F88 Schedule

	Monday	Tuesday	Wednesday	Thursday	Friday
Morning	Intro What is a crash? What is a dump?	Using PL/I listings Misc. commands Workshop	CPU Registers Data Formats ALM Intro.	Workshop HREGS Logs Locking	Recovery procedures Misc Workshop
After-noon	Fault/Int intro. How Mx crashes Stacks & linkage	analyze_ multics Workshop	Fault processng Signallng, crawlouts Types of Faults	AMs Metho-dology Workshop	Workshop

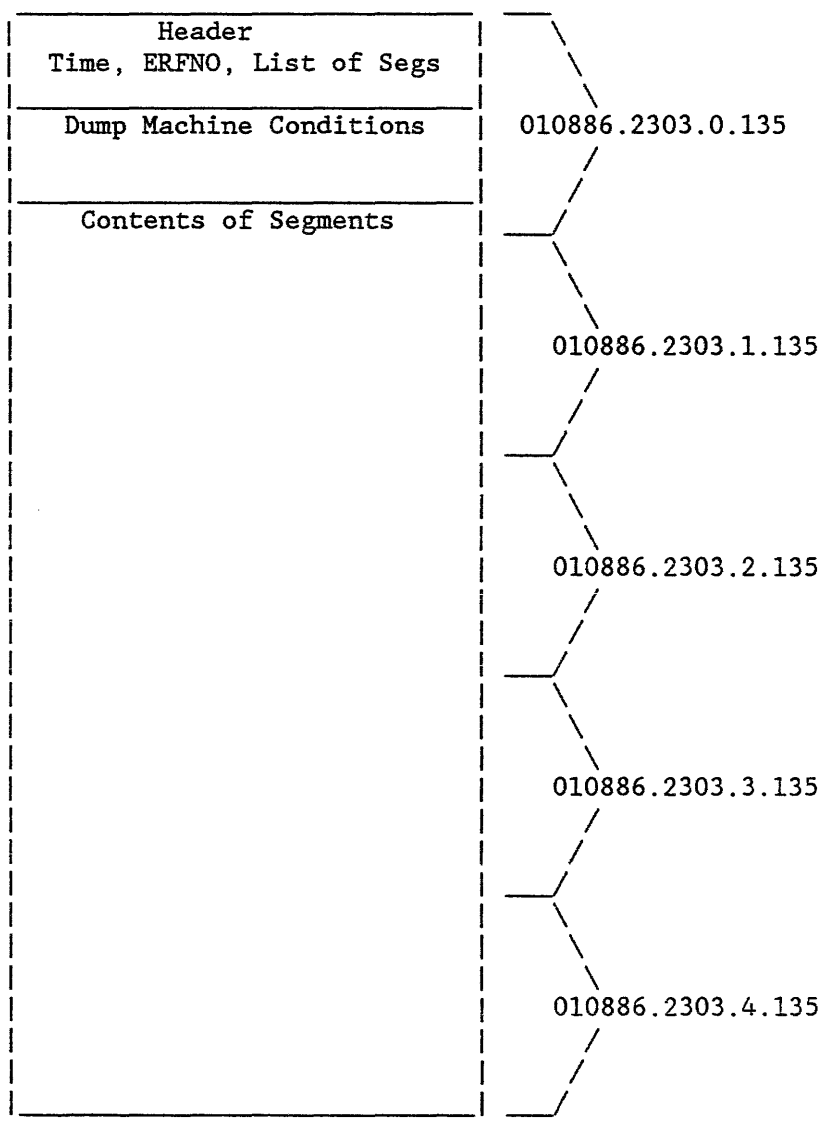
F88 Notebook Contents

CONTENTS OF DUMP 135

BCE dump command: dump -run hc pp moddir -elig hc stk -inzr hc pp

Stored by copy_dump in segments:

>dumps>010886.2303.0.135
>dumps>010886.2303.1.135
>dumps>010886.2303.2.135
>dumps>010886.2303.3.135
>dumps>010886.2303.4.135



```
Process   0, Initializer.SysDaemon.z, running (-inzr hc pp)

Seg#     Len      Name

0       3072      dseg
1       1024      bos_toehold
2       4096      config_deck
3       2048      dn355_mailbox
4       1024      fault_vector
5       1024      flagbox
6       9216      name_table
7       2048      slt
10      2048      toehold_data
11      2048      iom_mailbox
12      1024      unpaged_page_tables
13      2048      toehold
14      1024      breakpoint_page
15      1024      lot
16     17408      as_linkage
17      5120      ws_linkage
20     79872      definitions_
22         0      abs_seg1
25      2048      bound_hc_data_wired
52     50176      core_map
54      1024      disk_post_queue_seg
55     29696      disk_seg
56      1024      dn355_data
60      2048      emergency_shutdown
62      5120      idle_dsegs
63      5120      idle_pdses
64      2048      init_processor
65      8192      inzr_stk0
66     18432      ioi_abs_seg
67      6144      io_config_data
70     11264      ioi_data
71      3072      iom_data
72      8192      kst_seg
73      1024      oc_data
74      2048      pds
75      7168      prds
76      3072      pvt
77         0      rdisk_seg
102    27648      scas
104    55296      sst_names_
105   256000      sst_seg
106     1024      stack_0_data
107    10240      stock_seg
110     1024      sys_info
111     1024      syserr_data
112     1024      syserr_log_data
113    64512      syserr_log_laurel
114    10240      syserr_log_hardy
115    18432      tc_data
116     1024      active_all_rings_data
```

```
117     3072      active_hardcore_data
121     9216      ast_lock_meter_seg
143    34816      dbm_seg
144     3072      dir_lock_seg
146     2048      dm_journal_seg_
147        0      fnp_dump_seg
160     3072      io_page_table_seg
161        0      ioat
163     1024      lvt
175        0      salv_dir_space
176     1024      salv_data
177        0      salv_temp_dir
200   134144      scavenger_data
201    65536      str_seg
202     1024      syserr_daemon_dseg
203     2048      syserr_daemon_pds
204     2048      syserr_daemon_stack
210     2048      template_pds
211    63488      tty_area
212    77824      tty_buf
213     2048      tty_tables
214     6144      vtoc_buffer_seg
221    13312      >pdd> zzzzzzzbBBBBBB>stack_1
224    19456      >pdd> zzzzzzzbBBBBBB>stack_4
231    14336      >pdd
233     2048      >pdd> zzzzzzzbBBBBBB
273    14336      >pdd> zzzzzzzbBBBBBB> BBBJPMnkNPCxqj.area.linker
323    46080      >pdd> zzzzzzzbBBBBBB> BBBJPMnkPBlmHx.area.linker
331     4096      >pdd> zzzzzzzbBBBBBB> BBBJPMnkPFQWgB
354     1024      >pdd> zzzzzzzbBBBBBB>pit_temp_
457    11264      >pdd> zzzzzzzbBBBBBB> BBBJPMnkWJXXxF.area.ipc
465        0      >pdd> zzzzzzzbBBBBBB> BBBJPMnkWQkGkM.temp.0465
520     2048      >pdd> BLBDXhpbBBBBBB
563     2048      >pdd> BMLDXhxBBBBBB
565     1024      >pdd> BNBDXhzbBBBBBB
                  .
                  .
                  .
2014    2048      >pdd> BxBDbdFbBBBBBB
2015    2048      >pdd> CZLDbcdBBBBBB
2017    2048      >pdd> CXbDbbnbBBBBBB
2023    2048      >pdd> BmLDbcJBBBBBB
```

```
Process   1, Idle.SysControl.a, ready (-elig hc stk)

Seg#     Len      Name

0       1024      dseg
72      8192      kst_seg
74      1024      pds
75      8192      prds

Process   2, Idle.SysControl.b, ready (-elig hc stk)

Seg#     Len      Name

0       1024      dseg
72      8192      kst_seg
74      1024      pds
75      7168      prds

Process   3, Idle.SysControl.c, ready (-elig hc stk)

Seg#     Len      Name

0       1024      dseg
72      8192      kst_seg
74      1024      pds
75      3072      prds

Process   4, Idle.SysControl.d, ready (-elig hc stk)

Seg#     Len      Name

0       1024      dseg
72      8192      kst_seg
74      1024      pds
75      7168      prds

Process   5, Idle.SysControl.e, ready (-elig hc stk)

Seg#     Len      Name

0       1024      dseg
72      8192      kst_seg
74      1024      pds
75      7168      prds
```

```
Process   6, Demers.Flower.a, running (-run hc pp moddir)

Seg#     Len      Name

0       1024      dseg
72      1024      kst_seg
74      2048      pds
75      8192      prds
220    11264      >sl1>stack_0.035
221     3072      >pdd> BLbDbdKBBBBBBB>stack_1
222     3072      >pdd> BLbDbdKBBBBBBB>stack_2
224     7168      >pdd> BLbDbdKBBBBBBB>stack_4
237    10240      >pdd> BLbDbdKBBBBBBB> BBBJPNBqbbZbGF.area.linker
254     1024      >pdd> BLbDbdKBBBBBBB>pit
317     2048      >pdd> BLbDbdKBBBBBBB>process_search_segment_.4
330     5120      >pdd> BLbDbdKBBBBBBB> BBBJPNBqbjdQZd.area.linker
332     5120      >pdd> BLbDbdKBBBBBBB> BBBJPNBqbkFBCb.area.linker

Process   7,   Spratt.Multics.m, running (-run hc pp moddir)

Seg#     Len      Name

0       1024      dseg
72      2048      kst_seg
74      2048      pds
75      7168      prds
220    12288      >sl1>stack_0.016
224    38912      >pdd> BdLDbbjBBBBBBB>stack_4
237   191488      >pdd> BdLDbbjBBBBBBB> BBBJPNBmfbjNjK.area.linker
304     5120      >pdd> BdLDbbjBBBBBBB> BBBJPNBmfgJMCb.temp.0304
332     6144      >pdd> BdLDbbjBBBBBBB> BBBJPNBmfkWPFh.area.linker
340     5120      >pdd> BdLDbbjBBBBBBB> BBBJPNBmflwdLH.area.linker
347     2048      >pdd> BdLDbbjBBBBBBB>process_search_segment_.4
361     1024      >pdd> BdLDbbjBBBBBBB> BBBJPNBmfpWzmP.temp.0361
431     1024      >pdd> BdLDbbjBBBBBBB> BBBJPNBmgFcGQh.temp.0431
432     1024      >pdd> BdLDbbjBBBBBBB> BBBJPNBmgQmfch.temp.0432
433        0      >pdd> BdLDbbjBBBBBBB> BBBJPNBmgpLfKD.temp.0433

Process   8, Le.Mx.a, waiting (-elig hc stk)

Seg#     Len      Name

0       1024      dseg
72      2048      kst_seg
74      2048      pds
75      7168      prds
220    10240      >sl1>stack_0.022
221     3072      >pdd> BkBDbcWbBBBBBB>stack_1
222     4096      >pdd> BkBDbcWbBBBBBB>stack_2
224    41984      >pdd> BkBDbcWbBBBBBB>stack_4
```

04/05/85 dump

Syntax as a command:
```
dump {macro_keyword} {-process_group segment_option
   {...segment_options}} {-control_args}
```

Function: produces a diagnostic dump of system memory and tables after a hardware or software failure, for later analysis. The dump is produced by copying binary images of segments and directories into the DUMP partition of the disk described by the part dump config card. Arguments to the dump command specify which processes are to be examined and which segments from those processes are to be dumped. (See "Notes" for a general purpose command line.) This command is valid at all BCE command levels.

Arguments:

macro_keyword
specifies one of the following default group of processes and segments to dump.

-brief, -bf
is equivalent to -run hc pp mod dir

-long, -lg
is equivalent to -all wrt

-standard, -std
is equivalent to -run hc pp mod dir -elig hc stk -inzr hc stk

process_group
specifies a group of processes to be considered for dumping. The segments that get dumped for processes in this group are specified by segment options that follow the process group keyword. Allowed groups are:

-all
all processes

-eligible, -elig
all running and eligible processes (processes being considered for running)

-initializer, -inzr
the initializer process (first apte entry)

-running, -run
processes running on a processor (apte.state = running or stopped)

segment_option
specifies a class of segments to be dumped for the group of processes specified by the process group keyword. Segment classes are:

directories, dir
directory segments (aste.dirsw = "1"b)

hardcore, hc
the pds, kst, dseg and ring 0 stack for the process(es). If a process is running, this also dumps the prds for the processor in question.
modifying_dirs, moddir
directory segments (aste.dirsw = "1"b) which were being modified at the time of the crash (dir.modify ^= "0"b)

per_process, pp
the segments contained within the process directory of the process(es) (aste.per_process = "1"b)
stacks, stk
all stack segments in the process(es) not already dumped by the hc or pp keywords.
writeable, wrt
all segments to which the process(es) have write access. This keyword produces a very large dump.

Writable ring zero segments (system data bases) other than directories are dumped regardless of what keywords are specified.

Prefixing a segment option with a circumflex (^) reverts an earlier occurence of the given segment option. Thus, you can turn on a macro_keyword and turn off a specific segment option within it.

Control arguments:
-bce
dumps BCE itself (the dumper).
-crash
specifies that BCE is to dump the saved Multics image.
-drive, -dv drive_name
places the dump into the dump partition of the specified drive instead of the drive listed on the PART DUMP card.
-dump #
changes the dump number to a desired value. By default, dumps are assigned numbers sequentially.

-force, -fc
places the dump into the DUMP partition without querying you first, even if this means that an existing dump which hasn't been copied will be overwritten. If this control argument is not used, the dump command asks you if the existing dump should really be overwritten before it overwrites it.
-no_sstnt
disables sst_names_ generation. If sst_names_ generation is enabled for the system (by the astk parm in the config deck), this control argument has no effect.

-sstnt
causes the segment sst_names_ (the sst name table) to be filled in

and included in the dump. The segment sst_names_ provides a name for each ASTE in the system. This information is of use to dump analysis programs. If sst_names_ generation is enabled for the system (by the astk parm in the config deck), this control argument has no effect. This is the default.

Notes: For general purpose dump analysis, the command line:

```
dump -std
```

which is equivalent to

```
dump -run hc pp mod dir -elig hc stk -inzr hc stk
```

should give the user all of the useful processes and segments (to produce a smaller dump, remove the "mod dir" keyword). For simplicity, and to remove the possibility of operator error, this command line should be placed into a BCE exec_com, either by itself or in a site supplied crash exec_com.

The dump command examines the active process table entries (apte) within the specified image. For each entry, the criteria specified through the keywords are used to decide if any segments from this process are to be dumped. If any segments are to be dumped, the segment options are applied to each segment active within that process to decide whether or not they should be dumped. As each process is dumped, the dump command will produce an output line showing the apte number and the dbr value for the process. After scanning all apte entries, if the process in control when Multics crashed was not one of the processes dumped, it is dumped with a status line showing an apte number of zero. This process is dumped with the running and initializer segment options.

A counter and a valid flag are kept within the DUMP partition. When a dump is placed into the partition, the valid flag is set. It is reset when the dump is copied out during Multics service (by the copy_dump exec command). If the dump in the partition has not been copied, the dump command will ask you if it should be overwritten. You can avoid this query by specifying the -force (-fc) control argument.

The dump command provides a severity indicator, indicating the successful of its operation. This indicator may be obtained with the severity command/active function. The interpretation of the severity status is:

3 - the dump request was never called.
2 - the dump request was entered, but never completed.
1 - the dump was aborted because the DUMP partition contains an older dump.
0 - the dump was successfully generated.

MULTICS FAULT TYPES

#	Name
0	Shutdown
1	Store
2	MME 1
3	Fault Tag 1
4	Timer Runout
5	Command
6	Derail
7	Lockup
8	Connect
9	Parity
10	Illegal Procedure
11	Op Not Complete
12	Startup
13	Overflow
14	Divide Check
15	Execute
16	(DF0) Segment
17	(DF1) Page
18	Directed Fault 2
19	Directed Fault 3
20	Access Violation
21	MME 2
22	MME 3
23	MME 4
24	(FT2) Linkage
25	Fault Tag 3
26	Unassigned
27	Unassigned
28	Unassigned
29	Unassigned
30	Unassigned
31	Trouble

FAULTS BY CATEGORY

FAULTS THAT ALWAYS CRASH SYSTEM

#	Fault Name
26	Unassigned
27	Unassigned
28	Unassigned
29	Unassigned
30	Unassigned
15	Execute

FAULTS USED INTERNALLY BY SUPERVISOR

#	Fault Name
4	Timer Runout
8	Connect
20	Access Violation (Ring Alarm)

IMPLICIT REQUESTS FOR SUPERVISOR SERVICES

#	Fault Name
17	(DF1) Page
16	(DF0) Segment
24	(FT2) Linkage
20	Access Violation (boundsfault, etc.)

FAULTS THAT ALWAYS INDICATE HARDWARE PROBLEMS

#	Fault Name
0	Shutdown
11	Op Not Complete
12	Startup
9	Parity
1	Store

FAULTS THAT ALWAYS INDICATE HARDWARE OR SOFTWARE PROBLEMS

#	Fault Name
31	Trouble
18	Directed Fault 2
19	Directed Fault 3

FAULTS THAT CAN BE GENERATED BY USER

#	Fault Name
20	Access Violation
5	Command
6	Derail
2	MME 1
3	Fault Tag 1
7	Lockup
10	Illegal Procedure
13	Overflow
14	Divide Check
21	MME 2
22	MME 3
23	MME 4
25	Fault Tag 3

FAULT VECTOR IN NUMERICAL ORDER

#	Fault Name	SCU stored at	Handler
0	Shutdown	scu 500,* -> pds$fim_data (70|60)	tra 400,* -> fim$onc_start_shut_entry (34|14)
1	Store	scu 502,* -> pds$signal_data (70|140)	tra 402,* -> fim$signal_entry (34|300)
2	MME 1	scu 504,* -> pds$signal_data (70|140)	tra 404,* -> fim$signal_entry (34|300)
3	Fault Tag 1	scu 506,* -> pds$signal_data (70|140)	tra 406,* -> fim$signal_entry (34|300)
4	Timer Runout	scu 510,* -> prds$fim_data (71|160)	tra 410,* -> wired_fim$timer_runout (34|2324)
5	Command	scu 512,* -> pds$fim_data (70|60)	tra 412,* -> fim$primary_fault_entry (34|404)
6	Derail	scu 514,* -> pds$signal_data (70|140)	tra 414,* -> fim$drl_entry (34|30)
7	Lockup	scu 516,* -> pds$signal_data (70|140)	tra 416,* -> fim$signal_entry (34|300)
8	Connect	scu 520,* -> prds$fim_data (71|160)	tra 420,* -> prds$fast_connect_code (71|1054)
9	Parity	scu 522,* -> pds$fim_data (70|60)	tra 422,* -> fim$parity_entry (34|124)
10	Illegal Procedure	scu 524,* -> pds$signal_data (70|140)	tra 424,* -> fim$signal_entry (34|300)
11	Op Not Complete	scu 526,* -> pds$fim_data (70|60)	tra 426,* -> fim$onc_start_shut_entry (34|14)
12	Startup	scu 530,* -> pds$fim_data (70|60)	tra 430,* -> fim$onc_start_shut_entry (34|14)
13	Overflow	scu 532,* -> pds$signal_data (70|140)	tra 432,* -> fim$signal_entry (34|300)
14	Divide Check	scu 534,* -> pds$signal_data (70|140)	tra 434,* -> fim$signal_entry (34|300)
15	Execute	scu 536,* -> prds$sys_trouble_data (71|240)	tra 436,* -> wired_fim$xec_fault (34|2274)
16	(DF0) Segment	scu 540,* -> pds$fim_data (70|60)	tra 440,* -> fim$primary_fault_entry (34|404)
17	(DF1) Page	scu 542,* -> pds$page_fault_data (70|0)	tra 442,* -> page_fault$page_fault (41|1062)
18	Directed Fault 2	scu 544,* -> pds$signal_data (70|140)	tra 444,* -> fim$signal_entry (34|300)
19	Directed Fault 3	scu 546,* -> pds$signal_data (70|140)	tra 446,* -> fim$signal_entry (34|300)
20	Access Violation	scu 550,* -> pds$fim_data (70|60)	tra 450,* -> fim$access_violation_entry (34|0)
21	MME 2	scu 552,* -> pds$signal_data (70|140)	tra 452,* -> fim$signal_entry (34|300)
22	MME 3	scu 554,* -> pds$signal_data (70|140)	tra 454,* -> fim$signal_entry (34|300)
23	MME 4	scu 556,* -> pds$signal_data (70|140)	tra 456,* -> fim$signal_entry (34|300)
24	(FT2) Linkage	scu 560,* -> pds$fim_data (70|60)	tra 460,* -> fim$primary_fault_entry (34|404)
25	Fault Tag 3	scu 562,* -> pds$signal_data (70|140)	tra 462,* -> fim$signal_entry (34|300)
26	Unassigned	scu 564,* -> prds$sys_trouble_data (71|240)	tra 464,* -> wired_fim$unexp_fault (34|2310)
27	Unassigned	scu 566,* -> prds$sys_trouble_data (71|240)	tra 466,* -> wired_fim$unexp_fault (34|2310)
28	Unassigned	scu 570,* -> prds$sys_trouble_data (71|240)	tra 470,* -> wired_fim$unexp_fault (34|2310)
29	Unassigned	scu 572,* -> prds$sys_trouble_data (71|240)	tra 472,* -> wired_fim$unexp_fault (34|2310)
30	Unassigned	scu 574,* -> prds$sys_trouble_data (71|240)	tra 474,* -> wired_fim$unexp_fault (34|2310)
31	Trouble	scu 576,* -> pds$fim_data (70|60)	tra 476,* -> fim$primary_fault_entry (34|404)

FAULT VECTORY BY CATEGORY

FAULTS THAT ALWAYS CRASH SYSTEM

#	Fault Name	SCU stored at	Handler
26	Unassigned	scu 564,* -> prds$sys_trouble_data (71|240)	tra 464,* -> wired_fim$unexp_fault (34|2310)
27	Unassigned	scu 566,* -> prds$sys_trouble_data (71|240)	tra 466,* -> wired_fim$unexp_fault (34|2310)
28	Unassigned	scu 570,* -> prds$sys_trouble_data (71|240)	tra 470,* -> wired_fim$unexp_fault (34|2310)
29	Unassigned	scu 572,* -> prds$sys_trouble_data (71|240)	tra 472,* -> wired_fim$unexp_fault (34|2310)
30	Unassigned	scu 574,* -> prds$sys_trouble_data (71|240)	tra 474,* -> wired_fim$unexp_fault (34|2310)
15	Execute	scu 536,* -> prds$sys_trouble_data (71|240)	tra 436,* -> wired_fim$xec_fault (34|2274)

FAULTS USED INTERNALLY BY SUPERVISOR

#	Fault Name	SCU stored at	Handler
4	Timer Runout	scu 510,* -> prds$fim_data (71|160)	tra 410,* -> wired_fim$timer_runout (34|2324)
8	Connect	scu 520,* -> prds$fim_data (71|160)	tra 420,* -> prds$fast_connect_code (71|1054)
20	Access Violation (Ring Alarm)	scu 550,* -> pds$fim_data (70|60)	tra 450,* -> fim$access_violation_entry (34|0)

IMPLICIT REQUESTS FOR SUPERVISOR SERVICES

#	Fault Name	SCU stored at	Handler
17	(DF1) Page	scu 542,* -> pds$page_fault_data (70|0)	tra 442,* -> page_fault$page_fault (44|1062)
16	(DF0) Segment	scu 540,* -> pds$fim_data (70|60)	tra 440,* -> fim$primary_fault_entry (34|404)
24	(FT2) Linkage	scu 560,* -> pds$fim_data (70|60)	tra 460,* -> fim$primary_fault_entry (34|404)
20	Access Violation (boundsfault, e.g.)	scu 550,* -> pds$fim_data (70|60)	tra 450,* -> fim$access_violation_entry (34|0)

FAULTS THAT ALWAYS INDICATE HARDWARE PROBLEMS

#	Fault Name	SCU stored at	Handler
0	Shutdown	scu 500,* -> pds$fim_data (70|60)	tra 400,* -> fim$onc_start_shut_entry (34|14)
11	Op Not Complete	scu 526,* -> pds$fim_data (70|60)	tra 426,* -> fim$onc_start_shut_entry (34|14)
12	Startup	scu 530,* -> pds$fim_data (70|60)	tra 430,* -> fim$onc_start_shut_entry (34|14)
9	Parity	scu 522,* -> pds$fim_data (70|60)	tra 422,* -> fim$parity_entry (34|124)
1	Store	scu 502,* -> pds$signal_data (70|140)	tra 402,* -> fim$signal_entry (34|300)

FAULTS THAT ALWAYS INDICATE HARDWARE OR SOFTWARE PROBLEMS

#	Fault Name	SCU stored at	Handler
31	Trouble	scu 576,* -> pds$fim_data (70|60)	tra 476,* -> fim$primary_fault_entry (34|404)
18	Directed Fault 2	scu 544,* -> pds$signal_data (70|140)	tra 444,* -> fim$signal_entry (34|300)
19	Directed Fault 3	scu 546,* -> pds$signal_data (70|140)	tra 446,* -> fim$signal_entry (34|300)

FAULTS THAT CAN BE GENERATED BY USER

#	Fault Name	SCU stored at	Handler
20	Access Violation	scu 550,* -> pds$fim_data (70|60)	tra 450,* -> fim$access_violation_entry (34|0)
5	Command	scu 512,* -> pds$fim_data (70|60)	tra 412,* -> fim$primary_fault_entry (34|404)
6	Derail	scu 514,* -> pds$signal_data (70|140)	tra 414,* -> fim$drl_entry (34|30)
2	MME 1	scu 504,* -> pds$signal_data (70|140)	tra 404,* -> fim$signal_entry (34|300)
3	Fault Tag 1	scu 506,* -> pds$signal_data (70|140)	tra 406,* -> fim$signal_entry (34|300)
7	Lockup	scu 516,* -> pds$signal_data (70|140)	tra 416,* -> fim$signal_entry (34|300)
10	Illegal Procedure	scu 524,* -> pds$signal_data (70|140)	tra 424,* -> fim$signal_entry (34|300)
13	Overflow	scu 532,* -> pds$signal_data (70|140)	tra 432,* -> fim$signal_entry (34|300)
14	Divide Check	scu 534,* -> pds$signal_data (70|140)	tra 434,* -> fim$signal_entry (34|300)
21	MME 2	scu 552,* -> pds$signal_data (70|140)	tra 452,* -> fim$signal_entry (34|300)
22	MME 3	scu 554,* -> pds$signal_data (70|140)	tra 454,* -> fim$signal_entry (34|300)
23	MME 4	scu 556,* -> pds$signal_data (70|140)	tra 456,* -> fim$signal_entry (34|300)
25	Fault Tag 3	scu 562,* -> pds$signal_data (70|140)	tra 462,* -> fim$signal_entry (34|300)

PROHIBITED FAULTS

Systrouble Code	Reason
1	Page fault while on prds
2	Fault/Interrupt while on prds
3	Fault in idle process
4	Fault/Interrupt with PTL set
5	Unrecognized fault
6	Unexpected fault
7	Execute fault by operator
8	Out-of-Segment-Bound on prds
9	Fault while in masked environment
10	Fault while in bound_interceptors
11	Ring 0 derail

Crash Sequence for Type 1 Crash: Prohibited Fault

CPU A (Bootload CPU)	CPU B	CPU C
.	.	.
.	.	.
.	.	.
.	<<FAULT>>	.
.		.
.	fault_vector_scu/tra	.
.		.
.	handler	.
.		.
.	fim_util$check_fault	.
.	scs$sys_trouble_pending=-N	.
.	scs$trouble_processid=	.
.	PROCESSID	.
.	cioc <CPU B>	.
.	STOP	.
.		.
.		.

Crash Sequence for Type 1 Crash: Prohibited Fault (continued)

CPU A (Bootload CPU)	CPU B	CPU C
.		.
.		.
.	<<CONNECT from CPU B >>	.
.		.
.	prds$fast_connect_code	.
.	test scs$sys_trouble_pending	.
.		.
.	wired_fim$connect_handler	.
.	store MCs in prds$fim_data	.
.		.
.	sys_trouble	.
.	if scs$trouble_flags = 0:	.
.	scs$trouble_flags =	.
.	scs$processor	.
.	cioc <all other CPUs>	.
.	copy MCs from prds$fim_data	.
<<CONNECT from CPU B>>	to prds$sys_trouble_data	.
	scs$processor bit 1 = 0	.
prds$fast_connect_code	scs$trouble_dbrs(1) = DBR	.
test scs$sys_trouble_pending	test scs$bos_processor_tag	.
	STOP	.
wired_fim$connect_handler		.
store MCs in prds$fim_data		.
		.
sys_trouble		.
test scs$trouble_flags		.
copy MCs from prds$fim_data		<<CONNECT from CPU B>>
to prds$sys_trouble_data		
scs$processor bit 0 = 0		prds$fast_connect_code
scs$trouble_dbrs(0) = DBR		test scs$sys_trouble_pending
test scs$bos_processor_tag		
inhibit lockup fault		wired_fim$connect_handler
loop until scs$processor=0		store MCs in prds$fim_data
|		
|		sys_trouble
|		test scs$trouble_flags
|		copy MCs from prds$fim_data
|		to prds$sys_trouble_data
|		scs$processor bit 2 = 0
if scs$sys_trouble_pending < 0:		scs$trouble_dbrs(2) = DBR
make flagbox message		test scs$bos_processor_tag
loop 1 second for all I/O		STOP
to complete		
change derail fault vector		
execute drl instruction		
BOS/BCE		
print flagbox message		

Typical Dump Events for Type 1 Crash: Prohibited Fault

Event	CPU	MC location	Context/significance of MCs
DRL	Bootload	Dump Header (DREGS)	sys_trouble|221
CON	Non-trouble	prds$fim_data & prds$sys_trouble_data	Where executing when told to stop
CON	Non-trouble	prds$fim_data & prds$sys_trouble_data	Where executing when told to stop
CON	Trouble	prds$fim_data & prds$sys_trouble_data	fim_util|72 (cioc/dis in check_fault). PR2 -> MCs for prohibited fault
Fault	Trouble	PR2 in prds$sys_trouble_data	Prohibited fault that caused crash
		.	
		.	
		.	
		.	
		.	

Crash Sequence for Type 2 Crash: Syserr

CPU A (Bootload CPU)	CPU B	CPU C
.	.	.
.	.	.
.	.	.
.	hardcore_program	.
.	call syserr with code 1	.
.		.
.	syserr	.
.	make sure stack is wired	.
.		.
.	syserr_real	.
.	print message on console	.
.	put message in syserr log	.
.		.
.	pmut$bce_and_return	.
.	scs$sys_trouble_pending=	.
.	scs$processor	.
.	scs$trouble_processid=	.
.	PROCESSID	.
.	cioc <CPU B>	.
.	STOP	.
.		.
.		.
.		.

Crash Sequence for Type 2 Crash: Syserr (continued)

CPU A (Bootload CPU)	CPU B	CPU C
.		.
.		.
.	<<CONNECT from CPU B>>	.
.		.
.	prds$fast_connect_code	.
.	test scs$sys_trouble_pending	.
.		.
.	wired_fim$connect_handler	.
.	store MCs in prds$fim_data	.
.		.
.	sys_trouble	.
.	if scs$trouble_flags = 0:	.
.	scs$trouble_flags =	.
.	scs$processor	.
.	cioc <all other CPUs>	.
.	copy MCs from prds$fim_data	.
<<CONNECT from CPU B	to prds$sys_trouble_data	.
	scs$processor bit 1 = 0	.
prds$fast_connect_code	scs$trouble_dbrs(1) = DBR	.
test scs$sys_trouble_pending	test scs$bos_processor_tag	.
	STOP	.
wired_fim$connect_handler		.
store MCs in prds$fim_data		.
		.
sys_trouble		.
test scs$trouble_flags		.
copy MCs from prds$fim_data		<<CONNECT from CPU B>>
to prds$sys_trouble_data		
scs$processor bit 0 = 0		prds$fast_connect_code
scs$trouble_dbrs(0) = DBR		test scs$sys_trouble_pending
test scs$bos_processor_tag		
inhibit lockup fault		wired_fim$connect_handler
loop until scs$processor=0		store MCs in prds$fim_data
|		
|		sys_trouble
|		test scs$trouble_flags
|		copy MCs from prds$fim_data
|		to prds$sys_trouble_data
|		scs$processor bit 2 = 0
test scs$sys_trouble_pending		scs$trouble_dbrs(2) = DBR
loop 1 second for all I/O		test scs$bos_processor_tag
to complete		STOP
change derail fault vector		
execute drl instruction		
BOS/BCE		

Typical Dump Events for Type 2 Crash: Syserr

Event	CPU	MC location	Context/significance of MCs
DRL	Bootload	Dump Header (DREGS)	sys_trouble|221
CON	Non-trouble	prds$fim_data & prds$sys_trouble_data	Where executing when told to stop
CON	Non-trouble	prds$fim_data & prds$sys_trouble_data	Where executing when told to stop
CON	Trouble	prds$fim_data & prds$sys_trouble_data	pmut|315 (cioc/dis in bce_and_return); PR6 -> stack where syserr called
Syserr Message			Multics not in operation
Syserr Message			Message from syserr call that crashed the system
		.	
		.	
		.	
		.	
		.	

Crash Sequence for Type 3 Crash: EXECUTE Fault

```
CPU A (Bootload CPU)              CPU B                          CPU C

        .                          .                              .
        .                          .                              .
        .                        <<EXECUTE FAULT>>                .
        .                                                         .
        .                        fault_vector scu/tra             .
        .                                                         .
        .                        wired_fim$xec_fault              .
        .                        store MCs in                     .
        .                          prds$sys_trouble_data          .
        .                        scs$sys_trouble_pending=-7       .
        .                                                         .
        .                        sys_trouble                      .
        .                        if scs$trouble_flags = 0:        .
        .                          scs$trouble_flags =            .
        .                            scs$processor                .
        .                          cioc <all other CPUs>          .
        .                        copy MCs from prds$fim_data      .
<<CONNECT from CPU B>>             to prds$sys_trouble_data       .
                                 scs$processor bit 1 = 0          .
prds$fast_connect_code           scs$trouble_dbrs(1) = DBR        .
test scs$sys_trouble_pending     test scs$bos_processor_tag       .
                                 STOP                             .
wired_fim$connect_handler                                         .
store MCs in prds$fim_data                                        .
                                                                  .
sys_trouble                                                       .
test scs$trouble_flags                                            .
copy MCs from prds$fim_data                    <<CONNECT from CPU B>>
  to prds$sys_trouble_data
scs$processor bit 0 = 0                        prds$fast_connect_code
scs$trouble_dbrs(0) = DBR                      test scs$sys_trouble_pending
test scs$bos_processor_tag
inhibit lockup fault                           wired_fim$connect_handler
loop until scs$processor=0                     store MCs in prds$fim_data
          |
          |                                    sys_trouble
          |                                    test scs$trouble_flags
          |                                    copy MCs from prds$fim_data
          |                                      to prds$sys_trouble_data
          |                                    scs$processor bit 2 = 0
test scs$sys_trouble_pending                   scs$trouble_dbrs(2) = DBR
   make flagbox message                        test scs$bos_processor_tag
loop 1 second for all I/O                      STOP
  to complete
change derail fault vector
execute drl instruction

BOS/BCE
  print flagbox message
```

Typical Dump Events for Type 3 Crash: EXECUTE Fault

DUMP EVENTS SEQUENCE:

Event	CPU	MC location	Context/significance of MCs
DRL	Bootload	Dump Header (DREGS)	sys_trouble|221
CON	Others	prds$fim_data & prds$sys_trouble_data	Where looping when told to stop; PR6 -> stack frame
CON	Others	prds$fim_data & prds$sys_trouble_data	Where looping when told to stop; PR6 -> stack frame
EXF	Button pushed	prds$sys_trouble_data	Where looping when EXF done; PR6 -> stack frame
		.	
		.	
		.	
		.	
		.	

Crash Sequence for Type 4 Crash: EXECUTE Switches

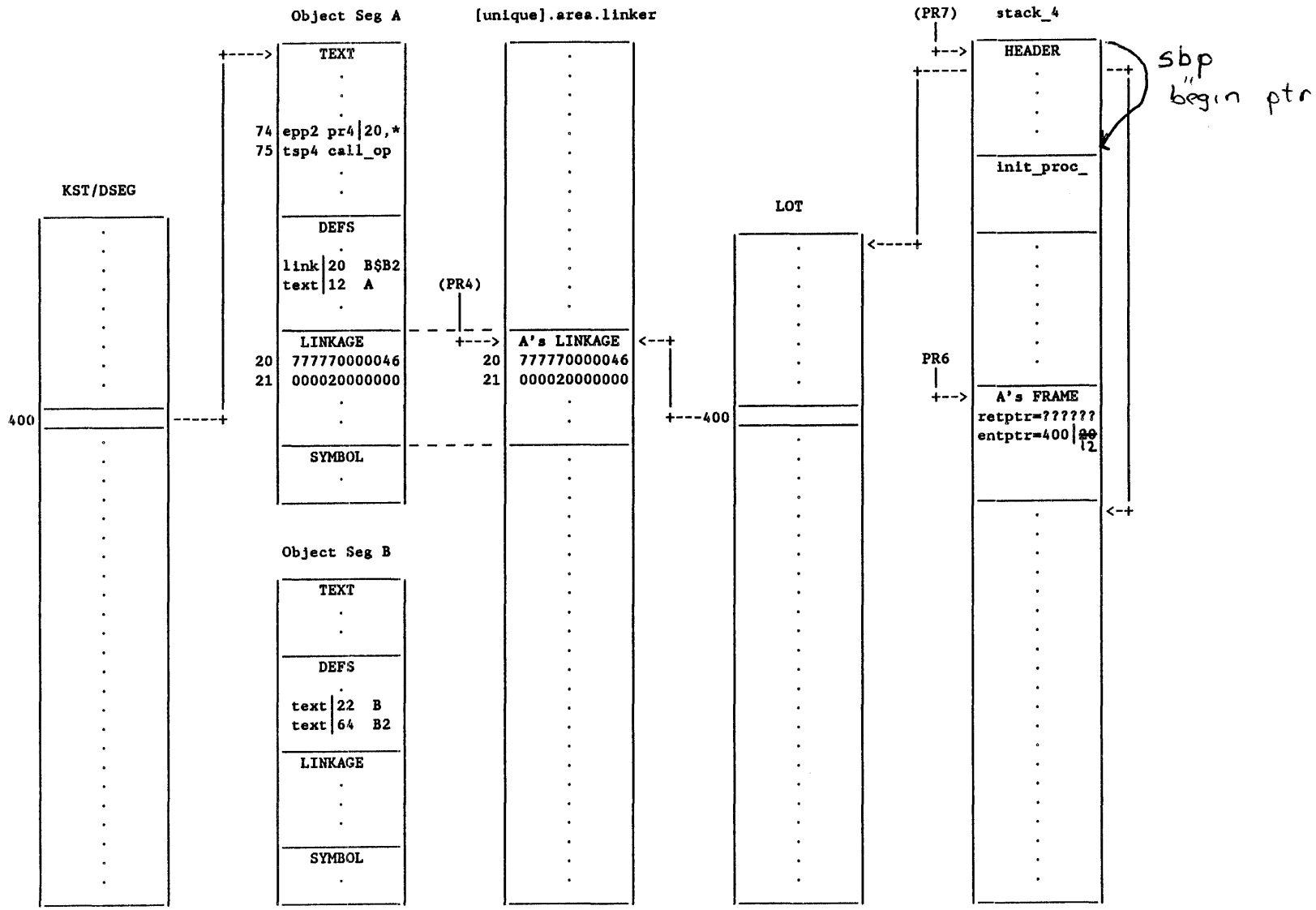
CPU A (Bootload CPU)	CPU B	CPU C
.	.	.
.	.	.
.	.	.
.	<<MANUAL HALT>>	.
.		.
.		<<MANUAL HALT>>
.		
<<EXECUTE SWITCHES>>		
(XED ~~10000~~)		
2400c		
BOS/BCE		
print MANUAL RETURN TO BOS		

Typical Dump Events for Type 4 Crash: EXECUTE Switches

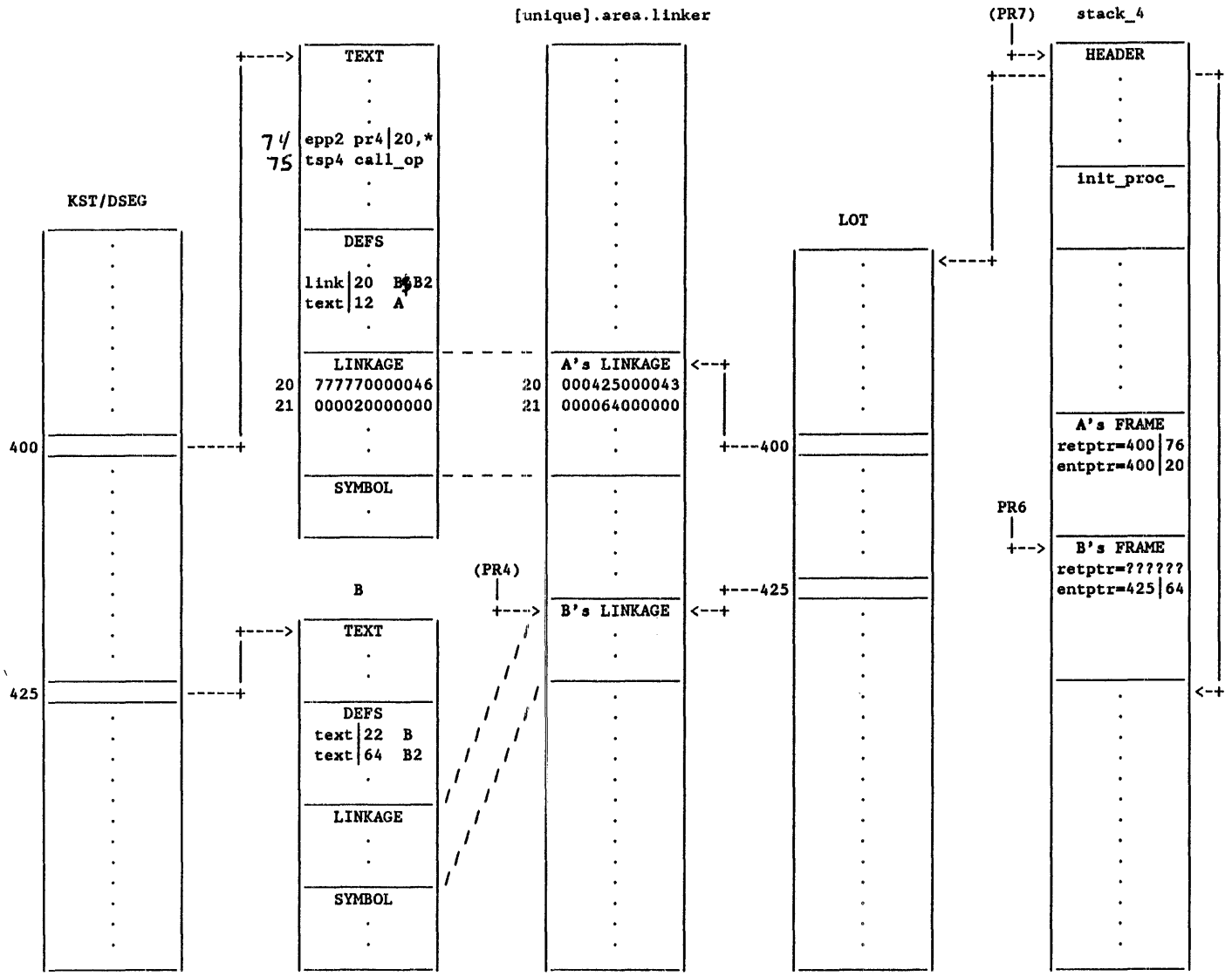
DUMP EVENTS SEQUENCE:

Event	CPU	MC location	Context/significance of MCs
XED	Bootload	Dump Header (DREGS)	PR6 -> stack frame of looping program
		.	
		.	
		.	
		.	
		.	

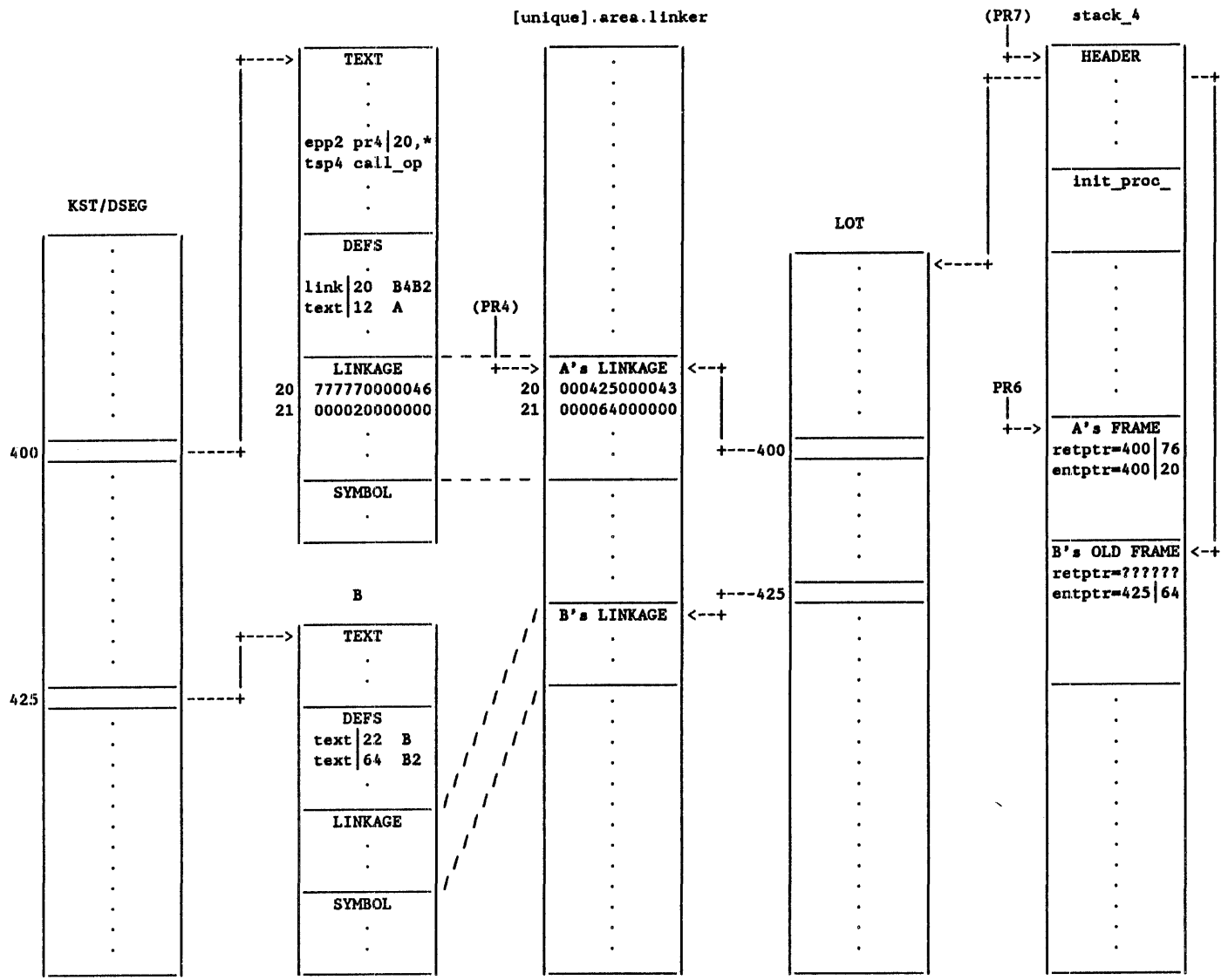
Stack, etc. Before Call to B$B2



Stack, etc. During Call to B$B2



Stack, etc. After Call to B$B2



```
   COMPILATION LISTING OF SEGMENT example
   Compiled by: Multics PL/I Compiler, Release 28e, of February 14, 1985
   Compiled at: Honeywell Multics Op. - System M
   Compiled on: 04/23/85  1108.7 mst Tue
       Options: table list

 1 /* format: style4,indattr,ifthen,^indproc */
 2
 3 example: proc;
 4
 5 dcl  autovar                fixed bin (35);
 6 dcl  intstatvar             fixed bin (35) int static;
 7 dcl  constant               fixed bin (35) int static options (constant)
 8                             init (4);
 9 dcl  ptrvar                 ptr;
10 dcl  1 basedstruct          based (ptrvar),
11         2 element1          fixed bin (35),
12         2 element2          fixed bin (35);
13 dcl  ioa_                   entry () options (variable);
14           allocate basedstruct;
15           basedstruct.element2 = 27;
16           intstatvar = 0;
17
18           do autovar = 1 to 1000000;
19                call intproc (intstatvar);
20                if divide (intstatvar, 7, 0, 0) * 7 = intstatvar then
21                     basedstruct.element2 =
22                          basedstruct.element2 + constant + intstatvar;
23           end;
24           call ioa_ ("autovar = ^d, element2 = ^d, intstatvar = ^d",
25                autovar, basedstruct.element2, intstatvar);
26           return;
27
28 intproc: proc (parm);
29 dcl  parm                   fixed bin (35);
30 dcl  i                      fixed bin (35);
31
32           i = parm;
33           parm = i + mod (i, 6) + 1;
34      end intproc;
35
36      end example;
```

SOURCE FILES USED IN THIS COMPILATION.

LINE NUMBER	DATE MODIFIED	NAME	PATHNAME
0	04/23/85 1108.1	example.pl1	>udd>ssa>jlh>F88>example.pl1

NAMES DECLARED IN THIS COMPILATION.

IDENTIFIER	OFFSET	LOC	STORAGE CLASS	DATA TYPE	ATTRIBUTES AND REFERENCES (* indicates a set context)
NAMES DECLARED BY DECLARE STATEMENT.					
autovar		000100	automatic	fixed bin(35,0)	dcl 5 set ref 18* 24*
basedstruct			based	structure	level 1 unaligned dcl 10 set ref 14
constant		000000	constant	fixed bin(35,0)	initial dcl 7 ref 20
element2	1		based	fixed bin(35,0)	level 2 dcl 10 set ref 15* 20* 20 24*
i		000112	automatic	fixed bin(35,0)	dcl 30 set ref 32* 33 33
intstatvar		000010	internal static	fixed bin(35,0)	dcl 6 set ref 16* 19* 20 20 20 24*
ioa_		000014	constant	entry	external dcl 13 ref 24
parm			parameter	fixed bin(35,0)	dcl 29 set ref 28 32 33*
ptrvar		000102	automatic	pointer	dcl 9 set ref 10 10 10 14* 15 20 20 24
NAMES DECLARED BY EXPLICIT CONTEXT.					
example		000020	constant	entry	external dcl 3
intproc		000123	constant	entry	internal dcl 28 ref 19
NAMES DECLARED BY CONTEXT OR IMPLICATION.					
divide				builtin function	ref 20
mod				builtin function	ref 33

STORAGE REQUIREMENTS FOR THIS PROGRAM.

	Object	Text	Link	Symbol	Defs	Static
Start	0	0	174	212	141	204
Length	612	141	16	364	33	2

BLOCK NAME	STACK SIZE	TYPE	WHY NONQUICK/WHO SHARES STACK FRAME
example	112	external procedure	is an external procedure.
intproc		internal procedure	shares stack frame of external procedure example.

STORAGE FOR INTERNAL STATIC VARIABLES.

LOC	IDENTIFIER	BLOCK NAME
000010	intstatvar	example

STORAGE FOR AUTOMATIC VARIABLES.

