

Program Services Guide

=====

--SVC--		FUNCTION
DEC	HEX	
3	03	EXIT
3	03	EOJ (VSE)
6	06	LINK
7	07	XCTL
8	08	LOAD
8	08	CDLOAD (VSE)
9	09	DELETE
9	09	CDDELETE (VSE)
13	0D	ABEND
18	12	BLDL
51	33	SNAP
60	3C	ESTAE/ESTAEX
109	6D	ESPIE

Supporting Macros

PSAD	DSECT for PSA fields.
CVTD	DSECT for CVT fields.
ZCVT	DSECT for ZCVT fields.

IHAEPIC	DSECT for the ESPIE control block
IHASDWA	DSECT for the SDWA (ESTAE) control block

EQUIREGS	Creates the standard equates for general and floating point registers.
----------	--

YREGS	Creates the standard equates for general registers.
-------	---

SUBENTRY	Creates savearea, standard linkage and sets base register 13.
----------	---

SUBEXIT	Returns using standard linkage.
---------	---------------------------------

PERFORM	Call local procedure using no registers or stack if
---------	---

RENT=YES.	
-----------	--

PM	Copy of PERFORM with shorter name.
----	------------------------------------

PENTRY	Define start of local procedure for use with PERFORM and
PEXIT.	

PEXIT	Exit from local procedure using no registers or stack if
RENT=YES.	

EXIT	Returns from current task to caller or OS via SVC 3.
------	--

CALL	Call a sub-program.
------	---------------------

ZDOCPROG.TXT

SAVE Save registers in savearea.
RETURN Restore callers registers and return with return code.

RESTORE Restore callers registers.

Savearea and linkage conventions

Standard save area is defined as follows:

DS	0CL72		
DS	F	+ 0	reserved
DS	F	+ 4	address of callers savearea
DS	F	+ 8	address of our savearea
DS	15F	+12	callers GR14 through GR12

A program is invoked with entrypoint in GR15 and return address in GR14.

GR15 is expected to contain a return code upon exit by convention.
GR13 conventionally points to our savearea.

Passing parameters to the initial program

There are two methods of doing this, via SYSPARM on CALL MZ390
or PARM on CALL EZ390.

In either case enclose the entire parm to be passed in double quotes.
The double quotes are required to handle commas and spaces that
otherwise

cause Windows BAT parser to split the parm. The double quotes are not
required if there are no commas or spaces in the text. If single
quotes

are included in text, they are passed on to &SYSPARM.
There is a limit of 32767 bytes for the text.

SYSPARM will be transferred to the Macro variable &SYSPARM.

eg. CALL MZ390 ... "SYSPARM(HELLO WORLD)"

Access the text by coding (eg)...

label DC C'&SYSPARM'

Which will become...

label DC C'HELLO WORLD'

PARM can be accessed via GR1 at program entry and consists of
a halfword length followed by the text.

ZDOCPROG.TXT

eg. CALL EZ390 ... "PARM(HELLO WORLD)"
GR1 points to DC H'11',C'HELLO WORLD'

If single quotes are included around text in PARM they are removed so
CALL EZ390 "PARM('HELLO WORLD')" will also result in
GR1 pointing to DC H'11',C'HELLO WORLD'

Macro Documentation

1) BLDL

name BLDL 0,list

Build a directory list for use with LOAD, DELETE, LINK and
XCTL. After a BLDL, an individual table entry may be used
in these macros via the DE= parameter.

list

May be a label or (reg) and points to a storage area in
the following format:

H'count' The number of entries in the table
The next 7 fields constitute an entry.

H'entry length'	The length of the following entry
CL8'name'	The name of the program
XL2'00'	TT (unused)
X'00'	R (1=found)
X'00'	K (unused)
X'00'	Z (1=program found in storage)
X'00'	C (unused)

entry length must be a minimum of 12, which would omit the
Z and C fields.

Names must be in alphameric order, a suffix of .390 is
assumed.

