, of up to 4 hexadecimal characters).
<value 1> to <value n> are the values in hexadecimal to be entered in the memory locations, starting from <address>, i.e.

<value 1> is stored in location <address>
<value 2> is stored in location <address>+2, etc.

The user program can request the monitor to perform certain functions. A request takes the form of an LKM (Link to Monitor) instruction followed by a DATA directive.
The directive has a number as operand, which specifies the function to be executed. If this number is negative, the user is scheduling a label on completion of the request.
Preceding a request, certain parameters may need to be loaded into the A7 and A8 registers.
After the monitor has processed the request it loads a return code in the A7 register. If the requested service module is not available, A7 will always contain the value -1.

Note: It is possible to use scheduled labels in conjunction with a monitor request. See chapter 4.

Under the DRTM, some of the modules handling monitor requests are core resident, to provide fast response. Others are stored on disc just as other user-written read only programs. Still, these user programs can use these monitor requests, because any work areas or parameters are built in the dynamic allocation area, to provide the possibility of communication.

However, the read only monitor requests cannot be used by programs connected to a priority level which is equal to or higher than that of the read only monitor request, i.e. if the level of the request handler (read only) is 49, only those user programs from level 50 to 62 can use it.
Note: No monitor requests except Activate and Set Event may be issued by a program with priority level smaller than 49.

Request	LXM	Page
Input/Output	1	1-107
Wait for an Event	2	1-112
Exit	3	1-114
Get Buffer	4	1-115
Release Buffer	5	1-117
Connect Program to Timer	10	1-118
Disconnect Program from Timer	11	1-120
Activate a Program	12	1-121
Switch Inside a Software Level	13	1-123
Attach Device to Program	14	1-124
Detach Device from Program	15	1-126
Get Time	17	1-127
Set an Event	18	1-129
Connect Program to Software Level	20	1-130
Disconnect Program from Level	21	1-131
Wait for a Given Time	22	1-132
Assign a File Code	23	1-134
Delete a File Code	24	1-137
Read Unsolicited Operator Message	25	1-138
Cancel Unsolicited Message	26	1-140

I/O REQUEST

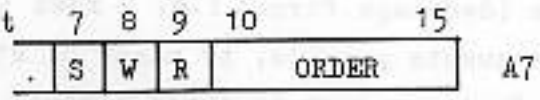
Calling Sequence

```
LDK     A7,CODE
LDKL    A8,ECBADR
LKM
DATA 1
```

Use

The user can ask the system to start a particular I/O operation on a peripheral device.
Processed at level 48 for physical I/O requests, at level 49 for logical I/O requests (Data Management).

Register A7 is loaded with a CODE specifying the details of the I/O function, as follows:



S, W and R specify the mode of operation:

S = 1: (W must also be 1): used by a swappable program; this program can then be swapped out immediately when its swap event count value becomes zero.

W = 1: the requesting program wants to wait for the completion of the requested I/O operation. Only after completion of the requested function, will the return to the calling program take place.

W = 0: a return to the calling program will be made as soon as the transfer has been initiated. The program will give a Wait request later on for synchronization.

R = 1: the program itself will process any abnormal condition concerning the requested operation (possible only with Basic Read/Write). The system will return the hardware status in ECB word 4. No retry is possible.

R = 0: any abnormal conditions will be processed by the system. The software status is returned in ECB word 4.

ORDER specifies which I/O function is required, by giving one of the following hexadecimal values:

01: Basic Read (These orders are used by Disc File Management and can be used
05: Basic Write by user programs only if DFM has not been selected at sysgen.
For Basic I/O requests the system does not provide for character checking or data conversion, only for control command initialization and end of operation signals.

02: Standard Read
This order can be used to input standard object code records in 4x4 or 8+8 format, as well as ASCII character strings.

06: Standard Write
Standard (ASCII) I/O requests provide, by means of standard conversions, for special features such as error control characters, conversion from external code to internal ASCII and vice versa.

07: Object Write (4+4+4+4 tape format i.e. 4 rows in 4 columns)

08: Object Write (8+8 tape format i.e. 2 rows in 8 columns)
Object I/O requests provide, by means of standard conversions, for special features such as error control characters, checksum and data conversion from external 4+4+4+4 or 8+8 tape format to internal 16-bit format.

