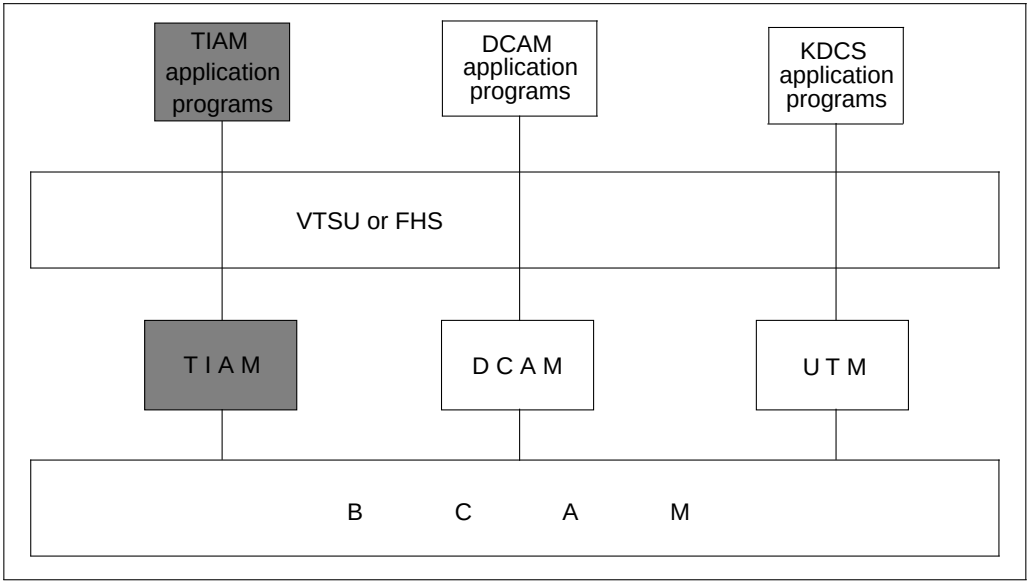
1 Preface

TIAM (Terminal Interactive Access Method) is a BS2000 access method that lets users work with the computer interactively as if they were the only user. All the services and the command language offered by the BS2000 operating system can be used, together with the BS2000 utility routines, and it is possible to work with all application programs, provided the appropriate authorization is available. The user must load the program which is to be to worked on, and must initiate, control and terminate the task. Before work with TIAM can be started, a logical connection must be set up to the communication application $DIALOG which is provided by the system (see also the manual "Network Access for Terminals" [8]).

The main tasks performed using this type of interaction include program development and testing, as well as editing files.



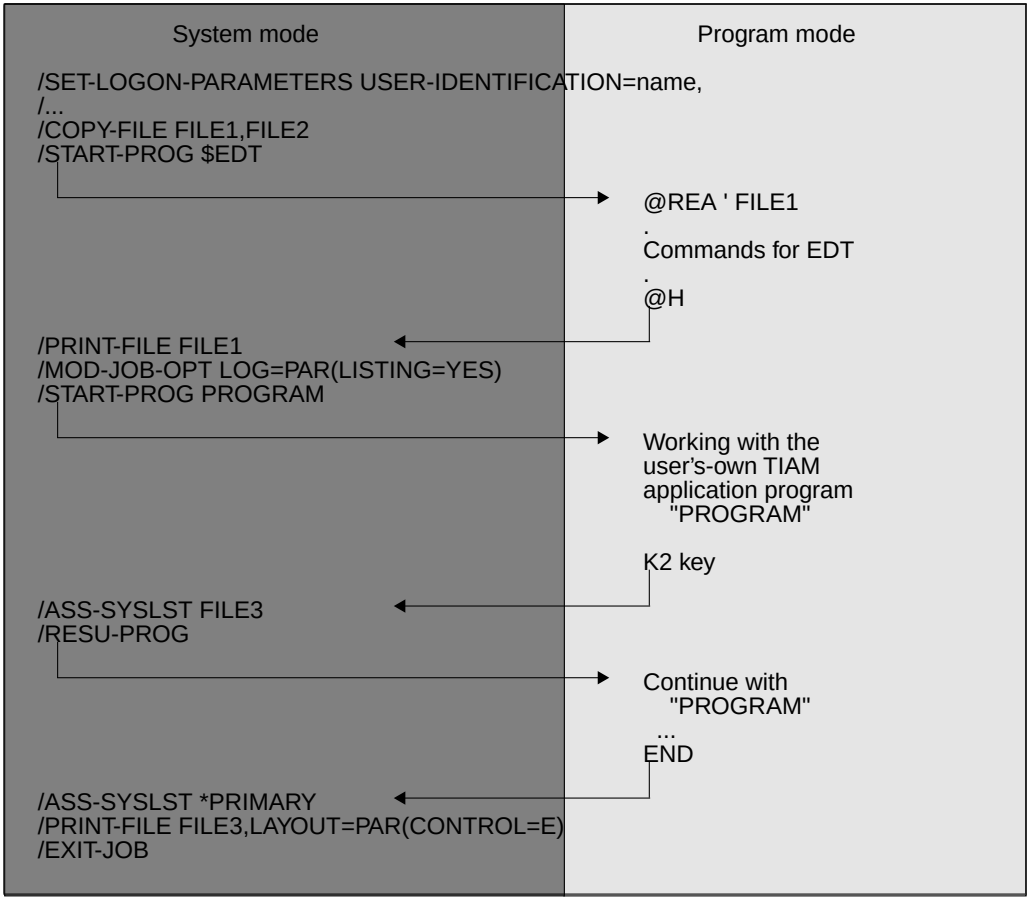
TIAM in the system environment

There are two ways of working with TIAM: in system mode and in program mode.

In **system mode** the user works directly with the operating system and can make use of the **TIAM commands** and the BS2000 command language. In this mode, TIAM provides the MODIFY-TERMINAL-OPTIONS command (and its ISP equivalent, TCHNG); this enables the user to control the task and to define the attributes of the display terminal. In addition, the SHOW-TERMINAL-OPTIONS command is available, and can be used to request that the display terminal attributes are output. The comand interfaces are defined in this manual.

In **program mode** the user works with a program called in system mode by means of a START-PROGRAM or LOAD-PROGRAM command. This program may be, for example, a utility routine or a user-written program. For writing these programs, TIAM offers macros (for the ASSEMBLER programmer) and COBOL, PL/I, C and FORTRAN calls.

It is possible to switch from program mode to system mode, and vice versa, at any time.



Switching from system to program mode and vice versa

1.1 Target group

This manual is written for BS2000 users and programmers working in BS2000 with the TIAM access method.

A basic knowledge of BS2000 is required for an understanding of this manual, as well as a knowledge of the programming languages used (Assembler, COBOL, FORTRAN, PL/I and C).

1.2 Summary of contents

This manual deals with the commands, macros and COBOL calls supported by the TIAM access method.

In the chapter **'TIAM commands'**, the commands supported by TIAM are described in SDF format. They are arranged in alphabetical order.

The chapter **'TIAM macros'** describes the TIAM macros, which are used to control input/output. These macros are relevant to ASSEMBLER programmers, and are arranged in alphabetical order.

The language-specific chapters which follow - **'COBOL interface'**, **'FORTRAN interface', 'PL/1 interface'** und **'C interface' -** describe the data structures for the TIAM calls, as well as the TIAM calls themselves, for the various programming interfaces. The TIAM calls allow the user to control input/output.

The **'POSIX interfaces'** chapter describes the support provided by TIAM for the POSIX interface.

The **Appendix** brings together a number of important tables and diagrams.

1.3 Changes since the last version of the manual

VTSU

The topic of VTSU is covered by a separate manual (see [1] in the Related publications chapter). Topics relating specifically to VTSU have therefore been removed from this manual. The items concerned are

– the VTSU macros DCSTA, VTCSET, VTSUCB

– the data structures LINE-MODE-CONTROL-CHARACTERS and VTSU-CONTROL-BLOCK for the programming interfaces

– the chapter on “Setting operating parameters”

– the PLUS program for loading and saving program keys

Command: MODIFY-TERMINAL-OPTIONS

In the TIAM command `MODIFY-TERMINAL-OPTIONS`, the parameter *name 1.. 8* has been added for the `CODED-CHARACTER-SET`. This parameter allows a character set to be assigned to a display terminal.

Command: SHOW-TERMINAL-OPTIONS

The command `SHOW-TERMINAL-OPTIONS` is new. It can be used to request the output of terminal attribute details.

Macro: TSTAT

The *length* operand of the `TSTAT` macro has new values.

TIAM C interface

The sample program in the chapter entitled “C interface” has been revised.

POSIX interface

As from Version V11.2A, TIAM supports the POSIX interface for input and output of data. This is described in the chapter entitled “POSIX interfaces”.

1.4 Readme file

Information on functional changes and additions to the current product version described in this manual can be found in the product-specific README file. This README file will be found on the BS2000 computer under the file name SYSRME.product.version.language. The user ID under which the README file is cataloged can be obtained from local system support staff. The README file can be read with the /SHOW-FILE command or an editor, or can be prined out on a standard printer using the following command:

```
/PRINT-FILE FILE-NAME=fileiname,LAYOUT-CONTROL=
    PARAMETERS(CONTROL-CHARACTERS=EBCDIC)
```

as from SPOOL V3.0A:

```
/PRINT-DOCUMENT FROM-FILE=filename, DOCUMENT-FORMAT=*PAGE-FORMAT(
    CONTROL-MODE=*PAGE-MODE(LINE-SPACING=*BY-EBCDIC-CONTROL))
```

2 TIAM commands

The commands are given in alphabetical order. Examples illustrating the use of the commands have been included after the operand descriptions.

The chapter begins with a description of the notational conventions applied in describing the SDF commands. Full details about the use of SDF commands will be found in the manual entitled "SDF Introduction to the dialog interface" [5].

2.1 SDF notational conventions

The example below illustrates the notational conventions used in manuals for defining commands. The format of each command starts with an entry for the command name; this is followed by a list of all the operands and the permissible operand values. Operand values which introduce a structure, and their dependent operands, are also listed.

HELP-SDF Mnemonic: **HPSDF**

GUIDANCE-**MODE** = ***NO** / ***Y**ES

,**SDF-COM**MANDS = ***NO** / ***Y**ES

,**ABBR**EVIATION-**RULES** = ***NO** / ***Y**ES

,**GUID**ED-**DIA**LOG = ***Y**ES (...)

***Y**ES(...)

SCREEN-STEPS = ***NO** / ***Y**ES

,**SPEC**IAL-**FUNC**TIONS = ***NO** / ***Y**ES

,**FUNC**TION-**KEYS** = ***NO** / ***Y**ES

,**NEXT-FIELD** = ***NO** / ***Y**ES

,**UNGUID**ED-**DIA**LOG = ***Y**ES (...) / ***NO**

***Y**ES(...)

SPECIAL-**FUNC**TIONS = ***NO** / ***Y**ES

,**FUNC**TION-**KEYS** = ***NO** / ***Y**ES

Notational conventions of the HELP-SDF user command

Notational conventions

Formal representation	Explanation	Examples
UPPERCASE LETTERS	Uppercase letters denote keywords. Keywords representing constant operand values start with an asterisk, *	**HELP-SDF** **SCREEN-STEPS** = ***NO**
UPPERCASE LETTERS in boldface	Uppercase letters in boldface identify guaranteed or suggested abbreviations of the keywords.	**GUID**ANCE**-MODE** = ***Y**ES
=	The equality sign links an operand name with its associated operand values.	**GUID**ANCE**-MODE** = ***NO**
< >	Angle parentheses identify variables for which the range of possible values is defined by the data types and their extensions.	**SYNTAX-F**ILE = <full-filename 1..54>
Underscoring	An underscore identifies the default value of an operand.	**GUID**ANCE**-MODE** = ***NO**
/	A backslash separates alternative operand values.	**NEXT-FIELD** = ***NO** / ***Y**ES
(...)	Round parentheses identify operand values which introduce a structure.	,**UNGUID**ED**-DIA**LOG = ***Y**ES (...) / ***NO**
[ ]	Square parentheses identify optional operand values which introduce a structure; the structure which follows the operand value may be specified without the introductory value.	**SELECT** = [***BY-ATTR**IBUTES](...)
Indentation	An indentation identifies a dependency on the higher level operand concerned.	,**GUID**ED**-DIA**LOG = ***Y**ES (...) ***Y**ES(...) **SCREEN-STEPS** = ***NO** / ***Y**ES

Notational conventions (Section 1 of 2)

Formal representation	Explanation	Examples
|	A vertical line identifies operands within a structure which belong together. It extends from the beginning to the end of the structure. Within any strucutre there may be other structures. The number of vertical lines preceding an operand corresponds to the structure depth.	**SUP**PORT = ***TAPE**(...) ***TAPE(...)** | **VOL**UME = ***ANY**(...) | ***ANY**(...) | | ...
,	A comma precedes additional operands at the same level in the structure.	**GUID**ANCE**-MODE** = ***NO** / ***Y**ES ,**SDF-COM**MANDS = ***NO** / ***Y**ES
list-poss(n):	The operand values which follow 'list-poss' can be used to form a list. If (n) is specified, the list may contain at most n elements. If the list contains more than one element, they must be enclosed in round parentheses.	list-poss: ***SAM** / ***ISAM** list-poss(40): <structured-name 1..30> list-poss(256): ***OMF** / ***SYSLST**(...) / <full-filename 1..54>
Mnemonic:	The name which follows here is a guaranteed alias for the command or statement name.	**HELP-SDF** Mnemonic: **HPSDF**

Notational conventions (Section 2 of 2)

2.2 Brief command description

SDF commands

Command	Meaning
EXIT-JOB	This command terminates the current job, releases the virtual memory pages and devices occupied by it and makes available the output system files to be output to printer or tape.
LOGOFF	This command terminates the current job, releases the virtual memory pages and devices occupied by it and makes available the output system files to be output to printer or tape.
MODIFY-JOB-OPTIONS	This command enables the user to modify definitions relating to the logging of the job.
MODIFY-MSG-OPTIONS	This command enables the user to receive or suppress messages sent by the operator using the MESSAGE or BROADCAST command.
MODIFY-TERMINAL-OPTIONS	This command enables the user to modify the output control for the terminal.
SET-LOGON-PARAMETERS	This command enables the user to initiate a dialog job at the terminal.
SHOW-TERMINAL-OPTIONS	This command enables the user to request output of the attributes of the display terminal.

2.3 EXIT-JOB

Application area: JOB

Privileges: All groups

Description of the function

The EXIT-JOB command terminates the current job. The virtual memory pages and devices occupied by the job are subsequently released and the system output files are readied for output to printer or tape.
If new file generations have been created during the job, the system outputs the names of the file generations concerned, their base values, and the names of the first and current file generations.

Note

In contrast to the LOGOFF command, output of the system files to tape is not supported.

Format

EXIT-JOB

MODE = ***NORM**AL / ***ABNORM**AL

,**SYS**TEM-**OUT**PUT = ***ALL** / ***NONE** / ***PAR**AMETERS(...)

***PAR**AMETERS(...)

SYSLST-OUTPUT = ***NONE** / ***PRINT**ER

SYSOUT-OUTPUT = ***NONE** / ***PRINT**ER

,**KEEP-CONN**ECTION = ***NO** / ***Y**ES

Description of operands

MODE =
Specifies whether the job is to be terminated normally or abnormally.

MODE = *NORMAL
The job is terminated normally. The status indicator of a monitoring job variable is set to '$T'.

MODE = *ABNORMAL
The job is terminated abnormally. The status indicator of a monitoring job variable is set to '$A'. Whether or not a dump is taken depends on the test options defined within the task (cf. MODIFY-TEST-OPTIONS command, DUMP operand).

SYSTEM-OUTPUT =
Controls output of the system files *SYSLST and *SYSOPT, and for batch jobs also output of the system file *SYSOUT.

SYSTEM-OUTPUT = *ALL
All non-empty system files are output: *SYSLST and *SYSOUT to printer, *SYSOPT to floppy disk.

SYSTEM-OUTPUT = *NONE
Output of the system files is suppressed.

SYSTEM-OUTPUT = *PARAMETERS(...)
Controls which of the two system files (*SYSLST or *SYSOUT) is output to the printer.

SYSLST-OUTPUT = *NONE
The system file *SYSLST is not output.

SYSLST-OUTPUT = *PRINTER
The system file *SYSLST is output to the printer.

SYSOUT-OUTPUT = *NONE
The system file *SYSOUT is not output.

SYSOUT-OUTPUT = *PRINTER
The system file *SYSOUT is output to the printer.

KEEP-CONNECTION =
Specifies whether the processor connection is to be retained, enabling a new job to be started immediately.
This operand is permitted only in interactive mode; it is ignored in batch mode.

KEEP-CONNECTION = *NO
Clears down the processor connection.

KEEP-CONNECTION = *YES
Retains the processor connection. A new job can be started immediately.

Command return code

(SC2)	SC1	Maincode	Meaning / guaranteed messages
	0	CMD0001	Command executed [1)]
	1	CMD0202	Syntax error
	32	JMS0221	System error; no message is sent to SYSOUT

1) Cannot be evaluated since in this case the task is terminated!

Notes

– Commands in non-S procedures and ENTER procedures:
if an error in the command sequence triggers the spin-off mechanism, a branch is taken to whichever of the following commands comes first: EXIT-JOB, LOGOFF, SET-JOB-STEP, CANCEL-PROCEDURE, END-PROCEDURE or EXIT-PROCEDURE.

– If an EXIT-JOB command is issued while a program is loaded, any ABEND-STXIT routine that may have been defined is activated. This does not happen if the EXIT-JOB command is called via the CMD macro (cf. "Executive Macros" manual [3]).

Example 1: Abnormal task termination and the test option DUMP=STD

```
/mod-test-opt dump=std
/exit-job mode=abnormal,keep-conn=yes
%  IDAON51 PROGRAM INTERRUPT AT LOCATION '00000000 (NONAME), (CDUMP)'
%  IDAON45 DUMP DESIRED? REPLY (Y = USER/AREA DUMP; Y,SYSTEM = SYSTEM DUMP;
N = NO)?n
%  EXC0419 /LOGOFF AT 1223 ON 93-02-23 FOR TSN '2DXV'
%  EXC0421 CPU TIME USED: 0.1210 , SERVICE UNITS: 000000000001854
%  JMS0150 INSTALLATION ' H120-I', BS2000 VERSION 'V110', HOST 'D516ZE04':
PL EASE ENTER '/SET-LOGON-PARAMETERS' OR '?'
```

Example 2: Abnormal task termination and the test option DUMP=YES

```
/mod-test-opt dump=yes
/exit-job mode=abnormal,keep-connection=yes
%  IDAON51 PROGRAM INTERRUPT AT LOCATION '00000000 (NONAME), (CDUMP)'
%  IDAON53 DUMP BEING PROCESSED. PLEASE HOLD ON
%  IDAON54 'USERDUMP' WRITTEN TO FILE '$USER1.DUMP.ULF.2D2L.00001'
%  IDAON55 TITLE: 'TSN-2D2L  UID-USER1  AC#-M0815RAN  USERDUMP  PC-00000000
   EC-50  VERS-110  DUMP-TIME  12:25:23  93-02-23'
%  EXC0419 /LOGOFF AT 1225 ON 93-02-23 FOR TSN '2D2L'
%  EXC0421 CPU TIME USED: 0.0738 , SERVICE UNITS: 000000000001040
%  JMS0150 INSTALLATION ' H120-I', BS2000 VERSION 'V110', HOST 'D516ZE04':
PLEASE ENTER '/SET-LOGON-PARAMETERS' OR '?'
```

2.4 LOGOFF

Application area: JOB

Privileges: All groups

Description of the function

The LOGOFF command terminates the current job. Subsequently, the virtual memory pages and devices occupied by the job are released and the output system files for output to printer or tape made available.
If new file generations were created during the job, the system lists the names of the file generations involved, their base number as well as the names of the first and the current file generation.

Note

The LOGOFF command is still supported for the sake of compatibility, but the EXIT-JOB command should be used to end tasks. The EXIT-JOB commands offers the user greater functionality.

Format

LOGOFF
KEEP-CONNECTION = ***NO** / ***Y**ES ,**SYS**TEM-**OUT**PUT = ***PRINT** / ***DEL**ETE / ***TAPE-OUT**PUT

Description of operands

KEEP-CONNECTION =
Specifies whether the connection to the computer is to be kept so that a new job can be started immediately.
This operand is permitted only in interactive mode; in batch mode it is ignored.

KEEP-CONNECTION = *NO
Clears down the connection.

KEEP-CONNECTION = *YES
Maintains the connection with the computer. A new job can be started immediately.

SYSTEM-OUTPUT =
Specifies whether system files are to be output and defines the output medium. SYSLST and SYSOUT are not output if they are empty. No output takes place if NO-SPOOL was set at system generation (CLASS-2-OPTION: SSMLGOF1).

SYSTEM-OUTPUT = *PRINT
Outputs the system files SYSLST and SYSOUT (batch mode) on printer.
In the case of interactive tasks, an additional query may be output if this was set at system generation (CLASS-2-OPTION: SSMLGOF1).

SYSTEM-OUTPUT = *DELETE
Output of the system file is suppressed.

SYSTEM-OUTPUT = *TAPE-OUTPUT
Outputs the system files on tape. SYSLST and SYSOUT (batch mode) are written to tape in the file: *TAPE.TSNnnnn*, where *nnnn* is the task sequence number of the job ended with LOGOFF.
SYSOPT is written to a separate tape, likewise in the file: *TAPE.TSNnnnn*, where *nnnn* is a new task sequence number. This number is output on SYSOUT.

Command return code

(SC2)	SC1	Maincode	Meaning / guaranteed messages
	0	CMD0001	Command executed 1)
	1	CMD0202	Syntax error
	32	CMD0221	System error; no message output on SYSOUT

1) Cannot be evaluated since in this case the task is terminated!

Notes

– Commands in non-S procedures and ENTER procedures:
if an error in the command sequence activates the SPIN-OFF mechanism, processing branches to whichever of the following commands comes first: EXIT-JOB, LOGOFF, SET-JOB-STEP, CANCEL-PROCEDURE, END-PROCEDURE or EXIT-PROCEDURE.

– If a LOGOFF command is issued while a program is loaded, any ABEND-STXIT routine that may have been defined is activated. This is not the case when the LOGOFF command is called via the CMD macro (see the "Executive Macros" manual [3]).

Example 1

```
/LOGOFF KEEP-CON=YES
%  EXC0419 LOGOFF AT 0847 ON 85-03-19, FOR TSN 1104
%  EXC0421 USED CPU TIME: 4.8930 SECONDS
%  PLEASE LOGON
```

The connection to the computer is not cleared down.

Example 2

```
/LOGOFF SYS-OUT=DEL
%  EXC0419 LOGOFF AT 0917 ON 85-03-19, FOR TSN 1154
%  EXC0421 USED CPU TIME: 0.7934 SECONDS
```

The system file SYSLST is not printed out.

2.5 MODIFY-JOB-OPTIONS

Application area: JOB

Privileges: STD-PROCESSING
HARDWARE-MAINTENANCE
SAT-FILE-EVALUATION
SAT-FILE-MANAGEMENT
SECURITY-ADMINISTRATION

Description of the function

The MODIFY-JOB-OPTIONS command controls

– the format in which the system messages are output (INFORMATION-LEVEL operand),
– the output of console messages (OPERATOR-INTERACTION operand),
– the maximum number of output records in SYSLST and SYSOPT, (SYSLST-LIMIT and SYSOPT-LIMIT operands),
– the logging of the job run for the current job (LOGGING operand).

The current settings can be output using the SHOW-JOB-OPTIONS command.

Format

MODIFY-**JOB-OPT**IONS Mnemonic: **MDJO**

INFORMATION-**LEV**EL = **<u>*UNCHANGED</u>** / ***STD / *MEDIUM / *MIN**IMUM

,**OPER**ATOR-**INTERACT**ION = **<u>*UNCHANGED</u>** / ***STD / *NO** / ***Y**ES

,**SYSLST-LIM**IT = **<u>*UNCHA</u>**NGED / ***STD** / ***NO-LIMIT** / <integer 0..999999>

,**SYSOPT-LIM**IT = **<u>*UNCHA</u>**NGED / ***STD** / ***NO-LIMIT** / <integer 0..999999>

,**LOG**GING = **<u>PAR</u>**AMETERS (...)

***PAR**AMETERS(...)

LISTING = **<u>*UNCHANGED</u>** / ***STD / *NO** / ***Y**ES

,**HARDCOPY** = **<u>*UNCHANGED</u>** / ***STD / *NO** / ***Y**ES

Description of operands

INFORMATION-LEVEL =
Format in which system messages are output.

INFORMATION-LEVEL = *UNCHANGED
The current specification applies, i.e. the value last defined via MODIFY-JOB-OPTIONS during the currently executing job. If no value has been specified, then the default value specified by System Administration applies.

INFORMATION-LEVEL = *STD
The default value specified by System Administration applies (class 2 system parameter ECLMSG).

INFORMATION-LEVEL = *MEDIUM
System messages are output in full.

INFORMATION-LEVEL = *MINIMUM
System messages are output in coded short form.

OPERATOR-INTERACTION =
Specifies whether console messages and operator replies are to be output. Operator commands for controlling the current job (e.g. a change in priority), as well as general warning and error messages for the operator are not considered here.

OPERATOR-INTERACTION = *UNCHANGED
The current specification applies, i.e. the value last defined via MODIFY-JOB-OPTIONS during the currently executing job. If no value has been specified, then the default value specified by System Administration applies.

OPERATOR-INTERACTION = *STD / *NO
Console messages and operator replies are not output.

OPERATOR-INTERACTION = *YES
Console messages and operator replies are output.

SYSLST-LIMIT =
Maximum number of records that may be spooled out to SYSLST during the currently executing job.
Output records for SYSOUT are not taken into account.
This specification must not exceed the maximum value defined in the job class definition (this definition can be checked with the SHOW-JOB-CLASS command).
If the specified value is reached during logging, the job is terminated abnormally in batch mode. In interactive mode, the system asks whether the job is to be terminated or continued; if the job is continued, the counter is set to zero and the specified value again applies.

SYSLST-LIMIT = *UNCHANGED
The current specification applies, i.e. the value last defined via MODIFY-JOB-OPTIONS during the currently executing job. If no value has been defined, then the maximum value defined in the job class definition applies.

SYSLST-LIMIT = *STD
The value specified in the job class definition applies.

SYSLST-LIMIT = *NO-LIMIT
The number of records which may be output to SYSLST during the current job is unlimited. If a lower value is specified in the job class definition, then *NO-LIMIT is rejected.

SYSLST-LIMIT = <integer 0..999999>
Specifies the maximum number of records that may be spooled out to SYSLST during the currently executing job. The specified number must not exceed the limit defined in the job class definition.

SYSOPT-LIMIT =
Maximum number of records that may be spooled out to SYSOPT during the currently executing job.
Output records for SYSOUT are not taken into account.
This specification must not exceed the maximum value defined in the job class definition (this definition can be checked with the SHOW-JOB-CLASS command).
If the specified value is reached during logging, the job is terminated abnormally in batch mode. In interactive mode, the system asks whether the job is to be terminated or continued; if the job is continued, the counter is set to zero and the specified value again applies.

SYSOPT-LIMIT = *UNCHANGED
The current specification applies, i.e. the value last defined via MODIFY-JOB-OPTIONS during the currently executing job. If no value has been defined, then the maximum value defined in the job class definition applies.

SYSOPT-LIMIT = *STD
The value specified in the job class definition applies.

SYSOPT-LIMIT = *NO-LIMIT
The number of records which may be output to SYSOUT during the current job is unlimited. If a lower value is specified in the job class definition, then *NO-LIMIT is rejected.

SYSOPT-LIMIT = <integer 0..999999>
Specifies the maximum number of records that may be spooled out to SYSOUT during the currently executing job. The specified number must not exceed the limit defined in the job class definition.

LOGGING = *PARAMETERS(...)
Information on logging of the job run.

LISTING =

Specifies whether the job run is also to be logged on SYSLST.
Console messages and operator replies (OPERATOR-INTERACTION operand) are supplemented with the time of day when logged on SYSLST.
System messages requiring an answer from the user are not logged, nor is the message “ABNORMAL PROGRAM TERMINATION".

In line mode, logging proceeds by lines, i.e. NL control characters are evaluated.
In format mode (menu), logging takes place continuously, i.e. in the log the format is destroyed; NL control characters are not evaluated.

LISTING = *UNCHANGED
The current specification applies, i.e. the value last defined via MODIFY-JOB-OPTIONS during the currently executing job. If no value has been defined, then the maximum value defined by System Administration applies.

LISTING = *STD / *NO
The job run should not be logged additionally on SYSLST.

LISTING = *YES
The job run should be logged additionally on SYSLST.

HARDCOPY =
This operand is ignored in batch mode.
Specifies whether the job run is also to be logged on a hardcopy unit. Formats (menus) cannot be logged.

HARDCOPY = *UNCHANGED
The current specification applies, i.e. the value last defined via MODIFY-JOB-OPTIONS during the currently executing job. If no value has been defined, then the maximum value defined by System Administration applies.

HARDCOPY = *STD / *NO
The job run should not be additionally logged on a hardcopy unit.

HARDCOPY = *YES
The job run should be additionally logged on a hardcopy unit.

Command return code

(SC2)	SC1	Maincode	Meaning / guaranteed messages
	0	CMD0001	Command executed
	1	CMD0202	Syntax error
	32	CMD0221	System error
	64	JMS0630	Semantic error (see SYSOUT message)

Example 1

```
/mod-job-opt inf-level=min,oper-interact=yes,log=par(list=yes)————————(01)
/xxx
%  CMD0186 XXX
/mod-job-opt inf-level=med ————————————————————————————————————————(02)
/xxx
%  CMD0186 OPERATION NAME 'XXX' UNKNOWN
/mod-job-opt  log=par(list=no) ————————————————————————————————————(03)
```

(01) The following specifications are in effect:
- Coded short form of messages on SYSOUT
- Logging of console messages and operator answers on SYSOUT
- Logging on SYSLST.

(02) The specifications are modified to:
- Full-length messages on SYSOUT.

(03) Logging on SYSLST is deactivated.

Example 2

```
/mod-job-opt syslst-limit=100
/show-file-attr out=*syslst
%  SSM2222 SPECIFIED MAXLST LIMIT REACHED. CONTINUE? REPLY (Y=YES; N=NO)?n
%  EXC0736 ABNORMAL TASK TERMINATION. ERROR CODE 'SSM2223': /HELP-MSG SSM2223
%  EXC0419 /LOGOFF AT 1821 ON 93-03-17 FOR TSN '3F8W'
%  EXC0421 CPU TIME USED: 81.3804 , SERVICE UNITS: 000000000678129
%  SSM2075 SPOOLOUT REQUESTED FOR FILE 'SYSLST'?  REPLY (Y=YES; N=NO)?
```

Specifies a limit of 100 for the number of records to be output to SYSLST. Output from the SHOW-FILE-ATTRIBUTES command is routed to SYSLST, and reaches the limit on number of records. The answer *N* (NO) is given to message *SSM2222*; The task is terminated abnormally.

Example 3

```
/ass-syslst lst.file ------------------------------------------------------------(01)
/mod-job-opt syslst-limit=10,log=par(list=yes) ----------------------------------(02)
/show-file-attr sdfu.sam.,output=*syslst ----------------------------------------(03)
SSM2222 SPECIFIED MAXLST LIMIT REACHED. CONTINUE? REPLY (Y=YES; N=NO)?y
%  SSM2222 SPECIFIED MAXLST LIMIT REACHED. CONTINUE? REPLY (Y=YES; N=NO)?y
%:N:    PUBLIC:     12 FILES RES=     5535  FREE=      128  REL=      117 PAGES
/ass-syslst *primary ------------------------------------------------------------(04)
/show-file lst.file -------------------------------------------------------------(05)
```

```
          (IN)     MOD-JOB-OPT SYSLST-LIMIT=10,LOG=PAR(LIST=YES)
          (IN)     SHOW-FILE-ATTR SDFU.SAM.,OUTPUT=*SYSLST
 AFILE STATUS      SDFU.SAM.,LIST=((SYSLST),STANDARD)

   FILENAME                                          PAM-     FREE-     SE
                                                     PAGES    PAGES      A

   :N:$USERXY01.SDFU.SAM.ALT                          1854        0
        (OUT)   %  SSM2222 SPECIFIED MAXLST LIMIT REACHED. CONTINUE? REPLY (Y=
        (IN)    Y
   :N:$USERXY01.SDFU.SAM.ALT.INH                        18        0
   :N:$USERXY01.SDFU.SAM.ALT.STW                        39        0
   :N:$USERXY01.SDFU.SAM.FORM                         1824        2
   :N:$USERXY01.SDFU.SAM.FORM.INH                       18        0

   :N:$USERXY01.SDFU.SAM.FORM.STW                       39        1
   :N:$USERXY01.SDFU.SAM.HFORTX                        291        2
   :N:$USERXY01.SDFU.SAM.HFORT1                        618        1
   :N:$USERXY01.SDFU.SAM.HFORT2                        261       76
 +                                                   S*SOF+   1(         1)
```

```
  :N:$USERXY01.SDFU.SAM.HFORT2.ORIG                    261        2
       (OUT)    %  SSM2222 SPECIFIED MAXLST LIMIT REACHED. CONTINUE? REPLY (Y=
       (IN)     Y

  :N:$USERXY01.SDFU.SAM.HFORT3                         156       43
  :N:$USERXY01.SDFU.SAM.HFORT3.ORIG                     56        1

   PUBLIC SPACE:       12 FILES                       5535

       (OUT)    :N:    PUBLIC:     12 FILES RES=     5535  FREE=      128  REL
       (IN)     SHOW-FILE LST.FILE
       (OUT)    %  SHO0003 'DMS' REPORTED ERROR '0D99'. COMMAND NOT PROCESSED
       (IN)     ASS-SYSLST *P




%  SHO0301 WARNING: END OF FILE REACHED
e                                                    S*SOF+    24(     1)
```

(01) The file *LST.FILE* is assigned to the system file SYSLST.

(02) Logging on SYSLST is switched on, and a limit of 10 records is set.

(03) Outputs from the SHOW-FILE-ATTRIBUTES command for all the files with names that start with *SDFU.SAM*. is routed to SYSLST. During output, the record limit is twice exceeded (the reply *Y* is given to message *SSM2222*).

(04) The system file SYSLST is again given the primary assignment.

(05) The SHOW-FILE command is used to output the contents of the log file *LST.FILE*.

2.6 MODIFY-MSG-OPTIONS

Application area: JOB

Privileges: STD-PROCESSING
HARDWARE-MAINTENANCE
SAT-FILE-EVALUATION
SAT-FILE-MANAGEMENT
SECURITY-ADMINISTRATION

Description of the function

With the MODIFY-MSG-OPTIONS command, users can suppress the output of messages for their job, sent by the operator

– to the user (job-oriented: MESSAGE),
– to all users (BROADCAST).

Urgent messages, e.g. messages at system shutdown, are nevertheless output.

Format

MODIFY-**MSG-OPT**IONS
OPERATOR-**BROAD**CAST = ***UNCHA**NGED / ***Y**ES / ***NO** ,**OPER**ATOR-**MSG** = ***UNCHA**NGED / ***Y**ES / ***NO** ,**INFO-OUTPUT** = ***UNCHA**NGED / list-poss(2): ***STD** / ***SYSTEMLINE**

Description of operands

OPERATOR-BROADCAST = *UNCHANGED / *YES / *NO
Specifies whether messages sent by the operator with BROADCAST are to be received by the user.

OPERATOR-MSG = *UNCHANGED / *YES / *NO
Specifies whether messages sent by the operator with MESSAGE are to be received by the user.

INFO-OUTPUT = *UNCHANGED / list-poss(2): *STD / *SYSTEMLINE
Specifies how the messages sent by the operator are to be output at the terminal.
With *STD, the messages are output to the top line of the screen.
With *SYSTEMLINE, the messages are output to the bottom line of the screen (message line). If both values are selected, messages are output to the current line and to the message line.

Command return code

(SC2)	SC1	Maincode	Meaning / guaranteed messages
	0	CMD0001	Command executed without error
1	32	NBR0940	Command not executed, because not possible to create an entry in the TCB

2.7 MODIFY-TERMINAL-OPTIONS

Application area: JOB

Privileges: STD-PROCESSING
TSOS
HARDWARE-MAINTENANCE
SAT-FILE-EVALUATION
SAT-FILE-MANAGEMENT
SECURITY-ADMINISTRATION

Description of the function

The MODIFY-TERMINAL-OPTIONS command allows users to change the logical attributes of their terminals. The command can be used only in timesharing mode and only has any effect if the terminal is being used as a line or page terminal.

The logical attributes of the terminal are defined in the system by means of class-2 system parameters at system generation time, through generation of the terminal, by the terminal itself, and through VTSU operating parameters. These defined values are in force at the start of the interactive task; they can be modified within the user's task by means of the MODIFY-TERMINAL-OPTIONS command. The user can set the following logical attributes.

Logical attribute	Operand	Preset by
Type of screen overflow control	OVERFLOW-CONTROL	Class-2 system parameter TCHOFLO
Number of lines after which screen overflow occurs	MAXIMUM-LINES	Generation of the terminal
Length of output lines	LINE-LENGTH	Generation of the terminal
Control of display of outputs and the type of input	WRITE-READ-MODE	Class-2 system parameter TCHREAD
End-of-line character	LINE-END-CHARACTER	VTSU operating parameter
Upper/lower-case distinction	LOWER-CASE	Generation of the terminal
Use of graphic characters	GRAPHICS	Generation of the terminal
Use of an APL character set	APL-CHARACTER-SET	Generation of the terminal
Support for a hardcopy printer	HARDCOPY	Generation of the terminal
Substitute characters for displaying nonprintable characters	SUBSTITUTE-CHARACTER	Dependent on terminal type: - X' 4A' for type 8193 - X' 44' for type 3270 - X' 07' for all other types ("smudge character")

Logical attribute	**Operand**	**Preset by**
Acknowledgment procedure	ACKNOWLEGE-OUTPUT	Class-2 system parameter TCHTACK
Code table	CODED-CHARACTER-SET	Dependent on – terminal type – user entry – VTSU operating parameter (see table below)

The default value UNCHANGED in the various operands means in each case that the existing specification applies.

The following table shows how the code tables preset for CODED-CHARACTER-SET depend on the terminal type, user entry for the default code and the VTSU operating parameter TIAM-PERM8.

Terminal type	**7-bit**	**8-bit**	**8-bit**	**8-bit**
Standard CCS	-	EDF03iRV	EDFxxx	EDFxxx
TIAM-PERM8	-	-	N	Y
Code table	EDF03iRV	EDF03iRV	EDF03iRV	EDFxxx

Format

MODIFY-**TERM**INAL-OPTIONS

OVERFLOW-**CONTR**OL = ***UNCHA**NGED / ***NO-CONTR**OL / ***USER-ACK**NOWLEDGE / ***TIME**(...)

***TIME**(...)

TIMEOUT = ***STD** / <integer 0..60>

,**MAX**IMUM-**LINES** = ***UNCHA**NGED / <integer 3..255>

,**LINE-LEN**GTH = ***UNCHA**NGED / <integer 10..255>

,**LINE-END**-CHARACTER = ***UNCHA**NGED / ***NONE** / <c-string 1..1>

,**WR**ITE-**READ-MODE** = ***UNCHA**NGED / ***MOD**IFIED-**FIELD**S / ***NO-FIELD**S

,**LOW**ER-**CASE** = ***UNCHA**NGED / ***Y**ES / ***NO**

,**GRAPHICS** = ***UNCHA**NGED / ***Y**ES / ***NO**

,**APL-CHAR**ACTER-SET = ***UNCHA**NGED / ***Y**ES / ***NO**

,**HARDCOPY** = ***UNCHA**NGED / ***NO** / ***LOC**AL / ***CENTR**AL

,**SUBST**ITUTE-**CHAR**ACTER = ***UNCHA**NGED / ***STD** / <c-string 1..1> / <x-string 1..2>

,**ACK**NOWLEDGE-**OUTP**UT = ***UNCHA**NGED / ***Y**ES / ***NO**

,**CODED-CHAR**ACTER-**SET** = ***UNCHA**NGED / ***7-BIT** / ***8-BIT-DEF**AULT / <name 1..8>

Description of operands

OVERFLOW-CONTROL = *UNCHANGED / *NO-CONTROL / *USER-ACKNOWLEDGE / *TIME(...)

Type of control in the event of screen overflow.
An overflow is assumed by the system any time the number of lines to be displayed by the next output from the computer exceeds the number specified in MAXIMUM-LINES.
When an output begins, the user's last input is taken into account in calculating the screen overflow.
If overflow output is controlled by user acknowledgement (*USER-ACKNOWLEDGE), the system request for an acknowledgement counts as one of the lines in determining the number to output to the screen.

OVERFLOW-CONTROL = *NO-CONTROL

No overflow control. In cases of long output from the computer, the system takes no steps to enable the ESCAPE function or to prevent premature overwriting of data on the screen.

OVERFLOW-CONTROL = *USER-ACKNOWLEDGE
Overflow control by means of acknowledgments. When overflow occurs, the system requests the terminal user to enter an acknowledgment with the message %PLEASE ACKNOWLEDGE. In this way, the user can determine the speed at which output from the computer is displayed. Any input other than ESCAPE and BREAK counts as an acknowledgment. After an acknowledgment is received, output is continued.