GR15 has a return code:

- 0 All programs found
- 4 Some programs not found
- 8 Invalid count or invalid entry length

eg.

BLDL 0,LIST1

ZDOCPROG.TXT

LOAD DE=BLDL2

```
...
LIST1 DC H'2'
BLDL1 DC H'14',CL8'MYPROG1',XL6'00'
BLDL2 DC H'14',CL8'MYPROG2',XL6'00'
```

Note: In Z390, there is no performance benefit in issuing a BLDL before a LOAD, DELETE, LINK or XCTL.

Register Usage:

R1 = BLDL list
R15= Return code

2) LOAD, CDLOAD, DELETE, CDDELETE, LINK, XCTL and RESTORE

Parameters common to some of these macros are described later.

name LOAD EP=,EPLOC=,DDNAME=,DSNAME=,DE=,LOADPT=,ERRET=
Load a program or module.

GR0 returns the address of the loaded module.
If LOADPT is used, then GR0 may be stored at a label, or the address contained in a general register.

GR1 returns the length as follows:
a) For a program, the number of doublewords (8-byte units).
b) For other modules, the length rounded up to the next doubleword boundary.

ERRET=(reg) or label
If the return code is non-zero, the label or the address in reg is invoked.

Register Usage:

R0 = Pointer to program name or BLDL entry, returned address
R1 = Returned length
R15= Path pointer and return code

GR15 has a return code:
0 Load ok
4 Module not found

Abends:

S80A Out of memory

ZDOCPRG.TXT
name CDLOAD phasename Maps to LOAD EP=phasename
name CDLOAD (reg) Maps to LOAD EPLOC=(reg)
VSE only.
Load a program or module.

GR0 and GR1 return the address of the loaded module.
Length is not returned.

Register Usage:

R0 = Pointer to program name, returned address
R1 = Returned address
R15= Return code

GR15 has a return code:

0 Load ok
4 Module not found

Abends:

S80A Out of memory

name DELETE EP=,EPLOC=,DDNAME=,DSNAME=,DE=
Delete a program or module.

Register Usage:

R0 = Pointer to program name or BLDL entry
R15= Path pointer

GR15 has a return code:

0 Load ok
4 Module not found

name CDDELETE phasename Maps to DELETE EP=phasename
name CDDELETE (reg) Maps to DELETE EPLOC=(reg)
VSE only.
Delete a program or module.

Register Usage:

R0 = Pointer to program name

GR15 has a return code:

0 Load ok
4 Module not found

name LINK EP=,EPLOC=,DDNAME=,DSNAME=,DE=,PARAM=,VL=
Load and pass control to another program.

ZDOCPROG.TXT

Return to 'linker'.

Register Usage:

R0 = Pointer to program name or BLDL entry
R1 = Parameter list
R15= Path pointer

Abends:

S806 Module not found
S80A Out of memory

name XCTL

(fromreg,toreg),EP=,EPLOC=,DDNAME=,DSNAME=,DE=,PARAM=,VL=
Load and pass control to another program.
Return to last 'linker' or terminate.

(fromreg,toreg) is optional

Restores the specified register range from the savearea pointed to by GR13. The registers are restored from their conventional positions. The range must not specify or include the following general registers: 0, 1, 13, 15

Register Usage:

R0 = Pointer to program name or BLDL entry
R1 = Parameter list
R15= Path pointer
All registers in the range fromreg-toreg

Abends:

S806 Module not found
S80A Out of memory

When a program is loaded (with suffix .390) then relocation takes place.

EP, EPLOC, DDNAME and DSNAME are used to locate the program or module:

EP, EPLOC and DE are mutually exclusive.
DDNAME and DSNAME cannot both be present.

Various combinations are available:

EP=program

Specify the program name, maximum 8 bytes.

EPLOC=label or EPLOC=(reg)

ZDOCPROG.TXT

The label or the register must point to an 8-byte field containing the program name.