0A: Random Read

0B: Random Write

11: Read Sector (flexible disc; DAD)

14: Skip forward to EOS mark

15: Write Sector (flexible disc; DAD)

16: Skip forward to EOF mark

22: Write EOF mark (tape and disc sequential files)

24: Write EOV mark (tape and disc sequential files)

26: Write EOS mark

2D: Door Unlock (flexible disc)

2E: Door Lock (flexible disc)

2F: Write Deleted Data Address Mark (flexible disc)

30: Get Information about a File Code

31: Rewind File (magnetic and cassette tape)

33: Backspace one block (magnetic and cassette tape)

34: Space one block forward (not allowed for cassette)

36: Skip backward to EOF mark (magnetic and cassette tape)

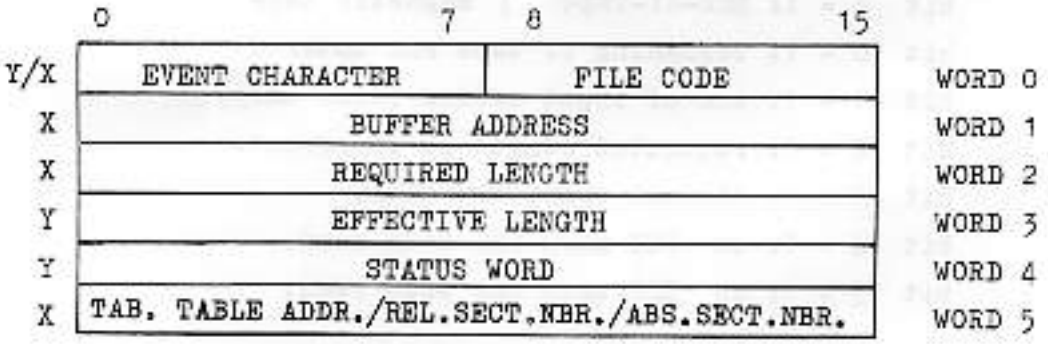
38: Unlock (cassette and magnetic tape)

37: Lock (cassette tape)/Off line (magnetic tape)
3A: Compound Read (flexible disc)
3B: Compound Write (flexible disc)
3C: Search Key with Mask (flexible disc)
3D: Write Deleted Data Address Mark and Verify (flexible disc)
3E: Search Key (flexible disc)
3F: Write Sector and Verify (flexible disc)

(Physical disc access on sector level is possible with orders /11 and /15, only for flexible disc and DAD. The Disc File Management module is not used in these cases. ECB5 must contain an absolute sector number.)

For each of these request orders, specific information applies to the various peripheral devices. This information is given in Appendix C at the end of the book.

The Event Control Block, of which the address must have been loaded into the A8 register, has the following format:



X: these words must be filled by the user
Y: these words are filled by the monitor

where:

WORD 0: event character:
bit 0 = 1: end of operation has occurred for the ECB.
The other bits remain unused.

WORD 1: address of the user buffer

WORD 2: requested length to be read or written, in words (basic read on card reader) or characters (other devices). The first character is always the character given by the buffer address. For standard write on typewriter or line printer, two characters must be added, at the beginning of the buffer.

WORD 3: effective length which has been transmitted, in words (basic read on card reader) or characters (others). Stored here by the monitor upon completion of the I/O operation.

WORD 4: status word, stored here by the monitor upon completion of the requested I/O operation.

- For Basic orders, this word will be filled with the hardware status by the control unit. However, if the monitor detects an error in the calling sequence, bit 0 will be set to 1 and the other bits will contain the software status. (Hardware status: Appendix D).
- For the other orders, the software status will be returned:

  bit 0 = 0: the operation has been successfully completed:
  bit 7 = 1: no data (tape cassette)
  bit 8 = 1: End-Of-Volume } cassette or
  bit 9 = 1: End-Of-Tape } magnetic tape
  bit 10 = 1: beginning of tape encountered.
  bit 11 = 1: end of input medium (disc only).
  bit 12 = 1: requested length is incorrect.
  bit 13 = 1: illegal character code.
  bit 14 = 1: an EOS mark has been read.
  bit 15 = 1: an EOF mark has been read.