OVERFLOW-CONTROL = *TIME(...)
Overflow control is effected on a time basis. After the set waiting time has expired, screen output is resumed. When the last line on the screen is reached, each further line which is output “pushes” the existing screen content up one line, so that the input or output data at the top of the screen is overwritten..

TIMEOUT = *STD / <integer 0..60>
Delay time, in seconds, that is to pass when screen overflow has occurred before further output is displayed.

TIMEOUT = STD
The value used is the default value defined at system generation (6 seconds), or the last value defined by the user.

Note

The value set for timing control (TIMEOUT) also applies when there is a change in screen mode (e.g. a change, after a line-oriented output has terminated, back into the formated input mode of the prompted dialog).

MAXIMUM-LINES = *UNCHANGED / <integer 3..255>
Maximum number of lines that may be displayed until the next overflow control.

LINE-LENGTH = *UNCHANGED / <integer 10..255>
This operand is only accepted for the sake of compatibility. It is not evaluated if specified.

LINE-END-CHARACTER = *UNCHANGED / *NONE / <c-string 1..1>
End-of-line character, to be appended to each line output at the terminal (only applicable to 8110 and 3270).
It may be any character that can be entered from the keyboard; it is used in output to indicate logical end-of-line. Subsequently, display continues in the next line. In inputs, this character in the input text is passed by the system as NEW LINE to the user program. (Default setting by the system is "\" or "ö".)

LINE-END-CHARACTER = *NONE
No end-of-line character. For output from the computer, logical end-of-line is represented only by a change of lines. For input, the function is not available.

WRITE-READ-MODE = *UNCHANGED / *MODIFIED-FIELDS / *NO-FIELDS
Controls the display of output messages and the type of input.

WRITE-READ-MODE = *MODIFIED-FIELDS
Reads modified fields. Data is displayed on the screen in fields, input to the computer takes place through the transfer of modified fields.

WRITE-READ-MODE = *NO-FIELDS
Reads unprotected fields. Data is displayed on the screen without fields, input takes place from the beginning of the screen or from the cursor position.

LOWER-CASE = *UNCHANGED / *YES / *NO
Specifies whether output of lowercase letters is to be possible.

GRAPHICS = *UNCHANGED / *YES / *NO
Specifies whether a graphics option can be used.

APL-CHARACTER-SET = *UNCHANGED / *YES / *NO
Specifies whether an APL character set can be used.

HARDCOPY = *UNCHANGED / *NO / *LOCAL / *CENTRAL
Specifies what type of hardcopy logging is to be possible.

HARDCOPY = *NO
No hardcopy printer.

HARDCOPY = *LOCAL
Local hardcopy printer, operable directly at the terminal. This operand is interpreted only if a local hardcopy device was assigned when the connection was established. Otherwise, the operand is ignored.

HARDCOPY = *CENTRAL
Central hardcopy printer, operable on channel 0,...,31 of the same cluster controller to which the terminal is connected. The channel is the one specified at generation of the terminal in PDN (default value = 0).

SUBSTITUTE-CHARACTER = *UNCHANGED / *STD / <c-string 1..1> / <x-string 1..2>
Substitute character to be used in place of nonprintable characters.
In output texts, nonprintable characters are replaced by the specified substitute character.

SUBSTITUTE-CHARACTER = *STD
A device-specific smudge character is used as the substitute character.

ACKNOWLEDGE-OUTPUT = *UNCHANGED / *YES / *NO
Specifies whether terminal output is to be acknowledged within the system.

ACKNOWLEDGE-OUTPUT = *NO
No internal acknowledgment.
Messages that are output immediately prior to EXIT-JOB or LOGOFF may be lost due to the connection being cleared down.
The value set for ACKNOWLEDGE-OUTPUT applies for all subsequent output in program and system mode until a new setting is made with MODIFY-TERMINAL-OPTIONS or EXIT-JOB/LOGOFF (default setting at system generation).

CODED-CHARACTER-SET = *UNCHANGED / *7-BIT / *8-BIT-DEFAULT / <name 1..8>
Specifies whether the terminal is to operate in 7-bit or 8-bit mode.

CODED-CHARACTER-SET = *7-BIT
Deactivates the 8-bit character set. The terminal operates in 7-bit mode.

CODED-CHARACTER-SET = *8-BIT-DEFAULT
Activates the 8-bit code table in the user entry, if there is an entry for it. The terminal then operates in 8-bit-mode. If the user entry does not contain an 8-bit code table, the terminal continues to operate in 7-bit mode.

CODED-CHARACTER-SET = <name 1..8>
Name of an 8-bit code table (CCS)

Depending on the code specified explicitly in the VTSUCB, the following situations can occur:

Standard CCS	7	7	7	7	7	7	name	name	name1	name	name	name1
8-bit termina	n	n	n	j	j	j	n	n	n	j	j	j
MOD-TERM-OPT C-C-S =	7	8	name	7	8	name	7	8	name2	7	8	name2
Result	1)	2)	2)	3)	4)	7)	1)	2)	2)	5)	6)	8)

1) The command is accepted.
The terminal will operate in 7-bit mode.
First data item supplied by TSTAT: blank character
Second data item supplied by TSTAT: blank character

2) The command is rejected.
The terminal will operate in 7-bit mode.
First data item supplied by TSTAT: blank character
Second data item supplied by TSTAT: blank character

3) The command is accepted.
The terminal will operate in 7-bit mode if no VTSUCB is used or if blanks have been defined in the VTSUCB.
First data item supplied by TSTAT: blank character
Second data item supplied by TSTAT: blank character

4) The command is rejected.
The terminal will operate in 7-bit mode if no VTSUCB is used or if blanks have been defined in the VTSUCB.
First data item supplied by TSTAT: blank character
Second data item supplied by TSTAT: blank character

5) The command is accepted.
The terminal will operate in 7-bit mode if no VTSUCB is used or if blanks have been defined in the VTSUCB.
The terminal will operate in 8-bit mode if a valid extended code name or *EXTEND has been defined in the VTSUCB. If *EXTEND is defined, the code 'name' is used.
First data item supplied by TSTAT: name
Second data item supplied by TSTAT: blank character

6) The command is accepted.
The terminal will always operate in 8-bit mode. The code 'name' is used if no VTSUCB is used or if *EXTEND or blanks are included in the VTSUCB. In all other cases the extended character set specified explicitly ('name') is used.
First data item supplied by TSTAT: name
Second data item supplied by TSTAT: name

7) The command is accepted provided that 'name' is supported by the terminal.
The terminal will operate in 8-bit mode (using the character set 'name') if no VTSUCB is used or if no character set name is specified in the VTSUCB.
First data item supplied by TSTAT: blank character
Second data item supplied by TSTAT: name

8) The command is accepted provided that 'name2' is supported by the terminal.
The terminal will operate in 8-bit mode (using the character set 'name2') if no VTSUCB is used or if no character set name is specified in the VTSUCB.
First data item supplied by TSTAT: name1
Second data item supplied by TSTAT: name2

Command return code

(SC2)	SC1	Maincode	Meaning / guaranteed messages
	0	CMD0001	Command executed successfully
1	0	CMD0001	Command not effective until overflow control has been reactivated
2	0	TIA0500	Command executed with warning; at least one entry has been ignored because of a semantic error
	1	TIA0502	Syntax error
	32	TIA0122	Unrecoverable VTSU-B error
	64	CMD0216	Privilege error

Example

```
/show-file proc.time ————————————————————————————————————————(01)
```

```
/      SET-PROC-OPT     ERROR-MECHANISM=*BY-RETURNCODE
/      SET-VAR          A = 1
/REPEAT-1:  REPEAT
/              WRITE-TEXT '** The time is now &(TIME()).'
/              SET-VAR A = (A + 1)
/           UNTIL   CONDITION = (A > 10)
/EXIT:  EXIT-PROC
/ERROR: IF-BLOCK-ERROR
/       WRITE-TEXT '** Error, with Subcode1 &(SC1) and Subcode2 (SC2) *'
/       HELP-MSG MSG-ID=&(MC)
/       END-IF
%  SHO0301 WARNING: END OF FILE REACHED
e                                                S*SOF+     1(     1
```

```
/mod-term-opt overflow-control=*no ————————————————————————(02)

/call-proc proc.time ——————————————————————————————————————(03)
** The time is now 17:31:43.
** The time is now 17:31:43.
** The time is now 17:31:43.
** The time is now 17:31:43.
** The time is now 17:31:43.
** The time is now 17:31:43.
** The time is now 17:31:43.
** The time is now 17:31:43.
** The time is now 17:31:43.
** The time is now 17:31:43.

/mod-term-opt ov-contr=*user-ack,max-lines=3 ——————————————(04)

/call-proc proc.time ——————————————————————————————————————(05)
** The time is now 17:33:46.
%PLEASE ACKNOWLEDGE
** The time is now 17:33:46.
** The time is now 17:34:14.
%PLEASE ACKNOWLEDGE
** The time is now 17:34:14.
** The time is now 17:34:27.
%PLEASE ACKNOWLEDGE
** The time is now 17:34:27.
** The time is now 17:34:29.
%PLEASE ACKNOWLEDGE
** The time is now 17:34:29.
** The time is now 17:34:30.
%PLEASE ACKNOWLEDGE
** The time is now 17:34:30.
```

```
/mod-term-opt ov-contr=*time(timeout=30) ————————————————————(06)

/call-proc proc.time ————————————————————————————————————(07)
** The time is now 17:35:32.
** The time is now 17:35:32.
** The time is now 17:35:32.
** The time is now 17:36:02.
** The time is now 17:36:02.
** The time is now 17:36:02.
** The time is now 17:36:32.
** The time is now 17:36:32.
** The time is now 17:36:32.
** The time is now 17:37:02.
```

(01) The SHOW-FILE command outputs the contents of the file *PROC.TIME*. It is an S procedure which obtains the current time ten times (builtin function TIME()) and outputs it to SYSOUT using the WRITE-TEXT command. *E* (END) terminates the output of the SHOW-FILE command.

(02) Switches off overflow control (*OVERFLOW-CONTROL=*NO-CONTROL*).

(03) Calls the procedure *PROC.TIME*. As there is no screen overflow control, the message with the time is output ten times one after another. As the output is immediate, the displayed time does not change.

(04) Overflow control is set to **USER-ACKNOWLEDGE*, with a maximum of 3 output lines.

(05) Calls the procedure *PROC.TIME*. After every third line the system asks for an acknowledgment from the user. The first overflow occurs after the time is output for the firts time, because the procedure call and acknowledgment request count towards the maximum number of lines. The user acknowledges the request, e.g. with the Enter key. Output then continues. The delay between entries means that the time changes for each output display.

(06) Sets overflow control to a waiting time of 30 seconds. The maximum number of lines is not changed.

(07) Calls the procedure *PROC.TIME*. After output of each third line there is a delay of 30 seconds. This wait time can be seen from the times which are output: after every third output line the time increases by another 30 seconds.

2.8 SET-LOGON-PARAMETERS

Application area: JOB

Privileges: All groups

Description of the function

With the SET-LOGON-PARAMETERS command, users can initiate an interactive job on the terminal. When SET-LOGON-PARAMETERS is given as the first command in an ENTER file, it initiates a batch job on starting with the ENTER-JOB command.

The specifications in the SET-LOGON-PARAMETERS command identify the user (system access authorization check), characterize the job and control logging of job execution.
The access authorization specifications are checked against the user entry; further specifications relating to the job class and to the job attributes (job/run priority, system resources) are also checked against the entry in the job class definition. The entries are accessible to the user by means of the SHOW-USER-ATTRIBUTES and SHOW-JOB-CLASS commands.
If different specifications appear for RUN-PRIORITY and CPU-LIMIT in the user entry and in the job class definition, the value more favorable to the user is permitted.

For batch jobs, the operands of the SET-LOGON-PARAMETERS command are only evaluated if the batch job is started from the console, but in this case only operands which the operator has not explicitly specified in the ENTER-JOB command are evaluated.

The JOB-PRIORITY, RERUN-AFTER-CRASH, FLUSH-AFTER-SHUTDOWN, START and REPEAT-JOB operands are possible only for batch jobs.

Format

SET-LOGON-PARAMETERS Mnemonic: **STLGP**

USER-IDENTIFICATION = **<u>*NO</u>** / <name 1..8>

,**ACCOUNT** = **<u>*NONE</u>** / <alphanum-name 1..8>

,**PASS**WORD = **<u>*NONE</u>** / <c-string 1..8> / <c-string 9..32> / <x-string 1..16> / ***SECRET**

,**JOB-CLASS** = **<u>*STD</u>** / <name 1..8>

,**JOB-NAME** = **<u>*NO</u>** / <name 1..8>

,**MONJV** = **<u>*NONE</u>** / <full-filename 1..54 without-gen-vers>

,**JV-PASS**WORD = **<u>*NONE</u>** / <c-string 1..4> / <x-string 1..8> / ***SECRET** / <integer -2147483648..2147483647>

,**JOB-PRIO**RITY = **<u>*STD</u>** / <integer 1..9>

,**RERUN-AF**TER**-CRASH** = **<u>*NO</u>** / ***Y**ES

,**FLUSH-AF**TER**-SHUTDOWN** = **<u>*NO</u>** / ***Y**ES

,**START** = **<u>*STD</u>** / ***SOON** / ***IMMED**IATELY / ***AT-STREAM**-STARTUP / ***WITHIN**(...) / ***AT**(...) / ***EARL**IEST(...) / ***LATEST**(...)

***WITHIN**(...)

HOURS = **<u>0</u>** / <integer 0..23>

,**MINUT**ES = **<u>0</u>** / <integer 0..59>

***AT**(...)

DATE = **<u>*TODAY</u>** / <date 8..10>

,**TIME** = <time>

***EARL**IEST(...)

DATE = **<u>*TODAY</u>** / <date 8..10>

,**TIME** = <time>

***LATEST**(...)

DATE = **<u>*TODAY</u>** / <date 8..10>

,**TIME** = <time>

continued ➠

```
,REPEAT-JOB = *STD / *NO / *DAILY / *WEEKLY / *AT-STREAM-STARTUP / *PERIOD(...)
    *PERIOD(...)
        HOURS = 0 / <integer 0..23>
        ,MINUTES = 0 / <integer 0..59>
,RESOURCES = *PARAMETERS (...)
    *PARAMETERS(...)
        RUN-PRIORITY = *STD / <integer 30..255>
        ,CPU-LIMIT = *STD / *NO / <integer 1..32767>
        ,SYSLST-LIMIT = *STD / *NO / <integer 0..999999>
        ,SYSOPT-LIMIT = *STD / *NO / <integer 0..999999>
,LOGGING = *PARAMETERS (...)
    *PARAMETERS(...)
        LISTING = *NO / *YES
        ,HARDCOPY = *NO / *YES
,JOB-PARAMETER = *NO / <c-string 1..127>
,PROTECTION = *NONE / *CANCEL
```

Description of operands

USER-IDENTIFICATION = *NO / <name 1..8>
User ID under which the interactive job is to run.

ACCOUNT = *NONE / <alphanum-name 1..8>
Account number of the user ID.

PASSWORD = *NONE / <c-string 1..8> / <c-string 9..32> / <x-string 1..16> / *SECRET
Password for the user ID.
The input of a “long” password (corresponding to <c-string 9..32>) is supported. A hash algorithm converts the “long” password into the internal 8 byte long representation. For details of the declaration of “long” passwords, see the MODIFY-USER-PROTECTION command.

The operand PASSWORD is defined as "secret":
– The input value is not logged.
– The input field is automatically blanked in a guided dialog.
– The specification SECRET is only permissible in an unguided dialog. This makes it possible to conceal the entry of the desired value. SDF prompts the user to enter the "secret" value and presents a blanked input field.

JOB-CLASS = *STD / <name 1..8>
Job class in which the interactive job is to run. The job class must be permitted for interactive jobs. The user can ascertain the job classes he is allowed to use from his user entry for the home pubset (SHOW-USER-ATTRIBUTES command output). This also displays the default job class that is preset with *STD. Users can obtain information about the characteristics of job classes (job class definition) by means of the SHOW-JOB-CLASS command.

JOB-NAME = *NO / <name 1..8>
Name for the interactive job. This name can be used to reference the interactive job (e.g. using SHOW-JOB-STATUS). All unnamed jobs started from within this interactive job are also assigned this name.

MONJV = *NONE / <full-filename 1..54 without-gen-vers>
Applies only if the JV software product is being used.
Name of the JV that is to monitor the interactive job. A JV can only be defined when the chargeable JV subsystem is installed.
Job monitoring is only started if the interactive job is accepted by the system's job management facility (JOB ACCEPTED).
The job originator must have write authorization because he instructs the system to write to the JV. If the JV is not accessible at the time of command processing, an error message is output to SYSOUT and the command is rejected. If the specified JV (own user ID) does not yet exist, it is created by the system (cf. default values for CREATE-JV). Users can address their interactive job via the specified JV (see "Job Variables" manual [10]).

$R	Job running
$T	Job terminated normally
$A	Job aborted

JV-PASSWORD = *NONE / <c-string 1..4> / <x-string 1..8> / <integer -2147483648..2147483647> / *SECRET
Only in conjunction with the JV software product
Password for the JV.
The operand is only evaluated when job monitoring has been defined (cf. operand MONJV).
The operand JV-PASSWORD is defined as "secret":

– The input value is not logged.
– The input field is automatically blanked in a guided dialog.
– The specification SECRET is only permissible in an unguided dialog. This makes it possible to conceal the entry of the desired value. SDF prompts the user to enter the "secret" value and presents a blanked input field.

RESOURCES = *PARAMETERS(...)
Specifications regarding run priority, CPU time and maximum number of SYSLST and SYSOPT records.

RUN-PRIORITY = *STD / <integer 30..255>
Run priority which the interactive job is to be assigned. The lower the value, the higher the priority. The maximum permissible value is specified in the job class definition and can be queried by means of the SHOW-JOB-CLASS command.

RUN-PRIORITY = *STD
Sets the default priority defined for the job class.

CPU-LIMIT = *STD / *NO / <integer 1..32767>
Maximum CPU time, in seconds, which the interactive job is allowed to use. The maximum permissible time depends on the specified job class. See also “Time limits in BS2000” in the Appendix to “BS2000/OSD-BC Executive Macros, Vol 3” [2])

CPU-LIMIT = *STD
Sets the default value for the selected job class.

CPU-LIMIT = *NO
Specifies no time limit (NTL) for the interactive job run. This operand value is permitted only if the user ID has the requisite authorization (JOIN entry).

SYSLST-LIMIT = *STD / *NO / <integer 0..999999>
Specifies the maximum number of records the job can output to the system files SYSLST, SYSLST01, SYSLST02, ..., SYSLST99. Data records in the system file SYSOUT which are simultaneously written to SYSLST are not included in the count. The entry made here must not exceed the limit set in the job class definition. This limit can be queried using the SHOW-JOB-CLASS command.

SYSLST-LIMIT = *STD
Default value of the selected job class. If the specified number is exceeded:

– in interactive mode, the user can specify whether the job is to be continued or aborted. If the job is continued, the records are output until "number" is reached again.
– in batch mode, the job is aborted.

SYSLST-LIMIT = *NO
Sets no limit on the number of records output.

SYSOPT-LIMIT = *STD / *NO / <integer 0..999999>
Specifies the maximum number of records the job can output to the system file SYSOPT.
The entry made here must not exceed the limit specified in the job class definition. This limit can be queried by means of the SHOW-JOB-CLASS command.

SYSOPT-LIMIT = *STD
Default value of the selected job class. If the specified number is exceeded:

– in interactive mode, the user can specify whether the job is to be continued or aborted. If the job is continued, the records are output until "number" is reached again.
– in batch mode, the job is aborted.

SYSOPT-LIMIT = *NO
Sets no limit on the number of records output.

LOGGING = *PARAMETERS(...)

LISTING = *NO / *YES
Specifies whether the job run is also to be logged on SYSLST.

JOB-PARAMETER =
Specifies additional attributes for the selected job class - assuming that the system administrator has defined some and made them known.

JOB-PARAMETER = *NO
Sets no additional attributes.

JOB-PARAMETER = <c-string 1..127>
c-string = sequence of any characters; assigned by the system administrator to identify additional job class attributes.

PROTECTION = *NONE / *CANCEL
Specifies whether the interactive job is to be protected against accidental termination with the CANCEL-JOB command.

PROTECTION = *NONE
The interactive job is not protected against accidental termination.

PROTECTION = *CANCEL

The interactive job is not protected against accidental termination. In interactive jobs that wish to terminate this interactive job with the CANCEL-JOB command, the system additionally requests confirmation. Accidental termination of the interactive job due to incorrect specification of the job number should thus be prevented.

JOB-PRIORITY = *STD / <integer 1..9>
Only for batch jobs
Job priority to be given to the batch job.
The lower the value, the higher the priority. The maximum permissible value is defined in the job class definition and may be queried using the SHOW-JOB-CLASS command.

JOB-PRIORITY = *STD
The standard priority specified for the job class applies.

RERUN-AFTER-CRASH = *NO / *YES
Only for batch jobs

Specifies whether the batch job is to be restarted in the next session if processing is aborted on account of a system error or end-of-session.

FLUSH-AFTER-SHUTDOWN = *NO / *YES
Specifies whether the batch job is to be removed from the job queue if it has not been processed by the end of the session.

START =
Only for batch jobs
Starting time for the batch job. Values other than *STD are appropriate only if they are permitted by virtue of the job class definition (see SHOW-JOB-CLASS command).

START = *STD
The default value for the chosen job class applies.

START = *SOON
The job is to be started as soon as possible, in accordance with its priority.

START = *IMMEDIATELY
The job is to be started immediately.

START = *AT-STREAM-STARTUP
The job is to be started after the next startup of the job scheduler.

START = *WITHIN(...)
The job is to be started within the specified time period.

HOURS = 0 / <integer 0..23>
Number of hours.

MINUTES = 0 / <integer 0..59>
Number of minutes.

START = *AT(...)
The job is to be started at exactly the time specified.

DATE = *TODAY / <date 8..10>
Date; can be specified in the form yy-mm-dd or yyyy-mm-dd.
Any seconds value specified is ignored.

TIME = <time>
Time of day in the format hh:mm, where hh = hours and mm = minutes.
Any seconds value specified is ignored.

START = *EARLIEST(...)
The job is to be started no sooner than the time specified.

DATE = *TODAY / <date 8..10>
Date; can be specified in the form yy-mm-dd or yyyy-mm-dd.

TIME = <time>
Time of day in the format hh:mm, where hh = hours and mm = minutes.
Any seconds value specified is ignored.

START = *LATEST(...)
The job is to be started no later than the time specified.

DATE = *TODAY / <date 8..10>
Date; can be specified in the form yy-mm-dd or yyyy-mm-dd.

TIME = <time>
Time of day in the format hh:mm, where hh = hours and mm = minutes.
Any seconds value specified is ignored.

REPEAT-JOB =
Only for batch jobs
Time interval at which the batch job is to be repeated. Values other than *STD are appropriate only if permitted in accordance with the job class definition (see the SHOW-JOB-CLASS command). The time interval for the repetitions depends on the specification in the START operand; see the note in this regard, "Combinations of the START and REPEAT-JOB operands". For the repetitions, the following applies:

– The i-th repetition ($i \geq 1$) of a job is not started until the (i-1)th repetition has ended.
– Cancelation of the currently executing job (i) has no effect on the start of (i+1); ($i \geq 0$).
– Cancelation of the entire job: Both the currently executing job (i) and the subsequent job (i+1) must be canceled, ($i \geq 0$); (CANCEL-JOB command, or make job (i) the last job in the series using the MODIFY-JOB... command, REPEAT-JOB=NO).

REPEAT-JOB = *STD
The default value for the chosen job class applies.

REPEAT-JOB = *NO
The batch job is not repeated.

REPEAT-JOB = *DAILY
Daily repetition at the time specified with START.

REPEAT-JOB = *WEEKLY
Weekly repetition at the time specified with START.

REPEAT-JOB = *AT-STREAM-STARTUP
Repetition following each startup of the job scheduler.

REPEAT-JOB = *PERIOD(...)
Repetition after the specified time interval.

HOURS = 0 / <integer 0..23>
Number of hours.

MINUTES = 0 / <integer 0..59>
Number of minutes.

Command return code

(SC2)	SC1	Maincode	Meaning / guaranteed messages
	0	CMD0001	Command executed
130	64	JMS0640	Command rejected

If SET-LOGON-PARAMETERS is the **first** command in the dialog (input after connection setup and LOGON prompt) or in an ENTER file, it is rejected in the event of an error (SC1 not 0) and the task aborted. The command return code cannot be evaluated in this case.

2.9 SHOW-TERMINAL-OPTIONS

Application area: JOB

Privileges: STD-PROCESSING
TSOS
HARDWARE-MAINTENANCE
SAT-FILE-EVALUATION
SAT-FILE-MANAGEMENT
SECURITY-ADMINISTRATION

Description of the function

The SHOW-TERMINAL-OPTIONS command allows users to request the output of the logical attributes of their terminals to SYSOUT.
The logical attributes of the terminal are defined in the system by means of class-2 system parameters at system generation time, through generation of the terminal, by the terminal itself and through VTSU operating parameters. These defined values are in force at the start of the interactive task; they can be modified within the user's task by means of the MODIFY-TERMINAL-OPTIONS command.

The command supports structured output into S variables (see the manual "BS2000/OSD-BC User Commands" Vol. 4 "Output to S variables" [2])

Format

SHOW-TERMINAL-OPTIONS

This command has no operands.

Command return code

(SC2)	SC1	Maincode	Meaning / guaranteed messages
	0	CMD0001	Command executed successfully
2	0	TIA0200	Information about attributes of the terminal nor available.
	32	CMD2009	Error during creation of S variables
	64	CMD0216	Privilege error
0	1	CMD0202	Syntax error

Output format

The output fields and the values shown correspond to the operands and operand values for the MODIFY-TERMIMAL-OPTIONS command.

Output field	Possible values	Explanation
ACKNOWLEGE-OUTPUT	YES / NO	Acknowledgment procedure
APL-CHARACTER-SET	YES / NO	Use of an APL character set
CODED-CHARACTER-SET	7-BIT / 8-BIT-DEFAULT / <name 1..8>	Code table
GRAPHICS	YES / NO	Use of graphic characters
HARDCOPY	NO / LOCAL / CENTRAL	Support for a hardcopy printer
LINE-END-CHARACTER	Smudge character (<x-string 1..1>)	LIne-end character
LINE-LENGTH	<integer 10..255>	Length of output lines
LOWER-CASE	YES / NO	Distinction between upper-/ lower-case letters
MAXIMUM-LINES	<integer 3..255>	Number of lines after which a screen overflow occurs
OVERFLOW-CONTROL	USER-ACKNOWLEDGE / NO-CONTROL / TIME(TIMEOUT=<integer 0..60>)	Nature of screen overflow control
WRITE-READ-MODE	MODIFIED-FIELDS / NO-FIELDS	Control of output representation and the nature of inputs
SUBSTITUTE-CHARACTER	Smudge character (<x-string 1..1>)	Replacement for undisplayable characters in output

Example

```
/show-term ——————————————————————————————————————————— (1)

%OVERFLOW-CONTROL     = USER-ACKNOWLEDGE
%MAXIMUM-LINES        =  24        LINE-LENGTH        =  80
%LINE-END-CHARACTER   = ? (X'00')  WRITE-READ-MODE    = MODIFIED-FIELDS
%LOWER-CASE           = YES        GRAPHICS           = NO
%APL-CHARACTER-SET    = NO         HARDCOPY           = LOCAL
%SUBSTITUTE-CHARACTER = ? (X'07')  ACKNOWLEDGE-OUTPUT = NO
%CODED-CHARACTER-SET  = 7-BIT

/mod-term ov-contr=*time(timeout=10) ——————————————————— (2)

/show-term ——————————————————————————————————————————— (3)

%OVERFLOW-CONTROL     = TIME(TIMEOUT = 10)
%MAXIMUM-LINES        =  24        LINE-LENGTH        =  80
%LINE-END-CHARACTER   = ? (X'00')  WRITE-READ-MODE    = MODIFIED-FIELDS
%LOWER-CASE           = YES        GRAPHICS           = NO
%APL-CHARACTER-SET    = NO         HARDCOPY           = LOCAL
%SUBSTITUTE-CHARACTER = ? (X'07')  ACKNOWLEDGE-OUTPUT = NO
%CODED-CHARACTER-SET  = 7-BIT
/
```

(1) Outputs the logical attributes of the terminal.

(2) Overflow control is to be based on timing, with a delay of 10 seconds each time.

(3) Outputs the logical attributes of the terminal again.

3 TIAM macros

The macros are given in alphabetical order. Examples illustrating their use have been included after the operand descriptions. The VTSU macros are described in the VTSU User Manual 1, which contains all the details specific to VTSU.

3.1 Notational conventions

In order to make it as easy as possible to use this manual, most of the notational conventions used here correspond to those used in the rest of the BS2000 User Guides. They are explained in the following table:

Formal representation	**Explanation**	**Example**
UPPERCASE LETTERS	Uppercase letters denote constants and must be entered by the user as they stand.	DEVICE=DISKETTE Specify: DEVICE=DISKETTE
lowercase letters	Lowercase letters denote variables for which the user must substitute current values on input, i.e. their contents vary from one command to the next.	SECTORS=number Specify e.g. SECTORS=5 or SECTORS=10 etc.
{ }	Braces enclose alternatives, i.e. one of the values enclosed in braces may or must be selected.	TAPE= { YES / NO } Specify: TAPE=YES or TAPE=NO
[ ]	Brackets enclose optional entries, i.e. specifications that can be left out. If the comma is inside the brackets, it need only be written when the optional entry is used. If it is outside, however, it must be included, even when the optional entry is omitted. (Parentheses must be entered)	filename[,ERASE] Specify e.g. FILE,ERASE or FILE or XYZ,ERASE etc.

<u>....</u>	Underscoring indicates a default value. This is the value assumed by the system when no value is specified by the user.	[<u>SAM</u> / ISAM] Specify: SAM or ISAM or nothing (=SAM)
...	Ellipses indicate repetition. They show that the preceding specification may be given more than once.	(vsn,...) Specify e.g. (PVT003) or (PVT003,PVT456) etc.
()	An expression used to represent variables is enclosed in parentheses. This representation enables you to see the value range at a glance. Since this involves a number of characters, it is necessary to use this formal delimitation.	(0 < length < 9)
≤	Relationship between two values: The value on the left is less than or equal to the value on the right; the value on the right is greater than or equal to the value on the left.	number ≤ 2047 0 ≤ position
≥	The value on the left is greater than or equal to the value on the right; the value on the right is less than or equal to the value on the left.	number ≥ 1
< >	The values are greater than or less than, but not equal to, one another.	0 < length quantity < 9

Brief description of the macros

Macro	Meaning
CUPAB	Generates addressing aids (DSECTs) for the operand tables of the RDATA, WROUT and WRTRD macros.
RDATA	Reads a record from the system file SYSDTA
TCHNG	Changes the characteristics of the virtual terminal
TSTAT	Supplies information on the interactive terminal' s attributes.
WROUT	Transfers a record to the system file SYSOUT.
WRTRD	Sends a message to the terminal in interactive mode and then reads a message from the terminal; see also CUPAB.

3.2 CUPAB (communication user parameter block)

The CUPAB macro allows the user to address the fields and flags in the operand tables of the RDATA, WROUT and WRTRD calls symbolically. CUPAB generates a dummy section (DSECT) for this purpose in 24-bit addressing mode. In order to generate 31-bit DSECTs, the RDATA, WRTRD, WROUT macros must be used with MF=D (MF=D,prefix).

Name	Operation	Operands
[name]	CUPAB	D

name — If an entry is specified in the name field, it is used as the DSECT name. If the name field is empty, CUPAB is generated as the DSECT name.

D — This entry causes a dummy section (DSECT) to be generated. If it is omitted, an MNOTE message is output. The generation of the DSECT is not affected, however.

The field names for the operand table of the RDATA macro are as follows:

Field name	**Byte**	**Meaning**
CURAREAW	0-3	Full word containing CUREDIT1 and CURAREA
CUREDIT1	0	Input edit option byte 1
CURAREA	1-3	Address of user input area
CURFTB	4	Flag table byte
CUREDIT2	5	Input edit option byte 2
CURALEN	6-7	Length of user input area
CURERRW	8-11	Full word containing CURACI and CURERROR
CURACI	8	Assignment change indicator for SYSDTA
CURERROR	9-11	Error return address
L@RDATAB		Length of RDATA operand table

The field names for the operand table of the WROUT macro are as follows:

Field name	**Byte**	**Meaning**
CUWMSGW	0-3	Full word containing CUWEDIT1 and CUWMSG
CUWEDIT1	0	Output edit option byte 1
CUWMSG	1-3	Address of message in application program
CUWERRW	4-7	Full word containing CUWEDIT2 and CUWERROR
CUWEDIT2	4	Output edit option byte 2
CUWERROR	5-7	Error return address
L@WROUTB		Length of WROUT operand table

The field names for the operand table of the WRTRD macro are as follows:

Field name	Byte	Meaning
CUBMSGW	0-3	Full word containing CUBOEDT1 and CUBMSG
CUBOEDT1	0	Output edit option byte 1
CUBMSG	1-3	Address of message output area
CUBAREAW	4-7	Full word containing CUBIEDT1 and CUBAREA
CUBIEDT1	4	Input edit option byte 1
CUBAREA	5-7	Address of input area
CUBOEDT2	8	Output edit option byte 2
CUBIEDT2	9	Input edit option byte 2
CUBALEN	10-11	Length of input area
CUBERRW	12-15	Full word containing CUBERROR
reserved	12	-
CUBERROR	13-15	Error return address
L@WRTRDB		Length of WRTRD operand table

Symbolic constants for edit option bytes 1 and 2

In addition to the field names for the RDATA, WROUT, and WRTRD operand tables CUPAB also defines symbolic constants (EQU values for ASSEMBLER) for the values of the edit, edit1 and edit2 operands.

The following tables give an overview of the names of the symbolic constants defined by CUPAB, their current values, their equivalents in the symbolic edit operands when MODE= is specified and their validity in the different modes.

a) Output edit option byte 1

CUPAB name	Bit	Corresponding MODE operand	Valid (X) or mandatory (1) in MODE=			
			COMP	LINE	FORM	PHYS
CWR1CODE	2^0	OTRSUP=	X	0	0	0
CWR1LNET	2^1	OLINEND=	X	0	1	1
	2^2	reserved for MODE=	0	1	0	1
CWR1RSET	2^3	OMANUAL=	X	0	0	0
CWR1HOM	2^4	OHOM=	0	X	0	0
CWR1PTPE	2^5	OPTAPE=	X	0	0	0
	2^6	reserved for MODE=	0	0	1	1
CWR1HARD	2^7	OHCOPY=	X	X	0	X

b) Output edit option byte 2

CUPAB name	**Bit**	**Corresponding MODE operand**	**Valid (X) or mandatory (1) in MODE=**			
			COMP	LINE	FORM	PHYS
CWR2HDR	2^0	OHDR=	X	0	1	X
CWR2NOLC	2^1	ONOLOGC=	0	X	0	0
CWR2EXT	2^2	EXTEND=	0	X	0	0
CWR2INFO	2^3	OINFO=	0	X	0	0
	2^4	reserved	0	0	0	0
CWR2POSN	2^5	ONOPOSN=	0	X	0	0
CWR2TRAN	2^5	OTRANS=	0	0	0	X
CWR2BEL	2^6	OBELL=	0	X	0	0
CWR2ETB	2^7	OETB=	0	0	0	X

c) Input edit option byte 1

CUPAB name	**Bit**	**Corresponding MODE operand**	**Valid (X) or mandatory (1) in MODE=**			
			COMP	LINE	FORM	PHYS
CRD1CODE	2^0	ITRSUP=	X	0	0	X
CRD1LNET	2^1	ILINEND=	X	0	1	1
CRD1BACK	2^2	IGETBS=	X	X	X	X
CRD1RSET	2^3	IMANUAL=	X	0	0	0
CRD1LCT	2^4	ILCASE=	X	X	X	X
	2^5	reserved for MODE=	0	1	0	1
	2^6	reserved for MODE=	0	0	1	1
CRD1HDR	2^7	IHDR=	X	0	1	X

d) Input edit option byte 2

CUPAB name	**Bit**	**Corresponding MODE operand**	**Valid (X) or mandatory (1) in MODE=**			
			COMP	LINE	FORM	PHYS
CRD2GFC	2^0	IGETFC=	0	X	0	0
	2^1	reserved	0	0	0	0
CRD2CFD	2^2	ICFD=	0	X	0	0
CRD2GIC	2^3	IGETIC=	0	X	0	0
	2^4	reserved	0	0	0	0
CRD2EXT	2^5	EXTEND=	0	X	0	0
	2^6 and 2^7	reserved	0	0	0	0

The following applies to all non-reserved bits:

MODE operand	Associated bit
= Y	set (1)
= N	reset (0)

The meaning of the MODE operands (and associated bits of the I/O edit option bytes) is given in the operand descriptions of the RDATA, WROUT and WRTRD macros.

Example

```
                          EXTERNAL SYMBOL DICTIONARY
BCUPAB   START
         CUPAB D
*
*               COMMUNICATIONS USER PARAMETER BLOCK
*
CUPAB    DSECT
*
*               RDATA PARAMETER BLOCK
*
CURAREAW DS    0A                       WORD NAME FOR CURAREA
*
CUREDIT1 DC    AL1(0)                   EDIT OPTION 1
CURAREA  DC    AL3(0)                   READ AREA ADDRESS
*
CURFTB   DC    AL1(0)                   FLAG TABLE BYTE
CUREDIT2 DC    AL1(0)                   EDIT OPTION 2
CURALEN  DC    H'0'                     READ AREA LENGTH
*
CURERRW  DS    0A                       WORD NAME FOR CURERROR
*
CURACI   DC    AL1(0)                   ASSIGNMENT CHANGE INDICATOR
CURERROR DC    AL3(0)                   ERROR RETURN ADDRESS
*
L@RDATAB EQU   *-CURAREAW               RDATA PARAMETER BLOCK LENGTH
*
*               WROUT PARAMETER BLOCK
*
         ORG   CURAREAW
*
CUWMSGW  DS    0A                       WORD NAME FOR CUWMSG
*
CUWEDIT1 DC    AL1(0)                   EDIT OPTION 1
CUWMSG   DC    AL3(0)                   OUTPUT MESSAGE ADDRESS
*
CUWERRW  DS    0A                       WORD NAME FOR CUWERROR
*
CUWEDIT2 DC    AL1(0)                   EDIT OPTION 2
CUWERROR DC    AL3(0)                   ERROR RETURN ADDRESS
*
L@WROUTB EQU   *-CUWMSGW                WROUT PARAMETER BLOCK LENGTH
*
*               WRTRD PARAMETER BLOCK
*
         ORG   CURAREAW
*
CUBMSGW  DS    0A                       WORD NAME FOR CUBMSG
*
CUBOEDT1 DC    AL1(0)                   OUTPUT EDIT OPTION 1
CUBMSG   DC    AL3(0)                   OUTPUT MESSAGE ADDRESS
*
CUBAREAW DS    0A                       WORD NAME FOR CUBAREA
*
CUBIEDT1 DC    AL1(0)                   INPUT EDIT OPTION 1
CUBAREA  DC    AL3(0)                   READ AREA ADDRESS
*
CUBOEDT2 DC    AL1(0)                   OUTPUT EDIT OPTION 2
CUBIEDT2 DC    AL1(0)                   INPUT EDIT OPTION 2
```

```
CUBALEN  DC    H'0'                    READ AREA LENGTH
*
CUBERRW  DS    0A                      WORD NAME FOR CUBERROR
*
         DC    AL1(0)
CUBERROR DC    AL3(0)                  ERROR RETURN ADDRESS
L@WRTRDB EQU   *-CUBMSGW               WRTRD PARAMETER BLOCK LENGTH
*
*
*              OUTPUT EDIT OPTION BYTE 1 - NAMES FOR BIT SETTINGS
*
CWR1CODE EQU   1               OTRSUP : CODE TRANSLATION
CWR1LNET EQU   2               OLINEND: LINE END TREATMENT
CWR1MBT1 EQU   4                        MODE BIT 1
CWR1RSET EQU   8               OMANUAL: RESET FROM PAPER TAPE
CWR1HOM  EQU   16              OHOM   : HOMOGENEOUS OUTPUT
CWR1PTPE EQU   32              OPTAPE : PAPER TAPE CONTROL
CWR1MBT2 EQU   64                       MODE BIT 2
CWR1HARD EQU   128             OHCOPY : HARDCOPY
CWR1MMSK EQU   68         MODE=        MODE MASK
CWR1COMP EQU   0                COMP   : COMPATIBLE MODE
CWR1LINE EQU   4                LINE   : LINE MODE
CWR1FORM EQU   64               FORM   : FORMAT MODE
CWR1PHYS EQU   68               PHYS   : PHYSICAL MODE
*
*              OUTPUT EDIT OPTION BYTE 2 - NAMES FOR BIT SETTINGS
*
CWR2HDR  EQU   1               OHDR   : HEADER PRESENT
CWR2NOLC EQU   2               ONOLOGC: NO LOGIC CHARS INTERPRET.
CWR2EXT  EQU   4               EXTEND : EXTENDED LINE OUTPUT
CWR2INFO EQU   8               OINFO  : INFORMATIVE MESSAGE
CWR2POSN EQU   32              ONOPOSN: NO POSITIONING
CWR2TRAN EQU   32              OTRANS : TRANSPARENT MODE
CWR2BEL  EQU   64              OBELL  : ACCOUSTIC ALARM
CWR2ETB  EQU   128             OETB   : ETB INSTEAD OF ETX
*
*              INPUT EDIT OPTION BYTE 1 - NAMES FOR BIT SETTINGS
*
CRD1CODE EQU   1               ITRSUP : CODE TRANSLATION
CRD1LNET EQU   2               ILINEND: LINE END TREATMENT
CRD1BACK EQU   4               IGETBS : BACKSPACE
CRD1RSET EQU   8               IMANUAL: RESET TO MANUAL CONTROL
CRD1LCT  EQU   16              ILCASE : LOWER CASE TRANSLATION
CRD1MBT1 EQU   32                       MODE BIT 1
CRD1MBT2 EQU   64                       MODE BIT 2
CRD1HDR  EQU   128             IHDR   : HEADER REQUIRED
CRD1MMSK EQU   96          MODE=        MODE MASK
CRD1COMP EQU   0                COMP   : COMPATIBLE MODE
CRD1LINE EQU   32               LINE   : LINE MODE
CRD1FORM EQU   64               FORM   : FORMAT MODE
CRD1PHYS EQU   96               PHYS   : PHYSICAL MODE
*
*              INPUT EDIT OPTION BYTE 2 - NAMES FOR BIT SETTINGS
*
CRD2GFC  EQU   1               IGETFC : GET FUNCTION CODE
CRD2CFD  EQU   4               ICFD   : CONFIDENTIAL INPUT DATA
CRD2GIC  EQU   8               IGETIC : GET IDENTITY CARD
CRD2EXT  EQU   32              EXTEND : EXTENDED LINE INPUT
               *,CUPAB      002    910215   53121055
         END
```

3.3 RDATA (read data from SYSDTA)

This macro reads the next data record from SYSDTA. The file assigned as SYSDTA may be a floppy disk unit, an element in a PLAM library, an SDF-P variable, a cataloged SAM or ISAM file or the terminal running the task. The record (or message in the case of a terminal) is transferred to an application program area as a variable-length record.

