RCSL No: 43-GL10166

Edition: March 1980

Author: Jørgen Hansen

Title:

DOMUS
User's Guide, Part 2

A/S REGNECENTRALEN
af 1979

Keywords:

RC3600, DOMUS, Utilities, Manual.

Abstract:

This manual describes the utility system for the disc operating system for the RC3600 line of computers.
This manual substitutes 43-RI0432.

(80 printed pages)

Printed by A/S Regnecentralen af 1979, Copenhagen

421 1341

TABLE OF CONTENTS PAGE

TABLE OF CONTENTS (continued) PAGE

1. INTRODUCTION 1.

This manual describes the utility program system running under the RC3600 DOMUS system.

Use of utility programs is based on the DOMUS S-function 'utility program load', and can be reviewed as an extension of the S-commands.

The utility programs are designed in order to ease the administration of disc files, the possibility to make backup and hard-copy of the files and to help the MUS - programmer with a set of tools in his program production.

2. THE UTILITY SYSTEM 2.

2.1 Utility Call 2.1

The utility call is based on the S-function 'utility program load' given by the format

<filename> [<params>]

As seen from DOMUS User's Guide, Part 1 the function is defined by the filename which is not a normal S-function.
The function is executed by loading the process placed in file <filename>, and transferring the <params> section to the process after fundamental syntax analysis and packing.
Further interpretation of the parameters is left to the utility program.
After completion the utility program is removed from the memory by DOMUS, and the operating system will write a message on the operator console: FINIS <program name>.

If a utility call is too long to be contained in one line, it may be divided into more lines by use of exclamation signs. This is described in DOMUS User's Guide, Part 1. An example is this call of the utility program 'DELETE':

```
DELETE A1 A2 A3 A4 !
       ! A5 A6
FINIS DELET
```

2.2 Parameter Format

In order to get a simple and standardized parameter interpretation all utility programs, with a few exeptions, use the same format and check procedure.
The utilities have each a fixed number of parameters, each specified with a parameter type, a mnemonic parameter name, a specified position in the parameter string and a default value.

Five parameter types are defined:

1) NAMES — as defined in DOMUS User's Guide, Part 1

2) FILENAMES — is defined as
<filename>::= <name>|<name>/<name>|<name>:<number>

3) TEXTS — as defined in DOMUS User's Guide, Part 1

4) NUMBERS — as defined in DOMUS User's Guide, Part 1

5) BOOLEANS — is defined as a name with the value NO or YES

The structure of <params> can then be defined as:

<params>::= $\left\{ \text{<sep> [<parameter name>.] <parameter>} \right\}^*$

<parameter name>::= <name>

<sep>::= <space> $\left\{ \text{<space>} \right\}^*$

<parameter>::= <name>|<filename>|<text>|<number>|NO|YES|<dummy>

<dummy>::= *

Occasionally it can be useful to use a name containing special characters, such as dot, slash or space. This can be done including the name in quotation marks.

As seen, each parameter is called by a unique mnemonic name, which can be typed in front of the actual parameter followed by '.', hereby changing the position in the sequence of the parameters. This way of naming the parameters can be omitted if all parameters are assigned in the sequence predefined by the program, and need only to be used if one or more parameters are skipped or the defined sequence is broken.
Skip of a single parameter can be done by use of the dummy parameter '*'.
If a parameter is not defined in the call, it will get the default value as defined in the program.

The parameter type <filename> has three possible formats which all refer to a catalog entry.
The first format '<name>' refers to the catalog entry with the name <name> in the current catalog (see DOMUS User's Guide, Part 1 about the S-functions CONNECT and RELEASE).
The second format '<name>/<name>' is interpreted as a catalog entry in a catalog possibly different from current catalog. Here the name before the slash is the name of the catalog and the name after the slash is the name of the catalog entry.
For further information about the catalog system, see chapter 3 in this manual.
The third format '<name>:<number>' is used when a catalog entry residing in the main catalog on a unit different from current unit is referred to. The referred entry is the one with the name <name> in the main catalog on unit number <number>.

If any conflict between the parameters and the parameter names typed and the expected format is detected, the utility function is terminated with an error message (***PARAM) written on the console.

For further explanation take the following fictive utility-program MAIN:

MAIN

Format:

MAIN IN.<filename> COUNT.<number> OP.<boolean> TX.<text>

Function:

Undefined.

Parameters:

Default values of call are:

IN.$PTR COUNT.20 OP.YES TX.'<0>...<0>'

Maximum textlength for parameter TX is 10 characters.

Examples:

1) MAIN $CDR 100 TX.'ABCD'

2) MAIN $CDR 10 NO

3) MAIN XX:3 TX.'AB'

4) MAIN PIP/XX

5) MAIN TX.'QV' OP.NO

6) MAIN $CDR * NO

Explanation of the examples:

1) Used values are:

```
IN:     $CDR
COUNT:  100
OP:     YES                          default
TX:     'ABCD<0><0><0><0><0><0>'
```

2) Used values are:

```
IN:     $CDR
COUNT:  10
OP:     NO
TX:     '<0>...<0>'                  default
```

3) Used values are:

```
IN:     XX                           file on unit 3
COUNT:  20                           default
OP:     YES                          default
TX:     'AB<0>...<0>'
```

4) Used values are:

```
IN:     XX                           file in subcatalog PIP
COUNT:  20                           default
OP:     YES                          default
TX:     '<0>...<0>'                  default
```

5) Used values are:

```
IN:     $PTR                         default
COUNT:  20                           default
OP:     NO
TX:     'QV <0><0>..<0>'
```

6) Used values are:

```
IN:     $CDR
COUNT:  20                           default as skipped by *
OP:     NO
TX:     '<0>...<0>'                  default
```

3. THE CATALOG SYSTEM

3.1 Catalog System Description

The RC3600 Catalog System makes it possible to divide a disc drive into smaller independent units, called files. These files are identified by name. The description of the files, file descriptions, are kept in a catalog (which itself is a file) stored on the disc. A file description contains information about the file such as name, length and starting position. The catalog consists of a number of catalog entries. Every file description is kept in a catalog entry.

The RC3600 Catalog System deals with units. A unit may be a disc drive, a part of a disc drive or include several disc drives. Each unit has its own main catalog, describing the files contained on that unit. A file may be a new catalog (subcatalog) for a number of files. Files in different catalogs may have the same names. A file containing a catalog is always named 'SYS'.

All units in the system are described in a file called CATW, which resides on unit 0. A unit may be initialized and a new catalog may be defined either at the time of system generation or by the utility NEWCAT.

A subcatalog is created by calling the utility SUBCAT.
When files described in a subcatalog are to be created, accessed or deleted, a link to the subcatalog must exist in a file called 'SYSSC'. This link describes on which unit the subcatalog resides and an eventual protection key. This link is created by the utility SUBCAT. If a protection key is given, it is not possible to write in any file in the subcatalog or to create, change or delete any entry in the subcatalog, unless the DOMUS function CONNECT has been executed with the correct protection key.

All files described in a subcatalog reside on the same unit as the subcatalog itself. To access files in a subcatalog, even on a unit different from unit 0, a link must be created in the SYSSC file on unit 0.

For each disc file and subcatalog to be accessed at the same time, one area process must be present in the memory.
These area processes are brought into memory by loading one or more of the modules CAP2, CAP3, CAP4, CAP8, CAP16 or CAP32, which contain respectively 2,3,4,8,16 and 32 area processes. These may be loaded in any number and in any combination. An area process can not be removed from the memory unless the system is autoloaded again. When just autoloaded, the system contains one area process to be able to load programs (or more area processes) from the disc.
When a disc file is opened for reading or writing, one area process is reserved, and is returned to the pool of free area processes, when the disc file is closed.
Up to three different processes may use the same area process to read the same disc file at the same time, but if a file is opened for writing by a process, no other process is allowed to access the file.

3.2 Filename References in Utility Calls 3.2

Most utilities has one or more filenames among their parameters. Such a filename is used to point out a certain entry in a certain catalog. As mentioned in chapter 2.2, a filename has one of three different formats. The use of these formats is described below.
In part 1 of this manual is described that DOMUS has a current catalog which may be selected by the DOMUS-functions CONNECT and RELEASE. After autoload of the system, current catalog is the main catalog on unit 0.
If reference to a file in the current catalog is wanted, the simple filename format '<name>' is used.
If reference to a file in the main catalog on another unit than unit 0 is wanted, the filename format '<name>:<number>' is used. The number then points out the selected unit.
If reference to a file in subcatalog, which is not current catalog is wanted, the filename format '<name>/<name>' us used. The first name is the name of the subcatalog, and the second name is the name of the wanted file.