  When the operation was not successfully completed, bit 0 is set to 1 and bit 1 is set to 0 (retry also was not possible). In this case bits 2 to 15 give the hardware status.
  When the monitor has detected an error in the calling sequence, bits 0 and 1 are both set to 1 and bits 2 to 15 have the following significance:
  bit 2 = 1: power failure
  bit 5 = 1: disc overflow (no more granules available)
  bit 6 = 1: no disc buffer available (dynamic area overflow)

bit 7 = 1: disc queue overflow (queue area reserved at system is too small)

bit 10 = 1: file is write-protected.

bit 11 = 1: function is unknown or not compatible with the device or bit 9 in A7 set for a DFM request

bit 12 = 1: illegal buffer size or address

bit 13 = 1: illegal ECB address

bit 14 = 1: device is attached to another program.

bit 15 = 1: an illegal file code has been used.

WORD 5: this word is used by the user to store:

- the relative sector number to be exchanged in the case of a random disc file (see Data Management).
- the absolute sector number in the case of physical disc access (orders 11 and 15, and orders for flexible disc through the user driver for that device).
- the tabulation table address in the case of a standard read operation on ASR or punched tape equipment. This tabulation table has the following format.

Number of Tackets	First Tacket
Second Tacket	Third Tacket

The tackets indicate an absolute position in the print line. Characters up to the following tacket are filled with blanks.

Example:

3	10
20	30

Input line: LABEL \OPER \OPERAND \COMMENT

Line in buffer:

LABEL␣␣␣␣OPER␣␣␣␣␣␣␣OPERAND␣␣␣␣COMMENT

1 10 20 30

At completion of the input, the buffer is filled with spaces, but the returned length is the length effectively entered and stored, including the spaces replacing the tabulation codes (\).

Note: If word 5 is used for tabulation, the required length in word 2 must contain both the characters and the blanks in the tackets, i.e. for the above example word 2 must be filled with 36.

WAIT FOR AN EVENT

Calling Sequence

```
LDKL  A8,ECBADR
LKM
DATA  2
```

where:

ECBADR gives the address of the Event Control Block (see I/O requests). The first character of the ECB is the event character. If the first bit of this character is set to 1, the event has been completed.

Use

This request causes a program to stop and wait for the completion of an event which has to take place in another program (user or system). If the event has occurred, the dispatcher returns control to the requesting program. If the event has not occurred, the program is put in wait state, to be restarted when the event has occurred.

Note:

It is recommended not to use a Wait request inside a scheduled label routine, as this causes the whole program to be blocked temporarily.

There are two kinds of events:

- Wait for the exit of a program:
  When a user program activates another program (see Activate), the first word pointed to by the A8 register is the address of the ECB which must be used to wait for the exit of the activated program. As the activated program and the calling program run concurrently, this provides a means of communication between the two programs.

- Wait for an I/O operation.
  The program waits for the end of an I/O operation it has requested.

Notes:

- In case of explicit or implicit wait (implicit wait inside I/O, Attach Device, etc.) processed by a Read Only program, the Read Only Area will be forbidden during the whole waiting time for all Read Only programs with equal or lower priority, so explicit wait or conditions resulting in implicit wait should be avoided in Read Only programs.
- When this request is used by a swappable program and bit 15 of A8 is set to 1 while the event has not been set (LKM 18), the calling program will be suspended immediately and it may then be swapped out as soon as its swap event count becomes zero.
- The user can create his own set of events. In this case he must reset the event bit in the ECB and inform the system by giving a Set Event monitor request.
- This request is processed at level 48 by a memory resident program.

EXIT

Calling Sequence

LKM
DATA 3

Use

This request is used to specify the end of a program. The program exit is effected after completion of all I/O operations and after all labels, if any, have been scheduled. A scheduled label exit passes control to the next scheduled label, if one is present, otherwise control passes to the main program.

The program becomes inactive, unless there is another activation request waiting for this program.
The program remains connected to its level.

The ECB supplied at the activation of the program must be updated by giving a 'Set Event' monitor request (LKM 18), to set bit 0 of the event word to 1.
This request is processed at level 48 by a memory resident program.

Note: For swappable programs, the contents of the Swap Area will not be swapped out.

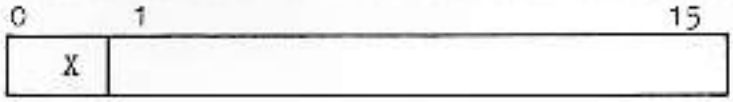
GET BUFFER REQUEST

Calling Sequence

```
LDK    A7, LENGTH
LKM
DATA 4
```

where:

LENGTH is the length, in characters, to be allocated to the buffer area (maximum 32k characters):



If bit 0 = 0: return to user in case of overflow.