The CUPAB macro generates a DSECT for the operand table of the RDATA macro for the call in 24-bit addressing mode.

The RDATA macro call accepts the MF operand (see section on "Format of generated statements" in the "Executive Macros" manual 3).

The PARMOD operand controls the macro expansion. In this way either the 24-bit interface or the 31-bit interface can be generated.

For the 31-bit addressing mode the following should be noted:

– At the beginning of the operand table a standard header is generated (see section on "Format of generated statements" in the "Executive Macros" manual 3). The user must ensure that this header is correctly initialized. If dynamically generated operand lists are used (access via CSECT/DSECT), no symbolic names or "equates" are generated. Hence, when the operand list is filled dynamically, the initialization values for the standard header must be taken from an operand list that was generated with MF=L.

– A CSECT/DSECT is generated with MF=C/D or MF=(C,p)/(D,p).

  p = prefix (max. 3 characters; a longer prefix is truncated to 3 characters);
  default p = CUR. The prefix changes only the field name, not the symbolic names in the equates.

Note on compatibility

The RDATA macro as of TIAM version V10.0A (in conjunction with BS2000 V10.0A) generates the header version 2 in the standard header. This macro cannot therefore be executed under BS2000 versions prior to V10.0. In versions earlier than V10.0, it branches to the error exit.
If a program is to execute using several BS2000 versions, it must be written for the earliest version (only upward compatibility is guaranteed).

Format 1

```
Operation     Operands
--------------------------------------------------------------------------------
RDATA         record,error[,length][,edit][,A]

                              ⎧ 24[,MF=⎧ L       ⎫]       ⎫
                              ⎪        ⎩ (E,...) ⎭        ⎪
                              ⎪                           ⎪
              [,PARMOD=       ⎨ 31[,MF=⎧ C       ⎫]       ⎬]
                              ⎪        ⎪ (C,p)   ⎪        ⎪
                              ⎪        ⎪ D       ⎪        ⎪
                              ⎪        ⎪ (D,p)   ⎪        ⎪
                              ⎪        ⎪ L       ⎪        ⎪
                              ⎩        ⎩ (E,...) ⎭        ⎭

              [,KEYOUT=⎧N⎫]
                       ⎩Y⎭

              [,KEYPOS=⎧N⎫] [,KEYLEN=⎧N⎫]
                       ⎩Y⎭           ⎩Y⎭

              ⎧,MODE=COMP [,ITRSUP= ⎧N[O]  ⎫] [,ILINEND=⎧N[O]  ⎫]⎫
              ⎪                     ⎩Y[ES] ⎭            ⎩Y[ES] ⎭ ⎪
              ⎪           [,ILCASE= ⎧Y[ES] ⎫] [,IHDR=   ⎧N[O]  ⎫]⎪
              ⎪                     ⎩N[O]  ⎭            ⎩Y[ES] ⎭ ⎪
              ⎪           [,IGETBS= ⎧Y[ES] ⎫]                    ⎪
              ⎪                     ⎩N[O]  ⎭                     ⎪
             [⎨,MODE=LINE [,ILCASE= ⎧N[O]  ⎫] [,IGETBS= ⎧N[O]  ⎫]⎬]
              ⎪                     ⎩Y[ES] ⎭            ⎩Y[ES] ⎭ ⎪
              ⎪           [,IGETFC= ⎧N[O]  ⎫] [,IGETIC= ⎧N[O]  ⎫]⎪
              ⎪                     ⎩Y[ES] ⎭            ⎩Y[ES] ⎭ ⎪
              ⎪           [,ICFD=   ⎧N[O]  ⎫]                    ⎪
              ⎩                     ⎩Y[ES] ⎭                     ⎭
              [,RC=⎧OLD⎫]
                   ⎩NEW⎭

              [,VTSUCBA=addr]

              [,TIMER=value]
--------------------------------------------------------------------------------
```

The operands edit, ITRSUP, ILINEND, MODE, RC, VTSUCBA and TIMER are only evaluated if SYSDATA is assigned to the terminal. The operands KEYOUT and KEYPOS are only evaluated if SYSDATA is assigned to an ISAM file.

The operands RC, VTSUCBA and TIMER are only supported for the 31-bit interface.

record — Symbolic address of the field into which the data record to be read is transferred. The field begins with the record length field.

Record format:

Bytes	0-1:	length of record + 4 bytes record length field
	2-3:	reserved
	4-n:	data record

Example

```
RECORD   DS   0CL74
LENGTH   DS   CL2
RESERVE  DS   CL2
DATA     DS   CL70
```

error — Symbolic address in the application program to which the program branches:

– in the event of an error or

– if operand A is set.

In the event of an error, register R14 contains the address of the command following the RDATA call. The error code is transferred to register R15.
31-bit interface: if error = 0 (address X'00..0') is specified, the program is resumed with the instruction following the RDATA call.

length — Size of the read area, including 4 bytes for the record length field. The maximum length is 32767 bytes. If this entry is omitted, the length attribute of the read area is assumed.

edit — Edit option for message input from the terminal. This operand is not required when using standard functions (all edit bits = 0), when making a MODE entry or when using the VTSU control block. Only the first edit byte for input can be set to the meaning given under the CUPAB macro by specifying it directly (X'xx').

Note

This operand is still supported for reasons of compatibility.
The edit options should be controlled using MODE specifications (see below) or via the VTSU control block (VTSUCBA)

A

The application program is notified of the initial standard assignment and all subsequent assignments of SYSDTA. Notification is given via the error address as soon as reading is completed. The assignment key is returned in 31-bit mode in the CURAIND field of the RDATA operand table and in 24-bit mode in the leftmost byte of register R15.

PARMOD

Controls the macro expansion. Either the 24-bit interface or the 31-bit interface is generated. If PARMOD is not specified, the macro expansion is performed in accordance with the specification for the GPARMOD macro (see "Executive Macros" manual 3) or the default for ASSEMBLER (= 24-bit addressing).

=24

The 24-bit interface is generated. Data lists and instructions use 24-bit addresses (address space ≤ 16 Mbytes).

=31

The 31-bit interface is generated. Data lists and instructions use 31-bit addresses (address space ≤ 2 Gbytes).
Data lists begin with the standard header.

MF

=L

Generates a parameter list; fields are filled in accordance with parameter specifications.

=(E,...)

Generates the instruction list.

=C

Generates a CSECT.

=(C,p)

Generates a CSECT; a prefix p with a maximum length of 3 bytes is assigned; default: p = CUR.

=D

Generates a DSECT.

=(D,p)

Generates a DSECT; a prefix p with a maximum length of 3 bytes is assigned; default: p = CUR.

KEYOUT

This operand is evaluated only if SYSDTA is assigned to an ISAM file.

=Y

The ISAM key is removed before the record is transferred to the application program (see note below).

=N

The ISAM key is not removed.

KEYPOS — This operand is evaluated only if SYSDTA is assigned to an ISAM file.

=Y — 24-bit mode:

The position of the ISAM key (less 1) is stored in register 0 as follows:

– in the two middle bytes if KEYLEN=Y

– in the two rightmost bytes if KEYLEN=N.

31-bit mode:

The position of the ISAM key (less 1) is stored in the CURKEYP field (RDATA operand list).

=N — The ISAM key position is not stored.

KEYLEN — This operand is evaluated only if SYSDTA is assigned to an ISAM file.

=Y — 24-bit mode:

The length of the ISAM key (less 1) is stored in the rightmost bytes of register 0 (see note below).

31-bit mode:

The length of the ISAM key (less 1) is stored in the CURKEYP field (RDATA operand list).

=N — The length of the ISAM key is not stored.

Note

Any character string specified starting with a 'Y' has the same meaning as 'Y'. Any character other than 'Y' is interpreted as if 'N' were specified; also any character string not beginning with 'Y' has the same effect as 'N' and causes a MNOTE message to be issued.

The following operands are interpreted only if SYSDTA is assigned to the data display terminal:

The MODE specifications, together with the edit options, are only supported for reasons of compatibility. They are now incorporated in the VTSU control block (VTSUCB). Any future additions will be made only to VTSUCB.

MODE

=COMP
Compatible operating mode. Via symbolic operands the application program can use all the edit options defined in the edit option table (see CUPAB macro). Any entries made in the 'edit' operand are ignored. Control characters in the text are passed unchecked to the application program. This operating mode is compatible with earlier versions (terminals supported: 8110, 8151, and 8152). For other terminals this mode is treated as MODE=LINE. All edit options except ILCASE are rejected (RS: X'08').

=LINE
If SYSDTA is a terminal, it is treated as a virtual line terminal. The message can be structured with the aid of logical control characters (see VTCSET macro). The device-specific message header is not supplied. The operands for message editing are interpreted.
If SYSDTA is a cataloged file or floppy disk unit - i.e. not a terminal – the message editing operands are not interpreted. Lowercase letters, for instance, are not converted into upper case letters when SYSDTA is a cataloged file and ILCASE has the value NO (default). Logical control characters in the message are not interpreted either.

ITRSUP

=YES
The translation of device code to EBCDIC is suppressed, i.e. the application program receives the message in device code.

=NO
The translation of device code to EBCDIC is not suppressed; the application program receives the message in EBCDIC.

ILINEND

=YES
The carriage return/line feed characters are passed to the application program.

=NO
The carriage return/line feed characters are not passed to the application program.

IGETBS

=YES
Underline (X'6D') characters are passed to the application program without being interpreted by the system (printer terminals only).

=NO	Underlines are not passed to the application program; instead, the correction function (backspace) is performed by the system (printer terminals only).
ILCASE	
=YES	Lowercase letters are also passed to the application program.
=NO	Lowercase letters are passed to the application program as upper case.
IHDR	
=YES	The entire message prefix is passed to the application program.
=NO	The message prefix is not passed to the application program.
IGETFC	
=YES	Byte 5 of the read area is to contain the standardized function key code. The latter identifies the terminal key used to initiate data transmission at the terminal. A table of the standardized function key codes is given on page 197 in the appendix.
=NO	No function key code is to be passed.
IGETIC	
=YES	Input is to be read from the connected ID card reader. The input data must consist of ID card information or the short code K14. This specification is only permitted for 8160, 9749, 975x, 9763 and 3270 terminals with a defined ID card reader (see also the macro TSTAT TYPE=TCHAR). The IGETIC operand is ignored if ICFD=YES is also specified, or if no ID card reader is connected. The source of input remains unchanged.
=NO	The source of input is not to be changed.
ICFD	
=YES	The input data is confidential and must not be visible on the terminal. Depending on the type of terminal, this is achieved by blanking, by clearing the screen, or by overwriting the input line (on printer terminals). When the screen is cleared the screen format is reset to 24x80.
=NO	No measures are to be taken to protect confidential data.

RC — This operand is only permitted for the 31-bit interface.

=NEW — The return code is stored in register 15 and in the standard header (with full 4-byte return code). A 4-byte return code is returned only if SYSDTA reads from the data display terminal. In all other cases only a 1-byte return code is returned, regardless of the value of the return code.

=OLD — The return code is stored in register 15. The three leftmost bytes are masked out.

VTSUCBA

=addr — Address of a VTSUCB generated with MF=L. This operand is only permitted for the 31-bit interface. If the VTSUCBA operand is used, the MODE operand and subsequent EDIT options are ignored. Their value is set to X'FF' in the parameter list. All the required EDIT options must be specified in VTSUCB.
Default: VTSUCB is not used.

TIMER — This operand is permitted only for the 31-bit interface.

=value — Defines a maximum waiting time for data entry. If no entry is registered within this time, a return code is issued. The waiting time is specified in seconds.
The default is UNLIMITED, which means that no timer is used.

Format 2

Operation	Operands
RDATA	(1)

(1) — Register R1 contains the address of the operand list. The table must be aligned on a word boundary.

Functional description

When format 1 of the macro is executed, the specified operands are stored in the operand table and the start address of this table is loaded into register 1. For format 2 the table specified in the application program is used.

Structure of the operand list

Addressing mode	Byte	Contents
24-bit mode	0	Input edit byte 1, in accordance with edit option table for input.
	1 - 3	Address of the field to which the record read in is to be transferred (read area operand).
	4	Flag; bit 2^7=1/0 corresponds to KEYOUT=Y/N bit 2^6=1/0 corresponds to KEYPOS=Y/N bit 2^5=1/0 corresponds to KEYLEN=Y/N
	5	Input edit byte 2, in accordance with edit option table for input.
	6 - 7	Max. length of the record to be read in (' length' operand).
	8	SYSDTA assignment indicator, bit 2^0=1 corresponds to operand A.
	9 - 11	Address branched to in the event of an error (' error' operand).
31-bit mode	0 - 7	Standard header (for format see the section on "Format of generated statements" in the "Executive Macros" manual [3]). The initialization values can be obtained from an operand list generated with MF=L. If RC=OLD, no return code is passed in the standard header.
	8 - 11	Address branched to in the event of an error (' error' operand).
	12 - 15	Address of the field to which the record read in is to be transferred (' read area' operand).
	16	Input edit byte 1 } in accordance with equates in the makro, generated with MF=D.
	17	Input edit byte 2 }
	18	Assignment key for SYSDTA; equates in macro generated with MF=D.
	19	Flag indicating use of the VTSUCB and return code behavior; equates in macro generated with MF=D.
	20 - 21	Max. length of the record to be read (' length' operand).
	22	Flag edit option byte of the ISAM key
	23	SYSDTA assignment indicator (operand A)
	24 - 25	Position of the ISAM key.
	26 - 27	Length of the ISAM key.
	28 - 31	Address of the VTSUCB
	32 - 33	Values of the timer
	34 - 35	Reserved (X' 0000')

The values for input edit bytes 1/2 can be found in the table given under the CUPAB macro (24-bit mode) or in a data list generated with MF=C/D (31-bit mode).

Note

If a BREAK is detected during the read operation, the P1 program counter is reset to the beginning of the macro expansion, causing the macro call to be repeated after interrupt processing.

Return information (acknowledgments) and error flags

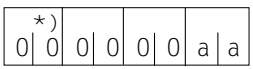
– Any edit options illegal for a particular device or the operating mode MODE= are corrected by the system as far as possible.

– During macro processing register R1 contains the address of the operand list.

– A return code is passed in register R15 concerning execution of the macro RDATA.

a) 24-bit mode:

Register R15:



*) if necessary, assignment key when operand A is specified.

X'aa'	Explanation
X'00'	Normal termination
X'04'	Unrecoverable error
X'08'	Operand error
X'0C'	Record truncation; record length ≥ specified length
X'10'	/EOF
X'14'	SYSDTA not assigned
X'18'	Volume access error
X'20'	Invalid edit option byte; error corrected by the system
x'38'	Problem in connection with POSIX

b) 31-bit mode:

When the 31-bit interface is used, the return codes can be stored in register 15 (RC=OLD) or in both register 15 and the standard header (RC=NEW).

RC=OLD

Register R15:

In addition to the return codes described under a) (right-justified, left 0), the return code X’24’ (VTSUCB error), X’34’ (timer expired) and the global system values may also occur.

RC=NEW

Error messages are entered in register 15 and in the standard header, as shown in the following table (see "Macro expansion" for the MAINCODE and SUBCODE equates).

SUBCODE		**MAINCODE**		**Meaning**
2	1	2	1	
X' 00'	X' 00'	X' 00'	X' 00' X' 20'	Successful processing – no additional information – operand error, corrected by TIAM/VTSU
X' 00'	X' 00'	X' 00'	X' 14' X' 18'	SYSFILE error – SYSDTA not assigned – volume access error
X' 00' X' 07' X' 08'	X' 01'	X' 00'	X' 08' X' 08' X' 08'	Parameter error – operand error, no further specification – RESERVED fields not 0 – timer setting out of range, not from 10 to 3600 seconds
X' 00' X' 02' X' 05' X' 06'	X' 20'	X' 00'	X' 04' X' 04' X' 04' X' 04'	Internal error – no further specification – BCAM message lost – input message too long – negative transport acknowledgment
X' 00'	X' 40'	X' 00'	X' 0C' X' 10' X' 34'	No class reaction – truncation of input record – end of file – timer timed out (no input within waiting time defined by timer setting)
X' 01'	X' 80'	X' 00'	X' 04'	Internal bottleneck – BCAM bottleneck
X' 09' X' 0A'	X' 80' X' 40'	X' 00' X' 00'	X' 38' X' 38'	Errors in conjunction with POSIX – I/O serialization error – If the LOGON task is in system mode, processes created by fork() cannot effect I/Os
from VTSUCB	from VTSUCB	from VTSUCB	X' 24'	VTSU has detected error (for MAINCODE 1 see error information in VTSUCB header, described in the VTSU manual [1])

Notes

– A record is truncated if the record being read exceeds the specification in the length operand. Only as much of the record as corresponds to the length operand value is transferred to the read area; the remainder is lost. If the record is shorter than the read area, it is entered left-justified. The rest of the read area is not blank-filled. The program continues without an error flag.

– Floppy disks:
When SYSDTA is assigned to a floppy disk unit of the type FD3170, the records read by the RDATA macro must not exceed 128 bytes. When SYSDTA is assigned to a floppy disk unit of the type FD3171, the records read by the RDATA macro must not exceed 1024 or 2048 bytes.

– An end-of-file condition can occur as follows:

  Terminals:
  If the ESCAPE function is activated (switchover from program mode to system mode) and an EOF command issued followed by /RESUME.

  Other input media:

  a) When the files SYSDTA and SYSCMD are the same, and a command (other than BREAK) is read.

     Note

     A command is a record beginning with '/', where '/' must not be replaced by a symbolic operand.

  b) When the files SYSDTA and SYSCMD are not the same, and /EOF is detected in columns 1 - 4 of the record.

– The "SYSDTA not assigned" flag appears when all the records have been read from the floppy disk and a new RDATA macro is issued.

– The "volume access error" flag appears when SYSDTA is assigned to a floppy disk unit and the desired floppy disk is not accessible.

– Assignment of SYSDTA

  a) 24-bit mode:

  In the leftmost byte of register R15 an indicator showing the assignment of SYSDTA is passed.

Register R15:

b) 31-bit mode:

The SYSDTA assignment key is returned in the field CURAIND of the RDATA parameter list. The RDATA CSECT/DSECT contains corresponding equates.

Assignment key

X'bb'	Explanation
X'00'	SYSDTA unchanged
X'01'	SYSDTA is a floppy disk unit
X'04'	SYSDTA is a sequential file
X'08'	SYSDTA is an indexed-sequential file
X'14'	SYSDTA is a terminal
X'18'	SYSDTA is an SDF-P variable
X'20'	SYSDTA is an element in a PLAM library

Macro expansion

```
RDATAP   RDATA MF=C,PARMOD=31
*
*              RDATA PARAMETER BLOCK
*
RDATAP   FHDR  MF=(C,*NO-NAMES)
*
RDATAP   DS    A               0   INTERFACE IDENTIFIER
         DS    A               4   GENERAL RETURN CODE
*
*        MAIN RETURN CODES AS IN FHDR, ADDITIONAL VALUES:
CURURE   EQU  4            UNRECOVERABLE ERROR
CUROPE   EQU  8            OPERAND ERROR
CURCUTE  EQU 12            MESSAGE IS CUT
CUREOF   EQU 16            END OF FILE
CURNDTA  EQU 20            NO SYSDTA   1
CURACCE  EQU 24            ACCESS ERROR
CUROPEC  EQU 32            OPERAND ERROR, CORRECTED
CURVTE   EQU 36            VTSU ERROR 1
CURTOUT  EQU 52            TIME OUT DURING READ
CURPOSE  EQU 56            ERROR IN POSIX CONTEXT
*        SUB1 RETURN CODE AS IN FHDR
*        SUB2 RETURN CODES:
CURRCNS  EQU  0            RC_S2_NOT_SPECIFIED
CURBCSH  EQU  1            BCAM SHORTAGE
CURBCLL  EQU  2            BCAM LETTER LOST
CURMTL   EQU  5            MESSAGE TOO LONG
CURNACK  EQU  6            NEGATIVE TACK 1
CURRFNZ  EQU  7            RESERVED FIELDS NOT ZERO
CURWT    EQU  8            WRONG TIMER
CURLOCE  EQU  9            SERIALIZATION LOCK ERROR
CURFTIO  EQU  10           NO FORKED TASK IOS POSSIBLE.
*
CURERRW  DC    A(0)                ERROR RETURN ADDRESS
CURAREAW DC    A(0)                READ AREA ADDRESS
```

```
CUREDIT1 DC    AL1(0)               EDIT OPTION 1
CUREDIT2 DC    AL1(0)               EDIT OPTION 2
CURAIND  DC    AL1(0)               SYSDTA ASSIGNMEN
CURANO   EQU   X'00'                -  NO CHANGE
CURADISC EQU   X'01'                -  FLOPPY DISC
CURASAM  EQU   X'04'                -  SAM FILE
CURAISAM EQU   X'08'                -  ISAM FILE
CURACARD EQU   X'0C'                -  CARD READER (OBSOLETE)
CURADSS  EQU   X'14'                -  TERMINAL
CURAVAR  EQU   X'18'                -  SDF-P VARIABLE
CURAPLAM EQU   X'20'                -  PLAM LIBRARY
CURFB1   DC    AL1(0)               FLAG BYTE 1
CURVTSI  EQU   X'80'                -  VTSU INDICATOR
CURRCIND EQU   X'40'                -  RC INDICATOR
CURALEN  DC    H'0'                 READ AREA LENGTH
CURFTB   DC    AL1(0)               FLAG TABLE BYTE
CURACI   DC    AL1(0)               ASSIGNMENT CHANGE INDICATOR
CURKEYP  DC    H'0'                 KEY-POSITION
CURKEYL  DC    H'0'                 KEY-LENGTH
CURVTSU  DC    A(0)                 VTSUCB ADDRESS
CURTVAL  DC    H'0'                 INPUT TIMER VALUE
CURRTIA  DC    H'0'                 RES_FOR_TIAM
*
CURL RDA EQU   *-RDATAP              RDATA PARAMETER BLOCK LENGTH
*
*              INPUT EDIT OPTION BYTE 1 - NAMES FOR BIT SETTINGS
*
CRD1CODE EQU   1                ITRSUP : CODE TRANSLATION
CRD1LNET EQU   2                ILINEND: LINE END TREATMENT
CRD1BACK EQU   4                IGETBS : BACKSPACE
CRD1RSET EQU   8                IMANUAL: RESET TO MANUAL CONTROL
CRD1LCT  EQU   16               ILCASE : LOWER CASE TRANSLATION
CRD1MBT1 EQU   32                        MODE BIT 1
CRD1MBT2 EQU   64                        MODE BIT 2
CRD1HDR  EQU   128              IHDR   : HEADER REQUIRED
CRD1MMSK EQU   96         MODE=          MODE MASK
CRD1COMP EQU   0                COMP   : COMPATIBLE MODE
CRD1LINE EQU   32               LINE   : LINE MODE
CRD1FORM EQU   64               FORM   : FORMAT MODE
CRD1PHYS EQU   96               PHYS   : PHYSICAL MODE
*
*              INPUT EDIT OPTION BYTE 2 - NAMES FOR BIT SETTINGS
*
CRD2GFC  EQU   1                IGETFC : GET FUNCTION CODE
CRD2CFD  EQU   4                ICFD   : CONFIDENTIAL INPUT DATA
CRD2GIC  EQU   8                IGETIC : GET IDENTITY CARD
CRD2EXT  EQU   32               EXTEND : EXTENDED LINE INPUT
```

Example 1a

This example uses format 1 without the VTSUCB.

```
RDATAT          START         0
*
                BALR          10,0
                USING         *,10
*
                RDATA INPUT1,FEHL,40,PARMOD=31,MODE=LINE,ILCASE=Y,ICFD=Y
*
ERR             NOP           0
                TERM
*
INPUT1          DS            0CL40
LENGTH          DS            CL2
UNUSED          DS            CL2
DATA            DS            CL36
*
                END
```

Example 1b

This example uses format 1 with the VTSUCB.

```
RDATAT          START         0
*
                BALR          10,0
                USING         *,10
*
                RDATA         INPUT1,ERR,40,PARMOD=31,VTSUCBA=VTSUPAR
*
ERR             NOP           0
                TERM
*
VTSUPAR         VTSUCB        MODE=LINE,LOW=YES,SPECIN=C
*
INPUT1          DS            0CL40
LENGTH          DS            CL2
UNUSED          DS            CL2
DATA            DS            CL36
*
                END
```

Example 2

Use of format 2

```
RDATA2          START         0
*
                BALR          10,0
                USING         *,10
*
                LA            1,PARAM
                RDATA         (1),PARMOD=31
*
END             NOP           0
                TERM
*
                DS            0F
PARAM           RDATA         INPUT,END,40,MF=L,PARMOD=31,MODE=LINE,  +
                              ILCASE=Y,ICFD=Y
*
INPUT           DS            0CL40
LENGTH          DS            CL2
UNUSED          DS            CL2
DATA            DS            CL36
*
                END
```

3.4 TCHNG (change terminal characteristics)

This macro enables the application program to change the characteristics of a virtual terminal. The TCHNG call remains in effect until the application program is terminated or until the next TCHNG call is issued. It applies to I/Os between the program and the terminal via the RDATA, WROUT and WRTRD macros. All values not specified in the TCHNG macro remain unchanged.

```
Operation | Operands
----------+-----------------------------------------------------------------
TCHNG     | [OFLOW={SYS  }][,EDOPT=DYN]
          |        {USER }
          |
          |                ,MODE=LINE[,OHCOPY={N}][,OHOM={N}][,OINFO={N}]
          |                                  {Y}           {Y}         {Y}
          |                          [,ONOPOSN={N}][,OBELL={N}][,IGETBS={N}]
          |                                    {Y}         {Y}         {Y}
          | [,EDOPT=STAT{            [,ILCASE={N}][,IGETFC={N}]            }]
          |                                  {Y}         {Y}
          |                          [,IGETIC={N}][,ICFD= {N}]
          |                                    {Y}        {Y}
          |                ,MODE=FORM [,IGETBS={N}][,ILCASE={Y}]
          |                                    {Y}         {N}
          |
          | [,SUB={OUT  }]
          |       {OUTIN}
          |
          | [,INFOLIN={N[O]  }]
          |           {Y[ES] }
          |
          | [,CLEAR={Y[ES]}]
          |         {N[O] }
```

OFLOW

=SYS — For long application program output the system is to perform an overflow check to prevent information overflow at the terminal. The terminal user specifies the type of system overflow check using the system command MODIFY-TERMINAL-OPTIONS (for default values and effect, see the section on this command).

=USER — The system is not to take any precautions to prevent information overflow during long application program outputs. Overflow control can be performed individually by the application program.

Note

Operating system messages are still subject to the overflow check defined by the system or using the MODIFY-TERMINAL-OPTIONS command.

EDOPT

=DYN — The edit option values specified in subsequent RDATA, WROUT and WRTRD calls are to be evaluated (system presetting).

=STAT — This parameter is ignored when the VTSUCB is used.

The edit option values specified in the TCHNG macro for MODE, OBELL, OHCOPY, OHOM, OINFO, ONOPOSN, IGETBS, ILCASE, IGETFC, IGETIC and ICFD are to be used in any subsequent RDATA, WROUT and WRTRD macros. This definition remains valid until a further change is made via TCHNG or until the end of program. If a program run is interrupted (with K2), the specified edit option values also apply to all outputs which occur during this interruption via WROUT and WRTRD (some system components use WROUT).

MODE=
OHCOPY=
OHOM=
OBELL=
ONOPOSN=
OINFO=
IGETBS=
ILCASE=
IGETFC=
IGETIC=
ICFD= — The meaning of these operands is given under the WRTRD and WROUT macros. If an operand is omitted, the respective underlined value is assumed.

SUB

=OUT — In line mode inputs, the defined substitute character is passed to the application program.

=OUTIN — In line mode inputs, the defined substitute character is passed to the application program as the logical control character SUB.

INFOLIN

=YES — On terminals without a hardware-based system line, any messages for a special information line (OINFO=YES or MODE=INFO) are to be displayed in the last line on the screen if the screen already contains a formatted or physical message (see WROUT macro).

=NO — On terminals without a system line, any messages for a special information line are to be displayed like normal line mode messages.

CLEAR

=YES — The screen is to be cleared and reset to the standard screen format whenever the output mode changes, e.g.

LINE → FORM
FORM → LINE, COMP
PHYS → LINE, FORM, COMP
COMP → FORM

If the mode is changed at a time other than after input, the system waits t seconds before clearing the screen to give the user time to read the previous output (see command MODIFY-TERMINAL-OPTIONS OVERFLOW-CONTROL=).

=NO — The screen is not cleared when the output mode changes and the system does not wait before outputting the next message.

Note

The application program must include specifications to ensure that the screen contents do not interfere with any of the functions of the new mode, e.g. by means of premodified fields.

Note

When the program changes the screen format, any logical number of lines previously defined by the user is retained in the new format.

Return information (acknowledgments) and error flags

A return switch (RS) is entered in the rightmost byte of register 15. The remaining three bytes are deleted.

RS	
X’00’	Normal termination
X’04’	Unrecoverable error
X’08’	Operand error
X’0C’	Calling task not a timesharing task
X’10’	Operand error
X’14’	Invalid edit option specification

3.5 TSTAT (terminal status)

With this macro the user can request information concerning the terminal. The generated device type is supplied.

The MF operand is accepted (see the section on "Format of generated statements" in the "Executive Macros" manual [3]).

A CSECT/DSECT is generated with MF=C/D or MF=(C,p)/(D,p). p = prefix (max. length 3 characters; any longer prefix is truncated to 3 characters); default: p = TST.

```
Operation | Operands
----------+-----------------------------------------------------------
TSTAT     | {TCHAR  }
          | |PHDIM  |
          | |LIDIM  |
          | |VDT[YP]|
          | |EDOPT  |
          | |OFLOW  |
          | {STNAM  },aerea[,length][,MF={L     }]
          | |PRNAM  |                    {(E,..)}
          | |ALL    |
          | |MONCS  |
          | |PERPH  |
          | |BASIC  |
or - - -  | - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
          |
TSTAT     |     {C     }
          |     |(C,p)|
          |  MF={      }
          |     |D     |
          |     |(D,p) |
----------+-----------------------------------------------------------
```

TCHAR	Queries the physical type of the terminal. The type given is the one under which the terminal was generated in PDN.
PHDIM	Queries the physical terminal attributes (line mode).
LIDIM	Queries the virtual terminal attributes (line mode).
VDT[YP]	Queries the virtual type of the terminal.
EDOPT	Queries the static edit options.
OFLOW	Queries the type of screen overflow control.
STNAM	Queries the station name.
PRNAM	Queries the processor name.

ALL	Requests all the terminal information except MONCS, PERPH and BASIC.
MONCS	Requests information on the monitor and on the character sets available on the terminal.
PERPH	Requests information about the connected peripherals.
BASIC	Requests basic information about the terminal.
area	Symbolic address of an area in which the requested information is to be stored. The area must be aligned on a halfword boundary.
length	Specifies the size of ’area’: 64 bytes for ALL 14 bytes for MONCS 64 bytes for BASIC for further details refer to the description of the DCSTA macro in the VTSU manual [1] 8 bytes for all other options If this specification is omitted, the "area" length attribute is used. If ALL, BASIC or MONCS is specified and the area is too small to accept all of the terminal information supplied, only as much information as fits into the area is made available.
MF	
=L	Generates a parameter list; fields are filled according to the parameter specifications.
=(E,...)	Generates the instruction list.
=C	Generates a CSECT.
=(C,p)	Generates a CSECT; a prefix p with a maximum length of 3 bytes is assigned; default: p = TST.
=D	Generates a DSECT.
=(D,p)	Generates a DSECT; a prefix p with a maximum length of 3 bytes is assigned; default: p = TST.

Note

When using a CSECT/DSECT, the fields of the parameter list must be filled as follows:

TSTMTYP	always with TSTSTMAC,
TSTSTTYP	with the desired status type in accordance with the equates,
TSTMLEN	with the length of the output area in the application program ('length'),
TSTMADR	with the address of the output area in the application program ('area').

The TCHAR, PHDIM, LIDIM, VDTYP, EDOPT, OFLOW, STNAM, PRNAM, ALL, MONCS, PERPH and BASIC operands correspond to the equivalent operands described under the DCSTA macro (see the VTSU manual [1]).

Functional description

The area which is to accept the information can either be defined by the calling program itself or via the macro DCSTA C,... (see Example 2). In the first case, the program can address the receiving area with the aid of a DSECT generated by the DCSTA D,... macro. The symbolic names generated by the DCSTA macro are shown in the tables given in the description of that macro. The names in parentheses are not symbolic addresses but symbolic constants that make it easier to test individual bits (see Example 2).

Return information (acknowledgments) and error flags

A return switch (RS) is entered in the rightmost byte of register 15. The remaining three bytes are deleted.

RS	
X'00'	Normal termination
X'04'	Unrecoverable error
X'08'	Parameter error
X'0C'	No terminal available
X'10'	Destination field is not long enough. Only part of the information was transferred if the ALL, MONCS or BASIC operand was specified. With any other operand, no information at all is transferred
X'14'	The required information is (partially) unavailable

Macro expansion

```
                        EXTERNAL SYMBOL DICTIONARY
TSTAT1   START
         TSTAT MF=D
         #INTF INTNAME=TSTAT,INTCOMP=001,REFTYPE=REQUEST
TSTPAR   DSECT
*
*        TSTAT PARAMETER BLOCK
*
TSTMTYP  DC    AL1(0)              MACRO TYPE
TSTSTMAC EQU   1                   TSTAT-MACRO
*
TSTSTTYP DC    AL1(0)              STATUS TYPE REQUESTED:
*
         @STIN ,MF=D,PRE=TST
*
*        TSTAT EQUATES
*
TSTTCHAR EQU   1                     - TERMINAL CHARACTERISTICS
TSTPHDIM EQU   2                     - PHYSICAL TERMINAL ATTRRIBUTES
TSTLIDIM EQU   3                     - VIRTUAL TERMINAL ATTRIBUTES
TSTVDT   EQU   4                     - VIRTUAL DEVICE TYPE
TSTEDOPT EQU   5                     - STATIC EDIT OPTIONS
TSTOFLOW EQU   6                     - OVERFLOW CONTROL
TSTSTNAM EQU   7                     - STATION NAME
TSTPRNAM EQU   8                     - PROCESSOR NAME
TSTMONCS EQU   9                     - MONITOR AND CHARACTER SETS
TSTPERPH EQU   10                    - PERIPHERIE OF TERMINAL
TSTBASIC EQU   11                    - BASIC TERMINAL INFORMATIONS
TSTALL   EQU   255                   - ALL INFORMATION REQUESTED
*
             *,@STIN      999    921011   53531032
TSTMLEN  DC    H'0'                AREA LENGTH (MIN. 8 BYTES)
*
TSTMADR  DC    A(0)                AREA ADDRESS FOR TSTAT INFO
         END
```

Example 1

```
  TSTAT1   START
           PRINT NOGEN
           BALR  10,0
           USING *,10
    TSTATVDTYP,LOG,8  ------------------------------------------- (01)
    TSTATOFLOW,UEL,8  ------------------------------------------- (03)
  DTH1     TERM
             *,VERSION 900
  *
  LOG      DS    CL8
  UEL      DS    CL8
           END
```

```
/START-PROG $ASSEMBH
%  BLS0500 PROGRAM 'ASSEMBH', VERSION '1.1A00' OF '1992-04-30' LOADED
%  BLS0552 COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1990. ALL RIGHTS
%  RESERVED
%  ASS6010 V 1.1A00 OF BS2000 ASSEMBH-  READY
//COMPILE SOURCE=*L(L=LIB.EX.MANUAL,E=TSTAT1),MAC-LIB=$XXXX.LIB.M.SYSLIB.V11.0T10,
  MOD-LIB=LIB.EX.MANUAL(E=TSTAT1),LIST=P(OUT=*L(L=LIB.EX.MANUAL,E=TSTAT1)),TEST-SUP=YES
%  ASS6011 ASSEMBLY TIME: 1056 MSEC
%  ASS6018 0 FLAGS, 0 PRIVILEGED FLAGS, 0 MNOTES
%  ASS6019 HIGHEST ERROR-WEIGHT: NO ERRORS
%  ASS6006 LISTING GENERATOR TIME: 760 MSEC
//END
%  ASS6012 END OF ASSEMBH-
/LOAD-PROG *M(L=LIB.EX.MANUAL,E=TSTAT1),TEST-OPT=AID
%  BLS0517 MODULE 'TSTAT1' LOADED
%insert dth1
/RESUME-PROG
STOPPED AT LABEL: DTH1 , SRC_REF: 38 , SOURCE: TSTAT1 , PROC: TSTAT1
/%d log %xl8
** ITN: #'00080107' *** TSN: 2UXT ******************************************
CURRENT PC: 00000024    CSECT: TSTAT1    ************************************
V'00000036' = LOG      + #'00000000'
00000036 (00000000) 5B010000 00000000                      $.......  ---------- (02)
/%d uel %xl8
V'0000003E' = UEL      + #'00000000'
0000003E (00000000) 02060000 00000000                      ........  ---------- (04)
/RESUME-PROG
/MOD-JOB-OPT LOG=P(L=NO)
```

(01) The virtual device type of the terminal is queried.

(02) The task is running on a data display terminal (bit 2^6 is set) on which line mode, format mode and physical mode are permitted (bits 2^0, 2^1 and 2^3 are set).

(03) The type of screen overflow control is queried.

(04) The system (bit 2^5=0) controls the overflow. When the screen is full, the system requests an acknowledgment at the terminal before overwriting the screen (bit 2^1=1).

