
RVEC Parameters and DMSTK Format

A Telecom Computer Security Bulletin File

RVEC Parameters

The commands RESTOR, RESUME, SAVE, PM, and START process a group of optional parameters associated with the PRIMOS RVEC vector. These parameters are stored on disk for every runfile (executable program).

Initial values for the RVEC parameters are usually specified in the PRIMOS SAVE command, or by LOADer's or SEG's SAVE command when the program was stored on disk.

Each parameter is a 16-bit processor word, represented by up to six octal digits.

Parameter	Memory Location	Definition
SA	--	Starting Address (first memory word used by program)
EA	--	Ending Address (last memory word used by program)
PC	7	P Register (Program Counter)
A	1	A Register (Arithmetic)
B	2	B Register (Arithmetic)
X	0	Index Register
Keys	--	Status keys associated with INK, OTK instructions

The RVEC parameters are optional in the command string. Any item that is specified replaces the previous value in RVEC, which is saved with the program. Thus, for any parameters that are not specified, the value previously stored in RVEC is saved with the program

Slash convention: An ordinal value followed by a slash and a value can be used to set a selected octal parameter without setting other octal parameters. For example, given the command format:

RESUME pathname [pc] [a] [b] [x] [keys]

the command:

R FILNAM 2/1000

sets the value of the RVEC parameter, B (ie, skip two octal parameters and then set the third to '1000').

Supplying RVEC parameters: RVEC parameters specified in RESUME or START commands replace the previous values in RVEC. Also, when a program returns to PRIMOS through the EXIT subroutine, RVEC is loaded from the processor values in effect at the time of the exit. Only the SAVE command alters the values of RVEC stored on disk with the program.

RESTOR returns a program from disk to memory and loads the SAVE parameters into RVEC in preparation for a START command.

RESUME combines the functions of RESTOR and START.

PM lists the current values of the RVEC parameters.

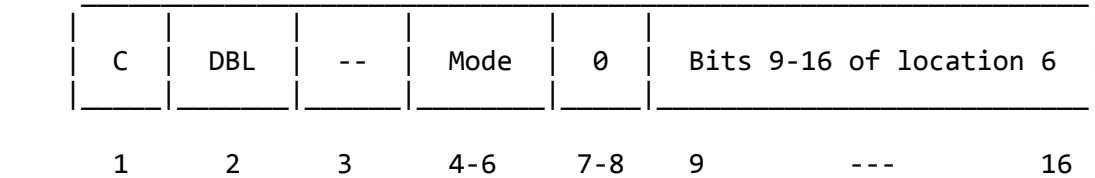
External commands have RVEC parameters that can be modified at the time the command is started (eg, PMA filename 1/740). Providing RVEC parameters to a command that does not need them will cause unpredictable results.

Keys

The item, keys, when specified among RVEC parameters, refers to the processor status keys handled by the INK and OTK instructions (refer to the System Architecture Reference Guide). These are represented by a single 16-bit word in one of the following formats (S-mode and R-mode programs use the first format; V-mode and I-mode programs use the second).

Keys (SR)

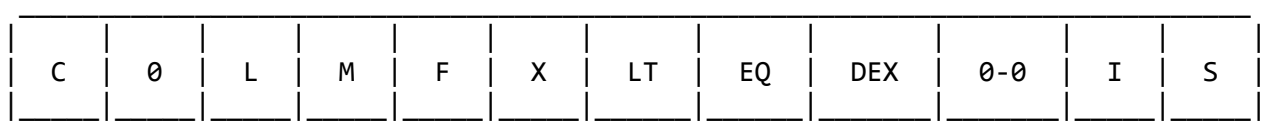
Process status information is available in a word called the keys, which can be read or set by the program. It's format is as follows:



C (Bit 1) Set by arithmetic error conditions
DBL (Bit 2) 0 - Single Precision, 1 - Double Precision

001 32S

Process status information is available in a 16-bit register known as the keys. It may be referenced by the LPSW, TKA, and TAK instructions.