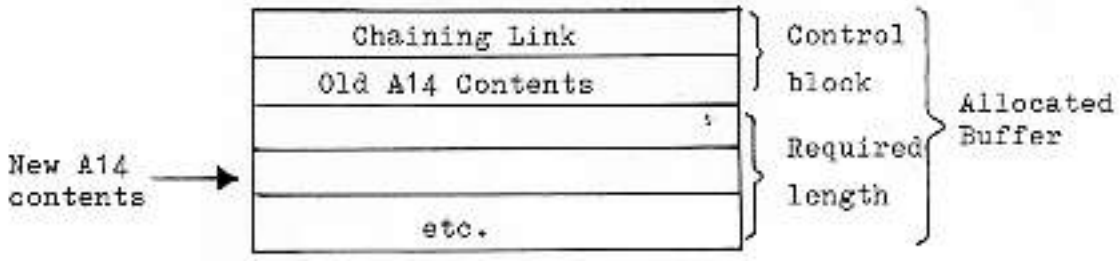
If bit 0 = 1: implicit wait of the calling program in case of overflow. This should be avoided in read only programs.

If 0 is loaded into A7, the monitor will return the highest address of memory in A7, except if the memory size is 32k, in which case 0 will be returned in A7.

Use

By means of this request, the user can allocate a memory area for temporary use, in the dynamic memory allocation area.

When the allocation is made, a control block is created by the system at the beginning of the allocated area. This block will contain a chaining link and the old contents of the A14 register:



The user must not destroy this control block

Upon completion of the request, the system responds as follows:

A7 = 0: the buffer is allocated

= 1: there is no memory space available (bit 0 in LENGTH = 0)

A14: contains the address of the fourth word of the allocated buffer, so that, as soon as the buffer is allocated, the user may give a Call Function instruction with the A14 register without having to update the A14 register first.

However, the user must then provide for stack handling.

This request is processed at level 48 by a memory resident program.

RELEASE BUFFER REQUEST

Calling Sequence

```
LDKL   A14, BUFADR
LKM
DATA 5
```

where:

BUFADR points to the second word in the buffer as given in register A14 after the Get Buffer request.

Use

To release the memory space previously reserved by a Get Buffer request. The A14 register is reloaded with the value it contained before the Get Buffer request was made.

The system responds as follows:

A7 = 0: the memory space is released.
= -1: the memory space has already been released.

If the A14 pointer is incorrect, or if the buffer area has been destroyed, the system issues a Halt.

If the dynamic area was in overflow state, this request frees it again, and programs waiting because of buffer overflow are restarted, and their requests reinitialized.

This request is processed at level 48 by a memory resident program.

CONNECT A PROGRAM TO A TIMER

Calling Sequence

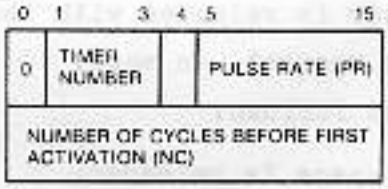
```
LDKL   A8, PARAM
LDKL   A7, PRNAME
LKM
DATA 10
```

where:

PARAM points to a two-word block, containing the necessary parameters.

Two formats are possible for this block:

- format 1 (bit 0 = 0):

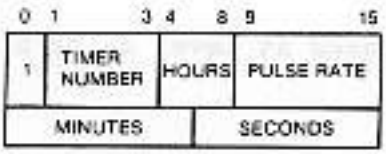


NTIM: the number of the timer (0=real time clock)

PR: requested pulse rate, i.e. the number of cycles of the timer between two activations: a number from 0 to 2047. If 0 is specified, the program is activated only once and then automatically disconnected.

NC: number of cycles of the timer before the first activation: a number from 0 to 32767.

- format 2 (bit 0 = 1):



where NC is replaced by an absolute time: HH (hour) - MM (minute) - SS (second), and PR is number from 0 to 127.

Use

A program running at a software level can be connected to a timer, according to the parameters given in register A6. The program must have been connected to a software level, otherwise it cannot be started by the dispatcher.

For timer numbering, see part 2, page 2-60.

The system responds as follows:

A7 = 0: the connection has been made.