STACK FRAME	LOC	IDENTIFIER	BLOCK NAME
example	000100	autovar	example
	000102	ptrvar	example
	000112	i	intproc

THE FOLLOWING EXTERNAL OPERATORS ARE USED BY THIS PROGRAM.

call_ext_out_desc	return	mod_fx1	ext_entry	alloc_based_storage

THE FOLLOWING EXTERNAL ENTRIES ARE CALLED BY THIS PROGRAM.

ioa_

NO EXTERNAL VARIABLES ARE USED BY THIS PROGRAM.

```
CONSTANTS
000000  aa      000000000004

000140  aa      000000000006

000001  aa      404000000043

000002  aa      524000000054

000003  aa      000003641100

000004  aa  141 165 164 157   auto
000005  aa  166 141 162 040   var
000006  aa  075 040 136 144   = ^d
000007  aa  054 040 145 154   , el
000010  aa  145 155 145 156   emen
000011  aa  164 062 040 075   t2 =
000012  aa  040 136 144 054    ^d,
000013  aa  040 151 156 164    int
000014  aa  163 164 141 164   stat
000015  aa  166 141 162 040   var
000016  aa  075 040 136 144   = ^d

BEGIN PROCEDURE example
ENTRY TO example                                        STATEMENT 1 ON LINE 3
example: proc;

000017  da      000027200000
000020  aa   000160 6270 00   eax7    112
000021  aa  7 00034 3521 20   epp2    pr7|28,*
000022  aa  2 01045 2721 00   tsp2    pr2|549         ext_entry
000023  aa      000000000000
000024  2s      000012000123
                                                        STATEMENT 1 ON LINE 14
         allocate basedstruct;

000025  aa   000002 2360 07   ldq     2,dl
000026  aa  0 01403 7001 00   tsx0    pr0|771         alloc_based_storage
000027  aa   777776 7100 04   tra     -2,ic           000025
000030  aa  6 00102 2521 00   spri2   pr6|66          ptrvar
                                                        STATEMENT 1 ON LINE 15
         basedstruct.element2 = 27;

000031  aa   000033 2360 07   ldq     27,dl
000032  aa  2 00001 7561 00   stq     pr2|1           basedstruct.element2
                                                        STATEMENT 1 ON LINE 16
         intstatvar = 0;

000033  aa  6 00044 3701 20   epp4    pr6|36,*
000034  ia  4 00010 4501 00   stz     pr4|8           intstatvar
                                                        STATEMENT 1 ON LINE 18
         do autovar = 1 to 1000000;

000035  aa   000001 2360 07   ldq     1,dl
```

```
000036  aa  6 00100 7561 00   stq       pr6|64              autovar000037  aa   000000 0110 03   nop       0,du
000040  aa  6 00100 2361 00   ldq       pr6|64              autovar
000041  aa   777742 1160 04   cmpq      -30,ic              000003 = 000003641100
000042  aa   000031 6054 04   tpnz      25,ic               000073
                                                           STATEMENT 1 ON LINE 19
            call intproc (intstatvar);

000043  aa  6 00044 3701 20   epp4      pr6|36,*
000044  ia  4 00010 3521 00   epp2      pr4|8               intstatvar
000045  aa  6 00116 2521 00   spri2     pr6|78
000046  aa  6 00114 3521 00   epp2      pr6|76
000047  aa   004000 4310 07   fld       2048,dl
000050  aa  2 00000 7571 00   staq      pr2|0
000051  aa   000052 6700 04   tsp4      42,ic               000123
                                                           STATEMENT 1 ON LINE 20
            if divide (intstatvar, 7, 0, 0) * 7 = intstatvar then
                basedstruct.element2 =
                    basedstruct.element2 + constant + intstatvar;

000052  aa  6 00044 3701 20   epp4      pr6|36,*
000053  ia  4 00010 2361 00   ldq       pr4|8               intstatvar
000054  aa   000007 5060 07   div       7,dl
000055  aa   000007 4020 07   mpy       7,dl
000056  ia  4 00010 1161 00   cmpq      pr4|8               intstatvar
000057  aa   000007 6010 04   tnz       7,ic                000066
000060  aa  6 00102 3735 20   epp7      pr6|66,*            ptrvar
000061  aa  7 00001 2351 00   lda       pr7|1               basedstruct.element2
000062  aa   000044 7330 00   lrs       36
000063  aa   000004 0330 07   adl       4,dl
000064  ia  4 00010 0331 00   adl       pr4|8               intstatvar
000065  aa  7 00001 7561 00   stq       pr7|1               basedstruct.element2
                                                           STATEMENT 1 ON LINE 23
        end;

000066  aa  6 00100 2351 00   lda       pr6|64              autovar
000067  aa   000044 7330 00   lrs       36
000070  aa   000001 0330 07   adl       1,dl
000071  aa  6 00100 7561 00   stq       pr6|64              autovar
000072  aa   777746 7100 04   tra       -26,ic              000040
                                                           STATEMENT 1 ON LINE 24
        call ioa_ ("autovar = ^d, element2 = ^d, intstatvar = ^d",
            autovar, basedstruct.element2, intstatvar);

000073  aa  000 100 100 404   mlr       (ic),(pr),fill(000)
000074  aa   777711 00 0054   desc9a    -55,44              000004 = 141165164157
000075  aa  6 00122 00 0054   desc9a    pr6|82,44
000076  aa  6 00122 3521 00   epp2      pr6|82
000077  aa  6 00140 2521 00   spri2     pr6|96
000100  aa  6 00100 3521 00   epp2      pr6|64              autovar
000101  aa  6 00142 2521 00   spri2     pr6|98
000102  aa  6 00102 3735 20   epp7      pr6|66,*            ptrvar
000103  aa  7 00001 3521 00   epp2      pr7|1               basedstruct.element2
000104  aa  6 00144 2521 00   spri2     pr6|100
000105  aa  6 00044 3701 20   epp4      pr6|36,*
000106  ia  4 00010 3521 00   epp2      pr4|8               intstatvar000107  aa  6 00146 2521 00   spri2     pr6|102
```

```
000110  aa   777672 3520 04   epp2      -70,ic            000002 = 524000000054
000111  aa  6 00150 2521 00   spri2     pr6|104
000112  aa   777667 3520 04   epp2      -73,ic            000001 = 404000000043
000113  aa  6 00152 2521 00   spri2     pr6|106
000114  aa  6 00154 2521 00   spri2     pr6|108
000115  aa  6 00156 2521 00   spri2     pr6|110
000116  aa  6 00136 6211 00   eaxl      pr6|94
000117  aa   020000 4310 07   fld       8192,dl
000120  la  4 00014 3521 20   epp2      pr4|12,*          ioa_
000121  aa  0 00622 7001 00   tsx0      pr0|402           call_ext_out_desc
                                                        STATEMENT 1 ON LINE 26
          return;

000122  aa  0 00631 7101 00   tra       pr0|409           return
                                                        STATEMENT 1 ON LINE 36
     end example;

BEGIN PROCEDURE intproc
ENTRY TO intproc                                        STATEMENT 1 ON LINE 28
intproc: proc (parm);

000123  aa  6 00104 6501 00   spri4     pr6|68
000124  aa  6 00106 2521 00   spri2     pr6|70
                                                        STATEMENT 1 ON LINE 32
          i = parm;

000125  aa  2 00002 2361 20   ldq       pr2|2,*           parm
000126  aa  6 00112 7561 00   stq       pr6|74            i
                                                        STATEMENT 1 ON LINE 33
          parm = i + mod (i, 6) + 1;

000127  aa   000011 3520 04   epp2      9,ic              000140 = 000000000006
000130  aa  0 00704 7001 00   tsx0      pr0|452           mod_fx1
000131  aa   000044 7770 00   llr       36
000132  aa   000044 7330 00   lrs       36
000133  aa  6 00112 0331 00   adl       pr6|74            i
000134  aa   000001 0330 07   adl       1,dl
000135  aa  6 00106 3735 20   epp7      pr6|70,*
000136  aa  7 00002 7561 20   stq       pr7|2,*           parm
                                                        STATEMENT 1 ON LINE 34
     end intproc;

000137  aa  6 00104 6101 00   rtcd      pr6|68
  END PROCEDURE intproc
  END PROCEDURE example
```

```
azm

azm:  ?
Available azm requests:

absolute_address, absadr            select_process, slp
add_request_table, arqt             search, srh
apply, ap                           segment_name, name
apte                                segment_number, number
associative_memory, am              set
aste                                syserr_log, slog
configuration_deck, cd              stack, sk
display, d                          traffic_control_queue, tcq
display_absolute, da                value, v
events, ev                          verify_associative_memory, vfam
history_regs, hregs                 why
list_dumps, lsd                     help
list_processes, lsp                 list_help, lh
machine_conditions, mc              list_requests, lr
page_trace, pgt                     abbrev, ab
quit, q                             exec_com, ec
replace, rp                         do
scus                                if
sdw                                 answer
select_deadproc, sldp               execute, e
select_dump, sld                    ds

Type "list_requests" for a short description of the requests.

azm:  lsd
Dumps in >dumps:
     032885.1120.0.156
     041185.0001.0.160

azm:  ..asp dumps >dumps>old_dumps

azm:  lsd
Dumps in >dumps:
     032885.1120.0.156
     041185.0001.0.160
Dumps in >dumps>old_dumps:
     121284.0638.0.127
     121384.0832.0.128
     122084.0728.0.135
     030885.2239.0.151
     031585.1653.0.152
     031585.2158.0.153
     031885.1148.0.154
     031885.1515.0.155
     040385.0023.0.158
     040385.0120.0.159

azm:  sld 159
  ERF 159 in directory >dumps>old_dumps dumped at 04/03/85  0120.6 mst Wed.
```

```
  System-ID MR11.0 Version-ID 41-13
Proc   0 DBR   6234350 running        on cpu c   Initializer.SysDaemon.z

azm:  d scs$trouble_processid
Segno 25 bound_hc_data_wired$scs|222

   222      0 003000777777

azm:  slog -last 20
There are 2 messages in syserr_data (segment #106).
0119.7 1479393  0 Multics not in operation; control process: Initializer.SysDaemon.z.

Syserr messages from log partition:

01:19:40 1479390  3 emergency interrupt from FNP a: unknown faul\000
01:19:39 1479389  3 emergency interrupt from FNP a: unknown fault
01:19:27 1479388  5 ioi_masked$interrupt: I/O error.
01:19:18 1479387  0 FNP a loaded successfully
01:19:17 1479386  5 ioi_masked$interrupt: I/O error.
01:19:02 1479385  5 ==
01:19:02 1479384  5 ==
01:19:02 1479383  5 ==
01:18:54 1479382  0 Loading FNP a, >udd>SysAdmin>a>mcs7.2>6670a>a 7.2
01:18:36 1479381  5 ioi_masked$interrupt: I/O error.
01:18:31 1479380  4 RCP: Unassigned prtd from Utility.SysDaemon.z
01:18:30 1479379  0 RCP: Detached prtd from Utility.SysDaemon.z
01:18:28 1479378  3 poll_mpc: I/O error on urpa: Parity error on I/O bus, data from c
01:18:26 1479377  0 RCP: Attached prtd for Utility.SysDaemon.z
01:18:26 1479376  4 RCP: Assigned prtd to Utility.SysDaemon.z
01:18:25 1479375  4 RCP: Unassigned tapa_00 from Utility.SysDaemon.z
01:18:25 1479374  0 RCP: Detached tapa_00 from Utility.SysDaemon.z
01:18:24 1479373  5 poll_mpc: Polled mtpa.
01:18:22 1479372  0 RCP: Attached tapa_00 for Utility.SysDaemon.z
01:18:22 1479371  4 RCP: Assigned tapa_00 to Utility.SysDaemon.z

azm:  help events -bf
Syntax: events {-control_args}

Function: Displays 'interesting events', in reverse chronological
order, from an FDUMP.  Those events considered to be interesting are
described in 'notes'.

Control Arguments:      -long, -lg
 -last {N}, -lt {N}     -time {NSECS}, -tm {NSECS}

azm:  ev -last 20
Events from 04/03/85  1:19:47.870284
    Time  CPU Proc Event                          Circumstances

47.870284  d       Fault:  DRL                    RTB Machine Conditions
43.436614  a    8  Fault:  CON                    prds$sys_trouble_data
  .436614  b   10  Fault:  CON                    prds$sys_trouble_data
  .436516  d    6  Fault:  CON                    prds$sys_trouble_data
  .436383  c    0  Fault:  CON                    prds$sys_trouble_data
  .435683  d    6  Interrupt: IOM A, Level 3      prds$interrupt_data
```

```
  .433229  a    8  APTE at   5300 changed to Running
  .433041  a   12  APTE at  10500 changed to Waiting for 000000231167
  .429753          Connect to A 36
  .426543  a   15  APTE at  11700 changed to Ready
  .423070  a   11  APTE at  10300 changed to Waiting for 000000542415
  .422441          Connect to A 29
  .418303  a    7  APTE at   5200 changed to Waiting for 000000332124
  .418192          Connect to B 28
  .417694          Connect to A 20
  .416592  a    7  Fault:  DF1                    pds$page_fault_data
  .414092  a   14  APTE at  11600 changed to Waiting for 000000215021
  .413204          Connect to A 28
  .405858  a   11  Fault:  DF1                    pds$page_fault_data
  .402857  a   14  Fault:  DF1                    pds$page_fault_data

azm:  cd cpu mem

cpu  a  3  on  168  80.  2.
cpu  b  4  on  168  80.  2.
cpu  c  5  on  dps8  70.  32.
cpu  d  6  on  dps8  70.  32.
cpu  e  7  off  dps8  70.  32.
mem  d  2048.  on
mem  a  1024.  on
mem  b  2048.  off
mem  c  4096.  on


azm:  scus
Memory Configuration:
     Mem D        0 to  7777777
     Mem A 20000000 to 23777777
     Mem C 40000000 to 57777777

azm:  tcq
ELIGIBLE QUEUE:
Proc   DBR      State     Process ID  CPU
  5 06060114   waiting   003600666666      SyserrLogger.SysDaemon.z
               SYSERR LOG EVENT 0000000
  0 06234350   running   003000777777  c   Initializer.SysDaemon.z
 14 06174074   waiting   011600105645      JNye.CSDUK.a
               PAGE 215021
  7 06154234   waiting   005200105632      Dist2.VIS.a
               PAGE 332124
  8 06154514   running   005300105633  a   Dumper.SysDaemon.z
 15 06200214   ready     011700105646      Backup.SysDaemon.z
 10 06157174   running   010200105635  b   Retriever.SysDaemon.z
 11 06155774   waiting   010300105634      Volume_Dumper.Daemon.z
               PAGE 542415
 12 06077554   waiting   010500105624      Cox.Multics.m
               PAGE 231167
  9 06122354   ready     010100105614      Operator.Operator.a
 13 06166434   ready     011300105642      UNCP.CII-HB.a
  6 06077754   running   005100105601  d   TR_Admin.TR.p
```

```
azm:  lsp
Proc   0 DBR   6234350 running         on cpu c   Initializer.SysDaemon.z
Proc   1 DBR     52416 ready     last on cpu a   Idle.SysControl.a
Proc   2 DBR     52417 ready     last on cpu b   Idle.SysControl.b
Proc   3 DBR     52420 ready     last on cpu c   Idle.SysControl.c
Proc   4 DBR     52421 ready     last on cpu d   Idle.SysControl.d
Proc   5 DBR   6060114 waiting   last on cpu b   SyserrLogger.SysDaemon.z
Proc   6 DBR   6077754 running         on cpu d   TR_Admin.TR.p
Proc   7 DBR   6154234 waiting   last on cpu a   Dist2.VIS.a
Proc   8 DBR   6154514 running         on cpu a   Dumper.SysDaemon.z
Proc   9 DBR   6122354 ready     last on cpu b   Operator.Operator.a
Proc  10 DBR   6157174 running         on cpu b   Retriever.SysDaemon.z
Proc  11 DBR   6155774 waiting   last on cpu a   Volume_Dumper.Daemon.z
Proc  12 DBR   6077554 waiting   last on cpu a   Cox.Multics.m
Proc  13 DBR   6166434 ready     last on cpu a   UNCP.CII-HB.a
Proc  14 DBR   6174074 waiting   last on cpu a   JNye.CSDUK.a
Proc  15 DBR   6200214 ready     last on cpu a   Backup.SysDaemon.z

azm:  slp 1
Process   1, Idle.SysControl.a, DBR     52416

azm:  slp [d scs$trouble_processid]
Process   0, Initializer.SysDaemon.z, DBR   6234350

azm:  mc -pds

Machine conditions from pds$page_fault_data:
(DF1) Page Fault (43)
   PR6 (sp) - 230|35500   >sl1>stack_0.027|35500
SCU Data:
By: 42|3121  bound_library_wired_$formline_|25
Ref: 230|36107  >sl1>stack_0.027|36107
On: cpu c (#2)
Indicators:  zero, ^bar
APU Status:  sdwamm, sd-on, pt-on, ptw
Instructions:
 16700   6 00407 7551 00      sta       pr6|407
 16701   6 00407 7551 00      sta       pr6|407
Fault Register: 000040000000  ($CON D)
MC Fault Time: 85-04-03 01:19:42.461292 mst Wed  (113442703553566554)
Setting Temporary pointers from 71|0.

Machine conditions from pds$fim_data:
Access Violation Fault (51), Out of Segment Bounds
   PR6 (sp) - 230|7020    >sl1>stack_0.027|7020
SCU Data:
By: 150|11213  bound_x25_mpx$x25_mpx|10513
Ref: 77777|15  >sl1>stack_0.027|15
On: cpu c (#2)
Indicators:  ^bar
APU Status:  sdwamm, sd-on, ptwamm, pt-on, fap
Instructions:
 16700   7 00014 4501 00      stz       pr7|14
 16701   7 00014 4501 00      stz       pr7|14
MC Fault Time: 85-04-03 01:19:41.494596 mst Wed  (113442703550026504)
```

```
Setting Temporary pointers from 71|60.

Machine conditions from pds$signal_data:
Access Violation Fault (51), Out of Segment Bounds
   PR6 (sp) - 230|7020    >sl1>stack_0.027|7020
SCU Data:
By: 150|11213  bound_x25_mpx$x25_mpx|10513
Ref: 77777|15  >sl1>stack_0.027|15
On: cpu c (#2)
Indicators:  ^bar
APU Status:  sdwamm, sd-on, ptwamm, pt-on, fap
Instructions:
 16700   7 00014 4501 00      stz       pr7|14
 16701   7 00014 4501 00      stz       pr7|14
MC Fault Time: 85-04-03 01:19:41.494596 mst Wed  (113442703550026504)
Setting Temporary pointers from 71|140.

azm:  mc -prds sys

Machine conditions from prds$sys_trouble_data:
Connect Fault (21)
   PR6 (sp) - 230|30000   >sl1>stack_0.027|30000
SCU Data:
By: 44|315  bound_priv_1$privileged_mode_ut|315
Ref: 230|36107  >sl1>stack_0.027|36107
On: cpu c (#2)
Indicators:  neg, cary, tro, ^bar
APU Status:  priv, sdwamm, sd-on, ptwamm, pt-on, fap
CU Status:   rfi, fif
Instructions:
 16700    000314 6162 00      dis       314 interrupt inhibit
 16701    000000 2350 07      lda       0,dl
Fault Register: 000040000000  ($CON D)
MC Fault Time: 85-04-03 01:19:43.436383 mst Wed  (113442703557347137)
Setting Temporary pointers from 72|240.

azm:  sk stack_4

Reverse trace of >pdd> zzzzzzzbBBBBBB>stack_4 (Seg 234)
Number of stack frames 6.
Stack begin = 234|2000 Stack end = 234|7340
FRAME       RETURN_PTR
234|5460    404|5674     >sss>bound_command_loop_$tty_io_|5674
234|4060    432|24345    >t>bound_as_mc_$mc_tty_|3007
234|3440    251|2476     >sl1>bound_ipc_$ipc_real_|2144
234|3260    251|327      >sl1>bound_ipc_$ipc_fast_|265
234|2400    256|2235     >sl1>bound_oc_$ocd_|2235
234|2000    263|410      >sl1>bound_system_control_$system_control_|410
Previous stack frame 77777|1

azm:  sk stack_0

Reverse trace of >sl1>stack_0.027 (Seg 230)
Number of stack frames 14.
Stack begin = 230|100 Stack end = 230|33620
```

```
FRAME        RETURN_PTR
230|30000    35|1577       bound_error_wired_1
230|12600    122|25116     bound_355_wired$channel_manager|2156
230|12320    41|5440       bound_library_1_$unwind_stack_|370
230|12160    41|0          bound_library_1_$init_vol_header_|0
230|11640    41|10366      bound_library_1_$signal_|726
230|11220    76|0          return_to_ring_0_|0
FIM FRAME found at 230|11220
Machine Conditions at 230|11300:
Access Violation Fault (51), Out of Segment Bounds
   PR6 (sp) - 230|7020    >sl1>stack_0.027|7020
By: 150|11213  bound_x25_mpx$x25_mpx|10513
Ref: 77777|15  >sl1>stack_0.027|15
230|7020     150|0         bound_x25_mpx$x25_mpx_data|0
230|6300     122|25335     bound_355_wired$channel_manager|2375
230|4660     122|6324      bound_355_wired$dn355|6324
230|3240     122|22153     bound_355_wired$fnp_multiplexer|10207
230|2520     122|24512     bound_355_wired$channel_manager|1552
230|1360     146|34565     bound_tty_active$tty_modes|1271
230|220      146|24224     bound_tty_active$tty_index|2670
230|100      317|3374      >sl1>hcs_|3374
Previous stack frame 234|5460

azm:  sk prds
Frames may be invalid.
Stack_begin and stack_end are equal 72|1220.
Use the -force and -fwd options and proceed at your own risk

azm:  sk prds -fc -fwd
          (fwd) next_sp not valid 27374|0 sp 72|2200

Forward trace of prds (Seg 72)
Number of stack frames 4.
Previous stack frame 230|33620.
Stack begin = 72|1220 Stack end = 72|1220
FRAME        RETURN_PTR
72|1220      45|3767       bound_tc_priv$pxss|3767
72|1520      145|455       bound_tc_wired$proc_int_handler|71
72|1700      122|5221      bound_355_wired$dn355|5221
72|2200      43|24040      bound_page_control$disk_control|2474

azm:  mc 230|11220 -lg
Machine Conditions from (230|11300) >sl1>stack_0.027|11300.
Pointer Registers:
   PR0 (ap) - 42|17350     bound_library_wired_$pl1_operators_|1426
   PR1 (ab) - 77777|1      NULL POINTER
   PR2 (bp) - 150|4155     bound_x25_mpx$x25_mpx|3455
   PR3 (bb) - 230|6174     >sl1>stack_0.027|6174
   PR4 (lp) - 150|11253    bound_x25_mpx$x25_mpx|10553
   PR5 (lb) - 77777|1      NULL POINTER
   PR6 (sp) - 230|7020     >sl1>stack_0.027|7020
   PR7 (sb) - 77777|1      NULL POINTER
Processor Registers:
   X0 - 25335 X1 - 6556 X2 - 777773 X3 - 460
   X4 - 0 X5 - 3357 X6 - 7 X7 - 1540
```

```
   A Register - 000000000000 Q Register - 000000000040 E Register - 0
   Timer Register - 775006421 Ring Alarm Register - 1
Access Violation Fault (51), Out of Segment Bounds
SCU Data:
By: 150|11213  bound_x25_mpx$x25_mpx|10513
Ref: 77777|15  >sl1>stack_0.027|15
On: cpu c (#2)
Indicators:  ^bar
APU Status:  sdwamm, sd-on, ptwamm, pt-on, fap
Instructions:
 20420   7 00014 4501 00      stz       pr7|14
 20421   7 00014 4501 00      stz       pr7|14
Mem Controller Mask: 000230000043 010560000000
MC Fault Time: 85-04-03 01:19:41.494596 mst Wed  (113442703550026504)
Setting Temporary pointers from 230|11300.

azm:  v -a
ap = 42|17350
ab = 77777|1
bp = 150|4155
bb = 230|6174
lp = 150|11253
lb = 77777|1
sp = 230|7020
sb = 77777|1
prfr = 230|7020
prmc = 230|11300

azm:  d sp -as stack_frame/arg_ptr/
stack_frame.arg_ptr = 230|6556 >sl1>stack_0.027|6556

azm:  sk sp -ag -for 1

Reverse trace of >sl1>stack_0.027 (Seg 230)
Number of stack frames 8.
Stack begin = 230|100 Stack end = 230|33620
FRAME       RETURN_PTR
230|7020    150|0        bound_x25_mpx$x25_mpx_data|0

          Entry ptr 150|4155 bound_x25_mpx$x25_mpx|3455
          Operator/Link ptr 42|17350
          Arg ptr 230|6556
          ARG  1: 221|165226  tty_buf|165226
          ARG  2: 000000000003
          ARG  3: 000000000000

Previous stack frame 230|6300

azm:  sdw 74
 ADDRESS RNGS  CA-MAX REWPUGCDF EBOUND SEGNO SEGMENT-NAME
 6057014  000    1777 R W  G DF      0    74 rdisk_seg

azm:  name 41
41|0 = bound_library_1_$init_vol_header_|0
```

```
azm:  d 41|10360 10
azm (display): Segment is not in the fdump.  41|10

azm:  d 41|10360 10 -inst
Segno 41 bound_library_1_$signal_|720
10360   005146 3520 00        epp2      5146
10361  0 00623 7001 00        tsx0      pr0|623
10362   000011 7260 07        lxl6      11,dl
10363   777103 3520 04        epp2      -675,ic              007466 = 000000000000
10364  0 00717 7001 00        tsx0      pr0|717
10365   000004 7100 04        tra       4,ic                 010371
10366   000154 0000 00        ....      154
10367   000021 7100 04        tra       21,ic                010410

azm:  absadr 41|10360
bound_library_1_$signal_|720 (41|10360):
Absolute Addr 7602360 (Word 7602360 in Mem d).


azm:  ..cpo fault_vector

azm:  number fault_vector
fault_vector = Segno 4|0.

azm:  ap 4 "do ""compare fault_vector &1 -ln 200"" "
Discrepancies:
    offset    contents    offset    contents
       114  000514657220       114  025720657200
       115  000414710220       115  025773630200
       116  000516657220       116  153450657200
       117  000416710220       117  153450613200

Total 1 discrepancy, 4 words

azm:  d 4 -as "fault_vector.fpair(6:7)"
  fpair (6)                    @ 4|114
    scu = "025720657200"b3, tra = "025773630200"b3
  fpair (7)                    @ 4|116
    scu = "153450657200"b3, tra = "153450613200"b3

azm:  d 4|114 4 -inst
Segno 4 fault_vector|114
114   025720 6572 00          scu       25720 interrupt inhibit
115   025773 6302 00          ret       25773 interrupt inhibit
116   153450 6572 00          scu       153450 interrupt inhibit
117   153450 6132 00          rcu       153450 interrupt inhibit

azm:  replace 4 [e wd]>fault_vector

azm:  d 4 -as "fault_vector.fpair(6:7)"
  fpair (6)                    @ 4|114
    scu = "000514657220"b3, tra = "000414710220"b3
  fpair (7)                    @ 4|116
    scu = "000516657220"b3, tra = "000416710220"b3
```

```
azm:  d 4|114 4 -inst
Segno 4 fault_vector|114
114   000514 6572 20          scu       514,* interrupt inhibit
115   000414 7102 20          tra       414,* interrupt inhibit
116   000516 6572 20          scu       516,* interrupt inhibit
117   000416 7102 20          tra       416,* interrupt inhibit

azm:  apte -cur

APTE #1 at ADDR 3000:
Processid:  003000777777 (Initializer.SysDaemon.z); DBR:      6234350
State:      running at 4/3/85 1:19:41.519133

azm:  d tc_data|3000 -as apte
apte                            @ 112|3000
  thread                        @ 112|3000
    fp = "011600"b3, bp = "003600"b3
  flags                         @ 112|3001
    state = "000001"b3
    ON:  wakeup_waiting, loaded, eligible, pre_empt_pending,
         default_procs_required, dbr_loaded, shared_stack_0, firstsw
    OFF: stop_pending, pre_empted, hproc, idle, interaction,
         realtime_burst, always_loaded, being_loaded, page_wait_flag
  page_faults = 10744, processid = "003000777777"b3, te = 2221386,
  ts = 0, ti = 0, timax = 0
  ipc_pointers                  @ 112|3010
    event_thread = "034710"b3
  ips_message = "000000000000"b3
  asteps                        @ 112|3012
    pds = "005300"b3, dseg = "162334"b3, prds = "000000"b3
  savex7 = "002402"b3, term_processid = "000000000000"b3,
  lock_id = "134426563417"b3, time_used_clock = 375911791,
  wait_event = "000000000000"b3, wct_index = "001110"b3
  flags2                        @ 112|3021(18)
    special_wakeups = "00"b3, pr_tag = "2"b3
    OFF: priority_scheduling, batch
  ,
  state_change_time = 2658817181519133   1985-04-03 01:19:41.519133 mst,
  alarm_event = 0, alarm_time_thread = "010700"b3,
  alarm_time = "113442703634400115"b3, term_channel = 0, ws_size = 0,
  temax = 2097152,
  deadline = 2658817174541992   1985-04-03 01:19:34.541992 mst,
  lock = "000212"b3, cpu_monitor = 0, paging_measure = 16320822,
  access_authorization = "000000000000000000770000"b3,
  dbr = 465269129579410522131, virtual_cpu_time = 314797070,
  ittes_sent = 1, ittes_got = 56, current_response_state = 0,
  number_processing = 91,
  last_response_state_time =
       2658817179468364   1985-04-03 01:19:39.468364 mst
  total_processing_time = 62781250, begin_interaction_vcpu = 312578134,
  saved_temax = 2097152, procs_required = "FF"b4, ipc_r_offset = 178898,
  ipc_r_factor = 57812222997, apad (1) through apad (10) = 0

azm:  hregs
History Registers at (71|220) pds|220
```

DPS8 History Register Analysis

```
HR          IC or              c   Memory
id   Seg# [tpr.ca] opcode tag y  Address mc flags

CU            307    tnz     *      304  4 poa raw ic pib port
OU                                       RS-REG=stac zero carry
CU            310   stac   n* i   236312  4 pia poa riw raw inf its internal
OU                                       RS-REG=stac zero carry
CU     17                    n* n   114776  4 poa riw raw its internal
OU                                       RS-REG=stac zero carry
AU          [2776]                114776 r0 fap sdwm (A17)
CU     25                       o   122222 40 poa internal
OU                                       RS-REG=stac zero carry
AU           [222]                122222 r0 fap sdwm (A5)
CU    230                       o   122222 60 poa raw saw pib port
OU                                       RS-REG=stac zero carry
CU     17      311    lda   n* n   115000  4 poa riw raw ic pib its port
OU                                       RS-REG=stac zero carry
AU          [3000]                122222 r0 fap sdwm (A17) ptwm (A1)
CU     25                       o   122220  4 poa raw ic pib port
OU                                       RS-REG=stac zero carry
AU           [220]                122220 r0 fap sdwm (A5)
CU     17      312    sta   n* i   236314  4 pia poa riw raw inf pib its internal
OU                                       RS-REG=stac zero carry
CU     17                    n* n   115002  4 poa riw raw pib its port
OU                                       RS-REG=lda sign carry
AU          [3002]                236314 r0 fap sdwm (A17) ptwm (A1)
CU     25                       o   122203 20 poa raw internal
OU                                       RS-REG=lda sign carry
AU           [203]                122203 r0 fap sdwm (A5)
CU     17      313   cioc   n* n   115004  4 poa riw raw ic inh pib its port
OU                                       RS-REG=sta sign carry
AU          [3004]                122203 r0 fap sdwm (A17)
CU     25                   ql* n   122104  4 poa riw raw ic inh pib port
OU                                       RS-REG=sta sign carry
AU           [104]                122104 r0 fap sdwm (A5)
CU     25                       o   122067 62 poa raw ic inh internal
OU                                       RS-REG=sta sign carry
AU            [67]                122067 r0 fap sdwm (A5)
CU     44      314    dis     i   236316  4 pia poa raw inf inh port
OU                                       RS-REG=sta sign carry
CU     44                       o   236314  4 poa raw inh internal
OU                                       RS-REG=sta sign carry
CU     44                       ?   236314  4 raw inh internal
OU                                       RS-REG=sta sign carry


azm:  name 17
17|0 = ws_linkage|0

azm:  d 44|310 10 -inst
Segno 44 bound_priv_1$privileged_mode_ut|310
310  4 00114 3541 20          stac      pr4|114,*
311  4 00116 2351 20          lda       pr4|116,*
```

```
312  4 00120 7551 20          sta       pr4|120,*
313  4 00122 0153 20          cioc      pr4|122,* interrupt inhibit
314   000314 6162 00          dis       314 interrupt inhibit
315   000000 2350 07          lda       0,dl
316  4 00114 3551 20          ansa      pr4|114,*
317  4 00112 3551 20          ansa      pr4|112,*

azm:  am -prds

SDW Associative Memory at prds$am_data.
 ADDRESS  RINGS  BOUND  REWPUGC    CL F/E USAGE-CT SEG # SEG_NAME
LEVEL [A]
 6234350  0,0,0  17760  R W  G      -  F    100000     0 dseg
 6357014  0,0,0 107760  RE   GC     -  F    111000    42 bound_library_wired_
       0  0,0,0    560  R W UG     -  F    111000     4 fault_vector
   52056  0,0,0   3760  R W  G      -  F    111000    25 bound_hc_data_wired
   52360  0,0,0   5760  R W  G      -  F    100000    66 iom_data
 6240074  0,0,0  37760  R W  GC     -  F    111000   230 >sl1>stack_0.027
 6057314  0,0,0   7760  R W  GC     -  F    100000    71 pds
 6241370  0,0,0  17760  REWP G      -  F    101100    72 prds
   52066  0,0,0  21760  RE P GC     -  F    111000    35 bound_error_wired_1
   52102  0,0,0   5760  RE P GC     -  F    100000    36 bound_interceptors
   52046  0,0,0  11760  REW  G      -  F    100000    17 ws_linkage
LEVEL [B]
   52436  0,0,0 763760  R W  G      -  F    111000   102 sst_seg
   52154  0,0,0   7760  RE P GC     -  F    111000    44 bound_priv_1
 6057334  0,0,0   1760  R W  G      -  F    111000   105 sys_info
   52232  0,0,0   3760  R W  G      -  F    111000    70 oc_data
   52366  0,0,0  43760  R W  G      -  F    101100   112 tc_data
   52042  0,0,0   1760  R W  G      -  F    111000    15 lot
LEVEL [C]
   52026  0,0,0   1760  R W  G      -  F    101100    12 unpaged_page_tables

PTW Associative Memory at prds$am_data.
 ADDRESS   M  F/E USAGE_CT SEG # PAGE SEG_NAME|OFFSET
LEVEL [A]
56316000  yes  F    110000    72    0 prds|200
  114000  yes  F    111100    17    1 ws_linkage|600
  136000  yes  F    100000    35    3 bound_error_wired_1
  140000  yes  F    100000    35    4 bound_error_wired_1
  142000  yes  F    100000    35    5 bound_error_wired_1
  144000  yes  F    100000    35    6 bound_error_wired_1
 7556000  no   F    111000    42    7 bound_library_wired_$formline_|504
 7554000  no   F    100000    42   10 bound_library_wired_$formline_|1104
 7552000  no   F    100000    42   11 bound_library_wired_$formline_|1504
 7550000  no   F    100000    42   12 bound_library_wired_$formline_|2104
 7546000  no   F    111000    42   13 bound_library_wired_$formline_|2504
 7544000  no   F    111000    42   14 bound_library_wired_$formline_|3104
42602000  yes  F    100000   230   15 >sl1>stack_0.027|6400
21244000  yes  F    100000   230   16 >sl1>stack_0.027|7000
47012000  yes  F    100000   230   17 >sl1>stack_0.027|7400
LEVEL [B]
 7330000  yes  F    110000    71    0 pds|200
  152000  yes  F    111100    36    1 bound_interceptors$fim|600
  146000  yes  F    111000    35    7 bound_error_wired_1
```

```
 6240000  yes  F    111000   102   73 sst_seg|35600
 3116000  yes  F    111000   230   14 >sl1>stack_0.027|6000
LEVEL [C]
  150000  yes  F    110000    36    0 bound_interceptors$fim|200
  132000  yes  F    111100    35    1 bound_error_wired_1
LEVEL [D]
  112000  yes  F    110000    17    0 ws_linkage|200
  312000  yes  F    111100    70    1 oc_data|400

azm:  vfam
No mis-matches or duplicate entries found in SDWAM or PTWAM.

azm:  q
```

07/27/83 analyze_multics, azm

Syntax: analyze_multics {-control_args}

Function: Invoke a subsystem that will permit the scanning of a Multics address space for analysis.

Control arguments:

-abbrev, -ab
   enables abbreviation expansion of request lines.

-no_abbrev, -nab
   does not enable abbreviation expansion of request lines. (Default)

-no_prompt
   suppresses the prompt for request lines in the request loop.

-no_start_up, -nsu
   specifies that no startup exec_com is to be executed. (Default)

-profile PATH, -pf PATH
   specifies the pathname of the profile to use for abbreviation expansion. The suffix "profile" is added if necessary. This control argument implies -abbrev.

-prompt STR
   sets the request loop prompt to STR. The default is the ioa_ STR:
                  ^/azm^[(^d)^]:^2x

-request STR, -rq STR
   executes STR as a analyze_multics request line before entering the request loop.

-start_up, -su
   specifies that the exec_com 'tart_up.azmec' is to be executed upon invocation of analyze_multics. This start_up exec_com is first searched for in the user's home directory, then in the user's project directory >udd>Project_id, and last in >site. The first exec_com found is used.

-quit
   Exit analyze_mulitcs after execution of other arguments. Can be used in conjunction with -request.

Notes: analyze_multics uses the standard search list mechanism to locate FDUMPs. If analyze_multics does not find a "dumps" search list, it will create one, placing >dumps in the search list as the default. If additional search paths are desired the "add_search_path" command should be used to define them.

Syntax: absadr VIRTUAL-ADDR
[absadr VIRTUAL-ADDR]

Function: Translates a 'virtual address' to an absolute memory address.

Argument:

VIRTUAL-ADDR
May be a segment number, name or symbolic address (e.g. 64, prds, prds$am_data). Do a 'help virtual_address' for more detailed information on acceptable virtual-address constructs.

Example as active request:

```
display_absolute [absadr sst$cmp] 2
```

07/27/83 add_request_table, arqt

Syntax: arqt PATH

Function: Adds a user defined request table in the list of request tables being searched by the current analyze_multics invocation.

Argument:
PATH
is the path name of the request table to be added. This request table must be consistent for use with subsystem utility. See the section on subsystem request language in the Programmer's Reference Manual for request table structure.

08/08/83 apply, ap

Syntax: apply VIRTUAL-ADDR {RANGE} command_line

Function: Extracts all or part of a segment, specified by VIRTUAL-ADDR from the selected FDUMP and places a copy in a temporary segment. The new path name is passed as the last argument in the command_line.

Argument:
VIRTUAL-ADDR
May be a segment number, name or symbolic address (e.g. 64, prds, prds$am_data). Do a 'help virtual_address' for more detailed information on acceptable virtual-address constructs.
RANGE
Specifies the number of words in octal to be copied. The default is the entire segment.
command_line
any command.

Notes: The offset in the virtual address specifies where the copying of the segment begins. When only part of a segment is extracted, it goes at the beginning of the temporary segment. For example:

```
apply pds$am_data 400 dump_segment
```

will put 256 (decimal) words at the beginning of the segment.

12/12/83 apte

Syntax: apte {proc_indicator} {-control_args}

Function: displays active page table (apte) information for processes in an FDUMP that match the states specified.

Argument:
proc_indicator
for specifying individual processes. It can take one of three forms:
- The decimal index (starting at zero) of a process inthe FDUMP.
- The octal apte offset of the process.
- The octal process_id of the process.

Control Arguments:
-all, -a
Displays apte info for all processes in any state (Default).
-blocked, -blk
Displays apte info for all processes in the blocked state.
-count, -ct
specifies the total number of processes meeting the criteria specified by the control_args. With -all, it gives the counts of each process state.
-current, -cur
displays apte info for the current process.
-page_tbl_lock, -ptl
Displays apte info for all processes marked as page table locking.
-ready, -rdy
Displays apte info for all processes in the ready state.
-run
Displays apte info for all processes in the running state.
-stopped, -stop
Displays apte info for all processes in the stopped state.
-wait
Displays apte info for all processes in the waiting state.

Examples:
apte 2
displays information for process 2 in the FDUMP.

apte 10600
displays information for the process with apte offset 10600 (octal).

apte 3500555555
displays information for the process with octal process_id 003500555555.

07/20/83 associative_memory, am

Syntax: am {-control_args}

Function: Display SDW and/or PTW Associative Memories.

Control Arguments (Location):
-dump
   displays the "dump" Associative Memories from the BOS CPU at the time the dump was taken. (Default).
-prds
   displays Associative memories that have been stored in the current processes prds.

Control arguments:
-all, -a
   Specifies that ALL entries in the Associative Memories are to be displayed. Default is to display only those entries that are valid (i.e., the full bit is on).
-ptw
   Specifies that only the PTW Associative memories are to be displayed.
-pageno PAGENO
   where PAGENO is an octal page number. Displays only those entries in the PTW Associative Memories that have a page number that matches the value of PAGENO.
-sdw
   Specifie35hat only the SDW Associative Memories are to be displayed.
-segno SEGNO
   where SEGNO is an octal segment number. Displays only those entries in the SDW and PTW Associative Memories that have a segment number that matches the value of SEGNO. See assoc_mem.incl.pl1.

Notes: If no control arguments are given, BOTH the SDW and PTW Associative Memories are displayed for the "dump" Associative Memories.

12/12/83 aste

Syntax: aste segno/segname {-control_args}

Function: displays active segment table (ast), page table, and trailer information. The default displays active segment table entry (aste) and page table information only.

Argument:
segno/name
   is the segment number or segment name of interest.

Control Arguments:
-aste
   Displays active segment table information for the selected entry.
-brief, -bf
   Displays everything excluding the page table info. This is equivalent to specifying -aste and -tr.
-long, -lg
   Displays everything which includes the aste, page table and trailer information. This is equivalent to specifying -aste, -pt and -tr.
-page_table, -pt
   Displays page table information for the selected segment.
-trailer, -tr
   Displays trailer information about the selected segment.

08/09/83 configuration_deck, cd

Syntax: cd {card_names} {control_args}

Function: displays the contents of the config_deck in the selected FDUMP. This request works exactly like the standard pcd command, the only difference is that it gets the config deck from the FDUMP.

Argument:
card_names
are the names of the particular configuration cards to be displayed. Up to 32 card names can be specified. If no card_names are given the the complete config_deck will be printed.

Control Arguments:
-brief, -bf
suppresses the error message when a requested card name is not found. (Default)
-exclude FIELD_SPECIFIERS, -ex FIELD_SPECIFIERS
excludes particular cards or card types from being displayed. One to 14 field specifiers can be supplied with each -exclude control argument, and up to 16 -exclude arguments can be specified. To be eligible for exclusion, a card must contain fields that match all field specifiers supplied with any -exclude argument.
-long, -lg
prints an error message when a requested card name is not found.
-match FIELD_SPECIFIERS
selects particular cards or card types to be displayed. One to 14 field specifiers can be supplied with each -match control argument, and up to 16 -match arguments can be specified. To be eligible for selection, a card must contain fields that match all field specifiers supplied with any -match argument.

Notes: Field specifiers can consist of a complete card field or a partial field and an asterisk (*). An asterisk matches any part of any field. Specifiers for numeric fields can be given in octal or decimal, but if decimal they must contain a decimal point. Asterisks cannot be specified in numeric field specifiers. All numeric field specifiers are converted to decimal and matched against numeric card fields, which are also converted to decimal. Hence, the field specifier "1024." would match a card containing the octal field 2000, and the field specifier "1000" would match a card containing the decimal field 512.

Note that all card names must be specified before the first -match or -exclude argument. Field specifiers following a -match or -exclude argument include all arguments until the next -match or -exclude argument.

11/19/84 display, d

Syntax: d VIRTUAL-ADDR {EXP} {RANGE} {-ctl_args}
[d VIRTUAL-ADDR {EXP} {RANGE} {-ctl_args}]

Function: displays a selected portion of a segment in the FDUMP.

Argument:
VIRTUAL-ADDR
specifies the initial offset of the virtual address space to be dumped. May be a segment number, name, or symbolic address (e.g., 64, prds, prds$am_data). Do a 'help virtual_address' for more detailed information on acceptable virtual-address constructs.
EXP
is an expression, which is either an octal value or a VIRTUAL-ADDR construct yielding an octal value. This value can be positive or negative, specified by the plus or minus sign.
RANGE
specifies the number of words to be dumped in octal. If a range is not specified the default is one word (if the data to be dumped is an ITS pair two words will be dumped).

Control arguments (Mode Specifications):
-character, -ch, -ascii
displays the selected number of characters in ascii. Characters that cannot be printed are represented as periods. As an active request, it returns the character representation of the requested address.
-instruction, -inst
displays the selected number of words as instructions. Usage as an active request is not allowed.
-octal, -oc
displays the selected number of characters in octal (Default). When used as an active request returns the octal value of the requested address.

-ptr, -p
displays the selected number of word pairs as pointers. When used as an active request returns the octal value of the form SEGNO|OFFSET.
-pptr, -pp
displays the selected number of words as a packed-pointer. When used as an active request returns the octal value of the form SEGNO|OFFSET.
-pptrx, -ppx
displays the selected number of words as packed-pointers and expands the segno|offset to a segment name. Usage as an active request is not allowed.

-ptrx, -px displays the selected number of word pairs as pointers and expands

the segno|offset to a segment name. Usage as an active request is not allowed.

Control Arguments:

-as STRUCTURE_NAME

displays the data as a hardcore PL/I structure defined by STRUCTURE_NAME. The STRUCTURE_NAME is a hardcore system-defined include file. The address given in the display request is taken as the address of the beginning of the structure. If the whole structure is being displayed, that is the address where display begins. If only certain elements are being displayed, that is the address used to compute offsets of the elements. The structure reference following -as must be a single string, containing no spaces, and follows the syntax described below. The single string is used to specify structure elements, array indexes, and substring matching. Usage as an active request is not allowed.

-long, -lg

displays each element of the structure on a separate line. This control argument is only implemented with -as.

Structure syntax:

The structure reference is made up of two parts: a structure element reference, and an optional set of match strings. If no match strings are supplied, no string matching is done. The structure element reference syntax consists of one or more element names, separated by periods, and may contain subscripts following some of these element names. The first name in a structure element reference must be a level one structure reference; partially qualified top level references are not permitted. Intermediate levels of qualification may be omitted as long as there is no ambiguity.

All subscripts must be supplied as decimal integers. The subscripts may be cross-section references such as "(1:4)" to reference elements one through four. Asterisk bounds may not be used: if a cross-section is desired, its upper and lower bounds must be given as decimal constants. If an element has more subscripts than are supplied, the complete cross-section is printed for the remaining subscripts. Also, to eliminate the need for quoting, subscripts may be surrounded by braces instead of parentheses.

In order to specify that only certain elements be displayed (such as all those with names containing the string "time"), a set of match strings may be given after the structure element reference. Each match string begins with a slash and is followed by the string itself. The final match string may be followed by a slash, but this is not required. If match strings are specified, any element which matches at least one string will be displayed.

Examples of structure references:

```
pvt
   The whole structure "pvt".
pvt.n_entries
   The single element "n_entries" in the structure "pvt".
sst/time/,  sst/time
   Any elements in the structure "sst" containing the string "time".
   Note that the final slash is optional.
sst/time/meter/
   Any elements in the structure "sst" containing either the string
   "time" or the string "meter".
sst.space{3}
   element three of "sst.space".
sst.space{2:4}
   elements two, three, and four of "sst.space".


sst.space
   all elements of "sst.space".
sst.level{1}
   both elements of the "level" array for "sst.level{1}"
sst.level{1}.ausedp, sst.level.ausedp{1}
   the single element "ausedp" of the "level" array for "sst.level{1}"


Structure Output format:
The default output format is a compressed form, which places as many
values on a line as will fit within the line length.  The -long control
argument places one value on a line.  The short form, additionally,
collects all bit(1) flags and displays them at the end of the display
for each substructure or array element, in two groups: one listing all
the flags which were on ("1"b), and one for all the ones which were off
("0"b).


All PL/I datatypes are displayed in the same representations used by
probe. Additionally, the following special formats are used:

 1) Bit strings are displayed as octal, if the length is divisible by
    three, in hex if divisible by four and not three, and as bit
    strings otherwise.
 2) Character strings are displayed as a string concatenated with a
    repeated constant, if the string is padded on the right with more
    than sixteen nulls, spaces, or octal 777 characters.
 3) Large precision (> 51) fixed binary values are also displayed as
    clock readings, if their values represent clock readings within
    ten years of the present.


Display Examples:
d 75|560 2
   displays the two words in seg number 75 starting at offset 560.

d pds|560 2   displays the two words in the segment named pds starting at
   offset 560.
```

```
d pds$trace
```
displays one word in the pds segment beginning at the offset specified by $trace.

```
display 244|260 +20 4
```
displays four words of segment number 244 starting at offset 300 octal.

```
d sp 20
```
displays 20 octal words starting with the segment offset defined in the azm internal temporary pointer (see set request).

```
d sst$cmp,* +sst$cmesize sst$strsize
```
causes the word at sst$cmp to be used as an indirect word, or an indirect pointer if the resultant address has ITS modification, to develop the starting virtual address. The value derived from sst$cmesize will then be added to the starting offset for the 'final' starting address. The range, or number of words to be displayed, is specified by the value contained in sst$strsize.

```
d sst|2 -as apte
```
displays the APTE entry at the given offset in the SST as it is defined by apte.incl.pl1.

07/26/83 display_absolute, da

Syntax: da ABS-ADDR {range} {-ctl_args}
[da ABS-ADDR {range} {-ctl_args}]

Function: dumps an absolute memory address space in the FDUMP.

Argument:
ABS-ADDR
is the starting absolute memory address, in octal.
RANGE
specifies the number of words to be dumped in octal. If a range is not specified the default is one word (if the data to be dumped is an ITS pair two words will be dumped).

Control arguments (Modes):
-character, -ch, -ascii
-instruction, -inst
-octal, -oc
-ptr, -p
-pptr, -pp
-pptrx, -ppx
-ptrx, -px

Notes: For a description of the mode specifications, see the display request.

01/23/85 events, ev

Syntax: events {-control_args}

Function: Displays 'interesting events', in reverse chronological order, from an FDUMP. Those events considered to be interesting are described in 'notes'.

Control Arguments:
-last {N}, -lt {N}
   specifies the number of events to print. If no N, the default is 10 events.
-long, -lg
   specifies that disk queue events are to be displayed.
-time {NSECS}, -tm {NSECS}
   specifies the time in seconds before the dump was taken when events were 'interesting'. Default is 10 seconds.

Notes: The following events are considered as interesting: Machine Conditions (from BOS, prds, pds and the mc_trace_buf), Traffic Control state change time, Syserr messages (from both syserr_data and syserr_log), Fim frames in any stack, connects by device and disk queues (long report ONLY).

If neither -time nor -last are specified, the default action is equivalent to "ev -time 10".

08/05/83 history_regs, hregs

Syntax: hregs {HREGS_specifier} {-control_args}

Function: Displays a composite analysis or octal dump of the processor history registers. This request, hregs, is useful by people who are knowledgable of the hardware. The default action is to display the AU, CU and OU history registers for the pds in a threaded order and interpreted format.

Argument (HREGS Specifiers):
-condition VIRTUAL-ADDR, -cond VIRTUAL-ADDR
displays history registers from a condition frame, the location of which is described by VIRTUAL-ADDR.
-dump
displays the "dump" history registers from the BOS CPU at the time the dump was taken.
-pds
displays the history registers that have been stored in the current processes pds (Default).
VIRTUAL-ADDR
displays the history registers that have been stored at the address space specified by VIRTUAL-ADDR. See virtual_address.info.

Control Arguments:
-au
specifies that only the APU history registers are to be displayed.
-cu
specifies that only the CU history registers are to be displayed.
-du
specifies that only the DU history registers are to be displayed.
-interpret
Display the interpreted form of the history registers only (Default), or, if -octal is specified, include the octal representation also.
-octal, -oc
Displays the octal values of history registers only, or, if -interpret is also specified, display octal and interpreted form. If neither -octal nor -interpret is specified, the default action is to display the interpreted form only.
-thread
Attempt to display the selected history registers in the "correct" order (Default).
-no_thread
Display the selected history registers in serial order, without attempting to sort them.
-ou
specifies that only the OU history registers are to be displayed.

Notes: Use of the -au, -ou, -cu and -du control arguments imply -no_thread and the

10/03/84 list_dumps, lsd

Syntax: lsd {PATH} {-ctl_args}

Function: Lists dumps in the selected dump directory. If PATH is not given, all dumps in the dump directories specified in the dumps search list are listed.

Arguments:
PATH
specifies PATH as the dump directory to list. Starnames are acceptable.

Control Arguments:
-deadproc, -dp
specifies list only dead process directories. If PATH is not given, it checks all directories in the dumps search list for dead processes.
-fdump, -fd
specifies list only fdumps. If PATH is not given, it checks all directories in the dumps search list.

Notes: If no arguments are given, the default is to list only fdumps.

01/19/84 list_processes, lsp

Syntax: lsp {proc_indicator} {-control_argument}
        [lsp {proc_indicator} {-control_argument}]

Function: Lists all known processes in the selected FDUMP.

Function as Active Request:
Returns the process_ids meeting the control argument criteria, returning a null string otherwise. If -count is specified, only the total is returned.

Argument:
proc_indicator
   for specifying individual processes. It can take one of three forms:
   - The decimal index (starting at zero) of a process inthe FDUMP.
   - The octal apte offset of the process.
   - The octal process_id of the process.

Control Arguments:
-all, -a
   Lists all processes in the FDUMP (Default).
-blocked, -blk
   Lists processes marked as blocked.
-count, -ct
   specified by the control_args. With -all, it gives the
   counts of each process state including the overall total.
-current, -cur
    Lists the current process.
-page_tbl_lock, -ptl
    Lists processes marked as page table locking.
-ready, -rdy
    Lists processes marked as ready.
-run
   Lists processes marked as running.
-stopped, -stop
   Lists processes marked as stopped.
-wait
   Lists processes marked as waiting.

Example: do "select_process &1;sdw 0" ([list_processes])
Would display the SDW for DSEG for all processes in the FDUMP.

01/19/84 machine_conditions, mc

Syntax: mc {MC_specifier} {-control_args}

Function: Displays all or parts of Machine Conditions based on the given pointer.

Argument (MC Specifiers):
-dump
   specifies the dump for the BOS CPU regs at time of dump.
-pds {STR1}
   where STR1 can be:
         all
         fim, fim_data
         pgf, page_fault, page_fault_data
         sig, signal, signaller, signal_data
   and defaults to 'all' if STR1 is not given.
-prds {STR2}
   where STR2 can be:
         all
         fim, fim_data
         int, interrupt, interrupt_data
         sys, system_trouble, sys_trouble_data
   and defaults to 'all' if not given.
VIRTUAL-ADDR
   is the virtual address construct used to define the address space
   containing Machine Conditions (see virtual_address.info).
   The virtual address can point to the machine conditons directly or
   it can point to the frame within which the machine conditions reside.
   In the latter case, the offset is calulated for the user.

Control arguments:
-eis
   display the EIS Pointers & Lengths (interpreted).
-faults, -flt
   display the fault register.
-long, -lg
   display all elements of the MC.
-mc_err
   display the mc_err data word.
-misc
   display the miscellaneous data (ie: mc_err, fault reg, time)
-octal, -oc
   display the eis info, scu data, or pointer registers in octal.
   -octal is used in conjunction with -scu, -eis or -regs.
-pointers {PR_LIST}, -prs {PR_LIST}
   displays pointer registers selected by PR_LIST (from 0 to 7,
   separated by spaces). If PR_LIST is not specified, all the pointers
   are displayed.
-ppr
   only display the PSR and IC from the MC.

-registers {REG_LIST}, -regs {REG_LIST}

displays only the basic OU registers. Where REGS_LIST can be any of the following:

x0 x1 x2 x3 x4 x5 x6 x7 a q all.

If REG_LIST is not specified, all of the basic OU registers are displayed.

-scu
display only the scu data of the MC.
-time, -tm
display the MC time.
-tpr
only display the TSR and the CA from the MC.

Notes: If no MC Specifiers are specified, the temporary pointer prmc is used. The default control arguments are:
-fault, -mc_err, -pointers -scu, -time and -tpr.
The machine_conditions request will set all azm-defined temporary pointers as seen in the machine_condition frame.

08/09/83 page_trace, pgt

Syntax: pgt {-control_arg}

Function: displays the contents of the page trace table in the current process data segment (PDS). The default is to display the last 15 trace entries. Trace entries are always displayed in reverse chronological order.

Control Arguments:
-all, -a
   specifies that all trace entries are to be displayed.
-last N, -lt N
   specifies the number of trace entries, where N is a positive decimal integer, to be displayed.

07/26/83 replace, rp

Syntax: replace segno/segname PATH

Function: Replaces the segment designated by segno/segname in the current translation table, with another segment designated by PATH.

Argument:
PATH
is the path name of the segment. The equal convention can be used: replace bound_system_faults [e wd]>=.new

segno/segname
the segment number or segment name within the translation table to be replaced.

Notes: Both per-process and per-system segments can be replaced. For example, if the pds is replaced in a process, it affects only the current process, whereas if tc_data is replaced in a process, it affects the whole FDUMP.

07/27/83 scus

Syntax: scus

Function: Prints the memory address space (in octal) of each scu from the registers saved in the FDUMP.

01/21/85 sdw

Syntax: sdw {segno/name} {segno/name}

Function: Displays the SDW's in the current processes DSEG.

Argument:
segno/name
is the segment number or name of interest. The first is the starting segment number and the second is the ending segment number. If only one is given then only one is displayed if none are given then all are displayed.

SDW Display:
The sdw request displays the segment number, name, memory address, ring brackets, the maximum computed address, the entry bound address and a bit string REWPUGCDF.

Display definitions:
ADDRESS
is the base address of the segment or segment page table.
RNGS
the ring brackets of the segment.
CA-MAX
the highest computed address that may be used in referencing the segment without causing an out_of_segment_bounds fault.
EBOUND
is the entry bound or call limiter. Any external call to this segment must be to an offset less than the EBOUND if the entry bound switch (G) is off.
SEGNO
segment number.
SEGMENT-NAME
segment name.

Display definition of REWPUGCDF:
REWPUGCDF
The letter will show in the sdw display of the segment if the bit is on. The REWPUGCDF string is broken down as follows:
R is the read permission bit
E is the execute permission bit
W is the write permission bit
P is the privileged bit
U is the unpaged bit, segment is unpaged is this is on
G is the gate indicator or entry bound bit. If off, the entry bound is checked by hardware
C is the cache enable switch.
DF is the directed fault bit. If on, the necessary page of the segment is in memory.

Example of SDW display:

```
 ADDRESS RNGS  CA-MAX REWPUGCDF EBOUND SEGNO SEGMENT-NAME
 6262154  000  177777 R W  G DF      0   200 str_seg
       0  000   37777 R W  G         0   300 >udd>Multics>GDixon
```

07/26/83 search, srh

Syntax: search VIRTUAL-ADDR {range} SEARCH_STRING
[search VIRTUAL-ADDR {range} SEARCH_STRING]

Function: This command will search a segment starting at VIRTUAL-ADDR matching on SEARCH_STRING. The search is performed on a 36 bit word boundary. As an active request, the virtual addresses matching the criteria specified is returned.

Argument:
VIRTUAL-ADDR
is the pointer to the address space to search. See virtual_address.info
range
specifies the number of words to be searched from the starting offset, where range is an octal value. The default is the rest of segment. The search is started from VIRTUAL-ADDR.
SEARCH_STRING
This is a 12 character string representing the 12 octal digits that make up a machine word (36 bit,3 bits per digit). This will be used to form both the search data and search mask, by using the hyphen (-) as a "don't care character" in the string. The "do care digits" are octal "0 -> 7". Any other character is illegal.

Examples: To search for:

1) all words in segment 76 that have the last two digits of 43:

```
search 76 ----------43
```

2) all words in tc_data where the upper half = 070707

```
search tc_data 070707------
```

3) words that end in 1234 in sst_seg starting at 1000 but only searching for 200 octal words

```
search sst_seg|1000 200 --------1234
```

4) words that start with 45 and end with 77 starting a sst_seg$ptl for 100 words

```
search sst_seg$ptl 100 45--------77
```

Syntax: name [VIRTUAL-ADDR | number]
[name number] or [name VIRTUAL-ADDR]

Function: Prints the segment name given a virtual address or a segment number.

Argument:
VIRTUAL-ADDR
is the virtual address construct used to define the segment (see virtual_address.info).

number
is the segment number of the segment to be referenced. Thus, "name 230" returns the name associated with the segment number 230, which is "stack_0".

12/12/83 segment_number, number

Syntax: number [VIRTUAL-ADDR | name]
[number name] or [number VIRTUAL-ADDR]

Function: Prints the segment number given either a virtual address or a segment name.

Argument:

VIRTUAL-ADDR
is the virtual address construct used to define the segment (see virtual_address.info).

name
is the name of a segment, e.g., stack_0. Thus, "number sst_seg" returns the segment number associated with the segment sst_seg, which is "77".

