Fujitsu Software BS2000 CAPRI

Version 21.0
June 2025

Readme

1 Intention

The subsystem CAPRI should replace the old SVC79 interface as it is today. The new one is featured by separation of TU part and TPR part (no BINDER problems anymore). The new interface is offered in a way, that the DSL interface is resolved completely and therefore a much simplified programming is possible.

2 Short description of functionality

Supporting new processors creates problems building new SVC79 programs. These programs must be compiled in multiple instantiations using the correct Assembler.

CAPRI offers 5 different interfaces:
1.) A command interface to start TPR routines, which use input files only.
2.) A command interface to load TPR modules, which are called using SVC 79.
3.) A new SVC79 interface, which can independent load TPR modules dynamically.
4.) Direct call of a system routine without a TPR module.
5.) Using a TPR SVC in a TU program.

SVC 79 is not strongly coupled to the user id $TSOS, but can be managed by a GUARD additively. The system administrator can assign the SVC97 function to other user ids, too. If GUARD does not exist, the behavior is according to the CLASS2 option SVC79.
With the new interface, the user id $SERVICE is not supported anymore.
The old SVC79 interface is still supported.

3 System assumptions

The extended access from user ids other than $TSOS using GUARD interface is not possible until SECOS V4.0B.

4 Command interface

4.1 Direct program execution using START-TPR-PROGRAM

```
 ------------------------------------------------------------------------
 | START-TPR-PROGRAM                                                    |
 ------------------------------------------------------------------------
 | FROM-FILE=*LIBRARY-ELEMENT(..)                                       |
 |           |                                                          |
 |           |  *LIBRARY-ELEMENT(..)                                    |
 |           |     LIBRARY=*STD | <filename>                            |
 |           |                                                          |
 |           |    ,SYMBOL= <composed-name 1..8 with underscore>         |
 |           |                                                          |
 |           *BY-IMON(..)                                               |
 |           |                                                          |
 |           |  LOGICAL-ID=SYSLNK | <composed-name 1..30>               |
 |           |                                                          |
 |           |  ,INSTALLATION-UNIT=<composed-name 1..30>                |
 |           |                                                          |
 |           |  ,VERSION=<version 3..7> | *STD                          |
 |           |                                                          |
 |           |  ,DEFAULT-LIBRARY=<filename>                             |
 |           |                                                          |
 |           |  ,SYMBOL=<composed-name 1..8 with underscore>            |
 |                                                                      |
 |                                                                      |
 |,PROGRAM-PARAMETERS= *NONE                                            |
 |                     | <c-string  1..64>                              |
 |                     | <x-string  1..128>                             |
 |                     | <alpha-num 1..64>                              |
 |                                                                      |
 |,DIAGNOSIS-STOP=*NO | *AFTER-LOAD                                     |
 |                                                                      |
 ------------------------------------------------------------------------
```

This command loads a routine into the privileged CL4 system memory and starts it. The routine returns to the command and will be unloaded. CAPRI puts its return codes to SYSOUT. If the entry is preloaded by the command LOAD-TPR-PROGRAM, so this entry will be called, and the module is not unloaded.

Operand description

LIBRARY
Name of the library containing the routine with the specified entry. If name is *STD, the library of CAPRI is used.

SYMBOL
Entry or csect of a TPR routine.

LOGICAL-ID
Logical name assigned by IMON to a file. For module libraries the ID is usually SYSNK.

```
INSTALLATION-UNIT
 Name of the product installed by IMON.

VERSION
 Version of the product installed by IMON. *STD is the actual version.

DEFAULT-LIBRARY
 If the product id is not installed by IMON, this will be the name of the
 module library.

PROGRAM-PARAMETERS
 Designates a field of 64 bytes and is passed to the TPR Routine unchecked.

DIAGNOSIS-STOP= *NO / *AFTER-LOAD
 This parameter enables to load the routine and delay its start, until a
 message on SYSOUT is answered.
 This permits a diagnose using HELGA, IDIAS or DAMP. The routine
 is waiting until the question CAP0004 is answered. Testing a routine please
 note, that the routine will be loaded for every call. Therefore, these
 entries exist repeatedly in the system.
 The diagnose switch additionally prevents the release of the parameter
 Areas after abnormal termination.
```

4.2 Load and unload TPR modules

These commands are for optimizing the accesses only. The programs may be preloaded. Any user uses this TPR program only. The requested memory is limited to that program and the dynamic loading is omitted.

