<table>
<thead>
<tr>
<th>Monday</th>
<th>Tuesday</th>
<th>Wednesday</th>
<th>Thursday</th>
<th>Friday</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intro</td>
<td>Using PL/I listings</td>
<td>CPU Registers</td>
<td>Workshop</td>
<td>Recovery procedures</td>
</tr>
<tr>
<td>What is a crash?</td>
<td>Misc. commands</td>
<td>Data Formats</td>
<td>HREGS Logs</td>
<td>Misc</td>
</tr>
<tr>
<td>What is a dump?</td>
<td>Workshop</td>
<td>ALM Intro.</td>
<td>Locking Workshop</td>
<td></td>
</tr>
<tr>
<td>Fault/Int intro.</td>
<td>analyze multics</td>
<td>Fault processing</td>
<td>AMs</td>
<td>Workshop</td>
</tr>
<tr>
<td>Afternoon</td>
<td>Workshop</td>
<td>signalling, crawlouts</td>
<td>Methodology</td>
<td></td>
</tr>
<tr>
<td>How Mx crashes</td>
<td>Stacks &amp; linkage</td>
<td>Types of Faults</td>
<td>Workshop</td>
<td></td>
</tr>
</tbody>
</table>

F88, i-1
## F88 Notebook Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dump Contents</td>
</tr>
<tr>
<td>2</td>
<td>Fault Vector</td>
</tr>
<tr>
<td>3</td>
<td>Crash Sequences</td>
</tr>
<tr>
<td>4</td>
<td>Stacks and Linkage Sections</td>
</tr>
<tr>
<td>5</td>
<td>Example PL/I Listing</td>
</tr>
<tr>
<td>6</td>
<td>analyze_multics Example</td>
</tr>
<tr>
<td>7</td>
<td>analyze_multics Documentation</td>
</tr>
<tr>
<td>8</td>
<td>Data Formats</td>
</tr>
<tr>
<td>9</td>
<td>Signalling and Crawlouts</td>
</tr>
<tr>
<td>10</td>
<td>Machine Condition Examples</td>
</tr>
<tr>
<td>11</td>
<td>Interrupt Types</td>
</tr>
<tr>
<td>12</td>
<td>Dump Analysis Methodology</td>
</tr>
</tbody>
</table>
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

---

<table>
<thead>
<tr>
<th>Header</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time, ERFNO, List of Segs</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dump Machine Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>010886.2303.0.135</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Contents of Segments</th>
</tr>
</thead>
<tbody>
<tr>
<td>010886.2303.1.135</td>
</tr>
<tr>
<td>010886.2303.2.135</td>
</tr>
<tr>
<td>010886.2303.3.135</td>
</tr>
<tr>
<td>010886.2303.4.135</td>
</tr>
</tbody>
</table>

---

F88, 1-1
Process 0, Initializer.SysDaemon.z, running (-inzr hc pp)

<table>
<thead>
<tr>
<th>Seg#</th>
<th>Len</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3072</td>
<td>dseg</td>
</tr>
<tr>
<td>1</td>
<td>1024</td>
<td>bos_toehold</td>
</tr>
<tr>
<td>2</td>
<td>4096</td>
<td>config_deck</td>
</tr>
<tr>
<td>3</td>
<td>2048</td>
<td>dn355_mailbox</td>
</tr>
<tr>
<td>4</td>
<td>1024</td>
<td>fault_vector</td>
</tr>
<tr>
<td>5</td>
<td>1024</td>
<td>flagbox</td>
</tr>
<tr>
<td>6</td>
<td>9216</td>
<td>name_table</td>
</tr>
<tr>
<td>7</td>
<td>2048</td>
<td>slt</td>
</tr>
<tr>
<td>10</td>
<td>2048</td>
<td>toehold_data</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>iom_mailbox</td>
</tr>
<tr>
<td>12</td>
<td>1024</td>
<td>unpaged_page_tables</td>
</tr>
<tr>
<td>13</td>
<td>2048</td>
<td>toehold</td>
</tr>
<tr>
<td>14</td>
<td>1024</td>
<td>breakpoint_page</td>
</tr>
<tr>
<td>15</td>
<td>1024</td>
<td>lot</td>
</tr>
<tr>
<td>16</td>
<td>17408</td>
<td>as_linkage</td>
</tr>
<tr>
<td>17</td>
<td>5120</td>
<td>ws_linkage</td>
</tr>
<tr>
<td>20</td>
<td>79872</td>
<td>definitions</td>
</tr>
<tr>
<td>22</td>
<td>0</td>
<td>abs_segl</td>
</tr>
<tr>
<td>25</td>
<td>2048</td>
<td>bound_hc_data_wired</td>
</tr>
<tr>
<td>52</td>
<td>50176</td>
<td>core_map</td>
</tr>
<tr>
<td>54</td>
<td>1024</td>
<td>disk_post_queue_seg</td>
</tr>
<tr>
<td>55</td>
<td>29696</td>
<td>disk_seg</td>
</tr>
<tr>
<td>56</td>
<td>1024</td>
<td>dn355_data</td>
</tr>
<tr>
<td>60</td>
<td>2048</td>
<td>emergency_shutdown</td>
</tr>
<tr>
<td>62</td>
<td>5120</td>
<td>idle_dsegs</td>
</tr>
<tr>
<td>63</td>
<td>5120</td>
<td>idle_pdses</td>
</tr>
<tr>
<td>64</td>
<td>2048</td>
<td>init_processor</td>
</tr>
<tr>
<td>65</td>
<td>8192</td>
<td>inzr_stk0</td>
</tr>
<tr>
<td>66</td>
<td>18432</td>
<td>ioi_abs_seg</td>
</tr>
<tr>
<td>67</td>
<td>6144</td>
<td>io_config_data</td>
</tr>
<tr>
<td>70</td>
<td>11264</td>
<td>ioi_data</td>
</tr>
<tr>
<td>71</td>
<td>3072</td>
<td>iom_data</td>
</tr>
<tr>
<td>72</td>
<td>8192</td>
<td>kst_seg</td>
</tr>
<tr>
<td>73</td>
<td>1024</td>
<td>oc_data</td>
</tr>
<tr>
<td>74</td>
<td>2048</td>
<td>pds</td>
</tr>
<tr>
<td>75</td>
<td>7168</td>
<td>prds</td>
</tr>
<tr>
<td>76</td>
<td>3072</td>
<td>pvt</td>
</tr>
<tr>
<td>77</td>
<td>0</td>
<td>rdisk_seg</td>
</tr>
<tr>
<td>102</td>
<td>27648</td>
<td>scas</td>
</tr>
<tr>
<td>104</td>
<td>55296</td>
<td>sst_names</td>
</tr>
<tr>
<td>105</td>
<td>256000</td>
<td>sst_seg</td>
</tr>
<tr>
<td>106</td>
<td>1024</td>
<td>stack_0_data</td>
</tr>
<tr>
<td>107</td>
<td>10240</td>
<td>stock_seg</td>
</tr>
<tr>
<td>110</td>
<td>1024</td>
<td>sys_info</td>
</tr>
<tr>
<td>111</td>
<td>1024</td>
<td>syserr_data</td>
</tr>
<tr>
<td>112</td>
<td>1024</td>
<td>syserr_log_data</td>
</tr>
<tr>
<td>113</td>
<td>64512</td>
<td>syserr_log_laurel</td>
</tr>
<tr>
<td>114</td>
<td>10240</td>
<td>syserr_log_hardy</td>
</tr>
<tr>
<td>115</td>
<td>18432</td>
<td>tc_data</td>
</tr>
<tr>
<td>116</td>
<td>1024</td>
<td>active_all_rings_data</td>
</tr>
</tbody>
</table>
117 3072 active_hardcore_data
121 9216 ast_lock_meter_segm
143 34816 dbm_segm
144 3072 dir_lock_segm
146 2048 dm_journal_segm
147 0 fnp_dump_segm
160 3072 io_page_table_segm
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_segm
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_segm
221 13312 >pdd> zzzzzzzbBBB BBBB>stack_1
224 19456 >pdd> zzzzzzzbBBB BBBB>stack_4
231 14336 >pdd
233 2048 >pdd> zzzzzzzbBBB BBBB
273 14336 >pdd> zzzzzzzbBBB BBBB> BBBJJMMkNPCqj.area.linker
323 46080 >pdd> zzzzzzzbBBB BBBB> BBBJJMMkPBJmHx.area.linker
331 4096 >pdd> zzzzzzzbBBB BBBB> BBBJJMMkPFQWgB
354 1024 >pdd> zzzzzzzbBBB BBBB>pit_temp
457 11264 >pdd> zzzzzzzbBBB BBBB> BBBJJMMkWJXXf.area.ipc
465 0 >pdd> zzzzzzzbBBB BBBB> BBBJJMMkWQkM.temp.0465
520 2048 >pdd> BLDXhpBBB BBBB
563 2048 >pdd> BMDXhxBBB BBBB
565 1024 >pdd> BMBDXhzBBB BBBB

2014 2048 >pdd> BxBDxbFbBBB BBBB
2015 2048 >pdd> CZLDbcdBBB BBBB
2017 2048 >pdd> CXbDdbnbBBB BBBB
2023 2048 >pdd> BmLDbcJBBB BBBB

F88, 1-3
Process 1, Idle.SysControl.a, ready (-elig hc stk)

<table>
<thead>
<tr>
<th>Seg#</th>
<th>Len</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1024</td>
<td>dseg</td>
</tr>
<tr>
<td>72</td>
<td>8192</td>
<td>kst_seg</td>
</tr>
<tr>
<td>74</td>
<td>1024</td>
<td>pds</td>
</tr>
<tr>
<td>75</td>
<td>8192</td>
<td>prds</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Seg#</th>
<th>Len</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1024</td>
<td>dseg</td>
</tr>
<tr>
<td>72</td>
<td>8192</td>
<td>kst_seg</td>
</tr>
<tr>
<td>74</td>
<td>1024</td>
<td>pds</td>
</tr>
<tr>
<td>75</td>
<td>7168</td>
<td>prds</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Seg#</th>
<th>Len</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1024</td>
<td>dseg</td>
</tr>
<tr>
<td>72</td>
<td>8192</td>
<td>kst_seg</td>
</tr>
<tr>
<td>74</td>
<td>1024</td>
<td>pds</td>
</tr>
<tr>
<td>75</td>
<td>3072</td>
<td>prds</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Seg#</th>
<th>Len</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1024</td>
<td>dseg</td>
</tr>
<tr>
<td>72</td>
<td>8192</td>
<td>kst_seg</td>
</tr>
<tr>
<td>74</td>
<td>1024</td>
<td>pds</td>
</tr>
<tr>
<td>75</td>
<td>7168</td>
<td>prds</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Seg#</th>
<th>Len</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1024</td>
<td>dseg</td>
</tr>
<tr>
<td>72</td>
<td>8192</td>
<td>kst_seg</td>
</tr>
<tr>
<td>74</td>
<td>1024</td>
<td>pds</td>
</tr>
<tr>
<td>75</td>
<td>7168</td>
<td>prds</td>
</tr>
</tbody>
</table>
Process 6, Demers.Flower.a, running (-run hc pp moddir)

<table>
<thead>
<tr>
<th>Seg#</th>
<th>Len</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1024</td>
<td>dseg</td>
</tr>
<tr>
<td>72</td>
<td>1024</td>
<td>kst_seg</td>
</tr>
<tr>
<td>74</td>
<td>2048</td>
<td>pds</td>
</tr>
<tr>
<td>75</td>
<td>8192</td>
<td>prds</td>
</tr>
<tr>
<td>220</td>
<td>11264</td>
<td>&gt;s11&gt;stack_0.035</td>
</tr>
<tr>
<td>221</td>
<td>3072</td>
<td>&gt;pdd&gt; BLbDbdKBBBBBBB&gt;stack_1</td>
</tr>
<tr>
<td>222</td>
<td>3072</td>
<td>&gt;pdd&gt; BLbDbdKBBBBBBB&gt;stack_2</td>
</tr>
<tr>
<td>224</td>
<td>7168</td>
<td>&gt;pdd&gt; BLbDbdKBBBBBBB&gt;stack_4</td>
</tr>
<tr>
<td>237</td>
<td>10240</td>
<td>&gt;pdd&gt; BLbDbdKBBBBBBB&gt; BBBJPNBqbbZbGF.area.linker</td>
</tr>
<tr>
<td>254</td>
<td>1024</td>
<td>&gt;pdd&gt; BLbDbdKBBBBBBB&gt;pit</td>
</tr>
<tr>
<td>317</td>
<td>2048</td>
<td>&gt;pdd&gt; BLbDbdKBBBBBBB&gt;process_search_segment_.4</td>
</tr>
<tr>
<td>330</td>
<td>5120</td>
<td>&gt;pdd&gt; BLbDbdKBBBBBBB&gt; BBBJPNBqbjdQZd.area.linker</td>
</tr>
<tr>
<td>332</td>
<td>5120</td>
<td>&gt;pdd&gt; BLbDbdKBBBBBBB&gt; BBBJPNBqbkFBCb.area.linker</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Seg#</th>
<th>Len</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1024</td>
<td>dseg</td>
</tr>
<tr>
<td>72</td>
<td>2048</td>
<td>kst_seg</td>
</tr>
<tr>
<td>74</td>
<td>2048</td>
<td>pds</td>
</tr>
<tr>
<td>75</td>
<td>7168</td>
<td>prds</td>
</tr>
<tr>
<td>220</td>
<td>12288</td>
<td>&gt;s11&gt;stack_0.016</td>
</tr>
<tr>
<td>224</td>
<td>38912</td>
<td>&gt;pdd&gt; BdLDbbjBBBBBBB&gt;stack_4</td>
</tr>
<tr>
<td>237</td>
<td>19148</td>
<td>&gt;pdd&gt; BdLDbbjBBBBBBB&gt; BBBJPNBmbfjNjK.area.linker</td>
</tr>
<tr>
<td>304</td>
<td>5120</td>
<td>&gt;pdd&gt; BdLDbbjBBBBBBB&gt; BBBJPNBmfgJMCb.temp.0304</td>
</tr>
<tr>
<td>332</td>
<td>6144</td>
<td>&gt;pdd&gt; BdLDbbjBBBBBBB&gt; BBBJPNBmfkWPFh.area.linker</td>
</tr>
<tr>
<td>340</td>
<td>5120</td>
<td>&gt;pdd&gt; BdLDbbjBBBBBBB&gt; BBBJPNBmlwdLH.area.linker</td>
</tr>
<tr>
<td>347</td>
<td>2048</td>
<td>&gt;pdd&gt; BdLDbbjBBBBBBB&gt;process_search_segment_.4</td>
</tr>
<tr>
<td>361</td>
<td>1024</td>
<td>&gt;pdd&gt; BdLDbbjBBBBBBB&gt; BBBJPNBmpfWzmP.temp.0361</td>
</tr>
<tr>
<td>431</td>
<td>1024</td>
<td>&gt;pdd&gt; BdLDbbjBBBBBBB&gt; BBBJPNBmgFcGQh.temp.0431</td>
</tr>
<tr>
<td>432</td>
<td>1024</td>
<td>&gt;pdd&gt; BdLDbbjBBBBBBB&gt; BBBJPNBmgQmfch.temp.0432</td>
</tr>
<tr>
<td>433</td>
<td>0</td>
<td>&gt;pdd&gt; BdLDbbjBBBBBBB&gt; BBBJPNBmgpLfKD.temp.0433</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Seg#</th>
<th>Len</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1024</td>
<td>dseg</td>
</tr>
<tr>
<td>72</td>
<td>2048</td>
<td>kst_seg</td>
</tr>
<tr>
<td>74</td>
<td>2048</td>
<td>pds</td>
</tr>
<tr>
<td>75</td>
<td>7168</td>
<td>prds</td>
</tr>
<tr>
<td>220</td>
<td>10240</td>
<td>&gt;s11&gt;stack_0.022</td>
</tr>
<tr>
<td>221</td>
<td>3072</td>
<td>&gt;pdd&gt; BkBDbcWbBBBBBBB&gt;stack_1</td>
</tr>
<tr>
<td>222</td>
<td>4096</td>
<td>&gt;pdd&gt; BkBDbcWbBBBBBBB&gt;stack_2</td>
</tr>
<tr>
<td>224</td>
<td>41984</td>
<td>&gt;pdd&gt; BkBDbcWbBBBBBBB&gt;stack_4</td>
</tr>
</tbody>
</table>

F88, 1-5
BCE dump Command

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
occurrence 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

F88, 1-7
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

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

<table>
<thead>
<tr>
<th>#</th>
<th>Fault Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>Access Violation</td>
</tr>
<tr>
<td>5</td>
<td>Command</td>
</tr>
<tr>
<td>6</td>
<td>Derail</td>
</tr>
<tr>
<td>2</td>
<td>MME 1</td>
</tr>
<tr>
<td>3</td>
<td>Fault Tag 1</td>
</tr>
<tr>
<td>7</td>
<td>Lockup</td>
</tr>
<tr>
<td>10</td>
<td>Illegal Procedure</td>
</tr>
<tr>
<td>13</td>
<td>Overflow</td>
</tr>
<tr>
<td>14</td>
<td>Divide Check</td>
</tr>
<tr>
<td>21</td>
<td>MME 2</td>
</tr>
<tr>
<td>22</td>
<td>MME 3</td>
</tr>
<tr>
<td>23</td>
<td>MME 4</td>
</tr>
<tr>
<td>25</td>
<td>Fault Tag 3</td>
</tr>
</tbody>
</table>
## Fault Vector in Numerical Order

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

**Fault Vector in Numerical Order**
### Faults That Always Crash System

<table>
<thead>
<tr>
<th>#</th>
<th>Fault Name</th>
<th>SCU stored at</th>
<th>Handler</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>Unassigned</td>
<td>scu 564,* -&gt; prds=sys_trouble_data (71240)</td>
<td>tra 464,* -&gt; wired_fim$unexp_fault (342310)</td>
</tr>
<tr>
<td>27</td>
<td>Unassigned</td>
<td>scu 566,* -&gt; prds=sys_trouble_data (71240)</td>
<td>tra 466,* -&gt; wired_fim$unexp_fault (342310)</td>
</tr>
<tr>
<td>28</td>
<td>Unassigned</td>
<td>scu 570,* -&gt; prds=sys_trouble_data (71240)</td>
<td>tra 470,* -&gt; wired_fim$unexp_fault (342310)</td>
</tr>
<tr>
<td>29</td>
<td>Unassigned</td>
<td>scu 572,* -&gt; prds=sys_trouble_data (71240)</td>
<td>tra 472,* -&gt; wired_fim$unexp_fault (342310)</td>
</tr>
<tr>
<td>30</td>
<td>Unassigned</td>
<td>scu 574,* -&gt; prds=sys_trouble_data (71240)</td>
<td>tra 474,* -&gt; wired_fim$unexp_fault (342310)</td>
</tr>
<tr>
<td>15</td>
<td>Execute</td>
<td>scu 536,* -&gt; prds=sys_trouble_data (71240)</td>
<td>tra 436,* -&gt; wired_fim$exec_fault (342274)</td>
</tr>
</tbody>
</table>

### Faults Used Internally by Supervisor

<table>
<thead>
<tr>
<th>#</th>
<th>Fault Name</th>
<th>SCU stored at</th>
<th>Handler</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Timer Runout</td>
<td>scu 510,* -&gt; prds$timer_data (71160)</td>
<td>tra 410,* -&gt; wired_fim$timer_runout (342324)</td>
</tr>
<tr>
<td>8</td>
<td>Connect</td>
<td>scu 520,* -&gt; prds$timer_data (71160)</td>
<td>tra 420,* -&gt; prds$fast_connect_code (711054)</td>
</tr>
<tr>
<td>20</td>
<td>Access Violation</td>
<td>scu 550,* -&gt; prds$timer_data (7060)</td>
<td>tra 450,* -&gt; fim$accessViolation_entry (340)</td>
</tr>
</tbody>
</table>