DE=label or DE=(reg)

The label or the register must point to a BLDL entry.

In all the above cases as neither DDNAME nor DSNAME are specified, then the Z390 search path is used. This may be overridden by the CALL

EZ390 parameter SYS390.

EP/EPLOC/DE and DDNAME=name

EP/EPLOC/DE and DDNAME=(reg)

DDNAME has or points to the name of a SET variable.

This variable (see examples below) may contain:

a) The complete path and filename.

In this case EP, EPLOC or DE are ignored.

b) A single path.

Only this path will be searched.

c) Multiple paths separated by plus signs.

All paths will be searched in the order specified.

eg.

LINK EP=MYLINK,DDNAME=MYPATH ...

For execution, define a .BAT (Batch) file...

SET MYPATH=drive:\path\file program specified, EP ignored

--or--

SET MYPATH=drive:\path single search path

--or--

SET MYPATH=drive:\path1+drive:\path2 multiple search paths

CALL EZ390 drive:\path\program parms

EP/EPLOC/DE and DSNAME=name

EP/EPLOC/DE and DSNAME=(reg)

DSNAME is or points to a label defined in the program which has the file spec.

The file spec must terminate with X'00' or be defined as a double-quoted string within the standard C-type constant.

ZDOCPROG.TXT

Either constant may contain:

- a) The complete path and filename.
In this case EP, EPLOC or DE are ignored.
- b) A single path.
Only this path will be searched.
- c) Multiple paths separated by semicolons or plus signs.
All paths will be searched in the order specified.

eg.

```
LINK EP=MYLINK,DSNAME=MYPATH ...  
      ...  
MYPATH DC C'drive:\path\file',X'00'  
      --or--  
MYPATH DC C'"drive:\path1;drive:\path2"'
```

Note:

In the above cases where the filename is specified in the SET variable or the DC constant, then the .390 suffix should be omitted.

The exceptions to this are LOAD and DELETE, which may be used to load or delete a non-program module and may have any suffix appended.

PARAM= and VL=

Only available on LINK and XCTL.

Used for passing a fixed or variable parameter list to a program.

PARAM=(parml,parm2,...)

The parameters can be anything that is permitted in an A-type constant. When the program is invoked GR1 points to the parameter list.

See 'Advanced topic' below for special considerations.

VL=0 or VL=1 (default)

If the called program can accept a variable parameter list, then VL=1 will turn on the senior bit (bit 0) of the last parameter.

name RESTORE (fromreg,toreg),RC=reg

ZDOCPROG.TXT

Restores the specified register range from the savearea pointed to by GR13. The registers are restored from their conventional positions. GR15 is loaded from the RC reg prior to any register restore.

Register Usage:

All registers in the range fromreg-toreg

Advanced topic:

Use counts and parameter passing.

On the first invocation and after a LOAD, LINK or XCTL, the program receiving control has its use count incremented.

When a program is DELETED, it terminates or loses control via an XCTL, then the use count is decremented.

When the use count is zero, the storage for that program is freed. When passing parameters it is important to consider whether those parameters are in a program whose storage may be reused. If in doubt, place parameters for passing on, in a separate GETMAINED area.

3) SNAP and ABEND

name SNAP STORAGE=(from,to),PDATA=(options, ...),ID=,TEXT=

Produces a component dump on the Z390 console without terminating the program.

STORAGE=(from,to)

STORAGE=((reg1),(reg2))

Optional parameter to dump some storage.

Either 'from' or 'to' can be labels or register pointers.

The first byte displayed is 'from' and the last is 'to'-1.

PDATA=(options ...)

Optional parameter to display registers and/or control blocks.

Default is PDATA=ALL.

ALL Display all registers, control blocks and storage.

When the STORAGE parameter is present only that area of storage is displayed.

REGS Display all general and floating point registers.

ZDOCPRG.TXT

GPR Display general registers.
FPR Display floating point registers.