If reference to a file in the main catalog on unit 0 is wanted, when connected to a subcatalog, the filename format <name>/<name> is used too, but the first name must be 'CAT', and the second name of the name of the file.

Examples:

1) COPY PIP PAP

The file PIP is copied to file PAP. Both files reside in current catalog.

2) COPY PIP:1 PAP

The file PIP in the main catalog on unit 1 is copied to file PAP on current catalog.

3) COPY PIP SUB/PAP

The file PIP in current catalog is copied to file PAP in subcatalog SUB. This is not allowed, if subcatalog SUB is protected by a key. In this case, use the next example.

4) CONNECT SUB 100
COPY CAT/PIP PAP

The file PIP in the main catalog on unit 0 is copied to file PAP in subcatalog SUB.

5) SUBCAT LINK SUB1:1
FINIS SUBCA
CONNECT SUB1
COPY CAT/$PTR PAP

First a link to subcatalog SUB1, which resides on disc unit 1, is created. Then the file $PTR in the main catalog on unit 0 is copied to file PAP in this subcatalog.

4. PERIPHERAL DEVICES 4.

4.1 Device Handling 4.1

Only few utility programs have been designed to use special devices, and to avoid each program from keeping track of the device characteristies in the handling, a general way of file i/o initialization has been implemented.
Documents which are not disc files are defined in catalog entries, called device descriptors, which hold information of the driver name to be used, the document kind, the mode of operation, the status bits to cause giveup, and finally the document position given by the file number and block number.
Supplied with the DOMUS system are a number of predefined device descriptiors, which can be found in appendix B, but the user can change these or create his own device descriptors. This is done by using the utility program SET.
Format of the SET call is:

SET NAME.<filename> DEVICE.<name> FILE.<number> BLOCK.<number>
MODE.<number> KIND.<number> MASK.<number>

Explanation of the parameters:

NAME: Entry name of the device descriptor to create or change.

DEVICE: Process name of the driver that supports the device.

FILE, BLOCK:
The wanted position of the document. Not relevant for all devices, or it may have special meaning.

MODE: Generally mode=1 means binary input, and mode=3 means binary output, but each device may have defined its own mode bits.
For devices enabling both input and output it is sufficient to specify the mode for input, as the utility programs are able to change the mode.

KIND: The kind is easily given in radix 2, and can be interpreted after the scheme:

Bit 15: character oritented devices, i.e. line printer, paper tape reader, paper tape punch.

Bit 14: block oriented devices, i.e. magnetic tape, card reader, discs.

Bit 13: positionable devices, i.e. magnetic tape, discs.

Bit 12: repeatable devices, which means devices supporting error recovery, i.e. magnetic tape, discs.

Bit 11: disc files only, not used in device descriptors.

MASK: Generally a mask which enables all status bits except bit 3, bit 4 and bit 5 (the soft status bits) is sufficient.

When initializing input/output, the utilities looks up the given entry in the given catalog, and depending of this entry and the function, one of the following actions is taken:

1. If the entry exists and describes an ordinary disc file, and the function is input, data is transferred from this disc file.

2. If the entry does not exist, and the function is output, a new disc file is created, and data is transferred to this disc file.

3. If the entry does not exist, and the function is input, the utility program will terminate with an error message.

4. If the entry does already exist and describes an ordinary disc file, and the function is output, the utility program will terminate with an error message to prevent overwrite of existing data.

5. If the entry exist and is a device descriptor, the file is to be transferred to or from the document described by the device descriptor.

The block size of the document is normally 512 bytes, unless it can be changed by the parameters.

4.2 Examples of Use of Device Descriptors

1. Define the device descriptor $MT describing file 10 on first magnetic tape station:
   SET $MT MT0 10 1 1 2'1110
   The block number and the mode are set to 1, and the kind is set to repeatable, positionable and blocked.

2. No device descriptors exist on a subcatalog just created, and this means that special care must be taken when the current catalog is such a catalog. The device descriptors on the main catalog may be used in the following way:

   PRINT PIP CAT/$LPT

   or

   COPY CAT/$PTR PAP

3. Another way to acces peripheral devices when connected to a subcatalog is to create the device descriptors as entries in the subcatalog:

   CONNECT SUB

   SET $MT0 1 1 1 2'1110 8'161777

4. The task 'copy file number 4 on second magtape station to file number 6 on first magtape station' is done in the following way:

   SET MTIN MT1 4 KIND.2'1110

   FINIS SET

   SET MTOUT MT0 6 KIND.2'1110

   FINIS SET

   COPY MTIN MTOUT

   FINIS COPY

5. Copying of a number of binary paper tapes, each to a separate file on magnetic tape can be done in the following way:

```
SET MT MT0 1 KIND.2'1110

FINIS SET

COPY $PTRN MT

FINIS COPY

SET MT FILE.2

FINIS SET

COPY $PTRN MT

FINIS COPY

SET MT FILE.3
```

and so on. This is possible, as all non given parameters are taken from the existing device descriptor.

4.3 Standard Coversion

When utility programs produce output to a printer, data is send as ASCII characters. Charaband printers and serial printers are able to accept these ASCII data, but some line printers requires the presence of a conversion table in memory, as some RC3600 line printers are able to use different print drums.
When the conversion table is loaded into memory, it must be connected to the line printer driver by call of the utility program STACO.
The function of STACO is to connect a conversion table placed in a core item to the selected driver and to prevent the conversion table from removal without the driver's knowledge. A call of the S-function LIST/CORE will produce a list of coreitems, which will show that the driver is placed as owner of the coreitem containing the conversion table.
Regret of a standard conversion can only be done by killing the owner driver, and reload of the process.
Standard conversion is only used by the driver, if the reserver program does not specify its own conversion table, hereby enabling application programs with its own defined table to run without change.
Load of line printer driver and the conversion table P218, and connecting it to the driver can be done in this way:

```
LOAD LPT P218 (STACO LPT P218)

FINIS STACO
```

5. SYSTEM MESSAGES

5.1 Standard Messages

All system message reside on the file SSYSE on the main catalog on unit 0, and each message is identified by a unique number in the interval 1 - 9999.
Each error occuring in the system is converted to an appropriate message number, and the text to be output is fetched from the common file. This enables the user to add, translate, change or re-formulate the errortext connected to the number.

In order to avoid conflicts the system message have been split into different groups depending of the use:

1 - 99	DOMUS operating system.
100 - 1999	Standard utility programs.
2000 - 2999	Standard device errors. Each text is found by adding a base connected to a specific device to the number of the leftmost status bit set.
3000 - 7999	Standard application programs.
8000 - 9999	Informative texts.
9000 - 9999	Customer available texts, free to use for any customer designed texts or message.

In groups from 1 - 7999 the texts are always defined with the heading 'NNNN***', where NNNN is the text number.

The messages in the group 1 - 99 are defined and explained in DOMUS User's Guide, part 1.
Message in the interval 100 - 2999 can be found in appendix A of this guide.

Message numbers used in case of errors in catalog operations are found by adding base 100 to the leftmost status bit number, when bit 3 and 4 are removed:

0100 *** CATALOG I/O ERROR, FILE <filename>
The catalog system is malfunctioning, possibly the catalog structure is destroyed or there are software errors in the catalog system.

0101 *** FILE DOES NOT EXIST, FILE <filename>
The referred file is not present in the selected catalog.

0106 *** ILLEGAL OPERATION, FILE <filename>
The operation requested on the referred file is not allowed, either because the file is protected by attributes or another process is using the file.
Non existing unit is specified.

0107 *** NOT ENOUGH DISC SPACE, FILE <filename>
Too few free segments are available for the referred file.

0111 *** FILE DOES ALREADY EXIST, FILE <filename>
The referred file does already exist in the selected catalog.

0112 *** INDEX BLOCK FULL, FILE <filename>
It is not possible to get any more slices for the referred file.

Message numbers in case of errors in the initialization of input-/output or datatransfer to files are found by adding base 120 to the leftmost status bit number after removal of bit 3 and 4:

0120 *** CATALOG I/O ERROR, FILE <filename>
See error 0100.

0121 *** FILE DOES NOT EXIST, FILE <filename>
See error 0101.

0126 *** FILE IN USE, FILE <filename>
The referred file is reserved for exclusive use by another process, or a process with the same name as the file is present in core.

0127 *** NO FREE AREA PROCESS TO FILE <filename>
The common pool of area processes is empty. It can be extended by loading more area processes.

0131 *** END MEDIUM ON FILE <filename>
Physical end of file is found before the expected logical end.

0132 *** MAP/FILE EXCEEDED, FILE <filename>
The maximum filelength has been reached on the referred file, or no more free segments due to a configuration error on the disc.