10/03/84 select_deadproc, sldp

Syntax: sldp {NAME}

Function: Selects and translates a dead process. Found via the dumps search list. The default path in which deadprocs are found is >dumps>save_pdirs.

Arguments:
NAME
Is the name of the process directory of interest. This can be a relative or absolute pathname. The dead process directory name is of the form person.pdir or person.N.pdir, where N is a numeric number, N=1 for the most recently copied dead process. The suffix "pdir" is assumed if not given.

Notes: When sldp is invoked with no arguments, it prints an identifying message. This is identical to how the select_dump request works.

10/24/84 select_dump, sld

Syntax: sld {NAME} {-control_args}

Function: Selects and translates an FDUMP of a system crash. Found via the dump search list which defaults to >dumps.

Argument:
NAME
is the ERF number or the path name of the zero (0) component of the FDUMP. It can also be the form path>35 where 35 is the erf number. Several control_args are also acceptable if NAME is not specified.

Control arguments:
-first, -ft
selects the first dump (by erf number) in the dump directory found via the dump search list.
-last, -lt
selects the last (most current) dump in the dump directory according to erf number.
-next, -nx
selects the next dump in the dump directory. This is relative to the dump currently being looked at.
-previous, -prev
selects the previous dump in the dump directory. This is relative to the dump currently being looked at.

NOTE:
The sld command will attempt to select the process as indicated by scs$trouble_processid. If this cannot be done the the default will be the first running process found in the dump.

07/20/83 select_process, slp

Syntax: slp {proc_indicator} {-control_argument}

Function: selects a process for examination.

Argument:
proc_indicator
   for specifying individual processes. It can take one of three
   forms:
   - The decimal index (starting at zero) of a process inthe FDUMP.
   - The octal apte offset of the process.
   - The octal process_id of the process.

Control arguments:
-brief, -bf
   suppresses the message about changing processes.
-cpu TAG
   selects the DBR for the process running on the CPU identified
   by TAG (where TAG is one character a -> h).
-dbr dbr_value
   selects the process defined by the dbr_value.
-long, -lg
   prints a message announcing the new process selected (Default).

07/27/83 set

Syntax: set PTR_N VIRTUAL-ADDR

Function: set a internal temporary pointer much like a cpu pointer register (i.e. "pr6" or "sp"). These pointers can then be used as a VIRTUAL-ADDR by other analyze_multics commands.

Argument:
VIRTUAL-ADDR
may be a segment number, name or symbolic address (e.g. 64, prds, prds$am_data). Do a 'help virtual_address' for more detailed information on acceptable virtual-address constructs.
PTR_N
can be either the name or number of a 'temporary pointer'. There are 8 temporary pointers and 2 special case pointers.

number	name	number	name
pr0	ap	pr4	lp
pr1	ab	pr5	lb
pr2	bp	pr6	sp
pr3	bb	pr7	sb
prmc	intended to be a pointer to the current MCs.		
prfr	intended to be a pointer to the current stack frame.		

Examples:

```
set pr6 240|100   this would set a temporary ptr named pr6 (sp).
set sb 240            this would set the temporary ptr (sb) to the
                      base of seg 240 (240|0).
```

Notes:
The value of a temporary pointer can be displayed via the value request: value {ptrn | -all}

08/10/83 stack, sk

Syntax: sk VIRTUAL-ADDR {-control_arguments}

Function: Traces a given stack.

Argument:
VIRTUAL-ADDR
is any segment and offset value that is acceptable to the dump command. See virtual_address.info

Control arguments:
-arguments, -ag
prints the arguments for the stack frames traced.
-for N
will trace for N stack frames. If no valid stack frames exist (stack_begin_ptr = stack_end_ptr), a -fc must be used.
-force, -fc
will force a forward stack trace. To be used when there are no valid frames for this stack (stack_begin_ptr = stack_end_ptr).
-forward, -fwd
will trace in a forward manner.
-long, -lg
will cause the arguments and an octal dump of the stack frames traced.

Notes: The default is to trace the stack in reverse order unless -fc or -fwd are specified. If the VIRTUAL-ADDR has a zero offset then the trace starts at the offset of the first stack (stack_header.stack_begin_ptr). If the VIRTUAL-ADDR has a non-zero offset then the trace is started from that offset in the given stack.

01/21/85 structure_names

This info file lists the structure names to be used for the -as control argument of the display request. Not all structure names are the same as what's used for the -as request. This is due to naming conflicts between different structures, structures containing refer extents, and unmeaningful names.

```
Structure_names(aim_template - bos_dump):
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
aim_template
   aim_template                  aim_template
answer_table
user_table_entry
   anstbl                        answer_table
apte
   apte                          apte
area_structures
   area_header                   area_header
   extend_block                  extend_block
   block                         area_block
ast_lock_meters
   ast_lock_meters               ast_lock_meters
aste
   aste                          aste
bos_dump
   dump                          bos_dump


Structure_names(cdt - dir.168):
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
cdt
   cdt                           cdt
condition_info
   condition_info                condition_info
config_deck
   config_deck                   config_deck
   config_card                   config_card
cmp
   cme                           cme
   cma                           cma
   mcme                          mcme
dbm
   dbm                           dbm
dbr.adp
   dbr.adp                       adp_dbr
dbr.168
   dbr.168                       168_dbr


Structure_names(definition - dir_link):INCLUDE FILE NAME
```

```
    STRUCTURE                     NAME USED IN DISPLAY
definition
   definition                     definition
dir_allocation_area
   area                           dir_allocation_area
dir_acl
   acl_entry                      dir_acl_entry
   access_name                    dir_access_name
dir_entry
   entry                          dir_entry
dir_header
   dir                            dir_header
dir_ht
   hash_table                     dir_hash_table
dir_link
   link                           dir_link


Structure_names(dir_name - dn355_data):
INCLUDE FILE NAME
    STRUCTURE                     NAME USED IN DISPLAY
dir_name
   names                          dir_name
dir_lock_seg
   dir_lock_seg                   dir_lock_seg
   dir_lock_seg_header            dir_lock_seg_header
   dir_lock                       dir_lock
   lock                           fast_lock
disk_table
   dt                             disk_table
   dte                            disk_table_entry
   lve                            disk_table_lv_entry
dn355_data
   datanet_info                   datanet_info
   fnp_info                       fnp_info


Structure_names(dn355_mailbox - dskdcl):
INCLUDE FILE NAME
    STRUCTURE                     NAME USED IN DISPLAY
dn355_mailbox
   datanet_mbx                    datanet_mbx
   sub_mbx                        short_fnp_sub_mbx
   short_fnp_sub_mbx              short_fnp_sub_mbx
   fnp_sub_mbx                    fnp_sub_mbx
   input_sub_mbx                  input_sub_mbx
dskdcl
   disk_data                      disk_data
   disktab                        disktab
   disk_channel_table             disk_channel_table
   chantab                        chantab
   devtab                         devtab
   quentry                        quentry

Structure_names(ect_structures - flagbox):
```

```
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
ect_structures
   ect_header                    ect_header
   wait_channel                  wait_channel
   call_channel                  call_channel
   event_message                 event_message
   itt_message                   itt_message
   event_channel_name            event_channel_name
   fast_channel_name             fast_channel_name
   event_message_data            event_message_data
event_call_info
   event_call_info               event_call_info
event_wait_info
   event_wait_info               event_wait_info
fault_vector
   fv                            fault_vector
flagbox
   fgbx                          fgbx


Structure_names(fs_vol_label - io_status_entry):
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
fs_vol_label
   label                         disk_label
int_unpaged_page_tables
   iupt                           iupt
   iupte                          iupte
io_page_tables
   io_page_tables                io_page_tables
   io_ptw                        io_ptw
   io_ptw                        io_page_table_256
   io_ptw                        io_page_table_64
io_special_status
   io_special_status             io_special_status
io_status
   status                        io_status
io_status_entry
   io_status_entry               io_status_entry
   io_status_word                io_status_word


Structure_names(io_syserr_msg - iom_data):
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
io_syserr_msg
   io_msg                        io_msg
iocbx
   iocb                          iocb
ioi_data
   ioi_data                      ioi_data
   gte                           gte   cte                          cte
   dte                           dte
iom_data
```

```
   iom_data                        iom_data
   per_device                      iom_per_device
   per_iom                         iom_per_iom
   iom_mailbox_seg                 iom_mailbox_seg
   iom_mailbox                     iom_mailbox
   channel_mailbox                 channel_mailbox


Structure_names(iom_dcw - its):
INCLUDE FILE NAME
    STRUCTURE                      NAME USED IN DISPLAY
iom_dcw
   dcw                             iom_ddcw
   tdcw                            iom_tdcw
iom_lpw
   lpw                             iom_lpw
   lpw_ext                         iom_lpw_ext
iom_pcw
   pcw                             iom_pcw
   idcw                            iom_idcw
iom_scw
   scw                             iom_scw
its
   its                             its
   its_unsigned                    its_unsigned
   itp                             itp
   itp_unsigned                    itp_unsigned


Structure_names(itt_entry - linkdcl):
INCLUDE FILE NAME
    STRUCTURE                      NAME USED IN DISPLAY
itt_entry
   itt_entry                       itt_entry
kst
   kst                             kst
   kste                            kste
lct
   lct                             lct
   lcte                            lcte
linkdcl
   link                            link_pair;
   exp_word                        link_exp_word
   type_pair                       link_type_pair
   header                          linkage_header
   virgin_linkage_header           virgin_linkage_header
   trap_word                       link_trap_word


Structure_names(lot - mcs_modes_change_list):
INCLUDE FILE NAME
    STRUCTURE                      NAME USED IN DISPLAYlot
   lot                             lot
   isot                            isot
   isotl                           isotl
```

```
lvt
   lvt                              lvt
   lvte                             lvte
mc
   mc                               mc
   scu                              scu
   scux                             scux
mc_trace_buf
   mc_trace_buf                     mc_trace_buf
mcs_modes_change_list
   mcl                              modes_change_list
   mcle                             modes_change_list_entry


Structure_names(mc_trace_data - oc_data):
INCLUDE FILE NAME
    STRUCTURE                       NAME USED IN DISPLAY
mcs_trace_data
   trace_array                      mcs_trace_array
   trace_entry                      mcs_trace_entry
mdcs
   mdcs                             mdcs
   mdirent                          mdcs_mdir
   acctent                          mdcs_account
   mdcs_path                        mdcs_path
mstr
   mst_label                        mst_label
   mstr_header                      mstr_header
   mstr_trailer                     mstr_trailer
   volume_identifier                mst_volume_id
   mst_volume_id                    mst_volume_id
oc_data
   oc_data                          oc_data


Structure_names(oc_log_meters - pv_holdt):
INCLUDE FILE NAME
    STRUCTURE                       NAME USED IN DISPLAY
oc_log_meters
   olm                              oc_log_meters
pathname_am
   pam                              pathname_am
pcb
   pcd                              pcd
pitmsg
   pit                              pit
ptw.adp
   adp_core_ptw                     adp_core_ptw
   adp_ptw                          adp_ptw
ptw.168
   168_core_ptw                     168_core_ptw   168_ptw                    168_ptw
pv_holdt
   pv_holdt                         pv_holdt
```

```
Structure_names(pvt - rnt):
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
pvt
   pvt                           pvt
pvte
   pvte                          pvte
   pvt_array                     pvt_array
rcp_com_seg
   rcs                           rcs
   rcse                          rcse
rcp_data
   rcpd                          rcpd
   dtype                         rcp_dtype
   device                        rcp_device
   volume                        rcp_volume
rnt
   rnt                           rnt
   rnte                          rnte


Structure_names(sdw.adp - slt):
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
sdw.adp
   adp_sdw                       adp_sdw
sdw.168
   168_sdw                       168_sdw
sdw_info
   sdw_info                      sdw_info
segdamage_msg
   segdamage_msg                 segdamage_msg
signaller_stack
   signaller_stack               signaller_stack
slt
   slt                           slt
   name_seg                      slt_name_seg
   segnam                        slt_segname
   path                          slt_path
   acls                          slt_acls


Structure_names(slte - str):
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
slte
   slte                          slte
sst
   sst                           sst
   sstnt                         sstntstack_0_data
   sdt                           stack_0_data
   sdte                          sdte
stack_frame
   stack_frame                   stack_frame
stack_header
```

```
   stack_header                  stack_header
str
   str                           segment_trailer


Structure_names(syserr_data - wct_entry):
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
syserr_data
   sd                            syserr_data
   wlog                          wired_syserr_log
   wmess                         wired_syserr_message
syserr_log
   slog                          syserr_log
   smess                         syserr_message
tcb
   tcb                           tcb
tcm
   tcm                           tc_data
   tcm                           tcm
   wct_entry                     wct_entry


Structure_names(tty_buf - volume_registration):
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
tty_buf
   tty_buf                       tty_buf
tty_buffer_block
   buffer                        tty_buffer
   free_block                    free_tty_buffer
tty_tables
   tty_tables_hdr                tty_tables_hdr
unpaged_page_tables
   upt                           upt
   upt_entry                     upt_entry
vol_map
   vol_map                       vol_map
volume_registration
   volume_registration           volume_registration


Structure_names(vtoc_buffer - wtcb):
INCLUDE FILE NAME
    STRUCTURE                    NAME USED IN DISPLAY
vtoc_buffer
   vtoc_buffer                   vtoc_buffer
vtoc_header
   vtoc_header                   vtoc_headervtoce
   vtoce                         vtoce
wire_proc_data
   wpd                           wire_proc_data
wtcb
   wtcb                          wtcb
```

01/21/85 syserr_log, slog

Syntax: syserr_log {-control_args}

Function: Displays all or parts of the syserr_log and syserr_data segments from the dump. It does not examine the perm_syserr_log. The default is to print the entire log and all actions.

Control arguments:

-action A
   displays messages starting at severity -100 and up to the action code specified by A, where A is a decimal integer between -100 and 100. A range can also be specified, consisting of a pair decimal integers separated by a colon ("20:29").

-exclude STR -ex STR
   where STR is a string that is matched against the log, as for -match. Any message that contains STR is not printed.

-last N, -lt N
   where N is a decimal integer. This argument specifies that the scan is to start N messages back from the end of the log.

-match STR
   where STR is a string to be matched against messages in the log. Any message that contains STR is a candidate to be printed.

-expand, -exp
   specifies that messages that have binary data will have that binary data interpreted. The format is generally dependent on the text of the message.

08/23/83 traffic_control_queue, tcq

Syntax: tcq {-control_args}

Function: Displays process DBR, process state, process ID, current CPU and USERID from the Traffic Controllers Eligible Queue, as well as the "process number" in the FDUMP. The default is to display only the eligible queue.

Control Arguments:

-all
Displays the eligible, realtime, interactive and workclass queue entries, including the unthreaded entries.

-ready, -rdy
Displays the eligible, realtime, interactive and workclass queues, excluding the unthreaded entries.

07/27/83 value, v

Syntax: v PTR_Ni..PTR_Nn | -all

Function: Displays the current value of one or all of the temporary pointers.

Argument:
PTR_N
   specifies which of the temporary pointers is to be displayed. Refer to the set request for a list of the azm defined pointer names. User-defined pointers can also be specified.
-all, -a
   specifies that all of the pointers are to be displayed (Default).

07/27/83 verify_associative_memory, vfam

Syntax: vfam {-control_args}
[vfam {-control_args}]

Function: Performs a consistency check on the Associative Memories stored at the time of a dump by comparing them to the appropriate entries in the "dump dseg" and page tables. When used as an active request returns "true" if any inconsistencies are found, "false" otherwise.

Control Arguments:
-ptw
restricts the verification to the PTW Associative Memories.
-sdw
restricts the verification to the SDW Associative Memories.

Notes: If no argument is given BOTH SDW and PTW Associative Memories are checked.

12/12/82 Virtual Address Constructs

Accessing data requires some pointer value to define an address space. The generation of the pointer value is performed by resolving a virtual address (VIRTUAL-ADDR). A VIRTUAL-ADDR consist of two parts, a segment number and a word offset.

Analyze_multics (azm) will resolve VIRTUAL-ADDR'S from the following types of information:

Symbols:
is a symbolic name for a segment number and an offset (i.e., sst$ptl can be resolved to the correct segment number and offset of the page table lock).

Segment name:
a segment name can be resolved in many ways, but it can only provide one part of the virtual address; azm uses 0 as the default offset for this pointer value (i.e., tc_data is resolved to SEGNO|0).

Segment number:
a segment number needs no resolution, but a default action needs to be taken for the offset (the default is 0, i.e., SEGNO|0).

Segment name/number and offset:
The VIRTUAL-ADDR in this case can be a segment name or segment number and an octal offset (i.e., the construct of pds|20 is translated to SEGNO|20 or dseg|5 is 0|5). The notation "|" and "$" must be used without spaces (e.g., 244|0 or sst$cmp).

Temporary pointers:
azm keeps a set of 11 temporary pointers per translation. A translation is one complete entity such as an "FDUMP". These pointers can be set with the set request (e.g., set sp 230|100). They can be referenced by other requests as another type of "symbol" in a VIRTUAL-ADDR expression, after they have been set. If not set, these pointers are null.

Offset Operators:
The operators "+N" and "-N" immediately preceding an octal number, or VIRTUAL-ADDR construct can be used to alter the offset of a virtual address. N is a number interpreted in octal. No spaces are allowed between the operator and the N. For example, sst$ptl +30 are resolved to be the SEGNO for sst_seg with the offset of ptl plus 30 octal locations; sst$ptl+30 is also valid.

Indirection: A VIRTUAL-ADDR can imply indirection. The indirect word can be use

as an ITS pair if it is a valid ITS word pair; if not, the upper half of the word is used. The following VIRTUAL-ADDR construct is used to specify indirection (sst$cmp,*). The format of an indirect pointer value is:

```
segno|offset,*      segname|offset,*      symbol,*
temp_ptr,*          temp_ptr|offset,*
```

Examples of indirection:

```
17|230,*            sst|230,*            sst$cmp,*+2
sp,*                sp|230,*
```

08/15/83 why

Syntax: why

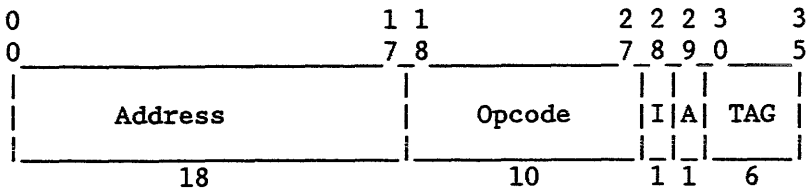
Function: The why request will try to find the stack that has a call to syserr_real$syserr_real or call_bce$call_bce and set the temporary pointers, pr6 and prfr, to the stack frame. This command will search the stacks for a frame that has a return_to_ring_0_ frame and set the temporary pointers from this set of machine conditions that called this entry.

Notes: If the crash was due to fim_util$check_fault finding a problem, the machine condition CU data is displayed and all temporary pointers are set from these machine conditions. If this was an Execute Fault then some lock info is printed an the process selected is lock ordered. First sst_seg$ptl followed by sst_seg$aslt then scs$connect_lock next tty_buf$slock and last tty_buf$timer_lock.

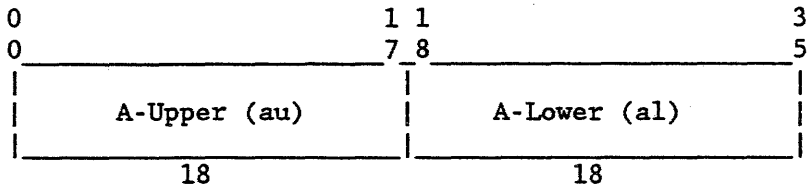
If this fdump was due to a manual return to BOS then some pertinent lock info will also be printed.

DATA FORMATS

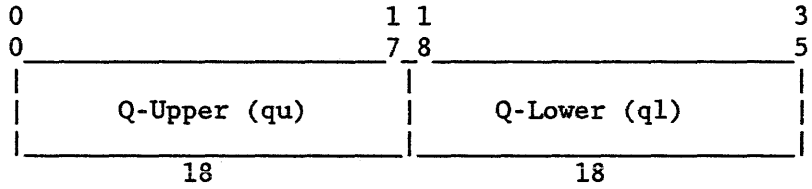
INSTRUCTION FORMAT



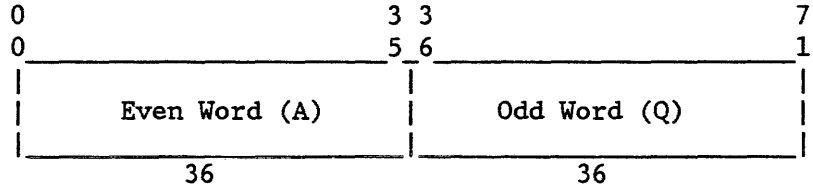
ACCUMULATOR REGISTER (A)



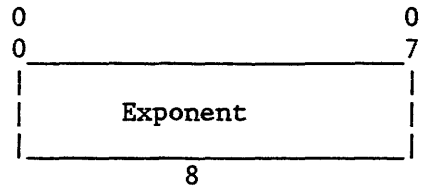
QUOTIENT REGISTER (Q)



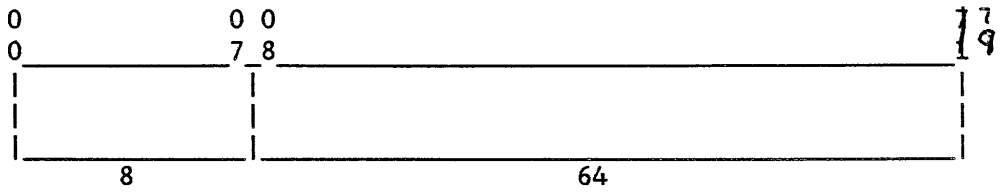
ACCUMULATOR-QUOTIENT REGISTER (AQ)



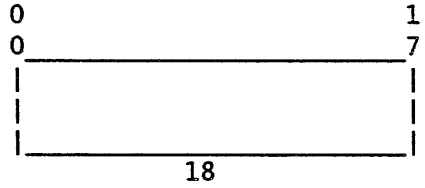
EXPONENT REGISTER (E)



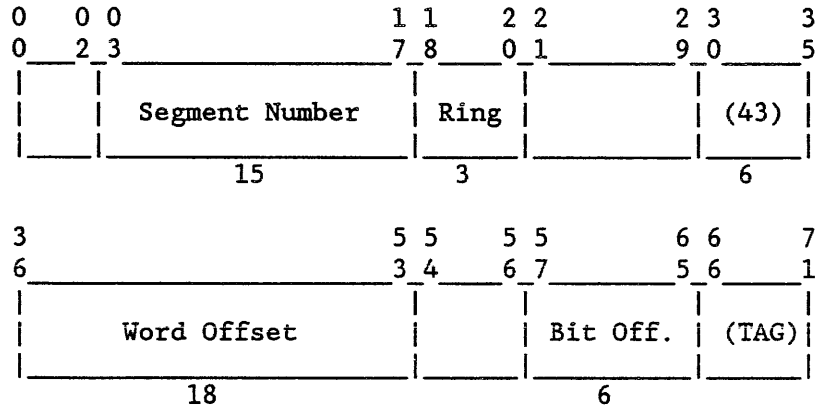
EXPONENT-ACCUMULATOR-QUOTIENT REGISTER (EAQ)



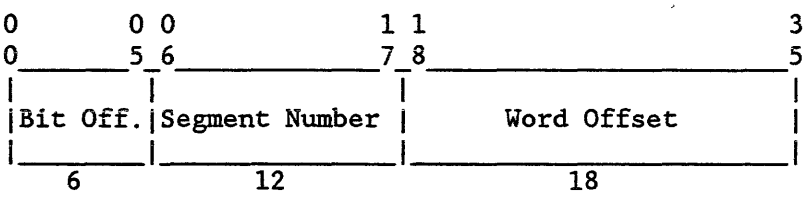
INDEX REGISTER (Xn)



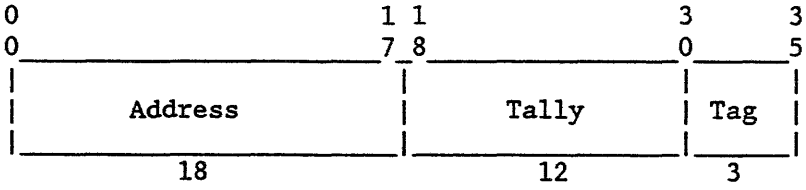
POINTER REGISTER (PRn)
or
ITS POINTER FORMAT



PACKED POINTER FORMAT



INDIRECT WORD FORMAT



```
/* format: style4,indattr,ifthen,^indproc */

/* Demonstrate that an uninitialized ptr may be treated as an indirect
   word, causing accidental modifications in stack. */

indirect_word_demo: proc;

dcl  uninitialized_pointer  ptr;
dcl  basedvar               based (uninitialized_pointer);
dcl  stackbaseptr           builtin;
dcl  stack_first_word       fixed bin (35) based (stackbaseptr ());
dcl  ioa_                   entry () options (variable);

          unspec (uninitialized_pointer) = "0"b;
          call ioa_ ("Word referenced as basedvar: ^w", basedvar);
          call ioa_ ("First word of stack: ^w", stack_first_word);
          call ioa_ ("Changing basedvar to -1.");
          basedvar = -1;
          call ioa_ ("Word referenced as basedvar: ^w", basedvar);
          call ioa_ ("First word of stack: ^w", stack_first_word);

     end indirect_word_demo;
```

```
indirect_word_demo
Word referenced as basedvar: 000000000004
First word of stack: 000000000004
Changing basedvar to -1.
Word referenced as basedvar: 777777777777
First word of stack: 777777777777
```

OCTAL FORMATS OF COMMON DATATYPES

ITS Pointer

000237400043 014040000000

Null pointer

077777000043 000001000000

Unsnapped Link

776772000046 000071000000

Packed Pointer

000237033036

ASCII

056165163145 162137157165 164160165164 040040040040

Fixed Binary 1

000000000001

Fixed Binary -1

777777777777

Fixed Binary 0

000000000000

Clock value

000000113267 753151076614

UIDs

101731476312
132643613643

Process ID

025700556327

Instructions

600534252100 600504621100 030000431007 600044370120
400062352120 000622700100 600303236100 600534252100

Processor Instructions

Mnemonic	Meaning
a4bd	Add 4-bit character displacement to AR
a6bd	Add 6-bit character displacement to AR
a9bd	Add 9-bit character displacement to AR
aarN	Alphanumeric descriptor to ARn
abd	Add bit displacement to AR
absa	Abs address to A register
ad2d	Add using two decimal operands
ad3d	Add using three decimal operands
ada	Add to A register
adaq	Add to AQ register
ade	Add to E register
adl	Add low to AQ register
adla	Add logical to A register
adlaq	Add logical to AQ register
adlq	Add logical to Q register
adlxN	Add logical to index N
adq	Add to Q register
adwpN	Add to word numeric field of PRn
adxN	Add to index N
alr	A register left rotate
als	A register left shift
ana	AND to A register
anaq	AND to AQ register
anq	AND to Q register
ansa	AND to storage from A register
ansq	AND to storage from Q register
ansxN	AND to storage from index N
anxN	AND to index N
aos	Add one to storage
araN	ARn to alpha descriptor
arl	A register right logical shift
arnN	ARn to numeric descriptor
ars	A register right shift
asa	Add stored to A register
asq	Add stored to Q register
asxN	Add stored to index N
awca	Add with carry to A register
awcq	Add with carry to Q register
awd	Add word displacement to AR
bcd	Binary-to-BCD
btd	Binary-to-dec
call6	Call
camp	Clear associative memory paged
cams	Clear associative memory segmented
cana	Comparative and with A register
canaq	Comparative and with AQ register
canq	Comparative and with Q register
canxN	Comparative and with index N
cioc	Connect
cmg	Compare magnitude

cmk	Compare masked
cmpa	Compare with A register
cmpaq	Compare with AQ register
cmpb	Compare bit strings
cmpc	Compare alphanumeric character strings
cmpn	Compare numeric
cmpq	Compare with Q register
cmpxN	Compare with index N
cnaa	Comparative not with A register
cnaaq	Comparative not with AQ register
cnaq	Comparative not with Q register
cnaxN	Comparative not with index N
csl	Combine bit strings left
csr	Combine bit strings right
cwl	Compare with limits
dfad	DP floating add
dfcmg	DP floating compare magnitude
dfcmp	DP floating compare
dfdi	DP floating divide inverted
dfdv	DP floating divide
dfld	DP floating load
dfmp	DP floating multiply
dfrd	DP floating round
dfsb	DP floating subtract
dfst	DP floating store
dfstr	DP floating store rounded
dis	Delay until interrupt signal
div	Divide integer
drl	Derail
dtb	Dec-to-binary convert
dufa	DP unnormalized floating add
dufm	DP unnormalized floating multiply
dufs	DP unnormalized floating subtract
dv2d	Divide using two decimal operands
dv3d	Divide using three decimal operands
dvf	Divide fraction
eaa	Effective address to A register
eaq	Effective address to Q register
easpN	Effective address to segment number field of PRn
eawpN	Effective address to word and bit fields of PRn
eaxN	Effective address to index N
epaq	Effective pointer to AQ register
epbpN	Effective pointer at base to PRn
eppN	Effective pointer to PRn
era	XOR to A register
eraq	XOR to AQ register
erq	XOR to Q register
ersa	XOR to storage with A register
ersq	XOR to storage with Q register
ersxN	XOR to storage with index N
erxN	XOR to index N
fad	Floating add
fcmg	Floating compare magnitude
fcmp	Floating compare
fdi	Floating divide inverted

fdv	Floating divide
fld	Floating load
fmp	Floating multiply
fneg	Floating negate
fno	floating Normalize
frd	Floating round
fsb	Floating subtract
fst	Floating store
fstr	Floating store rounded
fszn	Floating set zero and negative indicators
gtb	Gray-to-binary convert
larN	Load ARn
lbar	Load address registeristers
lca	Load complement into A register
lcaq	Load complement into AQ register
lcpr	Load central processor register
lcq	Load complement into Q register
lcxN	Load complement into index N
lda	Load A register
ldac	Load A register and clear
ldaq	Load AQ register
ldbr	Load descriptor base register
lde	Load E register
ldi	Load indicator register
ldq	Load Q register
ldqc	Load Q register and clear
ldt	Load timer register
ldxN	Load index N
llr	Long left rotate
lls	Long left shift
lpl	Load pointers and lengths
lpri	Load pointer registers from ITS pairs
lprpN	Load pointer register N from packed pointer
lptp	Load page table pointers
lptr	Load page table registers
lra	Load ring alarm register
lreg	Load registers
lrl	Long right logical
lrs	Long right shift
lsdp	Load segment descriptor pointers
lsdr	Load segment descriptor registers
lxlN	Load index N from lower
mlr	Move alphanumeric left to right
mme1	Master mode entry 1
mme2	Master mode entry 2
mme3	Master mode entry 3
mme4	Master mode entry 4
mp2d	Multiply using two decimal operands
mp3d	Multiply using three decimal operands
mpf	Multiply fraction
mpy	Multiply integer
mrl	Move alphanumeric right to left
mve	Move alphanumeric edited
mvn	Move numeric
mvne	Move numeric edited

mvt	Move alphanumeric with translation
narN	Numeric descriptor to ARn
neg	Negate (A register)
negl	Negate long (AQ registerister)
nop	No operandation
ora	Or to A register
oraq	Or to AQ register
orq	Or to Q register
orsa	Or to storage from A register
orsq	Or to storage from Q register
orsxN	Or to storage from index N
orxN	Or to index N
puls1	Pulse location 1
puls2	Pulse location 2
qlr	Q register left rotate
qls	Q register left shift
qrl	Q register right logical shift
qrs	Q register right shift
rccl	Read calendar clock
rcu	Restore control unit
ret	Return
rmcm	Read memory controller mask
rpd	Repeat double
rpl	Repeat link
rpt	Repeat
rscr	Read system controller register
rsw	Read switches
rtcd	Return control double
s4bd	Subtract 4-bit displacement from AR
s6bd	Subtract 6-bit displacement from AR
s9bd	Subtract 9-bit displacement from AR
sarN	Store ARn
sareg	Store address registers
sb2d	Subtract using two decimal operands
sb3d	Subtract using three decimal operands
sba	Subtract from A register
sbaq	Subtract from AQ register
sbar	Store base address register
sbd	Subtract bit displacement from AR
sbla	Subtract logical from A register
sblaq	Subtract logical from AQ register
sblq	Subtract logical from Q register
sblxN	Subtract logical from index N
sbq	Subtract from Q register
sbxN	Subtract from index N
scd	Scan character double
scdr	Scan character double reverse
scm	Scan with mask
scmr	Scan with mask reverse
scpr	Store central processor register
scu	Store control unit
sdbr	Store descriptor base register
smcm	Set memory controller mask
smic	Set memory interrupt cells
spbpN	Store segment base pointer of PRn

spl	Store pointers and lengths
spri	Store pointer registers as ITS pairs
spriN	Store PRn as an ITS pair
sprpN	Store pointer register N packed
sptp	Store page table pointers
sptr	Store page table registers
sra	Store ring alarm register
sreg	Store registers
ssa	Subtract stored from A register
sscr	Set system controller register
ssdp	Store segment descriptor pointers
ssdr	Store segment descriptor registers
ssq	Subtract stored from Q register
ssxN	Subtract stored from index N
sta	Store A register
stac	Store A register conditional
stacq	Store A register conditional on Q register
staq	Store AQ register
stba	Store 9-bit characters of A register
stbq	Store 9-bit characters of Q register
stc1	Store instruction counter + 1
stc2	Store instruction counter + 2
stca	Store 6-bit characters of A register
stcd	Store control double
stcq	Store 6-bit characters of Q register
ste	Store E register
sti	Store indicator register
stq	Store Q register
stt	Store timer register
stxN	Store index N
stz	Store zero
swca	Subtract with carry from A register
swcq	Subtract with carry from Q register
swd	Subtract word displacement from AR
sxlN	Store index N in lower
szn	Set zero and negative indicators
sznc	Set zero and negative indicators and clear
sztl	Set zero and truncation indicators with bit string left
sztr	Set zero and truncation indicators with bit string right
tct	Test character and translate
tctr	Test character and translate reverse
teo	Transfer on exponent overflow
teu	Transfer on exponent underflow
tmi	Transfer on minus
tmoz	Transfer on minus or zero
tnc	Transfer on no carry
tnz	Transfer on nonzero
tov	Transfer on overflow
tpl	Transfer on plus
tpnz	Transfer on plus and nonzer
tra	Transfer
trc	Transfer on carry
trtf	Transfer on truncation indicator off
trtn	Transfer on truncation indicator on
tspN	Transfer and set PRn

tss	Transfer and set slave
tsxN	Transfer and set index N
ttf	Transfer on tally indicator off
ttn	Transfer on tally indicator on
tze	Transfer on zero
ufa	Unnormalized floating add
ufm	Unnormalized floating multiply
ufs	Unnormalized floating sub
xec	Execute
xed	Execute double

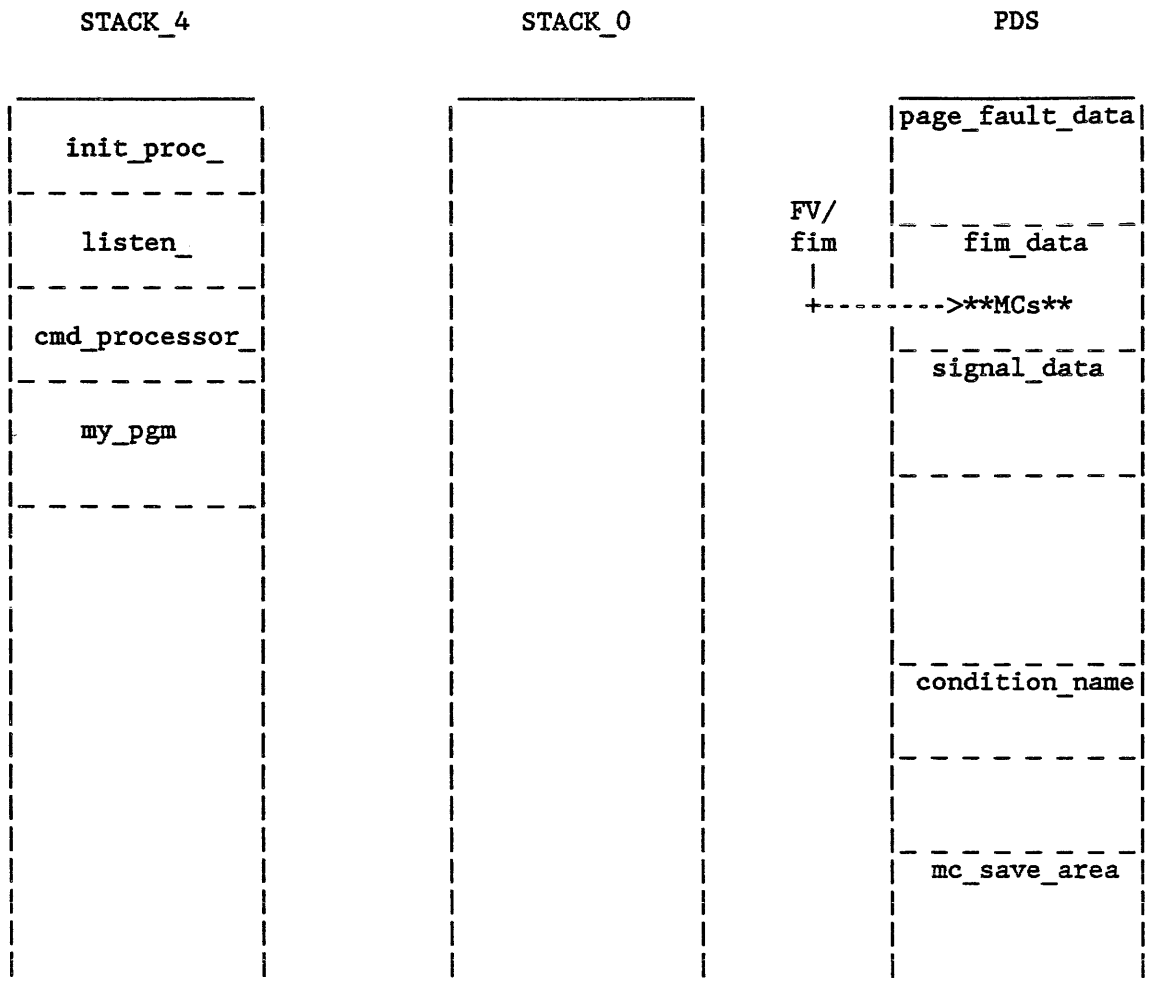
SIGNALLING AND CRAWLOUTS

RING 4 FAULT EXAMPLE

1. PROGRAM MY_PGM EXECUTING IN RING 4

RING 4 FAULT EXAMPLE

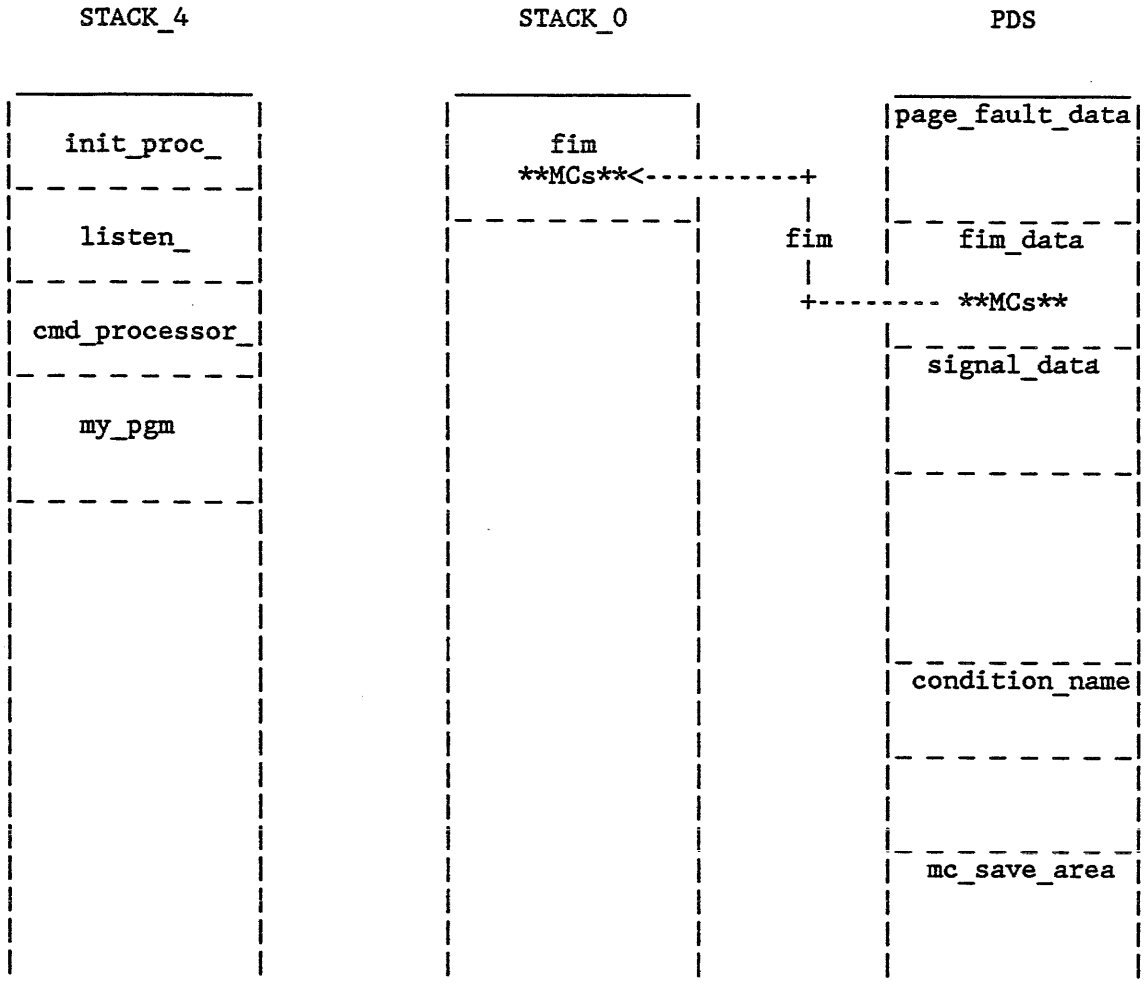
2. PARITY FAULT OCCURS



A Parity Fault occurs while executing an instruction of my_pgm. Transfer via Fault Vector to fim, storing Machine Conditions in pds$fim_data.

RING 4 FAULT EXAMPLE

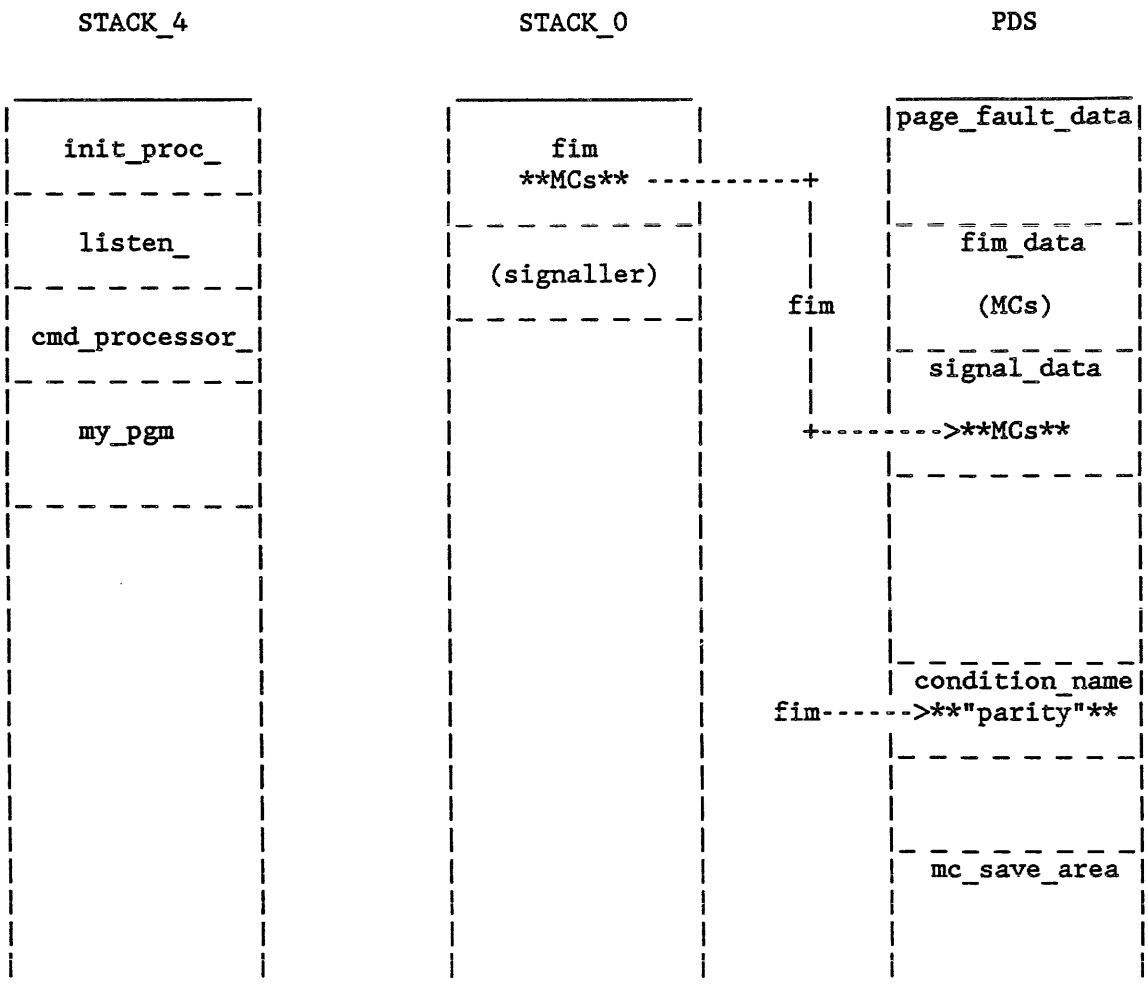
3. FIM BEGINS FAULT HANDLING



Fim pushes a stack frame and copies the Machine Conditions into it at offset 60 (octal), turning on the signal bit in the stack frame to indicate that it is a "FIM Frame".

RING 4 FAULT EXAMPLE

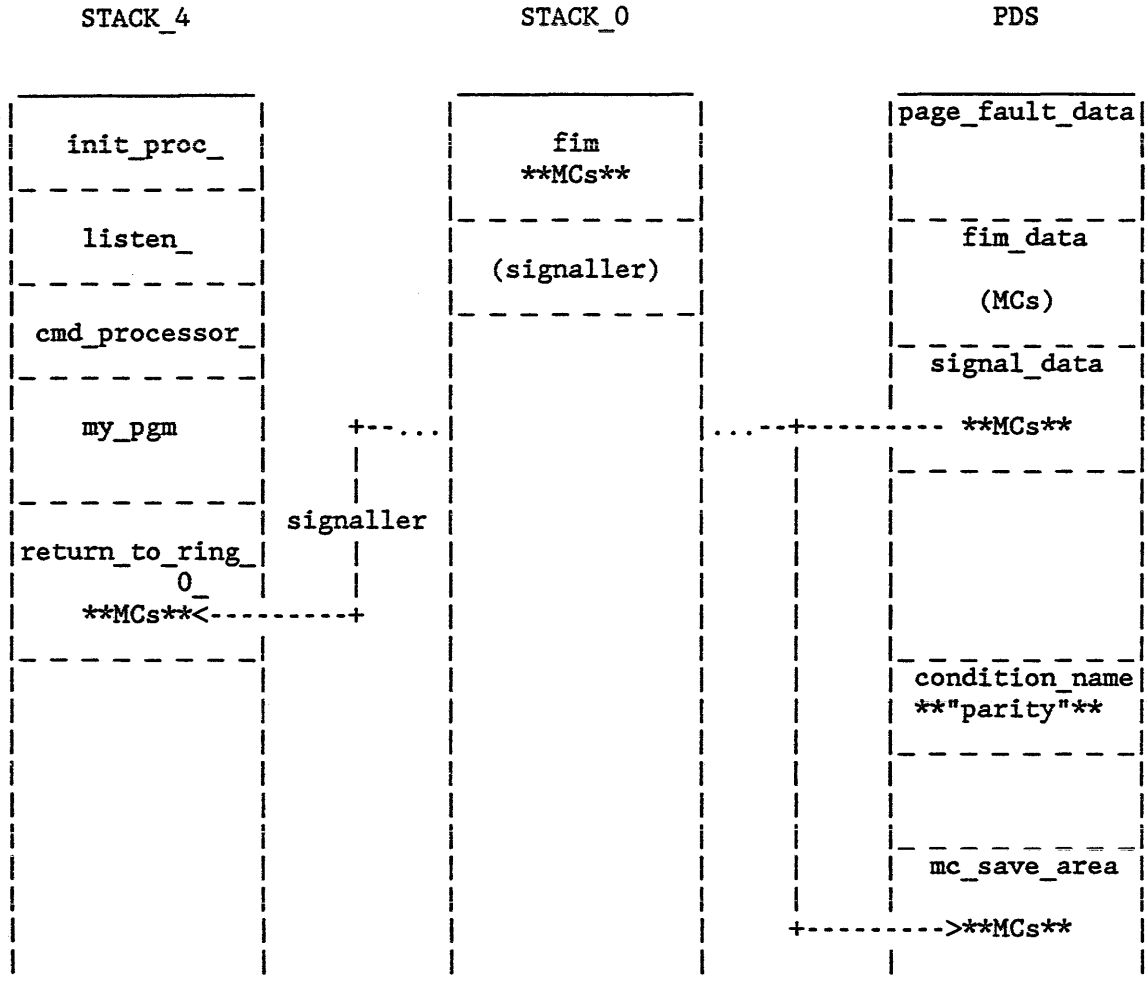
4. FIM DECIDES TO SIGNAL CONDITION



Fim decides to signal a condition as a result of the fault. To do so it copies the Machine Conditions to pds$signal data, puts the condition name in pds$condition_name, and calls signaller. (Signaller is shown on the Ring 0 stack to indicate that it is active, but in reality it does not push a stack frame.)

RING 4 FAULT EXAMPLE

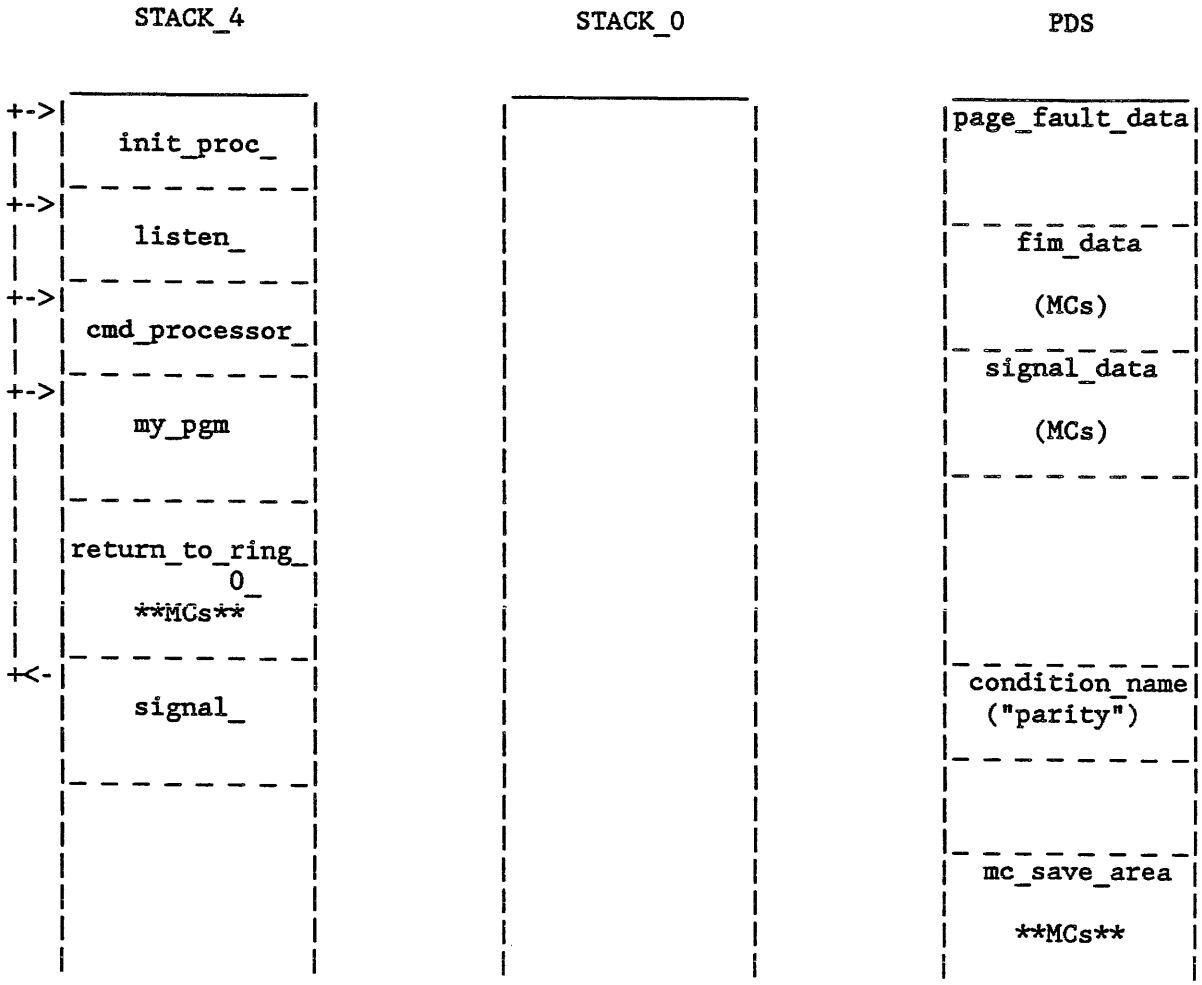
5. SIGNALLER FINDS ORIGINAL STACK



Signaller uses the Machine conditions in pds$signal_data to find the stack the process was using when the fault occurred. It adds a FIM frame to that stack and copies the Machine Conditions there. The Machine Conditions are also copied into a slot in pds$mc_save_area, to be used later if the fault is restarted.

RING 4 FAULT EXAMPLE

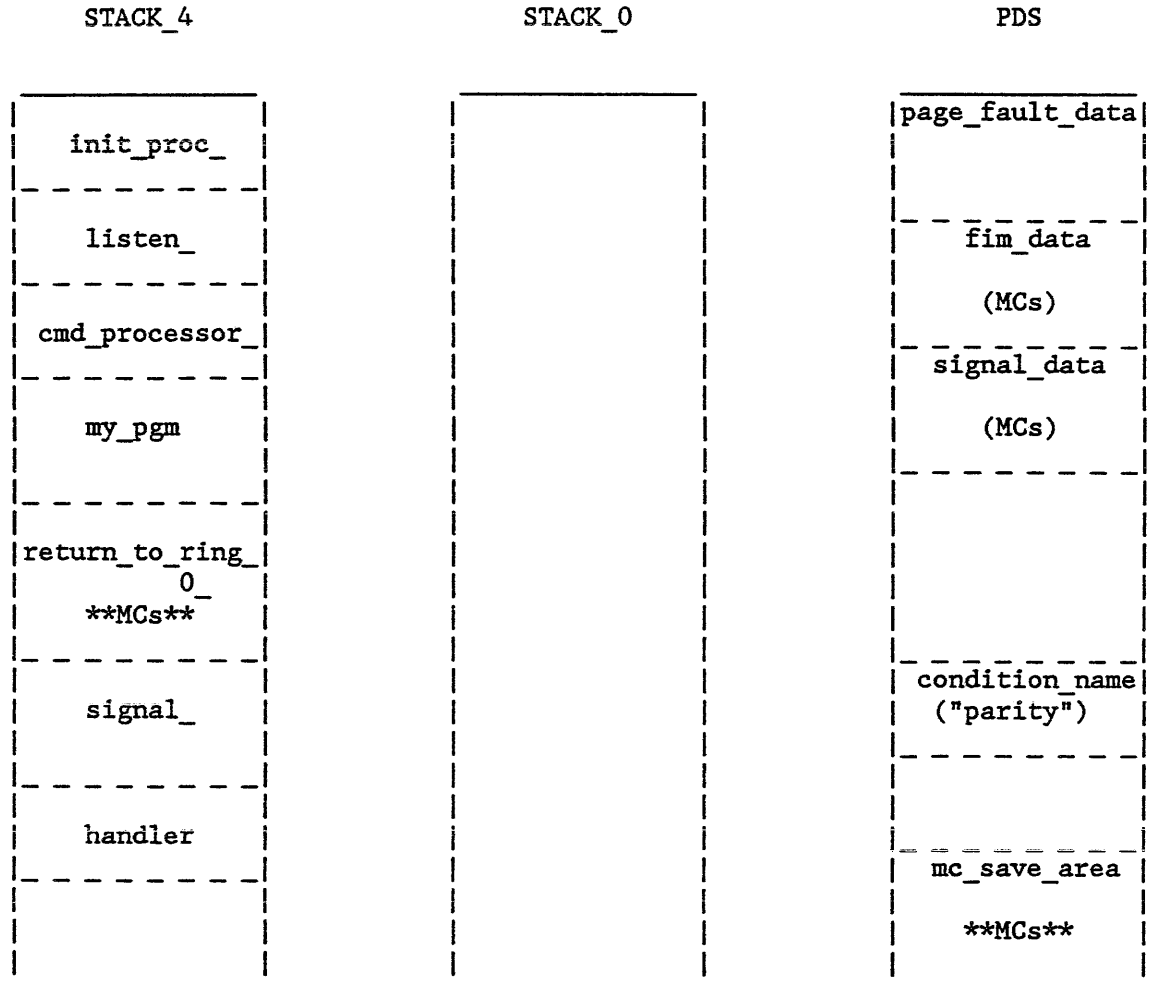
6. SIGNAL_ LOOKS FOR A CONDITION HANDLER



Signaller transfers control to signal_ in the original ring. Signal_ pushes a stack frame and then searches back through the stack frames looking for a handler.