### Implicit Requests for Supervisor Services

<table>
<thead>
<tr>
<th>#</th>
<th>Fault Name</th>
<th>SCU stored at</th>
<th>Handler</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>(DPI) Page</td>
<td>scu 542,* -&gt; prds$page_fault_data (700)</td>
<td>tra 442,* -&gt; page_fault$page_fault (441062)</td>
</tr>
<tr>
<td>16</td>
<td>(DF0) Segment</td>
<td>scu 540,* -&gt; prds$page_fault_data (7060)</td>
<td>tra 440,* -&gt; fim$primary_fault_entry (34404)</td>
</tr>
<tr>
<td>24</td>
<td>(FT2) Linkage</td>
<td>scu 560,* -&gt; prds$page_fault_data (7060)</td>
<td>tra 460,* -&gt; fim$primary_fault_entry (34404)</td>
</tr>
<tr>
<td>20</td>
<td>Access Violation</td>
<td>scu 550,* -&gt; prds$page_fault_data (7060)</td>
<td>tra 450,* -&gt; fim$accessViolation_entry (340)</td>
</tr>
</tbody>
</table>

### Faults That Always Indicate Hardware Problems

<table>
<thead>
<tr>
<th>#</th>
<th>Fault Name</th>
<th>SCU stored at</th>
<th>Handler</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Shutdown</td>
<td>scu 500,* -&gt; prds$signal_data (7060)</td>
<td>tra 400,* -&gt; fim$onc_start_shut_entry (3414)</td>
</tr>
<tr>
<td>11</td>
<td>Op Not Complete</td>
<td>scu 526,* -&gt; prds$signal_data (7060)</td>
<td>tra 426,* -&gt; fim$onc_start_shut_entry (3414)</td>
</tr>
<tr>
<td>12</td>
<td>Startup</td>
<td>scu 530,* -&gt; prds$signal_data (7060)</td>
<td>tra 430,* -&gt; fim$onc_start_shut_entry (3414)</td>
</tr>
<tr>
<td>9</td>
<td>Parity</td>
<td>scu 522,* -&gt; prds$signal_data (7060)</td>
<td>tra 422,* -&gt; fim$parity_entry (34124)</td>
</tr>
<tr>
<td>1</td>
<td>Store</td>
<td>scu 502,* -&gt; prds$signal_data (70140)</td>
<td>tra 402,* -&gt; fim$signal_entry (34300)</td>
</tr>
</tbody>
</table>

### Faults That Always Indicate Hardware or Software Problems

<table>
<thead>
<tr>
<th>#</th>
<th>Fault Name</th>
<th>SCU stored at</th>
<th>Handler</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>Trouble</td>
<td>scu 576,* -&gt; prds$signal_data (7060)</td>
<td>tra 476,* -&gt; fim$primary_fault_entry (34404)</td>
</tr>
<tr>
<td>18</td>
<td>Directed Fault 2</td>
<td>scu 544,* -&gt; prds$signal_data (70140)</td>
<td>tra 444,* -&gt; fim$signal_entry (34300)</td>
</tr>
<tr>
<td>19</td>
<td>Directed Fault 3</td>
<td>scu 546,* -&gt; prds$signal_data (70140)</td>
<td>tra 446,* -&gt; fim$signal_entry (34300)</td>
</tr>
<tr>
<td>#</td>
<td>Fault Name</td>
<td>SCU stored at</td>
<td>Handler</td>
</tr>
<tr>
<td>----</td>
<td>---------------------</td>
<td>---------------------------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>20</td>
<td>Access Violation</td>
<td>scu 550,.* -&gt; pds$signal_data (70</td>
<td>60)</td>
</tr>
<tr>
<td>5</td>
<td>Command</td>
<td>scu 512,.* -&gt; pds$signal_data (70</td>
<td>60)</td>
</tr>
<tr>
<td>6</td>
<td>Derail</td>
<td>scu 514,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
<tr>
<td>2</td>
<td>MME 1</td>
<td>scu 504,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
<tr>
<td>3</td>
<td>Fault Tag 1</td>
<td>scu 506,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
<tr>
<td>7</td>
<td>Lockup</td>
<td>scu 516,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
<tr>
<td>10</td>
<td>Illegal Procedure</td>
<td>scu 524,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
<tr>
<td>13</td>
<td>Overflow</td>
<td>scu 532,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
<tr>
<td>14</td>
<td>Divide Check</td>
<td>scu 534,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
<tr>
<td>21</td>
<td>MME 2</td>
<td>scu 552,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
<tr>
<td>22</td>
<td>MME 3</td>
<td>scu 554,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
<tr>
<td>23</td>
<td>MME 4</td>
<td>scu 556,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
<tr>
<td>25</td>
<td>Fault Tag 3</td>
<td>scu 562,.* -&gt; pds$signal_data (70</td>
<td>140)</td>
</tr>
</tbody>
</table>
## PROHIBITED FAULTS

<table>
<thead>
<tr>
<th>Code</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Page fault while on prds</td>
</tr>
<tr>
<td>2</td>
<td>Fault/Interrupt while on prds</td>
</tr>
<tr>
<td>3</td>
<td>Fault in idle process</td>
</tr>
<tr>
<td>4</td>
<td>Fault/Interrupt with PTL set</td>
</tr>
<tr>
<td>5</td>
<td>Unrecognized fault</td>
</tr>
<tr>
<td>6</td>
<td>Unexpected fault</td>
</tr>
<tr>
<td>7</td>
<td>Execute fault by operator</td>
</tr>
<tr>
<td>8</td>
<td>Out-of-Segment-Bound on prds</td>
</tr>
<tr>
<td>9</td>
<td>Fault while in masked environment</td>
</tr>
<tr>
<td>10</td>
<td>Fault while in bound_interceptors</td>
</tr>
<tr>
<td>11</td>
<td>Ring 0 derail</td>
</tr>
</tbody>
</table>
## Crash Sequence for Type 1 Crash: Prohibited Fault

<table>
<thead>
<tr>
<th>CPU A (Bootload CPU)</th>
<th>CPU B</th>
<th>CPU C</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>&lt;&lt;FAULT&gt;&gt;</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>fault_vector_scu/tra</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>handler</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>fim_util$check_fault</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>scs$sys_trouble_pending=-N</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>scs$trouble_processid-</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>PROCESSID</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>cioc &lt;CPU B&gt;</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>STOP</td>
<td></td>
<td>.</td>
</tr>
</tbody>
</table>

F88, 3-2
Crash Sequence for Type 1 Crash: Prohibited Fault (continued)

<table>
<thead>
<tr>
<th>CPU A (Bootload CPU)</th>
<th>CPU B</th>
<th>CPU C</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;&lt;CONNECT from CPU B &gt;&gt;</td>
<td>prds$fast_connect_code</td>
<td></td>
</tr>
<tr>
<td></td>
<td>test scs$sys_trouble_pending</td>
<td></td>
</tr>
<tr>
<td></td>
<td>wired_fim$connect_handler</td>
<td></td>
</tr>
<tr>
<td></td>
<td>store MCs in prds$fim_data</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sys_trouble</td>
<td></td>
</tr>
<tr>
<td></td>
<td>if scs$trouble_flags = 0:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>scs$trouble_flags =</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scs$processor =</td>
</tr>
<tr>
<td></td>
<td></td>
<td>cio&lt; &lt;all other CPUs&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>copy MCs from prds$fim_data to prds$sys_trouble_data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scs$processor bit 1 = 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>scs$trouble_dbrs(1) = DBR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>test scs$bos_processor_tag</td>
</tr>
<tr>
<td></td>
<td></td>
<td>STOP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;&lt;CONNECT from CPU B &gt;&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>prds$fast_connect_code</td>
</tr>
<tr>
<td></td>
<td></td>
<td>test scs$sys_trouble_pending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>wired_fim$connect_handler</td>
</tr>
<tr>
<td></td>
<td></td>
<td>store MCs in prds$fim_data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sys_trouble</td>
</tr>
<tr>
<td></td>
<td></td>
<td>if scs$sys_trouble_pending &lt; 0:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>make flagbox message</td>
</tr>
<tr>
<td></td>
<td></td>
<td>loop 1 second for all I/O to complete</td>
</tr>
<tr>
<td></td>
<td></td>
<td>change derail fault vector</td>
</tr>
<tr>
<td></td>
<td></td>
<td>execute drf instruction</td>
</tr>
<tr>
<td></td>
<td></td>
<td>BOS/BCE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>print flagbox message</td>
</tr>
</tbody>
</table>

F88, 3-3
## Typical Dump Events for Type 1 Crash: Prohibited Fault

<table>
<thead>
<tr>
<th>Event</th>
<th>CPU</th>
<th>MC location</th>
<th>Context/significance of MCs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRL</td>
<td>Bootload</td>
<td>Dump Header (DREGS)</td>
<td>sys_trouble</td>
</tr>
<tr>
<td>CON</td>
<td>Non-trouble</td>
<td>prds$fim_data &amp; prds$sys_trouble_data</td>
<td>Where executing when told to stop</td>
</tr>
<tr>
<td>CON</td>
<td>Non-trouble</td>
<td>prds$fim_data &amp; prds$sys_trouble_data</td>
<td>Where executing when told to stop</td>
</tr>
<tr>
<td>CON</td>
<td>Trouble</td>
<td>prds$fim_data &amp; prds$sys_trouble_data</td>
<td>fim_util</td>
</tr>
<tr>
<td>Fault</td>
<td>Trouble</td>
<td>PR2 in prds$sys_trouble_data</td>
<td>Prohibited fault that caused crash</td>
</tr>
</tbody>
</table>
Crash Sequence for Type 2 Crash: Syserr

<table>
<thead>
<tr>
<th>CPU A (Bootload CPU)</th>
<th>CPU B</th>
<th>CPU C</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>hardcore_program</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>call syserr with code 1</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>syserr</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>make sure stack is wired</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>syserr_real</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>print message on console</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>put message in syserr log</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>pmutSbce_and_return</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>scs$sys_trouble_pending=</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>scs$processor</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>scs$trouble_processid=</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>PROCESSID</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>cioc &lt;CPU B&gt;</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>STOP</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>
### Crash Sequence for Type 2 Crash: Syserr (continued)

<table>
<thead>
<tr>
<th>CPU A (Bootload CPU)</th>
<th>CPU B</th>
<th>CPU C</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>***&lt;CONNECT from CPU B&gt;***</td>
<td></td>
</tr>
<tr>
<td></td>
<td>prds$fast_connect_code</td>
<td></td>
</tr>
<tr>
<td></td>
<td>test scs$sys_trouble_pending</td>
<td></td>
</tr>
<tr>
<td></td>
<td>wired_fim$connect_handler</td>
<td></td>
</tr>
<tr>
<td></td>
<td>store MCs in prds$fim_data</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sys_trouble</td>
<td></td>
</tr>
<tr>
<td></td>
<td>if scs$trouble_flags = 0:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>scs$trouble_flags =</td>
<td></td>
</tr>
<tr>
<td></td>
<td>scs$processor</td>
<td></td>
</tr>
<tr>
<td></td>
<td>cioc &lt;all other CPUs&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>copy MCs from prds$fim_data</td>
<td></td>
</tr>
<tr>
<td></td>
<td>to prds$sys_trouble_data</td>
<td></td>
</tr>
<tr>
<td></td>
<td>scs$processor bit 1 = 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>scs$trouble_dbrs(1) = DBR</td>
<td></td>
</tr>
<tr>
<td></td>
<td>test scs$bo;_processor_tag</td>
<td></td>
</tr>
<tr>
<td></td>
<td>STOP</td>
<td></td>
</tr>
</tbody>
</table>

<<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
test scs$trouble_flags
copy MCs from prds$fim_data
to prds$sys_trouble_data
scs$processor bit 0 = 0
scs$trouble_dbrs(0) = DBR
test scs$bo;_processor_tag
inhibit lockup fault
loop until scs$processor=0

| \*\*\*<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 |       |
| test scs$trouble_flags |       |
| copy MCs from prds$fim_data |       |
| to prds$sys_trouble_data |       |
| scs$processor bit 2 = 0 |       |
| scs$trouble_dbrs(2) = DBR |       |
| test scs$bo;_processor_tag |       |
| STOP |       |

BOS/BCX
## Typical Dump Events for Type 2 Crash: Syserr

<table>
<thead>
<tr>
<th>Event</th>
<th>CPU</th>
<th>MC location</th>
<th>Context/significance of MCs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRL</td>
<td>Bootload</td>
<td>Dump Header (DREGS)</td>
<td>sys_trouble</td>
</tr>
<tr>
<td>CON</td>
<td>Non-trouble</td>
<td>prds$fim_data &amp; prds$sys_trouble_data</td>
<td>Where executing when told to stop</td>
</tr>
<tr>
<td>CON</td>
<td>Non-trouble</td>
<td>prds$fim_data &amp; prds$sys_trouble_data</td>
<td>Where executing when told to stop</td>
</tr>
<tr>
<td>CON</td>
<td>Trouble</td>
<td>prds$fim_data &amp; prds$sys_trouble_data</td>
<td>pmut</td>
</tr>
<tr>
<td>Syserr Message</td>
<td></td>
<td></td>
<td>Multics not in operation</td>
</tr>
<tr>
<td>Syserr Message</td>
<td></td>
<td></td>
<td>Message from syserr call that crashed the system</td>
</tr>
</tbody>
</table>
Crash Sequence for Type 3 Crash: EXECUTE Fault

CPU A (Bootload CPU)          CPU B          CPU C

.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    .
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                    ..
.                                    .                                   ..

<<CONNECT from CPU B>>

prds$fast_connect_code

wired_fim$connect_handler

store MCs in prds$fim_data

sys_trouble

test scs$sys_trouble_pending

copy MCs from prds$fim_data
to prds$sys_trouble_data

scs$processor bit 0 = 0

scs$trouble_dbrs(0) = DBR

test scs$bos_processor_tag

inhibit lockup fault

loop until scs$processor = 0

| | | | |

| | | | |

test scs$sys_trouble_pending

make flagbox message

loop 1 second for all I/O
to complete

change derail fault vector

execute drl instruction

BOS/RCE

print flagbox message

f88, 3-8
Typical Dump Events for Type 3 Crash: EXECUTE Fault

DUMP EVENTS SEQUENCE:

<table>
<thead>
<tr>
<th>Event</th>
<th>CPU</th>
<th>MC location</th>
<th>Context/significance of MCs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRL</td>
<td>Bootload</td>
<td>Dump Header (DREGS)</td>
<td>sys_trouble|221</td>
</tr>
<tr>
<td>CON</td>
<td>Others</td>
<td>prds$fim_data &amp; prds$sys_trouble_data</td>
<td>Where looping when told to stop; PR6 -&gt; stack frame</td>
</tr>
<tr>
<td>CON</td>
<td>Others</td>
<td>prds$fim_data &amp; prds$sys_trouble_data</td>
<td>Where looping when told to stop; PR6 -&gt; stack frame</td>
</tr>
<tr>
<td>EXF</td>
<td>Button pushed</td>
<td>prds$sys_trouble_data</td>
<td>Where looping when EXF done; PR6 -&gt; stack frame</td>
</tr>
</tbody>
</table>
Crash Sequence for Type 4 Crash: EXECUTE Switches

<table>
<thead>
<tr>
<th>CPU A (Bootload CPU)</th>
<th>CPU B</th>
<th>CPU C</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>&lt;&lt;MANUAL HALT&gt;&gt;</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>&lt;&lt;EXECUTE SWITCHES&gt;&gt;</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>(XED 10000)</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>240cc</td>
<td></td>
<td>.</td>
</tr>
</tbody>
</table>

**BOS/BCE**
print MANUAL RETURN TO BOS
Typical Dump Events for Type 4 Crash: EXECUTE Switches

DUMP EVENTS SEQUENCE:

<table>
<thead>
<tr>
<th>Event</th>
<th>CPU</th>
<th>MC location</th>
<th>Context/significance of MCs</th>
</tr>
</thead>
<tbody>
<tr>
<td>XED</td>
<td>Bootload</td>
<td>Dump Header (DREGS)</td>
<td>PR6 -&gt; stack frame of looping program</td>
</tr>
</tbody>
</table>
Stack, etc. Before Call to B$B2

Object Seg A

[unique].area.linker

KST/DSEG

TEXT

74 epp2 pr4 20,*
75 tsp4 call_op

DEFS

link 20 B$B2
text 12 A

LINKAGE

20 7777700000046
21 0000020000000

SYMBOL

Object Seg B

TEXT

DEFS

text 22 B
text 64 B2

LINKAGE

SYMBOL

[unique].area.linker

(Text continued on the next page)

F88,
Stack, etc. After Call to BSB2

(unique).area.linker

+----> TEXT
  
  *app2 ptr|20,*
  tapp call_op
  
  DEFS
  
  link|20 B4B2
  text|12 A
  
  LINKAGE
  
  SYMBOL

+----> B's LINKAGE

+----> 20 777770000046
  21 000020000000

+----> 21 000640000000

+----> 20 000425000043

+----> A's LINKAGE

+----> A's FRAME
  retptr=400 76
  entptr=400 20
  
+----> B's OLD FRAME
  retptr=????
  entptr=425|64

+----> 400

+----> 425

+----> HEADER

+----> init_proce
**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

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

example: proc;