CDE Display information related to loaded programs or
modules.

DCB Display information related to opened and closed files.

ID=nnnnn

ID=(reg)

Numeric identifier, either numeric value or general register
containing the identifier.

Specify values 0-32767, higher values are negative.

TEXT=string

TEXT='a string'

TEXT=(reg)

Character identifier.

Specify either a string without blanks, a string constant
enclosed by single quotes or a general register pointing to
a string terminated by X'00'.

The string in all cases is limited to 60 bytes.

Register Usage:

R0 = ID and flags

R1 = TEXT pointer

R14= STORAGE from

R15= STORAGE to

name ABEND id,DUMP

Terminate the program.

id Optional numeric identifier.

Values from 0 to 4095.

Displayed as abend Unnnn.

DUMP A dump is always produced, overrides the NODUMP parm on
CALL EZ390.

All storage areas are dumped.

Register Usage:

R1 = id and flags

4) ESTAE, ESTAEX and ESPIE

ESTAEX is provided for compatability, only ESTAE is described here.

ZDOCPROG.TXT

name ESTAE label,type,PARAM=
name ESTAE (reg),type,PARAM=
name ESTAE 0

When a program abends, control is given to the label or address specified. ESTAE 0 is used to cancel any previously established ESTAE routine.

type is optional
CT (default) adds a new exit
OV replaces an existing exit

PARAM=label is optional
PARAM=(reg) is optional

When specified, the address of the label or the contents of the register are made available in the ESTAE control block at ESTAPARM.

Exit invocation...

GR15 will contain the entry point, it is recommended that GR15 is not used as the base for the ESTAE routine.

GR1 contains the address of the SDWA control block.
The IHASDWA macro is the DSECT.
This area may also be addressed by using the ZCVT.
Note: SDWAXPAD points to SDWAXEME which points to SDWARC4

After processing the abend, several options are available:

a) Cancel the exit and retry the failing instruction.
Issue an ESTAE 0.
Load GR0 with the address in the rightmost bytes of

ESTAPSW

and ensure GR15=4, then return via BR R14.
This will cancel the ESTAE and re-execute the instruction that caused the abend. If the instruction abends again it will terminate the program.

b) Enter a retry or cleanup routine.
Place the retry address in GR0 and ensure GR15=4, then return via BR R14.
If ESTAE 0 has not been issued, then the ESTAE routine remains active.

ZDOCPRG.TXT

- c) Percolate through other recovery (ESTAE) routines.
The current ESTAE routine is automatically cancelled.
Set GR15=0 and return via BR R14.
This will invoke previous recovery routines or abend the program.
When percolate happens all LINK stack entries at a lower level than the latest ESTAE will be purged.

Notes:

- caused
- i) In the Z390 environment the abend code 0C5 may be caused by an internal error as well as a genuine addressing exception.
 - ii) If an abend occurs after the ESTAE exit is invoked and before ESTAE 0 or BR R14 are issued, then the program will be terminated.

See TESTSTAl for an example of ESTAE usage.

Register Usage:

R0 = exit address and flags
R1 = parameter list
R15= return code

GR15 has a return code:

0 ESTAE ok

Abends:

SFFF ESTAE stack exceeded

name ESPIE SET,addr,list,PARAM=
name ESPIE SET,(reg),list,PARAM=
name ESPIE RESET

When a program interruption occurs eg. fixed point overflow, control is given to the label or address specified.
ESPIE RESET will reset any previous ESPIE settings.

list

Syntax:

ESPIE SET,label,8	Single code
ESPIE SET,label,(1,4,6)	Multiple codes
ESPIE SET,label,((2,6))	Range of codes: 2 through 6
ESPIE SET,label,(3,5,(7,9),14)	Mixed codes: 3,5,7,8,9,14

ZDOCPRG.TXT

If any of the codes 8, 10, 13 or 14 are specified, then the appropriate bit is set on in the PSW using the SPM instruction.