RING 4 FAULT EXAMPLE

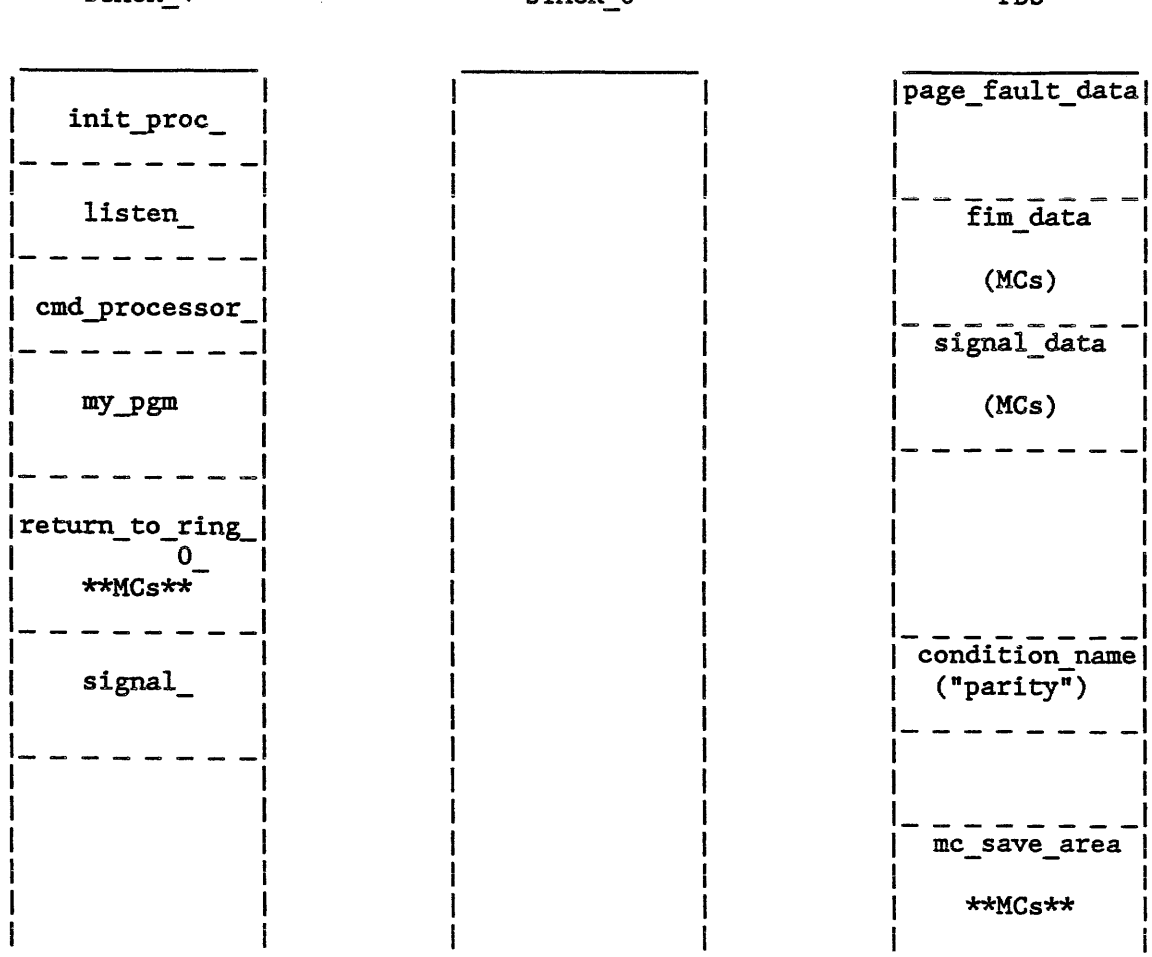
7. SIGNAL_ CALLS CONDITION HANDLER



Signal_ invokes the condition handler.

FAULT RESTART EXAMPLE

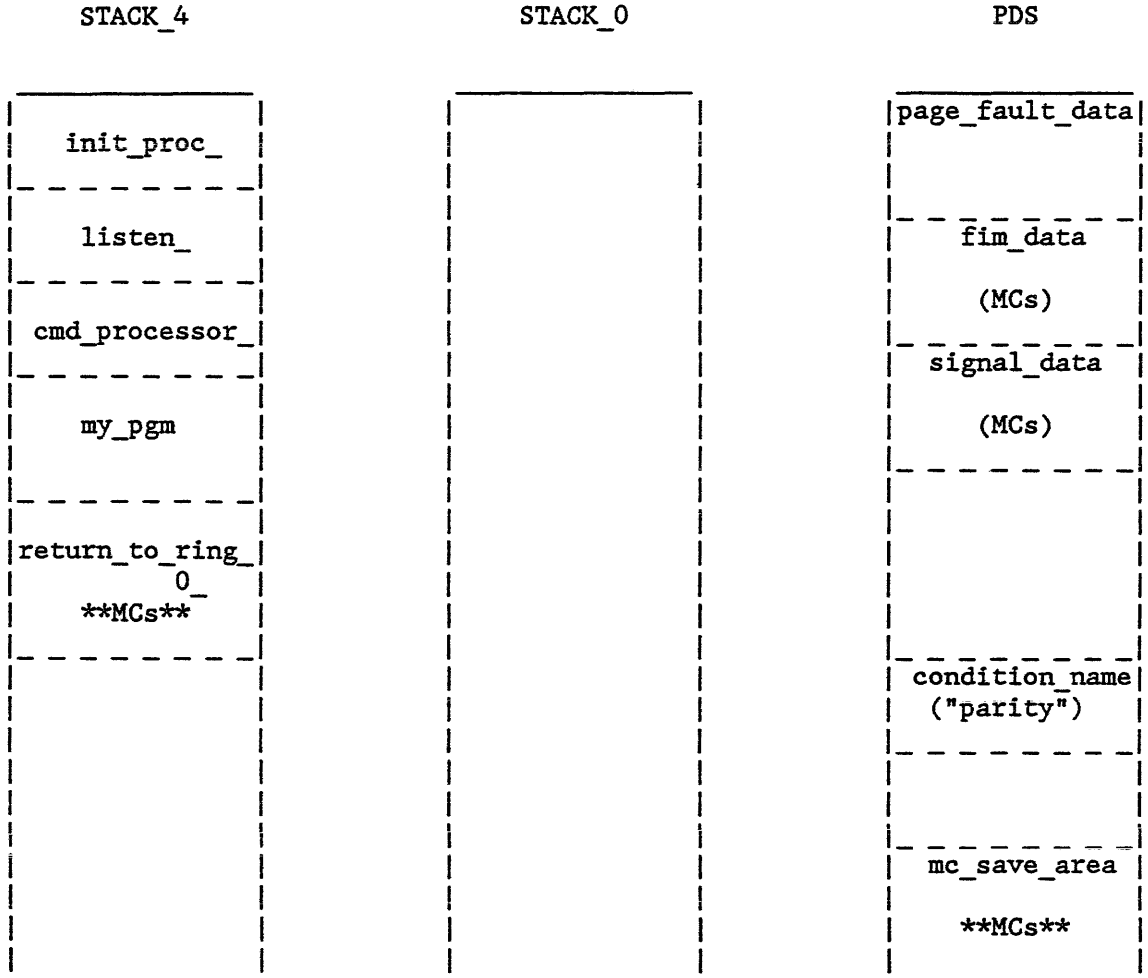
1. HANDLER DECIDES TO RESTART FAULT



Handler returns to its caller, i.e. to signal_.

FAULT RESTART EXAMPLE

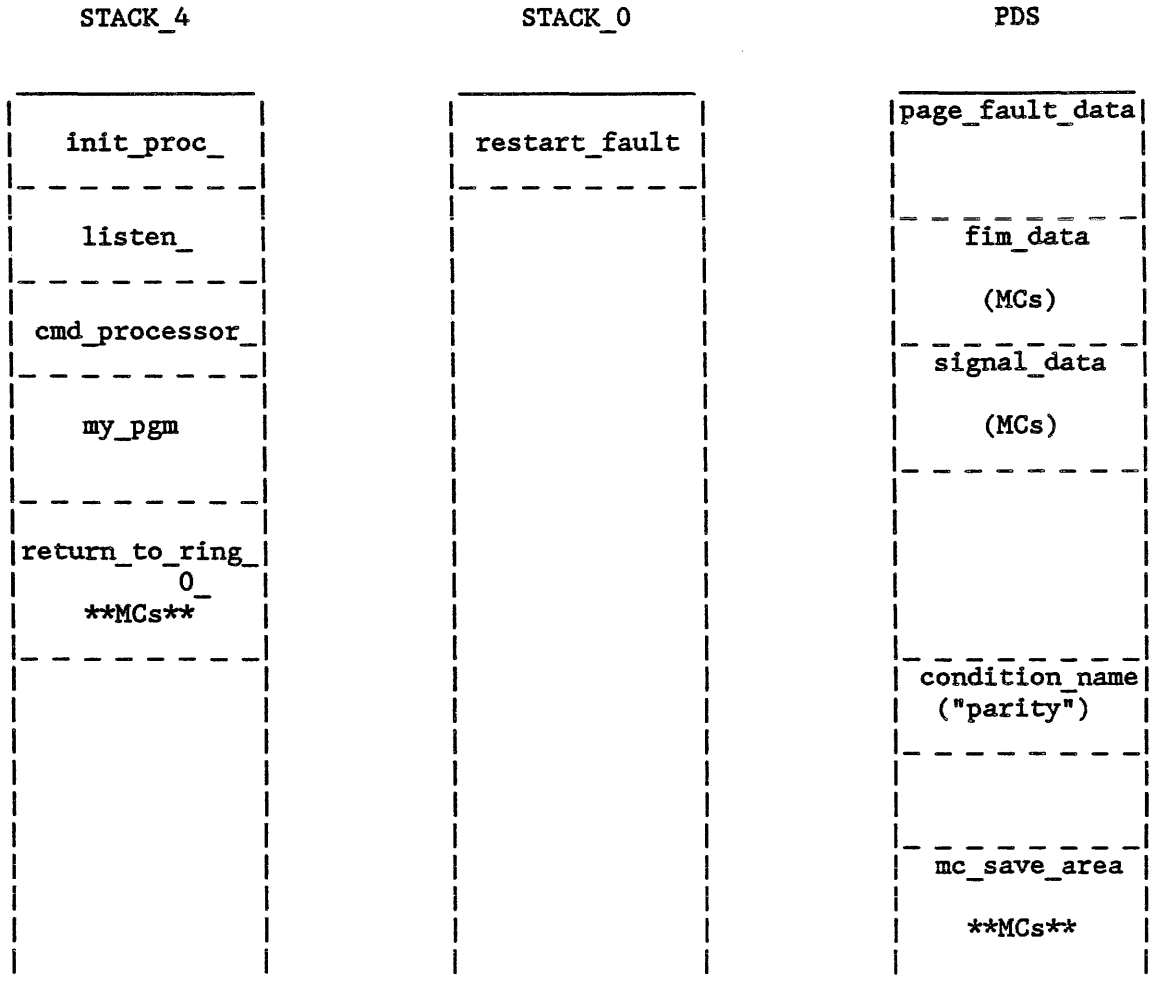
2. SIGNAL_ "RETURNS" TO RETURN_TO_RING_0_



Signal_ performs a normal return, i.e. it transfers control to the address indicated in the return pointer of the previous stack frame. The previous stack frame was actually created by signaller, which called signal_. However, signaller put in a return pointer that points not to signaller, but to return_to_ring_0_. Thus, signal_ transfers control to return_to_ring_0_. Return_to_ring_0_ is a non-deciduous hardcore program with ring brackets of 0 0 7. It is therefore a gate into ring 0. All that it does when it starts executing in ring 0 is to call the ring 0 program restart_fault.

FAULT RESTART EXAMPLE

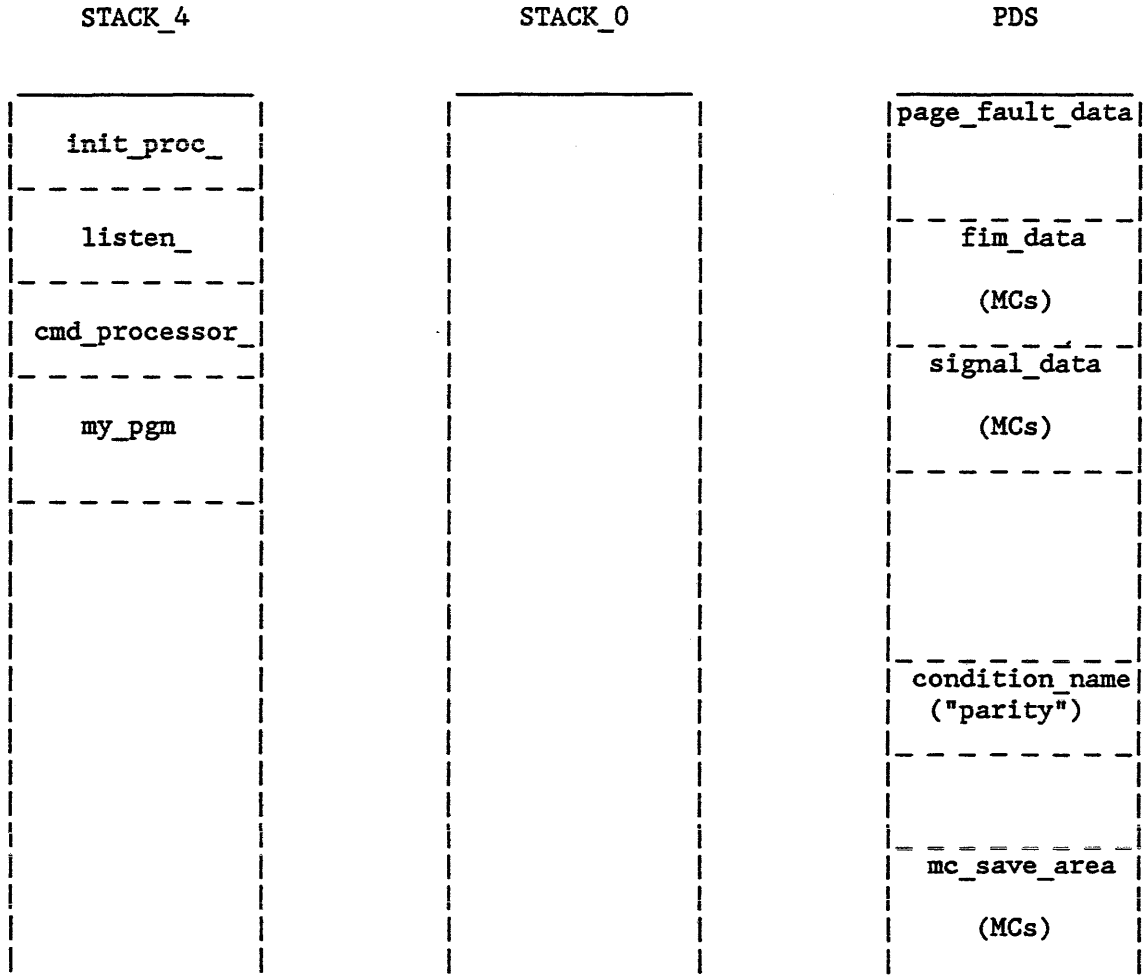
3. RESTART_FAULT



Restart_fault gets the machine conditions from the return_to_ring_0_ FIM frame, and finds the corresponding conditions in pds$mc_save_area. It compares the two, and if no illegal changes have been made to the copy in the FIM Frame, it restarts those conditions.

FAULT RESTART EXAMPLE

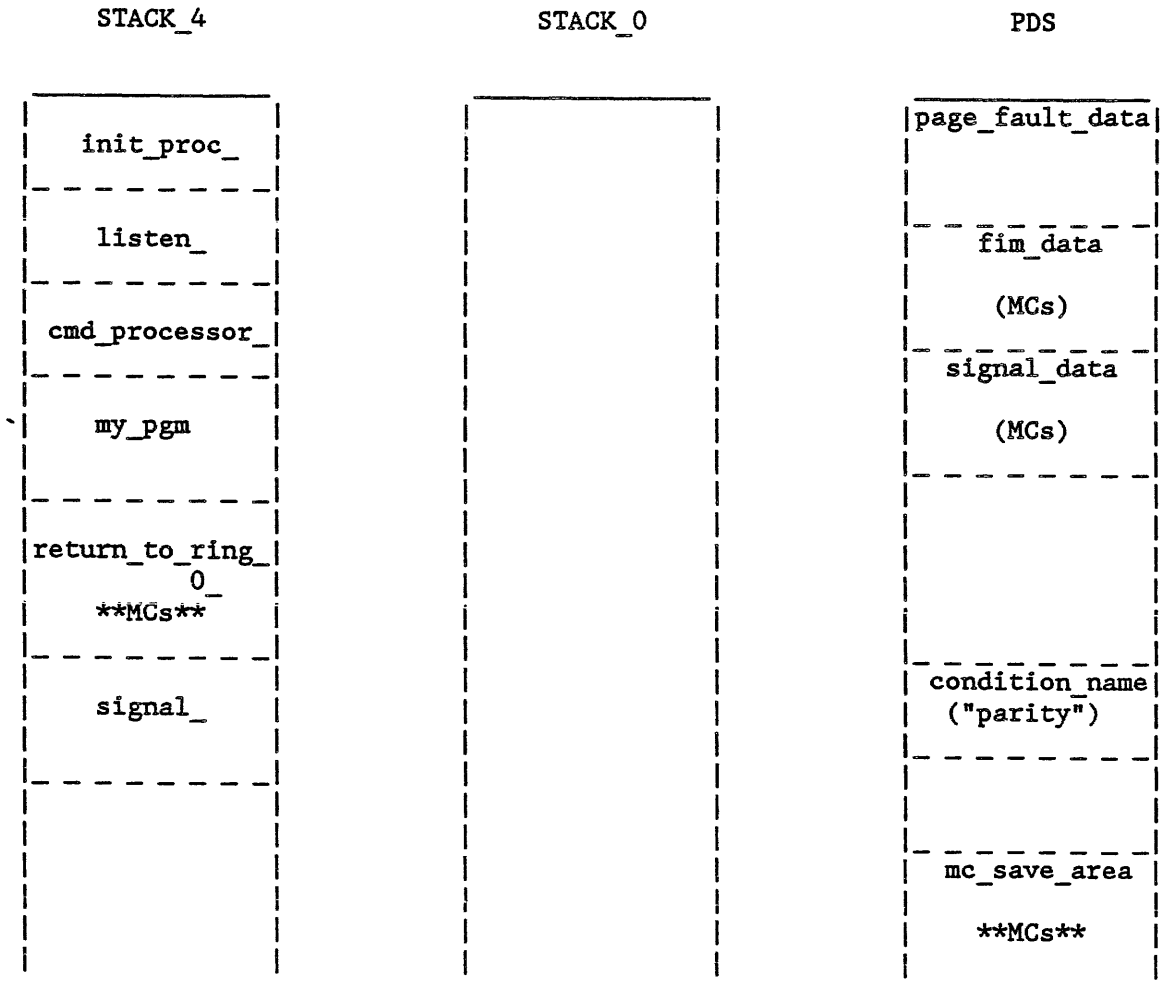
4. FAULT IS RESTARTED



After fault is restarted, everything is back in its original state.

FAULT ABANDONMENT EXAMPLE

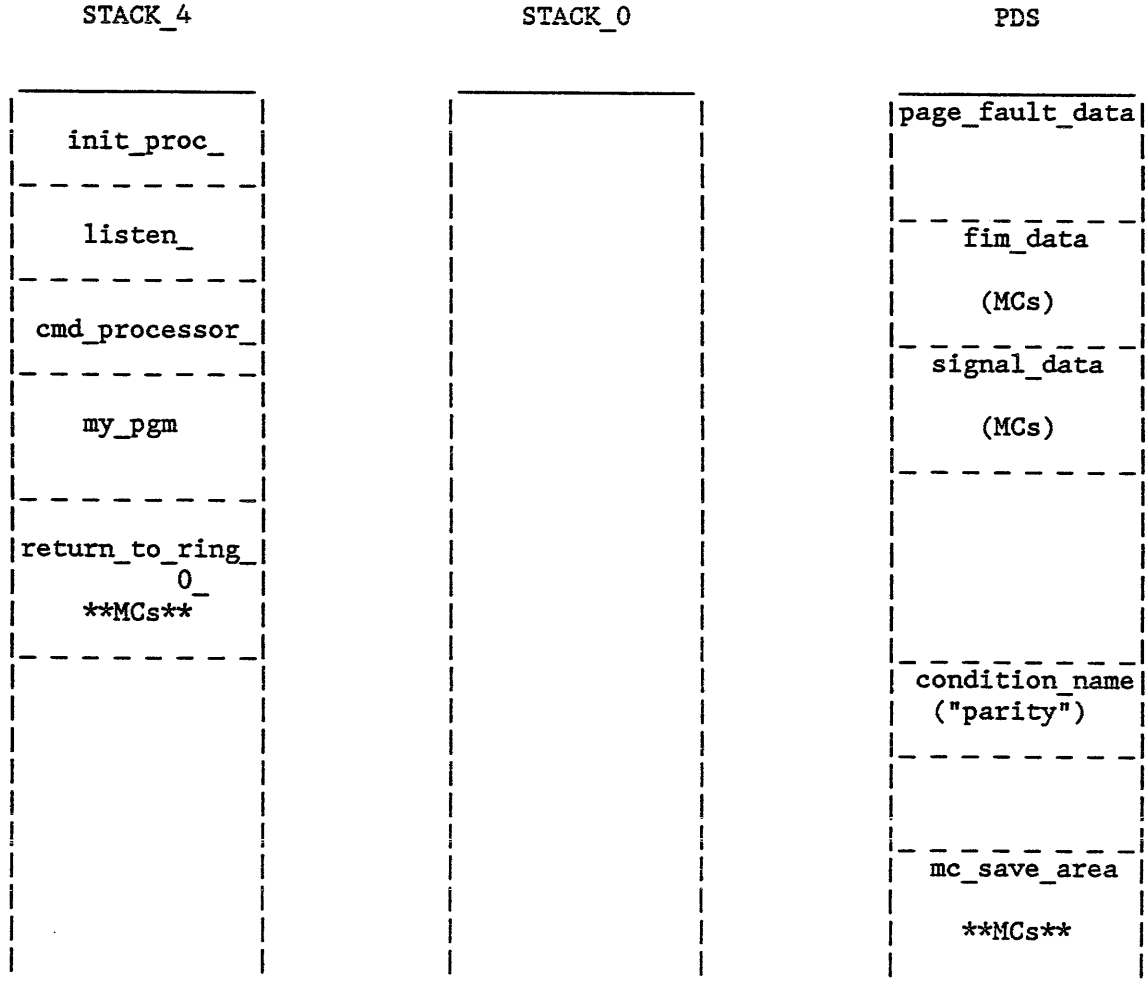
1. HANDLER IS UNWOUND



A non-local goto results in the handler's stack frame being unwound.

FAULT ABANDONMENT EXAMPLE

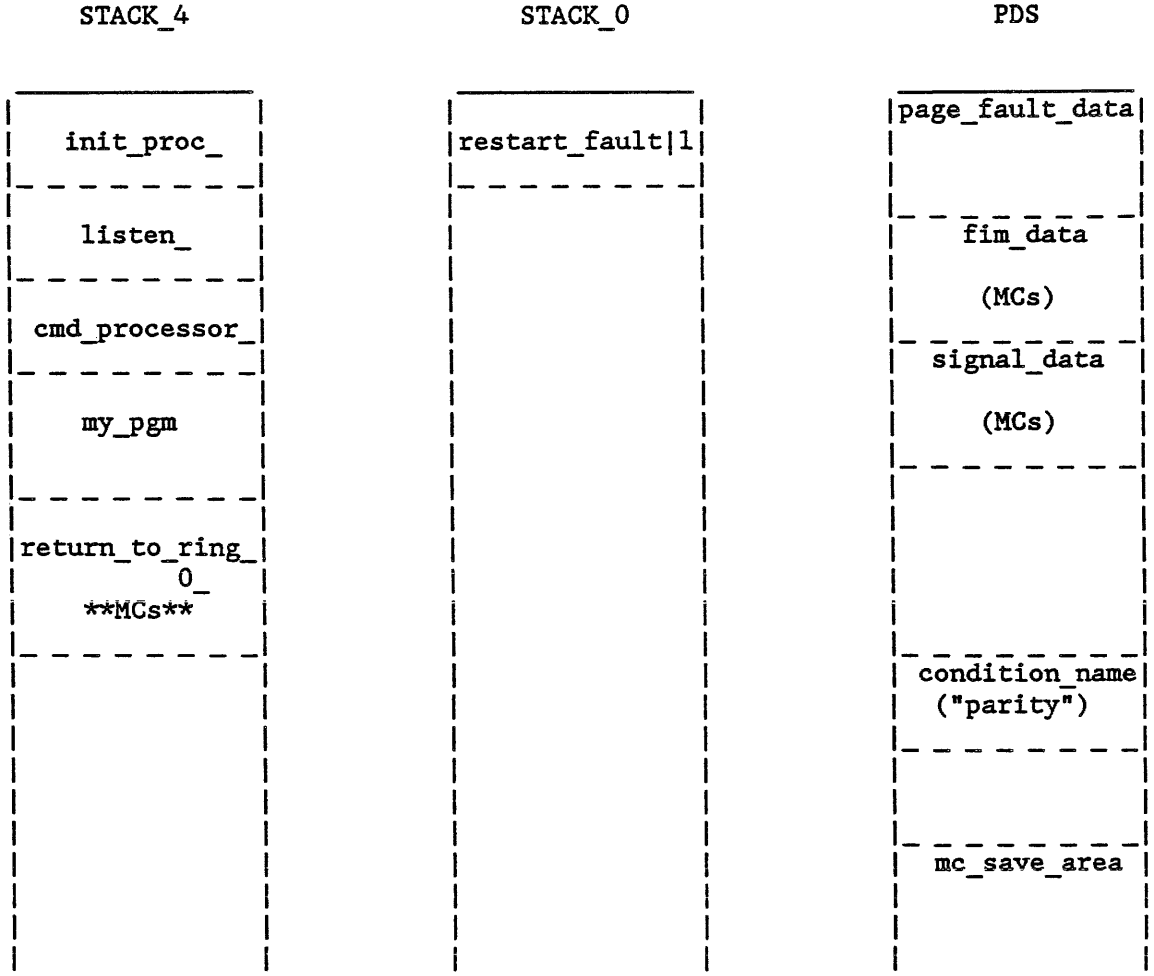
2. SIGNAL_ IS UNWOUND



Next, signal_'s stack frame is unwound.

FAULT ABANDONMENT EXAMPLE

3. FIM FRAME'S CLEANUP HANDLER IS INVOKED



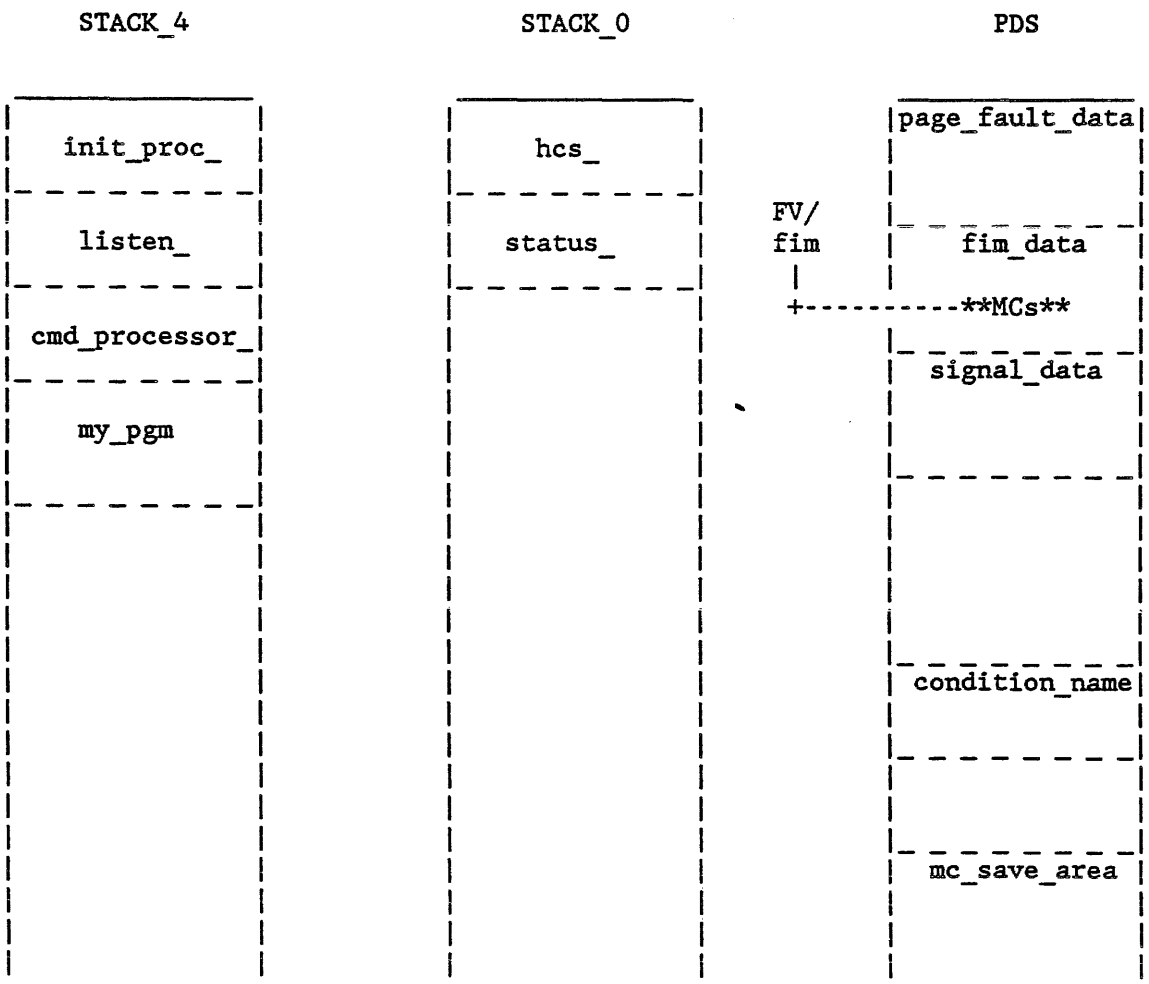
Restart_fault|1 is the cleanup handler that signaller created for the return_to_ring_0_ stack frame. When that stack frame is unwound, restart_fault is invoked in ring 0. It finds the machine conditions in the FIM frame, and finds the corresponding conditions in pds$mc_save_area. It removes them from mc_save_area since they will never be restarted.

RING 0 FAULT EXAMPLE

1. PROGRAM MY_PGM CALLS INTO RING 0

RING 0 FAULT EXAMPLE

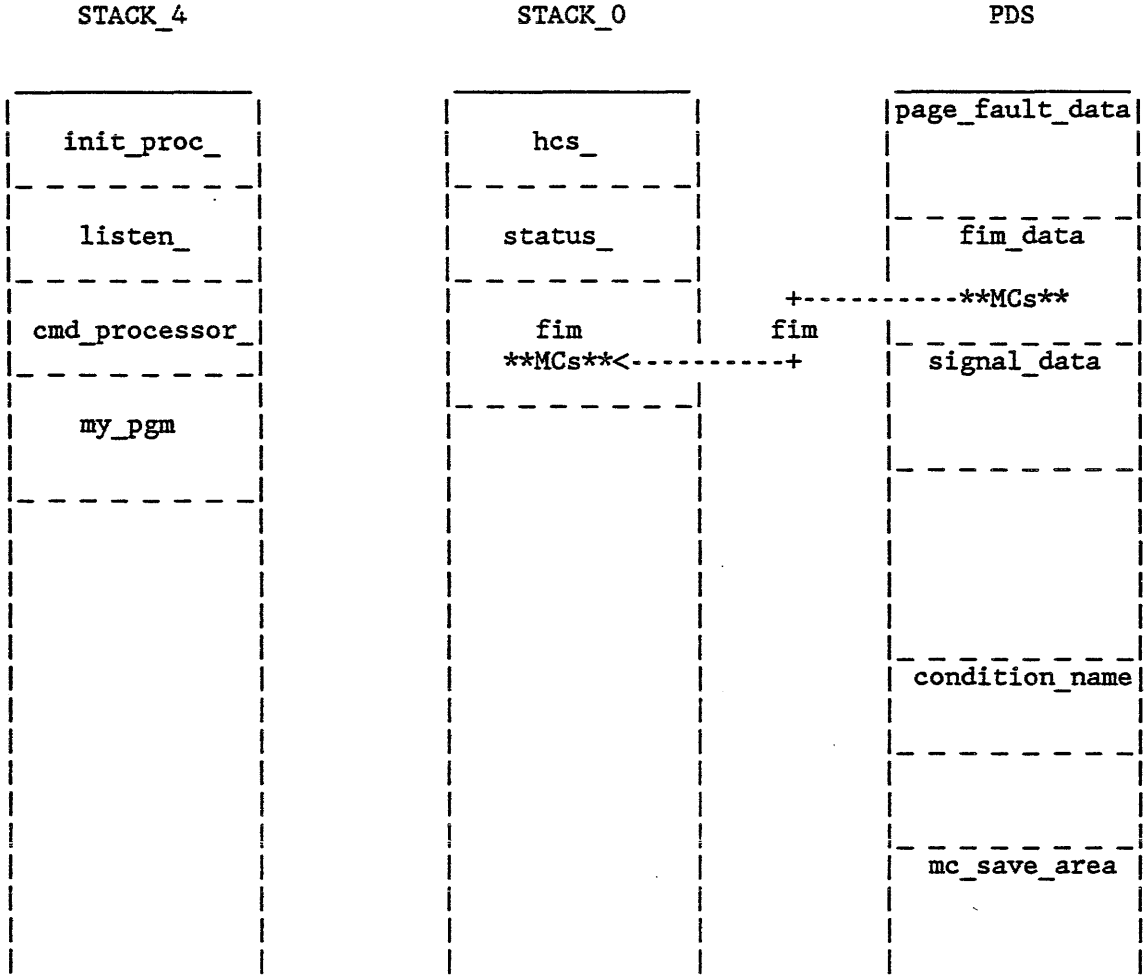
2. PARITY FAULT OCCURS



A Parity Fault occurs while executing an instruction of status_. Transfer via Fault Vecotr to fim, storing Machine Conditions in pds$fim_data.

RING 0 FAULT EXAMPLE

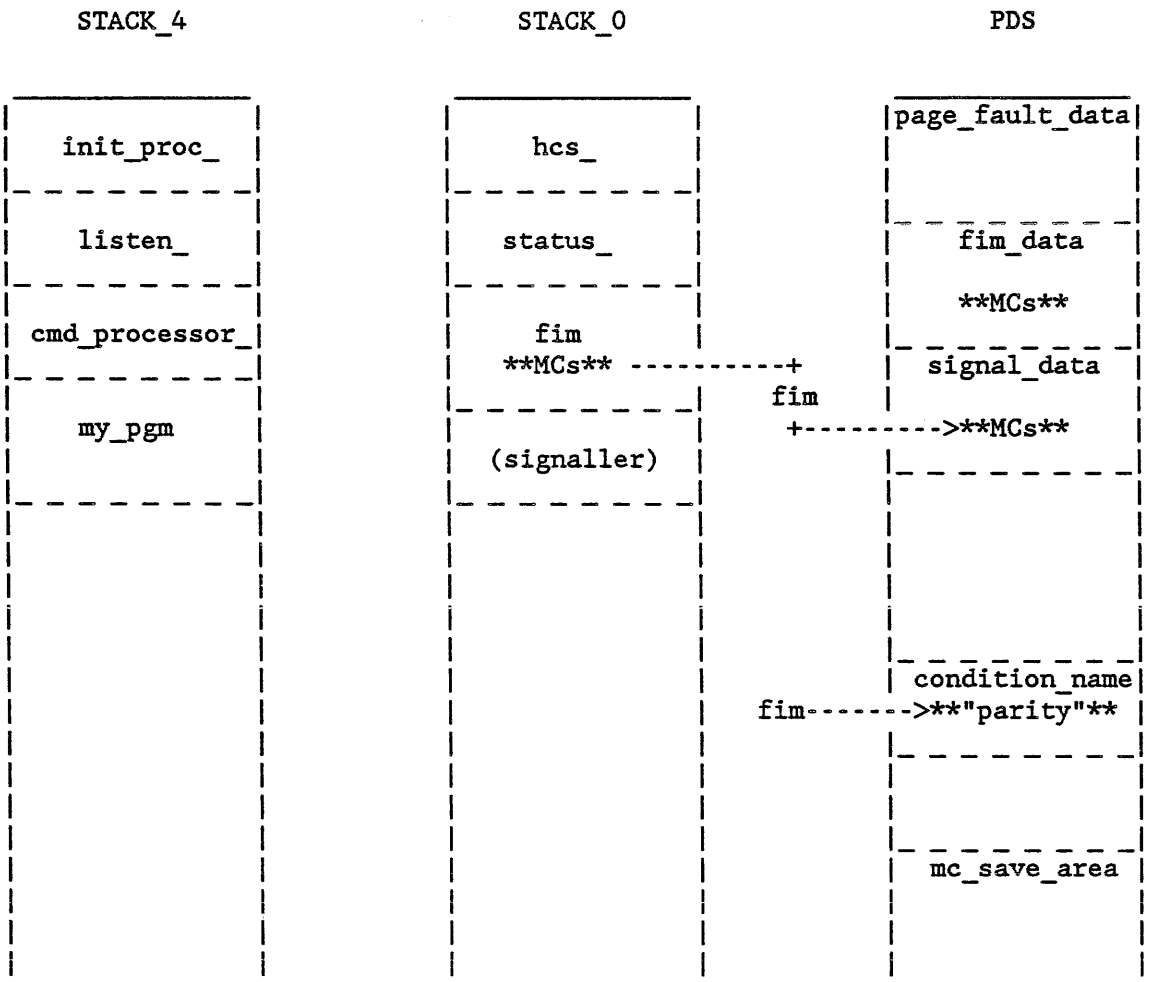
3. FIM BEGINS FAULT HANDLING



Fim pushes a stack frame and copies the Machine Conditions into it, turning on the signal bit in the stack frame to indicate that it is a "FIM Frame".

RING 0 FAULT EXAMPLE

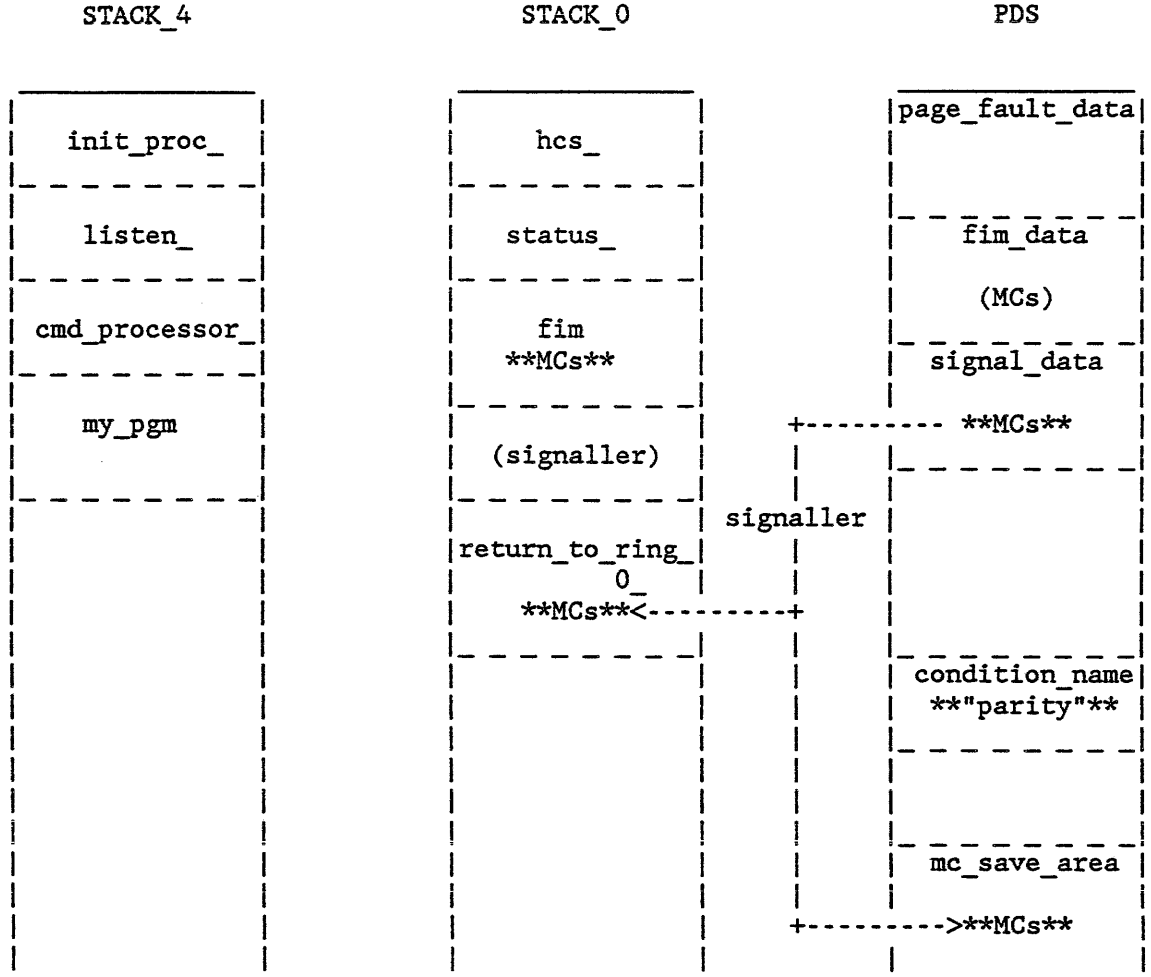
4. FIM DECIDES TO SIGNAL CONDITION



Fim decides to signal a condition as a result of the fault. To do so it copies the Machine Conditions to pds$signal data, puts the condition name in pds$condition_name, and calls signaller. (Signaller is shown on the Ring 0 stack to indicate that it is active, but in reality it does not push a stack frame.)

RING 0 FAULT EXAMPLE

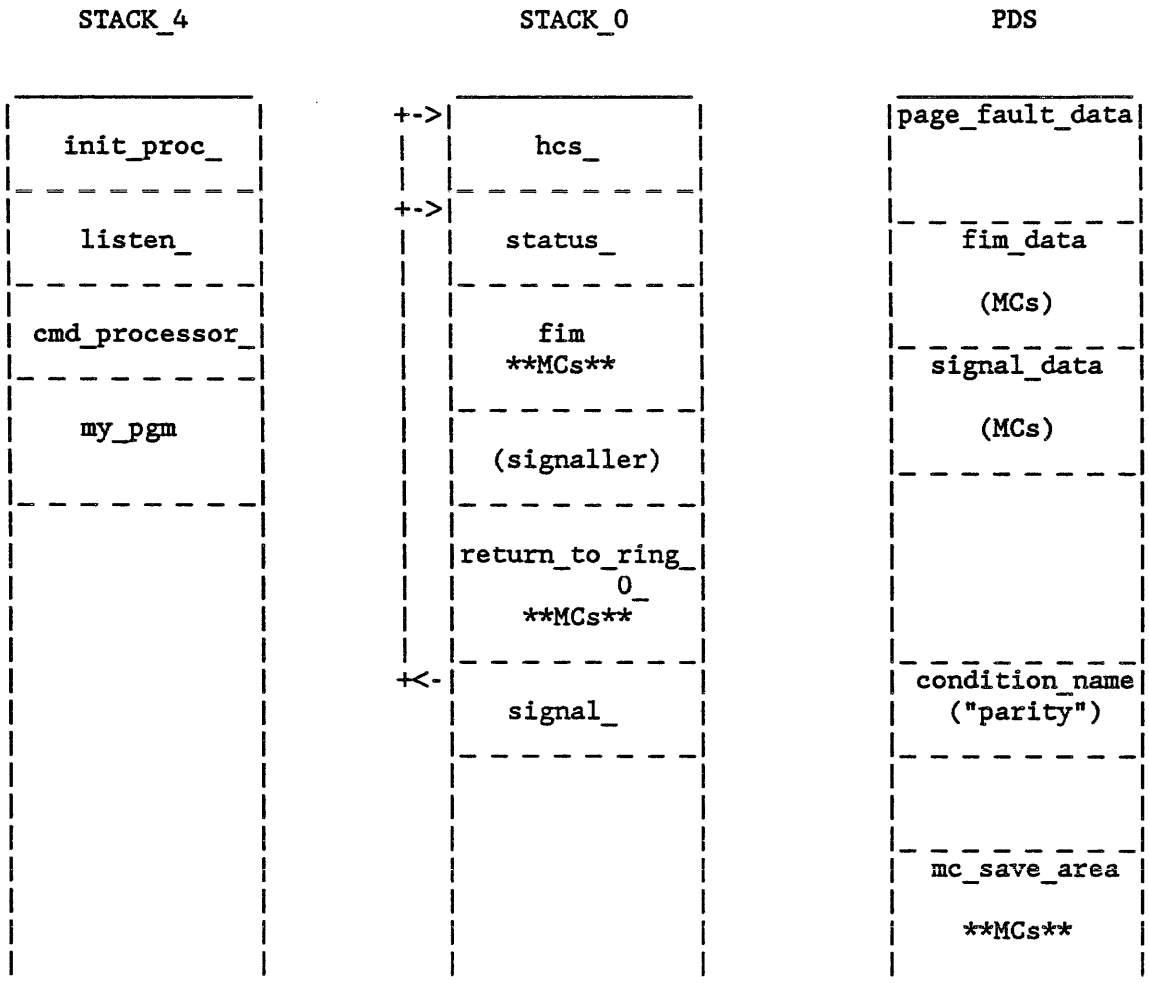
5. SIGNALLER FINDS ORIGINAL STACK



Signaller uses the Machine conditions in pds$signal_data to find the stack the process was using when the fault occurred. It adds a FIM frame to that stack and copies the Machine Conditions there. The Machine Conditions are also copied into a slot in pds$mc_save_area, to be used later if the fault is restarted.

RING 0 FAULT EXAMPLE

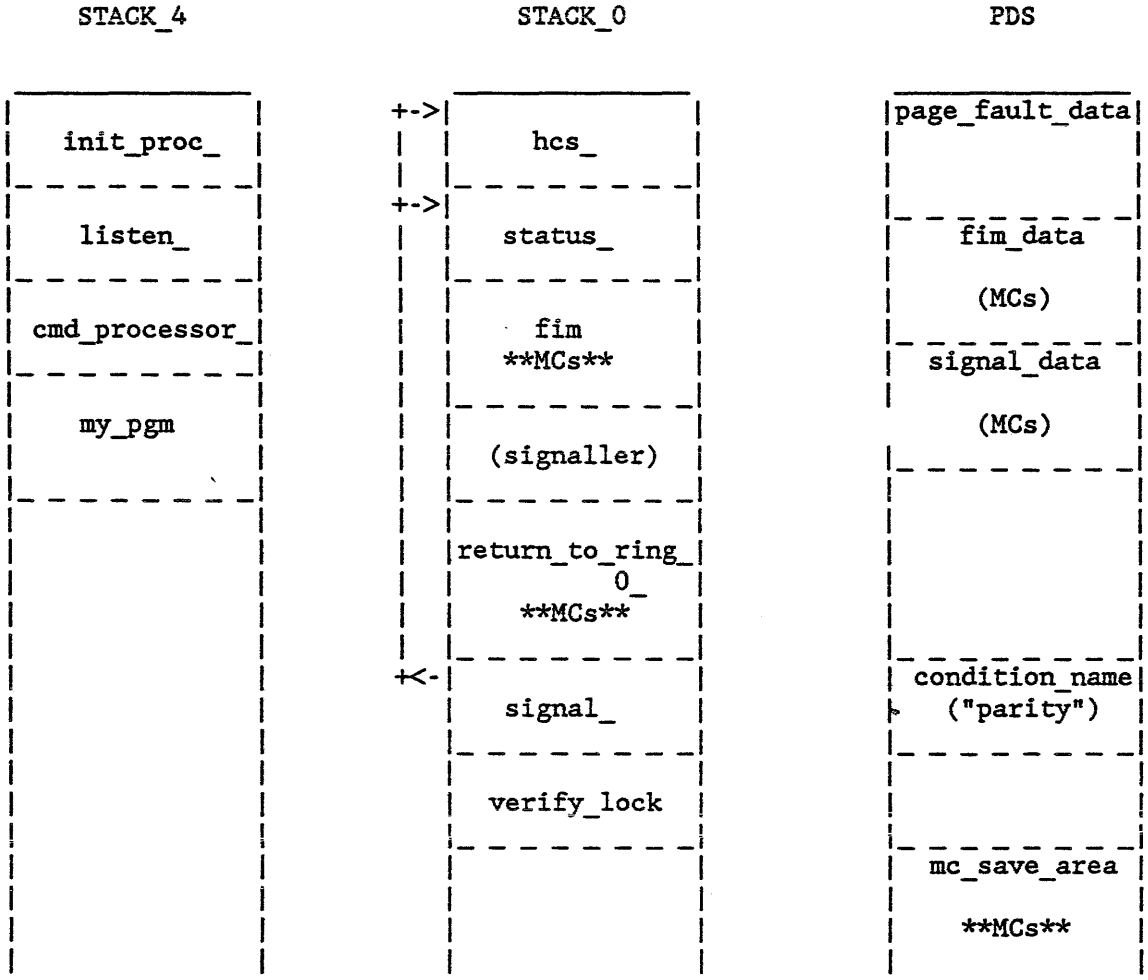
6. SIGNAL_ LOOKS FOR A CONDITION HANDLER



Signaller transfers control to signal_ in the original ring. Signal_ pushes a stack frame and then searches back through the stack frames looking for a handler.

RING 0 FAULT EXAMPLE

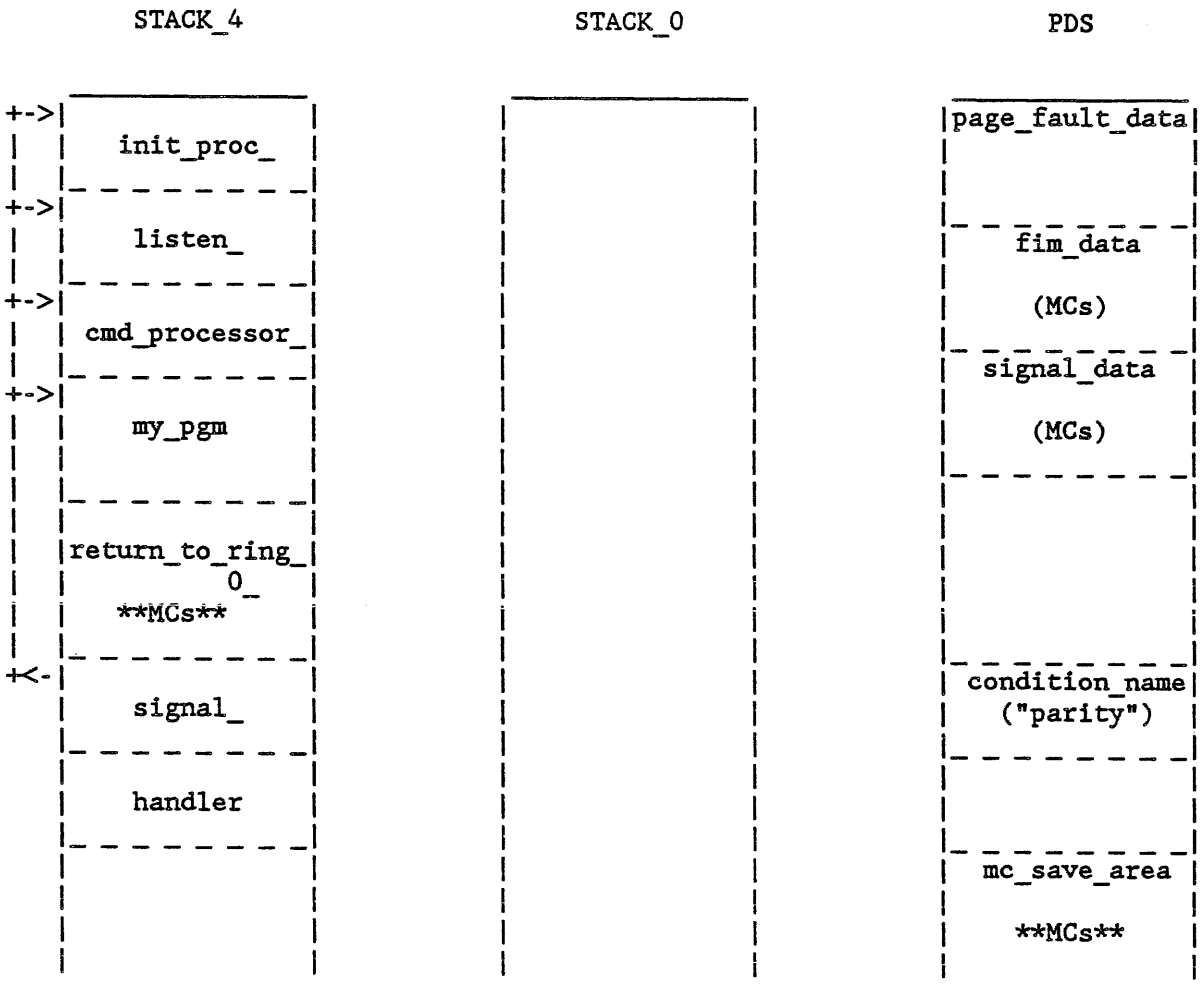
7. SIGNAL_ DECIDES TO LEAVE RING 0



Signal_ does not find a handler for the condition on the ring 0 stack. To continue the search for a handler it must follow stack_frame.prev_sp in the stack frame for hcs_. This transition from an inner ring to an outer ring in signalling a condition is called a "crawlout". When crawling out of a ring, that ring's stack is abandoned, and the condition (and therefore the fault in this example) will not be restartable. Crawling out from ring 0 is a special case. The programs that were active in ring 0 at the time of the fault may have locked supervisor databases or may be in the middle of modifying supervisor data. To be sure that nothing in ring 0 is left in an inconsistent state, verify_lock is called before crawling out.

RING 0 FAULT EXAMPLE

8. CRAWLOUT



The ring 0 stack is abandoned, the condition is essentially re-signalled in ring 4, the handler is found and executed.

Fault Type: Unassigned

26- Unassigned	scu 564,* -> prds$sys_trouble_data (71|240)
30	tra 464,* -> wired_fim$unexp_fault (34|2310)

Fault Type: Execute

15 Execute

scu 536,* -> prds$sys_trouble_data (71|240)
tra 436,* -> wired_fim$xec_fault (34|2274)

Group 2 Fault.

Definition:

1. The EXECUTE pushbutton on the processor maintenance panel has been pressed.

2. An external gate signal has been subsituted for the EXECUTE pushbutton.

(The selection between the above conditions is made by settings of various switches on the processor maintenance panel.)