4.2.1 LOAD-TPR-PROGRAM

```
 ------------------------------------------------------------------
 | LOAD-TPR-PROGRAM                                               |
 ------------------------------------------------------------------
 | FROM-FILE=*LIBRARY-ELEMENT(..)                                 |
 |           |                                                    |
 |           |    *LIBRARY-ELEMENT(..)                            |
 |           |         LIBRARY=*STD | <filename>                  |
 |           |                                                    |
 |           |        ,SYMBOL= <composed-name 1..8 with underscore> |
 |           |                                                    |
 |           *BY-IMON(..)                                         |
 |           |                                                    |
 |           |  LOGICAL-ID=SYSLNK | <composed-name 1..30>         |
 |           |                                                    |
 |           |  ,INSTALLATION-UNIT=<composed-name 1..30>          |
 |           |                                                    |
 |           |  ,VERSION=<version 3..7> | *STD                    |
 |           |                                                    |
 |           |  ,DEFAULT-LIBRARY=<filename>                       |
 |           |                                                    |
 |           |  ,SYMBOL=<composed-name 1..8 with underscore>      |
 |                                                                |
 ------------------------------------------------------------------
```

The command loads a module (SYMBOL) from the given library into
the CL4 system memory. Each loading has its context. It is accepted
with the privilege TSOS only.

Operand description

LIBRARY
 Name of the library containing the routine with the specified entry.
 If name is *STD, the library of CAPRI is used.

SYMBOL
 Entry or csect of a TPR routine.

LOGICAL-ID
 Logical name assigned by IMON to a file. For module libraries the ID
 is usually SYSNK.

INSTALLATION-UNIT
 Name of the product installed by IMON.

VERSION
 Version of the product installed by IMON. *STD is the actual version.

DEFAULT-LIBRARY
 If the product id is not installed by IMON, this will be the name of the
 module library.

4.2.2 UNLOAD-TPR-PROGRAM

```
 ------------------------------------------------------------------
 | UNLOAD-TPR-PROGRAM                                             |
 ------------------------------------------------------------------
 | SYMBOL= <text 1..8>                                            |
 ------------------------------------------------------------------
```

The command unloads the complete context with this symbol from the
system. It is accepted with the privilege TSOS only.
Erroneous loaded routines must be unloaded using this command.

Operand description

SYMBOL
 Entry or csect of a TPR routine.