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     do autovar = 1 to 1000000;
18        call intproc (intstatvar);
19        if divide (intstatvar, 7, 0, 0) * 7 = intstatvar then
20           basedstruct.element2 =
21              basedstruct.element2 + constant + intstatvar;
22        end;
23        call ioa_ ("autovar = "^d, element2 = "^d, intstatvar = "^d",
24            autovar, basedstruct.element2, intstatvar);
25        return;
26     end example;

intproc: proc (parm);
29 dcl parm fixed bin (35);
30 dcl i fixed bin (35);
31     i = parm;
32     parm = i + mod (i, 6) + 1;
33 end intproc;
34
end example;
```

F88, 5-1
<table>
<thead>
<tr>
<th>SOURCE FILES USED IN THIS COMPILATION.</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINE</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>
NAMES DECLARED IN THIS COMPILATION.

<table>
<thead>
<tr>
<th>IDENTIFIER</th>
<th>OFFSET</th>
<th>LOC</th>
<th>STORAGE CLASS</th>
<th>DATA TYPE</th>
<th>ATTRIBUTES AND REFERENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>autovar</td>
<td>000100</td>
<td></td>
<td>automatic</td>
<td>fixed bin(35,0)</td>
<td>dcl 5 set ref 18* 24*</td>
</tr>
<tr>
<td>basedstruct</td>
<td>000000</td>
<td></td>
<td>based</td>
<td>structure</td>
<td>level 1 unaligned dcl 10 set ref 14</td>
</tr>
<tr>
<td>constant</td>
<td>000000</td>
<td></td>
<td>constant</td>
<td>fixed bin(35,0)</td>
<td>initial dcl 7 ref 20</td>
</tr>
<tr>
<td>element2</td>
<td>1</td>
<td></td>
<td>fixed bin(35,0)</td>
<td></td>
<td>level 2 dcl 10 set ref 15* 20* 24*</td>
</tr>
<tr>
<td>i</td>
<td>000112</td>
<td></td>
<td>based</td>
<td>fixed bin(35,0)</td>
<td>dcl 30 set ref 32* 33 33</td>
</tr>
<tr>
<td>intstatvar</td>
<td>000010</td>
<td></td>
<td>internal static</td>
<td>fixed bin(35,0)</td>
<td>dcl 6 set ref 16* 19* 20 20 20 24*</td>
</tr>
<tr>
<td>loa</td>
<td>000014</td>
<td></td>
<td>entry</td>
<td></td>
<td>external dcl 13 ref 24</td>
</tr>
<tr>
<td>parm</td>
<td>000114</td>
<td></td>
<td>parameter</td>
<td>fixed bin(35,0)</td>
<td>dcl 29 set ref 28 32 33*</td>
</tr>
<tr>
<td>ptrvar</td>
<td>000102</td>
<td></td>
<td>pointer</td>
<td></td>
<td>dcl 9 set ref 10 10 10 14* 15 20 20 24</td>
</tr>
</tbody>
</table>

NAMES DECLARED BY DECLARE STATEMENT.

<table>
<thead>
<tr>
<th>IDENTIFIER</th>
<th>OFFSET</th>
<th>LOC</th>
<th>STORAGE CLASS</th>
<th>DATA TYPE</th>
<th>ATTRIBUTES AND REFERENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>example</td>
<td>000020</td>
<td></td>
<td>constant</td>
<td>entry</td>
<td>external dcl 3</td>
</tr>
<tr>
<td>intproc</td>
<td>000123</td>
<td></td>
<td>constant</td>
<td>entry</td>
<td>internal dcl 28 ref 19</td>
</tr>
</tbody>
</table>

NAMES DECLARED BY EXPLICIT CONTEXT.

<table>
<thead>
<tr>
<th>IDENTIFIER</th>
<th>OFFSET</th>
<th>LOC</th>
<th>STORAGE CLASS</th>
<th>DATA TYPE</th>
<th>ATTRIBUTES AND REFERENCES</th>
</tr>
</thead>
<tbody>
<tr>
<td>divide</td>
<td>000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mod</td>
<td>000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

STORAGE REQUIREMENTS FOR THIS PROGRAM.

<table>
<thead>
<tr>
<th>Object</th>
<th>Text</th>
<th>Link</th>
<th>Symbol</th>
<th>Defs</th>
<th>Static</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>0</td>
<td>0</td>
<td>174</td>
<td>212</td>
<td>141</td>
</tr>
<tr>
<td>Length</td>
<td>612</td>
<td>141</td>
<td>16</td>
<td>364</td>
<td>33</td>
</tr>
</tbody>
</table>

BLOCK NAME STACK SIZE TYPE WHY NONQUICK/WHO SHARES STACK FRAME

<table>
<thead>
<tr>
<th>IDENTIFIER</th>
<th>BLOCK NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>example</td>
<td>example</td>
</tr>
</tbody>
</table>

STORAGE FOR INTERNAL STATIC VARIABLES.

<table>
<thead>
<tr>
<th>LOC IDENTIFIER</th>
<th>BLOCK NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>000010</td>
<td>intstatvar</td>
</tr>
</tbody>
</table>

STORAGE FOR AUTOMATIC VARIABLES.

<table>
<thead>
<tr>
<th>STACK FRAME</th>
<th>LOC IDENTIFIER</th>
<th>BLOCK NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>example</td>
<td>000100 autovar</td>
<td>example</td>
</tr>
<tr>
<td></td>
<td>000102 ptrvar</td>
<td>example</td>
</tr>
<tr>
<td></td>
<td>000112 i</td>
<td>intproc</td>
</tr>
</tbody>
</table>

THE FOLLOWING EXTERNAL OPERATORS ARE USED BY THIS PROGRAM.

call_ext_out_desc  return  mod_false  ext_entry  alloc_based_storage

THE FOLLOWING EXTERNAL ENTRIES ARE CALLED BY THIS PROGRAM.

loa_
CONSTANTS
000000 aa 000000000004
0000140 aa 000000000006
0000001 aa 4040000000043
0000002 aa 5240000000054
0000003 aa 000003641100
0000004 aa 141 165 164 157 auto
0000005 aa 166 141 162 040 var
0000006 aa 073 040 136 154 = `d
0000007 aa 054 040 145 154 , e1
0000010 aa 145 155 145 156 emen
0000011 aa 164 062 040 075 t2 =
0000012 aa 040 136 144 054 `d,
0000013 aa 040 151 156 164 int
0000014 aa 163 164 141 164 stat
0000015 aa 166 141 162 040 var
0000016 aa 075 040 136 144 = `d

BEGIN PROCEDURE example
ENTRY TO example
example: proc;

allocate basedstruct;

begin item
000023 aa 00002 2360 07 ldq 2,dl
000026 aa 0 01403 7001 00 tax0 pr0|771 alloc_based_storage
000027 aa 777776 7100 04 tra -2,1c ptrvar
000030 aa 6 00102 2521 00 spr12 pr6|66

basedstruct.element2 = 27;

000031 aa 000033 2360 07 ldq 27,di
000032 aa 2 00001 7561 00 stq pr2|1 basedstruct.element2

intstatvar = 0;

000033 aa 6 00044 3701 20 epp4 pr6|36,* intstatvar
000034 ia 4 00010 4501 00 stz pr4|8

do autovar = 1 to 1000000;

000035 aa 000001 2360 07 ldq 1,dl
call intproc (intstatvar);

if divide (intstatvar, 7, 0) * 7 = intstatvar then
  basedstruct.element2 =
  basedstruct.element2 + constant + intstatvar;
end;

call loa_ ("autovar = &d, element2 = &d, intstatvar = &d", autovar, basedstruct.element2, intstatvar);
BEGIN PROCEDURE intproc
ENTRY TO intproc
intproc: proc (parm);

i = parm;

parm = i + mod (i, 6) + 1;

end intproc;

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.
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 fault
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

<table>
<thead>
<tr>
<th>Time</th>
<th>CPU</th>
<th>Proc</th>
<th>Event</th>
<th>Circumstances</th>
</tr>
</thead>
<tbody>
<tr>
<td>47.870284</td>
<td>d</td>
<td>Fault:</td>
<td>DRL</td>
<td>RTB Machine Conditions</td>
</tr>
<tr>
<td>43.436614</td>
<td>a</td>
<td>8</td>
<td>Fault: CON</td>
<td>prds$sys_trouble_data</td>
</tr>
<tr>
<td>.436614</td>
<td>b</td>
<td>10</td>
<td>Fault: CON</td>
<td>prds$sys_trouble_data</td>
</tr>
<tr>
<td>.436516</td>
<td>d</td>
<td>6</td>
<td>Fault: CON</td>
<td>prds$sys_trouble_data</td>
</tr>
<tr>
<td>.436383</td>
<td>c</td>
<td>0</td>
<td>Fault: CON</td>
<td>prds$sys_trouble_data</td>
</tr>
<tr>
<td>.435683</td>
<td>d</td>
<td>6</td>
<td>Interrupt: IOM A, Level 3</td>
<td>prds$interrupt_data</td>
</tr>
</tbody>
</table>
.433229 a 8 APTE at 5300 changed to Running
.433041 a 12 APTE at 10500 changed to Waiting for 000000231167
.429753 Connect to A 36
.426543 a 15 APTE at 11700 changed to Ready
.423070 a 11 APTE at 10300 changed to Waiting for 000000542415
.422441 Connect to A 29
.418303 a 7 APTE at 5200 changed to Waiting for 00000032124
.418192 Connect to B 28
.417694 Connect to A 20
.416592 a 7 Fault: DF1 pds$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 2000000 to 2377777
Mem C 4000000 to 5777777

azm: tcq
ELIGIBLE QUEUE:

<table>
<thead>
<tr>
<th>Proc</th>
<th>DBR</th>
<th>State</th>
<th>Process ID</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>06060114</td>
<td>waiting</td>
<td>003600666666</td>
<td>SyserrLogger.SysDaemon.z</td>
</tr>
<tr>
<td>0</td>
<td>06234350</td>
<td>running</td>
<td>003000777777</td>
<td>c Initializer.SysDaemon.z</td>
</tr>
<tr>
<td>14</td>
<td>06174074</td>
<td>waiting</td>
<td>011600105645</td>
<td>JNye.CSDUK.a</td>
</tr>
<tr>
<td>7</td>
<td>06154234</td>
<td>waiting</td>
<td>005200105632</td>
<td>Dist2.VIS.a</td>
</tr>
<tr>
<td>8</td>
<td>06154514</td>
<td>running</td>
<td>005300105633</td>
<td>a Dumper.SysDaemon.z</td>
</tr>
<tr>
<td>15</td>
<td>06200024</td>
<td>ready</td>
<td>011700105646</td>
<td>Backup.SysDaemon.z</td>
</tr>
<tr>
<td>10</td>
<td>06157174</td>
<td>running</td>
<td>010200105635</td>
<td>b Retriever.SysDaemon.z</td>
</tr>
<tr>
<td>11</td>
<td>06155774</td>
<td>waiting</td>
<td>010300105634</td>
<td>Volume_Dumper.Daemon.z</td>
</tr>
<tr>
<td>12</td>
<td>06077554</td>
<td>waiting</td>
<td>010500105624</td>
<td>Cox.Multics.m</td>
</tr>
<tr>
<td>9</td>
<td>06122354</td>
<td>ready</td>
<td>010100105614</td>
<td>Operator.Operator.a</td>
</tr>
<tr>
<td>13</td>
<td>06166434</td>
<td>ready</td>
<td>011300105642</td>
<td>UNCP.CII-HB.a</td>
</tr>
<tr>
<td>6</td>
<td>06077754</td>
<td>running</td>
<td>005100105601</td>
<td>d TR_Admin.TR.p</td>
</tr>
</tbody>
</table>

F88, 6-3
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 b  Dumper.SysDaemon.z
Proc 8 DBR  6122354 ready  last on cpu b  Operator.Operator.a
Proc 9 DBR  6157174 running  on cpu b  Retriever.SysDaemon.z
Proc 10 DBR  6155774 waiting  last on cpu a  Volume_Dumper.SysDaemon.z
Proc 11 DBR  6077554 waiting  last on cpu a  Cox.Multics.m
Proc 12 DBR  6166434 ready  last on cpu a  UNCP.CII-HB.a
Proc 13 DBR  6174074 waiting  last on cpu a  JNye.CSDUK.a
Proc 14 DBR  5242354 running  on cpu c  Initializer.SysDaemon.z
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$page_fault_data:
DF1 Page Fault (43)
PR6 (sp) - 230|35500 >sll>stack_0.027|35500
SCU Data:
By: 42|3121 bound_library_wired_$formline_25
Ref: 230|36107 >sll>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$fim_data:
Access Violation Fault (51), Out of Segment Bounds
PR6 (sp) - 230|7020 >sll>stack_0.027|7020
SCU Data:
By: 150|11213 bound_x25_mp=$x25_mp|10513
Ref: 77777|15 >sll>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)

F88, 6-4
Setting Temporary pointers from 71|60.

Machine conditions from pds$signal_data:
Access Violation Fault (51), Out of Segment Bounds
  PR6 (sp) - 230|7020  >sll>stack_0.027|7020
SCU Data:
By: 150|11213  bound_x25_mpx|x25_mpx|10513
Ref: 77777|15  >sll>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 01:19:41.49496 mst Wed (113442703550026504)
Setting Temporary pointers from 71|140.

  azm: mc -prds sys

Machine conditions from prds$sys_trouble_data:
Connect Fault (21)
  PR6 (sp) - 230|30000  >sll>stack_0.027|30000
SCU Data:
By: 44|315  bound_prv_l$privileged_mode_ut|315
Ref: 230|36107  >sll>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> zzzzzzzzBBB BBB>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$_tty_io_15674
  234|4060  432|24345  >t>bound_as_mc$_mc_tty_13007
  234|3440  251|2476  >sll>bound_ipc_$ipc_real_12144
  234|3260  251|327  >sll>bound_ipc_$ipc_fast_265
  234|2400  256|2235  >sll>bound_oc$_oc_12235
  234|2000  263|410  >sll>bound_system_control$_system_control_410
Previous stack frame 77777|1

  azm: sk stack_0

Reverse trace of >sll>stack_0.027 (Seg 230)
Number of stack frames 14.
Stack begin = 230|100 Stack end = 230|33620
<table>
<thead>
<tr>
<th>FRAME</th>
<th>RETURN_PTR</th>
</tr>
</thead>
</table>
| 230|30000 | 35|1577 | bound_error_wired$
| 230|12600 | 122|25116 | bound_355_wired$channel_manager|2156 |
| 230|12320 | 41|5440 | bound_library_1$_unwind_stack_1370 |
| 230|12160 | 41|0 | bound_library_1$_init_vol_header_10 |
| 230|11640 | 41|10366 | bound_library_1$_signal_1726 |
| 230|11220 | 76|10 | 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 >sll>stack_0.027|7020
By: 150|11213 bound_x25_mp$_x25_mp$10513
Ref: 77777|15 >sll>stack_0.027|15
230|7020 150|0 bound_x25_mp$25_mp$25_mp$25_mp_10513
230|6300 122|25335 bound_355_wired$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 >sll>hcs_13374
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_prv$pxss|3767
72|1520 145|455 bound_tc_wired$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) >sll>stack_0.027|11300.
Pointer Registers:
PR0 (ap) - 42|17350 bound_library_wired$pll_operators_1|1426
PR1 (ab) - 77777|1 NULL POINTER
PR2 (bp) - 150|4155 bound_x25_mp$_x25_mp$3455
PR3 (bb) - 230|6174 >sll>stack_0.027|6174
PR4 (lp) - 150|11253 bound_x25_mp$_x25_mp$10553
PR5 (lb) - 77777|1 NULL POINTER
PR6 (sp) - 230|7020 >sll>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

F88, 6-6
A Register - 000000000000 Q Register - 0000000000040 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_mpux25_mp|150513 
Ref: 77777|15  >s11>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 (113442703550025604) 
Setting Temporary pointers from 230|1300. 

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|1300 

azm:  d sp -as stack_frame/arg_ptr/ 
stack_frame.arg_ptr = 230|6556  >s11>stack_0.027|6556 

azm:  sk sp -ag -for 1 

Reverse trace of  >s11>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_mpux25_mp_data|0 

Entry ptr 150|4155 bound_x25_mpux25_mp_data|3455 
Operator/Link ptr 42|17350 
Arg ptr 230|6556 
ARG 1: 221|16526 tty_buf|16526 
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 

F88, 6-7
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_1720
10360 005146 3520 00 epp2 5146
10361 0 00623 7001 00 tsox0 pr0|623
10362 000011 7260 07 lx16 11,d1
10363 777103 3520 04 epp2 -675,ic 007466 = 000000000000
10364 0 00717 7001 00 tsox0 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_1720 (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:

<table>
<thead>
<tr>
<th>offset</th>
<th>contents</th>
<th>offset</th>
<th>contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>114</td>
<td>000514657220</td>
<td>114</td>
<td>025720657200</td>
</tr>
<tr>
<td>115</td>
<td>000414710220</td>
<td>115</td>
<td>025773630200</td>
</tr>
<tr>
<td>116</td>
<td>000516657220</td>
<td>116</td>
<td>153450657200</td>
</tr>
<tr>
<td>117</td>
<td>000416710220</td>
<td>117</td>
<td>153450613200</td>
</tr>
</tbody>
</table>

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: 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

F88, 6-8
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, preempt_pending,
default_procs_required, dbr_loaded, shared_stack_0, firstsw
OFF: stop_pending, preempted, 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,
tempax = 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 = 5781222997, apad (1) through apad (10) = 0

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

F88, 6-9
DPS8 History Register Analysis

<table>
<thead>
<tr>
<th>HR</th>
<th>IC or</th>
<th>c</th>
<th>Memory</th>
<th>Address mc</th>
<th>flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>CU</td>
<td>307</td>
<td>tnz</td>
<td>*</td>
<td>304</td>
<td>4 poa raw ic pib port</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-stac zero carry</td>
<td></td>
</tr>
<tr>
<td>CU</td>
<td>310</td>
<td>stac</td>
<td>* i</td>
<td>236312</td>
<td>4 pia poa riw raw inf its internal</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-stac zero carry</td>
<td></td>
</tr>
<tr>
<td>CU</td>
<td>17</td>
<td></td>
<td>n* n</td>
<td>114776</td>
<td>4 poa riw raw its internal</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-stac zero carry</td>
<td></td>
</tr>
<tr>
<td>AU</td>
<td>[2776]</td>
<td></td>
<td></td>
<td>114776</td>
<td>r0 fap sdwm (A17)</td>
</tr>
<tr>
<td>CU</td>
<td>25</td>
<td></td>
<td>o</td>
<td>122222</td>
<td>40 poa internal</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-stac zero carry</td>
<td></td>
</tr>
<tr>
<td>AU</td>
<td>[222]</td>
<td></td>
<td></td>
<td>122222</td>
<td>r0 fap sdwm (A5)</td>
</tr>
<tr>
<td>CU</td>
<td>230</td>
<td></td>
<td>o</td>
<td>122222</td>
<td>60 poa raw saw pib port</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-stac zero carry</td>
<td></td>
</tr>
<tr>
<td>CU</td>
<td>17</td>
<td></td>
<td>311</td>
<td>115000</td>
<td>4 poa riw raw ic pib its port</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-stac zero carry</td>
<td></td>
</tr>
<tr>
<td>AU</td>
<td>[3000]</td>
<td></td>
<td></td>
<td>122222</td>
<td>r0 fap sdwm (A17) ptwm (A1)</td>
</tr>
<tr>
<td>CU</td>
<td>25</td>
<td></td>
<td>o</td>
<td>122220</td>
<td>4 poa raw ic pib port</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-stac zero carry</td>
<td></td>
</tr>
<tr>
<td>AU</td>
<td>[220]</td>
<td></td>
<td></td>
<td>122220</td>
<td>r0 fap sdwm (A5)</td>
</tr>
<tr>
<td>CU</td>
<td>17</td>
<td></td>
<td>312</td>
<td>236314</td>
<td>4 pia poa riw raw inf pib its internal</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-stac zero carry</td>
<td></td>
</tr>
<tr>
<td>CU</td>
<td>17</td>
<td></td>
<td>n* n</td>
<td>115002</td>
<td>4 poa riw raw pib its port</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-lda sign carry</td>
<td></td>
</tr>
<tr>
<td>AU</td>
<td>[3002]</td>
<td></td>
<td></td>
<td>236314</td>
<td>r0 fap sdwm (A17) ptwm (A1)</td>
</tr>
<tr>
<td>CU</td>
<td>25</td>
<td></td>
<td>o</td>
<td>122203</td>
<td>20 poa raw internal</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-lda sign carry</td>
<td></td>
</tr>
<tr>
<td>AU</td>
<td>[203]</td>
<td></td>
<td></td>
<td>122203</td>
<td>r0 fap sdwm (A5)</td>
</tr>
<tr>
<td>CU</td>
<td>17</td>
<td></td>
<td>313</td>
<td>115004</td>
<td>4 poa riw raw ic inh pib its port</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-sta sign carry</td>
<td></td>
</tr>
<tr>
<td>AU</td>
<td>[3004]</td>
<td></td>
<td></td>
<td>122203</td>
<td>r0 fap sdwm (A17)</td>
</tr>
<tr>
<td>CU</td>
<td>25</td>
<td></td>
<td>q1* n</td>
<td>122104</td>
<td>4 poa riw raw ic inh pib port</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-sta sign carry</td>
<td></td>
</tr>
<tr>
<td>AU</td>
<td>[104]</td>
<td></td>
<td></td>
<td>122104</td>
<td>r0 fap sdwm (A5)</td>
</tr>
<tr>
<td>CU</td>
<td>25</td>
<td></td>
<td>o</td>
<td>122067</td>
<td>62 poa raw ic inh internal</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-sta sign carry</td>
<td></td>
</tr>
<tr>
<td>AU</td>
<td>[67]</td>
<td></td>
<td></td>
<td>122067</td>
<td>r0 fap sdwm (A5)</td>
</tr>
<tr>
<td>CU</td>
<td>44</td>
<td></td>
<td>314</td>
<td>236316</td>
<td>4 pia poa raw inf inh port</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-sta sign carry</td>
<td></td>
</tr>
<tr>
<td>CU</td>
<td>44</td>
<td></td>
<td>o</td>
<td>236314</td>
<td>4 poa raw inh internal</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-sta sign carry</td>
<td></td>
</tr>
<tr>
<td>CU</td>
<td>44</td>
<td></td>
<td>?</td>
<td>236314</td>
<td>4 raw inh internal</td>
</tr>
<tr>
<td>OU</td>
<td></td>
<td></td>
<td></td>
<td>RS-REG-sta sign carry</td>
<td></td>
</tr>
</tbody>
</table>

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,*

F88, 6-10
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_
52056 0,0,0 3760 R W G - F 111000 25 bound hc data_wired
52360 0,0,0 15760 R W G - F 100000 66 iom_data
6240074 0,0,0 17760 R W G - F 111000 230 stack
52640 0,0,0 11760 R W G - F 100000 71 pds
LEVEL [B]
52436 0,0,0 763760 R W G - F 111000 102 sst_seg
52154 0,0,0 7760 RE F GC - F 111000 44 bound_priv_1
52232 0,0,0 3760 R W G - F 111000 70 oc_data
52366 0,0,0 43760 R W G - F 101100 112 cc_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
755400 no F 111000 42 7 bound_library_wired |formline|1504
755400 no F 100000 42 10 bound_library_wired |formline|1104
755200 no F 100000 42 11 bound_library_wired |formline|1504
755210 no F 100000 42 12 bound_library_wired |formline|2104
754600 no F 111000 42 13 bound_library_wired |formline|2504
754400 no F 111000 42 14 bound_library_wired |formline|3104
42602000 yes F 100000 230 15 $sll>stack_0.027|6400
2124400 yes F 100000 230 16 $sll>stack_0.027|7000
47012000 yes F 100000 230 17 $sll>stack_0.027|7400
LEVEL [B]
7330000 yes F 110000 71 0 prds|200
152000 yes F 111100 36 1 bound_interceptors|formline|600
146000 yes F 111000 35 7 bound_error_wired_1

F88, 6-11
<table>
<thead>
<tr>
<th>Level</th>
<th>State</th>
<th>Value</th>
<th>Parameter 1</th>
<th>Parameter 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>6240000</td>
<td>yes</td>
<td>F</td>
<td>111000</td>
<td>102</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>sst_seg</td>
<td>35600</td>
</tr>
<tr>
<td>3116000</td>
<td>yes</td>
<td>F</td>
<td>111000</td>
<td>230</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&gt;s1l&lt;</td>
<td>stack_0.027</td>
</tr>
<tr>
<td>LEVEL [C]</td>
<td></td>
<td></td>
<td>150000</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>yes</td>
<td>F</td>
<td>110000</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>bound_interceptors</td>
<td>fim</td>
</tr>
<tr>
<td></td>
<td>132000</td>
<td>yes</td>
<td>111100</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>bound_error_wired_1</td>
<td></td>
</tr>
<tr>
<td>LEVEL [D]</td>
<td></td>
<td></td>
<td>112000</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>yes</td>
<td>F</td>
<td>110000</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>ws_linkage</td>
<td>200</td>
</tr>
<tr>
<td></td>
<td>312000</td>
<td>yes</td>
<td>111100</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>oc_data</td>
<td>400</td>
</tr>
</tbody>
</table>

azm: vfam

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

azm: q