Message numbers used in case of device errors are found by adding the base 2000 to the number of the leftmost bit set after removal of bit 3 and 4:

2000 *** DISCONNECTED, FILE <filename>
The device is not ready, or the hardware is missing.

2001 *** OFF-LINE, FILE <filename>
The device is set local.

2002 *** BUSY, FILE <filename>
The device is not able to accept input/output transfers.

2006 *** RESERVED, FILE <filename>
The driver is reserved by another process, or the operation is unknown, or write has been attempted on a write-protected device.

2007 *** END OF FILE, FILE <filename>
See error 131.

2008 *** BLOCKLENGTH ERROR, FILE <filename>
The block read was to big to be held in the used buffer size, or the block output was too big to be held on the document. Format error in input.

2009 *** DATA LATE, FILE <filename>
The CPU was too busy to respond on a memory reference from the device. On high speed devices only.

2010 *** PARITY ERROR, FILE <filename>
One or more characters had a parity error.

2011 *** END MEDIUM, FILE <filename>
See error 131.

2012 *** POSITION ERROR, FILE <filename>
The driver is unable to find the position requested.

2013 *** DRIVER MISSING, FILE <filename>
The driver process is not loaded.

2014 *** TIMEOUT, FILE <filename>
The device did not respond within a specified time.

5.2 Message Generation 5.2

Supplied with the system is a system message file, SSYSE. This file is produced from the text file STERR, which is also on the system.
It is possible for the user to modify the SSYSE file and thereby change the already existing messages or include new messages. This is done by changing the text file STERR by means of the text editor and then replace the old SSYSE file by a new one. This must be done very carefully. First the attributes of the old file must be changed by the utility call 'CHATR SSYSE', and then the file is deleted by the utility call 'DELET SSYSE'. The new file is created by the utility GENER.
It is recommended to protect the SSYSE file with the attributes P and W.
Please note that if SSYSE is not existing, any error will result in the message '*** SYSTEM ERROR 24'. You must also be aware, that it is not possible to autoload DOMUS if the disc does not contain a SSYSE file !

The format of the text file can be found in the description of utility GENER. The maximum text length is 502 characters, and undefined message numbers will not occupy disc space. If a non-exising message number is referred, DOMUS will return the text '*** UNREGISTERED ERROR'.

6. STANDARD UTILITY PROGRAMS 6.

This chapter contains descriptions of all available DOMUS utility programs. All the mentioned utility programs are not neccessarily present on a standard DOMUS system.

6.1 ADDEX 6.1

Format:

ADDEX COM.<name> EXP.<name>

Function:

This program is used togehter with the DOMUS utilities NAMEX and GEN. It is used to insert some filename explanations in a GEN command file. For further information see DOMUS Utility ADDEX, User's Guide. This utility works on main catalogs only.

Parameters:

COM: Name of GEN command file in which filename explanations should be inserted.

EXP: Name of filename explanation file.

Default: ADDEX COM.<0>...<0> EXP.SYSEX

Example:

ADDEX PIP

The filename explanations from the file SYSEX will be added to the GEN command file PIP.

6.2 AMXINIT 6.2

Format:

AMXINIT IN.<name>

Function:

This program is able to initialize a RC3682 asynchronous multiplexer acording to parameters given in a command file. For further information, please consult the manual RC3682 AMX Driver Initialization Program, User's Guide.

Parameters:

IN: Name of text file containing the parameters.
Default: AMXINIT IN.$PTR

Example:

AMXINIT AE082

The AMX driver is initialized according to parameters given in the text file AE082.

6.3 APPEND 6.3

Format:

APPEND OUT.<filename> IN.<filename> <filename>...

Function:

This program copies from IN.<filename> to OUT.<filename>. Trailing zeroes in each input file are skipped. Up to 10 input files may be specified.

Parameters:

OUT: output filename

IN: input filenames

Default: As default all filenames are empty. The output file and at least one input file must be specified.

Example:

APPEND $PTP DATA1 DATA2 DATA3

The files DATA1, DATA2 and DATA3 are copied to file $PTP (i.e. the paper tape punch).

6.4 CATLIST

Format:

CATLIST MASK.<filename> OUT.<filename> TEXT.<text>
ATT.<name> COMPR.<boolean>

Function:

This program produces a sorted list of catalog entries in a specified catalog. If the process TIME is loaded, current date and time is included in the headline.
Format of the output is for normal entries:

1: Entry name
2: Entry attributes, with following interpretation:
   C: catalog entry
   S: subcatalog
   B: big slice extension (special use)
   L: link entry (not used)
   P: permanent entry
   W: write protected entry
   E: entry only
   D: device descriptor
   V: extendable file
   F: fixed length file
3: Segment number of index block
4: Length of file (in segments)
5: Reserved length (in segments)
6: Entry optional words 1 as ASCII string
7: Entry optional words 2 as ASCII string

In case of device descriptor entry:

1: Entry name
2: Entry attributes as for normal entry
3: Driver name
4: File number (decimal)
5: Block number (decimal)
6: Mode (decimal)
7: Kind (binary)
8: Giveup mask (octal)

Parameters:

MASK: The name part of this parameter is a mask selecting the entries to be printed. Any entry name that fits the mask will be listed. The character $ replaces any character.
E.g: The mask PIP$ will list all entries with name-length 4 where the first three character are PIP.
The catalog part of the parameter specifies which catalog to examine. If no catalog is specified, current catalog will be examined.

OUT: Output filename.

TEXT: The ASCII string specified in this parameter will be output as a headline of the listing. Maximum length of the string is 40 characters.

ATT: If one or more attributes is specified to this parameter, only entries that have of least one attribute in common with this mask will be listed.

COMPR: If this parameter is specified as YES, a compressed listing will be produced. No headlines will be output. Entries with attributes C,S or E and some system entries will not be listed.
For each entry only the name and the filelength will be output.

Default: CATLIST MASK.$$$$$ OUT.$TTY TEXT.<0>...<0>
ATT.<0>...<0> COMPR.NO
i.e. all entries on current catalog is listed on device described by $TTY.

Example:

```
CATLIST SUB/A$$$$ 'SOURCE FILES' P
```

i.e. all entries in subcatalog SUB with A as the first character of the name and with attribute P will be output to file PIP on current catalog. The text 'SOURCE FILES' will be output as a headline of the listing.

6.5 CHATR

Format:

CHATR NAME.<filename> ATT.<name>

Function:

This program will change the attributes of the specified file to the specified value.

Parameters:

NAME: Name of entry to be changed

ATT: New attributes. The only changeable attributes are:

B: big slice extension (special use)
P: permanent entry
W: writeprotected file
V: extendable
F: fixed length

Default CHATR NAME.<0>...<0> ATT.V

Example:

CHATR PIP PW

The file PIP on current catalog is changed to have the file specifications permanent and writeprotected.

6.6 COMP 6.6

Format:

COMP IN.<filename> OUT.<filename> LIST.<filename>
INCOD.<filename> NAME.<name> IDENT.<name> OPCOM.<name>
MODIF.<name> BLOCK.<number>

Function:

This utility is the MUSIL compiler, which is able to produce executable binary programs from an ASCII source text. For further information see MUSIL Compiler, Operator's Guide.

Parameters:

IN:	Input source file name.
OUT:	If specified, binary output file name.
LIST:	If specified, filename where to list the source text.
INCOD:	The name of the file from which codeprocedures are loaded.
NAME:	Process name of object code.
IDENT:	If specified, the ident is output as an ASCII text in front of the object code.
OPCOM:	Driver name of operator device of object code.
MODIF:	One to six characters denoting special functions. The following are allowed: B (extra message buffers), C (coroutine program), N (no process descriptor), X (XREL code is produced), P, P1, P2, P4, P8 (paged program).
BLOCK:	Output block size. Maximal value is 512 bytes.
Default:	COMP IN.$PTR OUT.<0>..<0> LIST.<0>...<0> INCOD.<0>...<0> NAME.MAIN IDENT.<0>...<0> OPCOM.TTY MODIF.<0>...<0> BLOCK.512.

Example:

COMP S1000 P1000 $LPT ULIB PIP MODIF.X

The file S1000 is compiled and the binary output is placed in file P1000. A listing of the sourctext is output to file $LPT, and codeprocedures are fetched from file ULIB. The process name of the object program will be PIP.

The object program will contain XREL code.

Program size:

The computer will, besides its own size, use the largest free coreitem for working area.