Use in Multics:

Used to force a system crash.

Fault Type: Execute

```
Machine Conditions For System Trouble Data At prds|240
Time Stored - 02/10/83  1234.0 hfh Thu (111512461676151734)

Pointer Registers

PR0 (ap) - 240|610      |610
PR1 (ab) - 240|460      |460
PR2 (bp) - 113|4500     tc_data|4500
PR3 (bb) - 113|0        tc_data|0
PR4 (lp) - 15|2214      ws_linkage|2214
PR5 (lb) - 41|127       bound_tc_priv$pxss|127
PR6 (sp) - 76|1160      prds|1160
PR7 (sb) - 76|0         prds|0

Processor Registers

X0 - 76 X1 - 0 X2 - 0 X3 - 0 X4 - 0 X5 - 1 X6 - 146 X7 - 1
A Register - 777777777777 Q Register - 460172500752 E Register - 0
Timer Register - 117676224 Ring Alarm Register - 1
SCU masks       - 000000000014 000000000007

SCU Data at prds|270

   270    000041550022 000000000037 000032000000 000000000000
          001257100200 010150000000 001306235000 001306235000

Execute Fault (37)
At: 41|1257  bound_tc_priv$pxss|1257
On: cpu a (#0)
Indicators:  cary, ^bar
APU Status:  priv, sdwamm, sd-on, pt-on, fanp
Instructions:
137576    001306 2350 00      lda       1306
137577    001306 2350 00      lda       1306
```

Fault Type: Timer Runout

4	Timer Runout	scu 510,* -> prds$fim_data (71\|160) tra 410,* -> wired_fim$timer_runout (34|2324)

Group 7 Fault

Definition:

The timer register has decremented to or through the value zero. If the processor is in privileged mode or absolute mode, recognition of this fault is delayed until a return to normal mode or BAR mode. Counting in the timer register continues.

Use in Multics:

Timer runouts are used by traffic control to interrupt processes as the end of their eligibility quantum, and to implement pre-empt sampling. A process running in ring zero does not give up eligibility. However, it remembers that the TRO occured by setting the ring alarm register. When the process leaves ring zero a ring alarm fault will occur and the process will give up its eligibility at that time.

Fault Type: Timer Runout

```
Machine Conditions For Page Fault Data At pds|0
Time Stored - 03/11/83  1704.3 hfh Fri (111557226670514660)

Pointer Registers

PR0 (ap) - 270|7120    >sl1>bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 244|16200   >pdd> BjLBfXcBBBBBBB>stack_4|16200
PR2 (bp) - 22|4000     backup_abs_seg|4000
PR3 (bb) - 244|15234   >pdd> BjLBfXcBBBBBBB>stack_4|15234
PR4 (lp) - 257|24102   >pdd> BjLBfXcBBBBBBB> BBBJMjxJDHWnQq.area.linker|24102
PR5 (lb) - 325|0       >pdd> BjLBfXcBBBBBBB> BBBJMjxJDgXCjX.temp|0
PR6 (sp) - 244|16060   >pdd> BjLBfXcBBBBBBB>stack_4|16060
PR7 (sb) - 244|15360   >pdd> BjLBfXcBBBBBBB>stack_4|15360

Processor Registers

X0 - 0 X1 - 100 X2 - 777773 X3 - 352 X4 - 0 X5 - 0 X6 - 17 X7 - 120
A Register - 000000000000 Q Register - 000000010000 E Register - 0
Timer Register - 777777751 Ring Alarm Register - 0
SCU masks      - 031460000014 631460000007

SCU Data at pds|30

    30    400315170041 000000000011 400315000000 000000242000
          017775000240 001231000500 076615165440 000000165540

Timer Runout Fault (11)
At: 315|17775  >t>bound_volume_dumper_$dmpr_output_|5265
On: cpu a (#0)
Indicators:  ^bar, mif
APU Status:  sdwamm, sd-on, ptwamm, pt-on, fap
CU Status:   rfi, fif
Instructions:
636336   076 615 165 440      tctr      (rl),enablefault,fill(076)
636337   000 000 165 540      tctr      (pr,rl),fill(000)

EIS Pointers and Lengths

    50    000400001040 000400001040 005571000070 000000006740
          005735000030 000077777040 000256000030 000077777734
```

Fault Type: Timer Runout

```
Machine Conditions For Page Fault Data At pds|0
Time Stored - 02/09/83  2111.2 hfh Wed (111511625250263542)

Pointer Registers

PR0 (ap) - 67|1136     inzr_stk0|1136
PR1 (ab) - 104|244     scs|244
PR2 (bp) - 113|3300    tc_data|3300
PR3 (bb) - 113|3100    tc_data|3100
PR4 (lp) - 15|3100     ws_linkage|3100
PR5 (lb) - 76|1153     prds|1153
PR6 (sp) - 77777|1     NULL POINTER
PR7 (sb) - 67|0        inzr_stk0|0

Processor Registers

X0 - 1 X1 - 2 X2 - 200 X3 - 0 X4 - 2 X5 - 60 X6 - 3 X7 - 0
A Register - 000000777777 Q Register - 777777000000 E Register - 0
Timer Register - 777777760 Ring Alarm Register - 0
SCU masks      - 031460000014 631460000007

SCU Data at pds|30

    30    000065550021 000000000011 400260000200 000000000000
          000263100200 001266000500 000262710200 400064214320

Timer Runout Fault (11)
At: 65|263  init_processor|263
On: cpu c (#2)
Indicators:  cary, ^bar
APU Status:  priv, sdwamm, sd-on, pt-on, fanp
CU Status:   rfi, fif
Instructions:
452236    000262 7102 00      tra       262 interrupt inhibit
452237   4 00064 2143 20      sznc      pr4|64,* interrupt inhibit
```

Fault Type: Connect

8	Connect	scu 520,* -> prds$fim_data (71\|160) tra 420,* -> prds$fast_connect_code (71|1054)

Group 7 Fault

Definition:

A connect signal ($CON strobe) has been received from a system controller. This event is to be distinguished from a Connect Input/Output Channel (cioc) instruction encountered in the program sequence.

Use in Multics:

Used for all interprocessor signalling. Occurs when one CPU executes a cioc instruction to "send a connect" to another CPU, or to itself.

There are four types of interprocessor communication in Multics:

1. Tell another CPU to clear its cache memory (Level 68 only) or its associative memory.

2. To pre-empt a process running on another CPU if pre-empty sampling is not in use.

3. To tell a CPU that the system is crashing.

4. To tell a CPU that it is being deconfigured, due to delcpu command or a shutdown.

Zones in the segment scs are used to indicate what type of connect is being sent to a CPU. prds$fast_connect_code handles the cache/AM clearing case, others are sent on to wired_fim.

Fault Type: Connect

```
Machine Conditions For System Trouble Data At prds|240
Time Stored - 02/09/83  2111.4 hfh Wed (1115116253266454 25)

Pointer Registers

PR0 (ap) - 270|7120    >sl1>bound_system_control_$sc_parse_|4126
PR1 (ab) - 244|11016   >pdd> zzzzzzzbBBBBBB>stack_4|11016
PR2 (bp) - 244|15270   >pdd> zzzzzzzbBBBBBB>stack_4|15270
PR3 (bb) - 360|1425    >scl>as_meter_table|1425
PR4 (lp) - 257|31154   >sl1>bound_fsim_
PR5 (lb) - 257|33250   >sl1>bound_fsim_
PR6 (sp) - 244|10620   >pdd> zzzzzzzbBBBBBB>stack_4|10620
PR7 (sb) - 360|2016    >scl>as_meter_table|2016

Processor Registers

X0 - 1416 X1 - 36466 X2 - 174 X3 - 4 X4 - 136 X5 - 1 X6 - 777777 X7 - 4
A Register - 000000000000 Q Register - 000000000012 E Register - 107
Timer Register - 000044726 Ring Alarm Register - 0
SCU masks      - 031460000014 631460000007
Fault Register - 000400000000  ($CON A)

SCU Data at prds|270

   270    400270050011 000000000021 400253000207 000000000000
          036040400200 007524000500 000001316007 000001316007

Connect Fault (21)
At: 270|36040  >sl1>bound_system_control_
On: cpu c (#2)
Indicators:  zero, ^bar
APU Status:  sd-on, pt-on, fabs
CU Status:   rfi, fif
Instructions:
 25676    000001 3160 07      canq      1,dl
 25677    000001 3160 07      canq      1,dl
```

Fault Type: Connect

```
Machine Conditions For Signal Data At pds|140
Time Stored - 03/17/83  1531.7 hfh Thu (111566603700510407)

Pointer Registers

PR0 (ap) - 270|7120    >sl1>bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 244|52400   >pdd> BWbBhXGBBBBBBBB>stack_4|52400
PR2 (bp) - 244|52506   >pdd> BWbBhXGBBBBBBBB>stack_4|52506
PR3 (bb) - 244|43560   >pdd> BWbBhXGBBBBBBBB>stack_4|43560
PR4 (lp) - 456|16303   >udd>Croap>lib>executable>bound_pascal_runtime_$pascal_io_|163
PR5 (lb) - 244|43560   >pdd> BWbBhXGBBBBBBBB>stack_4|43560
PR6 (sp) - 244|53040   >pdd> BWbBhXGBBBBBBBB>stack_4|53040
PR7 (sb) - 257|110242  >pdd> BWbBhXGBBBBBBBB> BBBJMkWLfGMqHm.area.linker|110242

Processor Registers

X0 - 4147 X1 - 54640 X2 - 150732 X3 - 151116 X4 - 151116 X5 - 22 X6 - 71 X7 - 2220
A Register - 777777777775 Q Register - 000000000000 E Register - 0
Timer Register - 000143044 Ring Alarm Register - 0
SCU masks      - 000240000043 000340000000
Fault Register - 000400000000  ($CON A)

SCU Data at pds|170

   170    400456174043 000000000021 400456000100 000037000000
          016305100200 016305000500 000000066500 000000066500

Connect Fault (21)
At: 456|16305  >udd>Croap>lib>executable>bound_pascal_runtime_$pascal_io_|16305
On: cpu b (#1)
Indicators:  cary, ^bar
APU Status:  sdwamm, sd-on, ptwamm, pt-on, pi-ap, fap
CU Status:   rfi, fif
Instructions:
 53676   000 000 066 500      cmpb      (pr),(),fill(0)
 53677   000 000 066 500      cmpb      (pr),(),fill(0)

EIS Pointers and Lengths

   210    000400000000 000400000000 043617410030 010477777760
          151040000030 000000000000 054156000000 000000000077
```

Fault Type: Connect

```
Machine Conditions For Signal Data At pds|140
Time Stored - 03/17/83  1528.6 hfh Thu (111566602375501317)

Pointer Registers

PR0 (ap) - 270|7120    >sl1>bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 104|244     scs|244
PR2 (bp) - 244|4000    >pdd> BlLBhbjbBBBBBB>stack_4|4000
PR3 (bb) - 113|0       tc_data|0
PR4 (lp) - 14|12276    as_linkage|12276
PR5 (lb) - 75|3714     pds|3714
PR6 (sp) - 244|4000    >pdd> BlLBhbjbBBBBBB>stack_4|4000
PR7 (sb) - 244|0       >pdd> BlLBhbjbBBBBBB>stack_4|0

Processor Registers

X0 - 577777 X1 - 4 X2 - 0 X3 - 400000 X4 - 0 X5 - 127 X6 - 120 X7 - 1
A Register - 040000000000 Q Register - 000000000004 E Register - 0
Timer Register - 000141001 Ring Alarm Register - 0
SCU masks      - 000240000043 000340000000
Fault Register - 000400000000  ($CON A)

SCU Data at pds|170

   170    400253170041 000000000021 400253000200 000007000000
          000250500200 000246000500 600101116100 600101116100

Connect Fault (21)
At: 253|250  >sl1>bound_command_loop_$ipc_fast_|206
On: cpu c (#2)
Indicators:  zero, cary, ^bar
APU Status:  sdwamm, sd-on, ptwamm, pt-on, fap
CU Status:   rfi, fif
Instructions:
750076   6 00101 1161 00      cmpq      pr6|101
750077   6 00101 1161 00      cmpq      pr6|101
```

Fault Type: Connect

```
Machine Conditions For System Trouble Data At prds|240
Time Stored - 03/17/83  1533.7 hfh Thu (111566604603067407)

Pointer Registers

PR0 (ap) - 76|3176      prds|3176
PR1 (ab) - 104|244      scs|244
PR2 (bp) - 36|454       bound_priv_1$privileged_mode_ut|454
PR3 (bb) - 17|0         sst_seg|0
PR4 (lp) - 15|1424      ws_linkage|1424
PR5 (lb) - 15|1424      ws_linkage|1424
PR6 (sp) - 76|3120      prds|3120
PR7 (sb) - 76|0         prds|0

Processor Registers

X0 - 6111 X1 - 3176 X2 - 0 X3 - 0 X4 - 0 X5 - 0 X6 - 213 X7 - 4
A Register - 700000000000 Q Register - 000000000000 E Register - 0
Timer Register - 775232216 Ring Alarm Register - 1
SCU masks       - 000000000014 000000000007
Fault Register - 000400000000  ($CON A)

SCU Data at prds|270

   270    000036750021 000000000021 400300000000 000000440000
          000247200200 001266000700 000061015200 000000235007

Connect Fault (21)
At: 36|247  bound_priv_1$privileged_mode_ut|247
On: cpu a (#0)
Indicators:  neg, ^bar
APU Status:  priv, xsf, sdwamm, sd-on, pt-on, fanp
CU Status:   rfi, its, fif
Instructions:
422076    000061 0152 00      cioc      61 interrupt inhibit
422077    000000 2350 07      lda       0,dl
```

Fault Type: Connect

```
Machine Conditions For Fim Data At prds|160
Time Stored - 02/09/83  2212.8 hfh Wed (111511660674665306)

Pointer Registers

PR0 (ap) - 76|1420      prds|1420
PR1 (ab) - 76|336       prds|336
PR2 (bp) - 75|140       pds|140
PR3 (bb) - 113|0        tc_data|0
PR4 (lp) - 15|312       ws_linkage|312
PR5 (lb) - 15|2214      ws_linkage|2214
PR6 (sp) - 76|1160      prds|1160
PR7 (sb) - 76|0         prds|0

Processor Registers

X0 - 203 X1 - 1 X2 - 13740 X3 - 2270 X4 - 0 X5 - 2270 X6 - 344 X7 - 777776
A Register - 000000000000 Q Register - 000000000000 E Register - 4
Timer Register - 777737247 Ring Alarm Register - 0
SCU masks       - 000000000014 000000000007

SCU Data at prds|210

   210    000032750021 000000000021 000041000100 000000440000
          002505102200 002436000700 000063015200 777777710204

Connect Fault (21)
At: 32|2505  bound_interceptors$fim_util|53
On: cpu b (#1)
Indicators:  cary, tro, ^bar
APU Status:  priv, xsf, sdwamm, sd-on, pt-on, fanp
CU Status:   rfi, its, fif
Instructions:
241016    000063 0152 00      cioc      63 interrupt inhibit
241017    777777 7102 04      tra       -1,ic 241016 interrupt inhibit
```

Fault Type: Access Violation

20 Access Violation scu 550,* -> pds$fim_data (70|60)
tra 450,* -> fim$access_violation_entry (34|0)

Group 6 Fault

Definition:

The appending unit has detected one of the several access violations below. Word 1 of the Control Unit Data contains status bits for the condition.

1. Not in read bracket (ACV3=ORB)
2. Not in write bracket (ACV5=OWB)
3. Not in execute bracket (ACV1=OEB)
4. No read permission (ACV4=R-OFF)
5. No write permission (ACV6=W-OFF)
6. No execute permission (ACV2=E-OFF)
7. Invalid ring crossing (ACV12=CRT)
8. Call limiter fault (ACV7=(NO GA)
9. Outward call (ACV9=OCALL)
10. Bad outward call (ACV10=BOC)
11. Inward return (ACV11=INRET)
12. Ring alarm (ACV13=RALR)
13. Associative memory error
14. Out of segment bounds (ACV15=OOSB)
15. Illegal ring order (ACV0=IRO)
16. Out of call brackets (ACV8=OCB)

Use in Multics:

The most-used type of access fault is the ring alarm fault. The supervisor uses the Ring Alarm Register, described by AL39:

> If the RALR contains a vlue other than zero and the effective ruing number is greater than or equal to the contents of the RALR and the instruction for which an absolute main memory address is being prepared is a transfer instruction, an access violation, ring alarm, fault occurs. Operating system software may use this register to detect crossings from inner rings to outer rings.

A ring alarm fault is used for two purposes.

1. Used by traffic control to defer loss of eligibility when an end-of-eligibility timer runout fault or a pre-empt connect fault occurs in ring 0. When the timer runout or connect is handled, the RALR is set to 1.

2. Used to ensure that a process's validation level is set to a value at least equal to the new ring of execution when leaving an inner ring.

Fault Type: Access Violation

An out of segment bounds fault may indicate a boundsfault, in which a segment has exceeded the maximum size for its AST pool and must be promoted to a bigger pool.

All other types of access violation can be provoked by users.

Fault Type: Access Violation

```
Machine Conditions For Fim Data At pds|60
Time Stored - 02/09/83  2212.8 hfh Wed (111511660705216454)

Pointer Registers

PR0 (ap) - 76|240       prds|240
PR1 (ab) - 104|244      scs|244
PR2 (bp) - 76|240       prds|240
PR3 (bb) - 1|0          fault_vector|0
PR4 (lp) - 15|312       ws_linkage|312
PR5 (lb) - 76|320       prds|320
PR6 (sp) - 76|1160      prds|1160
PR7 (sb) - 76|0         prds|0

Processor Registers

X0 - 3036 X1 - 2561 X2 - 4000 X3 - 0 X4 - 0 X5 - 0 X6 - 2 X7 - 100
A Register - 000000000000 Q Register - 000000000000 E Register - 0
Timer Register - 773613312 Ring Alarm Register - 0
Fault Register - 010000000000  (OOB)

SCU Data at pds|110

   110    000032550022 004000000051 000032000017 000000000000
          010153410200 000120000007 000020057217 000020057217

Access Violation Fault (51), Write Bit Off
By: 32|10153  bound_interceptors
Referencing: 32|120  bound_interceptors$fim|120
On: cpu a (#0)
Indicators:  zero, eufl, ^bar
APU Status:  priv, sdwamm, sd-on, pt-on, fanp
CU Status:
CT Hold: dl
Instructions:
634216    000020 0572 17      sscr      20,7 interrupt inhibit
634217    000020 0572 17      sscr      20,7 interrupt inhibit
```

Fault Type: Access Violation

```
Machine Conditions For Signal Data At pds|140
Time Stored - 02/10/83  1230.3 hfh Thu (1115124601541301O2)

Pointer Registers

PR0 (ap) - 322|7120    >sl1>bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 244|0       >pdd> zzzzzzzbBBBBBB>stack_4|0
PR2 (bp) - 457|26353   >t>bound_oprcons_$mrd_util_|1
PR3 (bb) - 244|774602  >pdd> zzzzzzzbBBBBBB>stack_4|774602
PR4 (lp) - 0|0         dseg|0
PR5 (lb) - 244|774532  >pdd> zzzzzzzbBBBBBB>stack_4|774532
PR6 (sp) - 244|774620  >pdd> zzzzzzzbBBBBBB>stack_4|774620
PR7 (sb) - 244|774550  >pdd> zzzzzzzbBBBBBB>stack_4|774550

Processor Registers

X0 - 32017 X1 - 774744 X2 - 174 X3 - 0 X4 - 0 X5 - 0 X6 - 0 X7 - 17
A Register - 000002000000 Q Register - 000000000000 E Register - 0
Timer Register - 000037431 Ring Alarm Register - 0
SCU masks      - 000102400043 000004000000
Fault Register - 010400000000  (OOB, $CON A)

SCU Data at pds|170

   170    400457050401 040000000051 400000000120 000000000000
          032025000200 001350000007 401350352120 401350352120

Access Violation Fault (51), Not In Read Bracket
By: 457|32025  >t>bound_oprcons_$mrd_util_|3453
Referencing: 0|1350  dseg|1350
On: cpu b (#1)
Indicators:  ^bar
APU Status:  sd-on, pt-on, sdwp
CU Status:
CT Hold: dl
Instructions:
 25576   4 01350 3521 20      epp2      pr4|1350,*
 25577   4 01350 3521 20      epp2      pr4|1350,*
```

Fault Type: Access Violation

```
Machine Conditions For Signal Data At pds|140
Time Stored - 03/11/83  1704.2 hfh Fri (111557226651237015)

Pointer Registers

PR0 (ap) - 40|7120     bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 104|244     scs|244
PR2 (bp) - 230|45263   tty_buf|45263
PR3 (bb) - 77777|1741  CANNOT GET PATHNAME|1741
PR4 (lp) - 15|1616     ws_linkage|1616
PR5 (lb) - 230|45262   tty_buf|45262
PR6 (sp) - 240|600     >sl1>stack_0.015|600
PR7 (sb) - 240|200     >sl1>stack_0.015|200

Processor Registers

X0 - 30617 X1 - 2 X2 - 777773 X3 - 500 X4 - 0 X5 - 4 X6 - 7 X7 - 460
A Register - 777777007777 Q Register - 000000000174 E Register - 0
Timer Register - 000122364 Ring Alarm Register - 0
SCU masks      - 000102000043 000004000000
Fault Register - 010000000000  (OOB)

SCU Data at pds|170

   170    000145052001 000004000051 477777000206 000000000000
          003427100200 001741000400 300000100440 040140100540

Access Violation Fault (51), Out of Segment Bounds
By: 145|3427  bound_tty_active$tty_write|2033
Referencing: 77777|1741  CANNOT GET PATHNAME|1741
On: cpu c (#2)
Indicators:  cary, ^bar
APU Status:  sd-on, pt-on, dsptw
CU Status:   rfi
Instructions:
536076   300 000 100 440      mlr       (rl),(),fill(300)
536077   040 140 100 540      mlr       (pr,rl),(pr,rl),fill(040)

EIS Pointers and Lengths

   210    000400000000 000400000000 001741000070 002000000170
          001250000030 000000000000 002001000000 000077777735
```

Fault Type: Access Violation

```
Machine Conditions For Fim Data At pds|60
Time Stored - 02/10/83  1230.3 hfh Thu (111512460154134307)

Pointer Registers

PR0 (ap) - 244|775732  >pdd> zzzzzzzbBBBBBB>stack_4|775732
PR1 (ab) - 244|775240  >pdd> zzzzzzzbBBBBBB>stack_4|775240
PR2 (bp) - 321|14521   >sl1>bound_sss_active_$sct_manager_|15
PR3 (bb) - 244|776040  >pdd> zzzzzzzbBBBBBB>stack_4|776040
PR4 (lp) - 0|0         dseg|0
PR5 (lb) - 244|775140  >pdd> zzzzzzzbBBBBBB>stack_4|775140
PR6 (sp) - 244|775500  >pdd> zzzzzzzbBBBBBB>stack_4|775500
PR7 (sb) - 244|0       >pdd> zzzzzzzbBBBBBB>stack_4|0

Processor Registers

X0 - 17016 X1 - 775732 X2 - 174 X3 - 0 X4 - 0 X5 - 0 X6 - 0 X7 - 160
A Register - 000321000004 Q Register - 014521000000 E Register - 0
Timer Register - 000035274 Ring Alarm Register - 0
Fault Register - 010000000000  (OOB)

SCU Data at pds|110

   110    400322150201 000004000051 400244000100 000000000000
          017277100200 776060000000 300020652100 300020652100

Access Violation Fault (51), Out of Segment Bounds
By: 322|17277  >sl1>bound_sss_wired_$pl1_operators_|11541
Referencing: 244|776060  >pdd> zzzzzzzbBBBBBB>stack_4|776060
On: cpu b (#1)
Indicators:  cary, ^bar
APU Status:  sdwamm, sd-on, pt-on, ptw
Instructions:
 25516   3 00020 6521 00      spri6     pr3|20
 25517   3 00020 6521 00      spri6     pr3|20
```

Fault Type: Access Violation

```
Machine Conditions For Fim Data At pds|60
Time Stored - 02/09/83  2212.7 hfh Wed (111511660666116070)

Pointer Registers

PR0 (ap) - 270|7120    >sl1>bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 244|143701(6) >pdd> BL1BLDPBBBBBBB>stack_4|143701
PR2 (bp) - 244|260000  >pdd> BL1BLDPBBBBBBB>stack_4|260000
PR3 (bb) - 244|120740(3) >pdd> BL1BLDPBBBBBBB>stack_4|120740
PR4 (lp) - 257|30034   >pdd> BL1BLDPBBBBBBB> BBBJMgffLJXLPh.area.linker|30034
PR5 (lb) - 244|71220   >pdd> BL1BLDPBBBBBBB>stack_4|71220
PR6 (sp) - 244|257640  >pdd> BL1BLDPBBBBBBB>stack_4|257640
PR7 (sb) - 244|106140  >pdd> BL1BLDPBBBBBBB>stack_4|106140

Processor Registers

X0 - 17053 X1 - 106662 X2 - 272 X3 - 15 X4 - 1 X5 - 172 X6 - 50 X7 - 447
A Register - 000000000045 Q Register - 777777777777 E Register - 0
Timer Register - 777774627 Ring Alarm Register - 1
Fault Register - 010000000000  (OOB)

SCU Data at pds|110

   110    400272050401 000020000051 400272000200 000000000000
          017204300200 017204010624 017204613200 000116100600

Access Violation Fault (51), Ring Alarm
By: 272|17204  >udd>Attrisem>Jourdan>mesures>ORDA4|17204
Referencing: 272|17204  >udd>Attrisem>Jourdan>mesures>ORDA4|17204
On: cpu c (#2)
Indicators:  neg, cary, ^bar
APU Status:  sd-on, pt-on, sdwp
CU Status:   pon, rfi, its
CT Hold: ic*
Instructions:
243716    017204 6132 00      rcu       17204 interrupt inhibit
243717   000 116 100 600      mlr       (),(pr,x6),fill(000)
```

Fault Type: Access Violation

```
Machine Conditions For Fim Data At pds|60
Time Stored - 02/10/83  1230.3 hfh Thu (1115124601562713 32)

Pointer Registers

PR0 (ap) - 270|7120    >sl1>bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 76|336      prds|336
PR2 (bp) - 244|26020   >pdd> CBBBLLzbBBBBBB>stack_4|26020
PR3 (bb) - 113|0       tc_data|0
PR4 (lp) - 14|12276    as_linkage|12276
PR5 (lb) - 75|3714     pds|3714
PR6 (sp) - 244|26020   >pdd> CBBBLLzbBBBBBB>stack_4|26020
PR7 (sb) - 244|0       >pdd> CBBBLLzbBBBBBB>stack_4|0

Processor Registers

X0 - 577777 X1 - 4 X2 - 0 X3 - 400000 X4 - 0 X5 - 127 X6 - 120 X7 - 1
A Register - 040000000000 Q Register - 000000000000 E Register - 0
Timer Register - 000166755 Ring Alarm Register - 1
Fault Register - 010000000000  (OOB)

SCU Data at pds|110

   110    000040250401 000020000051 400301000100 000000672000
          015436000200 001266010200 001266610000 600076757100

Access Violation Fault (51), Ring Alarm
By: 40|15436  bound_sss_wired_$pl1_operators_|7700
Referencing: 301|1266  >sl1>bound_ipc_$ipc_real_|1266
On: cpu b (#1)
Indicators:  ^bar
APU Status:  xsf, sd-on, pt-on, sdwp
CU Status:   pon, its
Instructions:
674016    001266 6100 00      rtcd      1266
674017   6 00076 7571 00      staq      pr6|76
```

Fault Type: (DF1) Page

17 (DF1) Page

```
scu 542,* -> pds$page_fault_data (70|0)
tra 442,* -> 44|1036 page_fault$page_fault (41|1062)
```

Group 6 Fault

Definition:

A faulted segment descriptor word (SDW) or page table word (PTW) with the corresponding directed fault number has been fetched by the appending unit.

Use in Multics:

The only directed fault number ever found in PTWs in Multics is 1. A Directed Fault 1 always means a page fault.

Fault Type: (DF1) Page

```
Machine Conditions For Page Fault Data At pds|0
Time Stored - 02/09/83  2212.7 hfh Wed (111511660647626100)

Pointer Registers

PR0 (ap) - 322|7120    >sl1>bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 244|12462   >pdd> zzzzzzzbBBBBBB>stack_4|12462
PR2 (bp) - 375|33714   >t>bound_oprcons_$write_log_|32
PR3 (bb) - 433|1753    >scl>log|1753
PR4 (lp) - 337|34352   >pdd> zzzzzzzbBBBBBB> BBBJMgffHxwhwg.area.linker|34352
PR5 (lb) - 244|12374   >pdd> zzzzzzzbBBBBBB>stack_4|12374
PR6 (sp) - 244|12520   >pdd> zzzzzzzbBBBBBB>stack_4|12520
PR7 (sb) - 433|0       >scl>log|0

Processor Registers

X0 - 33730 X1 - 12374 X2 - 257 X3 - 1665 X4 - 0 X5 - 331 X6 - 30 X7 - 2000
A Register - 000000000104 Q Register - 000000000000 E Register - 0
Timer Register - 000154400 Ring Alarm Register - 0
SCU masks      - 031460000014 631460000007
Fault Register - 010000000000  (OOB)

SCU Data at pds|30

    30    400375150201 000000000043 400433000200 000000156000
          034305000240 002001000400 300000100440 040100100540

(DF1) Page Fault (43)
By: 375|34305  >t>bound_oprcons_$write_log_|423
Referencing: 433|2001  >scl>log|2001
On: cpu c (#2)
Indicators:  ^bar, mif
APU Status:  sdwamm, sd-on, pt-on, ptw
CU Status:   rfi
Instructions:
 20136   300 000 100 440      mlr       (rl),(),fill(300)
 20137   040 100 100 540      mlr       (pr,rl),(pr),fill(040)

EIS Pointers and Lengths

    50    000400000000 000400000000 012502000060 772077777774
          001777000030 000000000040 012056000000 000000000077
```

5	Command	scu 512,* -> pds$fim_data (70\|60) tra 412,* -> fim$primary_fault_entry (34|404)

Group 4 Fault

Definition:

1. The processor attempted to load or read the interrupt mask register in a system controller in which it did not have an interrupt mask assigned.

2. The processor issued an XEC system controller command to a system controller in which it did not have an interrupt mask assigned.

3. The processor issued a connect to a system controller port that is masked OFF.

4. The selected system controller is in TEST mode and a condition determined by certain system controller maintenance panel switches has been trapped.

5. An attempt was made to load a pointer register with packed pointer data in which the BITNO field value was greater than or equal to 60(8).

Use in Multics:

Entries in the LOT are packed pointers initialized with bit offset values that provoke a command fault.

Fault Type: (DF0) Segment

16 (DF0) Segment

```
scu 540,* -> pds$fim_data (70|60)
tra 440,* -> fim$primary_fault_entry (34|404)
```

Group 6 Fault

Definition:

A faulted segment descriptor word (SDW) or page table word (PTW) with the corresponding directed fault number has been fetched by the appending unit.

Use in Multics:

The only directed fault number ever found in SDWs in Multics is 0. A Directed Fault 0 always means a segment fault.

Fault Type: (FT2) Linkage

24 (FT2) Linkage

```
scu 560,* -> pds$fim_data (70|60)
tra 460,* -> fim$primary_fault_entry (34|404)
```

Group 5 Fault

Definition:

The corresponding indirect then tally variation has been detected during virtual address formation.

Use in Multics:

A Fault Tag 2 occurs when a pointer is used that contains octal 46 in the last six bits of the first word. All external references in Multics are made using links in the object segment. Such a link is a pointer with octal 46 as the tag. The linker is called to handle the fault by changing the link into a valid ITS pointer.

Fault Type: (FT2) Linkage

```
Machine Conditions For Signal Data At pds|140
Time Stored - 02/09/83  2212.7 hfh Wed (111511660664301450)

Pointer Registers

PR0 (ap) - 270|7120    >sll>bound_sss_wired_$pll_operators_|1362
PR1 (ab) - 374|35642   >exl>continuum>bound_continuum_$con_request_table_|0
PR2 (bp) - 244|13406   >pdd> BNLBLDfBBBBBBB>stack_4|13406
PR3 (bb) - 244|13300   >pdd> BNLBLDfBBBBBBB>stack_4|13300
PR4 (lp) - 257|35762   >pdd> BNLBLDfBBBBBBB> BBBJMgfjDqdxCN.area.linker|35762
PR5 (lb) - 257|5552    >pdd> BNLBLDfBBBBBBB> BBBJMgfjDqdxCN.area.linker|5552
PR6 (sp) - 244|13300   >pdd> BNLBLDfBBBBBBB>stack_4|13300
PR7 (sb) - 244|0       >pdd> BNLBLDfBBBBBBB>stack_4|0

Processor Registers

X0 - 51443 X1 - 14452 X2 - 36352 X3 - 36546 X4 - 1717 X5 - 0 X6 - 0 X7 - 36
A Register - 000006000000 Q Register - 000000000000 E Register - 0
Timer Register - 000004423 Ring Alarm Register - 0

SCU Data at pds|170

   170    400374250041 000000000061 400257000046 000001462000
          051501100200 036270000020 036270352020 400306352120

(FT2) Linkage Fault (61)
By: 374|51501  >exl>continuum>bound_continuum_$continuum_command_|361
Referencing: 257|36270  >pdd> BNLBLDfBBBBBBB> BBBJMgfjDqdxCN.area.linker|36270
On: cpu a (#0)
Indicators:  cary, ^bar
APU Status:  xsf, sd-on, pt-on, fap
CU Status:
CT Hold: n*
Instructions:
735176    036270 3520 20      epp2      36270,*
735177   4 00306 3521 20      epp2      pr4|306,*
```

Fault Type: Shutdown

0	Shutdown	scu 500,* -> pds$fim_data (70\|60) tra 400,* -> fim$onc_start_shut_entry (34|14)

Group 7 Fault

Definition:

An external power shutdown condition has been detected. DC POWER shutdown will occur in approximately one millisecond.

Fault Type: Op Not Complete

11	Op Not Complete	scu 526,* -> pds$fim_data (70\|60) tra 426,* -> fim$onc_start_shut_entry (34|14)

Group 2 Fault

Any of the following will cause an operation not complete fault:

1. The processor has addressed a system controller to which it is not attached, that is, there is no main memory interface port having its ADDRESS ASSIGNMENT switches set to a value including the main memory address desired.

2. The addressed system controller has failed to acknowledge the processor.3. The processor has not generated a main memory access request or a direct operand within 1 to 2 milliseconds and is not executing the Delay Until Interrupt Signal (dis) instruction.

4. A main memory interface port received a data strobe without a preceding acknowledgement from the system controller that it had received the access request.

5. A main memory interface port received a data strobe before the data previously sent to it was unloaded.

Fault Type: Op Not Complete

```
Machine Conditions For Fim Data At pds|60
Time Stored - 02/12/83  0312.5 hfe Sat (111514443375723610)

Pointer Registers

PR0 (ap) - 40|7120      bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 240|4512     >sl1>stack_0.019|4512
PR2 (bp) - 240|1540     >sl1>stack_0.019|1540
PR3 (bb) - 227|16006    tty_buf|16006
PR4 (lp) - 122|31131    bound_355_wired$tty_space_man|525
PR5 (lb) - 46|10        dn355_data|10
PR6 (sp) - 240|3140     >sl1>stack_0.019|3140
PR7 (sb) - 240|1400     >sl1>stack_0.019|1400

Processor Registers

X0 - 31121 X1 - 0 X2 - 777773 X3 - 500 X4 - 0 X5 - 0 X6 - 7 X7 - 460
A Register - 000000000000 Q Register - 000000000137 E Register - 0
Timer Register - 777766410 Ring Alarm Register - 1

SCU Data at pds|110

   110    000122050011 000000000027 400300000006 000000000000
          015671000200 001266000000 700001352106 700001352106

Op Not Complete Fault (27)
By: 122|15671  bound_355_wired$fnp_multiplexer|4223
Referencing: 300|1266  CANNOT GET PATHNAME|1266
On: cpu a (#0)
Indicators:  ^bar
APU Status:  sd-on, pt-on, fabs
Instructions:
565416   7 00001 3521 06      epp2      pr7|1,ql
565
```

12 Startup

```
scu 530,* -> pds$fim_data (70|60)
tra 430,* -> fim$onc_start_shut_entry (34|14)
```

Group 1 Fault

Definition:

DC POWER has been truned on. When the POWER ON button is pressed, the processor is first initialized and then the startup fault is generated.

Fault Type: Parity

9	Parity	scu 522,* -> pds$fim_data (70\|60) tra 422,* -> fim$parity_entry (34|124)

Group 4 Fault

Definition:

1. The selected system controller has returned an illegal action signal with an illegal action code for one of the various main memory parity error conditions.

2. A cache memory data or directory parity error has occurred either for read, write, or block load. Cache status bits for the condition have been set in the cache mode register.

3. The processor has detected a parity error in the system controller interface port while either generating outgoing parity or verifying incoming parity.

Fault Type: Parity

```
Machine Conditions For Fim Data At pds|60
Time Stored - 03/06/83  1343.4 hfh Sun (111550652327420005)

Pointer Registers

PR0 (ap) - 76|240       prds|240
PR1 (ab) - 104|244      scs|244
PR2 (bp) - 75|60        pds|60
PR3 (bb) - 75|220       pds|220
PR4 (lp) - 15|312       ws_linkage|312
PR5 (lb) - 76|740       prds|740
PR6 (sp) - 76|1160      prds|1160
PR7 (sb) - 76|0         prds|0

Processor Registers

X0 - 46 X1 - 0 X2 - 124155 X3 - 0 X4 - 0 X5 - 127 X6 - 0 X7 - 777774
A Register - 000000000000 Q Register - 000000000000 E Register - 0
Timer Register - 000114573 Ring Alarm Register - 0

SCU Data at pds|110

   110    000032750021 000000170023 000172000000 000000000000
          001714001200 000000000224 001714710200 001676352220

Parity Fault (23)
Illegal Action Code (17) - Data Parity (SCU -> Store)
By: 32|1714  bound_interceptors$wired_fim|16
Referencing: 172|0  kst_seg|0
On: cpu a (#0)
Indicators:  par, ^bar
APU Status:  priv, xsf, sdwamm, sd-on, pt-on, fanp
CU Status:   its
CT Hold: ic*
Instructions:
546016    001714 7102 00      tra       1714 interrupt inhibit
546017    001676 3522 20      epp2      1676,* interrupt inhibit
```

Fault Type: Store

1	Store	scu 502,* -> pds$signal_data (70\|140) tra 402,* -> fim$signal_entry (34|300)

Group 4 Fault

Definition:

The processor attempted to select a disabled port, an out-of-bounds address was generated in the BAR mode or absolute mode, or an attempt was made to access a store unit that was not ready.

Fault Type: Store

```
Machine Conditions For Signal Data At pds|140
Time Stored - 03/29/83  1507.9 hfe Tue (111605613135576571)

Pointer Registers

PR0 (ap) - 40|7120     bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 240|5000    >sl1>stack_0.016|5000
PR2 (bp) - 76|100      prds|100
PR3 (bb) - 172|1000    kst_seg|1000
PR4 (lp) - 15|312      ws_linkage|312
PR5 (lb) - 144|13224   bound_system_faults$lock|3162
PR6 (sp) - 240|3760    >sl1>stack_0.016|3760
PR7 (sb) - 152|0       dirlockt_seg|0

Processor Registers

X0 - 2237 X1 - 4504 X2 - 2521 X3 - 777671 X4 - 2 X5 - 520 X6 - 514 X7 - 2747
A Register - 000000000000 Q Register - 000172013277 E Register - 0
Timer Register - 000137225 Ring Alarm Register - 0
SCU masks      - 000240000043 000340000000
Fault Register - 000200000262  ($CON B, CACHE-PAR IA)
                 (Ilegal Action on CPU Port D: Store Not Ready ( 13))

SCU Data at pds|170

   170    000032550023 000000130003 000240000217 000000000000
          002747100200 013300000000 000000710217 000476710220

Store Fault ( 3)
Illegal Action Code (13) - Store Not Ready
By: 32|2747  bound_interceptors$fim_util|315
Referencing: 240|13300  >sl1>stack_0.016|13300
On: cpu c (#2)
Indicators:  cary, ^bar
APU Status:  priv, sdwamm, sd-on, pt-on, fanp
Instructions:
470576    000000 7102 17      tra       0,7 interrupt inhibit
470577    000476 7102 20      tra       476,* interrupt inhibit
```

Fault Type: Trouble

31 Trouble

```
scu 576,* -> pds$fim_data (70|60)
tra 476,* -> fim$primary_fault_entry (34|404)
```

Group 2 Fault

Definition:

The trouble fault is defined as the occurrence of a fault during the fetch or execution of a fault trap pair or interrupt trap pair. Such faults may be hardware generated (for example, operation not complete or parity), or operating system generated (e.g., the page containing a trap pair instruction operand is missing).

Fault Type: Trouble

```
Machine Conditions For Fim Data At pds|60
Time Stored - 03/29/83  1507.8 hfe Tue (111605613070163325)

Pointer Registers

PR0 (ap) - 270|7120    >sl1>bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 244|104760  >pdd> BdbBCBpbBBBBBB>stack_4|104760
PR2 (bp) - 244|106760  >pdd> BdbBCBpbBBBBBB>stack_4|106760
PR3 (bb) - 422|61224   >pdd> BdbBCBpbBBBBBB> BBBJM1PcJ1jbLc.area.compose|61224
PR4 (lp) - 257|54074   >pdd> BdbBCBpbBBBBBB> BBBJM1PbxgCHhJ.area.linker|54074
PR5 (lb) - 427|367314  >pdd> BdbBCBpbBBBBBB> BBBJM1PcKDQCdK.area.compose|367314
PR6 (sp) - 244|102620  >pdd> BdbBCBpbBBBBBB>stack_4|102620
PR7 (sb) - 427|17524   >pdd> BdbBCBpbBBBBBB> BBBJM1PcKDQCdK.area.compose|17524

Processor Registers

X0 - 1757 X1 - 102104 X2 - 32 X3 - 777732 X4 - 2 X5 - 0 X6 - 61 X7 - 2140
A Register - 000000000001 Q Register - 777777777777 E Register - 0
Timer Register - 000037651 Ring Alarm Register - 1
Fault Register - 010000000000  (OOB)

SCU Data at pds|110

   110    000032550022 000000130077 000000000000 000000000000
          002747100200 013300000000 000000000000 000000000000

Trouble Fault (77)
Illegal Action Code (13) - Store Not Ready
By: 32|2747  bound_interceptors$fim_util|315
Referencing: 0|13300  dseg|13300
On: cpu a (#0)
Indicators:  cary, ^bar
APU Status:  priv, sdwamm, sd-on, pt-on, fanp
Instructions:
470516    000000 0000 00      ....      0
470517    000000 0000 00      ....      0
```

Fault Type: Directed Fault 2

18 Directed Fault 2 scu 544,* -> pds$signal_data (70|140)
tra 444,* -> fim$signal_entry (34|300)

Group 6 Fault

Definition:

A faulted segment descriptor word (SDW) or page table word (PTW) with the corresponding directed fault number has been fetched by the appending unit.

Fault Type: Directed Fault 3

19	Directed Fault 3	scu 546,* -> pds$signal_data (70\|140) tra 446,* -> fim$signal_entry (34|300)

Group 6 Fault

Definition:

A faulted segment descriptor word (SDW) or page table word (PTW) with the corresponding directed fault number has been fetched by the appending unit.

Fault Type: Derail

6	Derail	scu 514,* -> pds$signal_data (70\|140) tra 414,* -> fim$drl_entry (34|30)

Group 5 Fault

Definition:

The Derail instruction has been decoded.

Use in Multics:

Certain supervisor programs execute a DRL instruction rather than calling syserr to provoke a system crash.

Fault Type: Derail

```
Bootload CPU Registers at Time of Dump

Pointer Registers

PR0 (ap) - 76|240       prds|240
PR1 (ab) - 104|245      scs|245
PR2 (bp) - 76|240       prds|240
PR3 (bb) - 1|0          fault_vector|0
PR4 (lp) - 15|312       ws_linkage|312
PR5 (lb) - 76|320       prds|320
PR6 (sp) - 76|1160      prds|1160
PR7 (sb) - 76|0         prds|0

Processor Registers

X0 - 3036 X1 - 2561 X2 - 164654 X3 - 0 X4 - 0 X5 - 1770 X6 - 0 X7 - 0
A Register - 010340657200 Q Register - 010011630200 E Register - 0
Timer Register - 767652141 Ring Alarm Register - 0
Descriptor Segment Base Register - 035444700000 017770000024
Mode Register - 000000000021
Cache Mode Register - 177532005003

SCU Data
  3540    000032450001 000000000015 000032000000 000000000000
          003146101200 000000000000 000000002200 000000002200

Derail Fault (15)
By: 32|3146  bound_interceptors$sys_trouble|132
Referencing: 32|0  bound_interceptors$fim|0
On: cpu a (#0)
Indicators:  cary, par, ^bar
APU Status:  priv, sd-on, pt-on
Instructions:
  3546    000000 0022 00      drl       0 interrupt inhibit
  3547    000000 0022 00      drl       0 interrupt inhibit
```

Fault Type: MME 1

2	MME 1	scu 504,* -> pds$signal_data (70\|140) tra 404,* -> fim$signal_entry (34|300)

Group 5 Fault

Definition:

The corresponding Master Mode Entry instruction has been decoded.

Fault Type: Fault Tag 1

3	Fault Tag 1	scu 506,* -> pds$signal_data (70\|140) tra 406,* -> fim$signal_entry (34|300)

Group 5 Fault

Definition:

The corresponding indirect then tally variation has been detected during virtual address formation.

Use in Multics:

A Fault Tag 1 occurs when a pointer is used that contains octal 40 in the last six bits of the first word. There is no normal use of a Fault Tag 1 fault. It is almost always the result of a reference to an uninitialized pointer. Because the most common ASCII character is octal 040 (a blank), an uninitialized pointer frequently has a tag of 40 if there was previously ASCII data in the zone used as a pointer. For this reason the error message when a Fault Tag 1 is signalled in the user process says "ASCII data where pointer expected."

Fault Type: Lockup

7	Lockup	scu 516,* -> pds$signal_data (70\|140) tra 416,* -> fim$signal_entry (34|300)

Group 4 Fault

The program is in a code sequence which has inhibited sampling for interrupts (whether present or not) and group 7 faults for longer than the prescribed time. In absolute mode or privileged mode the lockup time is 32 milliseconds. In normal mode or BAR mode the lockup time is specified by the setting for the lockup time in the cache mode register. The lockup time is program settable to 2, 4, 8, or 16 milliseconds.

While in absolute mode or privileged mode the lockup fault is signalled at the end of the time limit set in the lockup timer but is not recognized until the 32 millisecond limit. If the processor returns to normal mode or BAR mode after the fault has been signalled but before the 32 millisecond limit, the fault is recognized before any instruction in the new mode is executed.

Fault Type: Lockup

```
Machine Conditions For Signal Data At pds|140
Time Stored - 03/26/83  1429.1 hfh Sat (111602011334057774)

Pointer Registers

PR0 (ap) - 67|1136      inzr_stk0|1136
PR1 (ab) - 104|247      scs|247
PR2 (bp) - 75|60        pds|60
PR3 (bb) - 75|220       pds|220
PR4 (lp) - 15|312       ws_linkage|312
PR5 (lb) - 76|340       prds|340
PR6 (sp) - 77777|1      NULL POINTER
PR7 (sb) - 67|0         inzr_stk0|0

Processor Registers

X0 - 42 X1 - 2561 X2 - 350 X3 - 4 X4 - 0 X5 - 32 X6 - 0 X7 - 0
A Register - 500000154700 Q Register - 000000000000 E Register - 0
Timer Register - 777737756 Ring Alarm Register - 0

SCU Data at pds|170

   170    000032450001 000000000017 000076000000 000000566000
          002651101200 002642000000 500000154600 200044716300

Lockup Fault (17)
By: 32|2651  bound_interceptors$fim_util|217
Referencing: 76|2642  prds|2642
On: cpu a (#0)
Indicators:  cary, par, ^bar
APU Status:  priv, sd-on, pt-on
Instructions:
411176    500000 1546 00      sptr      -300000 interrupt inhibit
411177   2 00044 7163 00      xec       pr2|44 interrupt inhibit

EIS Pointers and Lengths

   210    000400000000 000400000000 001763000000 756077777735
          000000000000 000077777670 002005000000 000077777735
```

Fault Type: Illegal Procedure

10 Illegal Procedure scu 524,* -> pds$signal_data (70|140)
tra 424,* -> fim$signal_entry (34|300)

Group 5 Fault

Definition:

1. An illegal operation code has been decoded or an illegal instruction sequence has been encountered.

2. An illegal modifier or modifier sequence has been encountered during virtual address formation.

3. An illegal address has been given in an instruction for which the ADDRESS field is used for register selection.

4. An attempt was made to execute a privileged instruction in normal mode or BAR mode.

5. An illegal digit was encountered in a decimal numeric operand.

6. An illegal specification was found in an EIS operand descriptor.

The conditions for the fault will be set in the fault register, word 1 of the Control Unit Data, or in both.

Fault Type: Illegal Procedure

```
Machine Conditions For Signal Data At pds|140
Time Stored - 02/21/83  1235.6 hfh Mon (111530333460565000)

Pointer Registers

PR0 (ap) - 75|560      pds|560
PR1 (ab) - 221|260     syserr_daemon_stack|260
PR2 (bp) - 76|1660     prds|1660
PR3 (bb) - 17|0        sst_seg|0
PR4 (lp) - 15|762      ws_linkage|762
PR5 (lb) - 15|762      ws_linkage|762
PR6 (sp) - 76|1420     prds|1420
PR7 (sb) - 76|0        prds|0

Processor Registers

X0 - 10 X1 - 1110 X2 - 352 X3 - 200000 X4 - 1 X5 - 0 X6 - 2677 X7 - 2702
A Register - 120000106664 Q Register - 000000000000 E Register - 0
Timer Register - 000044167 Ring Alarm Register - 0
Fault Register - 400000000000  (ILL OP)

SCU Data at pds|170

   170    000035450001 200000000025 000035000020 000000000000
          003042500200 010000000000 010001000220 000005220100

Illegal Procedure Fault (25), Illegal Op Code
By: 35|3042  bound_page_control$post_purge|156
Referencing: 35|10000  bound_page_control$pc|146
On: cpu a (#0)
Indicators:  zero, cary, ^bar
APU Status:  priv, sd-on, pt-on
Instructions:
667776    010001 0002 20      ....      10001,* interrupt inhibit
667777   0 00005 2201 00      ldx0      pr0|5
```

Fault Type: Illegal Procedure

```
Machine Conditions For Signal Data At pds|140
Time Stored - 03/29/83  1508.0 hfe Tue (111605613146467522)

Pointer Registers

PR0 (ap) - 40|7120      bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 230|5540     tty_buf|5540
PR2 (bp) - 230|30354    tty_buf|30354
PR3 (bb) - 46|100       dn355_data|100
PR4 (lp) - 122|2463     bound_355_wired$dn355|2463
PR5 (lb) - 230|27344    tty_buf|27344
PR6 (sp) - 76|1260      prds|1260
PR7 (sb) - 5|410        dn355_mailbox|410

Processor Registers

X0 - 2500 X1 - 2040 X2 - 245576 X3 - 245560 X4 - 1730 X5 - 1104 X6 - 2254 X7 - 1300
A Register - 400000000000 Q Register - 000000000102 E Register - 0
Timer Register - 000364003 Ring Alarm Register - 1
Fault Register - 400000000000  (ILL OP)

SCU Data at pds|170

   170    000122050001 200000000025 000122000000 000000000000
          002500300200 000000000000 000000000000 000000000000

Illegal Procedure Fault (25), Illegal Op Code
By: 122|2500  bound_355_wired$dn355|2500
Referencing: 122|0  bound_355_wired$dn355|0
On: cpu a (#0)
Indicators:  neg, cary, ^bar
APU Status:  sd-on, pt-on
Instructions:
275276    000000 0000 00      ....      0
275277    000000 0000 00      ....      0
```

Fault Type: Overflow

13 Overflow	scu 532,* -> pds$signal_data (70\|140) tra 432,* -> fim$signal_entry (34|300)

Group 3 Fault

Definition:

An arithmetic overflow, exponent overflow, exponent underflow, or EIS truncation fault has been generated. The generation of this fault is inhibited when the overflow mask indicator is ON. Resetting of the overflow mask indicator to OFF does not generate a fault from previously set indicators. The overflow mask state does not affect the setting, testing or storing of indicators. The determination of the specific overflow condition is by indicator testing by the operating supervisor.

Fault Type: Overflow

```
Time Stored - 03/31/83  1752.0 hfe Thu (111610333767301724)

Pointer Registers

PR0 (ap) - 40|7120     bound_sss_wired_$pl1_operators_|1362
PR1 (ab) - 230|34220   tty_buf|34220
PR2 (bp) - 240|3520    >sl1>stack_0.011|3520
PR3 (bb) - 240|1420    >sl1>stack_0.011|1420
PR4 (lp) - 123|13240   bound_uncp_wired
PR5 (lb) - 240|1424    >sl1>stack_0.011|1424
PR6 (sp) - 240|3140    >sl1>stack_0.011|3140
PR7 (sb) - 240|3512    >sl1>stack_0.011|3512

Processor Registers

X0 - 33115 X1 - 3560 X2 - 777773 X3 - 500 X4 - 0 X5 - 4 X6 - 7 X7 - 500
A Register - 400000000000 Q Register - 000000000000 E Register - 0
Timer Register - 000115235 Ring Alarm Register - 0
SCU masks      - 000240000043 003140000000

SCU Data at >sl1>stack_0.011|4010

  4010    000123050011 000000000033 000244000200 000000000000
          014244340200 004136000000 000033735000 000011236007

Overflow Fault (33)
By: 123|14244  bound_uncp_wired
Referencing: 244|4136  >pdd> BxbBDCXbBBBBBB>stack_4|4136
On: cpu c (#2)
Indicators:  neg, cary, ovfl, ^bar
APU Status:  sd-on, pt-on, fabs
Instructions:
570716    000033 7350 00      als       33
570
```

Fault Type: Misc

Divide Check

14 Divide Check

```
scu 534,* -> pds$signal_data (70|140)
tra 434,* -> fim$signal_entry (34|300)
```

Group 3 Fault

Definition:

A divide check fault occurs when the actual division cannot be carried out for one of the reasons specified with individual divide instructions.

Fault Type: MME 2

21 MME 2

```
scu 552,* -> pds$signal_data (70|140)
tra 452,* -> fim$signal_entry (34|300)
```

Group 5 Fault

Definition:

The corresponding Master Mode Entry instruction has been decoded.

Fault Type: MME 3

22 MME 3

```
scu 554,* -> pds$signal_data (70|140)
tra 454,* -> fim$signal_entry (34|300)
```

Group 5 Fault

Definition:

The corresponding Master Mode Entry instruction has been decoded.

Fault Type: Misc

Fault Type: MME 4

23 MME 4 scu 556,* -> pds$signal_data (70|140)
tra 456,* -> fim$signal_entry (34|300)

Group 5 Fault

Definition:

The corresponding Master Mode Entry instruction has been decoded.