A7 ≠ 0: connection is impossible, because:

- the specified timer has not been defined, or the program does not exist: A7 = -3;
- the program has already been connected to a timer: A7 = -2;
- dynamic area overflow: A7 = -4.

Note:

The program must have been declared by the system as memory resident, swappable or read only. Background programs cannot be connected to a timer.

- the program processing this request may be memory resident or read only at level 49. If it is read only, it cannot be used by user read only programs connected to level 49.

DISCONNECT A PROGRAM FROM A TIMER

Calling Sequence

```
LDK   A8,NTIM
LDKL  A7,PRNAME
LKM
DATA  11
```

where:

NTIM is a constant specifying the timer number.
PRNAME points to a 3-word block containing the program name.

Use

A program can, by means of this request, be disconnected from the timer specified in register A8.
The program remains connected to its software level.
The system responds as follows:

A7 = 0: the program is disconnected from the timer
A7 = -3: it is impossible to disconnect the program, because the program does not exist, or was not connected to this timer.

The program processing this request may be memory resident or read only, at level 49. If it is read only, it cannot be used by user read only programs connected to level 49.

ACTIVATE A PROGRAM

Calling Sequence

```
LDKL    A7, PRNAME
LDKL    A8, BLOCK
LKM
DATA 12
```

where:

PRNAME points to a 3-word block containing the name of the program to be activated.

BLOCK points to a 2-word block of the following format:

Event Character	ECB
Parameter Block Address	

The first word of this block may be updated by means of a 'Set Event' monitor request (LKM18) at the exit of the activated program, so that, if the calling program has requested a wait for the activated program, this word may be considered its Event Control Block (see Wait Request). The second word contains the address of a parameter block. The calling program may give a Get Buffer Request to build this block in the Buffer pool, before the Activate request is given. Register A4 will then contain the address of the parameter block when the activated program is started by the dispatcher and register A14 will contain zero. That is, register A4 contains the value in ECB+2 and A6 points to the ECB for a possible Set Event request.

Note:

When a user program is activated by an interrrupt sequence, there is no parameter block and the contents of A4 is not significant. A14 is set to zero.

Use

This request can be made by a program running at any level, to activate a software level program.
Calling program and activated program may be processed concurrently, depending on their priority levels.
If the activated program is busy, the request is recorded in a stack, to be processed later.
The activated program must previously have been connected to a level.
The system responds as follows:

A7 = 0: the request has been processed or recorded in a stack;

A7 ≠ 0: the request has not been taken into account, because:

- the program has not been connected to a level: A7 = -2;
- the program does not exist: A7 = -3.
- dynamic area overflow: A7 = -4.
- PCT pool overflow: A7 = -5.
- Save Area pool overflow: A7 = -6.
- disc I/O error: A7 = -7.
- the program is too large for the background area: A7 = -8

The program processing this request is a memory resident program at level 48 for activation of memory resident, read only or swappable programs. Requests for activation of background programs are handled by a program at level 49, which may be either memory resident or read only.

SWITCH INSIDE A SOFTWARE LEVEL

Calling Sequence

```
LDK   A7, LEVEL
LKM
DATA 13
```

where:

LEVEL is a constant specifying the number of the level in which the switch is to be made. If LEVEL is specified as 0, the level to be switched is equal to the requesting level plus one.

Use

By means of this request it is possible to have timeslicing inside a software level (only for core resident programs or background programs which are already in core), by halting execution of the program running on that level (A7) and giving control to the next program on the same level.
See also page 1-28.
This request is processed at level 48 by a memory resident program.

ATTACH A DEVICE TO A PROGRAM

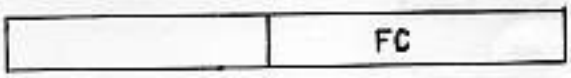
Calling Sequence

```
LDK    A7, FLAG
LDKL   A8, DEVBLK
LKM
DATA 14
```

where:

FLAG is either zero or not zero. When it is zero, control is returned to the calling program with -3 in A7, if the device was already attached to another program.
When it is not zero, the calling program is put in wait until the device is detached. This should be avoided for read only programs (see wait request).

DEVBLK points to a one-word block containing:



where:

FC is the file code assigned to the device concerned or a disc logical file code.