6.7 COMPARE 6.7

Format:

COMPARE IN1.<filename> IN2.<filename> MAX.<number>

Function:

This program will compare two files byte by byte. If any difference between the two files is detected, a message is output to the operator console. This message contains the absolue byte number and the two different values. All values are decimal. The program will terminate either when end of file is reached or when a spefified number of unmatching bytes have been detected. Leading and traling binary zeroes are skipped.

Parameters:

IN1: Name of first input file.

IN2: Name of second input file.

MAX: Maximum number of unmatching bytes before termination.

Default: IN1.<0>...<0> IN2.<0>...<0> MAX.1

Example:

COMPA DATA $PTR

The files DATA and $PTR are compared.

6.8 CONFIGURATION

Format:

CONFI LIST.<filename>

Function:

The titles of modules placed in the current DOMUS basic system are listed on the file specified by parameter LIST.

Parameters:

LIST: Name of list file.

Default: CONFI LIST.$TTY

Example:

CONFI $LPT

The files of the modules in the basic system are listed on file $LPT.

6.9 COPY 6.9

Format:

COPY IN.<filename> OUT.<filename> BLOCK.<number>

Function:

This program copies the file specified by parameter IN to the file specified by parameter OUT. Output blocksize is selectable. If a disc file is copied to another disc file, the tail part of the entry is copied too.

Parameters:

IN: Input filename

OUT: Output filename

BLOCK: Block size used on output file. Maximum value is 512 bytes

Default: COPY IN.<0>...<0> OUT.<0>...<0> BLOCK.512

Example:

COPY $PTR $PTP

The file $PTR (i.e. the paper tape reader) is copied to file $PTP (i.e. the paper tape punch).

6.10 CREATE

Format:

CREATE NAME.<filename> SIZE.<number> ATT.<name>

Function:

This utility is used to create a file with specified name, size and attributes.

Parameter:

NAME: Name of the file to be created.

SIZE: Number of segments in the file.

ATT: Attributes of the file. Allowed values are: B (big slice extension), P (permanent file), W (writeprotected file), V (extendable) and F (fixed size).

Default: CREATE NAME.<0>...<0> SIZE.1 ATT.V

Example:

CREATE WORK 100 PF

The file WORK is created on current catalog with size 100 segments with attributes permanent and fixed size.

6.11 DCOPY 6.11

Format:

DCOPY FUNC.<name> UNITA.<number> UNITB.<number>

Function:

This program is a disc copy and backup program. It is able to copy one disc unit to another, to compare two disc units, to copy a disc unit to magtape and to copy a magtape to a disc unit.
After having been loaded, the program will write the current function on the operator console, and the operator must confirm with 'yes', if the function is accepted.
The program is able to handle multi-reel magtapes. When the end of magtape is reached, the message 'mount next magtape' is output to the console, and when the next reel is mounted, the operator must answer 'return'.
When the function is completed, the message 'end function' is output to the console and must be answered with 'return'.

Parameters:

FUNC: Function of the program. Allowed values are:

- BACKU: The contents of the disc unit specified by parameter UNITA is copied to magtape. The parameter UNITB is dummy.
- RESTO: The contents of a magtape produced by the BACKU function is copied to the disc unit specified by UNITA. The parameter UNITB is dummy.
- COPY: The contents of the disc unit specified by parameter UNITA is copied to the disc unit specified by parameter UNITB.
- COMPA: The contents of the two disc units specified is compared to detect any difference between them.

UNITA: If function is BACKU or RESTO, this parameter specifies the disc unit to be used. If function is COPY or COMPA, it specifies the source disc unit.

UNITB: If function is COPY or COMPA, this parameter specifies the destination disc unit.

Default: DCOPY FUNC.BACKU UNITA.0 UNITB.1

Example:

DCOPY COPY

The contents of disc unit 0 is copied to disc unit 1.

Note:

This program must be used very carefully, as it is able to overwrite any disc unit in the system, regardless of any protection.

6.12 DELETE 6.12

Format:

DELETE NAME.<filename> <filename>....

Function:

This program deletes the files with the specified names. Max. 20 files may be specified.

Parameters:

NAME: Name(s) of file(s) to be deleted.
Default: DELETE NAME.<0>...<0> <0>...<0> ...

Example:

DELETE WORK SUB/WORK1

The file WORK on current catalog and the file WORK1 on subcatalog SUB will be deleted.

Note:

Ocasionally an erroneous termination of a utility program may leave a workfile with a name containing a dot or a space. These files may be deleted by including the name in quotation marks, for instance:
DELETE '.XEC'

6.13 DISK

Format:

DISK UNIT.<number>

Function.

Number of free segments and number of used segments on the catalog unit described by the parameter UNIT is output to the operator console.

Parameters:

UNIT: Number of the unit to be examined. Must not exceed 255.

Default: DISC UNIT.0

6.14 DOMAC

Format:

DOMAC MODE.<name> LIST.<filename> BIN.<filename>
LINES.<number> PERM.<filename> SYMB.<filename>
MACRO.<filename> XREF.<filename> <filename> <filename> ...

Function:

This program is the DOMAC assembler. For further information, see Introduction to the DOMAC Assembler and DOMAC, DOMUS Macro Assembler User's Guide. Please note that the parameter format is not fullfilling the standard of utility calls. Each parameter must be preceded by the parameter name, except for the source file specification which has no parameter name. The order of the parameters is not defined in the same may as other utilities. All used files must reside on the main catalog on unit 0.

Parameters:

MODE: A name of maximum five characters. Each character specifies a special function of the DOMAC assembler.
Allowed characters are:

- A: Add all semipermanent symbols to cross-reference listning.
- O: Overwrite all listing suppression.
- R: Add referenced semipermanent symbols to cross-reference listing.
- S: Skip pass 2 and create a new semipermanent symbol table and macro definition file.
- W: A warning is listed for data words where bit zero is set by means of @.
- X: Do not make a cross-reference listing.
- Z: Include a size block in binary output.

LIST: Name of file where to output the program listing.

BIN: Name of file where to output the relocatable binary object code.

6.15 DUAL 6.15

Format:

DUAL BOOT.<name> SYS.<name>

Function:

This program loads the second processor in a dual processor system. First the program autoloads the second processor via the front end processor adapter (FPA) and transmit the FPA relocatable binary loader, which is taken from a disc file specified as parameter. The FPA bootstrap loader is able to receive relocatable modules and link them together to form an exectutable core-image as neccessary to start up a MUS (DOMUS) system. The binary modules to be transmitted should be specified in a disc file given as the second parameter. For further information, please consult the manual DOMUS Utility DUAL, User's Guide.

Parameter:

BOOT: The name of a disc file on the main catalog containing the FPA relocatable binary bootstrap loader (FPBxx) in absolute binary format.

SYS: The name of a disc file on the main catalog containing the names of modules to be loaded as an ASCII text string.

Default:

DUAL BOOT.FPABT SYS.Q3600

Example:

DUAL

The second processor is loaded using the loader from the disc file FPABT. The basic system in the second processor is linked of the modules described in the disc file Q3600.

6.15 DUAL 6.15

Format:

DUAL BOOT.<name> SYS.<name>

Function:

This program loads the second processor in a dual processor system. First the program autoloads the second processor via the front end processor adapter (FPA) and transmit the FPA relocatable binary loader, which is taken from a disc file specified as parameter. The FPA bootstrap loader is able to receive relocatable modules and link them together to form an exectutable core-image as neccessary to start up a MUS (DOMUS) system. The binary modules to be transmitted should be specified in a disc file given as the second parameter. For further information, please consult the manual DOMUS Utility DUAL, User's Guide.

Parameter:

BOOT: The name of a disc file on the main catalog containing the FPA relocatable binary bootstrap loader (FPBxx) in absolute binary format.

SYS: The name of a disc file on the main catalog containing the names of modules to be loaded as an ASCII text string.

Default:

DUAL BOOT.FPABT SYS.Q3600

Example:

DUAL

The second processor is loaded using the loader from the disc file FPABT. The basic system in the second processor is linked of the modules described in the disc file Q3600.

6.16 EDIT

Format:

EDIT [<filename>]

Function:

This program is the system text editor. It is described in the manual RC3600 Text Editor.

Parameters:

If a filename is typed, the editor will perform a UY command on this file, and the first page of this file will be ready in the edit buffer.

Example:

EDIT PIP

The editor is loaded and the file PIP is opened for editing.

6.17 EXEC 6.17

Format:

EXEC IN.<filename> LIST.<filename> STOP.<boolean>
CONT.<filename>

Function:

This program is the DOMUS batch processor module which executes utility program calls and S-functions in a user defined sequence. For further information see DOMUS utility EXEC, User's Guide.

Parameters:

IN: Name of the file from where the commands are read.

LIST: Name of the file on which a log of the commands are listed.