Fault Type: Fault Tag 3

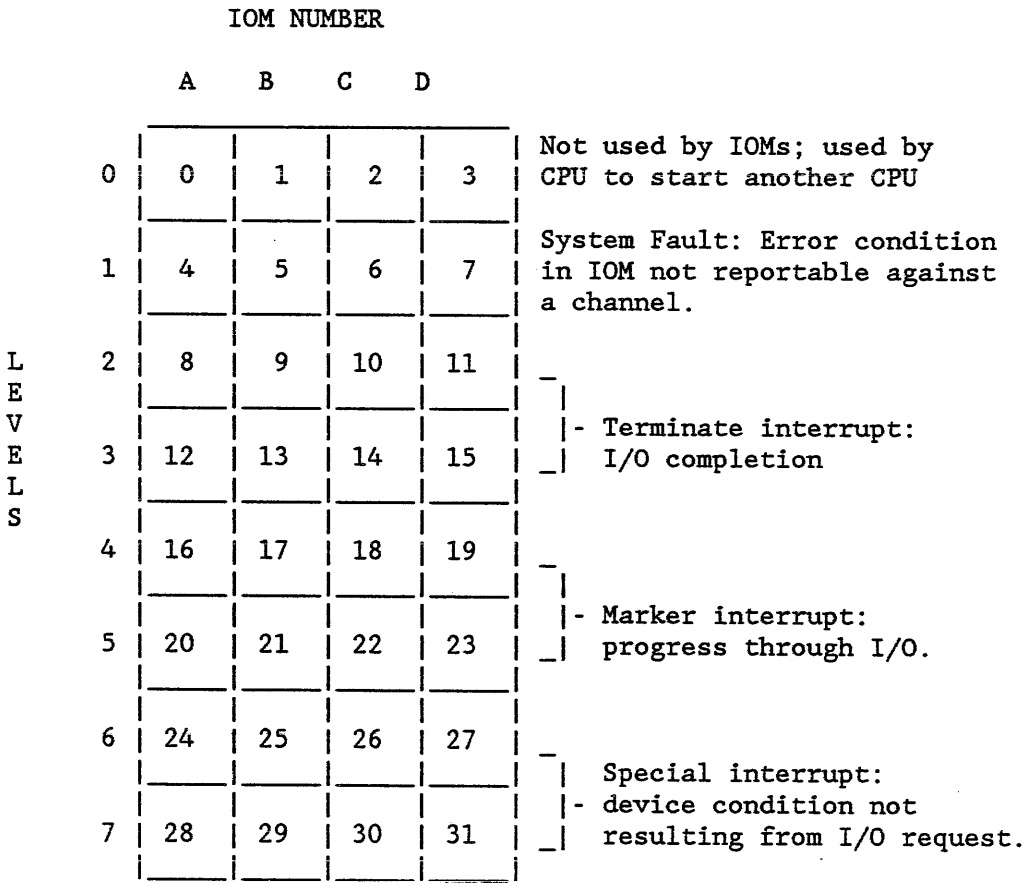
25 Fault Tag 3 scu 562,* -> pds$signal_data (70|140)
tra 462,* -> fim$signal_entry (34|300)

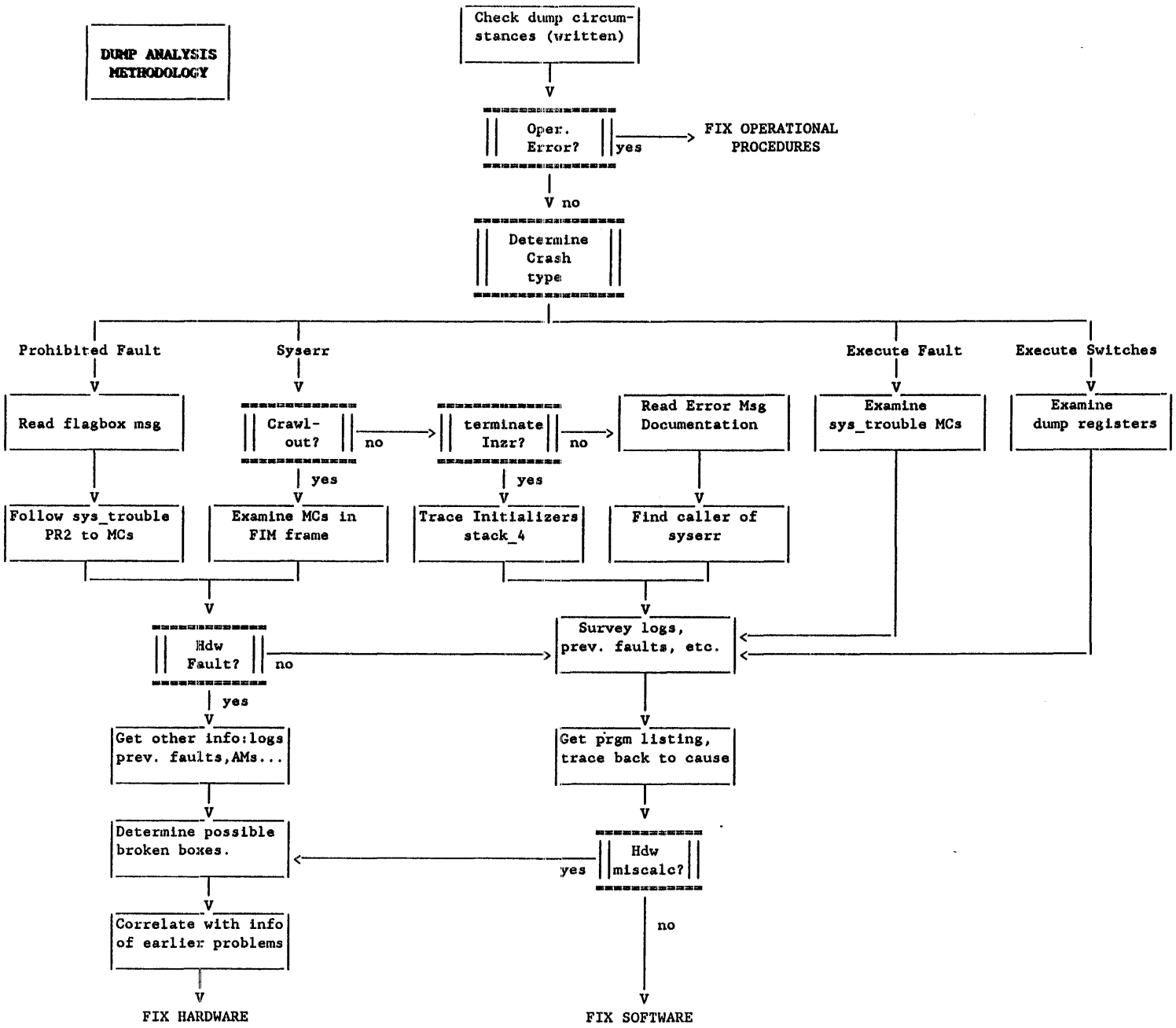
Group 5 Fault

Definition:

The corresponding indirect then tally variation has been detected during virtual address formation.