Example 2

```
                        EXTERNAL SYMBOL DICTIONARY
TSTAT2   START
         PRINT NOGEN
         BALR  3,0
         USING *,3
         TSTAT TCHAR,STATCHAR,8 ————————————————————————   (01)
             CLI   STADVTYP,STAD6180
         TSTAT PHDIM,PHYSFELD,8 ———————————————————————— (02)
         TSTAT LIDIM,FILIDIM,8  ————————————————————————   (03)
DTH1     TERM
               *,VERSION 100
         PRINT GEN
         DCSTA C,TYPE=TCHAR  ————————————————————————————  (04)
         IDLKG VER=002,ID=TCHAR,SECT=C,P=STA,ALIGN=D                 002
               *,VERSION 002
         CNOP  0,8
STATCHAR DS    0D
*
*              DEFINE TERMINAL CHARACTERISTICS FIELDS
*
STASTTCH DS    0D                       TERMINAL CHARACTERISTICS AREA
STAMNTCH DS    0D                       MINIMUM TERMINAL CHARICS. AREA
*
STAPTTYP DC    AL1(0)                   PARTNERTYPE
STADVTYP DC    AL1(0)                   DEVICE TYPE
STATCHR2 DC    AL1(0)                   TERMINAL CHARACTERISTICS BYTE 2
STATCHR3 DC    AL1(0)                   TERMINAL CHARACTERISTICS BYTE 3
STATCHR4 DC    AL1(0)                   TERM. CHARACTERISTIC BYTE 4 901
STATCHRS DC    AL1(0)                   TERM. CHAR FROM STATION     920
STACTRLU DC    AL1(0)                   CONTROL UNIT FOR PRINTER    701
STACHCAD DC    AL1(0)                   CENTRAL HARDCOPY ADDRESS
*
*              DEFINE PARTNER TYPES (PTTYP)
*
STADCAMP EQU   X'00'                  PARTNER IS A PROGRAM
STADCAMT EQU   X'01'                  PARTNER IS A TERMINAL
*
         DCDEVCH STA
*
*              DEFINE DEVICE TYPES (DVTYP)
*
STAD8103 EQU   X'02'                TELETYPE 8103
STAD8150 EQU   X'04'                VIDEO TERMINAL 8150
STAD8153 EQU   X'05'     *NO VTSU*  VIDEO TERMINAL 8153
STADHOST EQU   X'08'                INTELLIGENT PARTNER
STAD8151 EQU   X'15'                VIDEO TERMINAL 8151
STAD8152 EQU   X'16'                VIDEO TERMINAL 8152
STAD8110 EQU   X'17'                SS-8110
STAD6154 EQU   X'18'     *NO VTSU*  VIDEO 8161 54 CHAR PER LINE
STAD6164 EQU   X'19'     *NO VTSU*  VIDEO 8161 64 CHAR PER LINE
STAD6180 EQU   X'1A'     *NO VTSU*  VIDEO 8161 80 CHAR PER LINE
STAD8161 EQU   X'1A'     *NO VTSU*  VIDEO 8161
STAD8121 EQU   X'1C'                PRINTER STATION 8121
STADPT80 EQU   X'1D'     *AS 8103*  TELETYPE PT80
STAD1000 EQU   X'1E'     *AS 8103*  TELETYPE T1000
STADT100 EQU   X'23'     *AS 8103*  TELETYPE T100
STAD100E EQU   X'26'     *AS 8103*  FS100-E
STAD8122 EQU   X'2B'                PRINTER STATION 8122
```

```
STAD8162 EQU   X'2C'                VIDEO 8162
STAD8160 EQU   X'2D'                VIDEO 8160
STAD8124 EQU   X'2E'                PRINTER STATION 8124
STAD8167 EQU   X'2F'     *AS 8160*  VIDEO 8167
STADAP   EQU   X'30'     *AS HOST*  AP-STATION
STAD9750 EQU   X'35'                VIDEO 9750 OR 9749
STAD9003 EQU   X'36'                PRINTER STATION 9003
STAD9770 EQU   X'39'     *AS 8151*  DS 9770
STAD9002 EQU   X'3B'                PRINTER STATION 9002
STAD3974 EQU   X'3D'                VIDEO TERMINAL 3974
STAD9751 EQU   X'3F'     *AS 8160*  DSS 9751
STAD9752 EQU   X'40'     *AS 9750*  DSS 9752
STAD9753 EQU   X'41'     *AS 9750*  DSS 9753
STAD9001 EQU   X'42'                PRINTER 9001
STAD9731 EQU   X'43'     *AS 3974*  GRAFIC STATION 9731
STAD9004 EQU   X'45'                PRINTER 9004
STAD9754 EQU   X'4C'     *AS 8160*  VIDEO 9754
STAD9755 EQU   X'4E'                DSS-9755                          922
STAD9763 EQU   X'4F'                DSS-9763
STADBTXF EQU   X'55'     *AS HOST*  BTX-STATION T-3000 (FELDVERS.)
STADBTXE EQU   X'56'     *AS HOST*  BTX-EDITIER-STATION (DIENST)
STADBTXA EQU   X'57'     *AS HOST*  BTX-ABFRAGE-STATION (DIENST)
STADUTC  EQU   X'5A'                UTC FUER TELETEX
STAD9012 EQU   X'5B'                PRINTER 9012
STAD9013 EQU   X'5C'                PRINTER 9013
STAD3270 EQU   X'5E'                DSS-3270
STAD0131 EQU   X'65'                PRINTER 9001-31
STAD0189 EQU   X'66'                PRINTER 9001-8931
STAD9022 EQU   X'68'                PRINTER 9022
STAD1118 EQU   X'6B'                PRINTER 9011-18
STAD1119 EQU   X'6C'                PRINTER 9011-19
STAD3287 EQU   X'6E'                PRINTER 3287                      956
STADPCL  EQU   X'70'                PRINTERS PCL                      960
STAD9021 EQU   X'70'                PRINTERS 9021 / 9022-200, HP LJ960
STAD9014 EQU   X'72'                PRINTER 9014                      001
STAD9026 EQU   X'73'                PRINTER 9026 (HDLC,COMP.9025)     001
STADFE   EQU   X'78'                DSS-FE BOF-FRONT-END              999
*
*              DEFINE TERMINAL CHARACTERISTICS BYTE 2 (TCHR2) BITS
*
STATC2EX EQU   8                       SECONDARY CHARACTER SET
STATC2LC EQU   32                      LOWER CASE
STATC2DT EQU   64                      GERM KEYB WITH GERM NAT CHAR901
STATC2DF EQU   128                     BYTE 2 DEFINED
*
*              DEFINE TERMINAL CHARACTERISTICS BYTE 3 (TCHR3) BITS
*
STATC3H1 EQU   1                       HARDCOPY BIT 1  (LOCAL)
STATC3H2 EQU   2                       HARDCOPY BIT 2  (CENTRAL)
STATC3HC EQU   3                       HARDCOPY BITS
STATC3IC EQU   4                       IDENTITY CARD READER
STATC3FD EQU   8                       FLOPPY DISK
STATC3AP EQU   16                      APL CAPABILITY
STATC3GF EQU   32                      GRAPHICS
STATC3DZ EQU   64                      DEZENTRAL FORMATING
STATC3DF EQU   128                     BYTE 3 DEFINED
*
            DEFINE TERMINAL CHARACTERISTICS BYTE 4 (TCHR4) BITS
STATC4CO EQU   1                       4 COLOURS(ITALIC/HALFBRIGHT)
STATC4ZF EQU   2                       NEW ZAT AND FAT POSSIBLE    920
```

```
STATC4ST EQU   4                       STATUS QUERY POSSIBLE       920
STATC4HI EQU   8                       HARDWARE INFOLINE AVAILABLE 920
STATC4C8 EQU   16                      8 COLOURS                   953
STATC4HP EQU   32                      HP LASER JET II             954
STATC4DF EQU   128                     BYTE 4 DEFINED
*
*              DEFINE TERM CHAR FROM STATION   BYTE   (TCHRS) BITS  920
*
STATCSDT EQU   1                       GERMAN KEYBOARD             920
STATCSHC EQU   2                       LOCAL HARDCOPY PRINTER      920
STATCSIC EQU   4                       ID-CARD RAEDER              920
STATCSDF EQU   128                     TERM CHAR FROM STAT RECEIVED920
               *,DCDEVCH    001    920406   53113038
*
               *,DCSTA      002    920819   53531014
PHYSFELD DCSTA C,TYPE=PHDIM ------------------------------------------- (05)
PHYSFELD IDLKG VER=002,ID=PHDIM,SECT=C,P=STA,ALIGN=D               002
               *,VERSION 002
         CNOP  0,8
PHYSFELD DS    0D
*
*              DEFINE PHYSICAL TERMINAL ATTRIBUTES FIELDS
*
STASTPV  DS    0D                      PHYSICAL TERMINAL ATTR. AREA
STAMNPV  DS    0D                      MINIMUM PHYS. TERM. ATTR. AREA
*
STALLEN  DC    H'0'                    PHYSICAL LINE LENGTH
STANOLIN DC    H'0'                    PHYSICAL NUMBER OF LINES
STAMAXDB DC    H'0'                    MAX. PHYSICAL DEVICE BUFFER
         DC    2AL1(0)                 RESERVED FOR FUTURE DEVELOPMENT
               *,DCSTA      002    920819   53531014
  DCSTA C,FI,TYPE=LIDIM ------------------------------------------------ (06)
         IDLKG VER=002,ID=LIDIM,SECT=C,P=FI,ALIGN=D                002
               *,VERSION 002
         CNOP  0,8
FILIDIM  DS    0D
*
*              DEFINE VIRTUAL TERMINAL ATTRIBUTES FIELDS
*
FISTLV   DS    0D                      VIRTUAL TERMINAL ATTR. AREA
FIMNLV   DS    0D                      MINIMUM VIRTUAL TERM ATTR AREA
*
FILLLEN  DC    H'0'                    VIRTUAL LINE LENGTH
FILNOLN  DC    H'0'                    VIRTUAL NUMBER OF LINES
FILMAXB  DC    H'0'                    MAXIMUM VIRTUAL DEVICE BUFFER
         DC    2AL1(0)                 RESERVED FOR FUTURE DEVELOPMENT
               *,DCSTA      002    920819   53531014
         END
```

(01) The physical device type of the terminal is queried. The destination area generated by the macro DCSTA C,... is assigned the name STATCHAR by default. To illustrate this, field STADVTYP is tested by means of the symbolic constant STAD6180.

(02) The physical attributes of the terminal are queried. PHYSFELD is the name of the destination area chosen by the user.

(03) The logical attributes of the terminal are queried. In the name of destination area FILIDIM, the default prefix ’STA’ has been replaced by the prefix ’FI’.

(04) The destination area for information on the physical type is generated together with the symbolic constants for bit value testing.

(05) The destination area for information on the physical attributes is generated.

(06) The destination area for information on the logical attributes is generated. The prefix for the field names is to be ’FI’ (default: ’STA’).

```
/START-PROG $ASSEMBH
%  BLS0500 PROGRAM 'ASSEMBH', VERSION '1.1A00' OF '1992-04-30' LOADED
%  BLS0552 COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1990. ALL RIGHTS
%  RESERVED
%  ASS6010 V 1.1A00 OF BS2000 ASSEMBH-  READY
//COMPILE SOURCE=*L(L=LIB.EX.MANUAL,E=TSTAT2),MAC-LIB=$XXXX.LIB.M.SYSLIB.V11.0T10,
 MOD-LIB=LIB.EX.MANUAL(E=TSTAT2),LIST=P(OUT=*L(L=LIB.EX.MANUAL,E=TSTAT2)),TEST-SUP=YES
%  ASS6011 ASSEMBLY TIME: 1411 MSEC
%  ASS6018 0 FLAGS, 0 PRIVILEGED FLAGS, 0 MNOTES
%  ASS6019 HIGHEST ERROR-WEIGHT: NO ERRORS
%  ASS6006 LISTING GENERATOR TIME: 877 MSEC
//END
%  ASS6012 END OF ASSEMBH-
/LOAD-PROG *M(L=LIB.EX.MANUAL,E=TSTAT2),TEST-OPT=AID
%  BLS0517 MODULE 'TSTAT2' LOADED
%insert dth1
/RESU-PROG
STOPPED AT LABEL: DTH1 , SRC_REF: 52 , SOURCE: TSTAT2 , PROC: TSTAT2
/%d statchar %xl8
** ITN: #'00080107' *** TSN: 2UXT ****************************************
CURRENT PC: 00000038    CSECT: TSTAT2    ****************************************
V'00000050' = STATCHAR + #'00000000'
00000050 (00000000) 0135A081 88000000                      ...ah...   ── (07)
/%d physfeld %xl8
V'00000058' = PHYSFELD + #'u0000000'
00000058 (00000000) 00500018 0FF80000                      .&...8..   ── (08)
/%d filidim %xl8
V'00000060' = FILIDIM  + #'00000000'
00000060 (00000000) 00500018 07800000                      .&......   ── (09)
/RESU-PROG
/MOD-JOB-OPT LOG=P(L=NO)
```

(07) The task is running on a 9750 Data Display Terminal.

(08) X’0050’The physical line length is 80 characters.
X’0018’The physical number of lines is 24.
X’0FF8’The physical device buffer has a capacity of 4088 characters.

(09) X’0050’The logical line length is 80 characters (line mode).
X’0018’The logical number of lines is 24 (line mode).
X’0780’The logical character buffer has a capacity of 1920 characters (= 24 lines * 80 columns).

Further examples are given under the DCSTA macro in the VTSU manual [1].

3.6 WROUT (write data to SYSOUT)

The WROUT macro transfers a message to the file SYSOUT. The file declared as SYSOUT may be an element in a PLAM library, an SDF-P variable, a cataloged SAM or ISAM file or the terminal running the task. If the LOGGING=PAR(LISTING=YES) operand is specified in the SET-LOGON-PARAMETERS or /MODIFY-JOB-OPTION command, the results of the WROUT macro are also written to SYSLST.

The CUPAB macro generates a DSECT for the operand table of the WROUT macro for the call in 24-bit addressing mode.

The WROUT macro accepts the MF operand (see the section on "Format of generated statements" in the "Executive Macros" manual [3]).

The PARMOD operand controlling the macro expansion allows either the 24-bit or the 31-bit interface to be generated.

For the 31-bit addressing mode the following should be noted:

– At the beginning of the operand list a standard header is generated (see the section on "Format of generated statements" in the "Executive Macros" manual [3]). The user must ensure that this header is correctly initialized. If dynamically generated operand lists are used (access via CSECT/DSECT), no symbolic names or equates are created for the standard header. In such cases therefore the initialization values for the standard header must be taken from an operand list which was generated with MF=L.

– A CSECT/DSECT is generated with MF=C/D or MF=(C,p)/(D,p).

  p = prefix (max. 3 characters; any longer prefix is truncated to 3 characters};
  default p = CUW. The prefix only changes the field names, not the symbolic names in the equates.

Note on compatibility

The WROUT macro as of TIAM version V10.0A (in conjunction with BS2000 V10.0A) generates the header version 2 in the standard header. This macro cannot therefore be executed under BS2000 versions prior to V10.0. In versions earlier than V10.0, it branches to the error exit.
If a program is to execute using several BS2000 versions, it must be written for the earliest version (only upward compatibility is guaranteed).

Format 1

Operation	Operands
WROUT	record,error[,edit] [,PARMOD={24[,MF={L | (E,...)}] | 31[,MF={C | (C,p) | D | (D,p) | L | (E,...)}]}] [{,MODE=COMP [,OTRSUP={N[O] | Y[ES]}] [,OLINEND={N[O] | Y[ES]}] [,OHCOPY={N[O] | Y[ES]}] [,OHDR={N[O] | Y[ES]}] ,MODE=LINE [,OHCOPY={N[O] | Y[ES]}] [,OHOM={N[O] | Y[ES]}] [,OINFO={N[O] | Y[ES]}] [,ONOPOSN={N[O] | Y[ES]}] [,OBELL={N[O] | Y[ES]}] [,ONOLOGC={N[O] | Y[ES]}] [,EXTEND={N[O] | Y[ES]}] ,MODE=FORM ,MODE=PHYS [,OHDR={N[O] | Y[ES]}] [,OETB={N[O] | Y[ES]}] [,OTRANS={N[O] | Y[ES]}]}] [,RC={OLD | NEW}] [,VTSUCBA=addr]

The operands edit, MODE, RC and VTSUCBA are only evaluated if SYSOUT is assigned to the terminal.

The RC and VTSUCBA operands are only supported for the 31-bit interface.

record

Symbolic address of the record to be output. The record begins with the record length field followed by the feed control character and message to be output.

Format:

Byte 0-1: length of message + 4 bytes record length field; the length must be greater than 5.

2-3: reserved

4: line-feed control character; only interpreted for printer output

5-n: message

Example

```
record     DC   Y(recordend record)
           DS   CL2                    reserved byte
           DC   X'nm'                  print feed character
           DC   C'data record'         record to be transferred
record end EQU  *
```

Note

If the system file SYSOUT is a cataloged file, records longer than 2044 bytes are truncated (register R15=X'0C') and only the first portion of the record is written to the file. In EAM files records longer than 2040 bytes are also truncated.

error

Symbolic address of an error routine in the application program which is branched to in the event of an error.

In the event of an error, register R14 contains the address of the command following the WROUT call. The error code is passed to register R15.

31-bit interface: if error = 0 (address X'00..0') is specified, the program is resumed with the instruction following the WROUT call.

edit

Edit options for the message to be transferred to the terminal. Editing is skipped if SYSOUT is not a terminal. This operand is not required when default options are used (all edit bits = 0), when

MODE is specified or when the VTSU control block is used. Only the first output edit option byte can be set to the meaning given under the CUPAB macro by direct specification (X'xx').

Note

This operand is now supported for reasons of compatibility only. The edit options should be controlled via MODE specifications (see below) or via the VTSU control block (VTSUCBA)

PARMOD — Controls the macro expansion. Either the 24-bit interface or the 31-bit interface is generated. If PARMOD is not specified, the macro expansion is performed in accordance with the specification for the GPARMOD macro (see the "Executive Macros" manual [3]) or the default for ASSEMBLER (= 24-bit addressing).

=24 — The 24-bit interface is generated. Data lists and instructions use 24-bit addresses (address space ≤ 16 Mbytes).

=31 — The 31-bit interface is generated. Data lists and instructions use 31-bit addresses (address space ≤ 2 Gbytes).
Data lists begin with the standard header.

MF

=L — Generates a parameter list; fields are filled in accordance with parameter specifications.

=(E,...) — Generates the instruction list.

=C — Generates a CSECT.

=(C,p) — Generates a CSECT; a prefix p with a maximum length of 3 bytes is assigned; default: p = CUW.

=D — Generates a DSECT.

=(D,p) — Generates a DSECT; a prefix p with a maximum length of 3 bytes is assigned; default: p = CUW.

The MODE specifications, together with the edit options, are only supported for reasons of compatibility. They are now incorporated in the VTSU control block (VTSUCB). Any future additions will be made only to VTSUCB.

MODE — This operand is evaluated only if SYSDTA is assigned to the data display terminal.

=COMP — Compatible operating mode. Any specifications made directly in the 'edit' operand are ignored. Control characters are permitted in the output message, but they are not checked by the system. This mode is compatible with earlier operating system versions.
On 8160, 8162, 9749, 975x, 9763, 3270 and X.29 terminals, this mode is treated like MODE=LINE. Edit option OLINEND is ignored. Edit options OTRSUP and OHDR are rejected (RS: X'08').

=LINE — The current terminal is to be handled as a virtual line or page terminal. The message can be structured with the aid of logical control characters (see VTCSET macro). If SYSOUT is not a terminal, only the logical control characters NL and NP are interpreted, e.g. for printer output in batch mode.
No other control characters are permitted. If any occur, the system converts them to a substitute character defined by the user (see the MODIFY-TERMINAL-OPTIONS SUBSTITUTE-CHARACTER= command).

=FORM — Format mode. The application program uses the software component FHS or format handling, which also edits the message in a form suitable for output on a particular terminal.

=PHYS — The message is to be output at the terminal "physically", i.e. without being edited by the system. This allows special device functions to be executed for which the LINE or FORM mode is insufficient. If none of the valid edit options are specified, the system prefixes the message with a standard device-specific message header.

OTRSUP

=YES — Translation of the message is suppressed. The program must supply the message in device code.

=NO — Default. Message translation from EBCDIC to device code is not suppressed, i.e. the program supplies the message in EBCDIC and the system converts it to device code.

OLINEND

=YES — The message is output to the terminal without CR/LF control characters supplied by the system. The line feed must be controlled by the application program.

=NO — Each message output to a terminal always starts on a new line. The necessary control characters are either prefixed to the message by the system or, if the terminal type requires it, inserted in the message when the physical end of line is reached.

OHCOPY

=YES — The message output to a data display terminal is simultaneously output to a hardcopy unit (printer) connected to the terminal. The hardcopy unit (printer) must be generated or assigned by means of MODIFY-TERMINAL-OPTIONS.

3270 terminals:
The contents of the entire screen are printed out on the hardcopy unit. As a result, earlier inputs and outputs may also be printed out. In the event of several consecutive outputs, the hardcopy function is only initiated for the last output. A hard copy is produced only if a hardcopy unit was generated for the data display terminal at connection setup.

If OINFO=YES/MODE=INFO or EXTEND=YES/MODE=EXTEND was specified, there is no hardcopy output.

If OHCOPY=YES/MODE=YES is specified, and the message contains the logical control character SPA, EPA, CHS or NUM, only the last unprotected portion of the message is printed, not the entire message.

If OVERFLOW-CONTROL=NO (MODIFY-TERMINAL-OPTIONS command) is also specified, it may happen that only part of the output is printed out on the hardcopy unit.

=NO — No hard copy is produced.

OHDR

=YES

The message has a special user header (to be specified in ASCII code) which the system prefixes to the output text. The length of the message header + 1 must be specified as a binary value in the first byte of the message.

3270 terminals:
The message header is specified in EBCDIC and comprises the CMD byte and the WCC byte. It must be preceded by a byte with the contents X'01' (length of the TRANSDATA message header +1).

Note

When output is sent to 8160, 975x, 9763 Data Display Terminals and their attached local printers, neither the system (MODE=LINE or COMP) nor FHS (MODE=FORM) uses a message header (PARAM0, PARAM1), but instead relies on parameter entries (PAG). The differences between the two modes are discussed in the manuals on the data display terminals and printers.

=NO

The message header (in ASCII code) is prefixed to the output text by the system.

OHOM

=YES

Only valid for 816x, 9749, 975x and 9763 Data Display Terminals.

The message is to be output in unstructured form, i.e. the entire message is treated as a single, homogeneous output unit. The message length is limited by the size of the system's output buffer.

Effect in mode 1:

The entire message - provided it contains no logical display control characters - can be transmitted back by modifying one character.

Note

If SYSOUT is assigned to a file, the WROUT macro is not executed and SYSLST logging is suppressed.

=NO

The message is to be output as a structured message, i.e. in non-homogeneous form so that one logical line is regarded as the output unit. There is no limit to the message length as long as the logical lines do not exceed 255 characters.

Effect in mode 1:

Individual logical lines can be modified separately and thus transmitted back selectively.

OINFO

=YES — The message can be displayed in a special information line on the terminal, without the risk of destroying important data.

This operand is intended primarily for application programs sending "asynchronous" messages to terminals when the information currently displayed is not known.

The message is displayed

– protected, in a hardware system line (e.g. 9749, 975x and 976x DDTs) or

– protected, in the last line of the screen (e.g. 816x, 9751, 9753 and 3270 DDTs), if this was specified in the application program (see TCHNG macro, INFOLIN operand) following output with MODE=FORM or MODE=PHYS.

– in all other cases, like a normal line-mode message.

If the message is longer than one screen line, it is split up and output line by line. The system observes the wait period specified in the /MODIFY-TERMINAL-OPTIONS OVERFLOW-CONTROL=TIME command.

In conjunction with OINFO=YES, the entry OHCOPY=YES is ignored, i.e. neither the information line nor the screen contents are printed.

The hardware system line is reset after the next input.

=NO — The message is not output in the special information line.

ONOPOSN

=YES — The message begins at the start of the current line (valid for printer terminals only).

=NO — The message begins at the start of the next line.

OBELL

=YES — Output is accompanied by an audible alarm at the end of a message (only on 9749, 975x, 9763, 816x and 3270 terminals with special bell hardware option).

=NO — No audible alarm sounds during output.

ONOLOGC

=YES — Logical control characters are not interpreted. All characters less than X'40' in the EBCDI code are replaced by SUB. Only characters greater than or equal to X'40' are accepted.

=NO — All logical control characters are interpreted and special physical control characters are accepted (see VTCSET, e.g. ESC, DC4). Other characters less than X'40' are replaced by SUB. Characters greater than or equal to X'40 are accepted.

EXTEND

=YES — (For 9749, 975x, 9763, 816x and 3270 terminals only)

This option supports the use of protected and unprotected fields by way of the logical control characters EPA, DAR, NUM and SPA (see VTCSET macro).

Output of text is protected and displayed at low intensity as the default option. The message can be structured by logical control characters (see VTCSET). With 3270 terminals it must be noted that logical control characters take up space on the screen. However, a number of consecutive logical control characters require only one character position. Areas in which the terminal user may enter data begin with EPA, DAR or NUM and end with SPA.

In input/output NUL is treated as a valid character; it is sent from the program to the terminal and vice versa. With 3270 terminals it must be noted that null characters are not transmitted to the computer. VTSU-B pads fields which are returned in shortened form by adding null characters until the original length is reached. The user thus always receives output-length fields.

The beginning of an output message is displayed at the start of the line following the cursor position. Before the first text character is displayed, the screen is cleared from the cursor position if the message does not begin with VPA. If the end of screen is reached during output, output continues from the start of screen. This continuation is unprotected in each case. Screen overflow control is disabled.
The keys RU, EFZ, AFZ and LSP are locked.

All other edit options apart from OBELL, ILCASE and IGETFC are ignored. (For programming notes see page 96 below.)

=NO — Operator prompting is performed by the system. Only the input request by the system or the application program is protected against overwriting.

Output is unprotected and displayed at low intensity.

Any null characters in the output text are converted to the substitute character; NULs in an input are removed.

Depending on the mode, the screen may be cleared from the cursor position before the start of an output.

OTRANS

=YES — The output data is to be transmitted "transparently", i.e. it may consist of any binary characters (5, 7 or 8 bits per character depending on the device code) and is not converted on the transmission path. If the transmission path has not been generated as a "potentially transparent" path, the output is rejected with the return code X'04'.

=NO — The output data is to be transmitted in "standardized" form, i.e. code conversion does take place.

OETB

=YES — The output message for the terminal is terminated with the control character ETB.

=NO — The output message for the terminal is terminated with the control character ETX.

RC — This operand is only permitted for the 31-bit interface.

=NEW — The return code is stored in register 15 and in the standard header (with full 4-byte return code). A 4-byte return code is returned only if SYSDTA reads from the data display terminal. In all other cases only a 1-byte return code is returned, regardless of the value of the return code.

=OLD — The return code is stored in register 15. The three leftmost bytes are masked out.

VTSUCBA — This operand is evaluated only if SYSDTA is assigned to the data display terminal. Furthermore, it is permissible only for the 31-bit interface.

=addr — Address of a VTSUCB generated with MF=L. If the VTSUCBA operand is used, the MODE operand and subsequent EDIT options are ignored. Their value is set to X'FF' in the parameter list. All the required EDIT options must be specified in VTSUCB (see the VTSU manual [1]).
Default: VTSUCB is not used.

Programming notes on the use of the operand MODE=EXTEND or MODE=LINE with EXTEND=YES (for the 3270 see appendix, page 199)

If MODE=EXTEND or EXTEND=YES is specified in line mode, the use of formats is possible without the need for format handling.

– If the user wishes to work in the same way as with formats, the first output must start with NP so as to clear the screen and enable the user to start the text in position (1.1).

– NL permits positioning to the start of the next line as well as clearing the remainder of the screen; VPAn positions to the start of line n without clearing the rest of the screen.

– Positioning within a line can only be achieved by means of text, blanks or NULs.

– Text following VPAn, NL and CHS is protected and displayed at low intensity.

– Unprotected fields are generated with 'EPA text SPA'.

  Numeric fields are generated with 'NUM text SPA'.

– VPAn at the end of a message enables the cursor to be positioned to the start of the first unprotected field in line n without destroying the screen contents. If no unprotected field starts in line n, the cursor is positioned to the start of the first unprotected field after line n.
  If there is no VPAn at the end of a message, the cursor is set to the first unprotected field on the screen.

– Continuation output / screen update

  NP produces a new screen display.

  VPAn at the beginning of the output changes line n on the screen. VPAn permits one or more lines to be skipped. In the current line a dark area is generated from the cursor position to the end of line or a field before it. Next the cursor is positioned to line n. After the update the cursor should again be positioned with VPAn as otherwise the screen would be cleared from the cursor position (see above). If NL is used during an update, the screen is also cleared from the cursor position. To prevent this, VPAn should always be used for moving to a new line.

Format 2

Operation	Operands
WROUT	(1)

(1) Register R1 contains the address of the operand list. The list must be aligned on a word boundary.

Functional description

Format 1: When the macro is executed, the specified operands are stored in an operand table and the start address of this table is loaded into register 1.
Format 2: The table specified in the application program is used.

Format of operand list

Addressing mode	Byte	Contents
24-bit mode	0	Output edit byte 1, in accordance with edit option table for output.
	1 - 3	Address of the record to be output (' record' operand).
	4	Output edit byte 2, in accordance with edit option table for output.
	5 - 7	Address branched to in the event of an error (' error' operand).
31-bit mode	0 - 7	Standard header (for format see the section on "Format of generated statements" in the "Executive Macros" manual [3]) The initialization values can be obtained from an operand list generated with MF=L. If RC-OLD, no return code is passed in the standard header.
	8 - 11	Address branched to in the event of an error (' error' operand).
	12 - 15	Address of the record to be output (' record' operand).
	16	Output edit byte 1}
		} in accordance with equates in
	17	Output edit byte 2} the macro, generated with MF=D.
	18	Reserved (X' 00')
	19	Flag indicating use of VTSUCB and for return code behavior; equates in macro generated with MF=D.
	20 - 23	VTSUCB address
	24 - 27	Reserved (X' 00000000')

The values for output edit bytes 1/2 can be found in the table given under the CUPAB macro (24-bit mode) or a data list generated with MF=C/D (31-bit mode).

Note

If the system file SYSOUT is a cataloged file, records longer than 2044 bytes are truncated (register 15 = X'0C') and only the first portion of each such record is written to the file. In EAM files records longer than 2040 bytes are truncated (register 15 = X'OC').

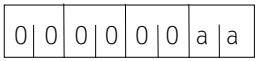
Return information (acknowledgments) and error flags

– During macro processing register R1 contains the address of the operand list.

– A return code is passed in register R15 concerning execution of the macro WROUT.

a) 24-bit mode:

Register R15:



If the macro is correctly executed, register R15 remains unchanged.

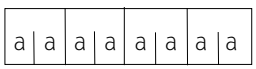
X' aa'	Explanation
X' 04'	Unrecoverable error
X' 08'	Operand error
X' 0C'	Truncation of output record. The record contents (excluding the record length field) exceed the size of the I/O buffer: for data display terminals depends on the device type and network generation; for output to a cataloged file it is 2044 bytes.
X' 10'	BREAK was issued at the terminal during execution of the macro.
X' 20'	Invalid edit option; corrected by the system.
X' 38'	Error in conjunction with POSIX.

b) 31-bit mode:

When using the 31-bit interface, the return codes can be stored either in register 15 (RC=OLD) or in both register 15 and the standard header (RC=NEW).

RC=OLD

Register R15:



If the macro is correctly executed, register R15 is cleared.

In addition to the return codes described under a) (right-justified, left 0), the return code X'00000000' (normal termination), the return code X'24' (error in VTSUCB) and the global system values may also occur.

RC=NEW

Error messages are stored both in register 15 and in the standard header, see following table (for the equates for MAINCODE and SUBCODE, see macro expansion).

SUBCODE		MAINCODE		Meaning
2	1	2	1	
X' 00'	X' 00'	X' 00'		Successful processing
			X' 00'	– no additional information
			X' 20'	– operand error, corrected by TIAM/VTSU
	X' 01'	X' 00'		Parameter error
X' 00'			X' 08'	– operand error, no further specification
X' 07'			X' 08'	– RESERVED fields not 0
	X' 20'	X' 00'		Internal error
X' 00'			X' 04'	– no further specification
X' 02'			X' 04'	– BCAM message lost
X' 06'			X' 04'	– negative transport acknowledgment
X' 00'			X' 28'	– problems with memory assignment *)
X' 00'	X' 40'	X' 00'		No class reaction
			X' 0C'	– output message truncated
			X' 10'	– BREAK during execution
	X' 80'	X' 00'		Internal bottleneck
X' 01'			X' 04'	– BCAM bottleneck
				Error in conjunction with POSIX
X' 09'	X' 80'	X' 00'	X' 38'	– I/O serialization error
X' 0A'	X' 40'	X' 00'	X' 38'	– If the LOGON task is in system mode, processes created by fork() cannot effect I/Os
from VSUCB	from VSUCB	from VTSUCB	X' 24'	VTSU has detected error (for MAINCODE 1 see error information in the VTSUCB header)

*) SYSOUT is assigned to a file which fully occupies the memory on primary assignment and fails to assign space in memory on secondary assignment.

Macro expansion

```
WROUTP   WROUT MF=C,PARMOD=31
*
*              WROUT PARAMETER BLOCK
*
WROUTP   FHDR  MF=(C,*NO-NAMES)
*
WROUTP   DS    A       0   INTERFACE IDENTIFIER
         DS    A       4   GENERAL RETURN CODE
*        MAIN RETURN CODES AS IN FHDR, ADDITIONAL VALUES:
CUWURE   EQU  4            UNRECOVERABLE ERROR
CUWOPE   EQU  8            OPERAND ERROR
CUWCUTE  EQU 12            MESSAGE IS CUT
CUWBRE   EQU 16            BREAK
CUWOPEC  EQU 32            OPERAND ERROR, CORRECTED
CUWVTE   EQU 36            VTSU ERROR
CUWSATE  EQU 40            SPACE SATURATION ERROR
CUWPOSE  EQU 56            ERROR IN POSIX CONTEXT
*        SUB1 RETURN CODES AS IN FHDR
*        SUB2 RETURN CODES:
CUWRCNS  EQU  0            RC_S2_NOT_SPECIFIED
CUWBCSH  EQU  1            BCAM SHORTAGE
CUWBCLL  EQU  2            BCAM LETTER LOST
CUWNACK  EQU  6            NEGATIVE TACK
CUWRFNZ  EQU  7            RESERVED FIELDS NOT ZERO
CUWLOCE  EQU  9            SERIALIZATION LOCK ERROR
CUWFTIO  EQU  10           NO FORKED TASK IS POSSIBLE
*
CUWERRW  DC    A(0)                ERROR RETURN ADDRESS
CUWMSGW  DC    A(0)                OUTPUT MESSAGE ADDRESS
CUWEDIT1 DC    AL1(0)              EDIT OPTION 1
CUWEDIT2 DC    AL1(0)              EDIT OPTION 2
CUWRES01 DC    AL1(0)              RESERVED
CUWFB1   DC    AL1(0)              FLAG BYTE 1
CUWVTSI  EQU   X'80'               - VTSU INDICATOR
CUWRCIND EQU   X'40'               - RC INDICATOR
CUWVTSU  DC    A(0)                VTSUCB ADDRESS
CUWRTIA  DC    F'0'                RES_FOR_TIAM
*
CUWL@WRO EQU   *-CUWHEAD           WROUT PARAMETER BLOCK LENGTH
*
*              OUTPUT EDIT OPTION BYTE 1 - NAMES FOR BIT SETTINGS
*
CWR1CODE EQU   1              OTRSUP : CODE TRANSLATION
CWR1LNET EQU   2              OLINEND: LINE END TREATMENT
CWR1MBT1 EQU   4                       MODE BIT 1
CWR1RSET EQU   8              OMANUAL: RESET FROM PAPER TAPE
CWR1HOM  EQU   16             OHOM   : HOMOGENEOUS OUTPUT
CWR1PTPE EQU   32             OPTAPE : PAPER TAPE CONTROL
CWR1MBT2 EQU   64                      MODE BIT 2
CWR1HARD EQU   128            OHCOPY : HARDCOPY
CWR1MMSK EQU   68         MODE=        MODE MASK
CWR1COMP EQU   0              COMP   : COMPATIBLE MODE
CWR1LINE EQU   4              LINE   : LINE MODE
CWR1FORM EQU   64             FORM   : FORMAT MODE
CWR1PHYS EQU   68             PHYS   : PHYSICAL MODE
*
*              OUTPUT EDIT OPTION BYTE 2 - NAMES FOR BIT SETTINGS
*
```

```
CWR2HDR  EQU   1               OHDR   : HEADER PRESENT
CWR2NOLC EQU   2               ONOLOGC: NO LOGIC CHARS INTERPRET.
CWR2EXT  EQU   4               EXTEND : EXTENDED LINE OUTPUT
CWR2INFO EQU   8               OINFO  : INFORMATIVE MESSAGE
CWR2POSN EQU   32              ONOPOSN: NO POSITIONING
CWR2TRAN EQU   32              OTRANS : TRANSPARENT MODE
CWR2BEL  EQU   64              OBELL  : ACCOUSTIC ALARM
CWR2ETB  EQU   128             OETB   : ETB INSTEAD OF ETX
*
         END
```

Example 1

```
WROUT1   START

         BALR  10,0
         USING *,10
         WROUT MESSAGE.ERROR
               *,@DCEO      952    900503   53531004
         TERM
               *,VERSION 010
ERROR    TERMD
               *,VERSION 010
MESSAGE  DC    Y(END-MESSAGE)        Record length
         DS    L2                    Reserved
         DC    X'01'                 Print control character (ignored)
         DC    'EXAMPLE WROUT'       Text
END      EQU   *
         END
```

```
/START-PROG $ASSEMBH
%  BLS0500 PROGRAM 'ASSEMBH', VERSION '1.1A00' OF '1992-04-30' LOADED
%  BLS0552 COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1990. ALL RIGHTS
%  RESERVED
%  ASS6010 V 1.1A00 OF BS2000 ASSEMBH-  READY
//COMPILE SOURCE=*L(L=LIB.EX.MANUAL,E=WROUT1),MAC-LIB=$XXXX.LIB.M.SYSLIB.V11.0T10,
 MOD-LIB=LIB.EX.MANUAL(E=WROUT1),LIST=PAR(OUT=*L(L=LIB.EX.MANUAL,E=WROUT1)),
 TEST-SUP=YES
%  ASS6011 ASSEMBLY TIME: 1275 MSEC
%  ASS6018 0 FLAGS, 0 PRIVILEGED FLAGS, 0 MNOTES
%  ASS6019 HIGHEST ERROR-WEIGHT: NO ERRORS
%  ASS6006 LISTING GENERATOR TIME: 801 MSEC
//END
%  ASS6012 END OF ASSEMBH-
/START-PROG *MOD(L=LIB.EX.MANUAL,E=WROUT1)
%  BLS0517 MODULE 'WROUT1' LOADED
BEISPIEL WROUT
/MOD-JOB-OPT LOG=P(L=NO)
```

Example 2a

This example does not use the VTSUCB.

```
WROUTT          START         0
*
                BALR          10,0
                USING         *,10
*
                WROUT         MSG1,ERR,PARMOD=31,MODE=LINE,OBELL=Y
*
                TERM
*
ERR             TERMD
*
MSG1            DC            Y(EMSG1-MSG1)
                DS            3X
                DC            C'  WROUT-output'
EMSG1           EQU           *
*
                END
```

Example 2b

This example uses the VTSUCB.

```
WROUTT          START         0
*
                BALR          10,0
                USING         *,10
*
                WROUT         MSG1,ERR,PARMOD=31,VTSUCBA=VTSUPAR
*
                TERM
*
ERR             TERMD
*
VTSUPAR         VTSUCB        MODE=LINE,BELL=YES
*
MSG1            DC            Y(EMSG1-MSG1)
                DS            3X
                DC            C'  WROUT-output'
EMSG1           EQU           *
*
                END
```

Example 3

Use of format 2

```
WROUT2          START         0
*
                BALR          10,0
                USING         *,10
*
                LA            1,PARAM
                WROUT         (1),PARMOD=31
*
                TERM
*
ERR             TERMD
*
PARAM           WROUT         MSG1,ERR,MF=L,PARMOD=31,MODE=LINE,OBELL=Y
*
MSG1            DC            Y(EMSG1-MSG1)
                DS            3X
                DC            C'   WROUT-OUTPUT'
EMSG1           EQU           *
*
                END
```

3.7 WRTRD (terminal tandem write/read)

WRTRD can only be used with the TIAM access method. It sends a message to the terminal and immediately afterwards reads a message from the terminal. Apart from the message transmitted to the terminal, no other input prompting character appears.

The CUPAB macro generates a DSECT for the operand table of the WRTRD macro for the call in 24-bit addressing mode.

The WRTRD macro accepts the MF operand (see the section on "Format of generated statements" in the "Executive Macros" manual [3]).

For the 31-bit addressing mode the following should be noted:

– At the beginning of the operand list a standard header is generated (see the section on "Format of generated statements" in the "Executive Macros" manual [3]). The user must ensure that this header is correctly initialized. If dynamically generated operand lists are used (access via CSECT/DSECT), no symbolic names or equates are created for the standard header. In such cases therefore the initialization values for the standard header must be taken from an operand list which was generated with MF=L.

– A CSECT/DSECT is generated with MF=C/D or MF=(C,p)/(D,p).
p = prefix (max. 3 characters; any longer prefix is truncated to 3 characters);
default p = CUB. The prefix only changes the field names, not the symbolic names in the equates.

Note on compatibility

The WRTRD macro as of TIAM version V10.0A (in conjunction with BS2000 V10.0A) generates the header version 2 in the standard header. This macro cannot therefore be executed under BS2000 versions prior to V10.0. In versions earlier than V10.0, it branches to the error exit.
If a program is to execute using several BS2000 versions, it must be written for the earliest version (only upward compatibility is guaranteed).

Format 1

Operation	Operands
WRTRD	record1,[edit1],record2,[edit2],[length],error [,PARMOD={24[,MF={L | (E,...)}] | 31[,MF={C | (C,p) | D | (D,p) | L | (E,...)}]}] {,MODE=COMP[,OTRSUP={N[O] | Y[ES]}][,OLINEND={N[O] | Y[ES]}][,OHCOPY={N[O] | Y[ES]}] [,OHDR={N[O] | Y[ES]}][,ITRSUP={N[O] | Y[ES]}][,ILINEND={N[O] | Y[ES]}] [,ILCASE={N[O] | Y[ES]}][,IHDR={N[O] | Y[ES]}][,IGETBS={N[O] | Y[ES]}] [{,MODE=LINE[,OHCOPY={N[O] | Y[ES]}][,OHOM={N[O] | Y[ES]}][,OBELL={N[O] | Y[ES]}] [,ONOLOGC={N[O] | Y[ES]}][,ONOPOSN={N[O] | Y[ES]}][,ILCASE={N[O] | Y[ES]}] [,EXTEND={N[O] | Y[ES]}][,IGETFC={N[O] | Y[ES]}][,IGETBS={N[O] | Y[ES]}] [,ICFD={N[O] | Y[ES]}][,IGETIC={N[O] | Y[ES]}] }]

Operation	Operands
WRTRD (cont.)	[,MODE=FORM [,IGETBS= {N[O]}] [,ILCASE= {N[O]}] {Y[ES]} {Y[ES]} ,MODE=PHYS [,OHDR= {N[O]}] [,OTRANS= {N[O]}] {Y[ES]} {Y[ES]} [[,OETB= {N[O]}] [,ILCASE= {N[O]}]] {Y[ES]} {Y[ES]} [,IHDR= {N[O]}] [,IGETBS= {N[O]}] {Y[ES]} {Y[ES]} [,ITRSUP= {N[O]}] {Y[ES]}] [,RC={OLD}] {NEW} [,VTSUCBA=addr] [,TIMER=value]

The operands RC, VTSUCBA and TIMER are only supported for the 31-bit interface.

record1 Symbolic address of the record to be output. The record begins with the record length field followed by any character and the message to be output.