4.2.3 SHOW-TPR-PROGRAM-STATUS

```
 ------------------------------------------------------------------------
 | SHOW-TPR-PROGRAM-STATUS                                              |
 ------------------------------------------------------------------------
 |  SYMBOL=*ALL | *OWN | *SHARE | *SYSTEM                               |
 |                                                                      |
 |                                                                      |
 |  ,INFORMATION=*STD | *TASK                                           |
 |                                                                      |
 ------------------------------------------------------------------------
```

```
The command lists the loaded entries and the corresponding libraries
and their status.

INFORMATION=*STD
Symbol=*ALL     all symbols loaded by Capri will be shown.
       *OWN     only the symbols in the own task (CL5) will be shown.
       *SHARE   all symbols loaded share into the CL4 will be shown.
       *SYSTEM all directly called system symbols from the EXEC will be shown.

The output format is as follows:

 ------------------------------------------------------------------------------
 |TYP| SYMBOL |ADDRESS |STATE   |RC      |LIBRARY
 |C01|TSTROUT |EC000080|LOADED  |00000000|$TSOS.SKMLNK.CAPRI.210
 |C04|TSTY    |EF980CC0|WARNING |04010604|$XXX.SKMLNK
 |E04|TSTY1   |EF980D00|WARNING |04010604|$XXX.SKMLNK
 |X  |PBVSVS  |CC4738C0|SYSTEM  |00000000|$TSOS.SYSTEM-ENTRY
 |C07|ABCSECT |BEFF0010|LOD_0AJK|00000000|$TSOS.SKMLNK.AB.010
 |S06|TEST99  |CF800000|LDY_0ASJ|00000000|$TSOS.SKMLNK.TEST.011

Column        Meaning
TYP=@nn       Type of the symbol
              nn        Number of a link context; all symbols with the
                        same number belong to one program
              @         type of symbol
              S           Start/Csect
              E           Entry
              I           ILE
              C           Common
              X           System entry of the EXEC

SYMBOL        Name of the entry or csect loaded

ADDRESS       Address of the entry in CL2/CL4 or CL5 of the corresponding task

STATE         LOADED      module loaded without error
              WARNING     module loaded with error(s)
              LOD_<tsn>   module loaded locally to the task without error
              WRN_<tsn>   module loaded locally to the task with error(s)
              LDY_<tsn>   module loaded by START-TPR-PROG in CL4 memory
              WDY_<tsn>   module loaded by START-TPR-PROG in CL4 memory
                          with warnings
              SYSTEM      This entry is in the EXEC

RC            Return code of $PBBND1 (the return code is usually identical to
                                      the return code of the BIND call)

LIBRARY       Library containing the loaded module



INFORMATION=*TASK
If the program counter of a task is in the loaded TPR routine right now,
the task is shown here.
The output format is as follows:
+----+--------+------------------------+
|TSN |SYMBOL  |CONTXT                  |
+----+--------+------------------------+
|0ABC|TEST099 |$SYSTEM#CAPRI###1000004E  Context CL4 task specifically suffix=TID
|0BCE|TEST05  |SYSTEM##CAPRI###20000050  Context CL5 task locally      suffix=TID
|0DEE|TEST002 |$SYSTEM#CAPRI###00000004  Context CL4 system globally   suffix=counter
```

```
 TSN           Task Sequence Number
 SYMBOL         entry of the TPR routine
 CONTEXT        name of the bind context
                $ = privileged system context
                suffix = TID (task ID) or unique number
```

4.3 Return codes

```
Each command has a CMD return code

Command successful:

RC2:    X'00'
RC1:    X'00'
MRC:    C'CMD0001'


TPR program does not return a return code:

RC2:    X'02'
RC1:    X'00'
MRC:    C'CAPNORC'


Error in subsystem CAPRI:

RC2:    X'00'
RC1:    X'20'
MRC:    C'CAP@@@@'

Any main return code CAPnnnn exists in the help file and
can be questioned using /HELP-MSG-INFORMATION CAPnnnn.


Error in TPR program:

RC2:    X'00'
RC1:    X'40'
MRC:    C'@@@@@@@'

The main return code results from the TPR program and is passed in CAPKEY.
```

4.4 Redefinition of the START-TPR-PROGRAM command in START-XXX

```
Using this version, it is possible for the START-TPR-PROGRAM to generate an own
START-XXX command.
To enable it the program needs to be prepared. Most important the internal
operand names must be changed: they must be unique in the whole system but
fulfill the name convention of Capri.
The internal operand name usually is $CAPXexe. The check is done from position 5
for "EXE". The $ designates a privileged interface.
The character sequence "CAP" should be replaced by the components identifier (3 char)
and "X" should be replaced by an arbitrary character.
The next chapter shows a template of the START-xxx command.
```

