RCSL : 43-GL 1349
Authors : APR/PO
Edited : October, 1975.

[redacted]

ORIGINALMANUAL FINDES HOS HELGA



# MUSIL COMPILER
## Operators Guide

Keywords : RC 3600 Musil Compiler

Abstract : Compiler Guide, Operators Guide.

The MUSIL COMPILER RCSL 43-GL1855 has a new input/output structure for loading program sources and code procedures and for outputting object code. This structure makes it possible to use various peripheral devices. It has become possible to compile coroutine programs (modification C), to order one extra buffer per zone (modification B), and to get the object code without a process descriptor (modification N).

## Input/Output Devices

When the compiler is ready to start a compilation, the text MUSIL READY is written on the operator device. Now a command followed by a parameter can be typed on the operator device. This is done e.g. to select the files one wants to use as input, output, or list files, or it may be the name of the object module. All the compiler commands have a standard value for the parameter. When a peripheral file is selected, the parameter contains its driver name with a $ sign in front. Further, if the device is file-oriented, the file No. should follow as :<fileno> after the driver name, e.g. file No. 7 on tape unit 0 must be types as $MT0:7. The compiler checks that the driver is loaded. If the selected device is a disc, the name of the disc area must be used, but without a $ sign in front. In this case the disc driver and the catalog system must be loaded. The compiler initiates the selected disc kit, using the old disc catalog (use e.g. the editor to initiate a new disc) and updates this catalog when the compilation is finished. A disc output file is created if it does not exist. An input file must exist. A file on a disc unit different from unit 0 must contain a :<unitno> after the file name, e.g. the disc file LIBRY on unit 1 must be typed as LIBRY:1.

In version 3 of the compiler the possible devices are PTR, PTP, TTY, MT0, MT1, CDR, RDP, CT0, CT1, FD0, FD1, LPT, CPT.

Notice that the CDR driver should be CR002 and the RDP driver should be RP001 or newer versions. If no object code or listing is wanted, <nl> may be typed after the OUT or the LIST command.

## Compiler Commands

| | | | |
|---|---|---|---|
| Input source | IN | <name> | The standard name is $PTR |
| Output | OUT | <name> | - - - - $PTP |
| Listing | LIST | <name> | - - - - no device |
| Input code procedures | INCOD | <name> | - - - - no device |

All the code procedure must be loaded from the device or file specified by <name>, e.g. one may have all the code procedure needed or more in a library disc or magtape file. When a code procedure is loaded, its name is written on the operator device.

Error messages during the loading:

| | |
|---|---|
| STATUS EM | a code procedure is missing |
| INCOD ERROR | no load device is specified |
| END ERROR | a start block is missing |
| SUM ERROR | sumcheck error |
| ILL ERROR | inconsistent block |

| | | | |
|---|---|---|---|
| Process name | NAME | <name> | The standard process name of the object code is MAIN. Only the first 5 characters of name are used. |
| Identification | IDENT | <ident> | The standard ident is blank. Only the first 5 characters in the ident are used. The ident is output as an ascii text in front of the object code. |
| Operators device | OPCOM | <device> | The standard device is TTY. The name, max. 5 characters, is the driver name of the object code's operator device without a $-sign in front. |

Compiler Commands (cont'd)

| | | | |
|---|---|---|---|
| Modification | MODIF | <letters> | The standard modification is empty. A C in the modification letters means that the source program is a coroutine program. A B means that one buffer more per file is wanted. The modification B is included in C. An N means that the object code must be without a process descriptor. |
| Display | DISP | | The command will display the contents of the other commands. |
| Initialization | INIT | | The command will set the contents of the other commands to the standard values. |
| Start of compilation | START | | The command will start the compilation, using the specified parameters. |

An example of a set of compiler commands:

| | | | |
|---|---|---|---|
| IN | TEST | ; | source input from disc unit 0 file TEST |
| OUT | $MT0:7 | ; | object code output on MT0 file 7 |
| LIST | $LPT | ; | listing on LPT |
| INCOD | $PTR | ; | code procedure input from PTR |
| MODIF | B | ; | one extra buffer per file |
| NAME | TEST | ; | name of object code |
| IDENT | VER07 | ; | ident of object code |
| START | | ; | starts the compilation with the given parameters |

## Check of Commands

If an error is made in giving commands, a message is printed on the operator device.