Format:

Bytes 0-1: length of record + 4 bytes record length field; the length must be greater than 5.

2-3: reserved

4: any character; is neither passed nor interpreted

5-n: record

Example

```
record     DC   Y (recordend record)  reserved byte
           DS   CL2
           DC   X'00'
           DC   C'data record'        record to be tranferred
record end EQU  *
```

edit1

Edit option for the record being transferred to the terminal. Only the first output edit option byte can be set to the meaning described under in the CUPAB macro by direct specification (X'xx'). This operand is not required when standard functions are used (all edit bits = 0), when MODE is specified or when the VTSU control block is used.

record2

Symbolic address of a field to which the record read in by the terminal is to be transferred. The record is read as a record of variable length (the first four bytes contain the record length).

Format:

Byte	0-1:	Length of record + 4 bytes record length field
	2-3:	reserved
	4-n:	record

Example

```
RECORD2    DS   0CL74
LENGTH     DS   CL2
RESERVE    DS   CL2
DATA       DS   CL70
```

edit2

Edit option for the record being transferred to the application program. Only the first input edit option byte can be set to the meaning described under the CUPAB macro by direct specification (X'xx'). This operand is not required when standard functions are used (all edit bits = 0), when MODE is specified or when the VTSU control block is used.

Note

The edit1 and edit2 operands are now supported for reasons of compatibility only. The edit options should be controlled via MODE specifications (see below) or via the VTSU control block (VTSUCBA).

length
: Length of the field specified under record2 (including 4 bytes for the record length field); 5 ≤ length ≤32767.
If this is omitted, the length attribute of the specified field is assumed.

error
: Symbolic address of an error routine in the application program which is branched to in the event of an error.

In the event of an error, register R14 contains the address of the instruction following the WRTRD call. The error code is passed in register R15.
31-bit interface: if error = 0 (address X'00..0') is specified, the program is resumed with the instruction following the WRTRD call.

PARMOD
: controls the macro expansion. Either the 24-bit interface or the 31-bit interface is generated. If PARMOD is not specified, the macro expansion is performed in accordance with the specification for the GPARMOD macro (see the "Executive Macros" manual [3]) or the default for ASSEMBLER (= 24-bit addressing).

=24
: The 24-bit interface is generated. Data lists and instructions use 24-bit addresses (address space ≤ 16 Mbytes).

=31
: The 31-bit interface is generated. Data lists and instructions use 31-bit addresses (address space ≤ 2 Gbytes).
Data lists begin with the standard header.

MF

=L
: Generates a parameter list; fields are filled in accordance with parameter specifications.

=(E,...)
: Generates the instruction list.

=C
: Generates a CSECT.

=(C,p)
: Generates a CSECT; a prefix p with a maximum length of 3 bytes is assigned; default: p = CUB.

=D
: Generates a DSECT.

=(D,p)
: Generates a DSECT; a prefix p with a maximum length of 3 bytes is assigned; default: p = CUB.

Note

The MODE specifications, together with the edit options, are only supported for reasons of compatibility. They are now incorporated in the VTSU control block (VTSUCB). Any future additions will be made only to VTSUCB.

MODE

=COMP
: Compatible operating mode. Control characters are permitted in the output messages but are not checked by the system. This mode is compatible with earlier operating system versions.
On 8160, 8162, 9749, 975x, 9763, 3270 and X.29 terminals this mode is treated like MODE=LINE. Edit options OLINEND and ILINEND are ignored. Edit options OTRSUP, OHDR, ITRSUP and IHDR are rejected (RS: X'08').

=LINE
: The current terminal is to be treated as a virtual line or page terminal. The message can be structured with the aid of logical control characters (see VTCSET macro).
No other control characters are permitted for output. Any that occur are converted into a substitute character defined by the user (see the MODIFY-TERMINAL-OPTIONS SUBSTITUTE-CHARACTER= command).
If SYSOUT is not a terminal, only the logical control characters NL and LP are interpreted, e.g. for printer output in batch mode.
The device-specific message header is not supplied with input.

=FORM
: Format mode. The application program is supported by the software component FHS/format handling, which also edits the message in a form suitable for output to a particular terminal.

=PHYS
: Physical message output. The message is to be output to or input from the terminal without being edited. This allows special device functions to be executed for which the LINE or FORM mode is insufficient. If none of the valid edit options are specified, the system prefixes the output message with a standard device message header; the device-specific message header is not removed from the input message. Lowercase letters are converted to upper case and backspacing is performed if necessary.

OTRSUP

=YES
: Translation of the message is suppressed. In this case the program must supply the message in device code.

=NO
: Message translation from EBCDIC to device code is not suppressed, i.e. the program supplies the message in EBCDIC and the system converts it to device code.

OLINEND

=YES
: The message is output to the terminal without CR/LF control character supplied by the system. The line feed must be controlled by the application program.

=NO — Each message output to a terminal starts on a new line. The control characters required are either prefixed to the message by the system, or, if the terminal type demands it, inserted in the message when the physical end of line is reached.

OHCOPY

=YES — The message output to a data display terminal is simultaneously output to a hardcopy unit (printer) attached to the terminal. The hardcopy unit (printer) must be generated or assigned by means of MODIFY-TERMINAL-OPTIONS.

3270 terminals:
The contents of the entire screen are printed out on the hardcopy unit. As a result, earlier inputs and outputs may also be printed out. In the event of several consecutive outputs, the hardcopy function is only initiated for the last output. A hard copy is produced only if a hardcopy unit was generated for the data display terminal at connection setup.

If EXTEND=YES/MODE=EXTEND was specified, there is no hardcopy output.

If OHCOPY=YES is specified, and the message contains the logical control character SPA, EPA, CHS or NUM, only the last unprotected portion of the message is printed, not the entire message.

If OVERFLOW-CONTROL=NO (MODIFY-TERMINAL-OPTIONS command) is also specified, it may be that only part of the output is printed out on the hardcopy unit.

=NO — No hard copy is produced.

OHDR

=YES — The message includes a user-specific header (specified in ASCII code) which the system prefixes to the output text. The length of the message header +1 must be specified in binary in the first byte of the message.

3270 terminals:
The message header is specified in EBCDIC and comprises the CMD byte and the WCC byte. It must be preceded by a byte containing X'01' (length of the TRANSDATA message header + 1).

Note

When output is sent to 8160, 975x, 9763 Data Display Terminals and local printers attached to them, the system (MODE=LINE or COMP) or FHS (MODE=FORM) does not use a message header

(PARAM0, PARAM1), but instead relies on parameter specifications (PAG). The differences between the two modes are discussed in the manuals on the data display terminals and printers.

=NO — The message header (in ASCII code) is prefixed to the output text by the system.

OHOM

=YES — Only valid for 816x, 9749, 975x, 9763 and 3270 terminals.

The message is to be output in unstructured form, i.e. the entire message is treated as a single, homogeneous output unit. The message length is limited by the size of the system's output buffer.

Effect on 816x, 975x, 7963 and 3270 terminals in mode 1:
By modifying one character, the entire message can be transmitted back to the computer (provided it has not been explicitly structured with logical display control characters).

=NO — The message is to be output as a structured message, i.e. in non-homogeneous form so that one logical line is regarded as the output unit. There is no limit on the message length as long as the logical lines do not exceed 255 characters.

Effect on 816x, 975x, 9763 and 3270 terminals in mode 1:
Individual logical lines can be modified separately and thus transmitted back selectively.

ONOPOSN

=YES — The output message begins at the start of the current line (valid for printer terminals only).

=NO — The output message begins at the start of the next line.

OBELL

=YES — An audible signal sounds at the end of the output message (only on 9749, 975x, 9763, 816x and 3270 terminals with special hardware option).

=NO — No bell sounds during output.

ONOLOGC

=YES — Logical control characters are not interpreted. All characters less than X'40' in the EBCDI code are replaced by SUB. Only characters greater than or equal to X'40' are accepted.

=NO — All logical control characters are interpreted and special physical control characters are accepted (see VTCSET, e.g. ESC, DC4). Other characters less than X'40' are replaced by SUB. Only characters greater than or equal to X'40' are accepted.

EXTEND

=YES — (For 9749, 975x, 9763, 816x and 3270 display terminals only).

This option supports the use of protected and unprotected fields by way of the logical control characters EPA, DAR, NUM and SPA (see VTCSET macro).

Output of text is protected and displayed at low intensity as the default option. The message can be structured by logical control characters (see VTCSET). With 3270 terminals it must be noted that logical control characters take up space on the screen. However, a number of consecutive logical control characters require only one character position. Areas in which the user may enter data begin with EPA, DAR or NUM and end with SPA.

In input/output NUL is treated as a valid character; it is sent from the program to the terminal and vice versa. With 3270 terminals it must be noted that null characters are not transmitted to the computer. VTSU-B pads fields which are returned in shortened form by adding null characters until the original length is reached. The user thus always receives output-length fields.

The beginning of an output message is displayed at the start of the line following the cursor position. Before the first text character is displayed, the screen is cleared from the cursor position if the message does not begin with VPA.
If the end of the screen is reached during output, output continues from the start of the screen. This continuation is unprotected in every case. Screen overflow control is disabled.
The keys RU, EFZ, AFZ and LSP are locked.

All other edit options except OBELL, ILCASE and IGETFC are ignored.

If the control character NL is detected in an input message, processing continues and return code X'2C' is supplied (RS: X'2C').

=NO — Operator prompting is performed by the system. Only the input request by the system or the application program is protected against overwriting.

Output is unprotected and displayed at low intensity.

Null characters in the output text are converted to substitute characters and removed on input.

Depending on the mode, the screen may be cleared from the cursor position prior to output.

OETB

=YES — The message output to the terminal is terminated with the control character ETB.

=NO — The message output to the terminal is terminated with the control character ETX.

OTRANS

=YES — The output data is to be transmitted "transparently", i.e. it may consist of any binary characters (5, 7 or 8 bits per character depending on the device code) and is not converted on the transmission path. If the transmission path has not been generated as a "potentially transparent" path, the output is rejected with the return code X'04'.

=NO — The output data is to be transmitted in "standardized" form, i.e. code conversion does take place.

ITRSUP

=YES — Translation from device code to EBCDIC is suppressed, i.e. the application program receives the message in the device code.

=NO — Translation from device code to EBCDIC is not suppressed, i.e. the application program receives the message in EBCDIC.

Exception

On 816x, 9749, 975x and 9763 Data Display Terminals the message header is always supplied in device code.

ILINEND

=YES — The CR/LF characters are passed to the application program.

=NO — The CR/LF characters are not passed to the application program.

ILCASE

=YES — Lowercase letters are passed to the application program.

=NO — Lowercase letters are passed to the application program as upper case.

IHDR

=YES — The entire message header is passed to the application program. On 3270 terminals the message header comprises the application ID (AID byte) and the two-byte cursor position.

=NO — The message header is not passed to the application program.

IGETBS

=YES — "Underline" (X'6D') characters are passed to the application program without being interpreted by the system (printer terminals only).

=NO — Underlines are not passed to the application program. instead, the correction function is performed by the system (printer terminals only).

IGETFC

=YES — The first byte of the record in the read area is to contain the standardized function key code. This identifies the key used to initiate data transmission from the terminal. A table of the standardized function key codes is given in the appendix on page 175. This table also includes the function key codes for 3270 terminals.

=NO — The function key code is not to be passed.

IGETIC

=YES — The input data is to be read from the ID card reader. This data may consist of the badge information or the short code K14. This entry is permitted only for 9749, 975x, 9763, 816x and 3270 terminals with a defined ID card reader (see also macro TSTAT TYPE=TCHAR). Unlike TRANSDATA devices, 3270 terminals allow data input from an ID card reader at any time. If inputs are requested from an ID card reader, every other entry is converted into K14.

Note

The IGETIC operand is ignored if ICFD=YES is also specified, or if no ID card reader is connected. The source of input remains unchanged.

=NO — The source of input is not to be changed.

ICFD

=YES — The input data is confidential and must not be visible at the terminal. Depending on the terminal type, this is achieved by blanking, by clearing the screen (which resets the screen format to 24x80) or by overwriting the input line (on printer terminals).

=NO — No measures are taken to protect confidential data.

RC — This operand is only permitted for the 31-bit interface.

=NEW — The return code is stored in register 15 and in the standard header (with full 4-byte return code). A 4-byte return code is returned only if SYSDTA reads from the data display terminal. In all other cases only a 1-byte return code is returned, regardless of the value of the return code.

=OLD — The return code is stored in register 15. The three leftmost bytes are masked out.

VTSUCBA

=addr — Address of a VTSUCB generated with MF=L. This operand is only permitted for the 31-bit interface. If the VTSUCBA operand is used, the MODE operand and subsequent EDIT options are ignored. Their value is set to X'FF' in the parameter list. All the required EDIT options must be specified in VTSUCB.
Default: VTSUCB is not used.

TIMER — This operand is permitted only for the 31-bit interface.

=value — This field allows the user to specify whether a TIMER is to be used or not (UNLIMITED). The timer defines a maximum waiting time for data entry. If no entry is registered within this time, a return code is issued. The timer value is set in seconds.
The default is UNLIMITED, which means that no timer is used.

Programming notes on the use of the operand MODE=EXTEND or MODE=LINE with EXTEND=YES will be found under the description of the WROUT macro (see page 96).

Format 2

Operation	Operands
WRTRD	(1)

(1) Register R1 contains the address of the operand list. The list must be aligned on a word boundary.

Functional description

Format 1: When the macro is executed, the specified operands are stored in an operand table and the start address of this table is loaded in register 1.

Format 2: The table specified in the application program is used.

Format of operand list

Addressing mode	Byte	Contents
24-bit mode	0	Output edit byte 1, in accordance with edit option table for output.
	1 - 3	Address of the record to be output (' record1' operand).
	4	Input edit byte 2, in accordance with edit option for input.
	5 - 7	Address of the field to which the record to be read in is transferred (' record2' operand).
	8	Output edit byte 2, in accordance with edit option table for output.
	9	Input edit byte 2, in accordance with edit option table for input.
	10 - 11	Max. length of record to be read in (' length' operand).
	12 - 15	Address branched to in the event of an error (' error' operand).
31-bit mode	0 - 7	Standard header (for format see the section on "Format of generated statements" in the "Executive Macros" manual [3]). The initialization values can be obtained from an operand generated with MF=L. If RC=OLD, no return code is passed in the standard header.
	8 - 11	Address branched to in the event of an error (' error' operand).
	12 - 15	Address of the record to be output (' record1' operand).
	16 - 19	Address of the field to which the record to be read in is transferred (' record2' operand).
	20	Output edit byte 1 (in accordance with the equates in the macro created with MF=D)
	21	Output edit byte 2
	22	Input edit byte 1
	23	Input edit byte 2
	24 - 25	Max. length of the record to be read (' length' operand).
	26	Reserved (X' 00')
	27	Flag indicating use of VTSUCB and for return code behavior; equates in macro generated with MF=D.
	28 - 31	VTSUCB address
	32 - 33	Values of the timer
	34 - 35	Reserved (X' 0000')

The values for input/output edit bytes 1/2 can be found in the table given under the CUPAB macro (24-bit mode) or in a data list generated with MF=C/D (31-bit mode).

Note

– If the length of the record to be written (less 4 bytes for the length field and 1 reserved byte) exceeds the size of the terminal buffer, the record is truncated. The program branches to the user's error address with return code X'10' in register 15.

– If the length of the record to be read exceeds the specified length (less 4 bytes for the length field), the record is truncated. The program branches to the user's error address with return code X'0C' in register 15.

Return information (acknowledgments) and error flags

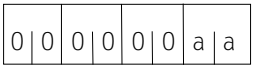
– Any edit options illegal for a particular device or the selected mode are corrected by the system as far as possible (return code X'20').

– During macro processing register R1 contains the address of the operand list.

– A return code in passed in register R15 concerning execution of the macro WRTRD.

a) 24-bit mode:

Register R15:



If the macro is correctly executed, register R15 remains unchanged.

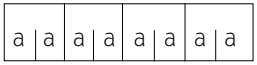
X' aa'	Explanation
X' 04'	Unrecoverable error.
X' 08'	Operand error (invalid operand in program).
X' 0C'	Record truncated during reading. The length of the record exceeds the specified length. A header is prefixed to the input message by an MSV terminal. If the message has already reached the length of the system buffer, the prefixed header will cause the message text to be truncated.
X' 10'	Output record truncated. The length of the record to be written exceeds the size of the terminal buffer. The excess characters are not output.
X' 14'	WRTRD was called in a batch job.
X' 18'	End of input (ETX) detected.
X' 20'	Illegal edit option byte was corrected by the system.
X' 2C'	Only with EDIT option EXTEND=YES. Input begins with control character NL. On 3270: input length reduced.
X' 38'	Error in conjunction with POSIX.

b) 31-bit mode:

When using the 31-bit interface, the return codes can be stored either in register 15 (RC=OLD) or in both register 15 and the standard header (RC=NEW).

RC=OLD

Register R15:



If the macro is correctly executed, register R15 is cleared.

In addition to the return codes described under a) (right-justified, left 0), the return code X'24' (error in VTSUCB) and the global system values may also occur.

RC=NEW

Error messages are stored both in register 15 and in the standard header, see following table (for the equates for MAINCODE and SUBCODE, see macro expansion).

SUBCODE		**MAINCODE**		**Explanation**
2	1	2	1	
X' 00'	X' 00'	X' 00'		Successful processing
			X' 00'	– no additional information
			X' 20'	– operand error, corrected by TIAM/ VTSU
	X' 01'	X' 00'		Parameter error
X' 00'			X' 08'	– operand error, no further specification
X' 07'			X' 08'	– RESERVED fields not 0
X' 08'			X' 08'	– timer setting out of range not a value from 10 to 3600 seconds
	X' 20'	X' 00'		Internal error
X' 00'			X' 04'	– no further specification
X' 02'			X' 04'	– BCAM message lost
X' 05'			X' 04'	– input message too long
X' 06'			X' 04'	– negative transport acknowledgment

SUBCODE		**MAINCODE**		**Explanation**
X' 00'	X' 40'	X' 00'		No class reaction
			X' 0C'	– input record truncated
			X' 10'	– output record truncated
			X' 14'	– BREAK in WRTRD *)
			X' 18'	– end of input
			X' 2C'	– only with edit options where EXTEND=YES: NL detected
			X' 34'	– timer timed out (no input within waiting time defined by timer setting)
X' 00'	X' 00'	X' 00'		Error in SYSFILE
			X' 14'	– WRTRD in batch mode
	X’ 80’	X’ 00’		Internal bottleneck
X’ 01’			X’ 04’	– BCAM bottleneck
				error in conjunction with POSIX
X’ 09’	X’ 80’	X’ 00’	X’ 38’	– I/O serialization error
X’ 0A’	X’ 40’	X’ 00’	X’ 38’	– If the LOGON task is in system mode, processes created by fork() cannot effect I/Os
from VTSUCB	from VTSUCB	from VTSUCB	X’ 24’	VTSU has detected an error (for MAINCODE 1 see error information in the VTSUCB header)

*) If a BREAK is detected in a write/read operation and if RC=NEW, the return code received by the user is X’00400014’. If RC=OLD, the P1 command counter is reset to the start of macro expansion, which means that the macro is repeated once the BREAK has been handled.

Macro expansion

```
WRTRDP   WRTR  MF=C,PARMOD=31
*
*              WRTRD PARAMETER BLOCK
*
CUBHEAD  FHDR  MF=(C,*NO-NAMES)
CUBHEAD  DS    A               0   INTERFACE IDENTIFIER
         DS    A               4   GENERAL RETURN CODE
*
*        MAIN RETURN CODES AS IN FHDR, ADDITIONAL VALUES:
CUBURE   EQU  4            UNRECOVERABLE ERROR
CUBOPE   EQU  8            OPERAND ERROR
CUBCUTE  EQU 12            TRUNCATION DURING READ
CUBTRUN  EQU 16            TRUNCATION DURING WRITE
CUBATCH  EQU 20            WRTRD IN BATCH
CUBEOF   EQU 24            WRTRD EOF
CUBOPEC  EQU 32            OPERAND ERROR, CORRECTED
CUBVTE   EQU 36            VTSU ERROR
CUBNL    EQU 44            NL EXT INPUT
CUBTOUT  EQU 52            TIME OUT DURING READ
CUBPOSE  EQU 56            ERROR IN POSIX CONTEXT

*        SUB1 RETURN CODES AS IN FHDR
*        SUB2 RETURN CODES:
CUBRCNS  EQU  0            RC_S2_NOT_SPECIFIED
CUBBCSH  EQU  1            BCAM SHORTAGE
CUBBCLL  EQU  2            BCAM LETTER LOST
CUBMTL   EQU  5            MESSAGE TOO LONG
CUBNACK  EQU  6            NEGATIVE TACK
CUBRFNZ  EQU  7            RESERVED FIELDS NOT ZERO
CUBWT    EQU  8            WRONG TIMER
CUBLOCE  EQU  9            SERIALIZATION LOCK ERROR
CUBFTIO  EQU  10           NO FORKED TASK IS POSSIBLE
*
CUBERRW  DC    A(0)             ERROR RETURN ADDRESS
CUBMSGW  DC    A(0)             OUTPUT MESSAGE ADDRESS
CUBAREAW DC    A(0)             READ AREA ADDRESS
CUBOEDT1 DC    AL1(0)           OUTPUT EDIT OPTION 1
CUBOEDT2 DC    AL1(0)           OUTPUT EDIT OPTION 2
CUBIEDT1 DC    AL1(0)           INPUT EDIT OPTION 1
CUBIEDT2 DC    AL1(0)           INPUT EDIT OPTION 2
CUBALEN  DC    H'0'             READ AREA LENGTH
CUBRES01 DC    AL1(0)           RESERVED
CUBFB1   DC    AL1(0)           FLAG BYTE 1
CUBVTSI  EQU   X'80'            - VTSU INDICATOR
CUBRCIND EQU   X'40'            - RC INDICATOR
CUBTIND  EQU   X'20'            - INPUT TIMER INDICATOR
CUBVTSU  DC    A(0)             VTSUCB ADDRESS
CUBTVAL  DC    H'0'             INPUT TIMER VALUE
CUBRTIA  DC    H'0'             RES_FOR_TIAM
*
CUBL@WRT EQU   *-CUBHEAD        WRTRD PARAMETER BLOCK LENGTH
*
*              OUTPUT EDIT OPTION BYTE 1 - NAMES FOR BIT SETTINGS
*
CWR1CODE EQU   1           OTRSUP : CODE TRANSLATION
CWR1LNET EQU   2           OLINEND: LINE END TREATMENT
CWR1MBT1 EQU   4                    MODE BIT 1
CWR1RSET EQU   8         OMANUAL: RESET FROM PAPER TAPE
```

```
CWR1HOM  EQU   16        OHOM   : HOMOGENEOUS OUTPUT
CWR1PTPE EQU   32        OPTAPE : PAPER TAPE CONTROL
CWR1MBT2 EQU   64                 MODE BIT 2
CWR1HARD EQU   128                OHCOPY : HARDCOPY
CWR1MMSK EQU   68   MODE=         MODE MASK
CWR1COMP EQU   0         COMP   : COMPATIBLE MODE
CWR1LINE EQU   4         LINE   : LINE MODE
CWR1FORM EQU   64        FORM   : FORMAT MODE
CWR1PHYS EQU   68        PHYS   : PHYSICAL MODE
*
*              OUTPUT EDIT OPTION BYTE 2 - NAMES FOR BIT SETTINGS
*
CWR2HDR  EQU   1         OHDR   : HEADER PRESENT
CWR2NOLC EQU   2         ONOLOGC: NO LOGIC CHARS INTERPRET.
CWR2EXT  EQU   4         EXTEND : EXTENDED LINE OUTPUT
CWR2INFO EQU   8         OINFO  : INFORMATIVE MESSAGE
CWR2POSN EQU   32        ONOPOSN: NO POSITIONING
CWR2TRAN EQU   32        OTRANS : TRANSPARENT MODE
CWR2BEL  EQU   64        OBELL  : ACCOUSTIC ALARM
CWR2ETB  EQU   128       OETB   : ETB INSTEAD OF ETX
*
*              INPUT EDIT OPTION BYTE 1 - NAMES FOR BIT SETTINGS
*
CRD1CODE EQU   1         ITRSUP : CODE TRANSLATION
CRD1LNET EQU   2         ILINEND: LINE END TREATMENT
CRD1BACK EQU   4         IGETBS : BACKSPACE
CRD1RSET EQU   8         IMANUAL: RESET TO MANUAL CONTROL
CRD1LCT  EQU   16        ILCASE : LOWER CASE TRANSLATION
CRD1MBT1 EQU   32                MODE BIT 1
CRD1MBT2 EQU   64                MODE BIT 2
CRD1HDR  EQU   128       IHDR   : HEADER REQUIRED
CRD1MMSK EQU   96   MODE=         MODE MASK
CRD1COMP EQU   0         COMP   : COMPATIBLE MODE
CRD1LINE EQU   32        LINE   : LINE MODE
CRD1FORM EQU   64        FORM   : FORMAT MODE
CRD1PHYS EQU   96        PHYS   : PHYSICAL MODE
*
*              INPUT EDIT OPTION BYTE 2 - NAMES FOR BIT SETTINGS
*
CRD2GFC  EQU   1         IGETFC : GET FUNCTION CODE
CRD2CFD  EQU   4         ICFD   : CONFIDENTIAL INPUT DATA
CRD2GIC  EQU   8         IGETIC : GET IDENTITY CARD
CRD2EXT  EQU   32        EXTEND : EXTENDED LINE INPUT
*
         END
```

Format 2

Operation	Operands
WRTRD	NA1,,NA2,,60,FEHL,MODE=LINE

NA1 Address of the message to be transmitted to the terminal.

NA2 Address of buffer for input message which is to be transmitted from the terminal to the application program.

60 Length of the area for NA2 in the application program.

FEHL Address in the application program which is to be branched to in the event of an error.

Example 1

```
WRTRD    START
         BALR  10,0
         USING *,10
         PRINT NOGEN
FRAGE    WRTRD ABFRAG,,EINB,,5,ENDE
               *,@DCEO      999    921011   53531004
               *,@DCEI      999    921011   53531002
         CLI   ANTW,'N'
         BE    FRAGE
ENDE     TERM
               *,VERSION 100
ABFRAG   DC    Y(ABEND-ABFRAG)
         DS    CL2
         DC    X'01'
         DC    'PROGRAMM BEENDEN?(J/N)'                                    (1)
ABEND    EQU   *
EINB     DS    0CL5
         DS    CL4
ANTW     DS    CL1
         END
```

```
/START-PROG $ASSEMBH
%  BLS0500 PROGRAM 'ASSEMBH', VERSION '1.1A00' OF '1992-04-30' LOADED
%  BLS0552 COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1990. ALL RIGHTS
%  RESERVED
%  ASS6010 V 1.1A00 OF BS2000 ASSEMBH-  READY
//COMPILE SOURCE=*L(L=LIB.EX.MANUAL,E=WRTRD),MAC-LIB=$XXXX.LIB.M.SYSLIB.V11.0T10,
 MOD-LIB=LIB.EX.MANUAL(E=WRTRD),LIST=PAR(OUT=*L(L=LIB.EX.MANUAL,E=WRTRD)),TEST-SUP=YES
%  ASS6011 ASSEMBLY TIME: 1510 MSEC
%  ASS6018 0 FLAGS, 0 PRIVILEGED FLAGS, 0 MNOTES
%  ASS6019 HIGHEST ERROR-WEIGHT: NO ERRORS
%  ASS6006 LISTING GENERATOR TIME: 745 MSEC
//END
%  ASS6012 END OF ASSEMBH-
/START-PROG *M(L=LIB.EX.MANUAL,E=WRTRD)
%  BLS0517 MODULE 'WRTRD' LOADED
PROGRAMM BEENDEN? (J/N)                                                    (1)
N
PROGRAMM BEENDEN? (J/N)                                                    (1)
J
/MOD-JOB-OPT LOG=P(L=N)
```

Key

(1) Terminate program? (Y/N)

Example 2

Use of format 2

```
WRTRD2   START
         BALR  10,0
         USING *,10
*
LOOP     LA    1,PARAM
         WRTRD (1),PARMOD=31
*
         CLI   CHAR,'N'
         BE    LOOP
END      TERM
*
DS       OF
PARAM    WRTRD QUEST,,INPUT,,5,MF=L,PARMOD=31
*
QUEST    DC    Y(EQUEST-QUEST)
         DS    3X
         DC    'PROGRAMM BEENDEN?(J/N)'                                    (1)
EQUEST   EQU   *
*
INPUT    DS    OCL5
LENGTH   DS    CL2
UNUSED   DS    CL2
CHAR     DS    CL1
         END
```

Key

(1) Terminate program? (Y/N)

Example 3

WRTRD with extended line mode

```
WRTRD3   START
ANF      BALR  10,0
         USING *,10
         PRINT NOGEN
FRAGE    WRTRD AUSGABE,,EINB,,69,ENDE,MODE=LINE,EXTEND=YES
               *,@DCEO      999    921011   53531004
               *,@DCEI      999    921011   53531002
         CLC   EINNAM(4),=C'ENDE'
         BNE   FRAGE
ENDE     TERM
               *,VERSION 100
VTCSET   LOG
EINB     DS    OCL69
         DS    CL4
EINNAM   DS    CL12
EINVOR   DS    CL12
EINSTR   DS    CL25
EINPLZ   DS    CL4
EINORT   DS    CL12
*
*
```

```
AUSGABE  DS    0H
         DC    Y(ABEND-AUSGABE)
         DS    CL2
         DC    X'01'
F1       DC    AL1(LOGNP)                                        NP
F2       DC    AL1(LOGSPA)                                       SPA
F3       DC    AL1(LOGEM3)                                       EM3
         DC    C'BITTE GEBEN SIE NAMEN UND ANSCHRIFT EIN !'
F4       DC    AL1(LOGNL)                                        2xNL
         DC    AL1(LOGNL)
F5       DC    AL1(LOGNOR)                                       NOR
         DC    C'NAME: '
F6       DC    AL1(LOGEPA)                                       EPA
NAME     DS    CL12
F7       DC    AL1(LOGSPA)                                       SPA
         DC    C'      VORNAME: '
F8       DC    AL1(LOGEPA)                                       EPA
VORNAME  DS    CL12
F9       DC    AL1(LOGNL)                                        NL
F10      DC    AL1(LOGSPA)                                       SPA
         DC    C'STRASSE: '
F11      DC    AL1(LOGEPA)                                       EPA
STRASSE  DS    CL25
F12      DC    AL1(LOGNL)                                        NL
F13      DC    AL1(LOGSPA)                                       SPA
         DC    C'WOHNORT: '
F14      DC    AL1(LOGEPA)                                       EPA
F15      DC    AL1(LOGNUM)                                       NUM
PLZ      DS    CL4
F16      DC    AL1(LOGSPA)                                       SPA
         DC    C'  '
F17      DC    AL1(LOGEPA)                                       EPA
ORT      DS    CL12
F18      DC    AL1(LOGNL)                                        NL
F19      DC    AL1(LOGNL)                                        NL
F20      DC    AL1(LOGSPA)                                       SPA
         DC    C'BEI PROGRAMMENDE BITTE DEN NAMEN "ENDE" EINGEBEN !'
ABEND    EQU   *
         END   ANF
```

```
/START-PROG $ASSEMBH
%  BLS0500 PROGRAM 'ASSEMBH', VERSION '1.1A00' OF '1992-04-30' LOADED
%  BLS0552 COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1990. ALL RIGHTS RESERVED
%  ASS6010 V 1.1A00 OF BS2000 ASSEMBH-  READY
//COMPILE SOURCE=*L(L=LIB.EX.MANUAL,E=WRTRD3),MAC-LIB=$XXXX.LIB.M.SYSLIB.V11.0T10,        -
 MOD-LIB=LIB.EX.MANUAL(E=WRTRD3),
LIST=PAR(OUT=*L(L=LIB.EX.MANUAL,E=WRTRD3)),TEST-SUP=YES
%  ASS6011 ASSEMBLY TIME: 1576 MSEC
%  ASS6018 0 FLAGS, 0 PRIVILEGED FLAGS, 0 MNOTES
%  ASS6019 HIGHEST ERROR-WEIGHT: NO ERRORS
%  ASS6006 LISTING GENERATOR TIME: 833 MSEC
//END
%  ASS6012 END OF ASSEMBH-
/START-PROG *M(L=LIB.EX.MANUAL,E=WRTRD3)
%  BLS0517 MODULE 'WRTRD3' LOADED
```

```
BITTE GEBEN SIE NAMEN UND ANSCHRIFT EIN !

NAME: ............      VORNAME: ............
STRASSE: ........................
WOHNORT: ....  ............

BEI PROGRAMMENDE BITTE DEN NAMEN "ENDE" EINGEBEN !
```

```
/MOD-JOB-OPT LOG=P(L=NO)
```

Key

```
FRAGE = Question
AUSGABE = Output
BITTE GEBEN SIE NAMEN UND ANSCHRIFT EIN = Please enter name and address
VORNAME = First name
STRASSE = Street
WOHNORT = City
ENDE = End
BEI PROGRAMMENDE BITTE DEN NAMEN "ENDE" EINGEBEN = Please enter name "ENDE"
at end of program
```

Example 4a

This example does not use the VTSUCB.

```
WRTRDT   START 0
*
         BALR  10,0
         USING *,10
*
         WRTRD MELD1,,INPUT1,,40,FEHL,PARMOD=31,MODE=LINE,       +
               OBELL=Y,ILCASE=Y,ICFD=Y
*
FEHL     NOP   0
         TERM
*
MELD1    DC    Y(EMELD1-MELD1)
         DS    3X
         DC    C'  WRTRD-Ausgabe'
EMELD1   EQU   *
*
*
INPUT1   DS    0CL40
LAENGE   DS    CL2
UNUSED   DS    CL2
DATEN    DS    CL36
*
         END
```

Example 4b

This example uses the VTSUCB.

```
WRTRDT   START  0
*
         BALR   10,0
         USING  *,10
*
         WRTRD  MELD1,,INPUT1,,40,FEHL,PARMOD=31,VTSUCBA=VTSUPAR
*
FEHL     NOP    0
         TERM
*
VTSUPAR  VTSUCB MODE=LINE,BELL=YES,LOW=YES,SPECIN=C
*
MELD1    DC     Y(EMELD1-MELD1)
         DS     3X
         DC     C'  WRTRD-Ausgabe'
EMELD1   EQU    *
*
INPUT1   DS     0CL40
LAENGE   DS     CL2
UNUSED   DS     CL2
DATEN    DS     CL36
*
         END
```

4 TIAM COBOL interface

The TIAM COBOL interface offers the following functions to the COBOL programmer:

RDATA Read record from SYSDTA

WROUT Write record to SYSOUT

WRTRD Terminal tandem write/read

The functions are implemented as subroutines to which the program branches when they are called (CALL ...).
Before issuing I/O calls, the user merely enters the appropriate parameters in certain data structures from which TIAM can then extract information about the desired input/output and where it returns an error code.

Format application with FHS is available as a special function integrated into CALL WROUT and CALL WRTRD (see also FHS manuals [6] and [7]).

Notes

– Before using the TIAM-COBOL interface, make sure that the TIAM interface module DCCOBRTS is explicitly linked to the COBOL program by means of an INCLUDE statement.

– Before using the integrated FHS mode (mode='F'), make sure that the FHS module MFHSCALL is explicitly linked to the COBOL program by means of an INCLUDE statement. If the FHS module is not found, the WROUT or WRTRD call will be rejected an a return code issued.

Brief description of the COBOL calls

COBOL calls	Brief description
RDATA	Reads a record from the system file SYSDTA
WROUT	Sends a message to the system file SYSOUT
WRTRD	Sends a message to the data display terminal in interactive mode and subsequently reads a message.

4.1 Data structures for TIAM COBOL calls

TIAM and the application program can communicate by way of data structures which are contained in a library as COPY members and are copied into the application program. The following data structures are available:

VTSU-CONTROL-BLOCK

This data structure enables special VTSU parameters for input/output to be set. The specifications in the VTSUCB replace the EDIT options in the TIAM-CONTROL-INFO data structure. Parameters which differ from the default values must be set in this VTSUCB using MOVE.

The data structure VTSU-CONTROL-BLOCK is copied to the application program by means of the call

```
COPY VTSUCBC
```

This COPY member is located in the library $TSOS.SYSLIB.VTSU-B.111. VTSU-CONTROL-BLOCK is described in the VTSU manual (see [1] in the Related publications chapter).

LINE-MODE-CONTROL-CHARACTERS

This data structure contains symbolic names for the control characters required for output in LINE-MODE, EXTENDED-LINE or INFO in the VTSUCB. The control characters are transferred to the output area with a simple MOVE statement.
The data structure LINE-MODE-CONTROL-CHARACTERS is copied into the application program by means of the call

```
COPY TIAMCTRC
```

This COPY member is located in the library $TSOS.SYSLIB.VTSU-B.111.
TIAMCTRC is described in the VTSU manual (see [1] in the Related publications chapter).

4.1.1 Data structure TIAM-CONTROL-INFO

This data structure controls I/Os:

– Application programs pass important data to TIAM in it (e.g. the output mode).

– TIAM evaluates this data structure. Where no entries have been made, TIAM assumes default values where necessary.

– TIAM stores data here (e.g. an error code) about the execution of the TIAM-COBOL call.

The following call is used to copy TIAM-CONTROL-INFO data structure into the application program.

```
COPY TIAMINFO
```

This copy element is held in the $TSOS.SYSLIB.TIAM.112 library.

```
 COPY TIAMINFO
*****************************************************************
*  TIAMINFO         002       920620           TIAM          U *
*****************************************************************
*                                                               *
* COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1991     *
*               ALL RIGHTS RESERVED                             *
*                                                               *
*****************************************************************
*
 01  TIAM-CONTROL-INFO.
     41  EDIT-OUT.
         42  EDIT-MODE               PIC X.
                 88  LINE-MODE                VALUE "L".
                 88  PHYSICAL-MODE            VALUE "P".
                 88  FHS-MODE                 VALUE "F".
         42  FILLER                  PIC X.
         42  EDIT-OPTIONS            PIC X(6).
                 88  NO-OPTIONS               VALUE "NOOPTS".
                 88  HARDCOPY                 VALUE "HCOPY".
                 88  HOMOGENEOUS-OUTPUT       VALUE "HOMOUT".
                 88  SYSTEM-HEADER            VALUE "HEADER".
                 88  ETB-CLOSED               VALUE "ETB".
                 88  INFORMATIVE-MESSAGE      VALUE "INFO".
                 88  TRANS-CODE               VALUE "TRNCD".
                 88  EXTENDED-LINE-OUTPUT     VALUE "EXTEND".
                 88  NO-LOG-CONTROL           VALUE "NLOGC".
                 88  BELL                     VALUE "BELL".
                 88  OVERWRITE                VALUE "OWRITE".
                 88  BELL-HOMOGENEOUS         VALUE "BELHOM".
                 88  BELL-EXTEND              VALUE "BELEXT".
                 88  BELL-INFORMATIVE         VALUE "BELINF".
                 88  BELL-NO-LOG-CONTROL      VALUE "BELNLC".
                 88  HARDCOPY-NO-LOG-CONTROL  VALUE "HCNLC".
                 88  VTSUCB-USED              VALUE "VTCBU".
         42  FILLER                  PIC X(8).
```

```
           41  EDIT-IN.
               42  EDIT-MODE                PIC X.
                       88  LINE-MODE                   VALUE "L".
                       88  PHYSICAL-MODE               VALUE "P".
               42  FILLER                   PIC X.
               42  EDIT-OPTIONS             PIC X(6).
                       88  NO-OPTIONS                  VALUE "NOOPTS".
                       88  NO-CORRECTION               VALUE "NOCORR".
                       88  LOWER-CASE                  VALUE "LCASE".
                       88  DELETE-DEVICE-HEADER        VALUE "NOHDR".
                       88  GET-FUNCTION-CODE           VALUE "GETFC".
                       88  CONFIDENTIAL-DATA           VALUE "CFDATA".
                       88  GET-ID-CARD                 VALUE "GETIC".
                       88  EXTENDED-LINE-INPUT         VALUE "EXTEND".
                       88  LOWER-CASE-EXTEND           VALUE "LOWEXT".
                       88  CONF-DATA-LOW-CASE          VALUE "CFDLOW".
                       88  FCT-CODE-LOW-CASE           VALUE "GFCLOW".
                       88  FCT-CODE-CONF-DATA          VALUE "GFCCFD".
                       88  FCT-CODE-CONF-DATA-LCASE    VALUE "GFCCDL".
                       88  FCT-CODE-EXTEND             VALUE "GFCEXT".
                       88  FCT-CODE-EXTEND-LCASE       VALUE "GFCEXL".
                       88  DELETE-DEVICE-HEADER-LCASE VALUE "NOHDRL".
                       88  VTSUCB-USED                 VALUE "VTCBU".
               42  FILLER                   PIC X(8).
           41  READLENGTH                   PIC 9(5) COMP SYNC.
           41  ISAM-REQUEST                 PIC X.
                       88  NO-KEY                      VALUE "N".
                       88  GET-KEY                     VALUE "K".
                       88  GET-KEY-LENGTH              VALUE "L".
                       88  GET-KEY-POSITION            VALUE "P".
                       88  GET-KEY-POSITION-LENGTH     VALUE "B".
           41  FILLER                       PIC X(3).
           41  ASSIGNMENT-REQUEST           PIC X.
                       88  NO-ASSIGNMENT-CODE          VALUE "N".
                       88  GET-ASSIGNMENT-CODE         VALUE "G".
           41  FILLER                       PIC X(3).
           41  COPYMEM-ID                   PIC 9(4) COMP SYNC.
           41  FILLER                       PIC X(2).
           41  TIAM-RETURN-INFO.
               42  TIAM-RC                  PIC 9(4) COMP SYNC.
               42  FILLER                   PIC X(2).
               42  ASSIGNMENT               PIC 9(4) COMP SYNC.
               42  FILLER                   PIC X(2).
               42  KEY-POSITION             PIC 9(4) COMP SYNC.
               42  KEY-LENGTH               PIC 9(4) COMP SYNC.
               42  FILLER                   PIC X(4).
       *
       *
   *
```

Description of the data items

EDIT-OUT This part of TIAM-CONTROL-INFO controls the output of the message. EDIT-OUT is interpreted for CALL "WROUT" and CALL "WRTRD".