4.5 START-XXX Template

The template for the IMON interface is

```
      //ADD-COMMAND NAME=START-<xxx>,INTERNAL-NAME=$<ccc><f>EXE,                -
            STANDARD-NAME=START-<xxx>,                                         -
            ALIAS-NAME=<xxx>,                                                  -
            DOMAIN=(SYSTEM-MANAGEMENT,PROGRAM),                                -
            IMPLEMENTOR=P2(ENTRY=NLKRES56,INTERFACE=ISL,                       -
                CMD-INTERFACE=*TRANSFER-AREA(MAX-STRUC-OPERAND=11)),           -
            PRIVILEGE=(STD-PROC)

      //ADD-OPERAND NAME=FROM-FILE,INTERNAL-NAME=FROFI                         -
            STANDARD-NAME=FROM-FILE,RESULT-OPERAND-NAME=*POSITION(5),          -
            DEFAULT='*BY-IMON'
      //ADD-VALUE KEYWORD(STAR=MAN),STRUCTURE=Y,VALUE='*BY-IMON'

      //ADD-OPERAND NAME=LOGICAL-ID,INTERNAL-NAME=ID,STANDARD-NAME=*NAME,      -
            DEFAULT=C'<syslnk>'(ANALYSE-DEFAULT=NO)
      //ADD-VALUE TYPE=COMP-NAME(LONG=30,SHORT=1)

      //ADD-OPERAND NAME=INSTALLATION-UNIT,INTERNAL-NAME=UNA,DEFAULT='<unit>',  -
-
            STANDARD-NAME=INSTALLATION-UNIT,RESULT-OPERAND-NAME=*POS(POS=7)
      //ADD-VALUE TYPE=COMP-NAME(LONG=30,SHORT=1)

      //ADD-OPERAND NAME=VERSION,INTERNAL-NAME=UVER,STANDARD-NAME=VERSION,     -
            RESULT-OPERAND-NAME=*POS(POS=8),DEFAULT='<*std>'
      //ADD-VALUE TYPE=PRODUCT-VERSION(USER-INT=*ANY(CORR-STATE=*ANY)
      //ADD-VALUE TYPE=KEYWORD(STAR=MAN),VALUE='*STD'

      //ADD-OPERAND NAME=SYMBOL,INTERNAL-NAME=ENTR,STANDARD-NAME=SYMBOL,       -
            DEFAULT=<symbol>,RESULT-OPERAND-NAME=*POS(POS=1)
      //ADD-VALUE TYPE=COMPOSED-NAME(LONG=8,SHORT=1,UNDERSCORE=YES)

      //ADD-OPERAND NAME=DEFAULT-LIBRARY,INTERNAL-NAME=DLIB,DEFAULT='<filename>',-
            STANDARD-NAME=DEFAULT-LIBRARY,RESULT-OPERAND-NAME=*POS(POS=9)
      //ADD-VALUE TYPE=FILENAME
      //CLOSE-STRUCTURE
```

Replaceable values:

<xxx>	program name
<ccc>	3-character components identifier
<f>	arbitrary character cccf must be unique in the whole system
<syslnk>	logical ID of the load library
<unit>	Installation Unit see IMON
<*std>	newest version (*STD) or version number
<symbol>	entry of the TPR module
<filename>	name of the load library if not identified by IMON

5 Program interface

5.1 SVC79

5.1.1 Old compatible interface

The SVC79 in its current form works as follows: register 1 must contain
the address of the TPR routine; then the SVC is processed.
If the user is privileged, the routine is called in TPR mode.
If not, the program is continued immediately after the SVC. No TPR
code is executed.

5.1.2 New interface

Register 1 of the old interface contains always an instruction address,
i.e. its value must be even. This fact is used to indicate the new
interface. If the 2**0 bit is on, it is the new interface.
The new interface has a parameter list containing the name of the entry.
The subsystem CAPRI assigns an address to the entry and calls the
routine. The parameter list is addressed by register 1. If CAPRI is not
loaded, this interface must not be called. It will result in a system
dump in module ETMSF or in the TPR routine with IW 5C.

5.2 Description

5.2.1 Direct execution

The TPR routine must be available in a library and compiled
according to the system conventions.

5.2.2 Old SVC79

The old SVC79 may still be called, whereby the TPR routine is part
of the program.
Additionally, it is possible to preload the TPR part using system
administrator commands. A program running under TSOS may get the
address of the TPR entry using the BLS interface VSVI1. Then register 1
must be provided with this address. Via SVC 79, the process continues
at this address.
The old interface specification will continue to exist.

5.2.3 New SVC79

The new SVC79 has a parameter list (chapter 5.4.2); it contains the name
of the entry. If the entry is preloaded, it is called. If not,
using the library the entry is loaded by CAPRI automatically.

The module is loaded using the parameter MEMCL (of CAPINT, see
chapter 5.4.2) into the CL4 or CL5 memory. In the CL4 memory, the routine

must be reentrant.

The parameter PERM of CAPINT controls the load/unload behavior of the module. If MEMCL=CL4, the call corresponds to an explicit LOAD-TPR-PROGRAM command. The module is not unloaded anymore.

If MEMCL=CL5, the module is unloaded during program termination.