STOP: If this parameter equals YES, a check is performed on each utility termination, and if the program reports a non-succesful termination the whole job is terminated with an error message.

CONT: If this parameter is typed, the operating system is requested to interpret this file after execution of the commands in the EXEC command file.

Default: EXEC IN.$PTR LIST.<0>...<0> STOP.NO CONT.<0>...<0>

Exsample:

EXEC COM $TTY

The S-commands written in file COM will be executed, and a log is output to file $TTY.

6.18 FCOPY

Format:

FCOPY FUNC.<name> MASK.<filename> LIST.<filename>
FILE.<number> DEV.<name> UPDAT.<boolean>

Function:

Dumps or loads all or selected discfiles in a single catalog, except system files and files with names equal to loaded processes, to/from magnatic tape or flexible disc. A log of the filenames and sizes (number of segments) is produced.

Parameters:

FUNC: Defines the function of the program. There are three possible values:

- DUMP: All files on the selected catalog with a name fitting the mask will be transferred from the disc to magtape or flexible disc.
- LOAD: All files on the magtape or flexible disc file fitting the mask are transferred to the specified catalog.
- SAVE: All disc files mentioned in the command file given by parameter MASK are transferred to magtape of flexible disc.
  The command file must reside in the same catalog as the files to be transferred.
  The structure of the command file is described below.

MASK: The catalog part of this parameter defines which catalog to use when reading/writing the disc files.
If a unit number is applied to this parameter, the main catalog on this unit is used.
The meaning of the name part of this parameter depends on the parameter FUNC. If this is DUMP or LOAD, the name is to be taken as a mask where the character $ means any character.
Only files with a name fitting this mask will be transferred.

If the function is SAVE, the name points out a command file which contains a number of names of disk files to be transferred to magtape or flexible disc. The structure of this command file is described below.

LIST: File where to output the log.

FILE: File number to be used on magtape or flexible disc.

DEV: Driver name of the magtape or flexible disc. The normally used drives are MT0 (first magtape station), MT1 (second magtape station), FD5 (first RC3751 flexible disc unit) and FD6 (second RC3751 flexible disc unit).

UPDAT: This parameter has only effect when the function is LOAD. If YES is specified, an already existing disc file with the same name as a file to be loaded will be overwritten, otherwise the disc transfer is skipped.

Default: FCOPY FUNC.DUMP MASK.$$$$$ LIST.$LPT FILE.1 DEV.MT0 UPDAT.NO

Examples:

FCOPY

All files on current catalog is transferred to file 1 on first magtape station. A log is output to file $LPT.

FCOPY LOAD A$$$ $LPT 2 FD5 YES

All files with a namelength of one to four characters and 'A' as the first character are transferred from file 2 on the first flexible disc unit to the current catalog. A log is output to file $LPT.

FCOPY SAVE SUB/COM $TTY

The files in subcatalog SUB described in the command file COM, which also is placed in subcatalog SUB, are transferred to file 1 on first magtape station.
A log is output to file $TTY.

Structure of command file:

The command file used when the function is SAVE is to be seen as a string of ASCII characters devided in lines by the characters CR and/or NL. The first characters of each line are interpreted as a filename. The name is terminated by any character with an ASCII value less than 33 or greater than 126. The rest of each line is skipped.

The file is terminated by the ASCII character EM or physical end of medium.

The output from a call of CATLIST with COMPR.YES fulfills the syntax.

Structure of log file:

The log is headed by some lines giving information about the transport. If process TIME is loaded, date and time is included in these headlines.

This is followed by one line for each disc file to be transferred. If no occurs, each line gives the name and size of the disc file.

If the file transport is not completed, the size is replaced by a text. Possible text are:

PROCESS EXISTS:	A process with the same name as the file to transfer does already exist.
DOES NOT EXIST:	May appear when the function is SAVE. It means that the filename stated in the command file does not exist as a disc file.
DISC ERROR, NOT DUMPED:	A disc file to be dumped or saved is erroreous and impossible to transfer.
NOT LOADED:	A file to be loaded does already exist as a disc file and the parameter UPDAT is specified as NO.

If during DUMP a catalog segment is found erroreous, an error message is written to the log, and the dump is continued with the entries described in the next catalog segment. The error message has the following layout:

*** ERROR ON FILE SYS, SEGMENT NNNNN

The number NNNNN is decimal and relative to the start of the catalog.

6.19 FLCOPY 6.19

Format:

FLCOPY FROM.<number> TO.<number> CYL.<number>

Function:

This program produces an exact copy from one RC3751 flexible disc to another. The two flexible discs must be formatted in the same way, for example by the utility program FLORMA. No bad tracks are allowed on the flexible discs.
The program will output the formatting characteristics to the operator console.

Parameters:

FROM: Unit number of source flexible disc drive.

TO: Unit number of destination flexible disc drive.

CYL: Number of cylinders to copy. Allowed values are 75, 76 and 77. The program FLFORMA does always initialize 77 cylinders.

Default: FLFCOPY FROM.0 TO.1 CYL.77

Example:

FLCOPY TO.2

This call will produce a copy of the flexible disc mounted in unit 0 to the one mounted in unit 2.

6.20 FLFORMA

Format:

FLFORMA UNIT.<number> LENGTH.<number> SIDE.<name>
DENS.<name> SCREW.<boolean> BAD1.<number> BAD2.<number>
ERMAP.<name>

Function:

This program is used to format a RC3751 flexible disc. It may be formatted as normal or screwed. After the flexible disc has been formatted, the data area is filled with binary zeroes, and then a read check is performed to ensure correct formatting.

Paramaters:

UNIT: Unit number of flexible disc to format. Must be specified in order to prevent overwrite of a system disc.

LENGTH: Number of bytes on each sector, must be a multiple of 128 bytes.

SIDE: Specification of number of sides of the flexible disc. Allowed values are S for single sided and D for double sided flexible discs.

DENS: Specifies the density of the flexible discs. Allowed values are S for single density and D or double density flexible discs.

SCREW: Specifies whether the flexible disc is to be formatted as normal or screwed NO mens normal and YES means screwed.

BAD1: Number of first bad cylinder, if any.

BAD2: Number of second bad cylinder, if any.

ERMAP: Format of the error map on the flexible disc. Allowed values are A for ASCII and E for EBCDIC.

Default: FLFORMA UNIT.999 LENGTH.128 SIDE.S DENS.S
SCREW.NO BAD1.0 BAD2.0 ERMAP.A

Example:

FLFOR 1 256 D

This call will format the flexible disc mounted in unit 1 with a sector length of 256 bytes and as double sided.

6.21 GEN 6.21

Format:

GEN OUT.<name> OCOPY.<number> CONTR.<name> LOG.<name>
LCOPY.<number> MARG.<number>

Function:

This program is used for generating program magtapes, card decks, flexible discs and paper tapes.
For further information, please consult the manual RC3600 System Generation with DOMUS GEN, User's Guide.

NB: The program is only able to handle entries in the main catalog.

Parameters:

OUT: The output device. Allowed values are: MT (magtape), FD (flexible disc (RC3650)), PTP (papertape punch), RDP (card reader/punch) and NFD (Flexible disc (RC3751)).

OCOPY: Number of copies of the output.

CONTR: A disc file generated by the text editor containing the commands to GEN.

LOG: If stated, GEN will use this file to output a log of the work.

LCOPY: Number of copies of the log.

MARG: Margin on the log. Number of spaces (max. 10) to be printed before each line of the log.

Default: GEN OUT.MT OCOPY.1 CONTR.<0>...<0> LCOPY.1 MARG.0

Example:

GEN MT 1 COM $LPT 2

This call will produce a program magtape as described in the control file COM and write two copies of the log on $LPT.

Program size:

If output on magtape or flexible disc (RC3650 or RC3751), memory must be available to get a coritem as big as the biggest absolute binary module to be put on the tape/disc.

6.22 GENER

Format:

GENER IN.<name> LIST.<name> OUT.<name>

Function:

The program reads an ASCII file containing all system messages, produces a listing of these message and of any error detected in the text file, and creates a disc file containing the message in a form accessible by DOMUS. Number of errors is output to the console, and line and page number of each error can be found in the listing.

NB: Only files on the main catalogs are handled by the program. Please refer to chapter 5.2 about the use of this utility.

Parameters:

IN: Input source file name.
LIST: List file name.
OUT: Output file name
Default: GENER IN.STERR LIST.$LPT OUT.SSYSE

Exemple

GENER OUT.MSYSE

This call causes GENER to read and check the ASCII text file STERR and to create the file MSYSE with listing on $LPT.