Use

By means of this request, a program running at any software level can obtain exclusive use of the device specified. Other programs will be unable to perform I/O operations on this device.
If the device has already been attached to another program, the requesting program is put in wait state until the device is detached by the other program (see following request).
This is checked via word 34 in the D.W.T. where bit 0 is 1 if the device is <u>not</u> attached. If it is attached, the wait is done on this bit.
If the file code has already been attached to another program and the Wait flag has been set to one, the calling program is put in Wait and the Time Slice is set to zero.

In case several programs have given an attach request and are waiting for a device to be detached, the highest level program will receive control first.

The system responds as follows:

A7 = 0: the device has been attached;

A7 = -2: the device does not exist, or it is impossible to attach it to this program.

A7 = -4: the file code specified is assigned to an extended file.

Note: - The ASR is considered as 3 devices, so 3 requests must be given to attach the ASR keyboard, tape reader and tape punch.

- When used with a disc file code, only the specified disc file is attached, the rest of the disc remains free for other programs. With DAD file code, only the DAD is attached.
- This request must not be used with file codes F0 to FF.
- User programs must not attach devices which may be used by the system, because in that case the results may be unpredictable.

The program processing this request may be memory resident or read only, at level 49. If it is read only, it cannot be used by user read only programs connected to level 49.

DETACH A DEVICE FROM A PROGRAM

Calling Sequence

```
LDKL A8, DEVBLK
LKM
DATA 15
```

where:

DEVBLK points to a one-word block of the following format:

FC

where:

FC is the file code of the device concerned.

Use

By means of this request, a device which has been attached to a program by an Attach request (LKM 14), is detached from that program.

To prevent another program having to wait un-necessarily for this device, the Detach request must be made as soon as the device is no longer required.

The system responds as follows:

A7 = 0: the device has been attached;

A7 = -2: the device does not exist, or it is attached to another program

The program processing this request may be memory resident or read only, at level 49. If it is read only, it cannot be used by user read only programs connected to level 49.

GET TIME

Calling Sequence

```
LDK  A7, FLAG
LDKL A8, TIMBLK
LKM
DATA 17
```

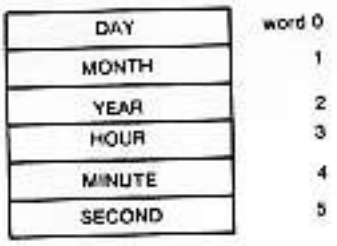
where:

FLAG is a value of either 0 or 1, to specify whether the time will be output in ASCII (0) or binary (1) format.
TIMBLK points to a six-word block into which the monitor will load the requested information.

Use

Upon receipt of this request, the monitor will return the time to the block specified in register A8, in the format indicated by the flag in register A7.

If A7 = 0, the information will be in the following format:



If A7 = 1, the information will be in the following format:

HOUR	word 0
MINUTE	1
SECOND	2
Tenth of Second	3
Fiftieth of Second	4
C:TIME	5

The values are given in binary.
C:TIME is 0, if the standard clock is included.
If a non-standard clock is included, it contains the pulse rate of this clock.

The program processing this request may be memory resident or read only, at level 49. If it is read only, it cannot be used by user read only programs connected to level 49.

SET AN EVENT

Calling Sequence

```
LDKL  A8, ECBADR
LKM
DATA  18
```

where:

The user can create his own set of events. Each time such an event has occurred, the user must inform the system about it by giving this request, so that it can restart any programs which were waiting for this event.
The system sets the event bit to 1.
For example, the system sends this request after monitor requests for I/O (twice: for ECB and for controller status), Release Buffer, Detach Device and Exit.

This request is processed at level 48 by a memory resident program.

Note: This request can be made by an interrupt routine. In that case, the interrupt routine is stopped and a branch made to level 48. After the request has been processed, a return is made to the interrupt routine, which is then restarted at its own hardware level.

CONNECT A PROGRAM TO A SOFTWARE LEVEL

Calling Sequence

```
LDK   A7, NUMBER
LDKL  A8, PRNAME
LKM
DATA  20
```

where:

NUMBER is the number of the software level to which the program is to be connected.
PRNAME points to a 3-word block containing the name of the program.

Use

A program, running at any level can make this request for another program to be connected to a software level.
The system responds as follows:

A7 = 0: the connection has been accomplished.

A7 ≠ 0: it is impossible to make the connection, because:

- the program does not exist: A7 = -3;
- the program is not compatible with the requested level. i.e. it is an interrupt level: A7 = -3;
- the program has already been connected to a level: A7 = -2.