If FUNCALL=Y, system routines are called directly. For parameter ENTRY, the system entry is given or *SVCnnn, with nnn as a valid SVC number. With the parameter USERAREA, the parameter list of the system function is passed and the corresponding function is executed.
The TU program continues immediately after the SVC 79.

5.3 Security

5.3.1 Class2 option SVC79

The execution of the CAPRI functions can - as with SVC79 - be controlled with CLASS2 option SVC79.

SVC79=3 Subsystem CAPRI may not be loaded.

SVC79=1 Command START-TPR-PROGRAM or CAPINT call must be acknowledged at console (console message CAP0007).

SVC79=0 Command START-TPR-PROGRAM or CAPINT call is executed immediately.

The subsystem is able to expect the load library xxxLNK.CAPRI.010 and the TPR module library with access right USER-ACCESS=*OWNER-ONLY.
The system administrator is responsible for the TPR routines in this library.

5.3.2 GUARD

The system administrator (user id TSOS) may provide this functionality to other user ids, independent from the CLASS2 option SVC79.
The release works via the coownership function of the load library.
The coownership function is described in SECOS reference manual volume 2. The commands described below enable the user to access a TPR module.
This is only a part of the possibilities provided by GUARD.
It is suggested to release load libraries only which accessible by the system administrator only. These libraries should have the property USER-ACCESS=*OWNER-ONLY (ISP syntax SHARE=NO).

5.3.2.1 Installation of an ACCESS CONDITION

First, a connection between the user id and the access rights must be established. The command is

```
ADD-ACCESS-CONDITION
    GUARD-NAME=<file name of the protection guard>
    ,SUBJECTS=*USER(USER-ID=<name 1..8> list-poss(20))
    ,ADMISSION=*YES
```

```
<file name of the protection guard> connection between the access
GUARD and the coowner GUARD. The simplest ADMISSION is: *YES,
access is permitted.
(More possibilities are in the SECOS manual, volume 2)
```

5.3.2.2 Installation of the SVC79 coownership

```
Now, the connection between the module to be loaded and the access
GUARD must be established. Therefore use the command:

     ADD-COOWNER-PROTECTION-RULE
         RULE-CONTAINER-GUARD=SYS.UCC
        ,PROTECTION-RULE=<administration-name>
        ,PROTECTION-OBJECT=
              *PARAMETER(NAME=<composed-name>,
                         CONDITION-GUARD=<file name of the protection guard>)

<Administration-name> name of the rule to be entered..
<composed-name> connection between the module and the object.
This name has a structure like a file name (wildcards permitted).
CAPRI builds a new file name from the user id, the library name and
the entry to be loaded. Its format is:

<user id without $>.<library name 1..24>.<entry>

If the file name (truncated to 24 characters) ends with a dot, this dot
will be used as separator.
For system calls, there are defined file names:

System entry:    TSOS.SYSTEM-ENTRY.<entry>
SVC:             TSOS.SYSTEM-SVC.SVC<nnn>


Example:

A library named SYSLNK.HUGO.030 is in $TSOS and an entry HUGOSTRT
is loaded from this library.

CAPRI creates the following file name to be checked:

$TSOS.TSOS.SYSLNK.HUGO.030.HUGOSTRT

Then, <composed-name> in GUARD must be

TSOS.SYSLNK.HUGO.030.HUGOSTRT

If all entries of this library are accessible, the following
name is possible:

TSOS.SYSLNK.HUGO.030.*

If this library is in the user id $ABC, the name to be checked is:

$TSOS.ABC.SYSLNK.HUGO.030.HUGOSTRT
```

5.4 Call

5.4.1 Old SVC79

The routine to be called is called via the system function
$FNAT; the used registers are that way.
Register:

R1	Start address of the routine
R2	A(TCB)
R3	A(XVT)
R4	A(interrupted PCB; caller)
R5	PC of the interrupted PCB

All other registers have the same contents as in the calling routine.

5.4.2 New SVC79

The new SVC79 is called via the macro CAPINT.