The following interruption codes can appear in the list.

- 1 - operation
- 2 - privilege
- 3 - execute
- 4 - protection
- 5 - addressing
- 6 - specification
- 7 - data exception
- 8 - fixed point overflow (SPM mask bit X'8')
- 9 - fixed point divide
- 10 - decimal overflow (SPM mask bit X'4')
- 11 - decimal divide
- 12 - HFP exponent overflow
- 13 - HFP exponent underflow (SPM mask bit X'2')
- 14 - HFP significance (SPM mask bit X'1')
- 15 - HFP divide

PARAM=label is optional

PARAM=(reg) is optional

When specified the address of the label or the contents of the register are made available in the ESPIE control block.

When the exit is invoked, GR1 contains the address of the ESPIE control block. The EPIED macro is the DSECT.

The ESPIE control block is located in the ZCVT and may also be addressed by the ZCVT and IHAEPIC macros.

Note: In the Z390 environment, interruption code 5 may be caused by an internal error as well as a genuine addressing exception.

Register Usage:

- R0 = program mask
- R1 = exit address
- R15= parameter list

5) SUBENTRY, SUBEXIT, PERFORM or PM, PENTRY, PEXIT, EXIT and EOJ

name SUBENTRY CSECT=,BASES=,RENT=,RWA=,RWALNG=,STACK=,
PSTACK=,PCHECK=

ZDOCPROG.TXT

Provides a standard entry for programs.

Although name is optional, care needs to be taken if it is omitted. A CSECT or sub-program should be named.

CSECT=YES (default)

Generates: name CSECT
Standard entry.

CSECT=NO

Generates: name DS 0D
This is useful for setting up sub-programs within the main program that are invoked by the CALL macro.

BASES= default BASES=(13)

Override and extend the base registers for this program.

For RENT=NO...

It is recommended that the first register specified is GR13, if it isn't then a program is generated with non-standard linkage and may cause problems.

For RENT=YES...

The first register specified must NOT be 13.

Each additional register generates the code and USING at the standard 4K intervals. eg. BASES=(13,7,8) will cover 12K of code.

RENT=NO (default)

A standard savearea is built, and GR13 is set as the default base register. This also serves as the pointer to the program's savearea to facilitate further linkage.

The default base register GR13 may be overridden by the BASES= parameter (qv).

RENT=YES

Defines a re-entrant program.

The GETMAINED area described below is defined in the SUBENTRY macro.

ZDOCPROG.TXT

BASES= must be specified and the first register must not be GR13.

Storage is GETMAIned and GR13 is set to the savearea within this storage. The STACK= parameter can generate multiple saveareas.

STACK=n (requires RENT=YES, default 0)

Generates an addition to the GETMAIned area acquired allowing for multiple saveareas, each of these may have an additional read-only work area defined by RWALNG.

RWA=dsectname (requires RENT=YES)

RWALNG=n (requires RENT=YES, default 0)

RWALNG defines the length of an additional work area to each savearea defined by STACK=.

RWA= defines the DSECT associated with this work area.

PSAVE=YES or NO (default=YES)

Code PSAVE=NO if GR14 or GR15 are not used for other purposes.

PSAVE=YES causes extra instructions to save and restore these registers.

PSTACK=reg (requires RENT=YES, default=0)

If PSTACK=0, then the user area address of each stack entry is stored at offset +80.

Otherwise the user area address is not stored at offset +80, but loaded into the register specified.

PCHECK=YES or NO (requires RENT=YES, default=YES)

PCHECK=YES clears the stack area and sets the senior bit of the front and end pointers to 1.

Register Usage:

R0,1,2,13,14,15 have multiple uses

name SUBEXIT RC=returncode

name SUBEXIT RC=(reg)

Provides a standard exit for programs.

name is optional.

RC will return the value in GR15, zero is the default.

ZDOCPROG.TXT

If SUBENTRY used the parameter RENT=YES then the whole stack area will be FREEMAINED before GR15 is set.