EDIT-MODE This item (1 byte) specifies the output mode. When the VTSUCB is used, the edit options specified here are ignored. All the required edit options must then be specified in the VTSUCB.

The following entries are possible:

"L" LINE-MODE (default): the terminal is treated as a virtual line or page terminal. The message can be structured using logical control characters.
If the user wishes to use the valid line mode control characters, he or she should copy the LINE-MODE-CONTROL-CHARACTERS data structure into the WORKING-STORAGE SECTION and enter the desired control characters in the output area via a MOVE statement.
No further control characters are permitted; they would be converted to a user-defined substitute character by the system.

"P" PHYSICAL-MODE: the message is to be output at the terminal without being edited by the system.

"F" FHS-MODE: the format handler FHS is called internally to edit the message. TIAM then outputs the edited message.
If EDIT MODE = "F", TIAM will not interpret the remaining items in the TIAM-CONTROL-INFO. For the integrated FHS connection see the FHS manuals, [6] and [7].

EDIT-OPTIONS This item (6 bytes) determines the edit options for output.
The following entries are possible:

"NOOPTS" NO-OPTIONS (default): none of the output options described in the following are desired. "NOOPTS" is legal only in line mode and physical mode.

"HCOPY" HARDCOPY: in addition to being displayed at the terminal, a hard copy is also produced. This entry is legal in line mode only.

"HOMOUT" HOMOGENEOUS-OUTPUT: the message is output in unstructured form. This option is legal in line mode only.

"HEADER" SYSTEM-HEADER: a system-generated message header is prefixed to the output message. This option is legal only in physical mode.

"ETB"	ETB-CLOSED: the output message is concluded with ETB. This option is legal in physical mode.
"INFO"	INFORMATIVE MESSAGE: the message can be displayed in a special information line without destroying important data at the terminal. This entry is intended primarily for application programs that send "asynchronous messages" to terminals without knowing the current terminal display. The message is displayed – as protected information in a hardware system status line (e.g. 9749, 9750, 9752 DDTs) or – as protected information in the last screen line – in all other cases: as a normal line mode message. A message that is longer than one screen line is split up and output a line at a time. Here, the system takes into account the waiting time set by the /MODIFY-TERMINAL-OPTIONS OVERFLOW-CONTROL=TIME command. If OINFO=YES is specified, OHCOPY=YES is ignored, i.e. neither the information line nor the screen contents are printed. The hardware system status line is only reset after the next input followed by an output.
"TRNCD"	TRANS-CODE: the output data is to be transmitted "transparently", i.e. it may consist of any binary characters (5, 7 or 8 bits per character depending on the device code) and is not converted on the transmission path. If the transmission path has not been generated as a "potentially transparent" path, the output is rejected with error code 4. This option is legal in physical mode.
"EXTEND"	EXTENDED-LINE-OUTPUT: the message supports the extended application of protected and unprotected fields. This option is legal in conjunction with LINE-MODE.
"NLOGC"	NO-LOG-CONTROL: all EBCDIC characters less than X’40’ are replaced by SUB. Logical control characters are not interpreted. The option is legal in conjunction with LINE-MODE.

"BELL"	BELL: an audible alarm sounds when an output takes place (only on 9749, 975x, 9763 and 816x Data Display Terminals with special hardware option). This option is valid in line mode only.
"OWRITE"	OVERWRITE: the message is output at the start of the current line. This option is legal in line mode for printer terminals only.
"BELHOM"	BELL-HOMOGENEOUS: combination of BELL and HOMOGENEOUS OUTPUT in line mode.
"BELEXT"	BELL-EXTEND: combination of BELL and EXTENDED-LINE-OUTPUT. The option is legal in conjunction with LINE-MODE.
"BELINF"	BELL-INFORMATIVE: combination of BELL and INFORMATIVE-MESSAGE in line mode.
"BELNLC"	BELL-NO-LOG-CONTROL: combination of BELL and NO-LOG-CONTROL. The option is legal in conjunction with LINE-MODE.
"HCNLC"	HARDCOPY-NO-LOG-CONTROL: combination of HCOPY and NO-LOG-CONTROL. This option is legal in conjunction with LINE-MODE.
"VTCBU"	VTSUCB-USED: a VTSU control block is used. The VTSUCB area must be defined and specified in the function call. If the VTSUCB is not to be used, the VTSUCB area may not be specified in the function call.

EDIT-IN This part of TIAM-CONTROL-INFO controls the input (reading) of messages. EDIT-IN is interpreted for CALL "RDATA" and CALL "WRTRD".

EDIT-MODE This item specifies the input mode (1 byte).

When the VTSUCB is in use, the edit options specified here are ignored. All required edit options must then be specified in the VTSUCB.

The following entries are possible:

"L"	LINE-MODE (default): the terminal is treated like a virtual line terminal. "L" is the only entry permitted in CALL "RDATA".
"P"	PHYSICAL MODE, only permitted in CALL" WRTRD": the message is to be passed to the application program without being edited by the system.

EDIT-OPTIONS Input edit options (6 bytes).
The following entries are possible:

"NOOPTS" NO-OPTIONS (default value): no special handling of the message, i.e. none of the following input edit options are desired.
"NOOPTS" is legal in line and physical mode.

"NOCORR" NO-CORRECTION: correction characters are treated as text characters (printer terminals only).

"LCASE" LOWER-CASE: lowercase letters are passed to the application program. This option is legal in line mode and physical mode.

"NOHDR" DELETE-DEVICE-HEADER: the device-dependent message header is stripped from the input message.
Legal only in physical mode.

"GETFC" GET-FUNCTION-CODE: the first byte of the input message contains the function key code.
Legal only in line mode.

"CFDATA" CONFIDENTIAL-DATA: the input data is confidential and must not be visible on the terminal. Depending on the terminal type, this is accomplished either by blanking, clearing the screen (which causes the screen format to be reset to 24x80) or overwriting the input line (on printer terminals).
This option is valid in line mode only.

"GETIC" GET-ID-CARD: the input is to be read from the attached ID card reader. It may consist of the ID card information or the short code K14. This entry is permitted only for 8160, 975x and 9763 Data Display Terminals with a defined ID card reader.

With this option, lowercase letters (LOWER-CASE) and the function key code (GET-FUNCTION-CODE) are also passed to the application program.
This option is permitted in line mode.

"EXTEND" EXTENDED-LINE-INPUT: when the control character NEW-LINE is detected during input, processing continues and return code 44 is set.
The option is legal in conjunction with LINE-MODE and an output option with ’extended output’.

"LOWEXT" LOWER-CASE-EXTEND: combination of LOWER-CASE and EXTENDED-LINE-INPUT.
The option is legal in conjunction with LINE-MODE and an output option with ’extended output’.

"CFDLOW"	CONF-DATA-LOW-CASE: combination of CONFIDENTIAL-DATA and LOWER-CASE. Legal in line mode.
"GFCLOW"	FCT-CODE-LOW-CASE: combination of GET FUNCTION CODE and LOWER-CASE. Legal in line mode.
"GFCCFD"	FCT-CODE-CONF-DATA: combination of GET FUNCTION CODE and CONFIDENTIAL-DATA. Legal in line mode.
"GFCCDL"	FCT-CODE-CONF-DATA-LCASE: combination of GET-FUNCTION-CODE, CONFIDENTIAL-DATA and LOWER-CASE. Legal in line mode.
"GFCEXT"	FCT-CODE-EXTEND: combination of GET-FUNCTION-CODE and EXTENDED-LINE-INPUT. The option is legal in conjunction with LINE-MODE and an output option with ’extended output’.
"GFCEXL"	FCT-CODE-EXTEND-LCASE: combination of GET-FUNCTION-CODE, EXTENDED-LINE-INPUT and LOWER-CASE. The option is legal in conjunction with LINE-MODE and an output option with ’extended output’.
"NOHDRL"	DELETE-DEVICE-HEADER-LCASE: combination of DELETE-DEVICE-HEADER and LOWER-CASE. Legal in physical mode only.
"VTCBU"	VTSUCB-USED: a VTSU control block is used. The VTSUCB area must be defined and specified in the function call. If the VTSUCB is not to be used, the VTSUCB area may not be specified in the function call.
READLENGTH	The application program writes the length of the input area, including 4 bytes for the record length field, in this item (4 bytes). This is a mandatory item, but there is no default value.
ISAM-REQUEST	The application program specifies in this item (1 byte) how the ISAM key of the record to be read is to be edited.
	The following entries are possible:
"N"	NO-KEY: the ISAM key is removed from the record.
"K"	GET-KEY (default): the ISAM key is not processed.
"L"	GET-KEY-LENGTH: the ISAM key length (less 1) is requested and stored in the KEY-LENGTH item.

"P"	GET-KEY-POSITION: the ISAM key position (less 1) is requested and stored in the KEY-POSITION item.
"B"	GET-KEY-POSITION-LENGTH: both the ISAM key position (less 1) and length (less 1) are requested and are stored in the KEY-POSITION and KEY-LENGTH items, respectively.

ASSIGNMENT-REQUEST

In this item (1 byte) the application program specifies whether or not it requires notification of any changes in the SYSDTA assignment. The following entries are possible:

"N"	NO-ASSIGNMENT-CODE (default): the application program is not informed of any changes in the assignment.
"G"	GET-ASSIGNMENT-CODE: the application program is notified; the changed assignment is stored in the ASSIGNMENT item.

COPYMEM-ID This field (2 bytes) is used to identify the COPY member. So that TIAMINFO can be used in the FILE SECTION, the user must initialize this field with the value 1.

TIAM-RETURN-INFO

After a call this area in TIAM-CONTROL-INFO contains information and return codes from TIAM for the application program.

TIAM-RC This item (2 bytes) contains the TIAM error codes. The meaning of the error codes is given in the descriptions of the calls.

ASSIGNMENT If requested in the RDATA call, this item (2 bytes) contains the assignment key.
The following entries are possible:

0	no change in assignment
1	SYSDTA is a floppy disk unit
4	SYSDTA is a SAM file
8	SYSDTA is an ISAM file
20	SYSDTA is a terminal.
24	SYSDTA is an SDF-P variable
32	SYSDTA is a member of a PLAM library.

KEY-POSITION If requested in the RDATA call, the item (2 bytes) contains the following value after return to the COBOL program:

ISAM key position - 1

KEY-LENGTH If requested in the RDATA call, this item (2 bytes) contains the following value after return to the COBOL program:

ISAM key length - 1

The descriptions of the calls explain which items are used in which TIAM COBOL calls.

4.2 TIAM COBOL calls

The following COBOL calls are available to the programmer:

CALL "RDATA" (see page 143)

CALL "WROUT" (see page 150)

CALL "WRTRD" (see page 155)

A diagram has been included for each call showing

– the data structures used by the call,

– the areas that must or can be filled before the call,

– the information returned by TIAM.

Key to the shading in the diagrams:

In this area the application program must enter the relevant parameters before the call.

Entries in these areas are required only under certain conditions.

The unshaded areas are optional, i.e. entries are only required if the function/characteristic is desired.

This area contains information returned by TIAM.

4.2.1 Read data from SYSDTA: CALL "RDATA"

This call enables the COBOL programmer to read a data record from SYSDTA. SYSDTA may be a floppy disk unit, an element in a PLAM library, and SDF-P variable, a cataloged SAM or ISAM file, or the terminal running the task.

Format of call

```
CALL "RDATA" USING TIAM-CONTROL-INFO
                   user-area
                   [VTSUCB].
```

Meaning of the USING entries:

TIAM-CONTROL-INFO

controls the TIAM call. TIAM stores the return information concerning the execution of the RDATA call in this data structure.

user-area

10 name of input area. TIAM writes the (actual) length of the input message, including the record length item, in the first four bytes of this area. Therefore, when this area is defined, the message items proper should be prefixed with a 4-byte length item, e.g. as follows:

```
01  BENUTZERBEREICH.                                (user area)
    41  LAENGENFELD    PIC 9(5) COMP SYNC.         (length item)
    41  TEXTFELD       PIC X(m).                   (text item)
```

Before the RDATA call is issued, the length of this input area (i.e. the value m+4) must be entered in the READLENGTH item of TIAM-CONTROL-INFO. If the message is longer than the input area, it is truncated and the appropriate error code is returned.

VTSUCB

The VTSUCB should be specified for message editing (see section on the VTSU control block in the VTSU manual [1]). "VTCBU" must then be specified in the edit options item.

Before a **CALL RDATA** is issued to read from SYSDTA, these mandatory parameters must be entered in the following items:

1 Item **EDIT-MODE** of EDIT-IN (TIAM-CONTROL-INFO) **must** have the value "L" (for line mode; default option).

3 Item **READLENGTH** of TIAM-CONTROL-INFO **must** contain the length of the input area (including 4 bytes for the length item).

The following entries in TIAM-CONTROL-INFO are optional:

2 When particular input options are required (see page 133), the appropriate value must be entered in the **EDIT-OPTIONS** item of EDIT-IN (e.g. "NOCORR", "LCASE", "GETFC", "CFDATA", "GETIC", "CFDLOW", "GFCLOW", "GFCCFD" or "GFCCDL"). If the VTSUCB is in use, "VTCBU" must be specified here.

4 If information is required from TIAM on the ISAM key of the input message, one of the values "N", "K", "L", "P" or "B" should be entered in the **ISAM-REQUEST** item.

5 For the application program to receive notification of any changes in the assignment of SYSDTA, "G" should be entered in the **ASSIGNMENT-REQUEST** item.

TIAM returns the following:

6 The TIAM error code in item **TIAM-RC** (see below). The error code 0 indicates that the call was completed without errors.

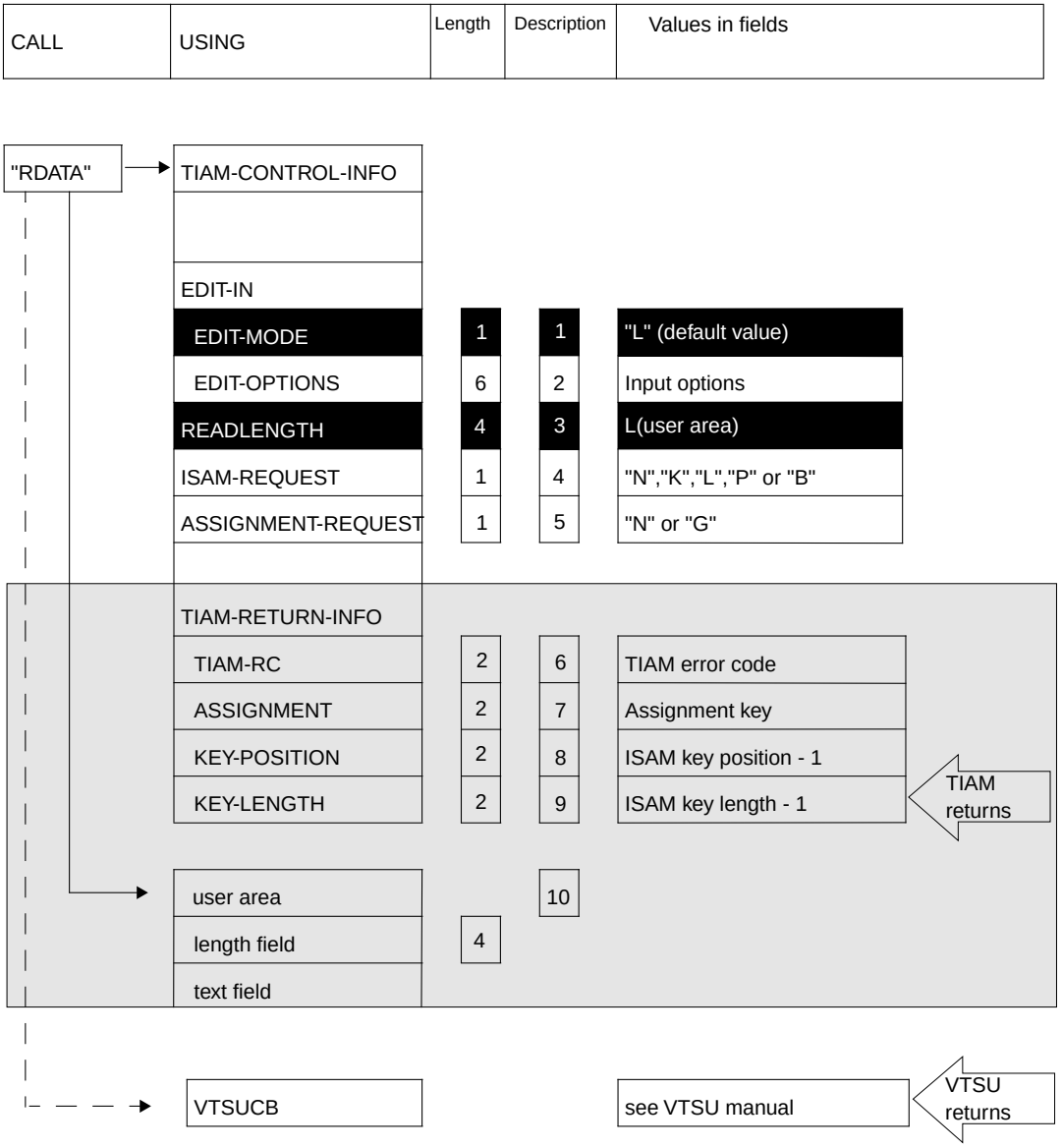
7 The assignment key in the **ASSIGNMENT** item if requested in the ASSIGNMENT-REQUEST item.

8 The value "ISAM key position − 1" in **KEY-POSITION** if requested in the ISAM-REQUEST item.

9 The value 'ISAM key length − 1' in the **KEY-LENGTH** item if requested in the ISAM-REQUEST item.

The following error codes may be returned for a CALL "RDATA":

0	normal termination
4	unrecoverable I/O error
8	parameter error
12	record truncation during reading
16	end of file
32	illegal parameter, corrected by the system.
48	VTSU has detected an error, see complete RC in the VTSUCB (only applicable if the VTSUCB is in use)


CALL
USING
Length
Description
Values in fields
"RDATA"
TIAM-CONTROL-INFO
EDIT-IN
EDIT-MODE
1
1
"L" (default value)
EDIT-OPTIONS
6
2
Input options
READLENGTH
4
3
L(user area)
ISAM-REQUEST
1
4
"N","K","L","P" or "B"
ASSIGNMENT-REQUEST
1
5
"N" or "G"
TIAM-RETURN-INFO
TIAM-RC
2
6
TIAM error code
ASSIGNMENT
2
7
Assignment key
KEY-POSITION
2
8
ISAM key position - 1
KEY-LENGTH
2
9
ISAM key length - 1
TIAM returns
user area
10
length field
4
text field
VTSUCB
see VTSU manual
VTSU returns

Programming example: RDATA

```
************************************************
* Example of R D A T A                         *
* The RDATA function is used to read records.  *
* These records are then output by means of    *
* the WROUT function. The length of the WROUT  *
* function. The length of the ISAM key (less 1) *
* and the position of the ISAM key (less 1) are *
* also output.                                 *
* The program is terminated upon reaching      *
* end of file (EOF) or when "END" is input.)   *
************************************************
*
 IDENTIFICATION DIVISION.
 PROGRAM-ID. RTEST.
 ENVIRONMENT DIVISION.
*
 DATA DIVISION.
 WORKING-STORAGE SECTION.
     COPY TIAMINFO.
     COPY TIAMCTRC.
*
 01  INAREA.
     41  MSGLENGTH  PIC 9(5)  COMP SYNC.
     41  TXT        PIC X(40).
*
 01  OUTAREA-TXT.
     41  MSGLEN     PIC 9(5)  COMP SYNC  VALUE 131.
     41  FILLER     PIC X.
     41  TXT-INPUT  PIC X(40).
     41  STZ1       PIC X.
     41  TXT-KEYPOS PIC X(39) VALUE
              "POSITION DES ISAM-SCHLUESSELS MINUS 1: ".              (1)
     41  KPOSITION  PIC X(4).
     41  STZ2       PIC X.
     41  TXT-KEYLEN PIC X(37) VALUE
              "LAENGE DES ISAM-SCHLUESSELS MINUS 1: ".                (2)
     41  KLENGTH    PIC X(4).
*
 01  OUTAREA-ERROR.
     41  MSGLENGTH  PIC 9(5) COMP SYNC VALUE 25.
     41  FILLER     PIC X.
     41  TXT        PIC X(20).
*
*
 PROCEDURE DIVISION.
 ANF.
     MOVE "L"       TO EDIT-MODE     IN EDIT-IN.
     MOVE "LCASE"   TO EDIT-OPTIONS  IN EDIT-IN.
     MOVE "B"       TO ISAM-REQUEST.
     MOVE 44        TO READLENGTH.
     MOVE NEW-LINE  TO STZ1  IN OUTAREA-TXT.
     MOVE NEW-LINE  TO STZ2  IN OUTAREA-TXT.
     MOVE 1         TO COPYMEM-ID.

*
 READLOOP.
     MOVE " "      TO TXT IN INAREA.
     CALL "RDATA"  USING  TIAM-CONTROL-INFO
```

```
                          INAREA.
*
     IF  TIAM-RC IN TIAM-RETURN-INFO IS NOT EQUAL 16
         AND TXT IN INAREA IS NOT EQUAL "END"
         IF  TIAM-RC IN TIAM-RETURN-INFO IS ZERO
             MOVE TXT          IN INAREA
                  TO TXT-INPUT IN OUTAREA-TXT
             MOVE KEY-POSITION IN TIAM-RETURN-INFO
                  TO KPOSITION IN OUTAREA-TXT
             MOVE KEY-LENGTH   IN TIAM-RETURN-INFO
                  TO KLENGTH   IN OUTAREA-TXT
             CALL "WROUT"  USING    TIAM-CONTROL-INFO
                                    OUTAREA-TXT
             GO TO READLOOP
             ELSE
             MOVE "FEHLERHAFTE EINGABE!"                          (3)
                  TO TXT       IN OUTAREA-ERROR
             CALL "WROUT"  USING    TIAM-CONTROL-INFO
                                    OUTAREA-ERROR
             GO TO READLOOP
         ELSE
         MOVE "ENDE DER EINGABE!"                                 (4)
              TO TXT       IN OUTAREA-ERROR.
         CALL "WROUT"  USING    TIAM-CONTROL-INFO
                                OUTAREA-ERROR.
     STOP RUN.
```

Key

(1) Position of ISAM key -1

(2) Length of ISAM key -1

(3) Errored input

(4) End of input

Programming example: RDATA using the VTSUCB

```
*******************************************
* Example of R D A T A                    *
* The RDATA function is used to read records.*
* These records are then output by means of  *
* the WROUT function. The length of the ISAM *
* key (less 1) and the position of the ISAM  *
* key (less 1) are also output.              *
* The program is terminated upon reaching    *
* end of file (EOF) or when "END" is input.) *
  *******************************************
*
 IDENTIFICATION DIVISION.
 PROGRAM-ID. RTEST.
 ENVIRONMENT DIVISION.
*
 DATA DIVISION.
 WORKING-STORAGE SECTION.
     COPY TIAMINFO.
     COPY TIAMCTRC.
     COPY VTSUCBC.
*
 01  INAREA.
     41  MSGLENGTH  PIC 9(5)  COMP SYNC.
     41  TXT        PIC X(40).
*
 01  OUTAREA-TXT.
     41  MSGLEN     PIC 9(5)  COMP SYNC  VALUE 131.
     41  FILLER     PIC X.
     41  TXT-INPUT  PIC X(40).
     41  STZ1       PIC X.
     41  TXT-KEYPOS PIC X(39) VALUE
              "POSITION DES ISAM-SCHLUESSELS MINUS 1: ".       (1)
     41  KPOSITION  PIC X(4).
     41  STZ2       PIC X.
     41  TXT-KEYLEN PIC X(37) VALUE
              "LAENGE DES ISAM-SCHLUESSELS MINUS 1: ".         (2)
     41  KLENGTH    PIC X(4).
*
 01  OUTAREA-ERROR.
     41  MSGLENGTH  PIC 9(5) COMP SYNC VALUE 25.
     41  FILLER     PIC X.
     41  TXT        PIC X(20).
*
*
*
 PROCEDURE DIVISION.
 ANF.
     MOVE "L"       TO MODUS         IN VTSUCB.
     MOVE "Y"       TO LOWER-CHARS   IN VTSUCB.
     MOVE "B"       TO ISAM-REQUEST.
     MOVE 44        TO READLENGTH.
     MOVE NEW-LINE  TO STZ1  IN OUTAREA-TXT.
     MOVE NEW-LINE  TO STZ2  IN OUTAREA-TXT.
     MOVE 1         TO COPYMEM-ID.
```

```
*
 READLOOP.
     MOVE " "      TO TXT IN INAREA.
     MOVE "VTCBU"  TO EDIT-OPTIONS IN EDIT-IN.
     CALL "RDATA"  USING  TIAM-CONTROL-INFO
                          INAREA
                          VTSUCB.
*
     IF  TIAM-RC IN TIAM-RETURN-INFO IS NOT EQUAL 16
         AND TXT IN INAREA IS NOT EQUAL "END"
         IF  TIAM-RC IN TIAM-RETURN-INFO IS ZERO
             MOVE TXT          IN INAREA
                  TO TXT-INPUT IN OUTAREA-TXT
             MOVE KEY-POSITION IN TIAM-RETURN-INFO
                  TO KPOSITION IN OUTAREA-TXT
             MOVE KEY-LENGTH   IN TIAM-RETURN-INFO
                  TO KLENGTH   IN OUTAREA-TXT
             CALL "WROUT"  USING    TIAM-CONTROL-INFO
                                    OUTAREA-TXT
             GO TO READLOOP
             ELSE
             MOVE "FEHLERHAFTE EINGABE!"                        (3)
                  TO TXT       IN OUTAREA-ERROR
             CALL "WROUT"  USING    TIAM-CONTROL-INFO
                                    OUTAREA-ERROR
             GO TO READLOOP
         ELSE
         MOVE "ENDE DER EINGABE!"                               (4)
              TO TXT       IN OUTAREA-ERROR.
         CALL "WROUT"  USING    TIAM-CONTROL-INFO
                                OUTAREA-ERROR.
     STOP RUN.
```

Key

(1) Position of ISAM key -1

(2) Length of ISAM key -1

(3) Errored input

(4) End of input

4.2.2 Write data to SYSOUT: CALL "WROUT"

This call enables the COBOL programmer to output a message to SYSOUT. SYSOUT may be an element in a PLAM library, an SDF-P variable, a cataloged SAM or ISAM file or the terminal running the task. As a special function integrated into CALL "WROUT", format handling with FHS is supported; FHS is called internally by TIAM.

Format of call

```
CALL "WROUT" USING TIAM-CONTROL-INFO
                   [VTSUCB]
                   [FHS-MAIN-PAR].
```

Meaning of the USING entries:

TIAM-CONTROL-INFO

controls the TIAM call. TIAM stores the return information concerning the execution of the WROUT call in this data structure.

user-area

4 name of output area. This area must have the following structure:

– A 4-byte length item must have been defined and must precede the output text.

– In line mode and physical mode, the length item is followed by a 1-byte filler (not required when integrated FHS is used).

– The last part of the output area is the text item containing the output message.

An example of the required declarations is:

– in line mode or physical mode:

```
01  BENUTZERBEREICH.                              (user area)
    41  LAENGENFELD    PIC 9(5) COMP SYNC.        (length item)
    41  FILLER         PIC X.                     (filler)
    41  AUSGABETEXT    PIC X(m).                  (output text)
```

Before issuing the WROUT call, the user must enter the length of this output area (i.e. the value m+5) in the length item (line and physical mode). The filler may have any arbitrary value; it does not affect the output.

– in FHS mode:

```
01     BENUTZERBEREICH.                           (user area)
       41 LAENGENFELD         PIC 9(5) COMP.     (length item)
       COPY addressing-aid
```

where "addressing-aid" is the COBOL addressing aid supplied by FHS or IFG when the format is generated. FHS supplies the entries for the length item.

VTSUCB

The VTSUCB should be specified for message editing (see the VTSU manual [1]). "VTCBU" must then be specified in the edit options item.

FHS-MAIN-PAR (with FHS only)

Controls formatting. See the "FHS" User Guide for a description of the items requiring parameter entries.

Before issuing a **CALL WROUT**, the user must enter these mandatory parameters in the following items:

1 Item **EDIT-MODE** of EDIT-OUT (TIAM-CONTROL-INFO) **must** have the value "L" (line mode), "P" (physical mode) or "F" (FHS mode). If EDIT-MODE="F" (FHS), TIAM will not interpret the remaining items of TIAM-CONTROL-INFO.
If EDIT-MODE="L" (line-Mode), control characters can be inserted in the message. TIAM supplies these control characters via the COPY member TIAMCTRC (see page 132).

5 If EDIT-MODE= "L" or "P" (line or physical mode), the first 4 bytes of the output area **must** contain the message length (including length item and filler).

The following entries in TIAM-CONTROL-INFO are optional:

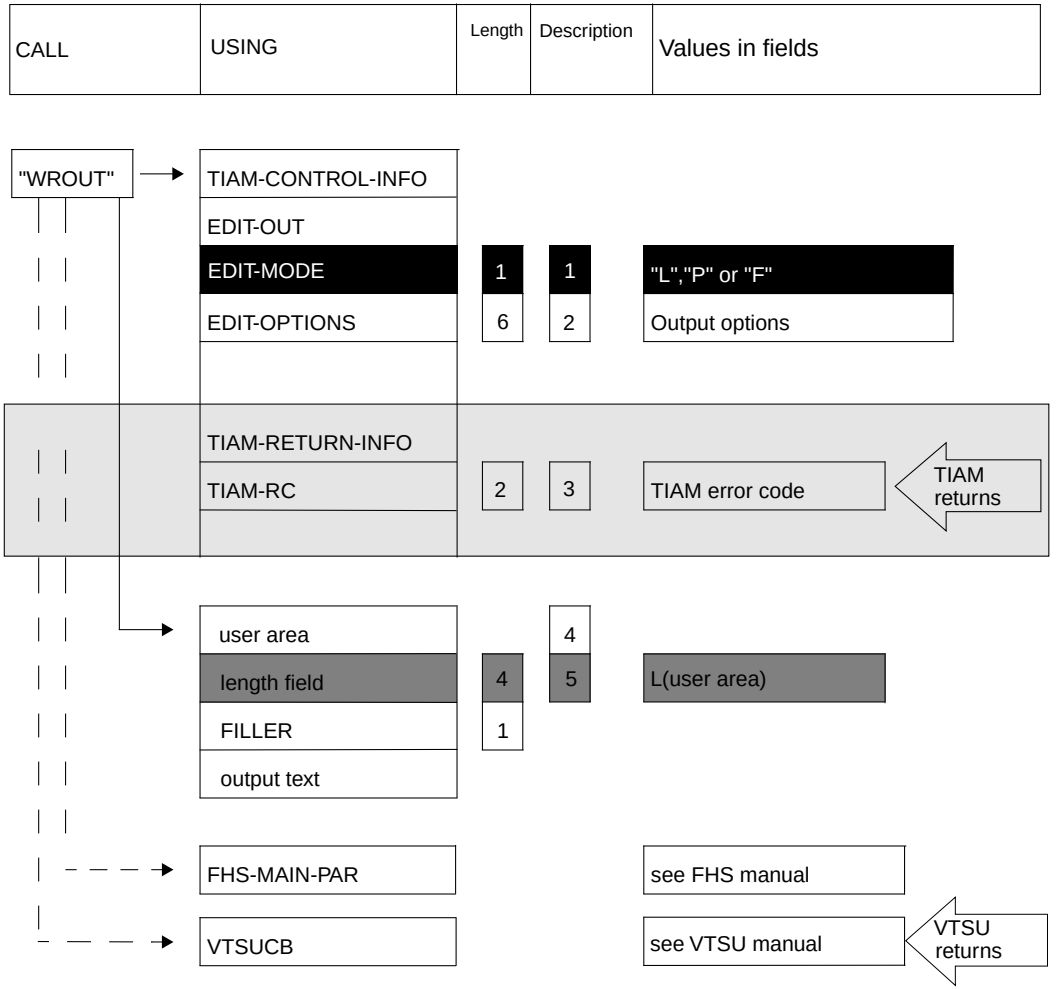
2 When particular output options are required (see page 133), the appropriate value must be entered in the **EDIT-OPTIONS** item of EDIT-OUT (e.g. "HCOPY", "HOMOUT", "HEADER", "ETB", "INFO", "TRNCD", "EXTEND", "NLOGC", "BELL", "OWRITE", "BELHOM", "BELEXT", "BELINF", "BELNLC", "HCNLC").
This item is not interpreted in FHS mode (EDIT-MODE="F"). If the VTSUCB is in use, "VTCBU" must be specified here.

TIAM returns the following:

3 the TIAM error code in item **TIAM-RC** (see below). Error code 0 indicates that the call was error-free.

The following error codes may be returned for a CALL "WROUT":

0	normal termination
4	unrecoverable I/O error
8	parameter error
12	record truncation during writing
32	illegal parameter, corrected by the system
36	no FHS formatting possible since the FHS modules are missing
40	FHS return code not equal to zero (see FHS return code)
48	VTSU has detected an error, see complete RC in the VTSUCB (only applicable if the VTSUCB is in use)



Note

When using integrated FHS mode (MODE="F"), the user should ensure on linking the application program that the FHS module **MFHSCALL** is linked explicitly with the COBOL program via an INCLUDE statement. If WROUT is called in MODE="F" and the FHS module is missing, TIAM will reject the call with a return code.

Programming example: WROUT in line mode

```
*******************************************
* Example of WROUT                        *
* A record is output to the terminal in   *
* line mode by means of WROUT.            *
*******************************************

 IDENTIFICATION DIVISION.
 PROGRAM-ID.   WTEST.
 ENVIRONMENT DIVISION.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
*
     COPY TIAMINFO.
*
     COPY TIAMCTRC.
*
 01  OUTPUT-AREA.
     41  MSGLENGTH PIC 9(5)  COMP SYNC  VALUE 85.
     41  FILLER    PIC X.
     41  TXT1      PIC X(36) VALUE
                       "DAS IST EIN BEISPIEL FUER DEN WROUT.".       (1)
     41  STZ1      PIC X.
     41  TXT2      PIC X(20) VALUE "DIESER TEXT WIRD IM ".          )
     41  STZ2      PIC X.                                           )
     41  TXT3      PIC X(9)  VALUE "LINE-MODE".                     } (2)
     41  STZ3      PIC X.                                           )
     41  TXT4      PIC X(12) VALUE " AUSGEGEBEN.".                  )
*
*
 PROCEDURE DIVISION.
 HP.
     MOVE NEW-LINE       TO STZ1  IN OUTPUT-AREA.
     MOVE EMPH-LAYOUT1   TO STZ2  IN OUTPUT-AREA.
     MOVE NORMAL-LAYOUT  TO STZ3  IN OUTPUT-AREA.
*
     MOVE  "L"      TO EDIT-MODE     IN EDIT-OUT.
     MOVE  "NOOPTS" TO EDIT-OPTIONS  IN EDIT-OUT.
     MOVE  1        TO COPYMEM-ID.
     CALL  "WROUT"  USING    TIAM-CONTROL-INFO OUTPUT-AREA.
*
     STOP RUN.
```

Key

(1) This is an example of WROUT.

(2) This text is being output in line mode.

Programming example: WROUT in line mode using the VTSUCB

```
******************************************
* Example of WROUT                       *
* A record is output to the terminal     *
* in line mode by means of WROUT.        *
******************************************

 IDENTIFICATION DIVISION.
 PROGRAM-ID.   WTEST.
 ENVIRONMENT DIVISION.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
*
     COPY TIAMINFO.
*
     COPY TIAMCTRC.
*
     COPY VTSUCBC.
*
 01  OUTPUT-AREA.
     41  MSGLENGTH PIC 9(5)  COMP SYNC  VALUE 85.
     41  FILLER    PIC X.
     41  TXT1      PIC X(36) VALUE
                       "DAS IST EIN BEISPIEL FUER DEN WROUT.".       (1)
     41  STZ1      PIC X.
     41  TXT2      PIC X(20) VALUE "DIESER TEXT WIRD IM ".          ⎫
     41  STZ2      PIC X.                                           ⎪
     41  TXT3      PIC X(9)  VALUE "LINE-MODE".                     ⎬ (2)
     41  STZ3      PIC X.                                           ⎪
     41  TXT4      PIC X(12) VALUE " AUSGEGEBEN.".                  ⎭
*
*
 PROCEDURE DIVISION.
 HP.
     MOVE NEW-LINE       TO STZ1  IN OUTPUT-AREA.
     MOVE EMPH-LAYOUT1   TO STZ2  IN OUTPUT-AREA.
     MOVE NORMAL-LAYOUT  TO STZ3  IN OUTPUT-AREA.
*
     MOVE  1         TO COPYMEM-ID.
     MOVE  "L"       TO MODUS          IN VTSUCB.
     MOVE  "VTCBU"   TO EDIT-OPTIONS   IN EDIT-OUT.
     CALL  "WROUT"   USING    TIAM-CONTROL-INFO OUTPUT-AREA
                              VTSUCB.
*
     STOP RUN.
```

Key

(1) This is an example of WROUT.

(2) This text is being output in line mode.

4.2.3 Terminal tandem write/read: CALL "WRTRD"

This call enables the COBOL programmer to output a message to a terminal and immediately afterwards read a message from the terminal. Apart from the message transmitted to the terminal, no other prompting character appears as an input request. CALL "WRTRD" can only be used with the TIAM access method.
As a special function integrated into the call, format handling with the aid of FHS is supported; FHS is called internally by TIAM.

On 8160, 9749, 975x and 9763 Data Display Terminals, a CALL "WRTRD" input causes the keyboard to be locked, preventing any further input until after the next output has been completed. Only short codes are allowed.

Format of call

```
CALL "WRTRD" USING TIAM-CONTROL-INFO
                   user-area1
                   user-area2
                   [VTSUCB]
                   [FHS-MAIN-PAR].
```

Meaning of the USING entries:

TIAM-CONTROL-INFO

controls the TIAM call. TIAM stores the return information on the execution of the WRTRD call in this item.

user-area1

7 name of output area. This area must have the following structure:

- A 4-byte length item must have been defined and must precede the output text.
- In the line and physical mode the length item is followed by a 1-byte filler (omitted when integrated FHS is used).
- The last part of the output area is the text item containing the output message.

  An example of the required declarations is:

- line mode or physical mode:

```
01  BENUTZERBEREICH1.                          (user area 1)
    41  LAENGENFELD    PIC 9(5) COMP SYNC.     (length item)
    41  FILLER         PIC X.                  (filler)
    41  AUSGABETEXT    PIC X(m).               (output text)
```

  Before the WRTRD call is issued, the length of this output area (i.e. the value m+5) must be entered in the length item (line and physical mode). The filler may have any arbitrary value; it does not affect the output.

– FHS mode:

```
01  BENUTZERBEREICH1.                          (user area 1)
    41 LAENGENFELD         PIC 9(5) COMP.      (length item)
    COPY addressing-aid.
```

where "addressing-aid" is the COBOL addressing aid supplied by FHS or IFG when the format is generated. FHS supplies the entries for the length item.

user-area2

9 name of input area. This area must have the following structure:

– A 4-byte length item must have been defined and must precede the input item.

– The length item is followed by the text item proper into which TIAM writes the input message.

An example of the required declarations is:

– line or physical mode:

```
01  BENUTZERBEREICH2.                          (user area 2)
    41  LAENGENFELD    PIC 9(5) COMP SYNC.     (length item)
    41  EINGABEFELD    PIC X(m).               (input item)
```

Before issuing the WRTRD call, the user must enter the length of this input area (i.e. the value m+4) in the READLENGTH item (line and physical mode).