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_multics 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.
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 in the 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.
- pagno 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
  Specifies that 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.pll.

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

analyze_multics  F88, 7 - 6  associative_memory
12/12/83   aste

Syntax:   aste segno/segname [-control_args]

Function: displays active segment table (aste), 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.
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.
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 segment 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:

analyze_multics F88, 7 - 10 display
The whole structure "pvt".

The single element "n_entries" in the structure "pvt".

Any elements in the structure "sst" containing the string "time".
Note that the final slash is optional.

Any elements in the structure "sst" containing either the string "time" or the string "meter".

Element three of "sst.space".

Elements two, three, and four of "sst.space".

All elements of "sst.space".

Both elements of the "level" array for "sst.level(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
d 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.pll.
Syntax: \texttt{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):
-\texttt{-character, -ch, -ascii}
-\texttt{-instruction, -inst}
-\texttt{-octal, -oc}
-\texttt{-ptr, -p}
-\texttt{-pptr, -pp}
-\texttt{-pptrx, -ppx}
-\texttt{-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 (NSEC), -tm (NSEC)
  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".
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.
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 in the 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.
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, signaler, 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 conditions directly or it can point to the frame within which the machine conditions reside.
In the latter case, the offset is calculated 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.
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.
Syntax: \texttt{sdw (segno/name) (segno/name)}

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

Argument:
\texttt{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 \texttt{sdw} request displays the segment number, name, memory address, ring brackets, the maximum computed address, the entry bound address and a bit string \texttt{REWPUGCDF}.

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

Display definition of \texttt{REWPUGCDF}:
\texttt{REWPUGCDF} is the read permission bit
\texttt{E} is the execute permission bit
\texttt{W} is the write permission bit
\texttt{P} is the privileged bit
\texttt{U} is the unpaged bit, segment is unpaged is this is on
\texttt{G} is the gate indicator or entry bound bit. If off, the entry bound is checked by hardware
\texttt{C} is the cache enable switch.
\texttt{DF} is the directed fault bit. If on, the necessary page of the segment is in memory.
Example of SDW display:

<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>RNGS</th>
<th>CA-MAX</th>
<th>REWPUGCDF</th>
<th>EBOUND</th>
<th>SEGNO</th>
<th>SEGMENT-NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>6262154</td>
<td>000</td>
<td>177777</td>
<td>R W G DF</td>
<td>0</td>
<td>200</td>
<td>str_seg</td>
</tr>
<tr>
<td>0</td>
<td>000</td>
<td>377777</td>
<td>R W G</td>
<td>0</td>
<td>300</td>
<td>&gt;udd&gt;Multics&gt;GDixon</td>
</tr>
</tbody>
</table>

analyze_multics
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:  \( \text{name [VIRTUAL-ADDR | number]} \)
  \( \text{[name number]} \) or \( \text{[name VIRTUAL-ADDR]} \)

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".
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".
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 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 in the 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.

<table>
<thead>
<tr>
<th>number</th>
<th>name</th>
<th>number</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>pr0</td>
<td>ap</td>
<td>pr4</td>
<td>lp</td>
</tr>
<tr>
<td>pr1</td>
<td>ab</td>
<td>pr5</td>
<td>lb</td>
</tr>
<tr>
<td>pr2</td>
<td>bp</td>
<td>pr6</td>
<td>sp</td>
</tr>
<tr>
<td>pr3</td>
<td>bb</td>
<td>pr7</td>
<td>sb</td>
</tr>
<tr>
<td>prmc</td>
<td></td>
<td>prfr</td>
<td></td>
</tr>
<tr>
<td></td>
<td>intended to be a pointer to the current MCs.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>intended to be a pointer to the current stack frame.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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
  answer_table                answer_table
  user_table_entry            anstbl
  apte                        apte
  area_structures             area_header
    area_header               extend_block
    extend_block             area_block
  ast_lock_meters             ast_lock_meters
  aste                        aste
  bos_dump                    bos_dump

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

Structure_names(definition - dir_link):
  INCLUDE FILE NAME

  analyze_multics             F88, 7 - 33
  structure_names
STRUCTURE

definition

dir_allocation_area

area
dir_acl

acl_entry

access_name
dir_entry

entry
dir_header
dir
dir_ht

hash_table
dir_link

link

STRUCTURE

definition
dir_allocation_area
dir_acl_entry
dir_access_name
dir_entry
dir_header
dir_hash_table
dir_link

Structure_names(dir_name - dn355_data):

INCLUDE FILE NAME

STRUCTURE

NAME USED IN DISPLAY

dir_name

dir_lock_seg
dir_lock_segment

dir_lock_segment_header

dir_lock

disk_table
disk

disk_table_lv_entry

dn355_data
datanet_infofnp_info

Structure_names(dn355_mailbox - dskdcl):

INCLUDE FILE NAME

STRUCTURE

dn355_mailbox
datanet_mbx

sub_mbx

gsub_mbx

gsub_mbx

fnp_sub_mbx

input_sub_mbx

dskdcl

disk_data
disktab
disk_channel_table

tab

devtab

quentry

Structure_names(ect_structures - flagbox):

analyze_multics

F88, 7 - 34

structure_names
INCLUDE FILE NAME
STRUCTURE
ect_structures
  ect_header
  wait_channel
  call_channel
  event_message
  itt_message
  event_channel_name
  fast_channel_name
  event_message_data
  event_call_info
  event_call_info
  event_wait_info
  event_wait_info
fault_vector
  fv
flagbox
  fgbx

Structure_names(fs_vol_label - io_status_entry):
INCLUDE FILE NAME
  STRUCTURE
  NAME USED IN DISPLAY
  fs_vol_label
    label
  int_unpaged_page_tables
    iupt
    iupte
  io_page_tables
    io_page_tables
    io_ptw
    io_ptw
    io_ptw
  io_special_status
    io_special_status
  io_status
    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
  iocbx
    iocb
  ioi_data
    ioi_data
    gte
    dte
  iom_data

analyze_multics
  F88, 7 - 35
<table>
<thead>
<tr>
<th>Structure</th>
<th>include file</th>
<th>name used in display</th>
</tr>
</thead>
<tbody>
<tr>
<td>iom_data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>per_device</td>
<td></td>
<td></td>
</tr>
<tr>
<td>per_iom</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iom_mailbox_seg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>iom_mailbox</td>
<td></td>
<td></td>
</tr>
<tr>
<td>channel_mailbox</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Structure_names(iom_dcw - its):

<table>
<thead>
<tr>
<th>Structure</th>
<th>include file</th>
<th>name used in display</th>
</tr>
</thead>
<tbody>
<tr>
<td>iom_dcw</td>
<td></td>
<td>iom_ddcw</td>
</tr>
<tr>
<td>dcw</td>
<td></td>
<td>iom_ddcw</td>
</tr>
<tr>
<td>tdcw</td>
<td></td>
<td>iom_tdcw</td>
</tr>
<tr>
<td>iom_lpw</td>
<td></td>
<td>iom_lpw</td>
</tr>
<tr>
<td>lpw</td>
<td></td>
<td>iom_lpw</td>
</tr>
<tr>
<td>lpw_ext</td>
<td></td>
<td>iom_lpw_ext</td>
</tr>
<tr>
<td>iom_pcw</td>
<td></td>
<td>iom_pcw</td>
</tr>
<tr>
<td>pcw</td>
<td></td>
<td>iom_pcw</td>
</tr>
<tr>
<td>idcw</td>
<td></td>
<td>iom_idcw</td>
</tr>
<tr>
<td>iom_scw</td>
<td></td>
<td>iom_scw</td>
</tr>
<tr>
<td>scw</td>
<td></td>
<td>iom_scw</td>
</tr>
<tr>
<td>its</td>
<td></td>
<td>its</td>
</tr>
<tr>
<td>its_unsigned</td>
<td></td>
<td>its_unsigned</td>
</tr>
<tr>
<td>itp</td>
<td></td>
<td>itp</td>
</tr>
<tr>
<td>itp_unsigned</td>
<td></td>
<td>itp_unsigned</td>
</tr>
</tbody>
</table>

Structure_names(itt_entry - linkdcl):

<table>
<thead>
<tr>
<th>Structure</th>
<th>include file</th>
<th>name used in display</th>
</tr>
</thead>
<tbody>
<tr>
<td>itt_entry</td>
<td></td>
<td>itt_entry</td>
</tr>
<tr>
<td>kst</td>
<td></td>
<td>kst</td>
</tr>
<tr>
<td>kste</td>
<td></td>
<td>kste</td>
</tr>
<tr>
<td>lct</td>
<td></td>
<td>lct</td>
</tr>
<tr>
<td>lcte</td>
<td></td>
<td>lcte</td>
</tr>
<tr>
<td>linkdcl</td>
<td></td>
<td>link_pair;</td>
</tr>
<tr>
<td>exp_word</td>
<td></td>
<td>link_exp_word</td>
</tr>
<tr>
<td>type_pair</td>
<td></td>
<td>link_type_pair</td>
</tr>
<tr>
<td>header</td>
<td></td>
<td>linkage_header</td>
</tr>
<tr>
<td>virgin_linkage_header</td>
<td></td>
<td>virgin_linkage_header</td>
</tr>
<tr>
<td>trap_word</td>
<td></td>
<td>link_trap_word</td>
</tr>
</tbody>
</table>

Structure_names(lot - mcs_modes_change_list):

<table>
<thead>
<tr>
<th>Structure</th>
<th>include file</th>
<th>name used in display</th>
</tr>
</thead>
<tbody>
<tr>
<td>lot</td>
<td></td>
<td>lot</td>
</tr>
<tr>
<td>isot</td>
<td></td>
<td>isot</td>
</tr>
<tr>
<td>isotl</td>
<td></td>
<td>isotl</td>
</tr>
</tbody>
</table>

analyze_multics | F88, 7 - 36 | structure_names
Structure_names(mc_trace_data - oc_data):

<table>
<thead>
<tr>
<th>INCLUDE FILE NAME</th>
<th>STRUCTURE</th>
<th>NAME USED IN DISPLAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>mc_trace_data</td>
<td>trace_array</td>
<td>mcs_trace_array</td>
</tr>
<tr>
<td></td>
<td>trace_entry</td>
<td>mcs_trace_entry</td>
</tr>
<tr>
<td>mdcs</td>
<td>mdcs</td>
<td>mdcs</td>
</tr>
<tr>
<td></td>
<td>mdc dirent</td>
<td>mdc_s account</td>
</tr>
<tr>
<td></td>
<td>accsent</td>
<td>mdcs_path</td>
</tr>
<tr>
<td>mstr</td>
<td>mst label</td>
<td>mst label</td>
</tr>
<tr>
<td></td>
<td>mstr_header</td>
<td>mstr_header</td>
</tr>
<tr>
<td></td>
<td>mstr_trailer</td>
<td>mstr_trailer</td>
</tr>
<tr>
<td></td>
<td>volume_identifier</td>
<td>mst_volume_id</td>
</tr>
<tr>
<td></td>
<td>mst_volume_id</td>
<td>mst_volume_id</td>
</tr>
<tr>
<td>oc_data</td>
<td>oc_data</td>
<td>oc_data</td>
</tr>
</tbody>
</table>

Structure_names(oc_log_meters - pv_holdt):

<table>
<thead>
<tr>
<th>INCLUDE FILE NAME</th>
<th>STRUCTURE</th>
<th>NAME USED IN DISPLAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>oc_log_meters</td>
<td>olm</td>
<td>oc_log_meters</td>
</tr>
<tr>
<td></td>
<td>pathname_am</td>
<td>pathname_am</td>
</tr>
<tr>
<td></td>
<td>pam</td>
<td>pam</td>
</tr>
<tr>
<td>pcb</td>
<td>pcd</td>
<td>pcd</td>
</tr>
<tr>
<td>pit</td>
<td>pit</td>
<td>pit</td>
</tr>
<tr>
<td>ptw.adp</td>
<td>adp_core_ptw</td>
<td>adp_core_ptw</td>
</tr>
<tr>
<td></td>
<td>adp_ptw</td>
<td>adp_ptw</td>
</tr>
<tr>
<td>ptw.168</td>
<td>168_core_ptw</td>
<td>168_core_ptw</td>
</tr>
<tr>
<td></td>
<td>168_ptw</td>
<td>168_ptw</td>
</tr>
<tr>
<td>pv_holdt</td>
<td>pv_holdt</td>
<td>pv_holdt</td>
</tr>
</tbody>
</table>

analyze_multics F88, 7 - 37
Structure names(pvt - rnt):

<table>
<thead>
<tr>
<th>STRUCTURE</th>
<th>NAME USED IN DISPLAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>pvt</td>
<td>pvt</td>
</tr>
<tr>
<td>pvte</td>
<td>pvte</td>
</tr>
<tr>
<td>pvt_array</td>
<td>pvt_array</td>
</tr>
<tr>
<td>rcp_com_seg</td>
<td>rcs</td>
</tr>
<tr>
<td>rcs</td>
<td>rcse</td>
</tr>
<tr>
<td>rcp_data</td>
<td>rcpd</td>
</tr>
<tr>
<td>rcpd</td>
<td>rcp_device</td>
</tr>
<tr>
<td>dtype</td>
<td>rcp_volume</td>
</tr>
<tr>
<td>device</td>
<td></td>
</tr>
<tr>
<td>volume</td>
<td></td>
</tr>
<tr>
<td>rnt</td>
<td>rnt</td>
</tr>
<tr>
<td>rnte</td>
<td></td>
</tr>
</tbody>
</table>

Structure names(sdw.adp - slt):

<table>
<thead>
<tr>
<th>STRUCTURE</th>
<th>NAME USED IN DISPLAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>sdw.adp</td>
<td>adp_sdw</td>
</tr>
<tr>
<td>sdw.168</td>
<td>168_sdw</td>
</tr>
<tr>
<td>sdw_info</td>
<td>sdw_info</td>
</tr>
<tr>
<td>segdamage_msg</td>
<td>segdamage_msg</td>
</tr>
<tr>
<td>signaller_stack</td>
<td>signaller_stack</td>
</tr>
<tr>
<td>slt</td>
<td>slt</td>
</tr>
<tr>
<td>name_seg</td>
<td>slt_name_seg</td>
</tr>
<tr>
<td>segnam</td>
<td>slt_segname</td>
</tr>
<tr>
<td>path</td>
<td>slt_path</td>
</tr>
<tr>
<td>acls</td>
<td>slt_acls</td>
</tr>
</tbody>
</table>

Structure names(slte - str):

<table>
<thead>
<tr>
<th>STRUCTURE</th>
<th>NAME USED IN DISPLAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>slte</td>
<td>slte</td>
</tr>
<tr>
<td>sst</td>
<td>sst</td>
</tr>
<tr>
<td>sstnt</td>
<td>sstntstack_0_data</td>
</tr>
<tr>
<td>sdt</td>
<td>stack_0_data</td>
</tr>
<tr>
<td>sdte</td>
<td>sdte</td>
</tr>
<tr>
<td>stack_frame</td>
<td>stack_frame</td>
</tr>
<tr>
<td>stack_header</td>
<td></td>
</tr>
</tbody>
</table>

analyze_multics | F88, 7 - 38 | structure_names
<table>
<thead>
<tr>
<th>INCLUDE FILE NAME</th>
<th>STRUCTURE NAME</th>
<th>USED IN DISPLAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>syserr_data</td>
<td>sd</td>
<td>wired_syserr_log</td>
</tr>
<tr>
<td></td>
<td>wlog</td>
<td>wired_syserr_message</td>
</tr>
<tr>
<td>syserr_log</td>
<td>slog</td>
<td>syserr_log</td>
</tr>
<tr>
<td></td>
<td>smess</td>
<td>syserr_message</td>
</tr>
<tr>
<td>tcb</td>
<td></td>
<td>tcb</td>
</tr>
<tr>
<td>tcm</td>
<td>tc_data</td>
<td>tc_data</td>
</tr>
<tr>
<td>wct_entry</td>
<td></td>
<td>wct_entry</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tty_buf</td>
<td>tty_buf</td>
<td>tty_buf</td>
</tr>
<tr>
<td>tty_buffer_block</td>
<td>buffer</td>
<td>free_tty_buffer</td>
</tr>
<tr>
<td>tty_tables</td>
<td>tty_tables_hdr</td>
<td></td>
</tr>
<tr>
<td>unpaged_page_tables</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>upt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>upt_entry</td>
<td></td>
</tr>
<tr>
<td>vol_map</td>
<td>vol_map</td>
<td></td>
</tr>
<tr>
<td>volume_registration</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vtoc_buffer</td>
<td>vtoc_buffer</td>
<td></td>
</tr>
<tr>
<td>vtoc_header</td>
<td>vtoc_header</td>
<td></td>
</tr>
<tr>
<td>vtoce</td>
<td></td>
<td>vtoce</td>
</tr>
<tr>
<td>wire_proc_data</td>
<td></td>
<td>wire_proc_data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>wtc_b</td>
<td></td>
<td>wtc_b</td>
</tr>
</tbody>
</table>

`Structure_names(syserr_data - wct_entry):`

`Structure_names(vtoc_buffer - wtcb):`

F88, 7 - 39  structure_names
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_N1..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).
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.
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$ptl +30 are resolved to be the SEGNO for sst_seg with the offset of pttl plus 30 octal locations; sst$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:

\[
\begin{align*}
\text{segno|offset,*} & & \text{segname|offset,*} & & \text{symbol,*} \\
\text{temp_ptr,*} & & \text{temp_ptr|offset,*} & &
\end{align*}
\]

Examples of indirection:

\[
\begin{align*}
17|230, * & & \text{sst|230,*} & & \text{sst$cmp,*+2} \\
\text{sp,*} & & \text{sp|230,*} & &
\end{align*}
\]
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$check_fault finding a problem, the machine condition CU data is displayed and all temporary pointers are set from these machine conditions. If this was an Execute Fault then some lock info is printed on the process selected is lock ordered. First sst_seg$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

<table>
<thead>
<tr>
<th>Address</th>
<th>Opcode</th>
<th>I</th>
<th>A</th>
<th>TAG</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td>8</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

#### ACCUMULATOR REGISTER (A)

<table>
<thead>
<tr>
<th>A-Upper (au)</th>
<th>A-Lower (al)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>0</td>
<td>8</td>
</tr>
</tbody>
</table>

#### QUOTIENT REGISTER (Q)

<table>
<thead>
<tr>
<th>Q-Upper (qu)</th>
<th>Q-Lower (ql)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>0</td>
<td>8</td>
</tr>
</tbody>
</table>

#### ACCUMULATOR-QUOTIENT REGISTER (AQ)

<table>
<thead>
<tr>
<th>Even Word (A)</th>
<th>Odd Word (Q)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
</tr>
</tbody>
</table>

#### EXPONENT REGISTER (E)

<table>
<thead>
<tr>
<th>Exponent</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

F88, 8-1
EXPONENT-ACCUMULATOR-QUOTIENT REGISTER (EAQ)

INDEX REGISTER (Xn)

POINTER REGISTER (PRn) or ITS POINTER FORMAT

F88, 8-2
PACKED POINTER FORMAT

0 0 0 1 1 3
0 5 6 7 8 5

<table>
<thead>
<tr>
<th>Bit Off.</th>
<th>Segment Number</th>
<th>Word Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>12</td>
<td>18</td>
</tr>
</tbody>
</table>

INDIRECT WORD FORMAT

0 1 1 3 3
0 7 8 0 5

<table>
<thead>
<tr>
<th>Address</th>
<th>Tally</th>
<th>Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>12</td>
<td>3</td>
</tr>
</tbody>
</table>
/* 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

F88, 8-5
### Processor Instructions

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>a4bd</td>
<td>Add 4-bit character displacement to AR</td>
</tr>
<tr>
<td>a6bd</td>
<td>Add 6-bit character displacement to AR</td>
</tr>
<tr>
<td>a9bd</td>
<td>Add 9-bit character displacement to AR</td>
</tr>
<tr>
<td>aarN</td>
<td>Alphanumeric descriptor to ARN</td>
</tr>
<tr>
<td>abd</td>
<td>Add bit displacement to AR</td>
</tr>
<tr>
<td>absa</td>
<td>Abs address to A register</td>
</tr>
<tr>
<td>ad2d</td>
<td>Add using two decimal operands</td>
</tr>
<tr>
<td>ad3d</td>
<td>Add using three decimal operands</td>
</tr>
<tr>
<td>ada</td>
<td>Add to A register</td>
</tr>
<tr>
<td>adaq</td>
<td>Add to AQ register</td>
</tr>
<tr>
<td>ade</td>
<td>Add to E register</td>
</tr>
<tr>
<td>adl</td>
<td>Add low to AQ register</td>
</tr>
<tr>
<td>adla</td>
<td>Add logical to A register</td>
</tr>
<tr>
<td>adlaq</td>
<td>Add logical to AQ register</td>
</tr>
<tr>
<td>adlq</td>
<td>Add logical to Q register</td>
</tr>
<tr>
<td>adlxN</td>
<td>Add logical to index N</td>
</tr>
<tr>
<td>adq</td>
<td>Add to Q register</td>
</tr>
<tr>
<td>adwpN</td>
<td>Add to word numeric field of PRN</td>
</tr>
<tr>
<td>adxN</td>
<td>Add to index N</td>
</tr>
<tr>
<td>alr</td>
<td>A register left rotate</td>
</tr>
<tr>
<td>als</td>
<td>A register left shift</td>
</tr>
<tr>
<td>ana</td>
<td>AND to A register</td>
</tr>
<tr>
<td>anaq</td>
<td>AND to AQ register</td>
</tr>
<tr>
<td>anq</td>
<td>AND to Q register</td>
</tr>
<tr>
<td>ansa</td>
<td>AND to storage from A register</td>
</tr>
<tr>
<td>ansq</td>
<td>AND to storage from Q register</td>
</tr>
<tr>
<td>ansxN</td>
<td>AND to storage from index N</td>
</tr>
<tr>
<td>anxN</td>
<td>AND to index N</td>
</tr>
<tr>
<td>aos</td>
<td>Add one to storage</td>
</tr>
<tr>
<td>araN</td>
<td>ARN to alpha descriptor</td>
</tr>
<tr>
<td>arl</td>
<td>A register right logical shift</td>
</tr>
<tr>
<td>arnN</td>
<td>ARN to numeric descriptor</td>
</tr>
<tr>
<td>ars</td>
<td>A register right shift</td>
</tr>
<tr>
<td>asa</td>
<td>Add stored to A register</td>
</tr>
<tr>
<td>asq</td>
<td>Add stored to Q register</td>
</tr>
<tr>
<td>asxN</td>
<td>Add stored to index N</td>
</tr>
<tr>
<td>awca</td>
<td>Add with carry to A register</td>
</tr>
<tr>
<td>awcq</td>
<td>Add with carry to Q register</td>
</tr>
<tr>
<td>awd</td>
<td>Add word displacement to AR</td>
</tr>
<tr>
<td>bcd</td>
<td>Binary-to-BCD</td>
</tr>
<tr>
<td>btd</td>
<td>Binary-to-dec</td>
</tr>
<tr>
<td>call16</td>
<td>Call</td>
</tr>
<tr>
<td>camp</td>
<td>Clear associative memory paged</td>
</tr>
<tr>
<td>cams</td>
<td>Clear associative memory segmented</td>
</tr>
<tr>
<td>cana</td>
<td>Comparative and with A register</td>
</tr>
<tr>
<td>canaq</td>
<td>Comparative and with AQ register</td>
</tr>
<tr>
<td>canc</td>
<td>Comparative and with Q register</td>
</tr>
<tr>
<td>canxN</td>
<td>Comparative and with index N</td>
</tr>
<tr>
<td>cioc</td>
<td>Connect</td>
</tr>
<tr>
<td>cmg</td>
<td>Compare magnitude</td>
</tr>
</tbody>
</table>
Compare masked
Compare with A register
Compare with AQ register
Compare bit strings
Compare alphanumeric character strings
Compare numeric
Compare with Q register
Compare with index N
Comparative not with A register
Comparative not with AQ register
Comparative not with Q register
Comparative not with index N
Combine bit strings left
Combine bit strings right
Compare with limits
DP floating add
DP floating compare magnitude
DP floating compare
DP floating divide inverted
DP floating divide
DP floating load
DP floating multiply
DP floating round
DP floating subtract
DP floating store
DP floating store rounded
Delay until interrupt signal
Divide integer
Derail
Dec-to-binary convert
DP unnormalized floating add
DP unnormalized floating multiply