Format of input file:

The ASCII input file contains the system message in ascending order each in the form:
<number> <comment> <message><nl>
<number> identifies the number of the message. It is written as a number of decimal digits terminated by a space. The value must be inside the range 0 - 9999.
<comment> is any string of characters not including quotation mark (').

<message> is the message coresponding to the message number.
It is a string of characters started and terminated by quotation marks. Character with a value less than 32 has no effect on the output file, and characters with a value greater than 127 are replaced by a question mark. If a character with a value outside the range 32-127 is wanted in the output file, it must be written in the form: <NNN>, where NNN is the decimal value of the character.
The very first line of the file must not include any message. It is used only to identify the file.
The file is terminated by physical or logical end medium.
Example of text file:

```
0000 DOMUS 77-02-15 REV 03.00
0001 DOMUS 77-02-15 '0001 *** SYNTAX'
0304 BASIC 78-01-25 'ILLEGAL KEY'
0349 BASIC 77-01-25 ''
2000 ANY   77-03-25 '2000 *** DISCONNECTED, FILE '
```

6.23 LIBE

Format:

LIBE LIB.<filename> OUT.<filename> FUNC.<name>
PROC.<filename>

Function:

This utility is used for maintenance of MUSIL codeprocedure libraraies. It is possible to perform insert, delete, extract and list operations on a library.
A codeprocedure is defined as a number of relocatable binary blocks with a leading title block and a trailing start block.

Parameters:

LIB: Name of the codeprocedure library to use.

OUT: Name of file to which a log will be output.

FUNC: Defines the function of the program. Possible values are:

- LIST: List names and sizes of all code procedures in the library on the logfile.
- ADD: Insert the codeprocedure from the file described in parameter PROC, in the library. If the library does not exist, it will be created.
  If one or more codeprocedures with the same name as the new one already exist in the libaray, they will be deleted.
- DEL: Delete the codeprocedure defined by parameter PROC from the library.
- EXT: Place the codeprocedure described by parameter PROC in a disc file with the same name. The library remains untrouched.

PROC: Name of file to contain the codeprocedure.

Default: LIBE LIB.<0>...<0> OUT.$TTY FUNC.LIST PROC.<0>...<0>

Examples:

LIBE ULIB $TTY ADD P0260

The codeprocedure contained in file P0260 is inserted in library ULIB. A log is output to file $TTY. If a codeprocedure with the title P0260 already exists in the library, it will be deleted.

6.24 LINK 6.24

Format

LINK OUT.<name> LOG.<name> TITLE.<name> ENTRY.<number>
CHECK.<boolean> FORM.<name> MODE.<name> IN.<name> ...

Function

The purpose of a linkage editor is to handle external references between relocatable binary modules. The DOMUS linkage editor (LINK) takes a number of relocatable binary input modules produced by the DOMAC assembler and outputs a single program in absolute or relocatable binary format, having filled in intermodule references and relocated approprately. The input modules are taken from disc files. When linking, a work file on disc named .INK is used. This file is removed automatically when the linking has finished.
The program is able to work on main catalogs only.
For further information, please consult the manual DOMUS Linkage Editor.

Parameters:

OUT: Name of file or entry to which the binary code is output. Must be specified.

LOG: Name of file or entry to which the log information is output.

TITLE: Title of the output module. Only relevant when parameter FORM is R or P.

ENTRY: Specifying the maximal number of entries defined in the input modules.

CHECK: If check of location overwrite is wanted, YES must be specified, otherwise NO.

FORM: Specifies the format of the binary output. Only one of the following letters must be specified: R (relocatable binary), A (absolute binary), B (Basic system), N (as B, but the output has no start address) or P (paged program).

MODE: One or more of the following, letters may be specified: S (Only the first module from each input file is read), M (the input files may contain more than

one module), X (a size block will be added to the binary output. Only relevant with relocatable binary output).

IN: Name of input file. Up to 50 input files may be specified, and they are linked in the order given.

Default: LINK OUT.<0>...<0> LOG.<0>...<0> TITLE.MAIN ENTRY.225 CHECK.YES FORM.R MODE.S IN.<0>...<0> <0>...<0> ...

Example:

LINK ABSBI $TTY FORM.A IN.TEXT1 TEXT2

Link the modules from TEXT1 and TEXT2 and output absolute binary of file ABSBI. The log will be output to $TTY.

6.25 NAMEX 6.25

Format:

NAMEX FUNC.<name> NAME.<name> TXT1.<text> TXT2.<text>
FILE.<name> LIST.<name>

FUNCTION:

This program maintains a filename explanation file, containing user supplied information about files on a DOMUS disc pack.
The filename explanation file is a set of records consisting of two fields. The first field is an ident, normally the name of a DOMUS file. The second field is a user defined text related to the ident. The text could describe the contents of the file identified by the ident.
The program contains ordinary editing functions, such as insert, change and delete. The program can also add a text to an existing record, print specified records an merge two filename explanation files.
The program operates on main catalogs only.
For further information, please consult the manual:
DOMUS Utility NAMEX, User's Guide.

Parameters:

FUNC: The function of the program. Allowed values are: I (insert), A (add), C (change), D (delete), P (print) and M (merge).

MAME: Identifies the ident on which the function should be performed, with two exceptions:
If function is P, the ident is a mask, where $ may substitute any character. In this case the default value is $$$$$.
If function is M, the ident is the name of a filename explanation file, from which filename explanations are added to the filename explanation file given by parameter FILE.

TXT1: User explanation to be associated with the ident. Only significant, when the function is I, A or C. The maximum length is 120 characters.

TXT2: May be defined, if TXT1 if specified. The text in this parameter will be added to TXT1. Maximum length is 120 characters.

FILE: Name of the filename explanation file, on which the operation is to be performed.

LIST: Name of file where to output the listing when the function is P.

Default: NAMEX FUNC.<0>...<0> NAME.<0>...<0> TXT1.<0>...<0> TXT2.<0 ...<0> FILE.SYSEX LIST.$TTY

Example:

NAMEX I DATA1 'THIS IS A DATA FILE' FILE.MYEXP

The text 'THIS IS A DATA FILE' is inserted in the filename explanation file MYEXP as belonging to the ident DATA1.

6.26 NEWCAT 6.26

Format:

NEWCAT UNIT.<number> CATSIZE.<number> SLICE.<number>
SEG.<number>

Function:

A new and empty catalog is created on the spefied disc unit. It is not possible to create a catalog on unit 0, as this unit is the system disc at running time.
Note: All files on selected unit are deleted independent of any protection.

Parameters:

UNIT: Specifies the unit on which to create a new catalog. Must be inside the range 1 - 15.

CATSIZE: Specifies the size of the catalog file in segments. This size must be an integral multiple of the slice size.

SLICE: Specifies the slice size in segments of the new catalog.

SEG: Specifies the total number of segments on the new unit.

Default: No default values, all parameters must be specified.

Example:

NEWCAT 1 24 6 4872

All files on catalog unit 1 are removed, and a new and empty catalog is created. The catalog size is 24 segments, and the slice size is 6 segments. Total number of segments is 4872, i.e. a 2.4 MB disc is used.

Requirements:

Catalog initialization process CATI must be loaded.

6.27 PRINT

Format:

PRINT IN.<filename> LINE.<boolean> OUT.<filename>

Function:

An ASCII text file is output to a printer. The character 'TAB' (ASCII value 9) is converted to the corresponding number of spaces. If wanted, linenumbers eqiuvalent to numbers printed by the MUSIL compiler may be output in front of each line. If the process TIME is loaded, date and time will be included in the headline.
It is recommended to use the DOMUS utility COPY for data transfer to non-printer files.

Parameters:

IN: Name of ASCII text file to be printed.

LINE: If YES is specified, the output is supplied with linenumbers in front of each line.

OUT: Name of output file (a device descriptor describing a printer).

Default: PRINT IN.$PTR LINE.NO OUT.$LPT

Example:

PRINT TEXT YES $SP

The file TEXT is output to device $SP with linenumbers.

Requirements:

Depending on printer drum, standard conversion can be used by connecting a conversion table to the printer driver by DOMUS utility STACO.

6.28 PUNCH 6.28

Format:

PUNCH IN.<filename> MODE.<name> PNO.<number>

Function:

The inputfile is output to the paper tape punch. The paper tape is punched with either no, even or odd parity.

Parameters:

IN: Name of file to be punched.

MODE: Defines the parity. Only the first character of the parameter is checked. Allowed values are: N (no parity), A (ACSII parity), E (even parity) and O (odd parity). If any other character is specified, no parity punch is performed. The modes A and E are equal.

PNO: Unit number of paper tape punch.

Default: PUNCH IN.<0>...<0> MODE.E PNO.0

Example:

FUNCH PIP A 1

File PIP is printed on PTP1 with even parity.