INTERRUPT TYPES




DUMP ANALYSIS METHODOLOGY
Check dump circum-stances (written)
Oper. Error?
yes
FIX OPERATIONAL PROCEDURES
no
Determine Crash type
Prohibited Fault
Syserr
Execute Fault
Execute Switches
Read flagbox msg
Crawl-out?
no
terminate Inzr?
no
Read Error Msg Documentation
Examine sys_trouble MCs
Examine dump registers
yes
yes
Follow sys_trouble PR2 to MCs
Examine MCs in FIM frame
Trace Initializers stack_4
Find caller of syserr
Hdw Fault?
no
Survey logs, prev. faults, etc.
yes
Get other info:logs prev. faults,AMs...
Get prgm listing, trace back to cause
Determine possible broken boxes.
Hdw miscalc?
yes
Correlate with info of earlier problems
no
FIX HARDWARE
FIX SOFTWARE

```
/*  BEGIN INCLUDE FILE ... apte.incl.pl1 */

/* Modified 1984-11-11 by E. Swenson for IPC event channel validation. */

dcl  aptep pointer;

dcl 1 apte based (aptep) aligned,                    /* APT entry declaration for an active (known) process */
    2 thread unaligned,                              /* List thread */
      3 fp bit (18),                                 /* Forward pointer */
      3 bp bit (18),                                 /* Backward pointer */
    2 flags unaligned,                               /* Flags and miscellaneous */
      3 mbz bit (1),                                 /* This bit must be zero (sentinel bit) */
      3 wakeup_waiting bit (1),                      /* ON if process has received wakeup */
      3 stop_pending bit (1),                        /* ON if process has received stop connect */
      3 pre_empted bit (1),                          /* ON if process is being pre-empted by get_processor */
      3 hproc bit (1),                               /* ON if process is hardcore process */
      3 loaded bit (1),                              /* ON if required per-process pages are in memory and wired */
      3 eligible bit (1),                            /* ON if process is eligible */
      3 idle bit (1),                                /* ON if  this is an idle process */
      3 interaction bit (1),                         /* ON if process has interacted recently */
      3 pre_empt_pending bit (1),                    /* ON if process has received pre-empt connect */
      3 default_procs_required bit (1),              /* ON if apte.procs_required is system default */
      3 realtime_burst bit (1),                      /* ON if next eligibility is realtime */
      3 always_loaded bit (1),                       /* ON if process is not to be unloaded */
      3 dbr_loaded bit (1),                          /* ON if DBR is loaded on some CPU */
      3 being_loaded bit (1),                        /* ON if somebody loading this process */
      3 shared_stack_0 bit (1),                      /* ON if a shared stack_0 is assigned */
      3 page_wait_flag bit (1),                      /* flag ON if waiting for page */
      3 firstsw bit (1),                             /* OFF until process is intialized */
      3 state bit (18),                              /* execution state */
    2 page_faults fixed bin (35),                    /* total page faults for the process */
    2 processid bit (36),                            /* bit 0-17: offset of ATPE */
                                                     /* bit 18-35: sequential number */
    2 te fixed bin (35),                             /* virtual time since eligibility award */
    2 ts fixed bin (35),                             /* virtual time since scheduling */
    2 ti fixed bin (35),                             /* virtual time since interaction */
    2 timax fixed bin (35),                          /* maximum value allowed for apte.ti */

/* * * * * * * * */

    2 ipc_pointers unaligned,
      3 event_thread bit (18),                       /* relative pointer to ITT list */
      3 pad3 bit (18),
    2 ips_message bit (36),                          /* IPS signals pending */
    2 asteps unaligned,                              /* relative ASTE pointers */
      3 pds bit (18),                                /* PDS (per-process) */
      3 dseg bit (18),                               /* DSEG (per-process) */
      3 prds bit (18),                               /* PRDS (per-processor) */
    2 savex7 bit (18) unaligned,                     /* x7 at call to getwork (return point in pxss) */
    2 term_processid bit (36),                       /* process to send wakeup at temination */
```

```
    2 lock_id bit (36),                          /* File System unqieu ID associated with process */
    2 time_used_clock fixed bin (71),            /* Total CPU time when process last lost CPU */

/* * * * * * * * */

    2 wait_event bit (36) aligned,               /* Event ID process awaiting */
    2 wct_index bit (18) unaligned,              /* rel offset of WCTE */
    2 flags2 unaligned,
      3 priority_scheduling bit (1),             /* ON if guaranteed eligibility */
      3 special_wakeups bit (6),                 /* Special wakeup channels */
      3 pad7 bit (7),
      3 batch bit (1),                           /* ON if absentee */
      3 pr_tag bit (3),                          /* CPU tag running or last run */
    2 state_change_time fixed bin (71),          /* Time apte.state last changed */
    2 alarm_event fixed bin (71),                /* wakeup event for alarm clock manager */
    2 alarm_time_thread bit (18) unaligned,      /* thread of processes with pending alarms */
    2 alarm_time bit (54) unaligned,             /* wakeup time for alarm */

/* * * * * * * */

    2 term_channel fixed bin (71),               /* wakeup event for account overflow */
    2 ws_size fixed bin,                         /* working set estimate for the process */
    2 temax fixed bin (35),                      /* maximum eligibility slice (vcpu) */
    2 deadline fixed bin (71),                   /* time of next run */
    2 lock bit (18) unaligned,                   /* 0 => APTE locked, unlocked => return point of last unlock */
    2 unusable bit (18) unaligned,               /* locking routines destroy */
    2 cpu_monitor fixed bin (35),                /* if not 0, send wakeup to term_processid when virtual cpu
                                                    reaches this (units = 1/1024 sec) */
    2 paging_measure fixed bin (71),             /* cumulative memory units */
    2 access_authorization bit (72),             /* authorization of this process */
    2 dbr fixed bin (71),                        /* DBR value (constant since DSEG entry-held) */

    2 virtual_cpu_time fixed bin (71),           /* cumulative virtual CPU time for the process */
    2 ittes_sent fixed bin (18),                 /* Unprocessed ITTs sent by this process */
    2 ittes_got fixed bin (18),                  /* Unprocessed ITTs received by this process */

/*  Cells used to drive and instrument finite-state model for response time
    measurement. Maintained by meter_response_time */

    2 current_response_state fixed bin (17) unaligned,   /* Process state in modle */
    2 pad18 bit (18) unaligned,
    2 number_processing fixed bin (35),          /* Number interactions */
    2 last_response_state_time fixed bin (71),   /* Clock time at last response state change */
    2 total_processing_time fixed bin (71),      /* Total interaction processing time */

/* * * * * * * */

    2 begin_interaction_vcpu fixed bin (71),     /* Virtual cpu at beginning of last interaction */

/*  End of cells for finite-state model */

    2 saved_temax fixed bin (35),                /* temax at eligibility award */
    2 procs_required bit (8) unaligned,          /* bit mask of CPUs this process can run */
    2 pad4 bit (28) unaligned,
    2 ipc_r_offset fixed bin (18) unsigned,
    2 ipc_r_factor fixed bin (35) unsigned,
```

```
    2 apad (10) fixed bin (35);

/*  END INCLUDE FILE ... apte.incl.pl1 */
```

```
/*        BEGIN INCLUDE FILE ...aste.incl.pl1 ... */

/* Template for an AST entry. Length = 12 words. */

/* Words 0 to 7, and 11 are read by PC; they are read and modified by SC.
   Words 8, 9 and 10 are modified by PC; they should never be modified without locking the PC lock */
/* Modified January 1985 by Keith Loepere for multi_class. */

dcl  astep ptr;

dcl 1 aste based (astep) aligned,

    (2 fp bit (18),                          /* forward  used list rel pointer */
    2 bp bit (18),                           /* backward used list rel pointer */

    2 infl bit (18),                         /* ptr to NEXT  in list of ASTE's of my brothers */
    2 infp bit (18),                         /* ptr to FIRST in list of ASTE's of my children */

    2 strp bit (18),                         /* rel pointer to process trailer */
    2 par_astep bit (18),                    /* rel pointer to parent aste */

    2 uid bit (36),                          /* segment unique id */

    2 msl bit (9),                           /* maximum segment length in 1024 word units */
    2 pvtx fixed bin (8),                    /* physical volume table index */
    2 vtocx fixed bin (17),                  /* vtoc entry index */

    2 usedf bit (1),                         /* ast entry is being used if non-zero */
    2 init bit (1),                          /* used bit - insure 1 lap */
    2 gtus bit (1),                          /* global transparent usage switch */
    2 gtms bit (1),                          /* global transparent modified switch */
    2 hc bit (1),                            /* hard core segment */
    2 hc_sdw bit (1),                        /* aste with sdw for hardcore seg if non-zero */
    2 any_access_on bit (1),                 /* any sdw allows access, unless write_access_on */
    2 write_access_on bit (1),               /* any sdw allows write access */
    2 inhibit_cache bit (1),                 /* flag not to reset above bits */
    2 explicit_deact_ok bit (1),             /* set if user can deactivate seg */
    2 deact_error bit (1),                   /* set if error occurred while deactivating */
    2 hc_part bit (1),                       /* set if pages are in a hardcore partition */
    2 fm_damaged bit (1),                    /* set if filemap checksum was ever bad */
    2 multi_class bit (1),                   /* set if page_control should watch state changes to this segment */
    2 pad1 bit (2),                          /* 00 */
    2 dius bit (1),                          /* dumper in use switch */
    2 nid bit (1),                           /* if on prevents addtion to incremental dump map */
    2 dmpr_pad bit (1),
    2 ehs bit (1),                           /* entry hold switch */
    2 nqsw bit (1),                          /* no quota switch - no checking for pages of this seg */
    2 dirsw bit (1),                         /* directory switch */
    2 master_dir bit (1),                    /* master dir - a root for the log volume */
    2 volmap_seg bit (1),                    /* volmap_seg for some volume */
```

```
    2 tqsw (0:1) bit (1),                       /* terminal quota switch - (0) for non dir pages */
    2 pad_ic bit (10),                          /* Used to be aste.ic */

    2 dtu bit (36),                             /* date and time segment last used */

    2 dtm bit (36),                             /* date and time segment last modified */


    2 quota (0:1) fixed bin (18) unsigned,      /* sec storage quota - (0) for non dir pages */

    2 used (0:1) fixed bin (18) unsigned,       /* sec storage used  - (0) for non dir pages */

    2 csl bit (9),                              /* current segment length in 1024 words units */
    2 fmchanged bit (1),                        /* turned on by page if file map changed */
    2 fms bit (1),                              /* file modified switch */
    2 npfs bit (1),                             /* no page fault switch */
    2 gtpd bit (1),                             /* global transparent paging device switch */
    2 dnzp bit (1),                             /* don't null out if zero page switch */
    2 per_process bit (1),                      /* use master quota for this entry */
    2 ddnp bit (1),                             /* don't deposit nulled pages */
    2 pad2 bit (2),
    2 records bit (9),                          /* number of records used by the seg in sec storage */
    2 np bit (9),                               /* number of pages in core */


    2 ht_fp bit (18),                           /* hash table forward rel pointer */
    2 fmchanged1 bit (1),                       /* value of "fmchanged" saved by pc$get_file_map */
    2 damaged bit (1),                          /* PC declared segment unusable */
    2 pack_ovfl bit (1),                        /* page fault on seg would cause pack overflow */
    2 synchronized bit (1),                     /* Data Management synchronized segment */
    2 pad3 bit (6),                             /* 000000000 */
    2 ptsi bit (2),                             /* page table size index */
    2 marker bit (6)) unaligned;                /* marker to indicate last word of ASTE */

dcl  asta (0 : 8000) bit (36*12 /* sst-> sst.astsize */) based aligned;


dcl 1 aste_part aligned based (astep),

    2 one bit (36) unaligned,                   /* fp and bp */
    2 two bit (36*11 - 8) unaligned,            /* part that has to be zeroed when ASTE is freed */
    2 three bit (8) unaligned;                  /* ptsi and marker */


dcl 1 seg_aste based (astep) aligned,           /* Overlay because quota is only for dirs */
    2 pad1 bit (8*36),
    2 usage fixed bin (35),                     /* page fault count: overlays quota */
    2 pad2 bit (3*36);

/*        END INCLUDE FILE ... aste.incl.pl1 */
```

```
/* BEGIN INCLUDE FILE ... bos_dump.incl.pl1 ... */
/* Modified 1 September 1976 */
/* Modified 11/11/80 by J. A. Bush for the DPS8/70M CPU */
/* Modified 6/12/81 by Rich Coppola to extend the dps8 extended fault reg to
   15 bits  */
/* Modified 02/23/81, W. Olin Sibert, to describe old and new FDUMP styles */


dcl  dumpptr ptr;                                   /* pointer to following structure */

dcl 1 dump based (dumpptr) aligned,                 /* header of dump by fdump */
    2 dump_header aligned like dump_header,

    2 segs (1008),                                  /* segment array */
      3 segno bit (18) unal,                        /* segment number */
      3 length bit (18) unal,                       /* length of segment in sector sized blocks */

    2 amptwregs (0 : 63) bit (36),                  /* assoc. mem. page table word regs */
    2 amptwptrs (0 : 63) bit (36),                  /* assoc. mem. page table word pointers */
    2 amsdwregs (0 : 63) bit (72),                  /* assoc. mem. segment descriptor word registers */
    2 amsdwptrs (0 : 63) bit (36),                  /* assoc. mem. segment descriptor word pointers */

    2 ouhist (0 : 63) bit (72),                     /* operations unit history registers */
    2 cuhist (0 : 63) bit (72),                     /* control unit history registers */
    2 duhist (0 : 63) bit (72),                     /* decimal unit history registers */
    2 auhist (0 : 63) bit (72),                     /* appending unit history registers */

    2 prs (0 : 7) ptr,                              /* pointer registers */

    2 regs aligned like dump_registers,             /* assorted machine registers */

    2 low_order_port bit (3),                       /* from which clock is read */
    2 pad4 bit (36),
    2 mctime fixed bin (52),                        /* time conditions were taken */
    2 pad5 (0 : 3) bit (36),

    2 misc_registers like dump_misc_registers,      /* Assorted registers & processor data */

    2 ptrlen (0 : 7) bit (36),                      /* pointers and lengths for EIS */

    2 coreblocks (0 : 7),
      3 num_first bit (18) unal,                    /* first addr in coreblock */
      3 num_blocks bit (18) unal,                   /* number of blocks used */
    2 pad7 (112) fixed bin;


dcl 1 dump_header aligned based,                    /* Standard header for FDUMP */
    2 words_dumped fixed bin (35),                  /* total words in dump */
    2 valid bit (1),                                /* = 1 if there is a 6180 dump to be had */
    2 time fixed bin (71),                          /* time of dump */
```

```
    2 erfno fixed bin (18),                  /* Error Report Form Number */
    2 num_segs fixed bin,                    /* number of segments dumped */
    2 valid_355 bit (1),                     /* = 1 if there is a dn355 dump to be had */
    2 dumped_355s bit (4),                   /* indicates which 355s were dumped */
    2 time_355 fixed bin (71),               /* time of 355 dump */
    2 version fixed bin,                     /* currently 2 */
    2 pad0 (5) fixed bin;                    /* pad0 to 16 words */

dcl 1 dump_registers aligned based,          /* Standard (SREG) arrangement of registers in dump */
   (2 x (0 : 7) bit (18),                    /* index registers */
    2 a bit (36),                            /* the a register */
    2 q bit (36),                            /* the q register */
    2 e bit (8),                             /* the e register */
    2 pad2 bit (28),                         /* pad */
    2 t bit (27),                            /* timer register */
    2 pad3 bit (6),                          /* pad */
    2 ralr bit (3)) unaligned;               /* ring alarm register */

dcl 1 dump_misc_registers aligned based,
    2 scu (0 : 7) bit (36),                  /* from store control unit instr. */
    2 mcm (0 : 7) bit (72),                  /* memory controller masks every 64 K */
    2 dbr bit (72),                          /* descriptor segment base register */
    2 intrpts bit (36),                      /* interrupts */
    2 bar bit (36),                          /* base address register */
    2 modereg bit (36),                      /* mode register */
    2 cmodereg bit (36),                     /* cache mode register */
    2 faultreg bit (36),                     /* fault register */
    2 ext_fault_reg bit (15) unaligned,      /* DPS8 extended fault register */
    2 pad6 bit (21) unaligned;



dcl 1 v1_dump aligned based (dumpptr),       /* Old version of FDUMP (pre March, 1981) */
    2 dump_header aligned like dump_header,

    2 segs (688),                            /* segment array */
      3 segno bit (18) unal,                 /* segment number */
      3 length bit (18) unal,                /* length of segment in sector sized blocks */

    2 amsdwregs (0 : 15) bit (72),           /* assoc. mem. segment descriptor word registers */
    2 amsdwptrs (0 : 15) bit (36),           /* assoc. mem. segment descriptor word pointers */
    2 amptwregs (0 : 15) bit (36),           /* assoc. mem. page table word regs */
    2 amptwptrs (0 : 15) bit (36),           /* assoc. mem. page table word pointers */
    2 pad1 (0 : 15) bit (36),

    2 ouhist (0 : 15) bit (72),              /* operations unit history registers */
    2 cuhist (0 : 15) bit (72),              /* control unit history registers */
    2 auhist (0 : 15) bit (72),              /* appending unit history registers */
    2 duhist (0 : 15) bit (72),              /* decimal unit history registers */

    2 prs (0 : 7) ptr,                       /* pointer registers */

    2 regs aligned like dump_registers,      /* assorted machine registers */

    2 mctime fixed bin (52),                 /* time conditions were taken */
    2 pad4 (0 : 5) bit (36),
```

```
    2 misc_registers aligned like dump_misc_registers,      /* Assorted registers */

    2 pad5 bit (36),
    2 ptrlen (0 : 7) bit (36),                              /* pointers and lengths for EIS */
    2 pad6 (15) bit (36),
    2 low_order_port bit (3),                               /* from which clock was read */

    2 coreblocks (0 : 7),
      3 num_first bit (18) unal,                            /* first addr in coreblock */
      3 num_blocks bit (18) unal;                           /* number of blocks used */


dcl  DUMP_VERSION_1 fixed bin internal static options (constant) init (1);
dcl  DUMP_VERSION_2 fixed bin internal static options (constant) init (2);

/* END INCLUDE FILE ... bos_dump.incl.pl1 ... */
```

```
/* BEGIN INCLUDE FILE cmp.incl.pl1 --- October 1982 */
/* Note: This include file has an ALM counterpart NOT made with cif (for historical reasons). Keep it up to date */

dcl  cmep ptr;                                          /* pointer to core map entry */

dcl 1 cme based (cmep) aligned,                         /* core map entry */
    2 fp bit (18) unaligned,                            /* forward pointer to next entry */
    2 bp bit (18) unaligned,                            /* backward pointer to previous entry */

    2 devadd bit (22) unaligned,                        /* device address of page in the core block */
    2 pad5 bit (1) unaligned,
    2 synch_held bit (1) unaligned,                     /* Page of synchronized seg held in memory */
    2 io bit (1) unaligned,                             /* input/output indicator 1=output, 0=input */
    2 pad2 bit (1) unaligned,
    2 er bit (1) unaligned,                             /* indicates error in previous IO activity */
    2 removing bit (1) unaligned,                       /* core is being removed by reconfiguration */
    2 abs_w bit (1) unaligned,                          /* absolute address must not be changed for page */
    2 abs_usable bit (1) unaligned,                     /* page may be assigned with fixed absolute address */
    2 notify_requested bit (1) unaligned,               /* notify requested on I/O completion */
    2 pad3 bit (1) unaligned,
    2 phm_hedge bit (1) unaligned,                      /* on => pc$flush_core ought write. */
    2 contr bit (3) unaligned,                          /* controller in which core block is located */

    2 ptwp bit (18) unaligned,                          /* pointer to page table word for the page */
    2 astep bit (18) unaligned,                         /* relative AST entry pointer of page */
    2 pin_counter fixed bin (17) unaligned,             /* number of times to skip eviction */
    2 synch_page_entryp bit (18) unaligned;             /* relp to synch page entry */


dcl 1 cma (0: 1) based aligned like cme;                /* Core map array */

dcl 1 mcme based (cmep) aligned,                        /* core map entry for extracting DID */
    2 pad bit (36) unaligned,
    2 record_no bit (18) unaligned,                     /* record number of device */
    2 add_type bit (4) unaligned,                       /* see add_type.incl.pl1 */
    2 flags bit (14) unal,
    2 pad1 bit (18) unal;


/* END INCLUDE FILE  cmp.incl.pl1 */
```

```
/*     BEGIN INCLUDE FILE ... dir_acl.incl.pl1 ... last modified Nov 1975 for nss */

/* Template for an ACL entry. Length = 8 words */

dcl aclep ptr;

dcl 1 acl_entry based (aclep) aligned,                  /* length is 8 words */
    2 frp bit(18) unaligned,                            /* rel ptr to next entry */
    2 brp bit(18) unaligned,                            /* rel ptr to previous entry */

    2 type bit (18) unaligned,                          /* type = dir acl */
    2 size fixed bin (17) unaligned,                    /* size of acl entry */


    2 name unaligned,                                   /* user name associated with this ACL entry */
      3 pers_rp bit(18) unaligned,                      /* name of user */
      3 proj_rp bit(18) unaligned,                      /* project of user */
      3 tag char(1) unaligned,                          /* tag of user */
    2 mode bit (3) unaligned,                           /* mode for userid */
    2 pad24 bit(24) unaligned,

    2 ex_mode bit(36),                                  /* extended access modes */

    2 checksum bit (36),                                /* checksum from acl_entry.name */
    2  owner bit (36);                                  /* uid of owning entry */

/* Template for a person or project name on ACL. Length = 14 words. */

dcl 1 access_name aligned based,                        /* person or project name */
    2 frp bit(18) unaligned,                            /* rel ptr to next name structure */
    2 brp bit(18) unaligned,                            /* rel ptr to prev name structure */

    2 type bit (18) unaligned,                          /* type = access name */
    2 size fixed bin (17) unaligned,                    /* size of access name */

    2 salv_flag fixed bin(17) unaligned,                /* used by salvager to check for ascii names */
    2 usage fixed bin(17) unaligned,                    /* number of ACL entries that refer to this name */

    2 pad1 bit (36),

    2 name char(32) aligned,                            /* person or project name itself */

    2 checksum bit (36),                                /* checksum from salv_flag */

    2 owner bit (36);                                   /* uid of containing directory */

/*     END INCLUDE FILE ... dir_acl.incl.pl1 */
```

```
/*     BEGIN INCLUDE FILE ... dir_allocation_area.incl.pl1 ... last modified December 1973 */

dcl  areap ptr;

dcl 1 area based (areap) aligned,
    2 nsizes fixed bin (18),                        /* Number of types. */
    2 lu fixed bin (18),                            /* Next available word in area. */
    2 lw fixed bin (18),                            /* Last usable word. */
    2 array (100) aligned,                          /* Array of types. */
      3 fptr bit (18) unaligned,                    /* Free pointer for this size. */
      3 size fixed bin (17) unaligned;              /* Size. */

/*     END INCLUDE FILE ... dir_allocation_area.incl.pl1 */
```

```
/*        BEGIN INCLUDE FILE ... dir_entry.incl.pl1 ...last modified August 1974 for nss */

/* Template for an entry. Length = 38 words */

dcl  ep ptr;

dcl 1 entry based (ep) aligned,

    (2 efrp bit (18),                                   /* forward rel ptr to next entry */
    2 ebrp bit (18)) unaligned,                         /* backward rel ptr to previous entry */

    2 type bit (18) unaligned,                          /* type of object = dir entry  */
    2 size fixed bin (17) unaligned,                    /* size of dir entry */

    2 uid bit (36),                                     /* unique id of entry */

    2 dtem bit (36),                                    /* date-time entry modified */

    (2 bs bit (1),                                      /* branch switch = 1 if branch */
    2 pad0 bit (17),
    2 nnames fixed bin (17),                            /* number of names for this entry */

    2 name_frp bit (18),                                /* rel pointer to start of name list */
    2 name_brp bit (18),                                /* rel pointer to end of name list */

    2 author,                                           /* user who created branch */
      3 pers_rp bit (18),                               /* name of user who created branch */
      3 proj_rp bit (18),                               /* project of user who created branch */

      3 tag char (1),                                   /* tag of user who created branch */
      3 pad1 char (3),

    2 primary_name bit (504),                           /* first name on name list */

    2 dtd bit (36),                                     /* date time dumped */

    2 pad2 bit (36),


/* the declarations below are for branch only */


    2 pvid bit (36),                                    /* physical volume id */

    2 vtocx fixed bin (17),                             /* vtoc entry index */
    2 pad3 bit (18),

    2 dirsw bit (1),                                    /* = 1 if this is a directory branch */
    2 oosw bit (1),                                     /* out of service switch  on = 1 */
```

```
    2 per_process_sw bit (1),                    /* indicates segment is per process */
    2 copysw bit (1),                            /* = 1 make copy of segment whenever initiated */
    2 safety_sw bit (1),                         /* if 1 then entry cannot be deleted */
    2 multiple_class bit (1),                    /* segment has multiple security classes */
    2 audit_flag bit (1),                        /* segment must be audited for security */
    2 security_oosw bit (1),                     /* security out of service switch */
    2 entrypt_sw bit (1),                        /* 1 if call limiter is to be enabled */
    2 master_dir bit (1),                        /* TRUE for master directory */
    2 tpd bit (1),                               /* TRUE if this segment is never to go on the PD */
    2 pad4 bit (11),
    2 entrypt_bound bit (14)) unaligned,         /* call limiter */

    2 access_class bit (72) aligned,             /* security attributes : level and category */

    (2 ring_brackets (3) bit (3),                /* ring brackets on segment */
    2 ex_ring_brackets (3) bit (3),              /* extended ring brackets */
    2 acle_count fixed bin (17),                 /* number of entries on ACL */

    2 acl_frp bit (18),                          /* rel ptr to start of ACL */
    2 acl_brp bit (18),                          /* rel ptr to end of ACL */

    2 bc_author,                                 /* user who last set the bit count */
      3 pers_rp bit (18),                        /* name of user who set the bit count */
      3 proj_rp bit (18),                        /* project of user who set the bit count */

      3 tag char (1),                            /* tag of user who set the bit count */
      3 pad5 bit (2),
    2 bc fixed bin (24)) unaligned,              /* bit count for segs, msf indicator for dirs */

    2 sons_lvid bit (36),                        /* logical volume id for immediat inf non dir seg */

    2 pad6 bit (36),

    2 checksum bit (36),                         /* checksum from dtd */

    2 owner bit (36);                            /* uid of containing directory */

/*        END INCLUDE FILE ... dir_entry.incl.pl1 ... */
```

```
/*       BEGIN INCLUDE FILE ... dir_header.incl.pl1 */
/* Modified 8/74 for NSS */
/* Modified 8/76 to add version number and hash table rel pointer for variable hash table sizes */
/* Modified 3/82 BIM for change pclock */
/* format: style3 */

/* Template for the directory header. Length = 64 words. */

dcl  dp ptr;

dcl 1 dir based (dp) aligned,

   2 modify bit (36),                              /* Process ID of last modifier */
   2 type bit (18) unaligned,                   /* type of object = dir header */
   2 size fixed bin (17) unaligned,                /* size of header in words */
    2 dtc (3),                                     /* date-time checked by salvager array */
      3 date bit (36),                             /* the date */
      3 error bit (36),                            /* what errors were discovered */

    2 uid bit (36),                                /* uid of the directory          - copied from branch */

    2 pvid bit (36),                               /* phys vol id of the dir        - copied from branch */

    2 sons_lvid bit (36),                          /* log vol id for inf non dir seg - copied from branch */

    2 access_class bit (72),                       /* security attributes of dir    - copied from branch */

    (2 vtocx fixed bin (17),                       /* vtoc entry index of the dir   - copied from branch */
    2 version_number fixed bin (17),               /* version number of header */

    2 entryfrp bit (18),                           /* rel ptr to beginning of entry list */
    2 pad2 bit (18),

    2 entrybrp bit (18),                           /* rel ptr to end of entry list */
    2 pad3 bit (18),

    2 pers_frp bit (18),                           /* rel ptr to start of person name list */
    2 proj_frp bit (18),                           /* rel ptr to start of project name list */

    2 pers_brp bit (18),                           /* rel ptr to end of person name list */
    2 proj_brp bit (18),                           /* rel ptr to end of project name list */

    2 seg_count fixed bin (17),                    /* number of non-directory branches */
    2 dir_count fixed bin (17),                    /* number of directory branches */

    2 lcount fixed bin (17),                       /* number of links */
    2 acle_total fixed bin (17),                   /* total number of ACL entries in directory */

    2 arearp bit (18),                             /* relative pointer to beginning of allocation area */
    2 per_process_sw bit (1),                      /* indicates dir contains per process segments */
```

```
    2 master_dir bit (1),                       /* TRUE if this is a master dir */
    2 force_rpv bit (1),                         /* TRUE if segs must be on RPV */
    2 rehashing bit (1),                         /* TRUE if hash table is being constructed */
    2 pad4 bit (14),

    2 iacl_count (0:7),
      3 seg fixed bin (17),                      /* number of initial acl entries for segs */
      3 dir fixed bin (17),                      /* number of initial acl entries for dir */

    2 iacl (0:7),                                /* pointer to initial ACLs for each ring */
      3 seg_frp bit (18),                        /* rel ptr to start of initial ACL for segs */
      3 seg_brp bit (18),                        /* rel ptr to end of initial ACL for segs */

      3 dir_frp bit (18),                        /* rel ptr to start of initial for dirs */
      3 dir_brp bit (18),                        /* rel ptr to end of initial ACL for dirs */

    2 htsize fixed bin (17),                     /* size of hash table */
    2 hash_table_rp bit (18),                    /* rel ptr to start of hash table */

    2 htused fixed bin (17),                     /* no. of used places in hash table */
    2 pad6 fixed bin (17),

    2 tree_depth fixed bin (17),                 /* number of levels from root of this dir */
    2 pad7 bit (18)) unaligned,

    2 dts bit (36),                              /* date-time directory last salvaged */

    2 master_dir_uid bit (36),                   /* uid of superior master dir */
    2 change_pclock fixed bin (35),              /* up one each call to sum$dirmod */
    2 pad8 (11) bit (36),                        /* pad to make it a 64 word header */
    2 checksum bit (36),                         /* checksummed from uid on */
    2 owner bit (36);                            /* uid of parent dir */

dcl version_number_2 fixed bin int static options (constant) init (2);

/*        END INCLUDE FILE ... dir_header.incl.pl1 */
```

```
/*     BEGIN INCLUDE FILE ... dir_ht.incl.pl1 */


dcl htp ptr;

dcl 1 hash_table based (htp) aligned,                  /* htp = ptr(dp,active_hardcore_data$htrp) */
    2 modify bit (36) unal,
    2 type bit (18) unal,                              /* type = dir hash table */
    2 size fixed bin (17) unal,                        /* size of current dir hash table entry */
    2 name_rp (0:1) bit(18) unal,                      /* rel ptr of name entry  */
    2 checksum bit (36) unal,
    2 owner bit (36) unal;
                                                       /* otherwise rel ptr to name */

/*     END INCLUDE FILE ... dir_ht.incl.pl1 */
```

```
/*     BEGIN INCLUDE FILE ... dir_link.incl.pl1 ... last modified August 1974 for nss */

/* Template for link. Note that it is identical to entry for first 24 words. */


dcl 1 link based (ep) aligned,

    (2 efrp bit (18),                                   /* forward rel ptr to next entry */
    2 ebrp bit (18),                                    /* backward rel ptr to previous entry */

    2 type bit (18),                                    /* type = dir link */
    2 size fixed bin (17),                              /* size of link in words */

    2 uid bit (36),                                     /* unique id of entry */

    2 dtem bit (36),                                    /* date-time entry modified */

    2 bs bit (1),                                       /* entry switch = 1 if entry */
    2 pad0 bit (17),
    2 nnames fixed bin (17),                            /* number of names for this entry */

    2 name_frp bit (18),                                /* rel pointer to start of name list */
    2 name_brp bit (18),                                /* rel pointer to end of name list */

    2 author,                                           /* user who created entry */
      3 pers_rp bit (18),                               /* name of user who created entry */
      3 proj_rp bit (18),                               /* project of user who created entry */

      3 tag char (1),                                   /* tag of user who created entry */
      3 pad1 char (3),

    2 primary_name bit (504),                           /* first name on name list */

    2 dtd bit (36),                                     /* date time dumped */

    2 pad2 bit (36),


/* the declarations below are only applicable to links */

    2 pad3 bit (18),
    2 pathname_size fixed bin (17),                     /* number of characters in pathname */

    2 pathname char (168 refer (pathname_size))) unaligned, /* pathname of link */

    2 checksum bit (36),                                /* checksum from uid */

    2 owner bit (36);                                   /* uid of containing directory */

/*     END INCLUDE FILE ... dir_link.incl.pl1 */
```

```
/* Begin include file dir_lock_seg_.incl.pl1 BIM 830312 */
/* From dirlockt.incl.pl1 */

/* format: style3,idind25,indcomtxt */

/**** Several arrays in this program are zero based. The zero-th
      entries are NEVER USED. referencers should start at 1, not lbound.
      The zero entries are there to improve the compiler's subscript
      calculations. The compiler can fetch dir_lock_all_dir_locks (foo).uid
      with an lda pr6|FOO,*ql */

dcl       dir_lock_seg$            external static;         /* name of the segment containing the directory locks */

dcl       dir_lock_segp            pointer;                 /* pointer to the dirlock table */

dcl       1 dir_lock_seg           based (dir_lock_segp) aligned,
            2 header               aligned like dir_lock_seg_header,
            2 dir_locks            (0:dir_lock_seg.header.n_dir_locks) aligned like dir_lock,
            2 readers              (0:dir_lock_seg.header.n_dir_locks, dir_lock_seg.header.max_readers) bit (36) aligned;

declare   (dir_lock_all_locksp, dir_lock_all_readersp)
                                   pointer;

declare   1 dir_lock_all_dir_locks (0:dir_lock_seg.header.n_dir_locks) aligned like dir_lock based (dir_lock_all_locksp);
declare   dir_lock_all_readers     (0:dir_lock_seg.header.n_dir_locks, dir_lock_seg.header.max_readers) bit (36)
                                   aligned based (dir_lock_all_readersp);

declare   DIR_LOCK_SEG_EVENT       char (4) aligned init ("drls") int static options (constant);

declare   1 dir_lock_seg_header    aligned based,
            2 seg_lock             aligned like lock,
            2 n_dir_locks          fixed bin,               /* max number */
            2 highest_in_use       fixed bin,
            2 max_readers          fixed bin,
            2 pad_even             bit (36) aligned,
            2 readers_ptr          pointer,
            2 locks_ptr            pointer,
            2 meters               aligned,
              3 find_calls         fixed bin (35),
              3 find_failures      fixed bin (35),
              3 max_in_use         fixed bin (35),
              3 pad_meters         fixed bin (35),
            2 pad                  (16) bit (36) aligned;   /* to 32 */

declare   dir_lockp                pointer;
declare   1 dir_lock               aligned based (dir_lockp),
            2 uid                  bit (36) aligned,
            2 flags                aligned,
              3 notify_sw          bit (1) unaligned,
              3 salvage_sw         bit (1) unaligned,       /* ON if dir was locked for salvage */
```

```
          3 pad                bit (34) unaligned,
          2 lock_count          fixed bin (17),          /* POSITIVE --> write_lock */
                                                         /* NEGATIVE --> -number of lockers */
                                                         /* ZERO --> not locked */
          2 write_locker        bit (36) aligned;        /* in case of read, see next declaration, and expect this 0 */

declare   dir_read_lockers_ptr  pointer;
declare   dir_read_lockers      (dir_lock_seg.header.max_readers) bit (36) aligned based (dir_read_lockers_ptr);

/* End include file dir_lock_seg_.incl.pl1 */
```

```
/*     BEGIN INCLUDE FILE ... dir_name.incl.pl1 ... last modified Nov 1975 for nss */

/* Template for names of branches or links. Length = 14 words. */

dcl np ptr;

dcl 1 names based aligned,                              /* based on ptr(dp,ep->entry.name_frp) */
    2 fp bit(18) unaligned,                             /* rel ptr to next name */
    2 bp bit(18) unaligned,                             /* rel ptr to prev name */

    2 type bit (18) unaligned,                          /* type = dir name */
    2 size fixed bin (17) unaligned,                    /* size of dir name */

    2 entry_rp bit(18) unaligned,                       /* rel ptr to entry */
    2 ht_index fixed bin(17) unaligned,                 /* index of hash table entry */

    2 hash_thread bit (18) unal,                        /* relative ptr to next hash  entry */
    2 pad3 bit (18) unal,

    2 name char(32) aligned,

    2 checksum bit (36),                                /* checksum from entry_rp */

    2 owner bit (36);                                   /* uid of entry */


/*     END INCLUDE FILE ... dir_name.incl.pl1 */
```

```
/* BEGIN INCLUDE FILE ... fault_vector.incl.pl1 ... last modified February 1981 */

dcl  fvp ptr;                                   /* pointer to the fault and interrupt vectors */

dcl 1 fv based (fvp) aligned,                   /* fault and interrupt vectors */
    2 ipair (0: 31),                            /* interrupt pairs */
      3 scu bit (36),                           /* SCU instruction */
      3 tra bit (36),                           /* TRA instruction */
    2 fpair (0: 31),                            /* fault pairs */
      3 scu bit (36),                           /* SCU instruction */
      3 tra bit (36),                           /* TRA instruction */
    2 i_tra_ptr (0: 31) ptr,                    /* ITS pair for interrupt TRA instruction */
    2 i_scu_ptr (0: 31) ptr,                    /* ITS pair for interrupt SCU instruction */
    2 f_tra_ptr (0: 31) ptr,                    /* ITS pairs for fault TRA instruction */
    2 f_scu_ptr (0: 31) ptr;                    /* ITS pairs for fault SCU instruction */

/* Fault Types by fault number                                                          */

   dcl (FAULT_NO_SDF  init (0),                 /* Shutdown                            */
        FAULT_NO_STR  init (1),                 /* Store                               */
        FAULT_NO_MME  init (2),                 /* Master Mode Entry 1                 */
        FAULT_NO_F1   init (3),                 /* Fault Tag 1                         */
        FAULT_NO_TRO  init (4),                 /* Timer Runout                        */
        FAULT_NO_CMD  init (5),                 /* Command                             */
        FAULT_NO_DRL  init (6),                 /* Derail                              */
        FAULT_NO_LUF  init (7),                 /* Lockup                              */
        FAULT_NO_CON  init (8),                 /* Connect                             */
        FAULT_NO_PAR  init (9),                 /* Parity                              */
        FAULT_NO_IPR  init (10),                /* Illegal Procedure                   */
        FAULT_NO_ONC  init (11),                /* Operation Not Complete              */
        FAULT_NO_SUF  init (12),                /* Startup                             */
        FAULT_NO_OFL  init (13),                /* Overflow                            */
        FAULT_NO_DIV  init (14),                /* Divide Check                        */
        FAULT_NO_EXF  init (15),                /* Execute                             */
        FAULT_NO_DF0  init (16),                /* Directed Fault 0 (Segment Fault)    */
        FAULT_NO_DF1  init (17),                /* Directed Fault 1 (Page Fault)       */
        FAULT_NO_DF2  init (18),                /* Directed Fault 2                    */
        FAULT_NO_DF3  init (19),                /* Directed Fault 3                    */
        FAULT_NO_ACV  init (20),                /* Access Violation                    */
        FAULT_NO_MME2 init (21),                /* Master Mode Entry 2                 */
        FAULT_NO_MME3 init (22),                /* Master Mode Entry 3                 */
        FAULT_NO_MME4 init (23),                /* Master Mode Entry 4                 */
        FAULT_NO_F2   init (24),                /* Fault Tag 2 (Linkage Fault)         */
        FAULT_NO_F3   init (25),                /* Fault Tag 3                         */
                                                /* Fault Numbers 26-30 unassigned      */
        FAULT_NO_TRB  init (31)                 /* Trouble                             */

                        ) fixed bin (17) int static options (constant);
```

```
/* END INCLUDE FILE ... fault_vector.incl.pl1 */
```

```
/* BEGIN INCLUDE FILE ... fgbx.incl.pl1 */
/* last modified 5/3/77 by Noel I. Morris         */
/* Modified 8/79 by R.J.C. Kissel to add FNP blast message. */
/* Modified 7/82 BIM for recognizable sentinel field */

/* NOTE: THIS INCLUDE FILE DESCRIBES THE FLAGBOX WITHIN THE BOS TOEHOLD, WHICH
HAS NO VALUE WHATSOEVER.  THE REAL MULTICS/BCE TOEHOLD FLAGBOX IS DESCRIBED BY
THE FLAGBOX INCLUDE FILE. */

/* The contents of this segment are data shared by Multics and BOS.
   This segment occupies the 2nd, 3rd, 4th, and 5th 16-word blocks of the BOS toehold. */

dcl  flagbox$ ext;
dcl  fgbxp ptr;

dcl 1 fgbx based (fgbxp) aligned,
    2 flags (36) bit (1) unal,                          /* communications switches */
    2 slt_segno bit (18),                               /* segment # of the SLT */
    2 pad1 fixed bin,
    2 rtb,                                              /* return to BOS info */
     (3 ssenb bit (1),                                  /* "1"b if storage system enabled */
      3 call_bos bit (1),                               /* "1"b if BOS called by operator */
      3 shut bit (1),                                   /* "1"b if BOS called after shutdown */
      3 mess bit (1),                                   /* "1"b if message has been provided */
      3 alert bit (1),                                  /* "1"b if audible alarm to be sounded */
      3 pad bit (25),
      3 bos_entry fixed bin (5)) unal,                  /* type of entry into BOS
                                                           0 => XED 10002 (BOS entry)
                                                           1 => XED 10004 (Multics entry)
                                                           2 => XED 10000 (manual entry) */
    2 sentinel char (32) aligned,                       /* set by BOS (for now) */
    2 sst_sdw bit (72),                                 /* set by init_sst */
    2 hc_dbr bit (72),                                  /* set by start_cpu, idle DBR */
    2 message char (64),                                /* message for return to BOS */
    2 fnp_blast char (128);                             /* message for FNP use when Multics is down. */

declare FLAGBOX_SENTINEL char (32) init ("Flagbox & Toehold Valid") int static options (constant);

/* END INCLUDE FILE ... fgbx.incl.pl1 */
```

```
/* BEGIN INCLUDE FILE ... fs_vol_label.incl.pl1 .. last modified January 1982 for new volume map format  */
/* This is the label at fixed location of each physical volume. Length 1 page */
/* Note: fsout_vol clears pad fields before writing the label */

dcl  labelp ptr;

dcl 1 label based (labelp) aligned,

/* First comes data not used by Multics.. for compatibility with GCOS */

    2 gcos (5*64) fixed bin,

/* Now we have the Multics label */

    2 Multics char (32) init ("Multics Storage System Volume"), /* Identifier */
    2 version fixed bin,                                /* Version 1 */
    2 mfg_serial char (32),                             /* Manufacturer's serial number */
    2 pv_name char (32),                                /* Physical volume name. */
    2 lv_name char (32),                                /* Name of logical volume for pack */
    2 pvid bit (36),                                    /* Unique ID of this pack */
    2 lvid bit (36),                                    /* unique ID of its logical vol */
    2 root_pvid bit (36),                               /* unique ID of the pack containing the root. everybody must agree. */
    2 time_registered fixed bin (71),                   /* time imported to system */
    2 n_pv_in_lv fixed bin,                             /* # phys volumes in logical */
    2 vol_size fixed bin,                               /* total size of volume, in records */
    2 vtoc_size fixed bin,                              /* number of recs in fixed area + vtoc */
    2 not_used bit (1) unal,                            /* used to be multiple_class */
    2 private bit (1) unal,                             /* TRUE if was registered as private */
    2 flagpad bit (34) unal,
    2 max_access_class bit (72),                        /* Maximum access class for stuff on volume */
    2 min_access_class bit (72),                        /* Minimum access class for stuff on volume */
    2 password bit (72),                                /* not yet used */
    2 pad1 (16) fixed bin,
    2 time_mounted fixed bin (71),                      /* time mounted */
    2 time_map_updated fixed bin (71),                  /* time vmap known good */

/* The next two words overlay time_unmounted on pre-MR10 systems. This
   forces a salvage if an MR10 pack is mounted on an earlier system.
 */
    2 volmap_version fixed bin,                         /* version of volume map (currently 1) */
    2 pad6 fixed bin,

    2 time_salvaged fixed bin (71),                     /* time salvaged */
    2 time_of_boot fixed bin (71),                      /* time of last bootload */
    2 time_unmounted fixed bin (71),                    /* time unmounted cleanly */
    2 last_pvtx fixed bin,                              /* pvtx in that PDMAP */
    2 pad1a (2) fixed bin,
    2 err_hist_size fixed bin,                          /* size of pack error history */
    2 time_last_dmp (3) fixed bin (71),                 /* time last completed dump pass started */
    2 time_last_reloaded fixed bin (71),                /* what it says */
```

```
    2 pad2 (40) fixed bin,
    2 root,
      3 here bit (1),                          /* TRUE if the root is on this pack */
      3 root_vtocx fixed bin (35),             /* VTOC index of root, if it is here */
      3 shutdown_state fixed bin,              /* Status of hierarchy */
      3 pad7 bit (1) aligned,
      3 disk_table_vtocx fixed bin,            /* VTOC index of disk table on RPV */
      3 disk_table_uid bit (36) aligned,       /* UID of disk table */
      3 esd_state fixed bin,                   /* State of esd */
    2 volmap_record fixed bin,                 /* Begin record of volume map */
    2 size_of_volmap fixed bin,                /* Number of records in volume map */
    2 vtoc_map_record fixed bin,               /* Begin record of VTOC map */
    2 size_of_vtoc_map fixed bin,              /* Number of records in VTOC map */
    2 volmap_unit_size fixed bin,              /* Number of words per volume map section */
    2 vtoc_origin_record fixed bin,            /* Begin record of VTOC */
    2 dumper_bit_map_record fixed bin,         /* Begin record of dumper bit-map */
    2 vol_trouble_count fixed bin,             /* Count of inconsistencies found since salvage */
    2 pad3 (52) fixed bin,
    2 nparts fixed bin,                        /* Number of special partitions on pack */
    2 parts (47),
      3 part char (4),                         /* Name of partition */
      3 frec fixed bin,                        /* First record */
      3 nrec fixed bin,                        /* Number of records */
      3 pad5 fixed bin,
    2 pad4 (5*64) fixed bin;

dcl  Multics_ID_String char (32) init ("Multics Storage System Volume") static;

/* END INCLUDE FILE fs_vol_label.incl.pl1 */
```

```
/* BEGIN INCLUDE FILE its.incl.pl1
   modified 27 July 79 by JRDavis to add its_unsigned
   Internal format of ITS pointer, including ring-number field for follow-on processor */

dcl 1 its based aligned,                                /* declaration for ITS type pointer */
    2 pad1 bit (3) unaligned,
    2 segno bit (15) unaligned,                         /* segment number within the pointer */
    2 ringno bit (3) unaligned,                         /* ring number within the pointer */
    2 pad2 bit (9) unaligned,
    2 its_mod bit (6) unaligned,                        /* should be 43(8) */

    2 offset bit (18) unaligned,                        /* word offset within the addressed segment */
    2 pad3 bit (3) unaligned,
    2 bit_offset bit (6) unaligned,                     /* bit offset within the word */
    2 pad4 bit (3) unaligned,
    2 mod bit (6) unaligned;                            /* further modification */

dcl 1 itp based aligned,                                /* declaration for ITP type pointer */
    2 pr_no bit (3) unaligned,                          /* number of pointer register to use */
    2 pad1 bit (27) unaligned,
    2 itp_mod bit (6) unaligned,                        /* should be 41(8) */

    2 offset bit (18) unaligned,                        /* word offset from pointer register word offset */
    2 pad2 bit (3) unaligned,
    2 bit_offset bit (6) unaligned,                     /* bit offset relative to new word offset */
    2 pad3 bit (3) unaligned,
    2 mod bit (6) unaligned;                            /* further modification */


dcl 1 its_unsigned based aligned,                       /* just like its, but with unsigned binary */
    2 pad1 bit (3) unaligned,
    2 segno fixed bin (15) unsigned unaligned,
    2 ringno fixed bin (3) unsigned unaligned,
    2 pad2 bit (9) unaligned,
    2 its_mod bit (6) unaligned,

    2 offset fixed bin (18) unsigned unaligned,
    2 pad3 bit (3) unaligned,
    2 bit_offset fixed bin (6) unsigned unaligned,
    2 pad4 bit (3) unaligned,
    2 mod bit (6) unaligned;

dcl 1 itp_unsigned based aligned,                       /* just like itp, but with unsigned binary where appropriate */
    2 pr_no fixed bin (3) unsigned unaligned,
    2 pad1 bit (27) unaligned,
    2 itp_mod bit (6) unaligned,

    2 offset fixed bin (18) unsigned unaligned,
    2 pad2 bit (3) unaligned,
    2 bit_offset fixed bin (6) unsigned unaligned,
```

```
    2 pad3 bit (3) unaligned,
    2 mod bit (6) unaligned;


dcl  ITS_MODIFIER bit (6) unaligned internal static options (constant) init ("43"b3);
dcl  ITP_MODIFIER bit (6) unaligned internal static options (constant) init ("41"b3);

/* END INCLUDE FILE   its.incl.pl1 */
```

```
/*       BEGIN INCLUDE FILE ... itt_entry.incl.pl1 ... Feb 1981 */

/* format: style3 */
dcl      itte_ptr           ptr;                        /* pointer to entry in ITT */

dcl      1 itt_entry        aligned based (itte_ptr),   /* declaration of single entry in the ITT */
           2 next_itt_relp  bit (18) unaligned,         /* thread of relative pointers */
           2 pad            bit (18) unaligned,
           2 sender         bit (36),                   /* id of sending process */
           2 origin,                                    /* origin of event message */
             3 dev_signal   bit (18) unaligned,         /* 0 = user-event, 1 = device-signal */
             3 ring         fixed bin (17) unaligned,   /* if user-event, sender's validation ring */
           2 target_id      bit (36),                   /* target process' id */
           2 channel_id     fixed bin (71),             /* target process' event channel */
           2 message        fixed bin (71);             /* event message */

/*       END INCLUDE FILE ... itt_entry.incl.pl1 */
```

```
/*  BEGIN INCLUDE FILE  - - - kst.incl.pl1 - - -

Modified March 1976 by R. Bratt
Modified November 1984 to remove hdr, Keith Loepere. */

dcl  pds$kstp ext ptr,
    (kstp, kstep) ptr;

dcl 1 kst aligned based (kstp),                              /* KST header declaration */
    2 lowseg fixed bin (17),                                 /* lowest segment number described by kst */
    2 highseg fixed bin (17),                                /* highest segment number described by kst */
    2 highest_used_segno fixed bin (17),                     /* highest segment number yet used  */
    2 lvs fixed bin (8),                                     /* number of private LVs this process is connected to */
    2 time_of_bootload fixed bin (71),                       /* bootload time during prelinking */
    2 garbage_collections fixed bin (17) unaligned,          /* KST garbage collections */
    2 entries_collected fixed bin (17) unaligned,            /* KST entries recovered by garbage collection */
    2 free_list bit (18) unaligned,                          /* relative pointer to first free kste */
    2 prelinked_ring (7) bit (1) unaligned,                  /* rings prelinked in process */
    2 template bit (1) unaligned,                            /* this is a template kst if set */
    2 allow_256K_connect bit (1) unaligned,                  /* can use 256K segments */
    2 unused_2 bit (9) unaligned,
    2 uid_hash_bucket (0 : 127) bit (18) unaligned,          /* hash buckets */
    2 kst_entry (0 refer (kst.lowseg):0 refer (kst.highseg)) aligned like kste, /* kst entries */
    2 lv (1:256) bit (36),                                   /* private logical volume connection list */
    2 end_of_kst bit (36);

dcl 1 kste based (kstep) aligned,                            /* KST entry declaration */
    2 fp bit (18) unaligned,                                 /* forward rel pointer */
    2 segno fixed bin (17) unaligned,                        /* segment number of this kste */
    2 usage_count (0:7) fixed bin (8) unaligned,             /* outstanding initiates/ring */
    2 entryp ptr unaligned,                                  /* branch pointer */
    2 uid bit (36) aligned,                                  /* unique identifier */
    2 access_information unaligned,
      3 dtbm bit (36),                                       /* date time branch modified */
      3 extended_access bit (33),                            /* extended access from the branch */
      3 access bit (3),                                      /* rew */
      3 ex_rb (3) bit (3),                                   /* ring brackets from branch */
    2 pad1 bit (3) unaligned,
    2 flags unaligned,
      3 dirsw bit (1),                                       /* directory switch */
      3 allow_write bit (1),                                 /* set if initiated with write permission */
      3 priv_init bit (1),                                   /* privileged initiation */
      3 tms bit (1),                                         /* transparent modification switch */
      3 tus bit (1),                                         /* transparent usage switch */
      3 tpd bit (1),                                         /* transparent paging device switch */
      3 audit bit (1),                                       /* audit switch */
      3 explicit_deact_ok bit (1),                           /* set if I am willing to have a user force deactivate */
      3 pad bit (3),
    2 infcount fixed bin (12) unaligned;                     /* if dirsw then inferior count else lv index */
```

```
/*  END INCLUDE FILE - - - - - - - - - - - - - kst.incl.pl1 - - - - - - - - - - - - - */
```

```
/* BEGIN INCLUDE FILE -- lot.incl.pl1   S.Webber 9/74, Modified by R. Bratt 04/76, modified by M. Weaver 7/76 */
/* modified by M. Weaver 3/77 */

dcl  lotp ptr;

dcl 1 lot based (lotp) aligned,
    2 lp (0:9999) ptr unaligned;                          /* array of packed pointers to linkage sections */

dcl lot_fault bit (36) aligned static options (constant) init ("111000000000000000000000000000000000"b);
                                                          /* lot fault has fault code = 0 and offset = 0 */

dcl  isotp ptr;
dcl 1 isot based (isotp) aligned,
    2 isp (0:9999) ptr unaligned;

dcl 1 isot1 (0 :9999) aligned based,
    2 flags unaligned,
      3 fault bit (2) unaligned,
      3 system bit (1) unaligned,
      3 mbz bit (6) unaligned,
    2 fault_code fixed bin (8) unaligned,
    2 static_offset bit (18) unaligned;


/* END INCLUDE FILE  lot.incl.pl1 */
```

```
/*
```

```
 */
/* BEGIN INCLUDE FILE  mc.incl.pl1   Created Dec 72 for 6180 - WSS. */
/* Modified 06/07/76 by Greenberg for mc.resignal */
/* Modified 07/07/76 by Morris for fault register data */
/* Modified 08/28/80 by J. A. Bush for the DPS8/70M CVPU */
/* Modified '82 to make values constant */

/* words 0-15 pointer registers */

dcl  mcp ptr;

dcl 1 mc based (mcp) aligned,
    2 prs (0:7) ptr,                                    /* POINTER  REGISTERS */
   (2 regs,                                             /* registers */
      3 x (0:7) bit (18),                               /* index registers */
      3 a bit (36),                                     /* accumulator */
      3 q bit (36),                                     /* q-register */
      3 e bit (8),                                      /* exponent */
      3 pad1 bit (28),
      3 t bit (27),                                     /* timer register */
      3 pad2 bit (6),
      3 ralr bit (3),                                   /* ring alarm register */

    2 scu (0:7) bit (36),

    2 mask bit (72),                                    /* mem controller mask at time of fault */
    2 ips_temp bit (36),                                /* Temporary storage for IPS info */
    2 errcode fixed bin (35),                           /* fault handler's error code */
    2 fim_temp,
      3 unique_index bit (18) unal,                     /* unique index for restarting faults */
      3 resignal bit (1) unal,                          /* recompute signal name with fcode below */
      3 fcode bit (17) unal,                            /* fault code used as index to FIM table and SCT */
    2 fault_reg bit (36),                               /* fault register */
    2 pad2 bit (1),
    2 cpu_type fixed bin (2) unsigned,                  /* L68 = 0, DPS8/70M = 1 */
    2 ext_fault_reg bit (15),                           /* extended fault reg for DPS8/70M CPU */
    2 fault_time bit (54),                              /* time of fault */

    2 eis_info (0:7) bit (36)) unaligned;


dcl (apx fixed bin init (0),
     abx fixed bin init (1),
     bpx fixed bin init (2),
     bbx fixed bin init (3),
     lpx fixed bin init (4),
     lbx fixed bin init (5),
     spx fixed bin init (6),
     sbx fixed bin init (7)) internal static options (constant);
```

```
dcl  scup ptr;

dcl 1 scu based (scup) aligned,                    /* SCU DATA */


/*        WORD (0)            */

   (2 ppr,                                         /* PROCEDURE POINTER REGISTER */
      3 prr bit (3),                               /* procedure ring register */
      3 psr bit (15),                              /* procedure segment register */
      3 p bit (1),                                 /* procedure privileged bit */

    2 apu,                                         /* APPENDING UNIT STATUS */
      3 xsf bit (1),                               /* ext seg flag - IT modification */
      3 sdwm bit (1),                              /* match in SDW Ass. Mem. */
      3 sd_on bit (1),                             /* SDW Ass. Mem.  ON */
      3 ptwm bit (1),                              /* match in PTW Ass. Mem. */
      3 pt_on bit (1),                             /* PTW Ass. Mem.  ON */
      3 pi_ap bit (1),                             /* Instr Fetch or Append cycle */
      3 dsptw bit (1),                             /* Fetch of DSPTW */
      3 sdwnp bit (1),                             /* Fetch of SDW non paged */
      3 sdwp bit (1),                              /* Fetch of SDW paged */
      3 ptw bit (1),                               /* Fetch of PTW */
      3 ptw2 bit (1),                              /* Fetch of pre-paged PTW */
      3 fap bit (1),                               /* Fetch of final address paged */
      3 fanp bit (1),                              /* Fetch of final address non-paged */
      3 fabs bit (1),                              /* Fetch of final address absolute */

    2 fault_cntr bit (3),                          /* number of retrys of EIS instructions */


/*        WORD (1)            */

    2 fd,                                          /* FAULT  DATA */
      3 iro bit (1),                               /* illegal ring order */
      3 oeb bit (1),                               /* out of execute bracket */
      3 e_off bit (1),                             /* no execute */
      3 orb bit (1),                               /* out of read bracket */
      3 r_off bit (1),                             /* no read */
      3 owb bit (1),                               /* out of write bracket */
      3 w_off bit (1),                             /* no write */
      3 no_ga bit (1),                             /* not a gate */
      3 ocb bit (1),                               /* out of call bracket */
      3 ocall bit (1),                             /* outward call */
      3 boc bit (1),                               /* bad outward call */
      3 inret bit (1),                             /* inward return */
      3 crt bit (1),                               /* cross ring transfer */
      3 ralr bit (1),                              /* ring alarm register */
      3 am_er bit (1),                             /* associative memory fault */
      3 oosb bit (1),                              /* out of segment bounds */
      3 paru bit (1),                              /* processor parity upper */
      3 parl bit (1),                              /* processor parity lower */
      3 onc_1 bit (1),                             /* op not complete type 1 */
      3 onc_2 bit (1),                             /* op not complete type 2 */
```

```
    2 port_stat,                                /* PORT  STATUS */
      3 ial bit (4),                            /* illegal action lines */
      3 iac bit (3),                            /* illegal action channel */
      3 con_chan bit (3),                       /* connect channel */

    2 fi_num bit (5),                           /* (fault/interrupt)  number */
    2 fi_flag bit (1),                          /*  1 => fault,  0 => interrupt  */


/*        WORD (2)          */

    2 tpr,                                      /* TEMPORARY POINTER REGISTER */
      3 trr bit (3),                            /* temporary ring register */
      3 tsr bit (15),                           /* temporary segment register */

    2 pad2 bit (9),

    2 cpu_no bit (3),                           /* CPU number */

    2 delta bit (6),                            /* tally modification DELTA */


/*        WORD (3)          */

    2 word3 bit (18),

    2 tsr_stat,                                 /* TSR STATUS for 1,2,&3 word instructions */
      3 tsna,                                   /* Word 1 status */
        4 prn bit (3),                          /* Word 1  PR  number */
        4 prv bit (1),                          /* Word 1  PR  valid bit */
      3 tsnb,                                   /* Word 2  status */
        4 prn bit (3),                          /* Word 2  PR  number */
        4 prv bit (1),                          /* Word 2  PR  valid bit */
      3 tsnc,                                   /* Word 3  status */
        4 prn bit (3),                          /* Word 3  PR  number */
        4 prv bit (1),                          /* Word 3  PR  valid bit */

    2 tpr_tbr bit (6),                          /* TPR.TBR  field */


/*        WORD (4)          */

    2 ilc bit (18),                             /* INSTRUCTION  COUNTER */

    2 ir,                                       /* INDICATOR  REGISTERS */
      3 zero bit (1),                           /* zero indicator */
      3 neg bit (1),                            /* negative indicator */
      3 carry bit (1),                          /* carryry indicator */
      3 ovfl bit (1),                           /* overflow indicator */
      3 eovf bit (1),                           /* eponent overflow */
      3 eufl bit (1),                           /* exponent underflow */
      3 oflm bit (1),                           /* overflow mask */
      3 tro bit (1),                            /* tally runout */
      3 par bit (1),                            /* parity error */
      3 parm bit (1),                           /* parity mask */
```

```
      3 bm bit (1),                         /* ^bar mode */
      3 tru bit (1),                        /* truncation mode */
      3 mif bit (1),                        /* multi-word instruction mode */
      3 abs bit (1),                        /* absolute mode */
      3 hex bit (1),                        /* hexadecimal exponent mode */
      3 pad bit (3),

/*        WORD (5)            */

    2 ca bit (18),                          /* COMPUTED  ADDRESS */

    2 cu,                                   /* CONTROL UNIT STATUS */
      3 rf bit (1),                         /* on first cycle of repeat instr */
      3 rpt bit (1),                        /* repeat instruction */
      3 rd bit (1),                         /* repeat double instruction */
      3 rl bit (1),                         /* repeat link instruciton */
      3 pot bit (1),                        /* IT  modification */
      3 pon bit (1),                        /* return type instruction */
      3 xde bit (1),                        /* XDE from Even location */
      3 xdo bit (1),                        /* XDE from Odd  location */
      3 poa bit (1),                        /* operation preparation */
      3 rfi bit (1),                        /* tells CPU to refetch instruction */
      3 its bit (1),                        /* ITS modification */
      3 if bit (1),                         /* fault occured during instruction fetch */

    2 cpu_tag bit (6)) unaligned,           /* computed tag field */

/*        WORDS (6,7)         */

    2 even_inst bit (36),                   /* even instruction of faulting pair */

    2 odd_inst bit (36);                    /* odd  instruction of faulting pair */



/*        ALTERNATE   SCU  DECLARATION          */

dcl 1 scux based (scup) aligned,

   (2 pad0 bit (36),

    2 fd,                                   /* GROUP II  FAULT DATA */
      3 isn bit (1),                        /* illegal segment number */
      3 ioc bit (1),                        /* illegal op  code  */
      3 ia_am bit (1),                      /* illegal address - modifier */
      3 isp bit (1),                        /* illegal slave  procedure */
      3 ipr bit (1),                        /* illegal procedure */
      3 nea bit (1),                        /* non existent address */
      3 oobb bit (1),                       /* out of bounds */
      3 pad bit (29),
```

```
    2 pad2 bit (36),

    2 pad3a bit (18),

    2 tsr_stat (0:2),                          /* TSR  STATUS  as an  ARRAY */
      3 prn bit (3),                           /* PR  number */
      3 prv bit (1),                           /* PR  valid bit */

    2 pad3b bit (6)) unaligned,

    2 pad45 (0:1) bit (36),

    2 instr (0:1) bit (36);                    /* Instruction ARRAY */


/*  END INCLUDE FILE   mc.incl.pl1 */
```

```
/* ************************************************************
   *                                                          *
   * Copyright, (C) Honeywell Information Systems Inc., 1982 *
   *                                                          *
   * Copyright (c) 1972 by Massachusetts Institute of         *
   * Technology and Honeywell Information Systems, Inc.       *
   *                                                          *
   ************************************************************ */


/* PDS - The Process Data Segment

   Last modified (Date and reason):
   2/6/76 by S. Webber Initial coding
   9/17/76 by R. Bratt to add seg_fault, bounds_fault, vtoc_read, and vtoc_write meters.
   11/03/76 by M. Weaver to extend stack header
   04/20/77 by M. Weaver to delete rntp and 7/77 to add name template_pds
   06/07/78 by E. Donner to add ring_events (to prevent delayed ipc wakeups)
   05/10/79 by B. Margulies to eliminate exmode_level
   05/09/79 by Mike Grady to use shared ring 0 stacks
   08/17/79 by J. A. Bush for exp under/overflow restart switches & cache parity diagnostics
   02/28/80 by B. Margulies to use the include file for the default overflow
   08/26/80 by J. A. Bush  for the DPS8/70M CPU
   value.
   02/23/81 by J. Bongiovanni to remove temp_mode_reg (moved to prds$mode_reg_enabled)
   03/81 by E. Donner to remove next_itt and ect_pointers
    3/82 BIM for lock_array cleanup.
   11/82 by J. Bongiovanni to make force_write_limit per-ring
    2/83 by E. N. Kittlitz for hfp_exponent_enabled.
    830621 BIM for level improvements.
   10/83 by E. N. Kittlitz to resurrect obsolescent network_ptbl_idx for MR10.2.
    83-11-02 by E. N. Kittlitz for block_lock_count in low page, hex exponent control.
    83-11-21 BIM to inhibit quota and save history registers by default
             in the initializer's process.
    83-12-01 E. N. Kittlitz for restart hex overflow fault control
    83-12-03 BIM to clear trace header properly. (and new trace format)
    84-12-10 Keith Loepere for throttle_segment_state_changes and other
             covert channel related variables.
  1985-01-21, BIM: admin_privileges to record ring 1 priv settings.
  1985-04-08, BIM: no_audit_ring1_fs_object_ops to suppress auditing
                   while in the mseg primitives and RCP.
*/

/* format: style3,idind25 */
pds:
     procedure;

/* This program creates the pds data base */

/* Automatic */
```

```
dcl       1 cdsa                    aligned like cds_args;
dcl       code                      fixed bin (35);

/* Constants */

dcl       pdsname                   char (3) aligned static init ("pds") options (constant);
dcl       exclude_pad               (1) char (32) aligned static options (constant) init ("pad*");

/* Builtins */

dcl       (addr, bin, bit, decimal, divide, float, hbound, mod, null, rel, size, string, unspec)
                                    builtin;

/* Entries */

dcl       com_err_                  entry options (variable);
dcl       create_data_segment_      entry (ptr, fixed bin (35));
dcl       get_temp_segment_         entry (char (*), ptr, fixed bin (35));
dcl       release_temp_segment_     entry (char (*), ptr, fixed bin (35));
dcl       hcs_$chname_file          entry (char (*), char (*), char (*), char (*), fixed bin (35));
dcl       get_wdir_                 entry () returns (char (168));

/* External Static */

dcl       error_table_$segnamedup   fixed bin (35) ext;
```

```
dcl     pdsp                    ptr;

dcl     1 pds                   aligned based (pdsp),
          2 page_fault_data     like mc,                /* MC for page faults and timer runouts */
          2 fim_data            like mc,                /* MC for normal faults */
          2 signal_data         aligned like mc,        /* storage for MC being signalled */
          2 history_reg_data    (64) fixed bin (71),    /* this must follow signal data */
          2 process_group_id    char (32),              /* user id for current process */
          2 cpu_time            fixed bin (52),         /* number that when subtracted from clock reading gives
                                                           virtual cpu time */
          2 virtual_delta       fixed bin (52),         /* temporary used in calculating VCPU time */
          2 virtual_time_at_eligibility
                                fixed bin (52),         /* temporary used in calculation of VCPU time */
          2 temp_1              fixed bin (71),         /* temporary */
          2 temp_2              fixed bin (71),         /* temporary */
          2 time_1              fixed bin (52),         /* page fault metering time */
          2 time_v_temp         fixed bin (52),         /* temporary used in calculating VCPU time */
          2 fim_v_temp          fixed bin (52),         /* VCPU temporary for the FIM */
          2 fim_v_delta         fixed bin (71),         /* VCPU temporary for the FIM */
          2 save_history_regs   bit (1) aligned,        /* = "1"b if history registers are to be saved */
          2 hregs_saved         bit (1) aligned,        /* = "1"b if history regs were saved */
          2 last_sp             ptr,                    /* stack pointer at getwork time */
          2 apt_ptr             ptr,                    /* pointer to this process's APT entry */
          2 arg_1               fixed bin (71),         /* argument for pxss */
          2 arg_2               fixed bin (71),         /* argument for pxss */
          2 arg_3               fixed bin (71),         /* argument for pxss */
          2 arg_4               fixed bin (71),         /* argument for pxss */
          2 access_authorization aligned like aim_template,
                                                        /* access authorization for the process */
          2 base_addr_reg       bit (18) aligned,       /* for BAR mode use */
          2 alarm_ring          fixed bin (3),          /* setting for ring alarm register */
          2 pxss_args_invalid   bit (36) aligned,       /* used by pxss masking/arg copying code */
          2 processid           bit (0) unaligned,      /* process ID (added segdef) */
          2 process_id          bit (36) aligned,       /* process ID */
          2 vtime_count         fixed bin,              /* depth counter used in VCPU calculation */
          2 pstep               bit (0) unaligned,      /* (added segdef for dstep) */
          2 dstep               bit (18) aligned,       /* rel pointer to ASTE for dseg */
          2 wakeup_flag         bit (36) aligned,       /* flag indicating type of wakeup */
          2 pc_call             bit (36) aligned,       /* flag saying type of wait */
          2 audit_flags         bit (36) aligned,       /* bits indicating types of auditing to do */
          2 quota_inhib         fixed bin aligned,      /* ON if quota checking to be inhibited */
          2 covert_event_count  fixed bin,              /* count of covert channel related segment state change events */
          2 page_waits          fixed bin,              /* page faults */
          2 number_of_pages_in_use
                                fixed bin,              /* used in calculating memory units */
          2 post_purged         fixed bin,              /* number of post purgings */
          2 connect_pending     bit (1) aligned,        /* turned on for delayed connects to be resent by fim */
          2 segment_faults      fixed bin (35),         /* count of segment faults taken by this process */
          2 bounds_faults       fixed bin (35),         /* count of bounds faults taken by this process */
          2 vtoc_reads          fixed bin (35),         /* vtoc read I/Os done for this process */
          2 vtoc_writes         fixed bin (35),         /* vtoc write I/Os done for this process */
          2 mc_trace_seg        fixed bin,              /* seg number of object segment being traced */
          2 mc_trace_sw         bit (2) aligned,        /* switch for M. C. Tracing "11"b => trace on */
          2 stack_0_sdwp        ptr aligned,            /* ptr to stack sdw in dseg */
```

```
2 stack_0_ptr          ptr aligned,            /* ptr to base of ring 0 stack (wired for esd) */
 2 tc_argp              ptr,                    /* arg ptr used by tc */
 2 tc_mask              bit (72) aligned,       /* save tc mask */
 2 exp_undfl_rest       bit (2) aligned,        /* fim restarts underflow: '1'b = binary, '01'b = hex */
 2 exp_ovfl_rest        bit (2) aligned,        /* fim restarts exp overflow: '1'b = binary, '01'b = hex */
 2 eovfl_value          bit (72) aligned,       /* value DFLD'ed by fim on restart binary overflow */
 2 hex_eovfl_value bit (72) aligned,            /* value DFLD'ed by fim on restart hex overflow */
 2 cpar_err_data        bit (72) aligned,       /* cache parity error data (from cache) */
 2 cpar_mem_data        bit (72) aligned,       /* cache parity error data (from memory) */
 2 cpar_info            bit (36) aligned,       /* diagnose flag, cache level and absaddr # */
 2 hfp_exponent_enabled bit (1) aligned,        /* user allowed to set IR hex exp bit */
 2 pre_empt_poll_return pointer,
 2 block_lock_count     fixed bin,              /* count of locks held */
 2 throttle_segment_state_changes bit (1) aligned,/* limit bandwidth of segment state covert channels */
 2 first_covert_event_time fixed bin (52),
 2 pad_for_trace_mod16  (6) fixed bin,
 2 trace                (306) fixed bin (71),   /* system trace data */
                                                /* pds$trace + 16 defines the pds for idle procs */
 2 timer_time_out       fixed bin (52),         /* time out time for the process */
 2 timer_channel        fixed bin (71),         /* event channel for time out event */
 2 term_channel         fixed bin (71),         /* channel used to signal process termination */
 2 term_proc            bit (36) aligned,       /* process ID of process to signal term process */
 2 pl1_machine          fixed bin,              /* nonzero if we do pl1-like things */
 2 validation_level     fixed bin (3),
 2 condition_name       aligned,                /* ACC string for condition name */
   3 len                fixed bin (8) unaligned,
   3 chars              char (31) unaligned,
 2 pad_obsolete         bit (36) aligned,
 2 ips_mask             (0:7) bit (35) aligned, /* IPS masks */
 2 auto_mask            (0:7) bit (36) aligned, /* array of automatic masks for IPS signals */
 2 ring_alarm_val       (0:7) fixed bin,        /* used in checking validation level changes */
 2 lock_id              bit (36) aligned,       /* UID used in some locking */
 2 mc_trace_buf         ptr unaligned,          /* packed ptr to mc_trace wired buffer */
 2 pad_end_of_page_0    bit (0) unaligned,
 2 pathname_am          aligned like pam,       /* pathname associative memory */
 2 initial_procedure    ptr,                    /* first procedure executed in a new process */
 2 account_id           char (32) aligned,      /* not used yet */
 2 access_name          aligned,                /* alternate form of process group id */
   3 user               char (32) aligned,
   3 project            char (32) aligned,
   3 tag                char (32) aligned,
 2 home_dir             char (168) aligned,     /* home directory */
 2 process_dir_name     char (32) aligned,      /* name of process directory */
 2 wdir                 (0:7) ptr,              /* pointers to per-ring working directories */
 2 wdir_uid             (0:7) bit (36) aligned, /* UID of per-ring working directories */
 2 transparent          bit (36) aligned,       /* transparent usage, mod, pd switch */
 2 itt_head             bit (18) aligned,       /* top of present ITT list */
 2 max_access_authorization
                        aligned like aim_template,
                                                /* max authorization this user can attain */
 2 stacks               (0:7) ptr,              /* per-ring stack pointers */
 2 kstp                 ptr,                    /* pointer to start of KST */
 2 events_pending       bit (36) aligned,       /* special wakeups pending */
 2 special_channels     bit (36) aligned,       /* special channels assigned */
 2 event_masks          (7) bit (36) aligned,   /* per-ring mask for special channels */
 2 initial_ring         fixed bin (3),          /* initial ring of execution for the process */
```

```
          2 interrupt_ring       fixed bin (3),          /* lowest ring in which IPS interrupts are allowed */
           2 highest_ring         fixed bin (3),          /* highest ring in which process can run */
           2 prelinked_ring       bit (8) aligned,        /* bit(i) is ON if ring (i) is prelinked */
           2 unique_scu_index     bit (36) aligned,       /* used to tag MC */
           2 max_lot_size         (0:7) fixed bin,        /* sizes lots can grow to */
           2 lot_stack_size       (0:7) fixed bin,        /* size of lot in stack (0 -> lot not in stack) */
           2 clr_stack_size       (0:7) fixed bin,        /* size of CLR in stack */
           2 link_meters_bins     (4) fixed bin,          /* histograms of linkage faults */
           2 link_meters_times    (4) fixed bin (30),     /* histogram of linkage fault times */
           2 link_meters_pgwaits  (4) fixed bin,          /* histogram of linkage faults PF's */
           2 dmpr_copy_dirsegp    ptr,                    /* ptr to temp segment into which dirs are copied */
           2 dmpr_pvid            bit (36),               /* pvid of volume being dumped */
           2 dmpr_pvtx            fixed bin,              /* pvtx of volume being dumped  */
           2 first_call           fixed bin,              /* ON until leave ring zero once */
           2 mc_save_area         bit (18) aligned,       /* rel pointer to start of saved MC area */
           2 mc_save_ptr          bit (18) aligned,       /* ptr to next mc save place */
           2 mc_save_limit        bit (18) aligned,       /* max address where MC can be saved */
           2 useable_lot          bit (8) aligned,        /* indicates whether lot can be referenced */
           2 ring_events          bit (36) aligned,       /* per-ring indicator that itt messages copied to ect */
           2 force_write_limit    (0:7) fixed bin,        /* limit on force-writing */
                                                          /* Following must be doubleword aligned! */
           2 ipc_vars             aligned,                /* holds state of fast_hc_ipc at block */
             3 ap                 pointer unal,
             3 retsw              fixed bin (35),
             3 save_entry_ret     fixed bin (35),
             3 truncated_stacks   fixed bin (35),
             3 chan               fixed bin (71),
             3 block_start_steps  fixed bin (35),
             3 stk_temp           fixed bin (35),
           2 ipc_block_return     bit (36),               /* ipc block return address */
           2 avg_block_steps      fixed bin (35, 18),
           2 admin_privileges     bit (36) aligned,       /* There is a 1 here for each privilege that must be reset on exit from ring 1 */
           2 no_audit_ring1_fs_object_ops bit (1) aligned, /* Ring 1 has asked to turn off ring 0 auditing */
           2 pad_for_data_mod16   (6) fixed bin (35),
           2 data                 bit (0) aligned;        /* to mark end of PDS for MC save area */
%page;
        call get_temp_segment_ ("pds", pdsp, code);      /* Returns ZEROS */

/* Now begins the initialization */

        pds.process_group_id = "Initializer.SysDaemon.z";

        pds.access_authorization.categories = (18)"0"b;
        pds.access_authorization.level = 0;
        pds.access_authorization.dir = "1"b;             /* for initializer */
        pds.access_authorization.seg = "1"b;
        pds.access_authorization.rcp = "1"b;
        pds.access_authorization.ipc = "1"b;
        pds.access_authorization.soos = "1"b;            /* .. */

        pds.max_access_authorization.categories = (18)"1"b || (18)"0"b;
        pds.max_access_authorization.level = 7;

        pds.quota_inhib = 1;                              /* initializer ignore rqover until it is enabled */
        pds.vtime_count = -1;
```

```
pds.process_id = (36)"1"b;
 pds.lock_id = (36)"1"b;
 pds.pl1_machine = 1;
 pds.ips_mask (*) = (35)"1"b;
 pds.force_write_limit (*) = 1;

 pds.save_history_regs = "1"b;
 pds.hregs_saved = "0"b;
 pds.history_reg_data (*) = 0;

 pds.mc_trace_buf = null;
 pds.mc_trace_sw = "0"b;
 pds.mc_trace_seg = 0;

 pds.eovfl_value = unspec (Default_exponent_control_overflow_value);
 pds.hex_eovfl_value = unspec (Default_hex_exponent_control_overflow_value);
                                                  /* set default exp overflow restart value */
 pds.exp_ovfl_rest, pds.exp_undfl_rest = "0"b;

 pds.stack_0_sdwp = null;
 pds.stack_0_ptr = null;
 pds.pad_for_trace_mod16 (*) = 0;

 unspec (pds.trace) = ""b;
 trace_ptr = addr (pds.trace);
 trace.last_available = divide (hbound (pds.trace, 1) * size (page_trace_entry) - 8, 2, 17, 0);
 trace.threshold = .75 * float (decimal (trace.last_available));

 pds.initial_procedure = null;

 pds.access_name.user = "Initializer";
 pds.access_name.project = "SysDaemon";
 pds.access_name.tag = "z";

 pds.home_dir = ">system_control_1";
 pds.process_dir_name = ">process_dir_dir>!zzzzzzzbBBBBBB";

 pds.wdir (*) = null;
 pds.wdir_uid (*) = "0"b;



 pds.stacks (*) = null;

 pds.dmpr_pvid = "0"b;
 pds.dmpr_pvtx = 0;
 pds.dmpr_copy_dirsegp = null;

 pds.kstp = null;
 pds.first_call = 1;
 pds.initial_ring = 1;
 pds.interrupt_ring = 4;
 pds.highest_ring = 7;

 pds.max_lot_size (*) = 1024;
```

```
          pds.mc_save_area = rel (addr (pds.data));
           pds.mc_save_ptr = rel (addr (pds.data));
           pds.mc_save_limit = bit (bin (4096, 18), 18);     /* Allow for as many as fit in 4K. */

/* Now make some checks on alignment of certain variables */

           call check (addr (pds.ipc_vars), "ipc_vars", 2);
           call check (addr (pds.page_fault_data), "page_fault_data", 16);
           call check (addr (pds.trace), "trace", 16);
           call check (addr (pds.signal_data), "signal_data", 16);
           call check (addr (pds.eovfl_value), "eovfl_value", 2);
           call check (addr (pds.hex_eovfl_value), "hex_eovfl_value", 2);
           call check (addr (pds.data), "data", 16);
           if bin (rel (addr (pds.pad_end_of_page_0)), 18) ^= 1024