DP unnormalized floating subtract
Divide using two decimal operands
Divide using three decimal operands
Divide fraction
Effective address to A register
Effective address to Q register
Effective address to segment number field of PRn
Effective address to word and bit fields of PRn
Effective address to index N
Effective pointer to AQ register
Effective pointer at base to PRn
Effective pointer to PRn
XOR to A register
XOR to AQ register
XOR to Q register
XOR to storage with A register
XOR to storage with Q register
XOR to storage with index N
XOR to index N
Floating add
Floating compare magnitude
Floating compare
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
lda  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
lp1  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
lx1N Load index N from lower
mlr  Move alphanumeric left to right
mm1  Master mode entry 1
mm2  Master mode entry 2
mm3  Master mode entry 3
mm4  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 register)
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
qlr1 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
rdp Repeat double
rpl Repeat link
rpt Repeat
rsr Read system controller register
rs 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
spbN 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
stcl 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	nnz 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
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tss</td>
<td>Transfer and set slave</td>
</tr>
<tr>
<td>tsxN</td>
<td>Transfer and set index N</td>
</tr>
<tr>
<td>ttf</td>
<td>Transfer on tally indicator off</td>
</tr>
<tr>
<td>ttn</td>
<td>Transfer on tally indicator on</td>
</tr>
<tr>
<td>tze</td>
<td>Transfer on zero</td>
</tr>
<tr>
<td>ufa</td>
<td>Unnormalized floating add</td>
</tr>
<tr>
<td>ufm</td>
<td>Unnormalized floating multiply</td>
</tr>
<tr>
<td>ufs</td>
<td>Unnormalized floating sub</td>
</tr>
<tr>
<td>xec</td>
<td>Execute</td>
</tr>
<tr>
<td>xed</td>
<td>Execute double</td>
</tr>
</tbody>
</table>
SIGNALLING AND CRAWLOUTS

RING 4 FAULT EXAMPLE

1. PROGRAM MY_PGM EXECUTING IN RING 4

STACK 4

init_proc_
listen_
cmd_processor_
my_pgm

STACK 0

PAGE_FAULT_DATA

PDS

fim_data
signal_data
condition_name
mc_save_area

F88, 9-1
### RING 4 FAULT EXAMPLE

2. **PARITY FAULT OCCURS**

<table>
<thead>
<tr>
<th>STACK_4</th>
<th>STACK_0</th>
<th>PDS</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>init_proc_</code></td>
<td></td>
<td><code>page_fault_data</code></td>
</tr>
<tr>
<td></td>
<td><code>listen_</code></td>
<td><code>fim</code></td>
</tr>
<tr>
<td><code>cmd_processor_</code></td>
<td></td>
<td><code>fim_data</code></td>
</tr>
<tr>
<td><code>my_pgm</code></td>
<td></td>
<td><code>+---**MCs**--</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>signal_data</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>condition_name</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>mc_save_area</code></td>
</tr>
</tbody>
</table>

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

---

F88, 9-2
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".
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.)
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

<table>
<thead>
<tr>
<th>STACK_4</th>
<th>STACK_0</th>
<th>PDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>+-&gt; init_proc</td>
<td></td>
<td>page_fault_data</td>
</tr>
<tr>
<td>+-&gt; listen</td>
<td></td>
<td>fim_data (MCs)</td>
</tr>
<tr>
<td>+-&gt; cmd_processor</td>
<td></td>
<td>signal_data (MCs)</td>
</tr>
<tr>
<td>+-&gt; my_pgm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>return_to_ring_0</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>MCs</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+&lt; signal</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

STACK 4

init_proc_
listen_
cmd_processor_
my_pgm
return_to_ring_0_
**MCs**
signal_
handler

STACK 0

PDS

page_fault_data

fim_data

signal_data

condition_name

"parity"

mc_save_area

**MCs**

Signal_ invokes the condition handler.
FAULT RESTART EXAMPLE

1. HANDLER DECIDES TO RESTART FAULT

STACK_4                      STACK_0                      PDS

init_proc_

listen_

cmd_processor_

my_pgm

return_to_ring_0

**MCs**

signal_

-----------------------------

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

---

page_fault_data

-----------------------------

fim_data

(MCs)

signal_data

(MCs)

-----------------------------

condition_name
("parity")

-----------------------------

mc_save_area

**MCs**

-----------------------------
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.
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

<table>
<thead>
<tr>
<th>STACK 4</th>
<th>STACK 0</th>
<th>PDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>init_proc_</td>
<td></td>
<td>page_fault_data</td>
</tr>
<tr>
<td></td>
<td>listen_</td>
<td>fim_data (MCs)</td>
</tr>
<tr>
<td>cmd_processor_</td>
<td></td>
<td>signal_data (MCs)</td>
</tr>
<tr>
<td>my_pgm</td>
<td></td>
<td>condition_name (&quot;parity&quot;)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mc_save_area (MCs)</td>
</tr>
</tbody>
</table>

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

1. HANDLER IS UNWOUND

STACK_4

init_proc_

listen_

cmd_processor_

my_pgm

return_to_ring_0_

**MCs**

signal_

STACK_0

PDS

page_fault_data

fim_data

(signal_data (MCs))

condition_name ("parity")

mc_save_area

**MCs**

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

2. SIGNAL IS UNWOUND

<table>
<thead>
<tr>
<th>STACK_4</th>
<th>STACK_0</th>
<th>PDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>init_proc</td>
<td></td>
<td>page_fault_data</td>
</tr>
<tr>
<td>listen</td>
<td></td>
<td>fim_data</td>
</tr>
<tr>
<td>cmd_processor</td>
<td></td>
<td>signal_data</td>
</tr>
<tr>
<td>my_pgm</td>
<td></td>
<td>(MCs)</td>
</tr>
<tr>
<td>return_to_ring_0</td>
<td></td>
<td>condition_name</td>
</tr>
<tr>
<td><strong>MCs</strong></td>
<td></td>
<td>(&quot;parity&quot;)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mc_save_area</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>MCs</strong></td>
</tr>
</tbody>
</table>

Next, signal's stack frame is unwound.
FAULT ABANDONMENT EXAMPLE

3. FIM FRAME'S CLEANUP HANDLER IS INVOKED

STACK_4

init_proc_

listen_

cmd_processor_

my_pgm

return_to_ring_0

**MCs**

STACK_0

restart_fault|1

PDS

<table>
<thead>
<tr>
<th>page_fault_data</th>
</tr>
</thead>
</table>

| fim_data |
| (MCs) |

| signal_data |
| (MCs) |

| condition_name |
| ("parity") |

| mc_save_area |

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

STACK_4

<table>
<thead>
<tr>
<th>init_proc</th>
</tr>
</thead>
<tbody>
<tr>
<td>listen</td>
</tr>
<tr>
<td>cmd_processor</td>
</tr>
<tr>
<td>my_pgm</td>
</tr>
</tbody>
</table>

STACK_0

<table>
<thead>
<tr>
<th>hcs</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
</tr>
</tbody>
</table>

PDS

<table>
<thead>
<tr>
<th>page_fault_data</th>
</tr>
</thead>
<tbody>
<tr>
<td>fim_data</td>
</tr>
<tr>
<td>signal_data</td>
</tr>
<tr>
<td>condition_name</td>
</tr>
<tr>
<td>mc_save_area</td>
</tr>
</tbody>
</table>

F88, 9-15
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".
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

STACK_4

init_proc_

listen_

cmd_processor_

my_pgm

STACK_0

+->  hcs_

+->  status_

**MCs**

fim

**MCs**

(MSG)

(return_to_ring_0

**MCs**

+<-

signal_

PDS

page_fault_data

fim_data

(MCs)

signal_data

(MCs)

condition_name

("parity")

mc_save_area

**MCs**

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

STACK_4          STACK_0          PDS

+-> init_proc_
+-> listen_
+-> cmd_processor_
+-> my_pgm

**MCs**

\*\*MCs\*\*

\*\*MCs\*\*

condition_name
("parity")

mc_save_area

**MCs**

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
30

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

**Fault Type: Execute**

**Definition:**

1. The EXECUTE pushbutton on the processor maintenance panel has been pressed.
2. An external gate signal has been substituted 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

<table>
<thead>
<tr>
<th>PR0 (ap)</th>
<th>PR1 (ab)</th>
<th>PR2 (bp)</th>
<th>PR3 (bb)</th>
<th>PR4 (lp)</th>
<th>PR5 (lb)</th>
<th>PR6 (sp)</th>
<th>PR7 (sb)</th>
</tr>
</thead>
<tbody>
<tr>
<td>240</td>
<td>610</td>
<td>240</td>
<td>460</td>
<td>113</td>
<td>4500</td>
<td>113</td>
<td>0</td>
</tr>
</tbody>
</table>

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

| 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

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 TR0 occurred 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 pdslO
Time Stored - 03/11/83 1704.3 hfh Fri (111557226670514660)

Pointer Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0 (ap)</td>
<td>270</td>
<td>7120</td>
</tr>
<tr>
<td>PR1 (ab)</td>
<td>244</td>
<td>16200</td>
</tr>
<tr>
<td>PR2 (bp)</td>
<td>22</td>
<td>4000</td>
</tr>
<tr>
<td>PR3 (bb)</td>
<td>244</td>
<td>15234</td>
</tr>
<tr>
<td>PR4 (lp)</td>
<td>257</td>
<td>24102</td>
</tr>
<tr>
<td>PR5 (lb)</td>
<td>325</td>
<td>10</td>
</tr>
<tr>
<td>PR6 (sp)</td>
<td>244</td>
<td>16060</td>
</tr>
<tr>
<td>PR7 (sb)</td>
<td>244</td>
<td>15360</td>
</tr>
</tbody>
</table>

Processor Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XO - 0</td>
<td>X1 - 100</td>
<td>X2 - 777773</td>
</tr>
<tr>
<td>A Register</td>
<td>000000000000</td>
<td>Q Register</td>
</tr>
<tr>
<td>Timer Register</td>
<td>777777751</td>
<td>Ring Alarm Register</td>
</tr>
<tr>
<td>SCU masks</td>
<td>- 031460000014</td>
<td>631460000007</td>
</tr>
<tr>
<td>SCU Data at pdslO</td>
<td>30</td>
<td>400315170041</td>
</tr>
<tr>
<td></td>
<td>017775000240</td>
<td>001231000500</td>
</tr>
<tr>
<td>Timer Runout Fault (11)</td>
<td>At: 315</td>
<td>17775</td>
</tr>
<tr>
<td>On: cpu a (#0)</td>
<td>Indicators: ^bar, mif</td>
<td></td>
</tr>
<tr>
<td>APU Status: sdwamm, sd-on, ptwamm, pt-on, fap</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CU Status: rfi, fif</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instructions:</td>
<td>636336</td>
<td>076</td>
</tr>
<tr>
<td></td>
<td>636337</td>
<td>000</td>
</tr>
<tr>
<td>EIS Pointers and Lengths</td>
<td>50</td>
<td>0004000001040</td>
</tr>
<tr>
<td></td>
<td>005735000030</td>
<td>000777777040</td>
</tr>
</tbody>
</table>

F88, 10-5
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 000252 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 (111511625326645425)

Pointer Registers

PR0 (ap) - 270|7120  >sll>bound_system_control $sc_parse_|4126
PR1 (ab) - 244|11016  >pdd> zzzzzzzbBBBBB>stack_4|11016
PR2 (bp) - 244|15270  >pdd> zzzzzzzbBBBBB>stack_4|15270
PR3 (bb) - 360|1425  >sc1>as_meter_table|1425
PR4 (lp) - 257|31154  >s1l>bound_fsim_
PR5 (lb) - 257|33250  >s1l>bound_fsim_
PR6 (sp) - 244|10620  >pdd> zzzzzzzbBBBBB>stack_4|10620
PR7 (sb) - 360|2016  >sc1>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  >sll>bound_system_control_
On: cpu c (#2)
Indicators: zero, ~bar
AFU Status: sd-on, pt-on, fabs
CU Status: rfi, fif
Instructions:

      25676  000001 3160 07  canq  1,d1
      25677  000001 3160 07  canq  1,d1

F88, 10-8
Fault Type: Connect

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

Pointer Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0 (ap)</td>
<td>270</td>
</tr>
<tr>
<td>PR1 (ab)</td>
<td>244</td>
</tr>
<tr>
<td>PR2 (bp)</td>
<td>244</td>
</tr>
<tr>
<td>PR3 (bb)</td>
<td>244</td>
</tr>
<tr>
<td>PR4 (lp)</td>
<td>456</td>
</tr>
<tr>
<td>PR5 (lb)</td>
<td>244</td>
</tr>
<tr>
<td>PR6 (sp)</td>
<td>244</td>
</tr>
<tr>
<td>PR7 (sb)</td>
<td>257</td>
</tr>
</tbody>
</table>

Processor Registers

X0 - 4147 X1 - 54640 X2 - 150732 X3 - 151116 X4 - 22 X6 - 71 X7 - 2220
A Register - 777777777775 Q Register - 000000000000 E Register - 0
Timer Register - 000143044 Ring Alarm Register - 0
SCU masks - 000240000004 000340000000 Fault Register - 000400000000 ($CON A)

SCU Data at pds|170

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>170</td>
</tr>
<tr>
<td>400456174043 000000000021 400456000100 000037000000</td>
</tr>
<tr>
<td>016305100200 016305000500 00000066500 00000066500</td>
</tr>
</tbody>
</table>

Connect Fault (21)
At: 456|16305 >udd>кроap>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:

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmpb</td>
<td>53676 000 000 066 500</td>
</tr>
<tr>
<td>cmpb</td>
<td>53677 000 000 066 500</td>
</tr>
</tbody>
</table>

EIS Pointers and Lengths

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>210</td>
</tr>
<tr>
<td>000400000000 000400000000 043617410030 010477777760</td>
</tr>
<tr>
<td>151040000030 000000000000 054156000000 000000000077</td>
</tr>
</tbody>
</table>

F88, 10-9
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 >sll>bound_sss_wired_$plll_operators_11362
PR1 (ab) - 104|244 scs|244
PR2 (bp) - 244|4000 >pdd> B1LbhbjbBBBBBB>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> B1LbhbjbBBBBBB>stack_4|4000
PR7 (sb) - 244|0 >pdd> B1LbhbjbBBBBBB>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 - 00040000000 ($CON A)

SCU Data at pds|170

170 400253170041 000000000021 400253000200 000007000000
000250500200 000246000500 600101116100 600101116100

Connect Fault (21)
At: 253|250 >sll>bound_command_loop_$ipc_fast_1206
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

F88, 10-10
Fault Type: Connect

Machine Conditions For System Trouble Data At prdsl240
Time Stored - 03/17/83 1533.7 hfh Thu (111566604603067407)

Pointer Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0 (ap)</td>
<td>prdsl3176</td>
</tr>
<tr>
<td>PR1 (ab)</td>
<td>scs</td>
</tr>
<tr>
<td>PR2 (bp)</td>
<td>bound_priv_l$privileged_mode_ut</td>
</tr>
<tr>
<td>PR3 (bb)</td>
<td>sst_segl0</td>
</tr>
<tr>
<td>PR4 (lp)</td>
<td>ws_linkage</td>
</tr>
<tr>
<td>PR5 (lb)</td>
<td>ws_linkage</td>
</tr>
<tr>
<td>PR6 (sp)</td>
<td>prdsl3120</td>
</tr>
<tr>
<td>PR7 (sb)</td>
<td>prdsl0</td>
</tr>
</tbody>
</table>

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 prdsl270

270  000036750021  000000000021  400300000000  000000440000  
     000247200200  001266000700  000061015200  000000235007

Connect Fault (21)
At: 36|247 bound_priv_l$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  1da  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

<table>
<thead>
<tr>
<th>Register</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0 (ap)</td>
<td>76</td>
</tr>
<tr>
<td>PR1 (ab)</td>
<td>76</td>
</tr>
<tr>
<td>PR2 (bp)</td>
<td>75</td>
</tr>
<tr>
<td>PR3 (bb)</td>
<td>113</td>
</tr>
<tr>
<td>PR4 (lp)</td>
<td>15</td>
</tr>
<tr>
<td>PR5 (lb)</td>
<td>15</td>
</tr>
<tr>
<td>PR6 (sp)</td>
<td>76</td>
</tr>
<tr>
<td>PR7 (sb)</td>
<td>76</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>210</td>
</tr>
<tr>
<td>000032750021</td>
</tr>
<tr>
<td>000000000021</td>
</tr>
<tr>
<td>000041000100</td>
</tr>
<tr>
<td>000000044000</td>
</tr>
<tr>
<td>002505102200</td>
</tr>
<tr>
<td>002436000700</td>
</tr>
<tr>
<td>00063015200</td>
</tr>
<tr>
<td>777777710204</td>
</tr>
</tbody>
</table>

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=OGB)

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 value 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 pdsl60
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 pdsl110

110 000032550022 004000000051 000032000017 000000000000 01013410200 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, euf1, ^bar
APU Status: priv, sdwam, 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 (111512460154130102)

Pointer Registers

PR0 (ap) - 322|7120 >s1l>bound_sss_wired_$p11_operators_|1362
PR1 (ab) - 244|0 >pdd> zzzzzzzbBBBBBBB>stack_4|0
PR2 (bp) - 457|26353 >t>bound_oprcons_$mrdf_util_|1
PR3 (bb) - 244|774602 >pdd> zzzzzzzbBBBBBBB>stack_4|774602
PR4 (lp) - 0|0 dseg|0
PR5 (lb) - 244|774532 >pdd> zzzzzzzbBBBBBBB>stack_4|774532
PR6 (sp) - 244|774620 >pdd> zzzzzzzbBBBBBBB>stack_4|774620
PR7 (sb) - 244|774550 >pdd> zzzzzzzbBBBBBBB>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_$mrdf_util_|3453
Referencing: 0|1350 dseg|1350
On: cpu b (#1)
Indicators: ^bar
APU Status: sd-on, pt-on, sdwp
CU Status:
CT Hold: d1
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 pdsl140
Time Stored - 03/11/83  1704.2 hfh Fri (111557226651237015)

Pointer Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0 (ap)</td>
<td>40</td>
<td>7120</td>
</tr>
<tr>
<td>PR1 (ab)</td>
<td>104</td>
<td>244</td>
</tr>
<tr>
<td>PR2 (bp)</td>
<td>230</td>
<td>45263</td>
</tr>
<tr>
<td>PR3 (bb)</td>
<td>77777</td>
<td>1741</td>
</tr>
<tr>
<td>PR4 (lp)</td>
<td>15</td>
<td>1616</td>
</tr>
<tr>
<td>PR5 (lb)</td>
<td>230</td>
<td>45262</td>
</tr>
<tr>
<td>PR6 (sp)</td>
<td>240</td>
<td>600</td>
</tr>
<tr>
<td>PR7 (sb)</td>
<td>240</td>
<td>200</td>
</tr>
</tbody>
</table>

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 (00B)

SCU Data at pdsl170

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>170</td>
<td>000145052001 000004000051 47777700206 00000000000</td>
</tr>
<tr>
<td></td>
<td>003427100200 001741000400 300000100440 040140100540</td>
</tr>
</tbody>
</table>

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 (r1),(),fill(300)
536077 040 140 100 540  mlr (pr,r1),(pr,r1),fill(040)

EIS Pointers and Lengths

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>210</td>
<td>000400000000 000400000000 001741000070 002000000170</td>
</tr>
<tr>
<td></td>
<td>001250000030 000000000000 002001000000 00007777735</td>
</tr>
</tbody>
</table>

F88, 10-17
Fault Type: Access Violation

Machine Conditions For Fim Data At pdsl60
Time Stored - 02/10/83 1230.3 hfh Thu (111512460154134307)

Pointer Registers

PR0 (ap) - 244|775732 >pdd> zzzzzzbBBBBBB>stack_4|775732
PR1 (ab) - 244|775240 >pdd> zzzzzzbBBBBBB>stack_4|775240
PR2 (bp) - 321|14521 >sll>bound_sss_active_$sct_manager_115
PR3 (bb) - 244|776040 >pdd> zzzzzzbBBBBBB>stack_4|776040
PR4 (lp) - 0|0 dseg|0
PR5 (lb) - 244|775140 >pdd> zzzzzzbBBBBBB>stack_4|775140
PR6 (sp) - 244|775500 >pdd> zzzzzzbBBBBBB>stack_4|775500
PR7 (sb) - 244|0 >pdd> zzzzzzbBBBBBB>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 (008)

SCU Data at pdsl110

110 400322150201 000004000051 400244000100 000000000000
017277100200 776060000000 300020652100 300020652100

Access Violation Fault (51), Out of Segment Bounds
By: 322|17277 >sll>bound_sss_wired_$pl1_operators_111541
Referencing: 244|776060 >pdd> zzzzzzbBBBBBB>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

F88, 10-18
Fault Type: Access Violation

Machine Conditions For Fim Data At pdsl60
Time Stored - 02/09/83 2212.7 hfh Wed (111511660666116070)

Pointer Registers

PR0 (ap) - 270|7120 >sll>bound_sss_wired$p11_operators|1362
PR1 (ab) - 244|143701(6) >pdd> BLLBLDPBBBBBB>stack 4|143701
PR2 (bp) - 244|260000 >pdd> BLLBLDPBBBBBB>stack 4|260000
PR3 (bb) - 244|120740(3) >pdd> BLLBLDPBBBBBB>stack 4|120740
PR4 (lp) - 257|30034 >pdd> BLLBLDPBBBBBB>BBBJmgffILJXLPh.area.linker|30034
PR5 (lb) - 244|71220 >pdd> BLLBLDPBBBBBB>stack 4|71220
PR6 (sp) - 244|257640 >pdd> BLLBLDPBBBBBB>stack 4|257640
PR7 (sb) - 244|106140 >pdd> BLLBLDPBBBBBB>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 pdsl110

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)

F88, 10-19
Fault Type: Access Violation

Machine Conditions For Pim Data At pd8|60
Time Stored - 02/10/83 1230.3 hfh Thu (111512460156271332)

Pointer Registers

PR0 (ap) - 270|7120 >sll>bound_sss_wired_$p11_operators_11362
PR1 (ab) - 76|336 prds|336
PR2 (bp) - 244|26020 >pdd> CBBBLzbBBBBBB>stack_4|26020
PR3 (bb) - 113|0 tc_data|0
PR4 (lp) - 14|12276 as_linkage|12276
PR5 (ib) - 75|3714 pds|3714
PR6 (sp) - 244|26020 >pdd> CBBBLzbBBBBBB>stack_4|26020
PR7 (sb) - 244|0 >pdd> CBBBLzbBBBBBB>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 (00B)

SCU Data at pd8|110

  110 000040250401 000020000051 400301000100 00000672000
       013436000200 001266010200 001266610000 600076757100

Access Violation Fault (51), Ring Alarm
By: 40|15436 bound_sss_wired_$p11_operators_17700
Referencing: 301|1266 >sll>bound_ipc_$ipc_rea1_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 pdsl0
Time Stored - 02/09/83 2212.7 hfh Wed (111511660647626100)

Pointer Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0  (ap)</td>
<td>32217120</td>
</tr>
<tr>
<td>PR1  (ab)</td>
<td>24412462</td>
</tr>
<tr>
<td>PR2  (bp)</td>
<td>375133714</td>
</tr>
<tr>
<td>PR3  (bb)</td>
<td>4331753</td>
</tr>
<tr>
<td>PR4  (bp)</td>
<td>375134352</td>
</tr>
<tr>
<td>PR5  (bp)</td>
<td>24412374</td>
</tr>
<tr>
<td>PR6  (sp)</td>
<td>24412520</td>
</tr>
<tr>
<td>PR7  (sb)</td>
<td>4330</td>
</tr>
</tbody>
</table>

Processor Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>X0</td>
<td>33730 X1</td>
</tr>
<tr>
<td>A Register</td>
<td>0000000000104</td>
</tr>
<tr>
<td>Timer Register</td>
<td>000154400</td>
</tr>
<tr>
<td>SCU masks</td>
<td>031460000014 631460000007</td>
</tr>
<tr>
<td>Fault Register</td>
<td>010000000000 (00B)</td>
</tr>
</tbody>
</table>

SCU Data at pdsl0