C (Bit 1) C-bit
L (Bit 3) L-bit

MODE (Bits 4-6) Addressing Mode:

000	16S
001	32S
011	32R
010	64R
110	64V
100	32I

F (Bit 7) Floating point exception disable:

0	take fault
1	set C-bit

X (Bit 8) Integer exception enable:

0	set C-bit
1	take fault

LT (Bit 9) Condition code bits:

EQ (Bit 10)

LT	set if result is negative
EQ	set if result is zero

DEX (Bit 11) Decimal exception enable:

0	set C-bit
1	take fault

I (Bit 15)	In dispatcher -- set/cleared only by process exchange
S (Bit 16)	Save done -- set/cleared only by process exchange

C-bit (VI): Set by error conditions in arithmetic operations and by shifts.

L-bit (VI): Set by an arithmetic or shift operation except IRS, IRX, DRX. Equal to carry out of the most significant bit (Bit 1) of an arithmetic operation. It is valuable for simulating multiple-precision operations and for performing unsigned comparisons following a CAS or a SUB.

Condition code bits (VI): The two condition code bits are designated "EQ" and

DMSTK Format

The DMSTK command traces the sequence of calls and returns by which the user's process arrived at its current state. Machine states for internal commands, condition frames, and fault frames are preserved on the user's command stack. In addition, the most recent activation of a static mode program or dump on the terminal or into a COMOUTPUT file. As it is an internal command, it does not overwrite the static mode stack, and so does not preclude re-entry into the faulting program.

The DMSTK command has several options. These options may be used in any order aout in octal. The command syntax is as follows:

Option	Definition
	done in full format (since "full format" is the default, there with the most recent condition frame (if there is one) or with vation (ie, frame) that is dumped is to be produced.

DMSTK lists each stack frame in the following general format (for an explanation of the registers and the rings involved, see the System Architecture Reference Guide):

```
(nn) offset: Owner= procname (LB= ownerlb).
       Called from pcl_addr; returns to return_addr.
```

The information is as follows:

Argument	Definition

nn	Frame index number of the stack frame
offset	The word number in the current stack segment where this activation's stack frame begins
procname	The name (if available) of the procedure that owns this stack frame
ownerlb	The stack value of the LB (linkage base) register belonging to the procedure that owns the stack frame
pcl_addr	Address of the PCL instruction that caused the procedure to be invoked
return_addr	The address to which the procedure will return

If the frame is a fault frame, the following format is used:

```
   (nn) offset: FAULT FRAME; fault type = fault type.
          Fault returns to ret_pb; LB= faulter_lb, keys= faulter_keys.
          Fault code= fcode; fault addr= faddr.
          Registers at time of fault:
               000001 000002 000003 000004 000005 000006
               000007 000010 000011 000012 000013 000014
               000015 000016 000017 000020 000021 000022
               000023 000024 000025 000026 000027 000030
```

Argument	Definition
fault-type	Location in the fault table of the type of fault that occurred
ret_pb	Address to which the fault returns
faulter_lb	LB register belonging to the procedure in which the fault occurred
faulter_keys	CPU keys at the time of the fault
register data	If present, a direct dump of the register save area (in the same format as that produced by the CPU RSAV instruction)
fcode	Fault code generated by this particular fault
faddr	Fault address generated by this particular fault

If the activation is a condition frame, the following format is used:

```
   (nn) offset: CONDITION FRAME for "condition_name"; returns to ret_pb.
          Condition raised at sigloc; LB= siglb; keys= sigkeys.
          [(Crawlout to outerpb; LB= outerlb; keys= outerkeys.)]
          [Registers at time of fault in inner ring:
                Save Mask= ssssss; XB= xb_value
                 000001 000002 000003 000004 000005 000006
                 000007 000010 000011 000012 000013 000014
```

000015 000016 000017 000020 000021 000022
000023 000024 000025 000026 000027 000030]

The latter two frames are displayed only if the condition was signalled in an inner ring and subsequently a crawlout to the current ring occurred.

If, during the trace, the stack switches to a different segment, DMSTK will print, "STACK SEGMENT IS xxxx", giving the octal segment number of the new stack segment.

Note

A called-from or return-to value such as 0(0)/0 or 0(0)/177776 usually means that the stack frame has an invalid return point and can never return. An example of such a frame is the first frame set up by SEG in a V-mode Static Mode program.