Operation	Operands
CAPINT	ENTRY=name
	,LIBNAM=pathname
	[,PAR=*NONE | text 1..64]
	[,DIAG=N|Y]
	[,PERM=N|Y]
	[,MEMCL=4|5]
	[,MSG=N|Y]
	[,FUNCALL=N|Y]
	[,USERAREA=adr]
	MF=(E,adr) | (E,(reg)) | L | C | (C,pre) | D | (D,pre)

ENTRY	Name of the symbol or system entry to be loaded or *SVCnnn with nnn = SVC number
LIBNAM	Complete path name of a library containing the routine with the given entry. This library does not need to have SHARE=YES.
PAR	64-byte field, passed to the routine. It is for parametrizing the TPR routine if wanted.

```
DIAG            For testing purposes only.  If "Y" it stops after loading.

PERM            Enables the routine to stay permanently loaded.
                "N": module will be unloaded after call.
                "Y": unloading is according to the memory class.
                   CL4: to unload the routine with the command
                        UNLOAD-TPR-PROGRAM only.
                   CL5: module is unloaded at program termination.

MEMCL           Memory class in which the module will be loaded. With
                MEMCL=5 the module is loaded task specifically; though
                routines which are not reentrant can be loaded.

MSG             "N": no messages of CAPRI are sent to SYSOUT.

FUNCALL         Enables direct call of a system routine or a system SVC.
                "Y": CAPRI uses $VSVI1 to get the entry in the system.
                The parameters for the routine must be located in USERAREA.
                CAPRI branches directly to this routine directly.


USERAREA        Enables the TU routine to pass a data area to the TPR
                routine or - if a direct system call or system SVC -
                to pass the actual parameters.
```

5.5 Interface Area (CAPINT)

```
CAPINT          DSECT
CAPHDR          DS      0F              Header
CAPFCTU         DS      H'480'          Function unit
CAPFCT          DS      X'00'           Function
CAPFCTV         DS      X'01'           Function Version Interface Version 01
CAPRC           DS      F               Return code
CAPDEF          EQU     X'FFFFFFFF'     RC undefined initial value
CAPOK           EQU     X'00000000'     RC Okay
                DS      C
CAPKEY          DS      CL7             MSGKEY as Insert for MSG CAP0003
CAPENTR         DS      CL8             Name of called entry
CAPLIB          DS      CL54            Name of Library
CAPPAR          DS      CL64            Parameter like CMD parameter
CAPNONE         EQU     C'*NONE'        No parameter available
CAPUADR         DS      A               Address of user area
CAPIND          DS      X               Indicator
CAPPERM         EQU     X'01'           Routine is loading permanently
CAPDIAG         EQU     X'02'           Stop after loading
CAPNOMS         EQU     X'04'           no Message to SYSOUT
CAPMC5          EQU     X'10'           Memory class 5 for BIND (task local)
CAPSYCA         EQU     X'20'           Entry is system entry or SVC
CAPICALL        DS      CL3             Indicator of calling interface
CAPCMD          EQU     C'CMD'          Interface is called by START-TPR-PRO
CAPSVC          EQU     C'SVC'          Interface is called by SVC
CAPEADR         DS      A               Address of entry
CAPTCB          DS      A               Address of TCB
CAPEXVT         DS      A               Address of EXVT
CAPISTK         DS      A               Address of interrupted PCB
CAPINIA         DS      A               Address of predecessor PCB
CAPSVR1         DS      F               Register 1 after SVC return
```

```
CAPSVR15        DS       F               Register 15 after SVC return
                DS       6F              reserved for future use
CAPEND          DS       0F              End of interface area
CAPLEN          EQU      *-CAPINT        Length of parameter

Description of parameters:

CAPFCTV         Interface version number V01
CAPRC           Return code    F'00' routine done correctly
                               CAPKEY=CAPOKAY, otherwise see MF=D
CAPKEY          This key is given with the end message. If it begins
                with "CAP", it is a return code of CAPRI an can be
                analyzed using /HELP-MSG-INF CAPaaaa.
CAPENTR         Name of the symbol to be called, of a system entry or
                a SVC.
CAPPAR          64-byte transfer parameter. This parameter can be
                passed with the command START-TPR-PROGRAM, too.
CAPLIB          File name of the load library
CAPUADR         Address of a user area. This address will be passed
                by CAPRI unchecked. With system entry or SVC it
                denotes the address of the parameter list.
CAPPERM         Routine loaded only once, and it is only branched to it
                afterwards.
CAPDIAG         After load/before unload a message to be acknowledged
                is given. Useful with diagnose of the TPR routine.
CAPNOMS         All messages are suppressed; only return codes are
                transferred.
CAPMC5          Task local loading of the routine (it does not need to
                be reentrant); useful with CAPPERM=Y only.
CAPSYCA         The symbol given in CAPENTR is a system entry or
                a system SVC.
CAPEADR         Address of the symbol (for diagnostic purposes only;
                not useful for direct call).
CAPSVR1         Register 1 execution of SVC, e.g. necessary with $REQM
CAPSVR15        Register 15 execution of SVC, e.g. necessary with $REQM
```