```
30  400375150201 000000000043 400433000200 000000156000 034305000240 002001000440 300000100440 040100100540
```

(DF1) Page Fault (43)
By: 375134305 >t>bound_oprcons$write_log |423
Referencing: 433 |2001 >sc1>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 00000000040 012056000000 000000000077
```

F88, 10-22
Fault Type: Command

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: (DFO) Segment

16 (DFO) 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.
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 pdsl140
Time Stored - 02/09/83  2212.7 hfh Wed (111511660664301450)

Pointer Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0 (ap)</td>
<td>270</td>
<td>7120</td>
</tr>
<tr>
<td>PR1 (ab)</td>
<td>374</td>
<td>35642</td>
</tr>
<tr>
<td>PR2 (bp)</td>
<td>244</td>
<td>13406</td>
</tr>
<tr>
<td>PR3 (bb)</td>
<td>244</td>
<td>13300</td>
</tr>
<tr>
<td>PR4 (lp)</td>
<td>257</td>
<td>35762</td>
</tr>
<tr>
<td>PR5 (lb)</td>
<td>257</td>
<td>5552</td>
</tr>
<tr>
<td>PR6 (sp)</td>
<td>244</td>
<td>13300</td>
</tr>
<tr>
<td>PR7 (sb)</td>
<td>244</td>
<td>0</td>
</tr>
</tbody>
</table>

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 pdsl170

170  400374250041  000000000061  400257000046  000001462000  051501100200  036270000020  036270352020  400306352120

(FT2) Linkage Fault (61)
By: 374|51501 >exl>continuum>bound_continuum_$continuum_command_1|361
Referencing: 257|36270 >pdd> BNLBDfBBBBBBB> BBBJgfjDqdxCN.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,*

F88, 10-26
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. 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 pdsl60
Time Stored - 02/12/83 0312.5 hfe Sat (111514443375723610)

Pointer Registers

PR0 (ap) - 40|7120     bound_sss_wired_$p11_operators_11362
PR1 (ab) - 240|4512     >s11>stack_0.019|4512
PR2 (bp) - 240|1540     >s11>stack_0.019|1540
PR3 (bb) - 227|16006     tty_buf|16006
PR4 (lp) - 122|31131     bound_355_wired$tty_space_man|525
PR5 (lb) - 46|10       dn355_data|10
PR6 (sp) - 240|3140     >s11>stack_0.019|3140
PR7 (sb) - 240|1400     >s11>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 - 0000000000137 E Register - 0
Timer Register - 777766410 Ring Alarm Register - 1

SCU Data at pdsl110

    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,q1
565
Fault Type: Startup

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 turned 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 pdsl60
Time Stored - 03/06/83  1343.4 hfh Sun (111550652327420005)

Pointer Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0</td>
<td>76</td>
</tr>
<tr>
<td>PR1</td>
<td>104</td>
</tr>
<tr>
<td>PR2</td>
<td>75</td>
</tr>
<tr>
<td>PR3</td>
<td>75</td>
</tr>
<tr>
<td>PR4</td>
<td>15</td>
</tr>
<tr>
<td>PR5</td>
<td>76</td>
</tr>
<tr>
<td>PR6</td>
<td>76</td>
</tr>
<tr>
<td>PR7</td>
<td>76</td>
</tr>
</tbody>
</table>

Processor Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>X0</td>
<td>46</td>
</tr>
<tr>
<td>X1</td>
<td>0</td>
</tr>
<tr>
<td>X2</td>
<td>124155</td>
</tr>
<tr>
<td>X3</td>
<td>0</td>
</tr>
<tr>
<td>X4</td>
<td>0</td>
</tr>
<tr>
<td>X5</td>
<td>127</td>
</tr>
<tr>
<td>X6</td>
<td>0</td>
</tr>
<tr>
<td>X7</td>
<td>777774</td>
</tr>
</tbody>
</table>