– FHS mode:

```
01  BENUTZERBEREICH2.                          (user area 2)
    41 LAENGENFELD         PIC 9(5) COMP.      (length item)
    COPY addressing-aid.
```

where "addressing-aid" is the COBOL addressing aid supplied by FHS or IFG when the format is generated. FHS supplies the entries for the length item.

VTSUCB

The VTSUCB should be specified for message editing (see the section on the VTSU control block in the VTSU manual [1]). "VTCBU" must then be specified in the edit options item.

FHS-MAIN-PAR (with FHS only)

handles formatting. See the "FHS User Guide" for a description of the items requiring parameter entries.

Before a **CALL WRTRD** is issued, these mandatory parameters must be entered in the following items:

1 The **EDIT-MODE** item of EDIT-OUT (TIAM-CONTROL-INFO) **must** have one of the values "L" (line mode), "P" (physical mode) or "F" (FHS mode).
If EDIT-MODE="F" (FHS), TIAM will not interpret the remaining items of TIAM-

CONTROL-INFO.
If EDIT-MODE="L" (line mode), control characters can be inserted in the message. TIAM supplies these control characters with the COPY member TIAMCTRC (see page 132).

In line mode or physical mode (EDIT-MODE= "L" or "P"), the following additional entries are mandatory:

3 The **EDIT-MODE** item in EDIT-IN must have either the value "L" (for line mode) or "P" (for physical mode).

5 The **READLENGTH** item must contain the length of the input area (including the length item).

8 The first 4 bytes of the output area (length item) must contain the length of the message (including the length item and filler).

The following entries in TIAM-CONTROL-INFO are optional:

2 When particular output options are required (see page 133), the appropriate value must be entered in the **EDIT-OPTIONS** item of EDIT-OUT (e.g. "HCOPY", "HOMOUT", "HEADER", "ETB", "TRNCD", "EXTEND", "NLOGC", "BELL", "OWRITE", "BELHOM", "BELEXT", "BELNLC" or "HCNLC").
This item is not interpreted in FHS mode. If the VTSUCB is in use, "VTCBU" must be specified here.

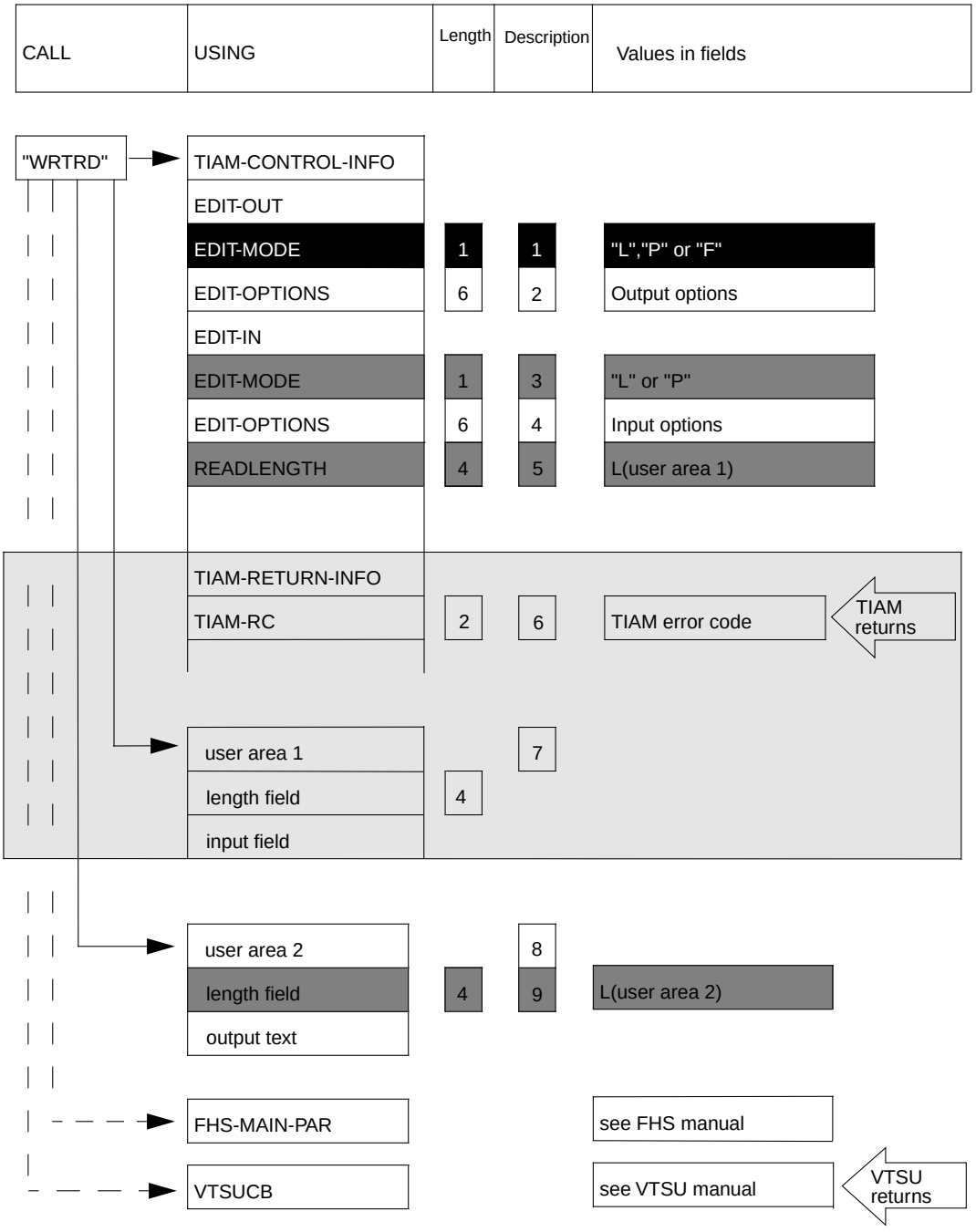
4 When particular input options are required (see page 133), the appropriate value must be entered in the **EDIT-OPTIONS** item of EDIT-IN (e.g. "NOCORR", "LCASE", "NOHDR", "GETFC","CFDATA", "GETIC", "EXTEND", "LOWEXT", "CFDLOW", "GFCLOW", "GFCCFD", "GFCCDL", "GFCEXT", "GFCEXL" or "NOHDRL").
This item is not interpreted in FHS mode. If the VTSUCB is in use, "VTCBU" must be specified here.

TIAM returns the following:

6 the TIAM error code in the **TIAM-RC** item (see below). Error code 0 indicates that the call was error-free.

The following error codes may be returned for a CALL "WRTRD":

0	normal termination
4	unrecoverable I/O error
8	parameter error
12	record truncation during reading
16	record truncation during writing
20	WRTRD called in a batch task
24	end of input
32	illegal parameter corrected by the system
36	no FHS formatting possible since FHS modules are missing
40	FHS return code not equal to zero (see FHS return code)
44	the character NEW LINE was detected in input (’extended line input’ only)
48	VTSU has detected an error, see complete RC in the VTSUCB (only applicable if the VTSUCB is in use)


CALL
USING
Length
Description
Values in fields
"WRTRD"
TIAM-CONTROL-INFO
EDIT-OUT
EDIT-MODE
1
1
"L","P" or "F"
EDIT-OPTIONS
6
2
Output options
EDIT-IN
EDIT-MODE
1
3
"L" or "P"
EDIT-OPTIONS
6
4
Input options
READLENGTH
4
5
L(user area 1)
TIAM-RETURN-INFO
TIAM-RC
2
6
TIAM error code
TIAM returns
user area 1
7
length field
4
input field
user area 2
8
length field
4
9
L(user area 2)
output text
FHS-MAIN-PAR
see FHS manual
VTSUCB
see VTSU manual
VTSU returns

Note

When using integrated FHS (MODE="F"), the user should ensure on linking the application program that the FHS module **MFHSCALL** is linked explicitly with the COBOL program via an INCLUDE statement. If WRTRD is called in MODE="F" and the FHS module is missing, TIAM will reject the call with a return code.

Programming example: WRTRD in line mode

```
*****************************************
* Example of W R T R D                  *
* The WRTRD call is used to request input *
* from the terminal, which is then output *
* by means of the WROUT function.       *
*****************************************

 IDENTIFICATION DIVISION.
 PROGRAM-ID.   WRTEST.
 ENVIRONMENT DIVISION.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
*
     COPY TIAMINFO.
*
     COPY TIAMCTRC.
*
 01  INAREA.
     41  MSGLENGTH PIC 9(5) COMP SYNC.
     41  TXT       PIC X(40) VALUE " ".
*
 01  OUTAREA.
     41  MSGLENGTH PIC 9(5) COMP SYNC  VALUE 46.
     41  FILLER    PIC X.
     41  TXT       PIC X(40) VALUE
                  "BITTE MAX. 40 BYTE LANGEN TEXT EINGEBEN.".         (1)
     41  STZ       PIC X.
*
 01  OUTAREA-INPUT-TXT.
     41  MSGLENGTH PIC 9(5) COMP SYNC  VALUE 45.
     41  FILLER    PIC X.
     41  TXT       PIC X(40).
*
 01  OUTAREA-ERROR.
     41  MSGLENGTH PIC 9(5) COMP SYNC  VALUE 25.
     41  FILLER    PIC X.
     41  TXT       PIC X(20) VALUE
                   "FEHLERHAFTE EINGABE!".                            (2)
*
 PROCEDURE DIVISION.
 ANF.
*
     MOVE "L"      TO EDIT-MODE    IN EDIT-OUT.
     MOVE "NOOPTS" TO EDIT-OPTIONS IN EDIT-OUT.
     MOVE "L"      TO EDIT-MODE    IN EDIT-IN.
     MOVE "LCASE"  TO EDIT-OPTIONS IN EDIT-IN.
     MOVE 44       TO READLENGTH.
     MOVE NEW-LINE TO STZ  IN OUTAREA.
     MOVE  1       TO COPYMEM-ID.
```

```
     CALL "WRTRD"  USING    TIAM-CONTROL-INFO
                            OUTAREA INAREA.
*
     IF  TIAM-RC IN TIAM-RETURN-INFO IS ZERO
         MOVE TXT IN INAREA  TO TXT IN OUTAREA-INPUT-TXT
         CALL "WROUT"  USING    TIAM-CONTROL-INFO
                                OUTAREA-INPUT-TXT
         ELSE
         CALL "WROUT"  USING    TIAM-CONTROL-INFO
                                OUTAREA-ERROR.
     STOP RUN.
```

Key

(1) Please input text of 40 bytes maximum.

(2) Errored input

Programming example: WRTRD in line mode using the VTSUCB

```
  ******************************************
  * Example of W R T R D                   *
  * The WRTRD call is used to request input *
  * from the terminal, which is then output *
  * by means of the WROUT function.        *
  ******************************************

   IDENTIFICATION DIVISION.
   PROGRAM-ID.   WRTEST.
   ENVIRONMENT DIVISION.
   DATA DIVISION.
   WORKING-STORAGE SECTION.
  *
       COPY TIAMINFO.
  *
       COPY TIAMCTRC.
  *
       COPY VTSUCBC.
  *
   01  INAREA.
       41  MSGLENGTH PIC 9(5) COMP SYNC.
       41  TXT       PIC X(40) VALUE " ".
  *
   01  OUTAREA.
       41  MSGLENGTH PIC 9(5) COMP SYNC  VALUE 46.
       41  FILLER    PIC X.
       41  TXT       PIC X(40) VALUE
                    "BITTE MAX. 40 BYTE LANGEN TEXT EINGEBEN.".        (1)
       41  STZ       PIC X.
  *
   01  OUTAREA-INPUT-TXT.
       41  MSGLENGTH PIC 9(5) COMP SYNC  VALUE 45.
       41  FILLER    PIC X.
       41  TXT       PIC X(40).
  *
   01  OUTAREA-ERROR.
       41  MSGLENGTH PIC 9(5) COMP SYNC  VALUE 25.
       41  FILLER    PIC X.
```

```
      41  TXT       PIC X(20) VALUE
                    "FEHLERHAFTE EINGABE!".                          (2)
 *
  PROCEDURE DIVISION.
  ANF.
 *
      MOVE "L"      TO MODUS        IN VTSUCB.
      MOVE "Y"      TO LOWER-CHARS  IN VTSUCB.
      MOVE 44       TO READLENGTH.
      MOVE NEW-LINE TO STZ  IN OUTAREA.
      MOVE  1       TO COPYMEM-ID.
      MOVE "VTCBU"  TO EDIT-OPTIONS IN EDIT-OUT.
      CALL "WRTRD"  USING    TIAM-CONTROL-INFO
                             OUTAREA
                             INAREA
                             VTSUCB.
 *
      IF  TIAM-RC IN TIAM-RETURN-INFO IS ZERO
          MOVE TXT IN INAREA  TO TXT IN OUTAREA-INPUT-TXT
          MOVE "NOOPTS"  TO EDIT-OPTIONS IN EDIT-OUT
          CALL "WROUT"  USING    TIAM-CONTROL-INFO
                                 OUTAREA-INPUT-TXT
          ELSE
          MOVE "NOOPTS"  TO EDIT-OPTIONS IN EDIT-OUT.
          CALL "WROUT"  USING    TIAM-CONTROL-INFO
                                 OUTAREA-ERROR.
      STOP RUN.
```

Key

(1) Please input text of 40 bytes maximum.

(2) Errored input

Programming example: WRTRD with FHS

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID.  FHSTIAM.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SPECIAL-NAMES. TERMINAL IS DSS.                                  (1)
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01  USER-AREA.
     41 U-A-LAENGE        PIC 9(5) COMP.                          (2)
     COPY BEISPIEL.                                               (3)
     COPY TIAMINFO.
     COPY FHSMAINP.
 PROCEDURE DIVISION.
*    FHS-MODE
     MOVE "F" TO EDIT-MODE IN EDIT-OUT.
     MOVE 1   TO COPYMEM-ID.
 MARK1.
*          FHS-BEREICHE AUSFUELLEN                                 (4)
     MOVE "BEISPIEL" TO FHS-MAP-NAME
     MOVE "Y" TO FHS-MAP-LIB-OPT
     MOVE "TIAM.MAPLIB" TO FHS-MAP-LIB-NAME.
 MARK2.
     CALL "WRTRD" USING TIAM-CONTROL-INFO
                        USER-AREA
                        USER-AREA
                        FHS-MAIN-PAR.
     IF FHS-MAIN-RC NOT = 0 GO TO FHSFEHLER.                      (5)
     IF TIAM-RC NOT = 0 GO TO TIAMFEHLER.                         (6)
     IF EINGABE = "ENDE" GO TO SCHLUSS.                           (7)
*********************************************************
*                                                       *
*   V E R A R B E I T U N G   D E R   E I N G A B E     *        (8)
*                                                       *
*********************************************************
     GO TO MARK2.
 FHSFEHLER.
     DISPLAY "FORMATIERUNGSFEHLER! RETURNCODE: "                  (9)
             ERROR-CATEGORY " " ERROR-REASON UPON DSS.
     GO TO SCHLUSS.
 TIAMFEHLER.
     DISPLAY "EIN/AUSGABEFEHLER! RETURNCODE: "                   (10)
             TIAM-RC UPON DSS.
 SCHLUSS.                                                        (11)
     STOP RUN.
```

Key

(1) DSS = data display terminal

(2) 41 user area length

(3) Copy example

(4) Fill in FHS areas

(5) FHSFEHLER = FHS error

(6) TIAMFEHLER = TIAM error

(7) If input = "end", go to end.

(8) Processing of input

(9) Display "formatting error! return code: "

(10) Display "I/O error! return code: "

(11) SCHLUSS = end

Programming example: WRTRD in extended line mode

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID.    WRTRDEX.
 ENVIRONMENT DIVISION.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
     COPY TIAMINFO.
     COPY TIAMCTRC.
 01  EINGABE.                                                    (1)
    41 LAENGE   PIC 9(5) COMP SYNC.                             (2)
    41 EINNAM   PIC X(12).
    41 EINVOR   PIC X(12).
    41 EINSTR   PIC X(25).
    41 EINPLZ   PIC X(4).
    41 EINORT   PIC X(12).
 01  AUSGABE.                                                   (3)
    41 LAENGE   PIC 9(5) COMP SYNC VALUE 223.
    41 FILLER   PIC X.
    41 F1NP     PIC X.
    41 F2SPA    PIC X.
    41 F3EM3    PIC X.
    41 TEXT1    PIC X(25) VALUE "BITTE GEBEN SIE NAMEN UND".    (4)
    41 TEXT1A   PIC X(16) VALUE " ANSCHRIFT EIN !".             (5)
    41 F4NL     PIC X.
    41 F5NL     PIC X.
    41 F6NOR    PIC X.
    41 TEXT2    PIC X(6) VALUE "NAME: ".
    41 F7EPA    PIC X.
    41 NAME     PIC X(12) VALUE SPACES.
    41 F8SPA    PIC X.
    41 TEXT3    PIC X(15) VALUE "      VORNAME: ".              (6)
    41 F9EPA    PIC X.
    41 VORNAME  PIC X(12) VALUE SPACES.
    41 F10NL    PIC X.
    41 F11SPA   PIC X.
    41 TEXT4    PIC X(9) VALUE "STRASSE: ".                     (7)
    41 F12EPA   PIC X.
    41 STRASSE  PIC X(25) VALUE SPACES.                         (7)
    41 F13NL    PIC X.
    41 F14SPA   PIC X.
    41 TEXT5    PIC X(9) VALUE "WOHNORT: ".                     (8)
    41 F15EPA   PIC X.
    41 F16NUM   PIC X.
    41 PLZ      PIC X(4) VALUE "0000".
    41 F17SPA   PIC X.
```

```
    41 BLNK     PIC X(2) VALUE "  ".
    41 F18EPA   PIC X.
    41 ORT      PIC X(12) VALUE SPACES.
    41 F19NL    PIC X.
    41 F20NL    PIC X.
    41 F21SPA   PIC X.
    41 TEXT6    PIC X(27) VALUE "BEI PROGRAMMENDE BITTE DEN ".     (9)
    41 TEXT6A   PIC X(23) VALUE "NAMEN 'ENDE' EINGEBEN !".        (10)

PROCEDURE DIVISION.
ANF.
    MOVE "L"       TO EDIT-MODE     IN EDIT-OUT.
    MOVE "EXTEND"  TO EDIT-OPTIONS  IN EDIT-OUT.
    MOVE "L"       TO EDIT-MODE     IN EDIT-IN.
    MOVE "EXTEND"  TO EDIT-OPTIONS  IN EDIT-IN.
    MOVE 223       TO READLENGTH.
    MOVE  1        TO COPYMEM-ID.
    MOVE NEW-LINE  TO F4NL F5NL F10NL F13NL F19NL F20NL.
    MOVE NEW-PAGE  TO F1NP.
    MOVE START-PROT-AREA TO F2SPA F8SPA F11SPA F14SPA F17SPA.
    MOVE START-PROT-AREA TO F21SPA.
    MOVE EMPH-LAYOUT3   TO F3EM3.
    MOVE NORMAL-LAYOUT  TO F6NOR.
    MOVE END-PROT-AREA  TO F7EPA F9EPA F12EPA F15EPA F18EPA.
    MOVE START-NUM-DATA TO F16NUM.
AUSGEBEN.                                                          (11)
    CALL "WRTRD" USING TIAM-CONTROL-INFO AUSGABE EINGABE.          (12)
    IF TIAM-RC IN TIAM-RETURN-INFO IS NOT ZERO GO TO ENDE.         (13)
    IF EINNAM = "ENDE        " GO TO ENDE.                         (13)
    GO TO AUSGEBEN.                                                (11)
ENDE.                                                              (13)
    STOP RUN.
```

Key

(1) EINGABE = input

(2) LAENGE= length

(3) AUSGABE = output

(4),(5) BITTE GEBEN SIE NAMEN UND ANSCHRIFT EIN !
= Please enter name and address!

(6) VORNAME = First name

(7) STRASSE = Street

(8) WOHNORT = City

(9),(10) BEI PROGRAMMENDE BITTE DEN NAMEN 'ENDE' EINGEBEN !
= Please enter name 'end' at end of program!

(11) AUSGEBEN = output

(12) AUSGABE EINGABE = output input

(13) ENDE = end

5 TIAM FORTRAN interface

The definitions of the FORTRAN data structure and the FORTRAN data fields correspond to the definitions of the COBOL data structure and the COBOL data fields. Consequently, this chapter deals only with those aspects of the FORTRAN language and syntax relevant to these data structures. See the chapter on the COBOL interface for detailed information on the data structures and data fields.

Notes

– The way in which the input and output interfaces are used depends on the version of the FORTRAN Compiler and the COMOPT statements entered.

  If the **FORTRAN Compiler V2.1** is used with `COMOPT EXTENDED-SYSTEM=NO`, the FORTRAN Compiler generates a parameter list in which the last listed parameter is marked by a flag bit set to 1. This method of transferring parameters is supported by the TIAM interface module DCCOBRTS.

  If the **FORTRAN Compiler V2.1** is used with `COMOPT EXTENDED-SYSTEM=YES`, the FORTRAN Compiler generates a parameter list in which the end of the list is not marked by a flag bit set to binary 1. The number of parameters in shown by register 0. This method of transferring parameters is defined by an ASSEMBLER interface and is not supported by the TIAM interface module DCCOBRTS.

```
EXTERNAL WRTRD
CALL OLDASS (WRTRD, TIAMCONTROLINFO, user-area1,
             user-area2 [,VTSUCB] [,FHSMAINPAR])
```

  If the **FORTRAN-Compiler V2.2** is used, it generates a parameter list and register 0 contains the number of parameters and register 1 contains a list of the parameter addresses. As of TIAM V11.0, this method of transferring parameters is supported by the TIAM interface module DCCOBRTS.

– The TIAM interface for FORTRAN is available as of TIAM V11.0A.

  The FHS interface for FORTRAN is available as of FHS V7.0.

  The VTSU interface for FORTRAN is available as of VTSU V10.1A.

5.1 Data structures for TIAM FORTRAN calls

TIAM and the application program can communicate by way of data structures which are contained in a library as INCLUDE members and are copied into the application program.

The VTSU-CONTROL-BLOCK data structure

The VTSU-CONTROL-BLOCK data structure is copied into the application program by means of the following %INCLUDE statement.

```
%INCLUDE $TSOS.SYSLIB.VTSU-B.111(VTSUCBF)
```

The VTSU-CONTROL-BLOCK is described in the VTSU manual (see [1] in the Related publications chapter).

The LINE-MODE-CONTROL-CHARACTER data structure

The LINE-MODE-CONTROL-CHARACTER is copied into the application program by means of the following %INCLUDE statement.

```
%INCLUDE $TSOS.SYSLIB.VTSU-B.111(FORCTRC)
```

LINE-MODE-CONTROL-CHARACTER is described in the VTSU manual (see [1] in the Related publications chapter).

Data structure TIAM-CONTROL-INFO

The data structure TIAM-CONTROL-INFO is copied to the application program by means of the following %INCLUDE statement:

```
%INCLUDE $TSOS.SYSLIB.TIAM.110(FORINFO)
```

The structure of TIAM-CONTROL-INFO is shown below. For detailed information on the individual parameters, see page 133 ff.

```
************************************************************************
*        FORINFO          001       920320            TIAM          U *
************************************************************************
*                                                                      *
*       COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1991      *
*                     ALL RIGHTS RESERVED                              *
*                                                                      *
************************************************************************
*
*                                              TIAM CONTROL INFO
      CHARACTER*64  TIAMCONTROLINFO
*
*                                                EDIT OUT
      CHARACTER*16  EDITOUT
*                                                  EDIT MODE
      CHARACTER*1   EDITMODEOUT
*                     'L'       - LINE MODE
*                     'P'       - PHYSICAL MODE
*                     'F'       - FHS MODE
*                                                  EDIT OPTIONS
      CHARACTER*6   EDITOPTIONSOUT
*                     'NOOPTS'  - NO OPTIONS
*                     'HCOPY'   - HARDCOPY
*                     'HOMOUT'  - MOMOGENEOUS OUTPUT
*                     'HEADER'  - SYSTEM HEADER
*                     'ETB'     - ETB CLOSED
*                     'INFO'    - INFORMATIVE MESSAGE
*                     'TRNCD'   - TRANS CODE
*                     'EXTEND'  - EXTENDED LINE OUTPUT
*                     'NLOGC'   - NO LOG CONTROL
*                     'BELL'    - BELL
*                     'OWRITE'  - OVERWRITE
*                     'BELHOM'  - BELL HOMOGENEOUS
*                     'BELEXT'  - BELL EXTEND
*                     'BELINF'  - BELL INFORMATIVE
*                     'BELNLC'  - BELL NO LOG CONTROL
*                     'HCNLC'   - HARDCOPY NO LOG CONTROL
*
*                                                EDITIN
      CHARACTER*16  EDITIN
*                                                  EDIT MODE
      CHARACTER*1   EDITMODEIN
*                     'L'       - LINE MODE
*                     'P'       - PHYSICAL MODE
*                                                  EDIT OPTIONS
```

```
      CHARACTER*6   EDITOPTIONSIN
*                     'NOOPTS'  - NO-OPTIONS
*                     'NOCORR'  - NO-CORRECTION
*                     'LCASE'   - LOWER-CASE
*                     'NOHDR'   - DELETE-DEVICE-HEADER
*                     'GETFC'   - GET-FUNCTION-CODE
*                     'CFDATA'  - CONFIDENTIAL-DATA
*                     'GETIC'   - GET-ID-CARD
*                     'EXTEND'  - EXTENDED-LINE-INPUT
*                     'LOWEXT'  - LOWER-CASE-EXTEND
*                     'CFDLOW'  - CONF-DATA-LOW-CASE
*                     'GFCLOW'  - FCT-CODE-LOW-CASE
*                     'GFCCFD'  - FCT-CODE-CONF-DATA
*                     'GFCCDL'  - FCT-CODE-CONF-DATA-LCASE
*                     'GFCEXT'  - FCT-CODE-EXTEND
*                     'GFCEXL'  - FCT-CODE-EXTEND-LCASE
*                     'NOHDRL'  - DELETE-DEVICE-HEADER-LCASE
*
*                                               READLENGTH
      INTEGER*4     READLENGTH
*                                               ISAM REQUEST
      CHARACTER*1   ISAMREQUEST
*                     'N'       - NO KEY
*                     'K'       - GET KEY
*                     'L'       - GET KEY LENGTH
*                     'P'       - GET KEY POSITION
*                     'B'       - GET KEY POSITION LENGTH
*
*                                               ASSIGNMENT REQUEST
      CHARACTER*1   ASSIGNREQUEST
*                     'N'       - NO ASSIGNMENT CODE
*                     'G'       - GET ASSIGNMENT CODE
*
*                                               COPYMEM ID
      INTEGER*2     COPYMEMID          / 1 /
*                                               TIAM RETURN INFO
      CHARACTER*16  TIAMRETURNINFO
*                                                 TIAM RC
      INTEGER*2     TIAMRC
*                                                 ASSIGNMENT
      INTEGER*2     ASSIGNMENT
*                                                 KEY POSITION
      INTEGER*2     KEYPOSITION
*                                                 KEY LENGTH
      INTEGER*2     KEYLENGTH
*
*****************************************************************
*
      CHARACTER*1   LINEMODE           / 'L' /
      CHARACTER*1   PHYISCALMODE       / 'P' /
      CHARACTER*1   FHSMODE            / 'F' /
*
      CHARACTER*6   NOOPTIONS          / 'NOOPTS' /
      CHARACTER*6   HARDCOPY           / 'HCOPY ' /
      CHARACTER*6   HOMOGENEOUSOUT     / 'HOMOUT' /
      CHARACTER*6   SYSTEMHEADER       / 'HEADER' /
      CHARACTER*6   ETBCLOSED          / 'ETB   ' /
      CHARACTER*6   INFORMATIVEMESS    / 'INFO  ' /
      CHARACTER*6   TRANSCODE          / 'TRNCD ' /
      CHARACTER*6   EXTENDEDLINEOUT    / 'EXTEND' /
```

```
      CHARACTER*6   NOLOGCONTROL       / 'NLOGC ' /
      CHARACTER*6   BELL               / 'BELL  ' /
      CHARACTER*6   OVERWRITE          / 'OWRITE' /
      CHARACTER*6   BELLHOMOGENEOUS    / 'BELHOM' /
      CHARACTER*6   BELLEXTEND         / 'BELEXT' /
      CHARACTER*6   BELLINFORMATIVE    / 'BELINF' /
      CHARACTER*6   BELLNOLOGCONTRL    / 'BELNLC' /
      CHARACTER*6   HARDCOPYNOLOGCT    / 'HCNLC ' /
*
      CHARACTER*6   NOCORRECTION       / 'NOCORR' /
      CHARACTER*6   LOWERCASE          / 'LCASE ' /
      CHARACTER*6   DELETEDEVICEHDR    / 'NOHDR ' /
      CHARACTER*6   GETFUNCTIONCODE    / 'GETFC ' /
      CHARACTER*6   CONFIDENTIALDTA    / 'CFDATA' /
      CHARACTER*6   GETIDCARD          / 'GETIC ' /
      CHARACTER*6   EXTENDEDLINEINP    / 'EXTEND' /
      CHARACTER*6   LOWERCASEEXTEND    / 'LOWEXT' /
      CHARACTER*6   CONFDATALOWCASE    / 'CFDLOW' /
      CHARACTER*6   FCTCODELOWCASE     / 'GFCLOW' /
      CHARACTER*6   FCTCODECONFDATA    / 'GFCCFD' /
      CHARACTER*6   FCTCODECONFDTLC    / 'GFCCDL' /
      CHARACTER*6   FCTCODEEXTEND      / 'GFCEXT' /
      CHARACTER*6   FCTCODEEXTENDLC    / 'GFCEXL' /
      CHARACTER*6   DELETEDEVHDRLC     / 'NOHDRL' /
*
      CHARACTER*1   NOKEY              / 'N' /
      CHARACTER*1   GETKEY             / 'K' /
      CHARACTER*1   GETKEYLENGTH       / 'L' /
      CHARACTER*1   GETKEYPOSITION     / 'P' /
      CHARACTER*1   GETKEYPOSLENGTH    / 'B' /
*
      CHARACTER*1   NOASSIGNCODE       / 'N' /
      CHARACTER*1   GETASSIGNCODE      / 'G' /
*
*****************************************************************
*
      EQUIVALENCE (TIAMCONTROLINFO (  1: 16), EDITOUT)
      EQUIVALENCE (EDITOUT         (  1:  1), EDITMODEOUT)
      EQUIVALENCE (EDITOUT         (  3:  8), EDITOPTIONSOUT)
      EQUIVALENCE (TIAMCONTROLINFO ( 17: 32), EDITIN)
      EQUIVALENCE (EDITIN          (  1:  1), EDITMODEIN)
      EQUIVALENCE (EDITIN          (  3:  8), EDITOPTIONSIN)
      EQUIVALENCE (TIAMCONTROLINFO ( 33: 36), READLENGTH)
      EQUIVALENCE (TIAMCONTROLINFO ( 37: 37), ISAMREQUEST)
      EQUIVALENCE (TIAMCONTROLINFO ( 41: 41), ASSIGNREQUEST)
      EQUIVALENCE (TIAMCONTROLINFO ( 45: 46), COPYMEMID)
      EQUIVALENCE (TIAMCONTROLINFO ( 49: 64), TIAMRETURNINFO)
      EQUIVALENCE (TIAMRETURNINFO  (  1:  2), TIAMRC)
      EQUIVALENCE (TIAMRETURNINFO  (  5:  6), ASSIGNMENT)
      EQUIVALENCE (TIAMRETURNINFO  (  9: 10), KEYPOSITION)
      EQUIVALENCE (TIAMRETURNINFO  ( 11: 12), KEYLENGTH)
*
*****************************************************************
```

5.2 TIAM FORTRAN calls

5.2.1 Read data from SYSDTA: CALL RDATA

This call is for reading a data record from SYSDTA. The file declared as SYSDTA may be a cataloged SAM or ISAM file, a floppy disk unit, a member of a PLAM library, an SDF-P variable or the terminal running the task.

Format of call:

```
EXTERNAL RDATA
CALL RDATA (TIAMCONTROLINFO, user-area [,VTSUCB])
```

Programming example RDATA

```
*****************************************************************
*                                                               *
* Example of R D A T A in FORTRAN programs.                     *
* You are prompted for an input, which is echoed on screen.     *
*                                                               *
*****************************************************************

      PROGRAM FORRDATA
*
      CHARACTER*24 IOAREA
      INTEGER*4    DTLNGT
      CHARACTER*20 TEXT
      EQUIVALENCE  (IOAREA  (1:4),  DTLNGT)
      EQUIVALENCE  (IOAREA  (5:24), TEXT)
      %INCLUDE $TSOS.SYSLIB.TIAM.112(FORINFO)
      %INCLUDE $TSOS.SYSLIB.VTSU-B.111(VTSUCBF)
*
      EXTERNAL RDATA
*
      READLENGTH = 24
      EDITOPTIONSIN = 'VTCBU'
      CALL RDATA(TIAMCONTROLINFO, IOAREA,VTSUCB)
      WRITE(*,*) 'TIAMRC     : ',TIAMRC
      WRITE(*,*) 'VTSU MRC   : ',MAINRETCODE
      WRITE(*,*) 'TEXTE      : ',TEXT
      WRITE(*,*) 'DATA LEN.  : ',DTLNGT
      END
```

5.2.2 Write data to SYSOUT: CALL WROUT

This call is for outputting a message to SYSOUT. The file declared as SYSOUT may be a cataloged SAM or ISAM file, a member of a PLAM library, an SDF-P variable or the terminal running the task.

As a special function integrated into CALL WROUT, format handling with FHS is supported; FHS is called internally by TIAM.

Format of call:

```
EXTERNAL WROUT
CALL WROUT (TIAMCONTROLINFO, user-area [,VTSUCB] [,FHSMAINPAR])
```

Programming example WROUT

```
*****************************************************************
* Example of W R O U T in FORTRAN programs.                     *
* A formatted screen (FHS mode), the TIAM and FHS return codes  *
* and the filled fields (Line mode) are output.                 *
*****************************************************************

      PROGRAM FORWROUT
*
      INTEGER * 4  DTLNGT
      %INCLUDE FORLIB(FWROUT)
      %INCLUDE FORLIB(FHSMAINP)
      %INCLUDE $TSOS.SYSLIB.TIAM.112(FORINFO)
*
      EXTERNAL WROUT
*
      FHSMAPNAME     = 'FWROUT  '
      FHSRESTARTOPT1 = 'Y'
      FHSMAPLIBOPT   = 'Y'
      FHSMAPLIBNAME  = 'FORLIB'
      EDITMODEOUT    = 'F'
      CALL WROUT(TIAMCONTROLINFO, IOAREA, FHSMAINPAR)
      WRITE(*,*) 'RCCATEGO   : ',FORRCCATEGO
      WRITE(*,*) 'RCREASON   : ',FORRCREASON
      WRITE(*,*) 'TIAMRC     : ',TIAMRC
      WRITE(*,*) 'FHSMAINRC  : ',FHSMAINRC
      WRITE(*,*) 'ERCATEGO   : ',ERRORCATEGORY
      WRITE(*,*) 'ERREASON   : ',ERRORREASON
      END
```

5.2.3 Terminal tandem write/read: CALL WRTRD

This call is for outputting a message to a data display station and immediately afterwards reading a message from the terminal. Apart from the message transmitted to the terminal, no other prompting character appears as an input request.
As a special function integrated into the call, format handling with the aid of FHS is supported; FHS is called internally by TIAM.

Format of call:

```
EXTERNAL WRTRD
CALL WRTRD (TIAMCONTROLINFO, user-area1, user-area2
                        [,VTSUCB] [,FHSMAINPAR])
```

Programming example WRTRD

```
*****************************************************************
* Example of W R T R D in FORTRAN programs                      *
* Initially, a formatted screen (FHS mode) is output,           *
* after which the terminal requests an input. This is followed  *
* by output of the TIAM and FHS return codes and the            *
* filled fields (Line mode)-                                    *
*****************************************************************

      PROGRAM FORWRTRD
*
      INTEGER * 4  DTLNGT
      %INCLUDE FORLIB(FWRTRD)
      %INCLUDE FORLIB(FHSMAINP)
      %INCLUDE $TSOS.SYSLIB.TIAM.112(FORINFO)
*
      EXTERNAL WRTRD
*
      FHSMAPNAME     = 'FWRTRD  '
      FHSRESTARTOPT1 = 'Y'
      FHSMAPLIBOPT   = 'Y'
      FHSMAPLIBNAME  = 'FORLIB'
      EDITMODEOUT    = 'F'
      CALL WRTRD(TIAMCONTROLINFO, IOAREA, IOAREA, FHSMAINPAR)
      WRITE(*,*) 'RCCATEGO   : ',FORRCCATEGO
      WRITE(*,*) 'RCREASON   : ',FORRCREASON
      WRITE(*,*) 'TIAMRC     : ',TIAMRC
      WRITE(*,*) 'FHSMAINRC  : ',FHSMAINRC
      WRITE(*,*) 'ERCATEGO   : ',ERRORCATEGORY
      WRITE(*,*) 'ERREASON   : ',ERRORREASON
      WRITE(*,*) 'FIELD      : ',FIELD
      END
```

6 TIAM PL/I interface

The definitions of the PL/I data structure and the PL/I data fields correspond to the definitions of the COBOL data structure and the COBOL data fields. Consequently, this chapter deals only with those aspects of the PL/I language and syntax relevant to these data structures. See the chapter on the COBOL interface for detailed information on the data structures and data fields.

Note

The TIAM interface for PL/I is available as of TIAM V11.0A.

The FHS interface for PL/I is available as of FHS V7.0

The VTSU interface for PL/I is available as of VTSU V10.1A.

6.1 Data structures for TIAM PL/I calls

TIAM and the application program can communicate by way of data structures which are contained in a library as INCLUDE members and are copied into the application program.

The VTSU-CONTROL-BLOCK data structure

The VTSU-CONTROL-BLOCK data structure is copied into the application program by means of the following %INCLUDE statement.

```
%INCLUDE $TSOS.SYSLIB.VTSU-B.111(VTSUCBP)
```

The VTSU-CONTROL-BLOCK is described in the VTSU manual (see [1] in the Relad publications chapter).

The LINE-MODE-CONTROL-CHARACTER data structure

The LINE-MODE-CONTROL-CHARACTER is copied into the application program by means of the following %INCLUDE statement.

```
%INCLUDE $TSOS.SYSLIB.VTSU-B.111(PL1CTRC)
```

Full details of the individual parameters will be found in the VTSU manual (see [1] in the Related publications chapter).