The program processing this request may be memory resident or read only, at level 49. If it is read only, it cannot be used by user read only programs connected to level 49.

Note: This request must not be used to connect background programs (level 62) or the idle task (63). This is done automatically.

DISCONNECT A PROGRAM FROM A LEVEL

Calling Sequence

```
LDK   A7, NUMBER
LDKL  A8, PRNAME
LKM
DATA  21
```

where:

NUMBER is the number of the software level to which the program has been connected.
PRNAME points to a 3-word block containing the name of the program which is to be disconnected.

Use

The program specified is disconnected from the level given in register A7.

The system responds as follows:

A7 = 0: the program is disconnected.

A7 ≠ 0: it is impossible to disconnect the program, because:

- the program does not exist: A7 = -3;
- the program is busy: A7 = -2;

The program processing this request may be memory resident or read only, at level 49. If it is read only, it cannot be used by user read only programs connected to level 49.

Note: This request must not be used with the levels 62 and 63.

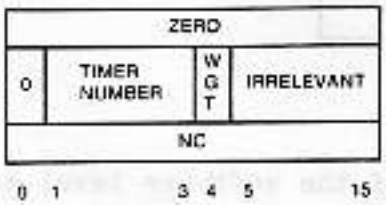
WAIT FOR A GIVEN TIME

Calling Sequence

```
LDKL   A8, PARAM
LKM
DATA 22
```

where:

PARAM is the address of a 3-word block of the following format:



where word 0 contains zero, as the ECB on which the calling program will be put in wait state, must be equal to zero.

word 1: Bit 0 = 0
Bits 1-3: timer number of the timer for which the program waits. See under LKM 10.
Bit 4: WGT flag: any value; destroyed by the system.
Bits 5-15: not significant.

word 2: Number of cycles, of the time defined in word 1, to be made before the program is restarted.

Note: When this request is used by a swappable program and bit 15 of A8 is set to 1, the calling program may be swapped out immediately, provided its swap event count is zero.

Use

This request is given to put a program in wait state, until a certain time has passed, as indicated in word 2 of the PARAM block. The precision of the waiting time depends on the precision of the timer defined in word 1 of this block.
When the number of cycles specified has passed, the requesting program is restarted with the system response in A7: 0 = request performed correctly.

If the response in A7 is -1, the wrong timer has been defined.
If it is -4, there has been dynamic area overflow.
The request is processed at level 49, by a memory resident or read only program.
If it is read only, it cannot be used by user read only programs connected to level 49.

ASSIGN A FILE CODE

Calling Sequence

LDKL A8, ASBLK
LKM
DATA 23

where:

ASBLK is a parameter block with the following layout:

- for assignment to a peripheral:

00000000	file code
device name	
device address	

- for assignment to a temporary file:

00000001		file code
D	-	disc file code
number of granules		
not used		
UF		
DAD name		

- for assignment to a permanent file:

00000010		file code
D	-	disc file code
file name		
UF		
DAD name		

- for assignment to an extended file:

00000100		file code
D		disc file code
file name		
UF		
DAD name		

- for assignment to a DAD:

00000011	file code
D	disc file code
not used	
DAD name	

- for assignment to another file code:

00000011	file code
	2nd file code
not used	

The DAD specified in the second word of the block, can be specified in one of three ways:

- by specifying the DAD file code previously assigned by an Assign of type 5).
- by specifying only the disc file code (F0 to FF), thus implicitly specifying the first DAD on the disc
- by specifying disc file code and DAD name, in which case bit D must be set to 1, to indicate that the block contains a DAD name.

For assignment to a DAD and another file code, the same assign type must be used (/03). The distinction lies in the second word: If it contains a DAD or disc file code it is a DAD assignment, otherwise it is an assignment to another file code.

Use

By means of this monitor request the user may assign a file code to either a physical device or a disc file.

Notes:

- A file code can be assigned to only one physical device or file at a time; however, a physical device or file can be used through different file codes.
- As the DRTM is used as a supervisor essentially, error checking facilities are limited and system file codes are not protected.

Therefore, the user must be very careful not to destroy any system (especially disc) file codes.

- When a new file code is assigned, the old assignment is first deleted. This implies, that when an assign request is refused, the old file code no longer exists and a new request must be given to restore the old assignment, if desired.
- This request is handled by a module connected to level 49, which may be either memory resident or read only.