SCU Data at pdsl110

```
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 pdsl140
Time Stored - 03/29/83 1507.9 hfe Tue (111605613135576571)

Pointer Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0 (ap)</td>
<td>40</td>
</tr>
<tr>
<td>PR1 (ab)</td>
<td>240</td>
</tr>
<tr>
<td>PR2 (bp)</td>
<td>76</td>
</tr>
<tr>
<td>PR3 (bb)</td>
<td>172</td>
</tr>
<tr>
<td>PR4 (lp)</td>
<td>15</td>
</tr>
<tr>
<td>PR5 (lb)</td>
<td>144</td>
</tr>
<tr>
<td>PR6 (sp)</td>
<td>240</td>
</tr>
<tr>
<td>PR7 (sb)</td>
<td>152</td>
</tr>
</tbody>
</table>

Processor Registers

X0 - 2237 X1 - 4504 X2 - 2521 X3 - 777671 X4 - 2 X5 - 520 X6 - 514 X7 - 2747
A Register - 0000000000000000 Q Register - 0001721013277 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 pdsl170

<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>170</td>
<td>000032550023 00000130003 000240000217 000000000000</td>
</tr>
<tr>
<td></td>
<td>002747100200 013300000000 00000710217 000476710220</td>
</tr>
</tbody>
</table>

Store Fault (3)
Illegal Action Code (13) - Store Not Ready
By: 32|2747 bound_interceptors$fim_util|315
Referencing: 240|13300 >s1l>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

PRO (ap) - 270|7120
PR1 (ab) - 244|104760
PR2 (bp) - 244|106760
PR3 (bb) - 422|61224
PR4 (lp) - 257|54074
PR5 (lb) - 427|367314
PR6 (sp) - 244|102620
PR7 (sb) - 427|17524

Processor Registers

X0 - 1757  X1 - 102104  X2 - 32  X3 - 777732  X4 - 2  X5 - 0  X6 - 61  X7 - 2140
A Register - 000000000000  Q Register - 777777777777  E Register - 0
Timer Register - 000037651  Ring Alarm Register - 1
Fault Register - 010000000000  (OOB)

SCU Data at pds|110

110  0000032550022  000000130077  000000000000  000000000000  000000000000  002747100200  013300000000  000000000000  000000000000

Trouble Fault (77)
Illegal Action Code (13) - Store Not Ready
By: 32|2747  bound_interceptors$fim_utilI315
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 - 035447000000 017770000024
Mode Register - 000000000000
Cache Mode Register - 177532005003

SCU Data

3540 000032450001 000000000015 000032000000 000000000000
003146101200 000000000000 00000002200 0000002200

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 00000566000
     02651101200 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 000007777735

F88, 10-44
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 pdsl|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 - 1200000106664 Q Register - 000000000000 E Register - 0
Timer Register - 000044167 Ring Alarm Register - 0
Fault Register - 400000000000 (ILL OP)

SCU Data at pdsl|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 1dx0 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

F88, 10-47
Fault Type: Overflow

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

<table>
<thead>
<tr>
<th>Register</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>PR0 (ap)</td>
<td>40</td>
</tr>
<tr>
<td>PR1 (ab)</td>
<td>230</td>
</tr>
<tr>
<td>PR2 (bp)</td>
<td>240</td>
</tr>
<tr>
<td>PR3 (bb)</td>
<td>240</td>
</tr>
<tr>
<td>PR4 (lp)</td>
<td>123</td>
</tr>
<tr>
<td>PR5 (lb)</td>
<td>240</td>
</tr>
<tr>
<td>PR6 (sp)</td>
<td>240</td>
</tr>
<tr>
<td>PR7 (sb)</td>
<td>240</td>
</tr>
</tbody>
</table>

Processor Registers

X0 - 33115 X1 - 3560 X2 - 777773 X3 - 500 X4 - 0 X5 - 4 X6 - 7 X7 - 500
A Register - 4000000000000000 Q Register - 0000000000000000 E Register - 0
Timer Register - 000115235 Ring Alarm Register - 0
SCU masks - 000240000043 003140000000

SCU Data at >sll>stack_0.011|4010

<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>4010</td>
<td>000123050011 000000000033 0002440000200 000000000000</td>
</tr>
<tr>
<td></td>
<td>014244340200 004136000000 000033735000 000011236007</td>
</tr>
</tbody>
</table>

Overflow Fault (33)
By: 123|14244 bound_uncp_wired
Referencing: 244|4136 >pdd> BxbBDXbBBBBBB>stack_4|4136
On: cpu c (#2)
Indicators: neg, cary, ovf1, ^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

### IOM NUMBER

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>12</td>
<td>13</td>
<td>14</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>21</td>
<td>22</td>
</tr>
<tr>
<td>6</td>
<td>24</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td>7</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
</tbody>
</table>

- Special interrupt: device condition not resulting from I/O request.
DUMP ANALYSIS
METODOLOGY

Check dump circumstances (written)

Oper. Error? yes → FIX OPERATIONAL PROCEDURES

Oper. Error? no →

Determine Crash type

Prohibited Fault →

Syserr →

Execute Fault →

Execute Switches →

Read flagbox msg

Crawl-out? no →

terminate

Read Error Msg

Documentation

Examine sys_trouble MCs

Examine dump registers

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.

Hdw miscalc? yes →

Get other info: logs prev. faults, AMs...

Determine possible broken boxes.

Correlate with info of earlier problems

FIX HARDWARE

no →

Get prgm listing, trace back to cause

yes →

Determine possibilities of earlier problems

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 aptep 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 ppc emptied bit (1), /* ON if process is being preempted 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 preempt_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 dba 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 initialized */
   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 */
   2 ts fixed bin (35), /* bit 18-35: sequential number */
   2 ts fixed bin (35), /* virtual time since eligibility award */
   2 ti fixed bin (35), /* virtual time since scheduling */
   2 timax fixed bin (35), /* virtual time since interaction */
   2 preempt bit (18), /* maximum value allowed for apte.ti */
   2 byte pointers unaligned, /* relative pointer to ITT list */
   3 event_thread bit (18), /* IPS signals pending */
   3 pad3 bit (18), /* relative ASTE pointers */
   2 ips_message bit (36), /* PSD (per-process) */
   2 aspecs unaligned, /* DSEG (per-process) */
   3 pds bit (18), /* PRDS (per-processor) */
   3 dseg bit (18), /* at call to getwork (return point in pxss) */
   3 prds bit (18), /* process to send wakeup at termination */
   2 savex7 bit (18) unaligned, /* * */
   2 term_processid bit (36), /* * */

/* * */
2 apad (10) fixed bin (35);

/* END INCLUDE FILE ... apte.incl.pl1 */
BEGIN INCLUDE FILE ...aste.incl.pl1 ... */

#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),
2 bp bit (18),
2 infl bit (18),
2 infp bit (18),
2 strp bit (18),
2 par_astrp bit (18),
2 uid bit (36),
2 msl bit (9),
2 pvtx fixed bin (8),
2 vtoex fixed bin (17),
2 usedf bit (1),
2 init bit (1),
2 gtus bit (1),
2 gtm us bit (1),
2 ho bit (1),
2 ho_sdw bit (1),
2 any_access_on bit (1),
2 write_access_on bit (1),
2 inhibit_cache bit (1),
2 explicit_deact_ok bit (1),
2 deact_error bit (1),
2 ho_part bit (1),
2 fm_damaged bit (1),
2 multi_class bit (1),
2 pdai bit (2),
2 dius bit (1),
2 nid bit (1),
2 dmpr_pad bit (1),
2 ehb_pad bit (1),
2 nqsw bit (1),
2 dirw bit (1),
2 master_dir bit (1),
2 volmap_seg bit (1),

/* forward used list rel pointer */
/* backward used list rel pointer */
/* ptr to NEXT in list of ASTR's of my brothers */
/* ptr to FIRST in list of ASTR's of my children */
/* rel pointer to process trailer */
/* rel pointer to parent aste */
/* segment unique id */
/* maximum segment length in 1024 word units */
/* physical volume table index */
/* vto entry index */
/* ast entry is being used if non-zero */
/* used bit - insure 1 lap */
/* global transparent usage switch */
/* global transparent modified switch */
/* hard core segment */
/* ast with sdw for hardcore seg if non-zero */
/* any sdw allows access, unless write_access_on */
/* any sdw allows write access */
/* any sdw allows write access */
/* flag not to reset above bits */
/* set if user can deactivate seg */
/* set if error occurred while deactivating */
/* set if pages are in a hardcore partition */
/* set if filemap checksum was ever bad */
/* set if page_control should watch state changes to this segment */
/* set if filemap checksum was ever bad */
/* entry hold switch */
/* no quota switch - no checking for pages of this seg */
/* directory switch */
/* master dir - a root for the log volume */
/* volmap_seg for some volume */
2 rqsw (0:1) bit (1),
2 pad1c bit (10),
2 dtu bit (36),
2 dtm bit (36),
2 quota (0:1) fixed bin (18) unsigned,
2 used (0:1) fixed bin (18) unsigned,
2 csl bit (9),
2 fms bit (1),
2 npfs bit (1),
2 dnpb bit (1),
2 per_process bit (1),
2 dnpb bit (1),
2 pad2 bit (2),
2 records bit (9),
2 np bit (9),
2 hr_fp bit (18),
2 fms变化 bit (1),
2 damaged bit (1),
2 pack_ovfl bit (1),
2 synchronized bit (1),
2 pad3 bit (6),
2 psei bit (2),
2 marker bit (6)) unaligned;

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

dcl 1 asta_part aligned based (astep),
2 one bit (36) unaligned,
2 two bit (36*11 - 8) unaligned,
2 three bit (8) unaligned;

dcl 1 asta_part aligned based (astep),
2 one bit (36) unaligned,
2 two bit (36*11 - 8) unaligned,
2 three bit (8) unaligned;

dcl 1 seg_aste based (astep) aligned,
2 pad1 bit (8*36),
2 usage fixed bin (35),
2 pad2 bit (3*36);

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

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

dol 1 dump based (dumpptr) aligned,
   2 dump_header aligned like dump_header,
      2 segs (1008),
         3 segno bit (18) unal,
         3 length bit (18) unal,
      2 amptwregs (0 : 63) bit (36),
      2 amptwptrs (0 : 63) bit (36),
      2 amsdwregs (0 : 63) bit (72),
      2 amsdwptrs (0 : 63) bit (36),
      2 ouihist (0 : 63) bit (72),
      2 ouihist (0 : 63) bit (72),
      2 duhist (0 : 63) bit (72),
      2 duhist (0 : 63) bit (72),
      2 duhist (0 : 63) bit (72),
      2 prs (0 : 7) ptr,
      2 regs aligned like dump_registers,
         2 low_order_port bit (3),
         2 pad4 bit (36),
         2 motime fixed bin (32),
         2 pad5 (0 : 3) bit (36),
      2 misc_registers like dump_misc_registers,
      2 ptrlen (0 : 7) bit (36),
      2 coreblocks (0 : 7),
         3 num_first bit (18) unal,
         3 num_blocks bit (18) unal,
         2 pad7 (112) fixed bin;

dcl 1 dump_header aligned based,
   2 words_dumped fixed bin (35),
   2 valid bit (1),
   2 time fixed bin (71),
   2 mctime fixed bin (52),
   2 pad5 (0 : 3) bit (36),
   2 misc_registers like dump_misc_registers,
   2 ptrlen (0 : 7) bit (36),
   2 coreblocks (0 : 7),
      3 num_first bit (18) unal,
      3 num_blocks bit (18) unal,
      2 pad7 (112) fixed bin;
   2 ouihist (0 : 63) bit (72),
   2 ouihist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 prs (0 : 7) ptr,
   2 regs aligned like dump_registers,
      2 low_order_port bit (3),
      2 pad4 bit (36),
      2 motime fixed bin (32),
      2 pad5 (0 : 3) bit (36),
   2 misc_registers like dump_misc_registers,
   2 ptrlen (0 : 7) bit (36),
   2 coreblocks (0 : 7),
      3 num_first bit (18) unal,
      3 num_blocks bit (18) unal,
      2 pad7 (112) fixed bin;
   2 ouihist (0 : 63) bit (72),
   2 ouihist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 prs (0 : 7) ptr,
   2 regs aligned like dump_registers,
      2 low_order_port bit (3),
      2 pad4 bit (36),
      2 motime fixed bin (32),
      2 pad5 (0 : 3) bit (36),
   2 misc_registers like dump_misc_registers,
   2 ptrlen (0 : 7) bit (36),
   2 coreblocks (0 : 7),
      3 num_first bit (18) unal,
      3 num_blocks bit (18) unal,
      2 pad7 (112) fixed bin;
   2 ouihist (0 : 63) bit (72),
   2 ouihist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 prs (0 : 7) ptr,
   2 regs aligned like dump_registers,
      2 low_order_port bit (3),
      2 pad4 bit (36),
      2 motime fixed bin (32),
      2 pad5 (0 : 3) bit (36),
   2 misc_registers like dump_misc_registers,
   2 ptrlen (0 : 7) bit (36),
   2 coreblocks (0 : 7),
      3 num_first bit (18) unal,
      3 num_blocks bit (18) unal,
      2 pad7 (112) fixed bin;
   2 ouihist (0 : 63) bit (72),
   2 ouihist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 prs (0 : 7) ptr,
   2 regs aligned like dump_registers,
      2 low_order_port bit (3),
      2 pad4 bit (36),
      2 motime fixed bin (32),
      2 pad5 (0 : 3) bit (36),
   2 misc_registers like dump_misc_registers,
   2 ptrlen (0 : 7) bit (36),
   2 coreblocks (0 : 7),
      3 num_first bit (18) unal,
      3 num_blocks bit (18) unal,
      2 pad7 (112) fixed bin;
   2 ouihist (0 : 63) bit (72),
   2 ouihist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 duhist (0 : 63) bit (72),
   2 prs (0 : 7) ptr,
   2 regs aligned like dump_registers,
      2 low_order_port bit (3),
      2 pad4 bit (36),
      2 motime fixed bin (32),
      2 pad5 (0 : 3) bit (36),
   2 misc_registers like dump_misc_registers,
   2 ptrlen (0 : 7) bit (36),
   2 coreblocks (0 : 7),
      3 num_first bit (18) unal,
      3 num_blocks bit (18) unal,
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>erfno</td>
<td>Error Report Form Number</td>
</tr>
<tr>
<td>num_segs</td>
<td>Number of segments dumped</td>
</tr>
<tr>
<td>valid_355</td>
<td>Indicates which 355s were dumped</td>
</tr>
<tr>
<td>dumped_355s</td>
<td>Time of 355 dump</td>
</tr>
<tr>
<td>time_355</td>
<td>Currently 2</td>
</tr>
<tr>
<td>pad0</td>
<td>Pad to 16 words</td>
</tr>
<tr>
<td>num_segs</td>
<td>Number of segments dumped</td>
</tr>
<tr>
<td>valid_355</td>
<td>Indicates which 355s were dumped</td>
</tr>
<tr>
<td>dumped_355s</td>
<td>Time of 355 dump</td>
</tr>
<tr>
<td>time_355</td>
<td>Currently 2</td>
</tr>
<tr>
<td>pad0</td>
<td>Pad to 16 words</td>
</tr>
</tbody>
</table>

/* Error Report Form Number */
/* Number of segments dumped */
/* Indicates which 355s were dumped */
/* Time of 355 dump */
/* Currently 2 */
/* Pad to 16 words */

```c
/* Error Report Form Number */
/* Number of segments dumped */
/* Indicates which 355s were dumped */
/* Time of 355 dump */
/* Currently 2 */
/* Pad to 16 words */
```