Register Usage: All registers may be affected

name PERFORM procedure

name PM procedure

PERFORM and PM are identical macros.

Generate a branch to a local procedure with base addressability.

Uses MVC and B if SUBENTRY RENT=NO or push/pop stack if

RENT=YES.

Register Usage:

R14=Return address

R15=Linkage register

name PENTRY

Define local procedure using name.

Generates an entrypoint for a local procedure preceded with a branch instruction if SUBENTRY RENT=NO

name PEXIT

Branch to last caller of local procedure.

Generate branch to last PENTRY name-4 if SUBENTRY RENT=NO or generate decrement stack pointer, load, and branch if

RENT=YES.

Register Usage:

R14=Stack address

R15=Saved linkage register

name EXIT

Returns immediately to the last caller.

No registers are restored.

Use of SUBEXIT is preferred

Register Usage: No registers affected

name EOJ RC=returncode

name EOJ RC=(reg)

VSE only.

name is optional.

RC will return the value in GR15, zero is the default.

Returns immediately to the last caller.

ZDOCPROG.TXT

Register Usage:

R15= Return code

6) CALL, SAVE and RETURN

a) CALL (list form)

name CALL ,(parm1,parm2,...),vl,MF=L

Generates a parameter list for use with the execute form of CALL. eg. name DC A(parm1,parm2...)

(parm1,parm2,...)

The parameters can be anything that is permitted in an A-type constant. Note that register forms like (R5) are not interpreted as general registers, but as constants.

vl is optional

If the called program can accept a variable parameter list, then VL will turn on the senior bit (bit 0) of the last parameter.

b) CALL (execute form)

name CALL routine,(parm1,parm2,...),vl,LINKINST=,MF=(E,parms)

name CALL (reg),(parm1,parm2,...),vl,MF=(E,parms)

Provides a standard internal or external subroutine call. Parameters are addressed by GR1 and linkage by GR14.

routine

If a label, it can be internal (resolved at assembly time) or external (loaded and resolved by the linkage editor).

If the routine is in register notation, it can be internal or separately loaded.

(parm1,parm2,...)

Modify a fixed or variable parameter list to be accessed by the called program.

The parameter list must have initially been defined using the list form of the CALL. The parameters specified here will overlay that parameter list.

It is important that the number of parameters specified here do not exceed those specified in the list form of the CALL.

ZDOCPRG.TXT

The parameters can be anything that is permitted in an A-type constant. Any parameters bounded by parentheses, eg. (R5) are assumed to be registers or register equates. The content of each register (fullword only) is stored at the parameter location.

vl is optional

If the called program can accept a variable parameter list, then VL will turn on the senior bit (bit 0) of the last parameter. Note: This is the last parameter in the modified parameter list.

LINKINST= is optional

Determines the calling instruction.
Choose BALR (default) or BASR.

MF=(E,label)

MF=(E,(reg))

The label or register points to a parameter list previously defined with the list form of the CALL.

eg.

Call subroutine MYSUBR, replace the two parameters and mark the last parameter.

```
MYCALL CALL MYSUBR,(8,MYDATA),VL,MF=(E,PARMS)
```

```
...
```

```
PARMS CALL ,(7,OLDDATA),VL,MF=L
```

c) CALL (standard form)

```
name CALL routine,(parm1,parm2,...),vl,LINKINST=,MF=I
```

```
name CALL (reg),(parm1,parm2,...),vl,LINKINST=,MF=I
```

Provides a standard internal or external subroutine call.

Parameters are addressed by GR1 and linkage by GR14.

routine

If a label, it can be internal (resolved at assembly time) or external (loaded and resolved by the linkage editor).

If the routine is in register notation, it can be internal or separately loaded.

(parm1,parm2,...) is optional

Pass a fixed or variable parameter list to the called program.