6.29 REMOVE

Format:

REMOVE MASK.<filename> LIST.<filename> VERIFY.<boolean>

Function:

Deletes all selected non-permanent discfiles in a specified catalog. Optionally each selected discfile must be verified before deletion. A list of all selected discfiles is produced. Each filename is followed by either 'deleted' or 'not deleted'.

Parameters:

MASK: The catalog part of this parameter specifies the catalog to be used. The name part is a mask of filenames to be deleted, where the character $ replaces any character. Only non-permanent files matching the mask can be deleted.

LIST: Name of file on which to output a list of selected filenames.

VERIFY: Specifies whether or not each matching filename should be output on the console and verified before deletion.
If the first character of the answer is 'Y', the file is deleted, otherwise it is not deleted.

Default: REMOVE MASK.<0>...<0> LIST.$TTY VERIFY.YES

Example:

REMOVE A$$$$ VERIFY.NO

All non-permanent files on the main catalog with 'A' as the first character of the name will be deleted unconditionally.

6.30 RENAME 6.30

Format:

RENAME OLD.<filename> NEW.<name>

Function:

This utility changes the name of a catalog entry into another.

Parameters:

OLD: Name of entry to be changed.

NEW New name of the entry. This entry is always placed in the same catalog as the old entry.

Default: RENAME OLD.<0>...<0> NEW.<0>...<0>

Examples:

RENAME WORK SAVE

The file WORK is renamed SAVE. The name WORK is removed from the catalog.

RENAME PIP:1 PAP

The file PIP on unit 1 is renamed PAP. No files on unit 0 are changed.

6.31 SET

Format:

SET NAME.<filename> DEVICE.<name> FILE.<number> BLOCK.<number>
MODE.<number> KIND.<number> MASK.<number>

Function:

This utility creates a new device descriptor or updates an existing device descriptor according to the values given of the call. Please see chapter 4.1 about the use of this utility.

Parameters:

NAME: Name of device descriptor to create or change.

DEVICE: Name of the equivalent document name. i.e. the driver name.

FILE: The filenumber of the document.

BLOCK: The blocknumber of the document.

MODE: The mode to be used.

KIND: The kind of the document.

MASK: The giveup mask to be used.

Default: SET NAME.<0>...<0> DEVICE.<0>...<0> FILE.1 BLOCK.1
MODE.1 KIND.1 MASK.8'163777

Note: These default values are only used when creating a new device descriptor. If the descriptor exists, the default values are taken from this entry.

Examples:

SET $PTP PTP MODE.11

Device descriptor $PTP describes the document punched tape with even parity.

SET $MT0 MT0 4 1 1 2'1110

Device descriptor $MT0 describes file 4 on MT0. Kind is set to repeatable, positionable and blocked.

Definition of the task to copy from MT1, file 2 to MT0, file 10 can be done in this way:

```
SET MTOUT MT0 10 1 3 2'1110
FINIS SET
SET MTIN MT1 2 1 1 2'1110
FINIS SET
COPY MTIN MTOUT
FINIS COPY
```

6.32 STACO

Format:

STACO <name> <name>

Function:

The program inserts a conversion table as standard conversion to a driver. Both the driver and the table must be loaded. The owner of the coreitem containing the table is changed to be the driver. Thus the table may only be removed by killing the driver.

Parameters:

This utility does not use the standard format. Two names has to be given: The first is the process name of the driver, and the second is the name of the coreitem containing the table. There are no default values.

Example:

```
LOAD LPT TAB1
STACO LPT TAB1
FINIS STACO
```

The table found in TAB1 is inserted as standard conversion for the LPT driver.

6.33 SUBCAT

Format:

SUBCAT FUNC.<name> NAME.<name> KEY.<number>

Function:

The utility subcat is able to create, link, delete and unlink subcatalogs. For further information, please consult the maunual DOMUS Utility SUBCAT, User's Guide.

Parameters:

FUNC: A name defining the function of the program. Possible values are: INIT, CREATE, LINK, CRELI (create and link), DELET and UNLINK.

NAME: Name of subcatalog in question.

KEY: An integer specifying the protection key of the subcatalog.

Default: SUBCAT FUNC.INIT NAME.<0>...<0> KEY.0

Example:

SUBCAT CRELI LIB 123

The subcatalog LIB is created on unit 0 and linked with the key 123.

6.34 TYPE

Format:

TYPE IN.<filename> LINE.<boolean>

Function:

A selected text file is output to the operator console.
The character 'TAB' (ASCII value 9) is converted to the corresponding number of spaces.
Optionally linenumbers are output in front of each line.
If the ESCAPE key is pressed during run of TYPE, the output will be terminated at once, and the program will be removed.
If the process TIME is loaded, date and time will be output as a headline.

Parameters:

IN: Name of file to be output.

LINE: If this parameter is specified as YES, linenumbers will be output in front of each line.

Default: TYPE IN.$PTR LINE.NO

Example:

TYPE TEXT YES

The file TEXT is output to the operator console with linenumbers.

6.35 XREF 6.35

Format:

XREF IN.<filename> OUT.<filename>

Function:

This program produces a cross-reference listing of all constants, types, variables, procedures and labels in a MUSIL source text. A sorted list of all symbols declared is produced, with the numbers of the lines in which they appear.
If process TIME is loaded, time and date is included in the headline.

Parameters:

IN: Name of text file containing the MUSIL source text.
OUT: Name of file to which the listing is output.
Default: XREF IN.$PTR OUT.$LPT

Example:

XREF TEXT
A cross-reference listing of the MUSIL source text in file TEXT is output to $LPT.

Requirements:

This program will use a coreitem XREFC for internal sort.

A. SURVEY OF STANDARD SYSTEM MESSAGES A.

0001 *** SYNTAX
0002 *** TOO MANY PARENTHESES
0003 *** PARAM
0004 *** END MEDIUM, FILE
0005 *** TOO MANY COMMANDS
0006 *** STATUS, FILE
0007 *** UNKNOWN, FILE
0008 *** RESERVATION, FILE
0009 *** COREITEM EXISTS, ITEM
0010 *** SIZE
0011 *** COREITEM DOES NOT EXIST, ITEM
0012 *** COREITEM NOT CLEARED, ITEM
0013 *** ENTRY NOT A FILE, ENTRY
0014 *** STATUS, DEVICE
0015 *** NOT ALLOWED
0016 *** NO SPACE FOR PAGES, FILE
0017 *** ILLEGAL PROGRAM, FILE
0018 *** SIZE ERROR, FILE
0019 *** CHECKSUM ERROR, FILE
0020 *** VIRTUAL ADDRESS ERROR, FILE
0021 *** PROCESS DOES NOT EXIST, PROCESS
0023 *** PROCESS EXISTS, PROCESS
0024 *** UNKNOWN, SUBCATALOG

0100 *** CATALOG I/O ERROR, FILE
0101 *** FILE DOES NOT EXIST, FILE
0106 *** ILLEGAL OPERATION, FILE
0107 *** NOT ENOUGH DISK SPACE, FILE
0111 *** FILE DOES ALREADY EXIST, FILE
0112 *** INDEX BLOCK FULL, FILE

0120 *** CATALOG I/O ERROR, FILE
0121 *** FILE DOES NOT EXIST, FILE
0126 *** FILE IN USE, FILE
0127 *** NO FREE AREA PROCESS TO FILE
0131 *** END MEDIUM ON FILE
0132 *** MAP/FILE EXCEEDED, FILE

0140 *** LINK ALREADY EXISTS TO SUBCATALOG
0141 *** NO FREE ENTRIES IN SYSSC
0142 *** LINK DOES NOT EXIST TO SUBCATALOG
0143 *** WRONG KEY, SUBCATALOG

0200 *** NOT ENOUTH ARGUMENTS
0201 *** UNIT NUMBER CONFLICT
0202 *** ILLEGAL FILE SPECIFICATION

0203 *** COMMUNICATION ERROR WITH S
0204 *** SHORT OF CORE STORAGE
0205 *** NON-ASCII CHARACTER IN XREF-INPUT

0206 *** UNITNUMBER GREATER THAN 255
0207 *** UNIT NOT MOUNTED
0208 *** UNIT DOES NOT EXIST

0209 *** ILLEGAL UNITNUMBER
0210 *** ILLEGAL DISC SIZE
0211 *** NOT DEVICE DESCRIPTOR, ENTRY
0212 *** FILE OR BLOCK TOO LARGE

0250 *** CHECKSUM ERROR, FILE
0251 *** OVERFLOW IN ENTRY TABLE
0252 *** FATAL ERROR, LINKAGE EDITOR
0253 *** WARNING, LINKAGE EDITOR