           then call com_err_ (0, pdsname, "Wired portion must end at 1024");

/* Now set up call to create data base */

           cdsa.sections (1).p = addr (pds);
           cdsa.sections (1).len = size (pds);
           cdsa.sections (1).struct_name = "pds";

           cdsa.seg_name = "pds";
           cdsa.num_exclude_names = 1;
           cdsa.exclude_array_ptr = addr (exclude_pad);

           string (cdsa.switches) = "0"b;
           cdsa.switches.have_text = "1"b;

           call create_data_segment_ (addr (cdsa), code);

           call release_temp_segment_ ("pds", pdsp, code);

           call hcs_$chname_file (get_wdir_ (), "pds", "", "template_pds", code);
           if code ^= 0
           then if code ^= error_table_$segnamedup
                then call com_err_ (code, pdsname, "Unable to add name template_pds.");
```

```
check:
     proc (where, message, modulo);

dcl       where                    ptr;
dcl       message                  char (*);
dcl       modulo                   fixed bin;
dcl       remainder                fixed bin;

          remainder = mod (bin (rel (where), 18), modulo);
          if remainder ^= 0
          then call com_err_ (0, pdsname, "The variable ^a is ^d words away from being aligned on a ^d-word boundary.",
                    message, (modulo - remainder), modulo);

     end check;
%page; %include aim_template;
%page; %include cds_args;
%page; %include exponent_control_info;
%page; %include mc;
%page; %include pathname_am;
%page; %include sys_trace;
     end pds;
```

```
/* ************************************************************
   *                                                          *
   * Copyright, (C) Honeywell Information Systems Inc., 1982 *
   *                                                          *
   ************************************************************ */
/* PRDS - The Processor Data Segment and Processor Stack.
   /* Last modified (Date and reason):
   2/6/76 by S. Webber Initial coding
   6/15/77 by M. Weaver to  null signal and sct pointers
   8/25/80 by J. A. Bush for the dps8/70m cpu
   2/22/81 by J. Bongiovanni for fast_connect_code
   6/27/81 by J. Bongiovanni for idle_temp
   10/11/83 by R. Coppola to adjust for size change of fast connect code
                  and validate that apt_ptr& ignore_pl are on correct mod
*/


/* ********************************************************
   *                                                      *
   *                                                      *
   * Copyright (c) 1972 by Massachusetts Institute of     *
   * Technology and Honeywell Information Systems, Inc.   *
   *                                                      *
   *                                                      *
   ******************************************************** */


prds: proc;

/* This program creates the prds data base */

/* Automatic */

dcl  i fixed bin;
dcl 1 cdsa aligned like cds_args;
dcl  code fixed bin (35);

/* Static */

dcl  prdsname char (4) aligned static init ("prds") options (constant);
dcl  exclude_pad (1) char (32) aligned static options (constant) init ("pad*");

/* The following must correspond to the size of the fast connect code in
   fast_connect_init                                                    */

dcl  FAST_CONNECT_CODE_WORDS init (72) fixed bin int static options (constant);

/* Builtins */

dcl (addr, baseptr, bin, mod, null, ptr, rel, size, string, unspec) builtin;
```

```
/* Entries */

dcl  com_err_ entry options (variable);
dcl  create_data_segment_ entry (ptr, fixed bin (35));
dcl  get_temp_segment_ entry (char (*), ptr, fixed bin (35));
dcl  release_temp_segment_ entry (char (*), ptr, fixed bin (35));
```

```
dcl  prdsp ptr;

dcl 1 prds aligned based (prdsp),
    2 header aligned like stack_header,                  /* standard stack header */
    2 interrupt_data aligned like mc,                    /* MC for interrupts */
    2 fim_data aligned like mc,                          /* MC for connect faults, timer runouts */
    2 sys_trouble_data aligned like mc,                  /* MC for saved sys trouble data */
    2 ignore_data aligned like scu,                      /* for SCU data to be ignored at certain times */
    2 iitemp fixed bin (71),                             /* temporary used by ii (surprise!) */
    2 last_recorded_time fixed bin (71),                 /* used by traffic control */
    2 idle_ptr ptr,                                      /* pointer to idle process APTE for this processor */
    2 simulated_mask fixed bin (71),                     /* simulated system controller mask register */
    2 am_data bit (0),                                   /* to get addr of associative memory data block */
    2 ptw_am_regs (4*16) fixed bin (35),                 /* page table regs (4 sets of 16 for dps8/70m) */
    2 ptw_am_ptrs (4*16) fixed bin (35),                 /* page table pointers (4 sets of 16 for dps8/70m) */
    2 sdw_am_regs (4*16) fixed bin (71),                 /* segment desc. regs (4 sets of 16 for dps8/70m) */
    2 sdw_am_ptrs (4*16) fixed bin (35),                 /* segment desc. pointers (4 sets of 16 for dps8/70m) */
    2 processor_pattern bit (8) aligned,                 /* 1 bit ON for this processor */
    2 processor_tag fixed bin (3),                       /* CPU tag from maintenance panel */
    2 last_timer_setting bit (27) aligned,               /* last timer value loaded for this CPU */
    2 depth fixed bin,                                   /* depth in eligible queue for running process */
    2 mode_reg bit (36) aligned,                         /* mode register for this processor */
    2 cache_luf_reg bit (36) aligned,                    /* cache mode register for this CPU */
    2 fault_reg bit (72) aligned,                        /* place to store the fault register */
    2 apt_ptr ptr,                                       /* -> apte running on this cpu                  */
    2 idle_temp fixed bin (71),                          /* used by idle process                         */


/*  The following contains code used for handling connect faults for this processor                       */


    2 fast_connect_code (FAST_CONNECT_CODE_WORDS) bit (36) aligned,
    2 fast_connect_code_end bit (36) aligned,            /* marker for fast_connect_init                 */
    2 mode_reg_enabled bit (36) aligned,                 /* used to set mode register                    */
    2 pad_mod_8 (2) fixed bin,
    2 ignore_pl (8) bit (36) aligned,                    /* used by wired fim to spl/lpl */
    2 pad_mod_16 (16) bit (36) aligned,
    2 processor_stack aligned like stack_frame;          /* first stack frame location */
```

```
          call get_temp_segment_ ("prds", prdsp, code);

          unspec (prds) = ""b;


/* Now make some checks on alignment of certain variables */

          call check (addr (prds.idle_ptr), "idle_ptr", 2);
          call check (addr (prds.processor_stack), "processor_stack", 16);
          call check (addr (prds.ptw_am_regs), "ptw_am_regs", 16);
          call check (addr (prds.sdw_am_regs), "sdw_am_regs", 32);
          call check (addr (prds.fast_connect_code), "fast_connect_code", 2);
          call check (addr (prds.ignore_pl), "ignore_pl",8);
          call check (addr (prds.apt_ptr), "apt_ptr", 2);

/* Now set up call to create data base */

          cdsa.sections (1).p = addr (prds);
          cdsa.sections (1).len = size (prds);
          cdsa.sections (1).struct_name = "prds";

          cdsa.seg_name = "prds";
          cdsa.num_exclude_names = 1;
          cdsa.exclude_array_ptr = addr (exclude_pad);

          string (cdsa.switches) = "0"b;
          cdsa.switches.have_text = "1"b;

          call create_data_segment_ (addr (cdsa), code);

          call release_temp_segment_ ("prds", prdsp, code);
```

```
check:    proc (where, message, modulo);

dcl  where ptr;
dcl  message char (*);
dcl  modulo fixed bin;

          if mod (bin (rel (where), 18), modulo) ^= 0
          then call com_err_ (0, prdsname, "The variable ^a is not aligned on a ^d-word boundary.", message, modulo);

        end check;
```

```
% include cds_args;
```

```
% include stack_header;
```

```
% include stack_frame;

% include mc;

     end prds;
```

```
/* BEGIN INCLUDE FILE ... ptw.168.incl.pl1 ... 02/26/81, for ADP conversion */
/* Note: This include file has an ALM counterpart made with cif. Keep it up to date */

dcl 1 168_core_ptw aligned based (ptp),                    /* In-core page descriptor */
    2 frame fixed bin (14) unsigned unaligned,             /* Core frame number */
    2 pad1 bit (4) unaligned,
    2 flags unaligned like 168_ptw_flags;

dcl 1 168_ptw aligned based (ptp),                         /* General declaration for out-of-core PTW */
    2 add bit (18) unaligned,
    2 flags like 168_ptw_flags unaligned;

dcl 1 168_special_ptw aligned based (ptp) like 168_ptw;    /* Page is somewhere peculiar -- add_type = "01"b */
dcl 1 168_real_disk_ptw aligned based (ptp) like 168_ptw;  /* PTW for page actually on disk -- add_type = "10"b */
dcl 1 168_null_disk_ptw aligned based (ptp) like 168_ptw;  /* PTW for page not yet on disk -- add_type = "11"b */

dcl 1 168_ptw_flags unaligned based,                       /* Various software/hardware flags */
   (2 add_type bit (4),                                    /* 0000=null, 1000=core, 0100=disk, 0010=pd, 0001=swap */
    2 first bit (1),                                       /* the page has not yet been written out */
    2 er bit (1),                                          /* error on last page I/O (also used by post-purge as temp) */

    2 pad1 bit (1),
    2 unusable1 bit (1),                                   /* can't be used because hardware resets this bit */
    2 phu bit (1),                                         /* page has been used bit */

    2 phm1 bit (1),                                        /* Cumulative OR of hardware phm's */
    2 nypd bit (1),                                        /* must be moved to paging device */
    2 phm bit (1),                                         /* page has been modified bit */

    2 phu1 bit (1),                                        /* page has been used in the quantum */
    2 wired bit (1),                                       /* page is to remain in core */
    2 os bit (1),                                          /* page is out-of-service (I/O in progress) */
    2 valid bit (1),                                       /* directed fault if this is 0 (page not in core) */
    2 df_no bit (2)) unaligned;                            /* directed fault number for page faults */

/* END INCLUDE FILE ... ptw.168.incl.pl1 */
```

```
/*        BEGIN INCLUDE FILE ... pv_holdt.incl.pl1 ... */

dcl pv_holdtp ptr;

dcl 1 pv_holdt (1 : 64) based (pv_holdtp) aligned,

    2 pvtx   fixed bin(17) unaligned,
    2 apterp bit(18) unaligned;


/*        END INCLUDE FILE ... pv_holdt.incl.pl1 ...*/
```

```
/*       BEGIN INCLUDE FILE ... pvt.incl.pl1 ... last modified January 1982 */


/* The physical volume table (PVT) is a wired-down table.
   It has one entry for each spindle present, be it for
   Storage System or "I/O" use.
*/

     dcl     pvt$                    ext,
             pvtp                    ptr;


     dcl     1 pvt                   based (pvtp) aligned,

               2 n_entries           fixed bin (17),         /* number of PVT entries */
               2 max_n_entries       fixed bin (17),         /* max number of PVT entries */
               2 n_in_use            fixed bin (17),         /* number of PVT entries in use */
               2 rwun_pvtx           fixed bin,              /* rewind_unloading pvtx */
               2 shutdown_state      fixed bin,              /* state of previous shutdown */
               2 esd_state           fixed bin,              /* state of ESD, >0 iff in ESD */
               2 prev_shutdown_state fixed bin,              /* shutdown state of previous bootload */
               2 prev_esd_state      fixed bin,              /* ESD state of previous bootload */

               2 time_of_bootload fixed bin (71),            /* Time of bootload */
               2 root_lvid           bit (36) aligned,       /* Logical volume ID of Root Logical Volume (RLV) */
               2 root_pvid           bit (36) aligned,       /* Physical volume ID of Root Physical Volume (RPV) */
               2 root_pvtx           fixed bin,              /* Index to PVTE for Root Physical Volume (RPV) */
               2 root_vtocx          fixed bin,              /* VTOCE index for root (>) */
               2 disk_table_vtocx    fixed bin,              /* VTOCE index for disk table on RPV */
               2 disk_table_uid      bit (36) aligned,       /* File System UID for disk_table */

               2 rpvs_requested      bit (1) aligned,        /* RPVS keyword given on BOOT */
               2 rpv_needs_salv      bit (1) aligned,        /* RPV required (not requested) salvage */
               2 rlv_needs_salv      bit (1) aligned,        /* RLV required (not requested) salvage */
               2 volmap_lock_wait_constant bit (36) aligned,/* For constructing wait event: OR pvte_rel into lower */
               2 volmap_idle_wait_constant bit (36) aligned,/* For constructing wait event: OR pvte_rel into lower */
               2 vtoc_map_lock_wait_constant bit (36) aligned, /* For constructing wait event: OR pvte_rel into lower */
               2 n_volmap_locks_held fixed bin (17),         /* Current number of volmap locks held */
               2 n_vtoc_map_locks_held fixed bin (17),       /* Current number of VTOC Map locks held */

               2 last_volmap_time    fixed bin (71),         /* Time a volmap was last locked/unlocked */
               2 last_vtoc_map_time fixed bin (71),          /* Time a VTOC Map was last locked/unlocked */
               2 total_volmap_lock_time fixed bin (71),      /* Total time volmap's were locked (integral) */
               2 total_vtoc_map_lock_time fixed bin (71),    /* Total time VTOC Maps were locked (integral) */

               2 n_volmap_locks      fixed bin (35),         /* Number times a volmap was locked */
               2 n_vtoc_map_locks    fixed bin (35),         /* Number times a vtoc_map was locked */
               2 volmap_lock_nowait_calls fixed bin (35),    /* Number calls to lock volmap, no wait */
               2 volmap_lock_nowait_fails fixed bin (35),    /* Number times lock failed */
               2 volmap_lock_wait_calls fixed bin (35),      /* Number calls to lock volmap, wait */
```

```
          2 volmap_lock_wait_fails fixed bin (35),      /* Number times lock failed */
           2 pad (2) bit (36) aligned,

           2 array              fixed bin (71);         /* Array of PVTE's -- must be double-word aligned */


/*        END INCLUDE FILE ...pvt.incl.pl1 */
```

```
/*  START OF:       pvte.incl.pl1       July 1982   * * * * * * * * * * * * * * * * * * * */
/* Added pc_vacating, Benson Margulies 84-10-17 */

     dcl     pvt$array              aligned external;
     dcl     pvt$max_n_entries      fixed bin external;


     dcl     pvt_arrayp             ptr;
     dcl     pvtep                  ptr;

     dcl     1 pvt_array            (pvt$max_n_entries) aligned like pvte based (pvt_arrayp);




     dcl     1 pvte                 based (pvtep) aligned,

               2 pvid               bit (36),              /* physical volume ID */

               2 lvid               bit (36),              /* logical volume ID */

               2 dmpr_in_use        (3) bit (1) unaligned, /* physical volume dumper interlock */
               2 pad3               bit (6) unaligned,
               2 skip_queue_count   fixed bin (18) unsigned unaligned, /* number of times this pv skipped for per-proc allocation due to saturation */
               2 brother_pvtx       fixed bin (8) unaligned,/* next pvte in lv chain */


               2 devname            char (4),              /* device name */

               (2 device_type       fixed bin (8),         /* device type */
               2 logical_area_number fixed bin (8),        /* disk drive number */
               2 used               bit (1),               /* TRUE if this entry is used */
               2 storage_system     bit (1),               /* TRUE for storage system (vs io disk) */
               2 permanent          bit (1),               /* TRUE if cannot be demounted */
               2 testing            bit (1),               /* Protocol bit for read_disk$test */
               2 being_mounted      bit (1),               /* TRUE if the physical volume is being mounted */
               2 being_demounted    bit (1),               /* TRUE if the pysical volume is being demounted */
               2 check_read_incomplete bit (1),            /* page control should check read incomplete */
               2 device_inoperative bit (1),               /* TRUE if disk_control decides dev busted */
               2 rpv                bit (1),               /* TRUE if this is the root physical volume */
               2 scav_check_address
                                    bit (1),               /* TRUE is page control should check deposits/withdrawals against scavenger table */
               2 deposit_to_volmap  bit (1),               /* TRUE if deposits should got to volume map, not stock */
               2 being_demounted2   bit (1),               /* No more vtoc I/O during demount */
               2 pc_vacating        bit (1),               /* No more withdraws from this volume -- for debugging */
               2 vacating           bit (1),               /* don't put new segs on this vol */
               2 hc_part_used       bit (1),               /* HC part set up by init_pvt */
```

```
          2 volmap_lock_notify bit (1) unal,          /* TRUE if notify required when volmap lock is unlocked */
           2 volmap_idle_notify bit (1) unal,          /* TRUE if notify required when volmap state is idle */
           2 vtoc_map_lock_notify bit (1) unal,        /* TRUE if notify required when vtoc map lock is unlocked */


           2 n_free_vtoce       fixed bin (17),        /* number of free VTOC entries */
           2 vtoc_size          fixed bin (17),        /* size of the VTOC part of the disk - in records */

           2 dbmrp              (2) bit (18),          /* rel ptr to dumber bit maps for this volume */

           2 nleft              fixed bin (17),        /* number of records left */
           2 totrec             fixed bin (17)) unaligned, /* Total records in this map */

           2 dim_info           bit (36),              /* Information peculiar to DIM */

           2 curn_dmpr_vtocx    (3) fixed bin unaligned,/* current vtocx being dumped */
           2 n_vtoce            fixed bin unaligned,   /* number of vtoce on this volume */

           2 baseadd            fixed bin (18) uns unaligned, /* Base of paging region */
           2 pad2               bit (18) unaligned,

           2 volmap_seg_sdw     fixed bin (71),        /* SDW describing volmap_seg */


           2 volmap_astep       ptr unal,              /* Packed pointer to ASTE for volmap_seg */

           2 volmap_offset      bit (18) unal,         /* Offset in volmap_seg of volume map */
           2 vtoc_map_offset    bit (18) unal,         /* Offset in volmap_seg of VTOC map */


           2 volmap_lock        bit (36) aligned,      /* Lock on volume map operations */

           2 vtoc_map_lock      bit (36) aligned,      /* Lock on VTOC map operations */

           2 volmap_stock_ptr   ptr unal,              /* Packed pointer to record stock */

           2 vtoc_map_stock_ptr ptr unal,              /* Packed pointer to VTOCE stock */

           2 volmap_async_state fixed bin (17) unaligned, /* Asynchronous update state of Volume Map */
           2 volmap_async_page  fixed bin (17) unaligned, /* Page number for asynchronous update */

           2 vol_trouble_count fixed bin (17) unaligned,  /* Count of inconsistencies since last salvage */
           2 scavenger_block_rel bit (18) unaligned;     /* Offset to scavenger block,  ^0 => scavenging */


     dcl     (VOLMAP_ASYNC_IDLE     init (0),           /* for volmap_async_state */
              VOLMAP_ASYNC_READ     init (1),
              VOLMAP_ASYNC_WRITE    init (2)) fixed bin int static options (constant);

/*  END OF:       pvte.incl.pl1                       * * * * * * * * * * * * * * * * * * * */
```

```
/* ************************************************************
   *                                                          *
   * Copyright, (C) Honeywell Information Systems Inc., 1984 *
   *                                                          *
   * Copyright (c) 1972 by Massachusetts Institute of         *
   * Technology and Honeywell Information Systems, Inc.       *
   *                                                          *
   ************************************************************ */


/* SCS - The System Communications Segment
   modified 3/27/77 by Noel I. Morris
   last modified 4/26/78 by J. A. Bush for processor testing
   Modified 2/79 by BSG for 8-cpu port expander
   Modified 9/16/80 by J. A. Bush for the DPS8/70M CPU
   Modified 1/09/81 W. Olin Sibert to remove all initializations to scs_and_clock_init
   Modified 01/16/81 W. Olin Sibert, to add scs$port_data
   Modified Jabuary 1981 by C. Hornig for new I/O.
   Modified 2/22/81 by J. Bongiovanni for fast connect code
   Modified 4/23/81 by J. Bongiovanni for cycle_priority_template
   Modified 4/09/82 by J. Bongiovanni for switch 0, processor_data_switch_value
   Modified 7/30/82 by J. Bongiovanni for trouble_processid
   Modified 4/11/83 by E. N. Kittlitz for drl_message_pointer.
   Modified 10/25/83 by Keith Loepere for start_of_scs
*/

scs:
     procedure;

/* Static */

dcl exclude_pad (1) char (32) static options (constant) init ("pad*");

/* Automatic */

dcl code fixed bin (35);
dcl 1 cdsa aligned like cds_args;
dcl i fixed bin;

/* Builtins */

dcl (addr, bin, bit, null, size, string, unspec) builtin;

/* Entries */

dcl create_data_segment_ entry (ptr, fixed bin (35));
%page;
dcl 1 scs aligned,                                      /* Information about system controllers */
      2 start_of_scs fixed bin (71),
      2 controller_data (0:7) aligned like scs$controller_data,
```

```
                                                    /* per-controller info */
                                                     /* Information about CPUs */
2 processor_data (0:7) aligned like scs$processor_data,
                                                    /* information about CPUs in the system */
2 port_data (0:7) like scs$port_data aligned,        /* Info on what is connected to each SCU port */
2 cow (0:7) like scs$cow,                           /* Actual COW's. */
2 cow_ptrs (0:7) aligned like scs$cow_ptrs,         /* Rel pointers to COW's. */
2 reconfig_general_cow aligned like scs$reconfig_general_cow,
                                                    /* Used for reconfiguration operations */
                                                    /* MASKS and PATTERNS */
2 sys_level aligned bit (72),                       /* mask used while handling I/O interrupts */
2 open_level aligned bit (72),                      /* mask used during normal operation */
2 processor_start_mask aligned bit (72),            /* mask used when starting up a CPU */
2 cpu_test_mask aligned bit (72),                   /* mask used for ISOLTS CPU testing */
2 number_of_masks fixed bin,                        /* number of masks (starting at sys_level) */
2 processor_start_pattern bit (36) aligned,         /* SMIC pattern used to send processor start interrupt */
2 cpu_test_pattern bit (36) aligned,                /* SMIC pattern used for ISOLTS processor testing */
2 expanded_ports bit (1) unaligned dim (0:7),       /* Which ports have expanders */
                                                    /* CAM and CACHE clear info */
2 cam_pair fixed bin (71),                          /* instructions XEDd when CAMing and clearing CACHE */
2 cam_wait bit (8) aligned,                         /* Used when evicting pages from main memory */
2 pad1 fixed bin,                                   /* MASKING INSTRUCTIONS & POINTERS */
2 set_mask (0:7) bit (36) aligned,                  /* instructions to set mask (STAQ or SMCM) */
2 read_mask (0:7) bit (36) aligned,                 /* instructions to read mask (LDAQ or RMCM) */
2 mask_ptr (0:7) ptr unaligned,                     /* pointers for real or simulated masks */
                                                    /* MISCELLANEOUS */
2 idle_aptep (0:7) ptr unaligned,                   /* pointer to idle process APTE for each processor */
2 connect_lock bit (36) aligned,                    /* lock for sending connects */
2 reconfig_lock bit (36) aligned,                   /* lock used during reconfiguration */
2 trouble_flags bit (8) aligned,                    /* checkoff flags for sys_trouble stopping */
2 bos_restart_flags bit (8) aligned,                /* checkoff flags for restarting after sys_trouble */
2 nprocessors fixed bin,                            /* number of processors online */
2 bos_processor_tag fixed bin (3),                  /* CPU tag of processor running BOS */
2 faults_initialized bit (1) aligned,               /* ON after faults have been enabled */
2 sys_trouble_pending bit (1) aligned,              /* sys_trouble event is pending in the system */
2 fast_cam_pending (0:7) bit (36) aligned,          /* checkoff flags for cam connect */
2 interrupt_controller fixed bin (3),               /* port number of low order controller */
2 cycle_priority_template bit (7) aligned,
2 set_cycle_switches bit (1) aligned,
2 processor_start_int_no fixed bin (5),             /* interrupt cell for starting a processor */
2 processor bit (8) aligned,                        /* bits ON for online CPUs */
2 processor_start_wait bit (8) aligned,             /* checkoff flags for waiting for new processor */
2 trouble_processid bit (36) aligned,               /* processid causing crash */
2 drl_message_pointer ptr unal,                     /* pointer to DRL message text */
2 processor_test_data aligned like scs$processor_test_data,
                                                    /* info for cpu testing */
2 pad2 fixed bin,
2 trouble_dbrs (0:7) fixed bin (71),                /* DBR values at system crash time */
2 port_addressing_word (0:7) bit (3) aligned,       /* active module port number for each controller */
2 cfg_data (0:7) fixed bin (71),                    /* RSCR-CFG data from each controller */
2 cfg_data_save fixed bin (71),                     /* RSCR-CFG save area for ISOLTS CPU testing */
2 processor_switch_data (0:4) bit (36) aligned,     /* actual processor RSW data */
2 processor_switch_template (0:4) bit (36) aligned, /* expected data from RSW 0 thru 4 */
2 processor_switch_compare (0:4) bit (36) aligned,  /* discrepancies from expected data */
2 processor_switch_mask (0:4) bit (36) aligned,     /* masks for comparing switch data */
2 processor_data_switch_value bit (36) aligned,     /* Correct value of CPU data switches */
```

```
                                                        /* Data used by init_sst and collect_free_core, from config cards. */
     2 controller_config_size (0:7) fixed bin (14) aligned,/* config card-stated size of controller */
     2 reconfig_locker_id char (32) aligned,             /* process group ID of process doing reconfiguration */
     2 scas_page_table (0:31) bit (36) aligned,          /* Page table for SCAS */
     2 end_of_scs fixed bin;                            /* For initialization */
%page;
         unspec (scs) = "0"b;                           /* clear entire structure */

/* Now set up for call to create_data_segment_ */

         cdsa.sections (1).p = addr (scs);
         cdsa.sections (1).len = size (scs);
         cdsa.sections (1).struct_name = "scs";

         cdsa.seg_name = "scs";
         cdsa.num_exclude_names = 1;
         cdsa.exclude_array_ptr = addr (exclude_pad);

         string (cdsa.switches) = "0"b;
         cdsa.switches.have_text = "1"b;

         call create_data_segment_ (addr (cdsa), code);
         return;
%page;
%include scs;
%include cds_args;

     end scs;
```

```
/* BEGIN INCLUDE FILE ... sdw.168.incl.pl1 ... Updated for ADP conversion 03/01/81 */
/* Note: This include file has an ALM counterpart made with cif. Keep it up to date */

dcl 1 168_sdw based (sdwp) aligned,                    /* Level 68 Segment Descriptor Word */

   (2 add bit (24),                                    /* main memory address of page table */
    2 rings,                                           /* ring brackets for the segment */
      3 r1 bit (3),
      3 r2 bit (3),
      3 r3 bit (3),
    2 valid bit (1),                                   /* directed fault bit (0 => fault) */
    2 df_no bit (2),                                   /* directed fault number */

    2 pad1 bit (1),
    2 bound bit (14),                                  /* boundary field (in 16 word blocks) */
    2 access,                                          /* access bits */
      3 read bit (1),                                  /* read permission bit */
      3 execute bit (1),                               /* execute permission bit */
      3 write bit (1),                                 /* write permission bit */
      3 privileged bit (1),                            /* privileged bit */
    2 unpaged bit (1),                                 /* segment is unpaged if this is 1 */
    2 not_a_gate bit (1),                              /* if this is 0 the entry bound is checked by hardware */
    2 cache bit (1),                                   /* cache enable bit */
    2 entry_bound bit (14)) unaligned;                 /* entry bound */

/* END INCLUDE FILE ... sdw.168.incl.pl1 */
```

```
/*   BEGIN INCLUDE FILE ... sst.incl.pl1 ... January 1971 */
/* Note: This include file has an ALM counterpart made with cif. Keep it up to date */
/* Deleted paging device info and added pc segmove info, Benson Margulies 84-01-03 */
/* Added covert channel meters, Keith Loepere 85-01-08. */

dcl  sst_seg$ external;
dcl  sstp ptr;

dcl 1 sst based (sstp) aligned,
    2 space (8) fixed bin,                          /* empty space to watch for bugs */

/* SST HEADER */

    2 pre_page_time fixed bin (71),                 /* total time spent pre-paging */
    2 post_purge_time fixed bin (71),               /* total time spent post-purging */
    2 post_in_core fixed bin,                       /* total pages in core (and in list) at purge time */
    2 thrashing fixed bin,                          /* meter of thrashing being done on system */
    2 npfs_misses fixed bin,                        /* meter of times npfs was on when pre-paging */
    2 salv fixed bin,                               /* flag which is ^=0 if and only if salvaging */

    2 ptl bit (36),                                 /* global page table loop lock */
    2 astl bit (36),                                /* global ast allocation block lock */
    2 astl_event bit (36),                          /* event used when waiting for AST lock */
    2 astl_notify_requested bit (1) aligned,        /* flag to notify AST lock */
    2 nused fixed bin,                              /* number of pages on used list */
    2 ptwbase fixed bin (24),                       /* absolute address of page table array */
    2 tfreep ptr,                                   /* pointer to first trailer on free list */

    2 astap ptr,                                    /* aste array pointer */
    2 ptl_wait_ct fixed bin,                        /* pxss: number is >= # of processes waiting to ptl */
    2 astsize fixed bin,                            /* size of an AST entry */
    2 cmesize fixed bin,                            /* size of a CME entry */
    2 root_astep ptr,                               /* pointer to the root AST entry */

    2 pts (0: 3) fixed bin,                         /* array of page table sizes */
    2 level (0:3),                                  /* per-list information about ASTE's */
      3 (ausedp, no_aste) bit (18) unaligned,       /* used list and count of number of entries */

    2 (atempp, atempp1) bit (18) unal,              /* temp seg list pointer */
    2 dm_enabled bit (1) aligned,                   /* ON => journal seg exists */
    2 (ainitp, ainitp1) bit (18) unal,              /* init seg list pointer */
    2 strsize fixed bin,                            /* Trailer size in words. */

/* CORE MAP HEADER */

    2 cmp ptr,                                      /* pointer to start of core map */
    2 usedp bit (18),                               /* pointer to first used core block */
    2 wtct fixed bin,                               /* count of pages being written */

    2 startp bit (18),                              /* pointer to solid page for lap counting (fsdct) */
```

```
    2 removep bit (18),                         /* pointer to list of pages being removed from use */
                                                /* MISC */

    2 double_write fixed bin,                   /* trigger for store through scheme */
                                                /* 0 = no double writes,
                                                   1 = all non-pd pages get written,
                                                   2 = all directories get written */
    2 temp_w_event bit (36) aligned,            /* wait event for temp wiring lock */
    2 root_pvtx fixed bin,                      /* pvtx or rpv */
    2 nolock bit (1) aligned,                   /* if on, don't lock ptl on interrupts */

    2 fc_skips_pinned fixed bin (35),           /* number of skips over pinned page in find_core */
    2 cl_skips_pinned fixed bin (35),           /* number of skips over pinned page in claim_mod_core */
    2 ast_ht_ptr ptr,                           /* AST hast table pointer */
    2 ast_ht_n_buckets fixed bin,               /* number of buckets in AST hash table */
    2 ast_ht_uid_mask bit (36) aligned,         /* mask to strip out low-order bits of uid */
    2 meter_ast_locking fixed bin,              /* non-zero enables AST lock meters */
    2 checksum_filemap fixed bin,               /* non-zero enables filemap checksumming */

    2 page_read_errors fixed bin,               /* read errors posted to page control */
    2 page_write_errors fixed bin,              /* write errors posted to page control */

    2 cycle_pv_allocation fixed bin,            /* flag to cycle VTOCE allocation among PVs */

    2 n_trailers fixed bin,                     /* Number of trailer entries in str_seg */
    2 synch_activations fixed bin (35),         /* Activation attempts for synchronized segs */
    2 synch_skips fixed bin (35),               /* get_aste skips because not synchronized */

    2 lock_waits fixed bin,                     /* Number of times we had to wait for a lock */
    2 total_locks_set fixed bin,                /* Total number of block locks set */
    2 pdir_page_faults fixed bin,               /* total page faults off >pdd */
    2 level_1_page_faults fixed bin,            /* total page faults in sys libes */
    2 dir_page_faults fixed bin,                /* Total page faults on directories */
    2 ring_0_page_faults fixed bin,             /* page faults in ring 0 */
    2 rqover fixed bin (35),                    /* errcode for record quota overflow */
    2 pc_io_waits fixed bin,                    /* Number of times pc had to wait on io */

/*  The following (until pdmap) used to be the 'cnt' in cnt.incl.pl1 */

    2 steps fixed bin,                          /* number of steps taken around used list */
    2 needc fixed bin,                          /* number of times core page needed */
    2 ceiling fixed bin,                        /* number of times ceiling hit */
    2 ctwait fixed bin,                         /* number of times write counter was full */
    2 wired fixed bin,                          /* number of pages wired by pc */
    2 laps fixed bin,                           /* number of times around used list */
    2 skipw fixed bin,                          /* number of pages skiped because they were wired */
    2 skipu fixed bin,                          /* because of being used */

    2 skipm fixed bin,                          /* because of being modified */
    2 skipos fixed bin,                         /* because out of service */
    2 aused fixed bin,                          /* number of AST entries on used list */
    2 damaged_ct fixed bin,                     /* count of segments that system damaged */
    2 deact_count fixed bin,                    /* count of deactivations */
    2 demand_deact_attempts fixed bin,          /* user requested deactivations */
    2 demand_deactivations fixed bin,           /* user instigated deactivations */
```

```
    2 reads (8) fixed bin,                  /* number of reads for each did */
    2 writes (8) fixed bin,                 /* number of writes for each did */

    2 short_pf_count fixed bin,             /* count of page faults on out of service pages */
    2 loop_locks fixed bin,                 /* count of times locked PTL */
    2 loop_lock_time fixed bin (71),        /* time spent looping on PTL */
    2 cpu_sf_time fixed bin (71),           /* cpu time spent in seg_fault */
    2 total_sf_pf fixed bin,                /* total page faults while in seg_fault */
    2 total_sf fixed bin,                   /* total number of seg_faults */
    2 pre_page_size fixed bin,              /* total pre-pagings expected */
    2 post_list_size fixed bin,
    2 post_purgings fixed bin,              /* total number of post-purgings */
    2 post_purge_calls fixed bin,           /* total number of calls to post-purge */
    2 pre_page_calls fixed bin,             /* total number of calls tp pre-page */
    2 pre_page_list_size fixed bin,
    2 pre_page_misses fixed bin,            /* total number of misses in pre-page list */
    2 pre_pagings fixed bin,                /* total number of pre-pagings */

/*  TEMPORARY WIRED PROCEDURE INFO  */

    2 wire_proc_data (8) fixed bin (71),    /* data for wire_proc */

/* MAIN MEMORY USAGE INFORMATION */

    2 abs_wired_count fixed bin,            /* count of abs-wired pages */
    2 system_type fixed bin,                /* ADP_SYSTEM or L68_SYSTEM */
    2 wired_copies fixed bin,               /* number of times a wired page was copied */
    2 recopies fixed bin,                   /* number of times recopied because modified */
    2 first_core_block fixed bin,           /* core map index for first block of core */
    2 last_core_block fixed bin,            /* core map index for last block of core */
    2 fw_retries fixed bin (35),            /* force_write retries due to ASTE move */
    2 pvhtp ptr unaligned,                  /* ptr to PV hold table for debugging */

/*  AST METERS  */

    2 askipsize (0: 3) fixed bin,           /* array of skips because wrong AST size */
    2 aneedsize (0: 3) fixed bin,           /* array of times needed each size */

    2 stepsa fixed bin,                     /* count of steps taken looking for an AST entry */
    2 askipsehs fixed bin,                  /* count of skips because EHS was ON */
    2 asearches fixed bin,                  /* count of full searches made */
    2 askipslevel fixed bin,                /* count of skips because pages were in core */
    2 askipsinit fixed bin,                 /* count of times turned OFF init switch */
    2 acost fixed bin,                      /* cumulative cost of deactivations */
    2 askipslock fixed bin,                 /* count of skips because couldn't lock parent */
    2 askipdius fixed bin,                  /* count of skips because DIUS was on */

    2 alaps fixed bin,                      /* lap counter for AST list */
    2 updates fixed bin,                    /* calls to updateb */
    2 setfaults_all fixed bin,              /* setfaults done to the entire SDW */
    2 setfaults_acc fixed bin,              /* setfaults done to the access field */
    2 total_bf fixed bin,                   /* count of bound faults */
    2 total_bf_pf fixed bin,                /* page faults during bound faults */
    2 cpu_bf_time fixed bin (71),           /* cpu time spent in bound fault */
```

```
    2 asteps (0: 3) fixed bin,                    /* per-size AST step counters */

    2 ast_locked_at_time fixed bin (71),           /* clock reading when ast last locked */
    2 ast_locked_total_time fixed bin (71),        /* total real time the ast lock was locked */
    2 ast_lock_wait_time fixed bin (71),           /* total real time of all waiting on ast lock */
    2 ast_locking_count fixed bin (35),            /* number of times ast was locked */
    2 cleanup_count fixed bin,                     /* calls to pc$cleanup */
    2 cleanup_real_time fixed bin (71),            /* total real time in pc$cleanup */

/* PRE-PAGE METERS */

    2 tree_count (0: 63) fixed bin,                /* counters for pre-page decisions */

    2 pp_meters (0: 63) fixed bin,                 /* counters for measuring pre-page success */


    2 wusedp bit (18) aligned,                     /* Relative cmep to next cme for writing */
    2 write_hunts fixed bin,                       /* Times claim_mod_core invoked */
    2 claim_skip_cme fixed bin,                    /* Times unacceptable cme found by c_m_c */
    2 claim_skip_free fixed bin,                   /* Times free cme passed by c_m_c */
    2 claim_notmod fixed bin,                      /* Times c_m_c passed pure page */
    2 claim_passed_used fixed bin,                 /* Times used page seen */
    2 claim_skip_ptw fixed bin,                    /* Times c_m_c saw unacceptable ptw */
    2 claim_writes fixed bin,                      /* Writes queued by c_m_c */
    2 claim_steps fixed bin,                       /* Steps passed in core claiming */
    2 pre_seeks_failed fixed bin,                  /* counter of times quick find_core_ failed */
    2 resurrections fixed bin,                     /* nulled addresses reinstated */
    2 volmap_seg_page_faults fixed bin (35),       /* Pseudo-page faults on volmap_seg */
    2 oopv fixed bin,                              /* out-of-physical-volume page faults */
    2 dblw_resurrections fixed bin,                /* addresses resurrected by double-writing */
    2 sgm_time fixed bin (71),                     /* Time (VCPU) in seg mover */
    2 sgm_pf fixed bin,                            /* Page faults in seg moving */
    2 bad_sgms fixed bin,                          /* Seg moves that failed */
    2 sgm_sgft fixed bin,                          /* Seg faults in seg moves */
    2 good_sgms fixed bin,                         /* Seg moves that completed */
    2 claim_runs fixed bin,                        /* Times claim_mod_core had to run */
    2 activations fixed bin,                       /* total count of activations */
    2 dir_activations fixed bin,                   /* count of directory activations */
    2 hedge_updatevs fixed bin,                    /* call-in updatevs */
    2 hedge_writes fixed bin,                      /* call in core flush writes */
    2 evict_recover_data,                          /* see evict_page.alm */
     3 evict_ptp bit (18) unal,                    /* ptp of page being moved */
     3 evict_phmbit bit (18) unal,                 /* N/Z if page was mod */

/* Data for metering force_write facility 08/19/78 */

    2 force_swrites fixed bin,                     /* Calls on segments to force write */
    2 force_pwrites fixed bin,                     /* Mod pages so written */
    2 fw_none fixed bin,                           /* Force write wrote none */
    2 force_updatevs fixed bin,                    /* Updatev's so forced */

    2 pf_unlock_ptl_time fixed bin (71),           /* Time unlocking ptln page faults */
    2 pf_unlock_ptl_meterings fixed bin,

    2 makeknown_activations fixed bin (35),        /* activations at makeknown time */
    2 backup_activations fixed bin (35),           /* activations for backup */
```

```
    2 metering_flags aligned,                                   /* small chunks of misc. information */
      3 activate_activated bit (1) unal,                        /* ON => last call to activate entry actually activated something */
      3 pad bit (35) unal,
    2 seg_fault_calls fixed bin (35),                           /* number calls to seg_fault for explicit activation */

/* METERS FOR STACK TRUNCATION */

    2 (stk_truncate_should_didnt,                               /* counts */
       stk_truncate_should_did,
       stk_truncate_shouldnt_didnt,
       stk_truncate_shouldnt_did) fixed bin (35),
    2 stk_pages_truncated fixed bin (35),
    2 stk_pages_truncated_in_core fixed bin (35),

/* SUPPORT FOR PC SEGMOVES */

    2 segmove_lock aligned,
      3 pid bit (36) aligned,
      3 event bit (36) aligned,
      3 notify bit (1) aligned,
    2 segmove_io_limit fixed bin, /* max read aheads */
    2 segmove_found_synch fixed bin (35), /* cme.synch_held */
    2 segmove_synch_disappeared fixed bin (35), /* page$check_synch fixed */
    2 segmove_n_reads fixed bin (35), /* total IO's queued. */
    2 segmove_max_tries fixed bin (35), /* max times through the read loop */

    2 segmove_astep ptr unal,                                   /* if non-null, addresses to be rescued from old_addr_astep */
    2 segmove_pvtx fixed bin,                                   /* if segmove_astep nonnull, valid */
    2 segmove_vtocx fixed bin,                                  /* ditto */
    2 segmove_old_addr_astep ptr unaligned,                     /* ditto */
    2 segmove_new_addr_astep ptr unaligned,                     /* if non-null, the addresses must be deposited. */

    2 mod_during_write fixed bin,                               /* times a page was modified while it was being written */
    2 zero_pages fixed bin,                                     /* count of pages truncated because all zero */
    2 trace_sw aligned,                                         /* tracing control flags */
      3 pad_trace bit (32) unaligned,
      3 pc_trace_pf bit (1) unaligned,                          /* tracing for page faults, done, etc. */
      3 tty_trace bit (1) unaligned,
      3 pc_trace bit (1) unaligned,                             /* flag used by page control primitives */
      3 sc_trace bit (1) unaligned,                             /* flag used by segment control primitives */
    2 new_pages fixed bin,                                      /* newly created pages */
    2 ast_track bit (1) aligned,                                /* "1"b => keep SST name table */
    2 dirlock_writebehind fixed bin,                            /* =1 to flush modified dir pages in lock$unlock */
    2 write_limit fixed bin,                                    /* Max # of outstanding writes by page control */
    2 crash_test_segmove bit (1) aligned,                       /* crash in mid-segmove */
    2 delayed_seg_state_chg fixed bin (35),                     /* count of times a process was delayed in affecting a seg state */
    2 audit_seg_state_chg fixed bin (35),                       /* count of times a process was audited for excessive seg state changes */
    2 seg_state_chg_delay fixed bin (52),                       /* total times processes were delayed for covert channels */
    2 seg_state_change_limit fixed bin,                         /* number of events over which we determine covert channel bandwidth */
    2 max_seg_state_change_bw fixed bin,                        /* maximum bps for covert channel before we delay */
    2 audit_seg_state_change_bw fixed bin,                      /* maximum bps for covert channel before we audit */
    2 seg_state_chg_operation bit (36) aligned,                 /* access_operation_ value for excessive_seg_state_chg */
    2 pad4 (126) bit (36) aligned;                              /* padding to 512 words (1000)8 */

/* END INCLUDE FILE sst.incl.pl1 */
```

```
/* Begin include file sstnt.incl.pl1   */

/* Created 10/03/74 by Bernard Greenberg */
/* modified 08/24/79 by J. A. Bush  for easier calculation of size of sstnt */
/* Modified 08/27/84 by Keith Loepere to purge BOS */

dcl  sst_names_$ ext;                                   /* Segment containing sst name table */

dcl  sstnp ptr;                                         /* Pointer to sst name segment */

dcl 1 sstnt based (sstnp) aligned,                      /* Major structure */
    2 valid bit (1) aligned,                            /* 1 => structure filled by Multics */
    2 multics_or_bce char (4) aligned,                  /* Origin of data in table */
    2 nentries fixed bin,                               /* number of entries in the sstnt */
    2 pad1 (5) fixed bin,

    2 (ast_sizes,                                       /* Sizes of ASTE's at each level */
     ast_name_offsets,                                  /* Starting index for names at each level */
     ast_offsets,                                       /* Starting rel addr of each AST region */
     pad2) (0 : 3) fixed bin,

    2 names (0 : 0 refer (sstnt.nentries)) char (32) varying; /* Names of AST entries */

dcl (sstnmx, ptsi_a) fixed bin (17);                    /* Index into name table */

dcl  nm_astep ptr;                                      /* astep to be used */

/* End include file sstnt.incl.pl1 */
```

```
/* BEGIN INCLUDE FILE ... stack_0_data.incl.pl1 */

/* Created 790509 by Mike Grady */

dcl  stack_0_data$ fixed bin ext;                       /* shared stack 0 data base seg */
dcl  stack_0_data_init_number_of_stacks fixed bin;      /* Make PL/I work */
dcl  sdtp ptr;

dcl 1 sdt aligned based (sdtp),                         /* stack 0 database */
    2 lock bit (36),                                    /* lock before changing threads */
    2 num_stacks fixed bin,                             /* number of stacks in pool */
    2 freep bit (18),                                   /* head of free thread, managed LIFO */
    2 pad fixed bin,
    2 stacks (stack_0_data_init_number_of_stacks
         refer (sdt.num_stacks)) like sdte;

dcl  sdtep ptr;

dcl 1 sdte aligned based (sdtep),                       /* stack data table entry */
    2 nextp bit (18) unal,                              /* thread to next free entry (if free) */
    2 pad bit (18) unal,
    2 astep bit (18) unal,                              /* ptr to ASTE for this stack seg */
    2 aptep bit (18) unal,                              /* ptr to APTE of process using this stack, if not free */
    2 sdw bit (72);                                     /* SDW for this stack seg */

/* END INCLUDE FILE ... stack_0_data.incl.pl1 */
```

```
/*       BEGIN INCLUDE FILE ... stack_frame.incl.pl1 ... */

/* Modified: 16 Dec 1977, D. Levin - to add fio_ps_ptr and pl1_ps_ptr */
/* Modified:  3 Feb 1978, P. Krupp - to add run_unit_manager bit & main_proc bit */
/* Modified: 21 March 1978, D. Levin - change fio_ps_ptr to support_ptr */
/* Modified: 03/01/84, S. Herbst - Added RETURN_PTR_MASK */

dcl RETURN_PTR_MASK bit (72) int static options (constant)  /* mask to be AND'd with stack_frame.return_ptr */
         init ("777777777777777777000000"b3);               /* when copying, to ignore bits that a call fills */
                                                            /* with indicators (nonzero for Fortran hexfp caller) */
                              /* say: unspec(ptr) = unspec(stack_frame.return_ptr) & RETURN_PTR_MASK; */


dcl  sp pointer;                                            /* pointer to beginning of stack frame */

dcl  stack_frame_min_length fixed bin static init(48);


dcl 1 stack_frame based(sp) aligned,
    2 pointer_registers(0 : 7) ptr,
    2 prev_sp pointer,
    2 next_sp pointer,
    2 return_ptr pointer,
    2 entry_ptr pointer,
    2 operator_and_lp_ptr ptr,                              /* serves as both */
    2 arg_ptr pointer,
    2 static_ptr ptr unaligned,
    2 support_ptr ptr unal, /* only used by fortran I/O */
    2 on_unit_relp1 bit(18) unaligned,
    2 on_unit_relp2 bit(18) unaligned,
    2 translator_id bit(18) unaligned,                      /* Translator ID
                                                               0 => PL/I version II
                                                               1 => ALM
                                                               2 => PL/I version I
                                                               3 => signal caller frame
                                                               4 => signaller frame */
    2 operator_return_offset bit(18) unaligned,
    2 x(0: 7) bit(18) unaligned,                            /* index registers */
    2 a bit(36),                                            /* accumulator */
    2 q bit(36),                                            /* q-register */
    2 e bit(36),                                            /* exponent */
    2 timer bit(27) unaligned,                                      /* timer */
    2 pad bit(6) unaligned,
    2 ring_alarm_reg bit(3) unaligned;


dcl 1 stack_frame_flags based(sp) aligned,
    2 pad(0 : 7) bit(72),                                   /* skip over prs */
    2 xx0 bit(22) unal,
    2 main_proc bit(1) unal,                                /* on if frame belongs to a main procedure */
```

```
    2 run_unit_manager bit(1) unal,                 /* on if frame belongs to run unit manager */
    2 signal bit(1) unal,                            /* on if frame belongs to logical signal_ */
    2 crawl_out bit(1) unal,                         /* on if this is a signal caller frame */
    2 signaller bit(1) unal,                         /* on if next frame is signaller's */
    2 link_trap bit(1) unal,                         /* on if this frame was made by the linker */
    2 support bit(1) unal,                           /* on if frame belongs to a support proc */
    2 condition bit(1) unal,                         /* on if condition established in this frame */
    2 xx0a bit(6) unal,
    2 xx1 fixed bin,
    2 xx2 fixed bin,
    2 xx3 bit(25) unal,
    2 old_crawl_out bit (1) unal,                    /* on if this is a signal caller frame */
    2 old_signaller bit(1) unal,                     /* on if next frame is signaller's */
    2 xx3a bit(9) unaligned,
    2 xx4(9) bit(72) aligned,
    2 v2_pl1_op_ret_base ptr,                        /* When a V2 PL/I program calls an operator the
                                                     *  operator puts a pointer to the base of
                                                     *  the calling procedure here. (text base ptr)  */
    2 xx5 bit(72) aligned,
    2 pl1_ps_ptr ptr;                                /* ptr to ps for this frame; also used by fio. */

/*       END INCLUDE FILE ... stack_frame.incl.pl1 */
```

```
/*        BEGIN INCLUDE FILE ... stack_header.incl.pl1 .. 3/72 Bill Silver  */
/*        modified 7/76 by M. Weaver for *system links and more system use of areas */
/*        modified 3/77 by M. Weaver to add rnt_ptr */
/*        Modified April 1983 by C. Hornig for tasking. (the trace stuff is temporary - MBW) */

/* format: style2 */
     dcl     sb                     ptr;                    /* the  main pointer to the stack header */

     dcl     1 stack_header         based (sb) aligned,
               2 pad1               (4) fixed bin,          /*  (0) also used as arg list by outward_call_handler  */
               2 old_lot_ptr        ptr,                    /*  (4)  pointer to the lot for current ring (obsolete) */
               2 combined_stat_ptr  ptr,                    /*  (6)  pointer to area containing separate static */
               2 clr_ptr            ptr,                    /*  (8)  pointer to area containing linkage sections */
               2 max_lot_size       fixed bin (17) unal,    /*  (10) DU  number of words allowed in lot */
               2 main_proc_invoked  fixed bin (11) unal,    /*  (10) DL  nonzero if main procedure invoked in run unit */
               2 have_static_vlas   bit (1) unal,           /*  (10) DL  "1"b if (very) large arrays are being used in static */
               2 pad4               bit (2) unal,
               2 run_unit_depth     fixed bin (2) unal,     /*  (10) DL  number of active run units stacked */
               2 cur_lot_size       fixed bin (17) unal,    /*  (11)  number of words (entries) in lot */
               2 pad2               bit (18) unal,          /*  (11)  reserved */
               2 system_free_ptr    ptr,                    /*  (12)  pointer to system storage area */
               2 user_free_ptr      ptr,                    /*  (14)  pointer to user storage area */
               2 null_ptr           ptr,                    /*  (16)  */
               2 stack_begin_ptr    ptr,                    /*  (18)  pointer to first stack frame on the stack */
               2 stack_end_ptr      ptr,                    /*  (20)  pointer to next useable stack frame */
               2 lot_ptr            ptr,                    /*  (22)  pointer to the lot for the current ring */
               2 signal_ptr         ptr,                    /*  (24)  pointer to signal procedure for current ring */
               2 bar_mode_sp        ptr,                    /*  (26)  value of sp before entering bar mode */
               2 pl1_operators_ptr  ptr,                    /*  (28)  pointer to pl1_operators_$operator_table */
               2 call_op_ptr        ptr,                    /*  (30)  pointer to standard call operator */
               2 push_op_ptr        ptr,                    /*  (32)  pointer to standard push operator */
               2 return_op_ptr      ptr,                    /*  (34)  pointer to standard return operator */
               2 return_no_pop_op_ptr
                                    ptr,                    /*  (36)  pointer to standard return / no pop operator */
               2 entry_op_ptr       ptr,                    /*  (38)  pointer to standard entry operator */
               2 trans_op_tv_ptr    ptr,                    /*  (40)  pointer to translator operator ptrs */
               2 isot_ptr           ptr,                    /*  (42)  pointer to ISOT */
               2 sct_ptr            ptr,                    /*  (44)  pointer to System Condition Table */
               2 unwinder_ptr       ptr,                    /*  (46)  pointer to unwinder for current ring */
               2 sys_link_info_ptr  ptr,                    /*  (48)  pointer to *system link name table */
               2 rnt_ptr            ptr,                    /*  (50)  pointer to Reference Name Table */
               2 ect_ptr            ptr,                    /*  (52)  pointer to event channel table */
               2 assign_linkage_ptr ptr,                    /*  (54)  pointer to storage for (obsolete) hcs_$assign_linkage */
               2 task_data_ptr      ptr,                    /*  (56)  for possible tasking (experimental) */
               2 trace,                                     /*        expected to be temporary */
                 3 frames,
                   4 count          fixed bin,              /*  (58)  number of trace frames */
                   4 top_ptr        ptr unal,               /*  (59)  pointer to last trace frame */
                 3 in_trace         bit (36) aligned,       /*  (60)  trace antirecursion flag */
               2 pad3               (3) bit (36) aligned;   /*  (61)  for future expansion */
```

```
/*       The following offset refers to a table within the  pl1  operator table.  */

     dcl     tv_offset              fixed bin init (361) internal static;
                                                          /* (551) octal */


/*       The following constants are offsets within this transfer vector table.  */

     dcl     (
             call_offset            fixed bin init (271),
             push_offset            fixed bin init (272),
             return_offset          fixed bin init (273),
             return_no_pop_offset   fixed bin init (274),
             entry_offset           fixed bin init (275)
             )                      internal static;



/*       The following declaration  is an overlay of the whole stack header.   Procedures which
         move the whole stack header should use this overlay.
*/

     dcl     stack_header_overlay   (size (stack_header)) fixed bin based (sb);


/*       END INCLUDE FILE ... stack_header.incl.pl1 */
```

```
/*  BEGIN INCLUDE FILE ... str.incl.pl1 ... last modified March 1970 */

dcl str_seg$ ext,
    strp ptr;

dcl 1 str based (strp) aligned,                 /* segment or process trailer declaration */

   (2 fp bit (18),                              /* forward ast trailer rel pointer */
    2 bp bit (18),                              /* backward ast trailer rel pointer*/

    2 segno bit (18),                           /* segment number*/
    2 dstep bit (18)) unaligned;                /* rel pointer to ring 0 dste */

dcl stra (0:8000) bit (72) based (strp) aligned;

/*  END INCLUDE FILE ... str.incl.pl1 */
```

```
/* BEGIN INCLUDE FILE ... tcm.incl.pl1 ... used to generate tc_data cds */
/* NOTE -- This include file has TWO counterparts in ALM: tc_meters.incl.alm and */
/* wcte.incl.alm. They cannot be produced with cif, and must be kept up to date manually. */
/* Modified 830914 to replace tty_polling_time with opc_polling_time... -E. A. Ranzenbach */
/* Modified 1984.05.21 by M. Pandolf to add tc_suspend_lock */
/* Modified 1984.11.26 by Keith Loepere for uid_array. */
/* Modified 1984.12.06 by Keith Loepere for page create delaying. */

dcl  tcmp ptr;

dcl 1 tcm aligned based (tcmp),
    2 tc_suspend_lock like lock,                    /* when locked, tc is suspended */
    2 cid2 fixed bin (18),
    2 cid3 fixed bin (18),
    2 cid4 fixed bin (18),
    2 depth_count fixed bin (18),                   /* depth last process run */
    2 loadings fixed bin (18),                      /* number of process loadings */

    2 blocks fixed bin (18),                        /* number of calls to block */
    2 wakeups fixed bin (18),                       /* number of calls to wakeup */
    2 waits fixed bin (18),                         /* number of calls to wait */
    2 notifies fixed bin (18),                      /* number of calls to notify */
    2 schedulings fixed bin (18),
    2 interactions fixed bin (18),                  /* number of interactive schedulings */
    2 avsqueue fixed bin (35, 18),                  /* recent time average of number in queue */
    2 te_wait fixed bin (18),                       /* times te called from wait */

    2 te_block fixed bin (18),                      /* times te updated from block */
    2 te_i_stop fixed bin (18),                     /* times te updated from i_stop */
    2 te_pre_empt fixed bin (18),                   /* times te updated from pre_empt */
    2 p_interactions fixed bin,                     /* times interaction bit turned off because of high priority */
    2 idle fixed bin (71),                          /* total idle time */
    2 mp_idle fixed bin (71),                       /* multi-programming idle */

    2 nmp_idle fixed bin (71),                      /* non-multi-programming idle time */
    2 zero_idle fixed bin (71),                     /* zero idle time */
    2 last_time fixed bin (71),                     /* last time a process was run */
    2 loop_locks fixed bin (18),                    /* times looped on the APT lock */
    2 loop_lock_time fixed bin (18),                /* time looping on the APT lock */
    2 avs_eligible fixed bin (35, 18),              /* average length of eligible queue */
    2 sort_to_elhead fixed bin (18),                /* 0=> no one,1 => int've only, 2 => everybody */
    2 processor_time fixed bin (71),                /* total processor time on system */
    2 response_time fixed bin (71),                 /* estimate of response time */
    2 eligible_time fixed bin (71),                 /* estimate of eligible time */
    2 response_count fixed bin,                     /* count of response meters */
    2 eligible_count fixed bin,                     /* count of eligible meters */
    2 quit_counts (0:5) fixed bin,                  /* array of buckets indexed by state */
    2 loading_idle fixed bin (71),                  /* loading_idle time */
    2 delta_vcpu fixed bin (71),                    /* delta virtual CPU time for the system */
    2 post_purge_switch fixed bin,                  /* ON if post purging is to be done */
```

```
    2 time_out_severity fixed bin,                  /* syserr first arg for notify time outs */
    2 notify_check fixed bin,                        /* obsolete */
    2 quit_priority fixed bin,                       /* factor for scheduler quit response */
    2 iobm_polling_time fixed bin (71),              /* time to poll iobm */
    2 end_of_time fixed bin (71),                    /* very large time */
    2 gp_at_notify fixed bin (18),                   /* 0 => just do get_idle_processor */
    2 gp_at_ptlnotify fixed bin (18),                /* 0 => just do get_idle_processor */
    2 int_q_enabled fixed bin (18),                  /* 0 => no intv q in percent mode */
    2 fnp_buffer_threshold fixed bin (18),           /* if fewer free buffs then stingy alloc strategy */
                                                     /* set this to >= half n_ttylines/fnp for safety */

/* 100 octal */

    2 depths (8) fixed bin (18),                     /* histogram of run depths */
    2 tdepths (8) fixed bin (71),                    /* histogram of times run per depth */
    2 pfdepth (8) fixed bin (18),                    /* histogram of page faults per depth */

    2 ptl_not_waits fixed bin (18),                  /* times ptl_wait noticed ptl was unlocked */
    2 gw_gp_window_count fixed bin (18),             /* times window noticed */
    2 metering_lock fixed bin (18),                  /* 0=locked, else unlocked */
    2 ptl_waits fixed bin (18),                      /* num calls to ptl_wait */
    2 gp_start_count fixed bin (18),                 /*  to detect gw_gp window lossage */
    2 gp_done_count fixed bin (18),
    2 nto_check_time fixed bin (71),                 /* next time at which nto code will be called */
    2 nto_delta fixed bin (35),                      /* microsec between nto checks */
    2 nto_count fixed bin (18),                      /* number of times nto detected */
    2 tcpu_scheduling fixed bin (18),                /* obsolete                                    */
    2 nto_event bit (36),                            /* last event which NTO'd */
    2 page_notifies fixed bin (18),
    2 notify_nobody_count fixed bin (18),
    2 notify_nobody_event bit (36),
    2 system_type fixed bin,                         /* used to be tcm.inter */

    2 stat (0:15) fixed bin (18),                    /* num apte's in each state */

/* 200 octal */

    2 wait (8),
      3 time fixed bin (18),                         /* histogram of page fault waiting times versus did */
      3 count fixed bin (18),

    2 ready (8),
      3 time fixed bin (18),                         /* histogram of times in ready queue */
      3 count fixed bin (18),

    2 total_pf_time fixed bin (71),                  /* total time spent from start to end of
                                                        all page faults */
    2 total_pf_count fixed bin (18),                 /* total number of page faults metered */
    2 auto_tune_ws fixed bin (18),                   /* 0=> dont, atherwise compensate for quantum len */
    2 ocore_delta fixed bin (18),                    /* number of pages reserved for int users */
    2 ws_sum fixed bin (18),                         /* total of eligible's ws_sizes */
    2 nonidle_force_count fixed bin (18),            /* count of eligibilities forced */
    2 itt_list_lock bit (36) aligned,                /* Lock on ITT free list */
    2 cpu_pf_time fixed bin (71),                    /* total cpu time spent handling page faults */
    2 cpu_pf_count fixed bin (18),                   /* total count of cpu time meterings */
    2 special_offsets unaligned,
```

```
      3 apt_offset bit (18),
      3 pad bit (18),
    2 getwork_time fixed bin (71),          /* total time spent in getwork */
    2 getwork_count fixed bin (18),         /* total times through getwork */
    2 short_pf_count fixed bin (18),        /* number of short page faults */
    2 interrupt_time fixed bin (71),        /* total time spent in interrupt */
    2 interrupt_count fixed bin (71),       /* total number of metered interrupts */
    2 ocore fixed bin (35, 18),             /* fraction of core for int've users */
    2 pre_empt_flag bit (36) aligned,       /* controls whether preempting at done time */
    2 cumulative_memory_usage fixed binary (71),  /* total number of memory usage units */
    2 processor_time_at_define_wc fixed bin (71), /* value of processor_time when WC's last defined */
    2 boost_priority fixed bin,             /* number of times priority process given high priority */
    2 lost_priority fixed bin,              /* number of times priority process lost eligibility */
    2 total_clock_lag fixed bin (71),       /* sum of all simulated clock delays */
    2 clock_simulations fixed bin,          /* number of times alarm clock interrupt was simulated */
    2 max_clock_lag fixed bin,              /* largest simulated alarm clock delay */

/* 300 octal */

    2 pdscopyl fixed bin (18),              /* amount of pds to copy for new process */
    2 max_hproc_segno fixed bin,            /* largest allowed hardcore segment number */
    2 prds_length fixed bin (18),           /* length of PRDS */
    2 pds_length fixed bin (18),            /* length of PDS */
    2 lock fixed bin (18),                  /* process id generator lock */
    2 id bit (36) aligned,                  /* next uid to be added to uid_array */
    2 system_shutdown fixed bin (18),
    2 working_set_factor fixed bin (35, 18),  /* working set factor */

    2 ncpu fixed bin (18),                  /* number of processors currently being used */
    2 last_eligible bit (18),               /* last process to gain eligibility */
    2 apt_lock fixed bin (35),              /* + write; 0 hidden; -1 unlocked; -(N+1) Nreaders */
    2 apt_size fixed bin (18),              /* number of APT entries */
    2 realtime_q aligned like based_sentinel,  /* processes with realtime deadlines */
    2 aht_size fixed bin (18),              /* APT hash table size */
    2 itt_size fixed bin (18),              /* number of ITT entries */

    2 dst_size fixed bin (18),              /* number of allowed DST entries */
    2 itt_free_list bit (18),               /* pointer to ITT free list */
    2 used_itt fixed bin (18),              /* number of used ITT entries */
    2 initializer_id bit (36) aligned,      /* process id of initializer */
    2 n_eligible fixed bin (18),            /* number of processes eligible */
    2 max_eligible fixed bin (30),          /* maximum allowed number of eligible processes */
    2 wait_enable fixed bin (18),           /* turned on when waiting mechanism works */
    2 apt_entry_size fixed bin (18),        /* size of an APT entry */

    2 interactive_q aligned like based_sentinel,  /* head of interactive queue */
    2 dst_ptr ptr,                          /* pointer to device signal table */
    2 old_user ptr,                         /* last process to run (apt ptr ) */
    2 initialize_time fixed bin (71),       /* time of initialization */

    2 init_event fixed bin (18),            /* wait event during initialization */
    2 oldt fixed bin (18),                  /* timer reading from previous process */
    2 newt fixed bin (18),                  /* timer setting for new process */
    2 tefirst fixed bin (30),               /* first eligible time */
    2 telast fixed bin (30),                /* last eligible time */
    2 timax fixed bin (35),                 /* time in queue for lowest level */
```

```
    2 empty_q bit (18),                          /* thread of empty APT entries */
    2 working_set_addend fixed bin (18),          /* additive working set parameter */
    2 ready_q_head bit (0) aligned,               /* for added segdef */
    2 eligible_q_head aligned like based_sentinel, /* head of eligible queue */
    2 ready_q_tail bit (0) aligned,               /* for added segdef */
    2 eligible_q_tail aligned like based_sentinel, /* tail of eligible queue */
    2 idle_tail aligned like based_sentinel,      /* tail of idle list */
    2 min_eligible fixed bin (30),
    2 alarm_timer_list bit (18) aligned,          /* rel pointer to apt entry for next alarm timer */
    2 guaranteed_elig_inc fixed bin (35),         /* amount of guaranteed eligibility time in microsecs. */
    2 priority_sched_inc fixed bin (35),          /* amount of block time before process is given priority */
    2 next_alarm_time fixed bin (71),             /* clock time for next alarm timer */
    2 priority_sched_time fixed bin (71),         /* time for priority process to be given priority */
    2 opc_polling_time fixed bin (71),            /* time to poll console DIM */
    2 disk_polling_time fixed bin (71),           /* time to poll disk DIM */
    2 tape_polling_time fixed bin (71),           /* time to poll tape DIM */
    2 imp_polling_time fixed bin (71),            /* time to poll imp */
    2 imp_polling_lock fixed bin (18),            /* do not poll if lock set */
    2 max_channels fixed bin (18),                /* num special channels per process */

/* 400 octal */

    2 system_virtual_time fixed bin (71),         /* non-idle virtual time */
    2 credit_bank fixed bin (71),                 /* credits not yet passed out */
    2 min_wct_index bit (18) aligned,             /* offset of initializer work class table entry */
    2 max_wct_index bit (18) aligned,             /* offset of highest wcte currently defined */
    2 delta_vt fixed bin (71),                    /* temp used by pxss.compute_virtual_clocks */
    2 gross_idle_time fixed bin (71),             /* idle time_used_clock */
    2 credits_per_scatter fixed bin (35),         /* total number of credits awarded at once */
    2 best_credit_value fixed bin (18),           /* temp for pxss.find_next_eligible */
    2 define_wc_time fixed bin (71),              /* clock time when workclasses last degined */
    2 max_batch_elig fixed bin (35),
    2 num_batch_elig fixed bin (35),
    2 deadline_mode fixed bin (35),               /* 0=> ti sorts, else deadline sorts */
    2 credits_scattered fixed bin (35),
    2 max_max_eligible fixed bin (30),            /* Maximum of maxe */
    2 max_stopped_stack_0 fixed bin (35),         /* Maximum stack_0's suspended by stopped procs   */
    2 stopped_stack_0 fixed bin (35),             /* Number stack_0's suspended by stopped procs    */
    2 mos_polling_interval fixed bin (35),        /* for heals */
    2 mos_polling_time fixed bin (71),            /* for heals */
    2 vcpu_response_bounds (VCPU_RESPONSE_BOUNDS) fixed bin (35),
    2 vcpu_response_bounds_size fixed bin (35),
    2 meter_response_time_calls fixed bin (35),
    2 meter_response_time_invalid fixed bin (35),
    2 meter_response_time_overhead fixed bin (71),
    2 init_wait_time fixed bin (71),              /* used by wait/notify during initialization      */
    2 init_wait_timeout fixed bin (71),           /* notify-timeout interval during initialization  */
    2 init_timeout_severity fixed bin,            /* notify-timeout severity during initialization  */
    2 init_timeout_recurse fixed bin,             /* count of NTO recursion during initialization   */
    2 max_timer_register fixed bin (71),          /* max cpu burst = # cpus x pre_empt_sample_time  */
    2 pre_empt_sample_time fixed bin (35),        /* tuning parameter - max time between samples    */
    2 governing_credit_bank fixed bin (35),       /* used for limiting eligibility on governed work classes*/
    2 process_initial_quantum fixed bin (35),     /* eligibility quantum first eligibility          */
    2 default_procs_required bit (8) aligned,     /* default mask of CPUs required */
    2 work_class_idle fixed bin (71),             /* idle time due to work class restrictions */
```

```
/* Tuning Parameters for Stack Truncation */

    2 stk_truncate bit (1) aligned,
    2 stk_truncate_always bit (1) aligned,
    2 stk_trunc_avg_f1 fixed bin (35, 18),
    2 stk_trunc_avg_f2 fixed bin (35, 18),
    2 lock_error_severity fixed bin,                    /* syserr severity */

    2 gv_integration fixed bin (35),                    /* Integration interval for governing */
    2 gv_integration_set bit (1) aligned,               /* ON => gv_integration set by ctp */
    2 pauses fixed bin (35),                            /* Calls to pause (reschedule) */
    2 volmap_polling_time fixed bin (71),
    2 next_ring0_timer fixed bin (71),                  /* next time that ring 0 timer goes off */
    2 realtime_io_priority_switch fixed bin,            /* 0 => give I/O interrupt wakeups realtime priotiry */
    2 realtime_io_deadline fixed bin (35),              /* Delta to clock for I/O realtime deadline */
    2 realtime_io_quantum fixed bin (35),               /* Quantum for I/O realtime burst */
    2 realtime_priorities fixed bin (35),               /* Count for metering */
    2 relinquishes fixed bin (35),                      /* Calls to relinquish_priority */
    2 abort_ips_mask bit (36) aligned,                  /* IPS mask for tc_util$check_abort */

/* 500 octal */

    2 uid_array (0:15) bit (36) aligned,                /* array from which a uid is chosen (randomly) */
    2 pad5 (176) fixed bin (35),                        /* room for expansion compatibly                */

/* 1000 octal */

    2 pad7 (64) fixed bin (35),

/* 1100 octal */

    2 pad6 (8) fixed bin (35),
    2 work_class_table aligned,                         /* array of per workclass information */
      3 wcte (0:16) aligned like wct_entry,

/* 3000 octal */

    2 apt fixed bin;

dcl  wctep ptr;

dcl 1 wct_entry aligned based (wctep),                  /* Work class entry */
    2 thread unaligned,                                 /* Ready list */
      3 fp bit (18),                                    /* Head of ready list */
      3 bp bit (18),                                    /* Tail of ready list */
    2 flags unaligned,
      3 mnbz bit (1),                                   /* Sentinel bit must not be zero. */
      3 defined bit (1),
      3 io_priority bit (1),
      3 governed bit (1),
      3 interactive_q bit (1),
      3 pad bit (31),
    2 credits fixed bin (35),                           /* Current worthiness of group */
    2 minf fixed bin (35),                              /* min fraction of cpu */
    2 pin_weight fixed bin (35),                        /* number of cycles to pin pages */
    2 eligibilities fixed bin (35),                     /* Count of eligibilities awarded */
```

```
    2 cpu_sum fixed bin (71),                          /* CPU used by members */
    2 resp1 fixed bin (71),
    2 resp2 fixed bin (71),
    2 quantum1 fixed bin (35),
    2 quantum2 fixed bin (35),
    2 rmeter1 fixed bin (71),
    2 rmeter2 fixed bin (71),
    2 rcount1 fixed bin (35),
    2 rcount2 fixed bin (35),
    2 realtime fixed bin (35),
    2 purging fixed bin (35),
    2 maxe1 fixed bin (35),
    2 ne1 fixed bin (35),
    2 number_thinks fixed bin (35),                    /* number times process entered "think" state     */
    2 number_queues fixed bin (35),                    /* number times process entered "queued" state    */
    2 total_think_time fixed bin (71),
    2 total_queue_time fixed bin (71),

/*  The next three arrays  correspond to the array vcpu_response_bounds                                   */

    2 number_processing (VCPU_RESPONSE_BOUNDS+1) fixed bin (35), /* number times entered "processing" state   */
    2 total_processing_time (VCPU_RESPONSE_BOUNDS+1) fixed bin (71),
    2 total_vcpu_time (VCPU_RESPONSE_BOUNDS+1) fixed bin (71),
    2 maxf fixed bin (35),                             /* maximum fraction of cpu time                   */
    2 governing_credits fixed bin (35),                /* for limiting cpu resources                     */
    2 pad1 (4) fixed bin (35);


dcl 1 based_sentinel aligned based,                    /* format of pxss-style sentinel */
    2 fp bit (18) unal,
    2 bp bit (18) unal,
    2 sentinel bit (36) aligned;

dcl VCPU_RESPONSE_BOUNDS fixed bin init (3) int static options (constant);

/* END INCLUDE FILE tcm.incl.pl1 */
```

```
/* BEGIN INCLUDE FILE ... vol_map.incl.pl1 */

dcl   vol_mapp      ptr;

dcl 1 vol_map based (vol_mapp) aligned,

    2 n_rec fixed bin(17),                  /* number of records represented in the map */
    2 base_add fixed bin(17),               /* record number for first bit in bit map */
    2 n_free_rec fixed bin(17),             /* number of free records */
    2 bit_map_n_words fixed bin(17),        /* number of words of the bit map */
    2 pad (60) bit(36),                     /* pad to 64 words */
    2 bit_map (3*1024 - 64) bit(36) ;       /* bit map - the entire vol map occupies 3 records */

/* END INCLUDE ... vol_map */
```

```
/*  START OF:       vtoc_buffer.incl.pl1   November 1982      * * * * * * * * * * * * * * * * * * */

     dcl     vtoc_buffer_seg$       ext;

     dcl     vtoc_buffer_segp       ptr;
     dcl     vtoc_buf_descp         ptr;
     dcl     vtoc_bufp              ptr;
     dcl     vtoc_buf_desc_arrayp   ptr;
     dcl     vtoc_buf_arrayp        ptr;

     dcl     vtoc_buf_n_buffers     fixed bin;
     dcl     vtoc_buf_n_buckets     fixed bin;


     dcl     1 vtoc_buffer          aligned based (vtoc_buffer_segp),

               2 lock,                                  /* Global lock for VTOC buffers */
                 3 processid        bit (36) aligned,   /* Owner */
                 3 wait_event       bit (36) aligned,   /* For lock */
                 3 notify_sw        bit (1) aligned,    /* ON => notify on unlock */

               2 n_bufs             fixed bin,          /* Number of full VTOCE buffers */
               2 n_hash_buckets     fixed bin,          /* Number of hash table buckets */
               2 hash_mask          bit (36) aligned,   /* Mask for hash algorithm */
               2 abs_addr           fixed bin (24),     /* Absolute address of vtoc_buffer_seg */
               2 wait_event_constant fixed bin (36) uns unal,   /* Constant to add to part index to form wait event */
               2 buf_desc_offset    bit (18),           /* Offset of buf_desc */
               2 buf_offset         bit (18),           /* Offset of buf */
               2 hash_table_offset  bit (18),           /* Offset of hash_table */
               2 search_index       fixed bin,          /* Roving pointer for buffer selection */
               2 unsafe_pvtx        fixed bin,          /* PVTE index with update in progress */
               2 scavenger_free_p_clock
                                    fixed bin (35),     /* Pseudo-Clock for scavenger-free-other-allocate race */
               2 meters,
                 3 call_get         fixed bin (35),     /* Calls to get_vtoce */
                 3 call_put         fixed bin (35),     /* Calls to put_vtoce */
                 3 call_alloc       fixed bin (35),     /* Calls to alloc_and_put_vtoce */
                 3 call_free        fixed bin (35),     /* Calls to free_vtoce */
                 3 call_await       fixed bin (35),     /* Calls to await_vtoce */
                 3 steps            fixed bin (35),     /* Steps through buffer allocation */
                 3 skip_os          fixed bin (35),     /* Skipped because out-of-service */
                 3 skip_hot         fixed bin (35),     /* Skipped because buffer hot */
                 3 skip_wait        fixed bin (35),     /* Skipped because notify_sw set */
                 3 disk_reads       fixed bin (35),     /* Number of same */
                 3 disk_writes      fixed bin (35),     /* Number of same */
                 3 get_buffer_calls fixed bin (35),     /* Number of calls to GET_BUFFER */
                 3 get_buffer_hits  fixed bin (35),     /* Number times VTOCE in buffer */
                 3 wait_calls       fixed bin (35),     /* Number of calls to WAIT */
                 3 wait_os          fixed bin (35),     /* Number of times had to wait */
                 3 scavenger_free_checks
```

```
                    fixed bin (35),      /* Number of times had to check pseudo-clock */
      3 scavenger_free_losses
                    fixed bin (35),      /* Number of times race lost between scavenger freeing and other allocate */
      3 pad (15)    fixed bin (35),

    2 hash_table    (vtoc_buf_n_buckets refer (vtoc_buffer.n_hash_buckets)) bit (18) aligned,


    2 buf_desc      (vtoc_buf_n_buffers refer (vtoc_buffer.n_bufs)) aligned like vtoc_buf_desc,

    2 buffer        (vtoc_buf_n_buffers refer (vtoc_buffer.n_bufs)) aligned like vtoce_buffer;



dcl 1 vtoc_buf_desc_array  (vtoc_buffer.n_bufs) aligned based (vtoc_buf_desc_arrayp) like vtoc_buf_desc;


dcl 1 vtoc_buf_desc        aligned based (vtoc_buf_descp),
      2 pvtx               fixed bin (17) unal,    /* PVTE index */
      2 vtocx              fixed bin (17) unal,    /* VTOCE Index */
      2 parts_used         bit (3) unal,           /* Mask of parts used or os */
      2 err                bit (1) unal,           /* ON => I/O error on buffer */
      2 notify_sw          bit (1) unal,           /* ON => notify requied on I/O completion */
      2 write_sw           bit (1) unal,           /* ON => write I/O */
      2 os                 bit (1) unal,           /* ON => I/O in progress */
      2 ioq                bit (1) unal,           /* ON => I/O has been requested */
      2 used               bit (1) unal,           /* ON => this descriptor is in use */
      2 pad                bit (9) unal,
      2 wait_index         fixed bin (17) unal,    /* Buffer index for forming wait event */
      2 ht_thread          bit (18) unal,          /* Offset of next entry in hash table */
      2 buf_rel            bit (18) unal;          /* Offset of buffer in segment */

dcl 1 vtoce_buffer_array   (vtoc_buffer.n_bufs) aligned based (vtoc_buf_arrayp) like vtoce_buffer;

dcl 1 vtoce_buffer         aligned based (vtoc_bufp),
      2 parts              (3) aligned,
        3 words            (64) bit (36) aligned;


dcl N_PARTS_PER_VTOCE      fixed bin int static options (constant) init (3);
dcl VTOCE_PART_SIZE        fixed bin int static options (constant) init (64);
dcl VTOCE_BUFFER_SIZE      fixed bin int static options (constant) init (3 * 64);
dcl N_VTOCE_PER_RECORD     fixed bin int static options (constant) init (5);
dcl N_SECTOR_PER_VTOCE     fixed bin int static options (constant) init (3);

/*  END OF:       vtoc_buffer.incl.pl1                  * * * * * * * * * * * * * * * * * * * */
```

```
/* BEGIN INCLUDE FILE ... vtoc_header.incl.pl1 */

dcl  vtoc_headerp ptr;

dcl 1 vtoc_header based (vtoc_headerp) aligned,

    2 version fixed bin (17),                    /* version number. The current version number is 1. * */
    2 n_vtoce fixed bin (17),                    /* number of vtoc entries */
    2 vtoc_last_recno fixed bin (17),            /* record number of the last record of the vtoc */
    2 n_free_vtoce fixed bin (17),               /* number of free vtoc entries */
    2 first_free_vtocx fixed bin (17),           /* index of the first vtoce in the free list */
    2 pad (3) bit (36),
    2 dmpr_bit_map (2048 - 8) bit (36);          /* space for dmpr bit map */

/* END INCLUDE ... vtoc_header */
```

```
/*       BEGIN INCLUDE FILE ...vtoce.incl.pl1 ... last modified September 1982 */
/* Template for a VTOC entry. Length = 192 words. (3 * 64). */
/* NOTE: vtoc_man clears pad fields before writing a vtoce. */

dcl  vtocep ptr;

dcl 1 vtoce based (vtocep) aligned,

    (2 pad_free_vtoce_chain bit (36),          /* Used to be pointer to next free VTOCE */

    2 uid bit (36),                            /* segment's uid - zero if vtoce is free */

    2 msl bit (9),                             /* maximum segment length in 1024 word units */
    2 csl bit (9),                             /* current segment length - in 1024 word units */
    2 records bit (9),                         /* number of records used by the seg in second storage */
    2 pad2 bit (9),

    2 dtu bit (36),                            /* date and time segment was last used */

    2 dtm bit (36),                            /* date and time segment was last modified */

    2 nqsw bit (1),                            /* no quota switch - no checking for pages of this seg */
    2 deciduous bit (1),                       /* true if hc_sdw */
    2 nid bit (1),                             /* no incremental dump switch */
    2 dnzp bit (1),                            /* Dont null zero pages */
    2 gtpd bit (1),                            /* Global transparent paging device */
    2 per_process bit (1),                     /* Per process segment (deleted every bootload) */
    2 damaged bit (1),                         /* TRUE if contents damaged */
    2 fm_damaged bit (1),                      /* TRUE if filemap checksum bad */
    2 fm_checksum_valid bit (1),               /* TRUE if the checksum has been computed */
    2 synchronized bit (1),                    /* TRUE if this is a data management synchronized segment */
    2 pad3 bit (8),
    2 dirsw bit (1),                           /* directory switch */
    2 master_dir bit (1),                      /* master directory - a root for the logical volume */
    2 pad4 bit (16)) unaligned,                /* not used */

    2 fm_checksum bit (36) aligned,            /* Checksum of used portion of file map */

    (2 quota (0:1) fixed bin (18) unsigned,    /* sec storage quota - (0) for non dir pages */

    2 used (0:1) fixed bin (18) unsigned,      /* sec storage used  - (0) for non dir pages */

    2 received (0:1) fixed bin (18) unsigned,  /* total amount of storage this dir has received */

    2 trp (0:1) fixed bin (71),                /* time record product - (0) for non dir pages */

    2 trp_time (0:1) bit (36),                 /* time time_record_product was last calculated */
```