The messages are:

| | | |
|---|---|---|
| ILLEGAL COMMAND | , | the command does not exist |
| DRIVER MISSING | , | the driver or disc catalog system is not loaded |
| ILLEGAL DEVICE | , | the device cannot be used for input or output |
| NO INPUT FILE | , | the input disc file does not exist |
| FILE NO MISSING | , | no file number is given for a file-oriented device |
| ILLEGAL MODIF ITEM | , | other letters than B, C, or N are used |

## COPY

A new compiler directive has been introduced which makes it possible to insert a piece of program source at a given place in the normal program source. This copy source must come from a file different from the device specified by the IN-command. If the directive $COPY name, where name specifies a load device or file with the same syntax as used above, is put into the program source, the compiler will insert the source read from the device or file until a $END is met with. The compiler then returns to the normal input source and continues with this source. It is possible to have one level of copy source and to have one conversion table called CCTAB for these sources. The directives $COPY name and $END must be terminated by a new line character.

## Conversion

It is possible to use conversion tables for the list devices and the input devices. These conversion tables are written as MUSIL programs. The name of the conversion table for the list device is CLTAB, for the normal input device it is called CTTAB, and for the copy sources it is called CCTAB. The format of the conversion table programs is

```
const
table = # XXX
          X
          XXX

        #;
.....
begin
  .....
end ;
```

Listing

When a compilation starts the text MUSIL COMPILER VERSION 3 is written on the list device if you want a listing. After this text the compiler outputs the name and the ident of the object modules. When the compilation is finished, the compiler writes:

END
size XXXXX

where XXXXX is the decimal size (bytes) of the compiled program. If no output is wanted, *** before size indicates that possible code procedures are not included in size.

In the case of errors the text ERRORS : XXXXX is written on the listing, and the object output is skipped.

Status Errors and Break Situations

If a status error occurs on a device in use, the compiler writes

DEVICENAME STATE : statusword REP?

If the answer is NO, the compilation is skipped and the compiler is ready for the next compilation. On all other answers than NO the compiler performs a repeat-share. Note that the compiler does not update the discs.

The error message COREOVERFLOW means that more core is needed for the compilation.

## Error Messages

MUSIL provides the programmer with a variety of error messages, indicated by error numbers on the compilation printout. The significance of those error numbers is as follows:

020202 Number overflow, a numeric constant exceeds 65535, or 16 bits.
020301 Illegal character in input.
030102 < appearing within a string is not followed by a small numeric literal and >.
040105 Name conflict in Constant Section.
040205 Name conflict in Type Section.
040302 Syntax in Type Section, no = following an ident.
040405 Name conflict in Variable Section.
040602 Procedure head not followed by ;
050102 Type is no identifier.
050202 ( is missing.
050203 Length undefined for string.
050502 ) is missing after string.
050604 Undefined type identifier. Note that no forward declarations are allowed.
050702 Improper termination of file specification.
051002 Field of structured type or too long.
051102 Incorrect use of FROM or integer field starting at odd byte address.
051205 Name conflict in GIVEUP procedure.
051304 Conversion table undeclared.
051406 Conversion table type error.
060206 Multiple defined label.
060302 Variable is no identifier. Or multiple.
060402 . is not followed by identifier or by undeclared field.
060504 Identifier undeclared.
060606 Type error with BYTE or WORD.
060702 Relational operator missing.
061002 Procedure statement with missing ).
061102 Type error in procedure parameter.
061306 Illegal number of parameters.
061406 Type error with operator.
061506 Overflow of work registers. Expression too complex.

Error Messages which cause skipping of program parts:

000040 Syntax in section delimiter.
000041 Syntax in constant section.
000043 Type specification incorrectly terminated.
000044 Variable declaration incorrect.
000045 Variable declaration incorrectly terminated.
000046 Procedure heading incorrect.
000047 Incorrect parameter format.
000051 Syntax in field list.
000052 Syntax in file declaration.
000063 Incomprehensible statement.
000064 Incorrect label declaration.
000065 Incomprehensible expression.

## Extensions of the MUSIL COMPILER

The new MUSIL COMPILER RCSL 43-GL 1855 contains the following changes and extensions:

1. A new input/output structure makes it possible to use various peripheral devices (e.g. discs) for loading program sources and for outputting object code. See MUSIL COMPILER OPERATORS GUIDE RCSL 43-GL 1349.

2. ELSE
   The statement if B then S1 [else S2 ]$^{1}$ is now legal.

3. Integer Field.
   In variables of the type record integer fields may by used, starting at even byte addresses.

4. Loadname of code procedure.
   It has become possible to specify a loadname for a code procedure.

   <procedure heading> :: = procedure <identifier> ;
   codebody [<loadname>]$^{1}$ ;

   If no <loadname> is specified, the <identifier> is used as loadname.

   Ex.

   ```
   procedure getdate (var date : string (1)) ;
   codebody P0003 ;
   ```

   In the MUSIL program the <identifier> of the procedure is getdate, but the title of the code procedure to be loaded is P0003.