Upon completion of the request, the system responds as follows:

A7 = 0: assignment performed

≠ 0: assignment refused, for one of the following reasons:

= 1: I/O error on disc

= 2: no spare entry in file code table

= 3: no file description table available

= 4: device or disc file code unknown

= 5: disc overflow or too many granules requested

= 6: file unknown

= 7: second file code unknown (assign type/11)

= 8: more than 7 file codes assigned to the same disc file.

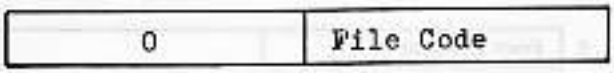
DELETE A FILE CODE

Calling Sequence

```
LDKL  A8,PARAM
LKM
DATA  24
```

where:

PARAM points to a parameter word:



File code indicates the file code which must be deleted.

Use

By means of this request the user can ask the system to delete a file code which has previously been assigned by monitor request or control command.
If the file code had already been deleted, this request will be ineffective. If the file code does exist, it is removed from the file code table.
If the file code to be deleted is a disc file code, its file description table will be released and, for temporary files, the granules of the file will become available again.
This request is handled by a module on level 49, which may be either memory resident or read only.

Note: The system does not protect any file codes, system or user. For sequential files, it will be assumed that they have already been closed implicitly by EOS or EOF mark detection.
The system responds as follows:
A7 = 0: the file code has been deleted
= 1: I/O error occurred while reading the file to release the granules occupied by the temporary file, if any. The file code is still removed from the file code table.

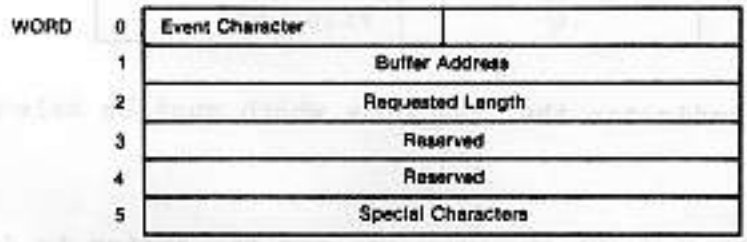
READ UNSOLICITED OPERATOR MESSAGE

Calling Sequence

```
LDKL  A8, PARAM
LKM
DATA  25
```

where:

PARAM points to a parameter block with the following layout:



The first bit of the event character is used to indicate whether the event has occurred or not.
Word 1 contains the address of the first character of the buffer where the message will be stored with its two leading characters.
Word 2 contains a value equal to the length of the buffer.
Word 5 contains the two leading characters which are used to identify the message.

Use

By means of this request the operator is given the possibility of interrupting the system through use of the INT button on the control panel and sending a message to the program in which this request was made. For the system to be able to recognize the message and send it to the right program, the operator must start the message with two special characters, as he has defined them in word 5 of the PARAM block, described above. It is recommended to use not alphanumeric characters, but special characters, so as not to interfere with the regular operator commands.

When the system has found the two characters matching the ones that were typed in, the message is stored in the user buffer and the event is declared as having occurred. The scheduled label facility can be used to process the message.
The message must be sent from the system keyboard (file code /EF) and the maximum message length is 72 characters.
The event count is incremented when the request is given and decremented when the operator types in the message.
If more requests are given, with the same identification characters, they are serviced first-in-first-out. Requests are queued, but removed from the queue after servicing, so if a message is to be given more than once, the request must be reinitialized.

Note: If this monitor request has been given, the operator must type in a message with the two special characters or give a monitor request (LKM26) to cancel this one, otherwise the program will not be able to make its exit (see below).

CANCEL REQUEST FOR AN UNSOLICITED OPERATOR MESSAGE

Calling Sequence

```
LDKL    A8, PARAM
LKM
DATA    26
```

where:

PARAM points to the same block to which is referred under the Request for an Unsolicited Operator Message (LKM25).

Use

When a monitor request is issued in a program to Read an Unsolicited Operator Message (LKM25), the event count is incremented by the monitor and decremented again only when the operator types in the defined message.

For a program to be able to make its exit, the event count must be zero. So, if it has been un-necessary or undesirable to type in the message, this request (LKM26) can be issued to have the event count decremented and the LKM25 request removed from the queue. The effect is that the event is considered as having occurred: the event bit is set and, if a scheduled label is attached to this request, it is started.