ZDOCPRG.TXT

The parameters can be anything that is permitted in an A-type constant. Any parameters bounded by parentheses, eg. (R5) are assumed to be registers or register equates. The content of each register (fullword only) is stored at the parameter location.

vl is optional

If the called program can accept a variable parameter list, then VL will turn on the senior bit (bit 0) of the last parameter.

LINKINST= is optional

Determines the calling instruction.
Choose BALR (default) or BASR.

eg.

Call subroutine MYSUBR, pass two parameters and mark the last parameter.

```
MYCALL CALL MYSUBR,(8,MYDATA),VL
```

Register Usage:

R0 = indirect parameter list
R1 = parameter list
R14= linkage
R15= program location

name SAVE (fromreg,toreg)

Saves the specified register range in the savearea pointed to by GR13. The registers are saved in their conventional positions.

Register Usage: No registers affected

name RETURN (fromreg,toreg),flag,RC=

Restores the specified register range from the savearea pointed to by GR13. The registers are restored from their conventional positions.

Return is by the restored GR14.

flag is optional

T specifies that the byte at savearea+15 has the junior bit (bit 7) turned on to indicate a return to a called program. This bit (rightmost bit of saved GR14) is set after GR14 has

ZDOCPROG.TXT

been loaded with the return address.

RC=nn

RC=(reg)

If RC is omitted, GR15 is assumed to contain the return code.
GR15 is loaded with this return code before returning via

GR14.

RC may have a numeric value or the value may be in GRreg.

eg.

MYRET RETURN (14,12),T,RC=12

Restore registers 14 through 12. After the register restore,

flag

the savearea to indicate return to caller and set return code to 12.

Register Usage:

R15= Return code

All registers in the range fromreg-toreg

7) Other macros

PSAD

Provides a DSECT for the limited fields available in the first 8K of memory (PSA). The CVT may be addressed from here.

ZCVTD

Provides a DSECT for the limited fields available in the ZCVT. This follows the PSA and may be addressed as follows:

```
L    reg,ZCVT
USING IHAZCVT,reg
...
ZCVTD
```

CVTD

Provides a DSECT for the limited fields available in the Common Vector Table. This may be addressed as follows:

```
L    reg,X'10'
USING IHACVT,reg
...
CVTD
```

EQUIREGS REGS=option,TYPE=option

ZDOCPRG.TXT

YREGS is identical to EQUREGS with default parameters.

Generates standard equates for the general or floating point registers.

eg.

EQUREGS (defaults to REGS=GPR,TYPE=DEC)

R0 EQU 0

...

R15 EQU 15

EQUREGS TYPE=HEX

R0 EQU 0

...

RF EQU 15

EQUREGS REGS=FPR

F0 EQU 0

...

F15 EQU 15

EQUREGS REGS=FPR,TYPE=HEX

F0 EQU 0

...

FF EQU 15

Change Summary

November 24, 2008

LOAD ERRET= added

RESTORE RC= added

ESTAD replaced with IHASDWA, ESTAE updated

EPIED replaced with IHAEPIE, ESPIE updated

June 27, 2008

Correction to CALL MF=

January 18, 2008

Addedabend and return code sections.

September 28, 2007

Added VSE Macros CDDELETE, CDLOAD, EOJ

Incorporated Don's doc for PERFORM, PM, PENTRY and PEXIT

Revision of SUBENTRY

Correction to ESTAE failing instruction retry

ZDOCPROG.TXT

July 10, 2007

All macros now list possible general register usage
Complete revision of SUBENTRY
Update to SUBEXIT
Update to PARAM=
Storage dumped for SNAP and ABEND
Addition of SNAP ID=(reg) and TEXT=(reg)
Addition of RETURN RC=(reg)
Addition of EXIT macro
Complete revision of ESTAE
Note about ESTAEX

March 13, 2006

The section on SYSPARM and PARM formats updated by Don Higgins

Author: Melvyn Maltz

Publication date: November 24, 2008

Z390 version: V1.4.04

→