Data structure TIAM-CONTROL-INFO

The data structure TIAM-CONTROL-INFO is copied to the application program by means of the following %INCLUDE statement:

```
%INCLUDE $TSOS.SYSLIB.TIAM.111(PL1INFO)
```

The structure of TIAM-CONTROL-INFO is shown below. For detailed information on the individual parameters, see page 133 ff.

```
/*********************************************************************/
/*      PL1INFO          001       920320           TIAM          U */
/*********************************************************************/
/*                                                                   */
/*     COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1991     */
/*                   ALL RIGHTS RESERVED                             */
/*                                                                   */
/*********************************************************************/
/*                                                                   */
/* **********************   Edit mode      *************************/
/* Possible values for EDIT_OUT.EDIT_MODE / EDIT_IN.EDIT_MODE        */
DCL LINE_MODE               CHAR (1) STATIC INIT ('L');
DCL PHYSICAL_MODE           CHAR (1) STATIC INIT ('P');

/* Possible values for EDIT_OUT.EDIT_MODE                            */
DCL FHS_MODE                CHAR (1) STATIC INIT ('F');

/* **********************   Edit options   *************************/
/* Possible values for EDIT_OUT.EDIT_OPTIONS / EDIT_IN.EDIT_OPTIONS  */
DCL NO_OPTIONS                   CHAR (6) STATIC INIT ('NOOPTS');
DCL VTSUCB_USED                  CHAR (6) STATIC INIT ('VTCBU ');

/* Possible values for EDIT_OUT.EDIT_OPTIONS                         */
DCL HARDCOPY                     CHAR (6) STATIC INIT ('HCOPY ');
DCL HOMOGENEOUS_OUTPUT           CHAR (6) STATIC INIT ('HOMOUT');
DCL SYSTEM_HEADER                CHAR (6) STATIC INIT ('HEADER');
DCL ETB_CLOSED                   CHAR (6) STATIC INIT ('ETB   ');
DCL INFORMATIVE_MESSAGE          CHAR (6) STATIC INIT ('INFO  ');
DCL TRANS_CODE                   CHAR (6) STATIC INIT ('TRNCD ');
DCL EXTENDED_LINE_OUTPUT         CHAR (6) STATIC INIT ('EXTEND');
DCL NO_LOG_CONTROL               CHAR (6) STATIC INIT ('NLOGC ');
DCL BELL                         CHAR (6) STATIC INIT ('BELL  ');
DCL OVERWRITE                    CHAR (6) STATIC INIT ('OWRITE');
DCL BELL_HOMOGENEOUS             CHAR (6) STATIC INIT ('BELHOM');
DCL BELL_EXTEND                  CHAR (6) STATIC INIT ('BELEXT');
DCL BELL_INFORMATIVE             CHAR (6) STATIC INIT ('BELINF');
DCL BELL_NO_LOG_CONTROL          CHAR (6) STATIC INIT ('BELNLC');
DCL HARDCOPY_NO_LOG_CONTROL      CHAR (6) STATIC INIT ('HCNLC ');

/* Possible values for EDIT_IN.EDIT_OPTIONS                          */
DCL NO_CORRECTION                CHAR (6) STATIC INIT ('NOCORR');
DCL LOWER_CASE                   CHAR (6) STATIC INIT ('LCASE ');
DCL DELETE_DEVICE_HEADER         CHAR (6) STATIC INIT ('NOHDR ');
DCL GET_FUNCTION_CODE            CHAR (6) STATIC INIT ('GETFC ');
DCL CONFIDENTIAL_DATA            CHAR (6) STATIC INIT ('CFDATA');
DCL GET_ID_CARD                  CHAR (6) STATIC INIT ('GETIC ');
DCL EXTENDED_LINE_INPUT          CHAR (6) STATIC INIT ('EXTEND');
DCL LOWER_CASE_EXTEND            CHAR (6) STATIC INIT ('LOWEXT');
DCL CONF_DATA_LOW_CASE           CHAR (6) STATIC INIT ('CFDLOW');
DCL FCT_CODE_LOW_CASE            CHAR (6) STATIC INIT ('GFCLOW');
DCL FCT_CODE_CONF_DATA           CHAR (6) STATIC INIT ('GFCCFD');
DCL FCT_CODE_CONF_DATA_LCASE     CHAR (6) STATIC INIT ('GFCCDL');
DCL FCT_CODE_EXTEND              CHAR (6) STATIC INIT ('GFCEXT');
```

```
DCL FCT_CODE_EXTEND_LCASE         CHAR (6) STATIC INIT ('GFCEXL');
DCL DELETE_DEVICE_HEADER_LCASE    CHAR (6) STATIC INIT ('NOHDRL');

/* ***********************   ISAM informations  **********************/
/* Possible values for ISAM_REQUEST                                   */
DCL NO_KEY                        CHAR (1) STATIC INIT ('N');
DCL GET_KEY                       CHAR (1) STATIC INIT ('K');
DCL GET_KEY_LENGTH                CHAR (1) STATIC INIT ('L');
DCL GET_KEY_POSITION              CHAR (1) STATIC INIT ('P');
DCL GET_KEY_POSITION_LENGTH       CHAR (1) STATIC INIT ('B');

/* ***********************   ASSIGNMENT informations   ***************/
/* ASSIGNMENT informations                                            */
/* Possible values for ASSIGNMENT_REQUEST                             */
DCL NO_ASSIGNMENT_CODE            CHAR (1) STATIC INIT ('N');
DCL GET_ASSIGNMENT_CODE           CHAR (1) STATIC INIT ('G');

/* ***********************    Call parameter list   ******************/
DECLARE
01  TIAM_CONTROL_INFO,
    02  EDIT_OUT,
        03 EDIT_MODE                 CHAR,
        03 FIL1                      CHAR,
        03 EDIT_OPTIONS              CHAR(6),
        03 FIL8                      CHAR(8),
    02  EDIT_IN,
        03 EDIT_MODE                 CHAR,
        03 FIL1                      CHAR,
      03 EDIT_OPTIONS               CHAR(6),
      03 FIL8                       CHAR(8),
    02 READLENGTH                    BIN FIXED(31) ALIGNED,
    02 ISAM_REQUEST                  CHAR,
    02 FIL3                          CHAR(3),
    02 ASSIGNMENT_REQUEST            CHAR,
    02 FILL3                         CHAR(3),
    02 COPYMEM_ID                    BIN FIXED(15) ALIGNED INIT(1),
    02 FIL2                          CHAR(2),
    02 TIAM_RETURN_INFO,
        03 TIAM_RC                   BIN FIXED(15) ALIGNED INIT(0),
        03 FIL2                      CHAR(2),
        03 ASSIGNMENT                BIN FIXED(15) ALIGNED,
        03 FILL2                     CHAR(2),
        03 KEY_POSITION              BIN FIXED(15) ALIGNED,
        03 KEY_LENGTH                BIN FIXED(15) ALIGNED,
        03 FIL4                      CHAR(4);
```

6.2 TIAM PL/I calls

6.2.1 Read data from SYSDTA: CALL RDATA

This call is for reading a data record from SYSDTA. The file declared as SYSDTA may be a cataloged SAM or ISAM file, a floppy disk unit, a member of a PLAM library, an SDF-P variable or the terminal running the task.

Format of call:

```
DCL RDATA ENTRY EXTERNAL OPTIONS (ASSEMBLER);
CALL RDATA (TIAM_CONTROL_INFO, user-area [,VTSUCB]);
```

Programming example RDATA

```
/*****************************************************************/
/* Example of R D A T A in PL/I programs.                        */
/* You are prompted for an input, which is echoed on screen.     */
/*****************************************************************/
PL1RDATA: PROC OPTIONS(MAIN);

  %INCLUDE $TSOS.SYSLIB.TIAM.112(PL1INFO)
  %INCLUDE $TSOS.SYSLIB.VTSU-B.111(VTSUCBP)

  DCL RDATA ENTRY EXTERNAL OPTIONS (ASSEMBLER);
  DCL    1 IOAREA,
        19 DTLNGT   BIN FIXED(31),
        19 TEXT     CHAR (20);

      IOAREA.TEXT                          = ' ';
      TIAM_CONTROL_INFO.READLENGTH         = 25;
      TIAM_CONTROL_INFO.EDIT_IN.EDIT_OPTIONS  = VTSUCB_USED;
      CALL RDATA(TIAM_CONTROL_INFO,IOAREA,VTSUCB);
      DISPLAY ('TIAMRC     : ' || CHAR(TIAM_RC));
      DISPLAY ('VTSUCB MRC : ' || CHAR(MAINCODE));
      DISPLAY ('VTSUCB SRC : ' || CHAR(SUBCODES));
      DISPLAY ('TEXT LEN.  : ' || CHAR(DTLNGT));
      DISPLAY ('TEXT       : ' || TEXT);
  END;
```

6.2.2 Write data to SYSOUT: CALL WROUT

This call is for outputting a message to SYSOUT. The file declared as SYSOUT may be a cataloged SAM or ISAM file, a member of a PLAM library, an SDF-P variable or the terminal running the task.
As a special function integrated into CALL "WROUT", format handling with FHS is supported; FHS is called internally by TIAM.

Format of call:

```
DCL WROUT ENTRY EXTERNAL OPTIONS (ASSEMBLER);
CALL WROUT (TIAM_CONTROL_INFO, user-area [,VTSUCB][,FHSMAINPAR]);
```

Programming example WROUT

```
/*****************************************************************/
/* Example of  W R O U T in PL/I programs.                       */
/* A formatted screen (FHS mode), the TIAM and FHS return codes  */
/* and the filled fields (Line mode) are output.                 */
/*****************************************************************/
PL1WROUT: PROC OPTIONS(MAIN);

  DCL WROUT ENTRY EXTERNAL OPTIONS (ASSEMBLER);
  DCL    1 IOAREA,
        19 DTLNGT   BIN FIXED(31),
        %INCLUDE PL1LIB(PWROUT);
  DCL   IOPR BIT(528) DEFINED IOAREA;
        %INCLUDE PL1LIB(FHSMAINP);
        %INCLUDE $TSOS.SYSLIB.TIAM.112(PL1INFO);

      IOPR               = (528)'0'B;
      FHS_MAP_NAME       = 'PWROUT';
      FHS_RESTART_OPT1   = 'Y';
      FHS_MAP_LIB_OPT    = 'Y';
      FHS_MAP_LIB_NAME   = 'PL1LIB';
      EDIT_OUT.EDIT_MODE = 'F';
      CALL WROUT(TIAM_CONTROL_INFO, IOAREA, FHS_MAIN_PAR);
      DISPLAY ('TIAMRC     : ' || CHAR(TIAM_RC));
      DISPLAY ('FHSMAINRC  : ' || CHAR(FHS_MAIN_RC));
      DISPLAY ('ERCATEGO   : ' || CHAR(ERROR_CATEGORY));
      DISPLAY ('ERREASON   : ' || CHAR(ERROR_REASON));
      DISPLAY ('FIELD      : ' || FIELD);
  END;
```

6.2.3 Terminal tandem write/read: CALL WRTRD

This call is for outputting a message to a data display terminal and immediately afterwards reading a message from the terminal. Apart from the message transmitted to the terminal, no other prompting character appears as an input request.
As a special function integrated into the call, format handling with the aid of FHS is supported; FHS is called internally by TIAM.

Format of call:

```
DCL WRTRD ENTRY EXTERNAL OPTIONS (ASSEMBLER);
CALL WRTRD (TIAM_CONTROL_INFO, user-area1, user-area2
                        [,VTSUCB] [,FHS-MAIN-PAR]);
```

Programming example WRTRD

```
/*****************************************************************/
/* Example of W R T R D in PL/I programs.                        */
/* Initially, a formatted screen (FHS mode) is output,           */
/* after which the terminal requests an input. This is followed  */
/* by output of the TIAM and FHS return codes and the            */
/* filled fields (Line mode).                                    */
/*****************************************************************/
PL1WRTRD: PROC OPTIONS(MAIN);

  DCL WRTRD ENTRY EXTERNAL OPTIONS (ASSEMBLER);
  DCL    1 IOAREA,
        19 DTLNGT   BIN FIXED(31),
        %INCLUDE PL1LIB(PWRTRD);
  DCL   IOPR BIT(528) DEFINED IOAREA;
        %INCLUDE PL1LIB(FHSMAINP);
        %INCLUDE $TSOS.SYSLIB.TIAM.112(PL1INFO);

      IOPR               = (528)'0'B;
      FHS_MAP_NAME       = 'PWRTRD';
      FHS_RESTART_OPT1   = 'Y';
      FHS_MAP_LIB_OPT    = 'Y';
      FHS_MAP_LIB_NAME   = 'PL1LIB';
      EDIT_OUT.EDIT_MODE = 'F';
      CALL WRTRD(TIAM_CONTROL_INFO, IOAREA, IOAREA, FHS_MAIN_PAR);
      DISPLAY ('TIAMRC     : ' || CHAR(TIAM_RC));
      DISPLAY ('FHSMAINRC  : ' || CHAR(FHS_MAIN_RC));
      DISPLAY ('ERCATEGO   : ' || CHAR(ERROR_CATEGORY));
      DISPLAY ('ERREASON   : ' || CHAR(ERROR_REASON));
      DISPLAY ('FIELD      : ' || FIELD);
  END;
```

7 TIAM C interface

The definitions of the C data structure and the C data fields correspond to the definitions of the COBOL data structure and the COBOL data fields. Consequently, this chapter deals only with those aspects of the C language and syntax relevant to these data structures. See the chapter on the COBOL interface for detailed information on the data structures and data fields.

Notes

– When the input and output interfaces are being used, the following points should be taken into account:

  1. If the version of the C compiler which is being used does not support the ILCS conventions, a small ASSEMBLER interface module will be required to generate the parameter lists in the format supported by TIAM (see the example below). This ASSEMBLER interface sets the most significant bit of the last parameter in the parameter list to binary 1.

     Example of an ASSEMBLER module for a WROUT with VTSUCB.

     a) C program

```
extern cwrout3 ();
cwrout3 (&tiam_control_info, &output_area, &vtsu_cb);
```

     b) ASSEMBLER module

```
AWROUT   START
AWROUT   AMODE ANY
AWROUT   RMODE ANY
CWROUT3  C$ENT ((TIAMCI,A),(DTA,A),(VTSUA,A))
         NI    TIAMCI,X'7F'  TIAM-CONTROL-INFO address
         NI    DTA,X'7F'     DATA address
         OI    VTSUA,X'80'   VTSUCB address
         L     R15,WROUT
         BASR  R14,R15
         C$EX
WROUT    DC    V(WROUT)
         END
```

2. If the version of the C compiler which is being used supports the ILCS conventions, the ASSEMBLER module described above does not have to be used. Direct calls can be passed to TIAM, as shown in the examples in the section entitled “TIAM C calls” (see page 188 ff).

– The TIAM interface for C is available as of TIAM V11.0A.

– The VTSUB and TIAMCTRC interfaces for C is available as of VTSU V10.1A.

– There is currently no FHS interface for C.

7.1 Data structures for TIAM C calls

TIAM and the application program can communicate by way of data structures which are contained in a library as INCLUDE members and are copied into the application program.

The VTSU-CONTROL-BLOCK data structure

The VTSU-CONTROL-BLOCK data structure is copied into the application program by means of the following call.

```
#include <vtsucb.h>
```

This Include member is held in the $TSOS.SYSLIB.VTSU-B.111 library.

The VTSU-CONTROL-BLOCK is described in the VTSU manual (see [1] in the Related publications chapter).

The LINE-MODE-CONTROL-CHARACTER data structure

The LINE-MODE-CONTROL-CHARACTER is copied into the application program by means of the following call.

```
#include <cctrc.h>
```

This Include member is held in the $TSOS.SYSLIB.VTSU-B.111 library.

Full details of the individual parameters will be found in the VTSU manual (see [1] in the Related publications chapter).

Data structure TIAM-CONTROL-INFO

The data structure TIAM-CONTROL-INFO is copied to the application program by means of the following call:

```
#include <cinfo.h>
```

This INCLUDE member is in the library $TSOS.SYSLIB.TIAM.112.

The structure of TIAM-CONTROL-INFO is shown below. For detailed information on the individual parameters, see page 133 ff.

```
/*****************************************************************************/
/*  TIAMINFO.H      211         950320              TIAM           U */
/*****************************************************************************/
/*                                                                           */
/*      COPYRIGHT (C) SIEMENS NIXDORF INFORMATIONSSYSTEME AG 1991     */
/*                    ALL RIGHTS RESERVED                                    */
/*                                                                           */
/*****************************************************************************/

#ifdef WAS_CINFO
#else
#define WAS_CINFO
/************************** TIAM RC values   **************************/
/*  Possible values for TIAM_RETURN_INFO.TIAM_RC                         */
#define TIAM_NO_ERROR                    0
#define TIAM_IO_ERROR                    4
#define TIAM_PARAMETER_ERROR             8
#define TIAM_RECORD_TRUNC_READ          12
#define TIAM_RECORD_TRUNC_WRITE         16
#define TIAM_WRTRD_CALLED_BATCH         20
#define TIAM_END_OF_INPUT               24
#define TIAM_PARAMETER_CORRECTED        32
#define TIAM_FHS_MISSING                36
#define TIAM_FHS_ERROR                  40
#define TIAM_NEWLINE_DETECTED           44
#define TIAM_VTSU_ERROR                 48

/**************************   Edit mode      **************************/
/*  Possible values for EDIT_OUT.EDIT_MODE / EDIT_IN.EDIT_MODE        */
#define LINE_MODE                       'L'
#define PHYSICAL_MODE                   'P'

/*  Possible values for EDIT_OUT.EDIT_MODE                             */
#define FHS_MODE                        'F'

/**************************   Edit options   **************************/
/* The edit options must be moved in corresponding parameter fields   */
/* (edit_in.edit_options, edit_out.edit_options) thanks to STRNCPY    */
/* function.                                                          */
/* EX: strncpy (edit_in.edit_options, "LCASE ",                       */
/*              sizeof (edit_in.edit_options));                       */

/*  Possible values for EDIT_OUT.EDIT_OPTIONS / EDIT_IN.EDIT_OPTIONS  */
/*  NO EDIT OPTIONS SPECIFIED           "NOOPTS"
    VTSUCB_USED                         "VTCBU "                      */
```

```
/*  Possible values for EDIT_OUT.EDIT_OPTIONS
    HARDCOPY                            "HCOPY "
    HOMOGENEOUS_OUTPUT                  "HOMOUT"
    SYSTEM_HEADER                       "HEADER"
    ETB_CLOSED                          "ETB   "
    INFORMATIVE_MESSAGE                 "INFO  "
    TRANS_CODE                          "TRNCD "
    EXTENDED_LINE_OUTPUT                "EXTEND"
    NO_LOG_CONTROL                      "NLOGC "
    BELL                                "BELL  "
    OVERWRITE                           "OWRITE"
    BELL_HOMOGENEOUS                    "BELHOM"
    BELL_EXTEND                         "BELEXT"
    BELL_INFORMATIVE                    "BELINF"
    BELL_NO_LOG_CONTROL                 "BELNLC"
    HARDCOPY_NO_LOG_CONTROL             "HCNLC "                    */

/*  Possible values for EDIT_IN.EDIT_OPTIONS
    NO_CORRECTION                       "NOCORR"
    LOWER_CASE                          "LCASE "
    DELETE_DEVICE_HEADER                "NOHDR "
    GET_FUNCTION_CODE                   "GETFC "
    CONFIDENTIAL_DATA                   "CFDATA"
    GET_ID_CARD                         "GETIC "
    EXTENDED_LINE_INPUT                 "EXTEND"
    LOWER_CASE_EXTEND                   "LOWEXT"
    CONF_DATA_LOW_CASE                  "CFDLOW"
    FCT_CODE_LOW_CASE                   "GFCLOW"
    FCT_CODE_CONF_DATA                  "GFCCFD"
    FCT_CODE_CONF_DATA_LCASE            "GFCCDL"
    FCT_CODE_EXTEND                     "GFCEXT"
    FCT_CODE_EXTEND_LCASE               "GFCEXL"
    DELETE_DEVICE_HEADER_LCASE          "NOHDRL"                    */

/*************************   ISAM informations  **********************/
/*  Possible values for ISAM_REQUEST                                  */
#define NO_KEY                          'N'
#define GET_KEY                         'K'
#define GET_KEY_LENGTH                  'L'
#define GET_KEY_POSITION                'P'
#define GET_KEY_POSITION_LENGTH         'B'

/*************************   ASSIGNMENT informations   ***************/
/*  ASSIGNMENT informations                                           */
/*  Possible values for ASSIGNMENT_REQUEST                            */
#define NO_ASSIGNMENT_CODE              'N'
#define GET_ASSIGNMENT_CODE             'G'

/*************************    Call parameter list   ******************/
typedef struct {
  char         edit_mode;
  char         filler_1;
  char         edit_options[6];
  char         filler_2[8];
} edit_out_type;

typedef struct {
  char         edit_mode;
  char         filler_1;
  char         edit_options[6];
```

```
  char         filler_2[8];
} edit_in_type;

typedef struct {
  short        tiam_rc;
  char         filler_1[2];
  short        assignment;
  char         filler_2[2];
  short        key_position;
  short        key_length;
  char         filler_3[4];
} tiam_return_info_type;

typedef struct {
  edit_out_type         edit_out;
  edit_in_type          edit_in;
  unsigned              readlength;
  char                  isam_request;
  char                  filler_1[3];
  char                  assignment_request;
  char                  filler_2[3];
  short                 copymem_id;
  char                  filler_3[2];
  tiam_return_info_type tiam_return_info;
} tiam_control_info_type;
#endif
```

7.2 TIAM C calls

7.2.1 Read data from SYSDTA

This call is for reading a data record from SYSDTA. The file declared as SYSDTA may be a cataloged SAM or ISAM file, a floppy disk unit, a member of a PLAM library, an SDF-P variable or the terminal running the task.

Format of call:

```
extern RDATA(tiam-control-info_type *ptr1,
             struct input_area *ptr2,
             vtsucb_type *ptr3);
```

Programming example RDATA

```
*********************************************************************
* Example of R D A T A in C programs.                               *
* You are prompted for an input, which is echoed on screen.         *
* The TIAM returncodes are then displayed.                          *
*********************************************************************
#include <stdio.h>
#include <cinfo.h>
#include <vtsucb.h>
#include <cctrc.h>
#include <string.h>

struct input_area {
   long     dtlngt;
   char     text[20];
} input_area;
tiam_control_info_type tiam_control_info;
vtsucb_type vtsucb1 = VTSUCB_DEFAULT;

extern RDATA(tiam_control_info_type *ptr1,
             struct input_area *ptr2,
             vtsucb_type *ptr3);
/**/
int main () {
   printf ("Enter your text (max. 20 characters)\n");
   strncpy (tiam_control_info.edit_in.edit_options, "VTCBU ",
            sizeof (tiam_control_info.edit_in.edit_options));
   tiam_control_info.readlength = sizeof(input_area);
   tiam_control_info.copymen_id = 1;
   RDATA(&tiam_control_info, &input_area, &vtsucbl);
   printf ("TIAM RC : %d\n",
           tiam_control_info.tiam_return_info.tiam_rc);
   printf ("VTSU MRC : %d\n",
           vtsucb1.vtsucb_std_hdr.std_vtsucb_rc.maincode);
   printf ("length of your text: %ld\n", input_area.dtlngt);
   printf ("text: %s\n", input_area.text);
   return(0);
}
```

7.2.2 Write data to SYSOUT

This call is for outputting a message to SYSOUT. The file declared as SYSOUT may be a cataloged SAM or ISAM file, a member of a PLAM library, an SDF-P variable or the terminal running the task.

Format of call:

```
extern WROUT(tiam_control_info_type *ptr1,
       out_area *ptr2);
```

Programming example WROUT

```
*****************************************************************
* Example of W R O U T in C programs.                           *
* A message is output, followed by the TIAM return code.        *
*****************************************************************
#include <stdio.h>
#include <cinfo.h>
#include <string.h>

typedef struct {
   long     dtlngt;
   char     pcc;
   char     text[22];
} out_area;
tiam_control_info_type tiam_control_info;
out_area output_area;

extern WROUT(tiam_control_info_type *ptr1,
       out_area *ptr2);
/**/
int main () {
   tiam_control_info.edit_out.edit_mode = LINE_MODE;
   tiam_control_info.copymen_id = 1;
   strncpy (tiam_control_info.edit_out.edit_options, "BELL  ",
            sizeof (tiam_control_info.edit_out.edit_options));
   output_area.dtlngt = sizeof(output_area.dtlngt) +
                        sizeof(output_area.pcc)    +
                        sizeof(output_area.text);
   strncpy (output_area.text, "test of the WROUT call", 22);

   WROUT(&tiam_control_info, &output_area);
   printf ("TIAM RC : %d\n",
           tiam_control_info.tiam_return_info.tiam_rc);
   return(0);
}
```

7.2.3 Terminal tandem write/read

This call is for outputting a message to a data display terminal and immediately afterwards reading a message from the terminal. Apart from the message transmitted to the terminal, no other prompting character appears as an input request.

Format of call:

```
extern WRTRD (tiam_control_info_type *,
              out_area *,
              in_area *);
```

Programming example WRTRD

```
*****************************************************************
* Example of W R T R D in C programs.                           *
*****************************************************************
#include <stdio.h>
#include <cinfo.h>
#include <string.h>

typedef struct {
   long     dtlngt;
   char     pcc;
   char     text[24];
} out_area;
typedef struct {
   long     dtlngt;
   char     text[24];
} in_area;
tiam_control_info_type tiam_control_info;
out_area output_area;
in_area input_area;

extern WRTRD (tiam_control_info_type *,
              out_area *,
              in_area *);


int main ()
{
   tiam_control_info.edit_in.edit_mode  = LINE_MODE;
   tiam_control_info.edit_out.edit_mode = LINE_MODE;
   tiam_control_info.copymen_id = 1;

   output_area.dtlngt = sizeof(output_area.dtlngt) +
                        sizeof(output_area.pcc)    +
                        sizeof(output_area.text);
   strncpy (tiam_control_info.edit_out.edit_options, "BELL  ",
            sizeof (tiam_control_info.edit_out.edit_options));
   strncpy (output_area.text, "please enter your text  ", 24);

   tiam_control_info.readlength = sizeof(input_area);
   strncpy (tiam_control_info.edit_in.edit_options, "LCASE",
            sizeof (tiam_control_info.edit_in.edit_options));

   WRTRD(&tiam_control_info, &output_area, &input_area);
```

```
    printf ("TIAM RC : %d\n",
            tiam_control_info.tiam_return_info.tiam_rc);
    printf ("text: %s\n", input_area.text);
    return(0):
}
```

8 POSIX interface

As from TIAM V11.2A, the POSIX I/O interfaces are also supported. However, this only applies to NDS data terminals (such as the 9750, 9755, 9763...), not for 3270 terminals or those connected via X29. Essentially, the POSIX I/O interfaces consist of the POSIX functions `read()` and `write()`.

This chapter describes those features of the POSIX I/O interfaces which are most important from the TIAM point of view. A full description of the POSIX interfaces will be found in the "POSIX Basics" [16] and "POSIX Interfaces" [17] manuals.

This chapter also describes special features and limitations of the BS2000 POSIX I/O interfaces.

8.1 Parallel I/O operations under BS2000 and POSIX

A BS2000 dialog process can handle the TIAM I/O requests (RDATA, WROUT and WRTRD) and the POSIX I/O requests (`read()` and `write()`) in any sequence at the same BS2000 terminal. This applies to a task called by a LOGON and to all the associated tasks created by the POSIX `fork()` function.

It is the responsibility of the application to synchronize the I/Os for the various tasks which belong to a terminal. The system applies a FIFO discipline in servicing the I/O requests from POSIX processes. The first POSIX process which identifies the need for an input is given the first user input. The decision as to which interfaces are used, i.e. the POSIX interfaces (`write()` and `read()`) or the BS2000 interfaces (RDATA, WROUT and WRTRD) is also the responsibility of the application.

8.1.1 Limitations

– There are some limitation on the use of BS2000 interfaces within tasks created by a `fork()`, when the system files are not assigned to the terminal (see the description of the SYSFILE environment for tasks created by a `fork()`, on page 196).

– If the LOGON task swaps into the system mode, then I/Os from tasks which belong to it and were created by a `fork()` will be deferred if there is still a program loaded for the LOGON task. This will be the case, for example, if a LOGON task is interrupted with the K2 key. The I/Os for the tasks created by a `fork()` will not be processed any further until the LOGON task returns to program mode. A POSIX shell cannot be interrupted by K2.

– I/Os for tasks created by a `fork()` will be rejected if the LOGON task has reached the end of its program. (See returncode X‘38‘ for RDATA, WROUT and WRTRD). I/Os requested by a task created by a `fork()` will also be rejected with returncode X‘38‘ if the task concerned was not created within the program which is being executed.

8.1.2 Handling of special characters

The functionality of the K2 (return to system mode) key can only be used to interrupt the LOGON task; only this task can read from SYSCMD.

The POSIX control characters (INTR, QUIT) are supported and can be invoked by the P key functions. The POSIX application user is responsible for assigning the correct values to the P keys. A description of the POSIX control characters and the corresponding values will be found in the "POSIX Commands" manual [15]).

8.1.3 Comparison of BS2000 I/O and POSIX I/O

BS2000 I/O	**POSIX I/O**
Only one dialog task per active terminal	Several POSIX processes (BS2000 tasks) on one terminal
Input/output is always synchronous – Half-duplex mode – Reads are always blocked (the task effecting the I/O always waits for user input)	Asynchronous I/O – Full-duplex mode – Non-blocking reads are possible (a returncode is provided if the data items are not immediately available)
Dialog task is created during the BCAM CONR processing	POSIX process is created by a `fork()` operation
Support for the F and K keys.	Support for special characters (QUIT, INTR)
Line-oriented reads (line mode)	Reads a specified number of characters (see the definition of the `read()` interface)
Three functions: RDATA, WROUT and WRTRD	Two functions: `read()` and `write()`
Input request characters for RDATA (* for TU, / for TPR)	No input request character for READ

8.2 TCHNG and TSTAT in conjunction with POSIX

Terminal attributes which are modified within a TU program by means of the TCHNG macro remain valid until there is a return to system mode. They are applicable for all tasks which are called in the program. It makes no difference which task the TCHNG macro was called from, i.e. whether the task was created by LOGON or by means of the POSIX `fork()` operation.

For example, if a task which was created by a `fork()` operation modifies a terminal attribute, this change continues to apply for the task created by the LOGON and for other tasks created using `fork()`. The change remains in effect until there is a return to the system mode, even if the task which called TCHNG is terminated.

8.3 The SYSFILE environment for tasks created by fork()

Initially, the SYSFILE environment for tasks created by a `fork()` is defined as follows:

System file	Macro	Primary assignment in dialog mode	Primary assignment in batch mode
SYSOUT	WROUT	Terminal	Temporary Spoolout file S.OUT.
SYSLST	WRLST	Temporary Spoolout file S.LST...	Temporary Spoolout file S.LST...
SYSLSTn	WRLST	SYSLST	SYSLST
SYSDTA	RDATA	Terminal	*DUMMY
SYSCMD	Priv.	*NONE	*NONE
	WRTRD	Terminal	Rejected (returncode)
SYSOPT	WRTOT	Temporary Spoolout file S.OPT...	Temporary Spoolout file S.OPT...
SYSIPT	RDCRD	*DUMMY	*DUMMY

The LOGON task is given a standard dialog SYSFILE environment; each task created by a `fork()` has its own SYSFILE environment, There is no transfer of the assigned system file from the parent process during a `fork()` operation.

Even for tasks created by a `fork()`, the primary assignments of the SYSDTA, SYSOUT and SYSLST system files can be modified. Note that for such an operation it may be necessary to activate SDF.

Only the LOGON task may read from SYSCMD. If a task created by a `fork()` attempts to execute a command (e.g. the CMD macro with /CALL-PROCEDURE) which requires access to SYSCMD, it will be rejected.

Unlike I/O operations under BS2000, POSIX I/O operations are not logged in SYSOUT/SYSLST.

9 Appendix

9.1 Table of standardized function key codes

Code									
normal	**with delete[1])**	**Meaning**	**8110**	**8151**	**8152**	**816x**	**9749 975x**	**9763**	**3270 [3])**
00	10	Data transmission (normal)	DÜ	DÜZ, DÜM	DÜZ, DÜM, DÜB	DÜ, DÜ1, DÜ2	DÜ, DÜ1, DÜ2	SEND	ENTER
01	11	Short code 1		FT1	F1	K1	K1	K1	PA 1
02	12	" 2		FT2 [2])	F2 [2])	K2 [2])	K2 [2])	K2 [2])	PA 2 [2])
03	13	" 3		DVA	F3	K3	K3	K3	PA 3 PF6
04	14	" 4			F1+FZ	K4	K4	K4	PF7
05	15	" 5			F2+FZ [2])	K5	K5	K5	PF8
06	16	" 6			F3+FZ	K6	K6	K6	PF9
07	17	" 7				K7	K7	K7	PF10
08	18	" 8				K8	K8	K8	PF11
09	19	" 9				K9	K9	K9	PF12
0A	1A	" 10				K10	K10	K10	PF13
0B	1B	" 11				K11	K11	K11	PF14
0C	1C	" 12				K12	K12	K12	PF15
0D	1D	" 13				K13	K13	K13	PF16
0E	1E	" 14				K14	K14	K14	PF17
20	30	Data transmission (marked) 0			DÜZ, DÜM, DÜB +FZ		DÜZ, DÜM, DÜB, +FZ		
21	31	" 1				F1	F1	F1	PF 1
22	32	" 2				F2	F2	F2	PF 2
23	33	" 3				F3	F3	F3	PF 3
24	34	" 4				F4	F4	F4	PF 4
25	35	" 5				F5	F5	F5	PF 5

Code									
normal	**with delete[1])**	**Meaning**	**8110**	**8151**	**8152**	**816x**	**9749 975x**	**9763**	**3270 [3])**
26	36	" 6						F6	PF 18
27	37	" 7						F7	PF 19
28	38	" 8						F8	PF 20
29	39	" 9						F9	PF 21
2A	3A	" 10						F10	PF22
2B	3B	" 11						F11	PF23
2C	3C	" 12						F12	PF24
2D	3D	" 13						F13	
2E	3E	" 14						F14	
2F	3F	" 15			PÜ			F15	
30		Data transmission (positional data)							
40	50	Data transmission (special) 0				Bypass-input			
41	51	" 1				positive acknowledgment			
42	52	" 2				negative acknowledgment			
43	53	" 3							
45	55	Mag. ID card reader					x	x	x
46	56	Data transmission (marked) 16						F16	
47	57	" 17						F17	
48	58	" 18						F18	
49	59	" 19						F19	
4A	5A	" 20						F20	
4B	5B	" 21						F21	
4C	5C	" 22						F22	
4D	5D	" 23						F23	
4E	5E	" 24						F24	
10		Delete memory							CLEAR
60		Chip card terminal						x	
80		ID card inserted					x	x	x

[1]) Screen cleared prior to input

[2]) When using the TIAM access method, reserved for ESCAPE/BREAK function

[3]) For default values, see VTSU operating parameter COMPKEYS (see the VTSU manual [1], chapter on “Operating parameters”).

9.2 Special characteristics of 3270 terminals

1. There is no roll-up mode. Once the last line on the screen has been reached, output continues in the first line.
2. When an output has terminated, the remaining information on the screen is not deleted; only the remainder of the current line and the entire following line are deleted (if output terminates in the last line on the screen, the following line is then the first on the screen). The display control character NOR is inserted at the start of the next line but one.
3. Use of logical control characters:
   - Each logical control character requires one character position on the screen. The control characters NL and VPA require two positions (one at the end of the line in which they are specified and one in the line in which output is to continue). Two or more successive control characters are merged into one character on the screen.
   - Each logical control character causes a new field to be created. Thus when logical control characters EM1 through EM4, DAR and NOR (these do not cause a new field to be created with TD terminals) are used with EXTEND=NO, an output reentered after it has been modified may be shorter than when TD terminals are used.
   - With the logical control character NUM, other specifications than those for TD terminals are possible.
4. Effect of the EDIT options:

HCOPY The entire screen is always printed out, which means that some inputs and outputs may be printed out more than once. Outputs for which no printout is required but which are on the screen at the time the hardcopy output is requested are also printed out.

EXTEND Only limited use can be made of null characters in input as they are not transferred to the computer. VTSU-B, however, adds null characters to all fields that are returned shortened, so that the user receives the length expected. In the event of individual fields not returning at all (e.g. because the "ERASE INPUT" key was pressed), VTSU-B supplies a return code.

GETFC For the mapping of 3270 function key codes, see the table on page 197.

For further details of the use of VTSUCB, refer to the VTSU manual (see [1]).

Related publications

[1] **VTSU**
Virtual Terminal Support
User Guide

Target group
Users of the DCAM and TIAM access methods and of UTM, and also the system and network administrator.
Contents
VTSU (Virtual Terminal Support) is a software product of the BS2000 operating system. It implements a virtual line terminal. A virtual terminal permits programming that is independent of the physical characteristics of the terminal in question.

[2] **BS2000/OSD-BC**
Commands, Volume 1, A-L
User Guide

Target group
The manual addresses both nonprivileged BS2000/OSD users and systems support.
Contents
This manual contains BS2000/OSD commands ADD-... to LOGOFF (basic configuration and selected products). The introduction provides information on command input. The command descriptions also specify the privileges required for using commands.

[3] **BS2000/OSD-BC**
Executive Macros
User Guide

Target group
The manual addresses all BS2000/OSD assembly language programmers.
Contents
The manual contains a summary of all Executive macros, detailed descriptions of each macro with notes and examples, including job variable macros, and a comprehensive general training section.

[4] **BS2000/OSD-BC**
System Installation
User Guide

Target group
BS2000/OSD system administration
Contents
This manual describes
– the generation of the hardware and software configuration with UGEN
– the following installation services:
  – disk organization with MPVS
  – program system SIR
  – volume installation with SIR
  – configuration update (CONFUPD)
  – utility routine IOCFCOPY

[5] **SDF V4.0A** (BS2000/OSD)
Introductory Guide to the SDF Dialog Interface
User Guide

Target group
BS2000/OSD users
Contents
This manual describes the interactive input of commands and statements in SDF format. A Getting Started chapter with easy-to-understand examples and further comprehensive examples facilitates use of SDF. SDF syntax files are discussed.

[6] **FHS** (TRANSDATA)
User Guide

Target group
Programmers
Contents
Program interfaces of FHS for TIAM, DCAM and UTM applications. Generation, application and management of formats.

[7] **FHS** (BS2000/OSD, TRANSDATA)
Dialog Extension for TIAM and SDF-P
User Guide

Target group
Application developers
Contents
The manual describes the program interface for using the FHS dialog manager in TIAM and SDF-P applications.

[8] **Network Access for Terminals** (TRANSDATA)
User's Guide

Target group
- All users wishing to utilize the capabilities of a computer via a terminal
- Network administrators

Contents
- Generation of network access
- Execution of network access for all permissible variants of network connections
- Network commands and messages

[9] **9749, 9750, 9752 Data Display Terminals** (TRANSDATA)
User's Guide

Target group
Programmers (applications programmers) who wish to program outputs to display terminals and interpret/analyze inputs from such terminals

Contents
- Notes on configuring TRANSDATA networks; description
- Description of the terminal functions
- Notes on the physical and logical programming of these functions
- Description of the message format
- Data exchange with printers
- Description of the software product PLUS

[10] **JV** (BS2000/OSD)
Job Variables
User Guide

Target group
The manual addresses both nonprivileged users and systems support.

Contents
The manual describes management and possible uses of job variables. The command descriptions are divided according to function areas. The macro calls are described in a separate chapter.

[11] **BS2000/OSD-BC**
Utility Routines
User Guide

Target group
The manual addresses both nonprivileged users and systems support.
Contents
The manual describes the utilities: CALENDAR V11.2A, CONDMPPD V11.2A, DPAGE V11.2A, INIT V11.2A, JMU V11.2A, LMSCONV V1.0B, PAMCONV V11.0A, PDPOOLS V11.2A, PVSREN V1.1A, RFUPD V11.0A, SODA V11.2A, SPCCNTRL V11.2A, TPCOMP2 V11.2A, VOLIN V11.2A.

[12] **Data Display Terminals**
Functional Description

Target group
BS2000 programmers

Contents
Description of the functional characteristics and connection capabilities of data display terminals, and also operation with connected printers

[13] **Data Display Terminals**
Code Tables

Target group
BS2000 programmers

Contents
Description of all the codes required for programming the data display terminals and for data exchange with printers

[14] **XHCS** (BS2000/OSD)
8-Bit Code Processing in BS2000/OSD
User Guide

Target group
Users of the DCAM, TIAM and UTM access methods, system administrators, and users migrating from EHCS to XHCS.
Contents
XHCS (Extended Host Code Support) is a software package of BS2000/OSD that lets you use extended character sets in conjunction with 8-bit terminals. XHCS is also the central source of information on the coded character sets in BS2000/OSD. XHCS replaces EHCS.

[15] **POSIX V1.0A** (BS2000/OSD)
Commands

User Guide

Target group
This manual addresses all users of the POSIX shell.
Content
This manual is designed as a work of reference. It describes working with the POSIX shell and the commands of the POSIX shell in alphabetical order.

[16] **POSIX** (BS2000/OSD)
POSIX Basics for Users and System Administrators
User Guide

Target group
BS2000 system administrators, POSIX administrators, BS2000 users, users of UNIX/SINIX workstations.
Contents
This manual describes the following: introduction to and working with POSIX; BS2000 software products in a POSIX environment; installing, controlling and exiting the POSIX subsystem; managing POSIX users via BS2000.

[17] **C Library Functions for POSIX Applications (BS2000/OSD**)
Reference Manual

Ordering manuals

The manuals listed above and the corresponding order numbers can be found in the Siemens Nixdorf *List of Publications*. New publications are described in the *Druckschriften-Neuerscheinungen (New Publications)*.

You can arrange to have both of these sent to you regularly by having your name placed on the appropriate mailing list. Please apply to your local office, where you can also order the manuals.

Index

Contents

TIAM V11.2 (BS2000/OSD, TRANSDATA)

User Guide

Target group
– BS2000 users (non-privileged)
– Programmers
Contents
– All TIAM commands and macros
– The TIAM COBOL interface with the TIAM COBOL macros
– Examples
Applications
BS2000 timesharing mode

Edition: Juli 1995

File: TIAM.PDF

Information on this document

On April 1, 2009, Fujitsu became the sole owner of Fujitsu Siemens Computers. This new subsidiary of Fujitsu has been renamed Fujitsu Technology Solutions.

This document from the document archive refers to a product version which was released a considerable time ago or which is no longer marketed.

Please note that all company references and copyrights in this document have been legally transferred to Fujitsu Technology Solutions.

Contact and support addresses will now be offered by Fujitsu Technology Solutions and have the format *...@ts.fujitsu.com*.

The Internet pages of Fujitsu Technology Solutions are available at *http://ts.fujitsu.com/*...
and the user documentation at *http://manuals.ts.fujitsu.com*.

Hinweise zum vorliegenden Dokument

Zum 1. April 2009 ist Fujitsu Siemens Computers in den alleinigen Besitz von Fujitsu übergegangen. Diese neue Tochtergesellschaft von Fujitsu trägt seitdem den Namen Fujitsu Technology Solutions.

Das vorliegende Dokument aus dem Dokumentenarchiv bezieht sich auf eine bereits vor längerer Zeit freigegebene oder nicht mehr im Vertrieb befindliche Produktversion.

Bitte beachten Sie, dass alle Firmenbezüge und Copyrights im vorliegenden Dokument rechtlich auf Fujitsu Technology Solutions übergegangen sind.

Kontakt- und Supportadressen werden nun von Fujitsu Technology Solutions angeboten und haben die Form *...@ts.fujitsu.com*.

Die Internetseiten von Fujitsu Technology Solutions finden Sie unter *http://de.ts.fujitsu.com/*..., und unter *http://manuals.ts.fujitsu.com* finden Sie die Benutzerdokumentation.