0260 *** ILLEGAL BLOCK TYPE, FILE
0261 *** CHECKSUM ERROR, FILE
0262 *** SYNTAX ERROR, FILE
0263 *** PROGRAM ERROR

0265 *** STACK OVERFLOW

0270 *** INTERNAL ERROR:
0271 *** DOMAC BREAK, NO:
0272 *** INSUFFICIENT CORE
0273 *** PARAMETER ERROR
0274 *** VIRTUAL CORE ERROR

2000 *** DISCONNECTED, FILE
2001 *** OFF-LINE, FILE
2002 *** BUSY, FILE
2003 *** DEVICE BIT 1, FILE
2004 *** DEVICE BIT 2, FILE
2005 *** DEVICE BIT 3, FILE
2006 *** RESERVED, FILE
2007 *** END OF FILE, FILE
2008 *** BLOCK LENGTH ERROR, FILE
2009 *** DATA LATE, FILE
2010 *** PARITY ERROR, FILE
2011 *** END MEDIUM, FILE
2012 *** POSITION ERROR, FILE
2013 *** DRIVER MISSING, FILE
2014 *** TIMEOUT FILE
2015 *** DATA FORMAT ERROR, FILE

2026 *** PUNCH RESERVED
2031 *** PAPER LOW ON PUNCH
2033 *** PUNCH DRIVER NOT LOEDED
2034 *** PUNCH ERROR OR TIMEOUT

2041 *** STATION OFF-LINE, STATION
2042 *** TAPE REWINDING, STATION
2043 *** NOISE RECORD, STATION
2045 *** WRITE LOCK, STATION
2046 *** ILLEGAL OPERATION, STATION
2047 *** END OF FILE, STATION
2048 *** BLOCK LENGTH ERROR, STATION
2049 *** DATA LATE, STATION
2050 *** PARITY ERROR, STATION
2051 *** END OF TAPE, STATION
2052 *** POSITION ERROR, STATION
2053 *** DRIVER MISSING, STATION
2054 *** TIMEOUT ERROR, STATION

Device descriptor name	Driver name	File number	Block number	Mode	Kind	Mask	Use
$CDR	CDR	irrelevant	irrelevant	9	2'10	8'161777	Input of decimal cards with termination
$CDRN	CDR	irrelevant	irrelevant	1	2'10	8'161777	Input of cards, binary bytes
$CPT	CPT	irrelevant	irrelevant	3	1	8'161777	Unformatted output on charaband printer
$CT0	CT0	1	1	1	2'1110	8'161777	Cassette tape input/output
$LPT	LPT	irrelevant	irrelevant	3	1	8'161777	Unformatted output on line printer
$MT0	MT0	1	1	1	2'1110	8'161777	Magnetic tape input/output
$PTP	PTP	irrelevant	irrelevant	11	1	8'1027	Output on paper tape, even parity
$PTPN	PTP	irrelevant	irrelevant	3	1	8'1027	Output on paper tape, no parity
$PTR	PTR	irrelevant	irrelevant	9	1	8'1067	Input of paper tape, even parity
$PTRN	PTR	irrelevant	irrelevant	1	1	8'1067	Input of paper tape, no parity
$SP	SP	irrelevant	irrelevant	1	1	8'161777	Unformatted output on serial printer
$TTY	TTY	irrelevant	irrelevant	1	1	0	Operator console, input/output

C. SURVEY OF UTILITY PROGRAM CALLS WITH DEFAULT VALUES C.

```
ADDEX     COM.<0> EXP.SYSEX
AMXINIT   IN.$PTR
APPEND    OUT.<0> IN.<0> <0> ...
CATLIST   MASK.$$$$$ OUT.$TTY TEXT.<0> ATT.<0> COMPR.NO
CHATR     NAME.<0> ATT.V
COMP      IN.$PTR OUT.<0> LIST.<0> INCOD.<0> NAME.MAIN IDENT.<0>
               MODIF.<0> BLOCK.512
COMPARE   IN1.<0> IN2.<0> MAX.<0>
CONFI     LIST.$TTY
COPY      IN.<0> OUT.<0> BLOCK.512
CREATE    NAME.<0> SIZE.1 ATT.V
DCOPY     FUNC.BACKU UNITA.0 UNITB.1
DELETE    NAME.<0> <0> <0> ...
DISK      UNIT.0
DOMAC     MODE.<0> LIST.<0> BIN.<0> LINES.60 PERM.DOMPS SYMB.DOMST
               MACRO.DOMMC XREF.DOMXF
DUAL      BOOT.FPABT SYS.Q3600
EDIT      <0>
EXEC      IN.$PTR LIST.<0> STOP.NO CONT.<0>
FCOPY     FUNC.DUMP MASK.$$$$$ LIST.$LPT FILE.1 DEV.MT0 UPDAT.NO
FLCOPY    FROM.0 TO.1 CYL.77
FLFORMA   UNIT.999 LENGTH.128 SIDE.S DENS.S SCREW.NO BAD1.0 BAD2.0 ERMAP.A
GEN       OUT.MT OCOPY.1 CONTR.<0> LOG.<0> LCOPY.1 MARG.0
GENER     IN.STERR LIST.$LPT OUT.SSYSE
LIBE      LIB.<0> OUT.$TTY FUNC.LIST PROC.<0>
LINK      OUT.<0> LOG<0> TITLE.MAIN ENTRY.255 CHECK.YES FORM.R
          MODE.S IN.<0> <0> ...
NAMEX     FUNC.<0> NAME.<0> TXT1.<0> TXT2 <0> FILE.SYSEX LIST.$TTY
NEWCAT    UNIT.<number> CATSIZE.<number> SLICE.<number>
          SEG.<number> (no default values)
PRINT     IN.$PTR LINE.NO OUT.$LPT
PUNCH     IN.<0> MODE.E PNO.0
REMOVE    MASK.<0> LIST.$TTY VERIFY.YES
RENAME    OLD.<0> NEW.<0>
SET       NAME.0 DEVICE.<0> FILE.1 BLOCK.1 MODE.1 KIND.1
          MASK.8'161777
STACO     <0> <0>
SUBCAT    FUNC.INIT NAME.<0> KEY.0
TYPE      IN.$PTR LINE.NO
XREF      IN.$PTR OUT.$LPT
```

D. REFERENCE LIST

D.

DOMUS User's Guide, Part 1
This manual describes the disc operating system for the RC3600 line of computers.

DOMUS Utility ADDEX, User's Guide
This manual is a User's Guide describing how to use the DOMUS Utility program ADDEX.

AMXINIT, RC3682 AMX driver initialization program, User's Guide.
This program describes how to use the program AMXINIT for initialization of a RC3682 AMX driver.

MUSIL Compiler, Operators Guide
This manual describes the parameters to the MUSIL compiler.

Introduction to DOMAC Assembler
This manual contains a short introduction to the RC3600 assembler language, a description of how to invoke the DOMAC assembler, and a list of possible errors from the DOMAC assembler.

DOMUS Utility DUAL, User's Guide
This manual describes how to autoload the second processor of a dual processor system.

RC3600 Text Editor
This manual describes the use of the text editor for creating, modifying and updating text files.

DOMUS Utility EXEC, User's Guide
This manual is a User's guide for the DOMUS batch processor EXEC.

RC3600 System Generation with DOMUS GEN, User's Guide
This manual describes the use of the DOMUS Utility GEN to produce program magnetic tapes, Flexible discs, card checks and paper tapes.

DOMUS Linkage Editor
This manual describes the linkage editor for the disc operating system DOMUS.

DOMUS Utility NAMEX, User's Guide
This manual is a User's Guide describing how to use the DOMUS Utility Program NAMEX.

DOMUS Utility SUBCAT, User's Guide
This manual is a user's guide for the DOMUS utility program SUBCAT used for maintenance of subcatalogs.

DOMAC, DOMUS Macro Assembler, User's Guide
This manual describes the RC3600 Macro Assembler language and operation of the DOMAC Macro Assembler.

RETURN LETTER

Title: DOMUS User's Guide Part 2 RCSL No.: 43-GL10166

A/S Regnecentralen af 1979/RC Computer A/S maintains a continual effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback, your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability, and readability:

Do you find errors in this manual? If so, specify by page.

How can this manual be improved?

Other comments?

Name: _____ Title: _____

Company: _____

Address: _____

Date: _____

Thank you

42-i 1288

..... Fold here .....

..... Do not tear - Fold here and staple .....

Affix
postage
here

REGNECENTRALEN
af 1979
Information Department
Lautrupbjerg 1
DK-2750 Ballerup
Denmark