5.6 Programming rules

5.6.1 New SVC79

The call is as follows:

```
                        CAPINT  MF=(E,CAPPAR)
                        ...
                        .....
         CAPPAR         CAPINT  MF=L,LIBNAM=LIBABC,ENTRY=TPRENTRY
```

The TPR routine is according to the DSL conventions.

```
        <entry>         @ENTR   TYP=E,ENV=SPLSPEC,RETURNS=NO
                        USING   CAPINT,R1
                        ......
                        MVC     CAPRC,...
                        MVC     CAPKEY,=CL7'_______'
                        @EXIT
                        @END
```

Calling a system routine

```
                        LA      3,VSVI
                        USING   VSVIDS,R3
                        LA      4,SYSCALL
                        USING   CAPINT,4
                        ......
                        CAPINT  MF=(E,SYSCALL)
                        @IF     EQ
                        CLC     CAPRC,=F'00'    Test if SVC79 okay
                        @THEN
                        @IF     EQ
                        CLC     PBVSVRET,=F'00' Test if system call okay
                        @THEN
                        ...
                        ....
                        @END
SYSCALL                 CAPINT  ENTRY=PBVSVS,FUNCALL=Y, -
                                USERAREA=VSVI, MF=L
VSVI                    $VSVI1  MF=L,SELECT=ALLLIST;SCOPE=GLOBAL, -
                                INCTX=RCTX,OUTADDR=INS1,OUTLEN=10000, -
                                RUNMOD=ADV
CAPINT                  CAPINT  MF=D
VSVIDS                  $VSVI1  MF=D
                        ....
```

Execute a SVC

```
                        LA      4,SYSSVC
                        USING   CAPINT,4
                        ......
                        CAPINT  MF=(E,SYSSVC)
                        @IF     EQ
                        CLC     CAPRC,=F'00'     Test if SVC79 okay
                        @THEN
                        @IF     EQ
                        CLC     CAPSVR15,=F'00'  Here: R15 after the SVC,
                                                 else in the parameter
                        @THEN
                        L       R1,CAPSVR1       Here: R1 after the SVC
                        ...
                        ....
                        @END
SYSSVC                  CAPINT  ENTRY=*SVC228,FUNCALL=Y, -
                                USERAREA=REQM,MF=L
REQM                    $REQM   MF=L,CLASS=5,BYTES=X'2000'
CAPINT                  CAPINT  MF=D
                        ....
```

5.6.2 Old SVC79

The routine must be programmed according to the following template:

```
<TPR-module>   CSECT
               ENTRY    <entry>
<entry>        @ENTR    TYP=D
               LR       R10,R1    Register 10 is base register of the routine
               USING    <entry>,R10
               [USING   ETCB,R2]  Register 2 points to the TCB
               [USING   EXVT,R3]  Register 3 points to the EXVT
               [USING   ESTK,R4]  Register 4 points to the interrupted PCB
* R13 points to a save area of 18F and R12 points to the address of V(ITSPRV)
*********************************************************

               Here is the routine itself:


               $FNDT    back to the caller
               @END
               DTCB     MF=D
               DXVT     MF=D
               DSTK     MF=D
               END
```

5.7 Compiler options

For compiling the privileged routine use following
ASSEMBLER/ASSTRAN parameters:

5.7.1 /390

ASSEMBH or ASSTRAN

MODULE-GENERATION(MODE=STD[,DESTINATION-CODE=STD])

5.7.2 x86

ASSTRAN from 5.2B

MODULE-GENERATION(MODE=STD,DESTINATION-CODE=X86(RUN-MODE=TPR))