```c
/* Error Report Form Number */
/* Number of segments dumped */
/* Indicates which 355s were dumped */
/* Time of 355 dump */
/* Currently 2 */
/* Pad to 16 words */
```
2 misc_registers aligned like dump_m misc_registers, /* Assorted registers */

2 pad5 bit (36),
2 ptrlen (0 : ?) bit (36),
2 pad6 (15) bit (36),
2 low_order_port bit (3),
2 coreblocks (0 : 7),
3 num_first bit (18) unal,
3 num_blocks bit (18) unal;

#include <stdio.h>

#endif

/* END INCLUDE FILE ... bos_dump.incl.pll ... */
/* 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 cmep based (cmep) aligned,
   2 fp bit (18) unaligned,
   2 bp bit (18) unaligned,
   2 devadd bit (22) unaligned,
   2 pad5 bit (1) unaligned,
   2 synch_held bit (1) unaligned,
   2 io bit (1) unaligned,
   2 pad2 bit (1) unaligned,
   2 ex bit (1) unaligned,
   2 removing bit (1) unaligned,
   2 abs_v bit (1) unaligned,
   2 abs_usable bit (1) unaligned,
   2 notify_requested bit (1) unaligned,
   2 pad3 bit (1) unaligned,
   2 phm_hedge bit (1) unaligned,
   2 contr bit (3) unaligned,
   2 ptwp bit (18) unaligned,
   2 astep bit (18) unaligned,
   2 pin_counter fixed bin (17) unaligned,
   2 synch_page_entryp bit (18) unaligned;

   /* Core map entry */
dcl 1 cmme (0: 1) based aligned like cmep;
   /* core map entry for extracting DID */
dcl 1 cmme based (cmep) aligned,
   2 pad bit (36) unaligned,
   2 record_no bit (18) unaligned,
   2 add_type bit (4) unaligned,
   2 flags bit (14) unaligned,
   2 pad1 bit (18) unaligned;
   /* record number of device */
   /* see add_type.incl.pl1 */

   /* pointer to page table word for the page */
   /* relative AST entry pointer of page */
   /* number of times to skip eviction */
   /* relp to synch page entry */

   /* pointer to core map entry */
   /* forward pointer to next entry */
   /* backward pointer to previous entry */

   /* device address of page in the core block */
   /* Page of synchronized seg held in memory */
   /* input/output indicator 1= output, 0=input */

   /* indicates error in previous IO activity */
   /* core is being removed by reconfiguration */
   /* absolute address must not be changed for page */
   /* page may be assigned with fixed absolute address */
   /* notify requested on I/O completion */

   /* on -> pc$flush_core ought write. */
   /* controller in which core block is located */

   /* on page */
   /* controller */

   /* Core map array */

/* 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,
  2 frp bit(18) unaligned,
  2 brp bit(18) unaligned,
  2 type bit(18) unaligned,
  2 size fixed bin (17) unaligned,
  2 name unaligned,
  3 pers_rp bit(18) unaligned,
  3 proj_rp bit(18) unaligned,
  3 tag char(1) unaligned,
  2 mode bit (3) unaligned,
  2 pad24 bit(24) unaligned,
  2 ex_mode bit (36),
  2 checksum bit (36),
  2 owner bit (36); /* person or project name on ACL. Length = 14 words. */
dcl 1 access_name aligned based,
  2 frp bit(18) unaligned,
  2 brp bit(18) unaligned,
  2 type bit(18) unaligned,
  2 size fixed bin (17) unaligned,
  2 usage fixed bin(17) unaligned,
  2 salv_flag fixed bin(17) unaligned,
  2 pad1 bit (36),
  2 name char(32) aligned,
  2 checksum bit (36),
  2 owner bit (36);

/* END INCLUDE FILE ... dir_acl.incl.pl1 */
BEGIN INCLUDE FILE ... dir_allocation_areas.incl.pl1 ... last modified December 1973 */

dcl areap ptrl;

dcl 1 area based (areap) aligned,
2 nsizes fixed bin (18),
2 lw fixed bin (18),
2 array (100) aligned,
3 fptr bit (18) unaligned,
3 size fixed bin (17) unaligned;

/* Number of types. */
/* Next available word in area. */
/* Last usable word. */
/* Array of types. */
/* Free pointer for this size. */
/* Size. */

END INCLUDE FILE ... dir_allocation_areas.incl.pl1 */
/* BEGIN INCLUDE FILE ... dir_entry.incl.pll ...last modified August 1974 for nss */

/* Template for an entry. Length = 38 words */
dol ep ptr;
dol 1 entry based (ep) aligned,

(2 afrc bit (18),
 2 ebrp bit (18)) unaligned,
/* forward rel ptr to next entry */
/* 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),
 2 pad0 bit (17),
 2 nnames fixed bin (17),
/* number of names for this entry */
2 name_fr bit (18),
2 name_brp bit (18),
/* rel pointer to start of name list */
/* rel pointer to end of name list */
2 author,
 3 pers_rp bit (18),
 3 proj_rp bit (18),
/* user who created branch */
/* name of user who created branch */
/* project of user who created branch */
3 tag char (1),
 3 pad1 char (3),
/* tag of user who created branch */
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 vtoox 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),
2 copysw bit (1),
2 safety_sw bit (1),
2 multiple_class bit (1),
2 audit_flag bit (1),
2 security_oosw bit (1),
2 encrypt_sw bit (1),
2 master_dir bit (1),
2 tpd bit (1),
2 pad4 bit (11),
2 encrypt_bound bit (14) unaligned,
2 access_class bit (72) aligned,
(2 ring_brackets (3) bit (3),
2 ex_ring_brackets (3) bit (3),
2 acl_count fixed bin (17),
2 acl_frp bit (18),
2 acl_brp bit (18),
2 bc_author,
3 pers_rp bit (18),
3 proj_rp bit (18),
3 tag char (1),
3 pad5 bit (2),
2 bc fixed bin (24) unaligned,
2 sons_lvid bit (36),
2 pad6 bit (36),
2 checksum bit (36),
2 owner bit (36);
/*
*  END INCLUDE FILE ... dir_entry.incl.pll ... */

/* indi
cates segment is per process */
/* = 1 make copy of segment whenever initiated */
/* if 1 then entry cannot be deleted */
/* segment has multiple security classes */
/* segment must be audited for security */
/* security out of service switch */
/* 1 if call limiter is to be enabled */
/* TRUE for master directory */
/* TRUE if this segment is never to go on the PD */
/* call limiter */
/* security attributes : level and category */
/* ring brackets on segment */
/* extended ring brackets */
/* number of entries on ACL */
/* rel ptr to start of ACL */
/* rel ptr to end of ACL */
/* user who last set the bit count */
/* name of user who set the bit count */
/* project of user who set the bit count */
/* tag of user who set the bit count */
/* bit count for segs, msf indicator for dirs */
/* logical volume id for immediat inf non dir seg */
/* checksum from dti */
/* uid of containing directory */
/
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),
2 type bit (18) unaligned,
2 size fixed bin (17) unaligned,
2 dtc (3),
3 date bit (36),
3 error bit (36),
2 uid bit (36),
2 pvid bit (36),
2 sons_lvid bit (36),
2 access_class bit (72),
(2 vtox fixed bin (17),
2 version_number fixed bin (17),
2 entryfrp bit (18),
2 pad2 bit (18),
2 entrybrp bit (18),
2 pad3 bit (18),
2 pers_frp bit (18),
2 proj_frp bit (18),
2 pers_brp bit (18),
2 proj_brp bit (18),
2 seg_count fixed bin (17),
2 dir_count fixed bin (17),
2 lcount fixed bin (17),
2 acle_total fixed bin (17),
2 arearp bit (18),
2 per_process_sw bit (1),

2 /* Process ID of last modifier */
2 /* type of object = dir header */
2 /* size of header in words */
2 /* date-time checked by salvager array */
2 /* the date */
2 /* what errors were discovered */
2 /* uid of the directory - copied from branch */
2 /* phys vol id of the dir - copied from branch */
2 /* log vol id for inf non dir seg - copied from branch */
2 /* security attributes of dir - copied from branch */
2 /* vtox entry index of the dir - copied from branch */
2 /* version number of header */
2 /* rel ptr to beginning of entry list */
2 /* rel ptr to end of entry list */
2 /* rel ptr to start of person name list */
2 /* rel ptr to start of project name list */
2 /* rel ptr to end of person name list */
2 /* rel ptr to end of project name list */
2 /* number of non-directory branches */
2 /* number of directory branches */
2 /* number of links */
2 /* total number of ACL entries in directory */
2 /* relative pointer to beginning of allocation area */
2 /* indicates dir contains per process segments */
/* TRUE if this is a master dir */
/* TRUE if segs must be on RPV */
/* TRUE if hash table is being constructed */

/* number of initial acl entries for segs */
/* number of initial acl entries for dir */

/* pointer to initial ACLs for each ring */
/* rel ptr to start of initial ACL for segs */
/* rel ptr to end of initial ACL for segs */

/* rel ptr to start of initial for dirs */
/* rel ptr to end of initial ACL for dirs */

/* size of hash table */
/* no. of used places in hash table */

/* number of levels from root of this dir */

/* date-time directory last salvaged */

/* uid of superior master dir */
/* up one each call to sum$dirmod */
/* pad to make it a 64 word header */
/* checksummed from uid on */
/* uid of parent dir */

dc1 version_number_2 fixed bin int static options (constant) init (2);

/

END INCLUDE FILE ... dir_header.incl.pll */
/* BEGIN INCLUDE FILE ... dir_ht.incl.pll */

dcl htp ptr;

dcl 1 hash_table based (htp) aligned,
   2 modify bit (36) unal,
   2 type bit (18) unal,
   2 size fixed bin (17) unal,
   2 name rp (0:1) bit(18) unal,
   2 checksum bit (36) unal,
   2 owner bit (36) unal;

/* htp = ptr(dp,active_hardcore_data$htrp) */
/* type = dir hash table */
/* size of current dir hash table entry */
/* rel ptr of name entry */
/* otherwise rel ptr to name */

/* END INCLUDE FILE ... dir_ht.incl.pll */
/* BEGIN INCLUDE FILE ... dir_link.incl.p11 ... 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),
2 ebrp bit (18),
/* forward rel ptr to next entry */
/* 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),
/* number of names for this entry */
2 names fixed bin (17),
2 name_frp bit (18),
/* rel pointer to start of name list */
/* rel pointer to end of name list */
2 name_brp bit (18),
2 author,
/* user who created entry */
3 pers_rp bit (18),
/* name of user who created entry */
5 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.p11 */
Several arrays in this program are zero based. The zero-th entries are NEVER USED. Reference should start at 1, not lboud. 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,"qi */
dcl dir_lock_seg$ external static; /* name of the segment containing the directory locks */
dcl dir_lock_seg$p pointer; /* pointer to the dirlock table */
dcl 1 dir_lock_seg based (dir_lock_seg$p) 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,
  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
2 lock_count
    bit (34) unaligned,
    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
 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.pll */
/* BEGIN INCLUDE FILE ... dir_name.incl.pll ... last modified Nov 1975 for nss */

/* Template for names of branches or links. Length = 14 words. */
dcl np ptr;

dcl 1 names based aligned,
    2 fp bit(18) unaligned,
    2 bp bit(18) unaligned,
    2 type bit (18) unaligned,
    2 size fixed bin (17) unaligned,
    2 entry_rp bit(18) unaligned,
    2 ht_index fixed bin(17) unaligned,
    2 hash_thread bit (18) unal,
    2 pad3 bit (18) unal,
    2 name char(32) aligned,
    2 checksum bit (36),
    2 owner bit (36);

/* END INCLUDE FILE ... dir_name.incl.pll */
/* BEGIN INCLUDE FILE ... fault_vector.incl.pll ... last modified February 1981 */

dcl 1 fpv based (fpv) aligned,
    2 ipair (0: 31),
    3 scu bit (36),
    3 tra bit (36),
    2 ipair (0: 31),
    3 scu bit (36),
    3 tra bit (36),
    2 i_tra_ptr (0: 31) ptr,
    2 i_scu_ptr (0: 31) ptr,
    2 f_tra_ptr (0: 31) ptr,
    2 f_scu_ptr (0: 31) ptr;

/* pointer to the fault and interrupt vectors */

dcl fpv ptr;

dcl (FAULT_NO_SDF init (0), /* Shutdown */
    FAULT_NO_STR init (1), /* Store */
    FAULT_NO_MME init (2), /* Master Mode Entry */
    FAULT_NO_F1 init (3), /* Fault Tag */
    FAULT_NO_TRO init (4), /* Timer Runout */
    FAULT_NO_CMD init (5), /* Command */
    FAULT_NO_DRL init (6), /* Derail */
    FAULT_NO_LUP init (7), /* Lookup */
    FAULT_NO_CON init (8), /* Connect */
    FAULT_NO_PAR init (9), /* Parity */
    FAULT_NO_IPR init (10), /* Illegal Procedure */
    FAULT_NO_QMC init (11), /* Operation Not Complete */
    FAULT_NO_SUP init (12), /* Startup */
    FAULT_NO_OPL init (13), /* Overflow */
    FAULT_NO_DIV init (14), /* Divide Check */
    FAULT_NO_EXF init (15), /* Execute */
    FAULT_NO_DP0 init (16), /* Directed Fault 0 (Segment Fault) */
    FAULT_NO_DP1 init (17), /* Directed Fault 1 (Page Fault) */
    FAULT_NO_DP2 init (18), /* Directed Fault 2 */
    FAULT_NO_DP3 init (19), /* Directed Fault 3 */
    FAULT_NO_AGV 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_NO_TRB init (31), /* Trouble */

) fixed bin (17) int static options (constant):
/* END INCLUDE FILE ... fault_vector.incl.pl1 */
/* BEGIN INCLUDE FILE ... fgbx.incl.pll */
/* 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 flasbox$ ext;
dcl fgbxp ptr;

dcl 1 fgbx based (fgbxp) aligned,
  2 flags (36) bit (1) unal,
  2 slt_segno bit (18),
  2 pad1 fixed bin,
  2 rtb,
    3 assmb bit (1),
    3 call_bos bit (1),
    3 shut bit (1),
    3 mess bit (1),
    3 alert bit (1),
  3 pad bit (25),
  3 bos_entry fixed bin (5) unal,
  2 sentinel char (32) aligned,
  2 set_sdwr bit (72),
  2 hc_dbr bit (72),
  2 message char (64),
  2 fnp_blast char (128); /* communications switches */ /* segment # of the SLT */ /* return to BOS info */ /* "1b" if storage system enabled */ /* "1b" if BOS called by operator */ /* "1b" if BOS called after shutdown */ /* "1b" if message has been provided */ /* "1b" if audible alarm to be sounded */ /* type of entry into BOS
  0 => XED 10002 (BOS entry)
  1 => XED 10004 (Multics entry)
  2 => XED 10000 (manual entry) */ /* set by BOS (for now) */ /* set by init_sst */ /* set by start_cpu, idle DBR */ /* message for return to BOS */ /* 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.pll */
BEGIN INCLUDE FILE ... fs_vol_label.incl.pll .. last modified January 1982 for new volume map format */
/* This is the label at fixed location of each physical volume. Length 1 page */
/* Note: fsvolout clears pad fields before writing the label */

dcl labelp ptr;
dcl 1 label base (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,
2 mfg_serial char (32), /* Version 1 */
2 pv_name char (32), /* Manufacturer's serial number */
2 lv_name char (32), /* Physical volume name. */
2 pvid bit (36), /* Name of logical volume for pack */
2 lvid bit (36), /* Unique ID of this pack */
2 root_pvid bit (36), /* Unique ID of its logical vol */
2 time_registered fixed bin (71), /* unique ID of the pack containing the root. everybody must agree */
2 n_pv_in_lv fixed bin, /* time imported to system */
2 vol_size fixed bin, /* # phys volumes in logical */
2 vtoc_size fixed bin, /* total size of volume, in records */
2 not_used bit (1) unal, /* number of recs in fixed area + vtoc */
2 private bit (1) unal, /* used to be multiple_class */
2 flagged bit (34) unal, /* TRUE if was registered as private */
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 time_mounted fixed bin (71), /* time mounted */
2 time_vmap_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 padd 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 padda (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),
  3 root_vtocx fixed bin (35),
  3 shutdown_state fixed bin,
  3 pad7 bit (1) aligned,
  3 disk_table_vtoc fixed bin,
  3 disk_table_vtocx fixed bin,
  3 esd_state fixed bin,
2 volmap_record fixed bin,
2 size_of_volmap fixed bin,
2 vtoc_map_record fixed bin,
2 size_of_vtoc_map fixed bin,
2 volmap_unit_size fixed bin,
2 vtoc_origin_record fixed bin,
2 dumpper_bit_map_record fixed bin,
2 vol_trouble_count fixed bin,
2 pad3 (52) fixed bin,
2 nparts fixed bin,
2 parts (47),
  3 part char (4),
  3 frec fixed bin,
  3 nrec fixed bin,
  3 pad5 fixed bin,
2 pad4 (5*64) fixed bin;

/* TRUE if the root is on this pack */
/* VTOC index of root, if it is here */
/* Status of hierarchy */
/* VTOC index of disk table on RPV */
/* UID of disk table */
/* State of esd */
/* Begin record of volume map */
/* Number of records in volume map */
/* Begin record of VTOC map */
/* Number of records in VTOC map */
/* Number of words per volume map section */
/* Begin record of VTOC */
/* Begin record of dumpper bit-map */
/* Count of inconsistencies found since salvage */
/* Number of special partitions on pack */
/* Name of partition */
/* First record */
/* Number of records */
BEGIN INCLUDE FILE its.incl.pll
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, /* segment number within the pointer */
  2 segno bit (13) unaligned, /* ring number within the pointer */
  2 ringno bit (3) unaligned, /* should be 43(8) */
  2 pad2 bit (9) unaligned,
  2 its_mod bit (6) unaligned, /* word offset within the addressed segment */
  2 offset bit (18) unaligned,
  2 pad3 bit (3) unaligned,
  2 bit_offset bit (6) unaligned,
  2 pad4 bit (3) unaligned,
  2 mod bit (6) unaligned; /* bit offset within the word */
/* 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, /* should be 41(8) */
  2 itp_mod bit (6) unaligned, /* word offset from pointer register word offset */
  2 offset bit (18) unaligned,
  2 pad2 bit (3) unaligned,
  2 bit_offset bit (6) unaligned,
  2 pad3 bit (3) unaligned,
  2 mod bit (6) unaligned; /* bit offset relative to new word offset */
/* 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;
/* just like itp, but with unsigned binary where appropriate */
dcl 1 itp_unsigned based aligned,
  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.pll */
/* 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,
   3 dev_signal bit (18) unaligned, /* origin of event message */
     3 ring fixed bin (17) unaligned, /* 0 = user-event, 1 = device-signal */
   2 target_id bit (36), /* if user-event, sender's validation ring */
   2 channel_id fixed bin (71), /* target process' id */
   2 message  fixed bin (71); /* target process' event channel */
/* 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 (kstep), /* KST header declaration */
    2 lowseg fixed bin (17), /* lowest segment number described by kst */
    2 highest_used_segnr 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 kst */
    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) /* can use 256K entries */
    2 unused_2 bit (9) 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 */

    end_of_kst bit (36); /* KST entry declaration */

dcl 1 kste based (kstep) aligned, /* 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 entyp ptr unaligned, /* branch pointer */
    2 uid bit (36) aligned, /* unique identifier */
    2 access_information unaligned, /* date time branch modified */
    3 dtsm bit (36), /* extended access from the branch */
    3 access bit (3), /* rew */
    3 ex_rb (3) bit (3), /* ring brackets from branch */
    2 padd bit (3) unaligned, /* directory switch */
    2 flags unaligned, /* set if initiated with write permission */
    3 allow_write bit (1), /* privileged initiation */
    3 priv_init bit (1), /* transparent modification switch */
    3 tms bit (1), /* transparent usage switch */
    3 tus bit (1), /* transparent paging device switch */
    3 tpd bit (1), /* audit switch */
    3 audit bit (1), /* set if I am willing to have a user force deactivate */
    3 explicit_deact_ok bit (1), /* if dirsw then inferior count else lv index */
    3 pad bit (3),
    2 infcount fixed bin (12) unaligned;
/* 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 ("Ul000000000000000000000000000000000000000b");
   /* 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 isol1 (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 mcp based (mcp) aligned,
2 prs (0:7) ptr,
   (2 regs,
    3 x (0:7) bit (18),
    3 a bit (36),
    3 q bit (36),
    3 e bit (8),
    3 pad1 bit (28),
    3 t bit (27),
    3 pad2 bit (6),
    3 ralr bit (3),

2 scu (0:7) bit (36),
   2 mask bit (72),
   2 lps_temp bit (36),
   2 errcode fixed bin (35),
   2 fim_temp,
    3 unique_index bit (18) unal,
    3 resignal bit (1) unal,
    3 fcode bit (17) unal,
   2 fault_reg bit (36),
   2 pad2 bit (3),
   2 cpu_type fixed bin (2) unsigned,
   2 ext_fault_reg bit (15),
   2 fault_time bit (54),

2 eis_info (0:7) bit (36)) unaligned;

dcl (apx fixed bin init (0),
   abx fixed bin init (1),
   bpw fixed bin init (2),
   bpx fixed bin init (3),
   lpw fixed bin init (4),
   lpx fixed bin init (5),
   spx fixed bin init (6),
   abx fixed bin init (7)) internal static options (constant);
dcl scup ptra;
dcl 1 scu based (scup) aligned, /* SCU DATA */

/*
 WORD (0) */

(2 ppr,
  3 prt bit (4),
  3 str bit (15),
  3 p bit (1),
2 app,
  0 xsf bit (1),
  3 sdwn bit (1),
  3 sd_on bit (1),
  3 ptw bit (1),
  3 pt_on bit (1),
  3 pi_ap bit (1),
  3 daptw bit (1),
  3 sdump bit (1),
  3 sdwp bit (1),
  3 ptw bit (1),
  3 pt_on bit (1),
  3 fap bit (1),
  3 fabs bit (1),
2 fault_cntxt bit (3),
3 )

/*
 WORD (1) */

2 fd,
  3 lio bit (1),
  3 oeb bit (1),
  3 e_off bit (1),
  3 orb bit (1),
  3 r_off bit (1),
  3 w_off bit (1),
  3 no_fa bit (1),
  3 ocb bit (1),
  3 oc_off bit (1),
  3 boc bit (1),
  3 inret bit (1),
  3 crt bit (1),
  3 rair bit (1),
  3 am_er bit (1),
  3 orc bit (1),
  3 par bit (1),
  3 pari bit (1),
  3 one_l bit (1),
  3 one_z bit (1),
  3 fap bit (1),
  3 fabs bit (1),
/*
 FD */

2 fault cntxt bit (3),

/*
 number of retries of EIS instructions */

/*
 FAULT DATA */

/* illegal ring order */
/* out of execute bracket */
/* no execute */
/* out of read bracket */
/* no read */
/* out of write bracket */
/* no write */
/* not a gate */
/* out of call bracket */
/* outward call */
/* bad outward call */
/* inward return */
/* cross ring transfer */
/* ring alarm register */
/* associative memory fault */
/* out of segment bounds */
/* processor parity upper */
/* processor parity lower */
/* op not complete type 1 */
/* op not complete type 2 */
2 port_stat,
3 ial bit (4),
3 iac bit (3),
3 con_chan bit (3),
2 fl_num bit (5),
2 fl_flag bit (1),
/* PORT STATUS */
/* illegal action lines */
/* illegal action channel */
/* connect channel */
/* (fault/interrupt) number */
/* 1 => fault, 0 => interrupt */

2 tpsr,
3 trr bit (3),
3 tsr bit (15),
2 pad2 bit (9),
2 cpu_no bit (3),
2 delta bit (6),
/* TEMPORARY POINTER REGISTER */
/* temporary ring register */
/* temporary segment register */
/* CPU number */
/* tally modification DELTA */

2 word3 bit (18),
2 tsr_stat,
3 tana,
4 prn bit (3),
4 prv bit (1),
3 tsnb,
4 prn bit (3),
4 prv bit (1),
3 tsnc,
4 prn bit (3),
4 prv bit (1),
2 tpr_tbr bit (6),
/* TSR STATUS for 1, 2, 43 word instructions */
/* Word 1 status */
/* Word 1 PR number */
/* Word 1 PR valid bit */
/* Word 2 status */
/* Word 2 PR number */
/* Word 2 PR valid bit */
/* Word 3 status */
/* Word 3 PR number */
/* Word 3 PR valid bit */
/* TPR.TBR field */

2 lloc bit (18),
2 lr,
3 zero bit (1),
3 neg bit (1),
3 carry bit (1),
3 ovfl bit (1),
3 sovf bit (1),
3 eufi bit (1),
3 ofle bit (1),
3 tro bit (1),
3 par bit (1),
3 parb bit (1),
/* INSTRUCTION COUNTER */
/* INDICATOR REGISTERS */
/* zero indicator */
/* negative indicator */
/* carryry indicator */
/* overflow indicator */
/* eponent overflow */
/* exponent underflow */
/* overflow mask */
/* tally runout */
/* parity error */
/* parity mask */
/* WORD (5) */

2 ca bit (18),
2 cu,
3 rf bit (1),
3 rpt bit (1),
3 rd bit (1),
3 ri bit (1),
3 pot bit (1),
3 pon bit (1),
3 xde bit (1),
3 xdo bit (1),
3 poa bit (1),
3 rfi bit (1),
3 if bit (1),
2 cpu_tag bit (6) unaligned,

/* WORDS (6,7) */

2 even_inst bit (36),
2 odd_inst bit (36);

/* ALTERNATE SCU DECLARATION */

dcl 1 scux based (scup) aligned,
(2 pad0 bit (36),
2 fd,
3 lsn bit (1),
3 loc bit (1),
3 l_a_sm bit (1),
3 lap bit (1),
3 lpr bit (1),
3 nea bit (1),
3 oeb bit (1),
3 pad bit (29),

/* CONTROL UNIT STATUS */
/* on first cycle of repeat instr */
/* repeat instruction */
/* repeat double instruction */
/* repeat link instruction */
/* IT modification */
/* return type instruction */
/* XDE from Even location */
/* XDE from Odd location */
/* operation preparation */
/* tells CPU to refetch instruction */
/* ITS modification */
/* fault occurred during instruction fetch */

/* COMPUTED ADDRESS */

/* GROUP II FAULT DATA */
/* illegal segment number */
/* illegal op code */
/* illegal address - modifier */
/* illegal slave procedure */
/* illegal procedure */
/* non existent address */
/* out of bounds */
2 pad2 bit (36),
2 pad3a bit (18),
2 tsr_stat (0:2),
3 prn bit (3),
3 prv bit (1),
2 pad3b bit (6) unaligned,
2 pad45 (0:1) bit (36),
2 instr (0:1) bit (36);

/* TSR STATUS as an ARRAY */
/* FR number */
/* FR valid bit */

/* Instruction ARRAY */
/* PDS - The Process Data Segment */

Last modified (Date and reason):
2/6/76 by S. Webber Initial coding
9/17/76 by R. Brett 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 ipo 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 J. A. Bush for the DPS8/70M CPU value.
02/23/81 by J. Longioanni to remove temp_mode_reg (moved to prds$mode_reg_enabled)
03/81 by E. Donner to remove next_itl 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 obsolete 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_fa_object_ops to suppress auditing
while in the msg primitives and RCP.
*/

/* format: style3,14ind25 */
pds: procedure;

/* This program creates the pds data base */
/* Automatic */

*******pds.cds*************
dcl 1 cdss aligned like cdss_arg;
dcl code fixed bin (35);

/* Constants */
dcl pdname char (3) aligned static init ("pdns") 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 options (variable);
dcl get_temp_segment entry (ptr, fixed bin (35));
dcl release_temp_segment entry (char (*), ptr, fixed bin (35));
dcl hcs_schname_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) extend;
dcl 

1 pds aligned based (pds), /* MC for page faults and timer runouts */ 
2 page_fault_data like mc, /* MC for normal faults */ 
2 fim_data like mc, /* storage for MC being signalled */ 
2 signal_data aligned like mc, /* this must follow signal data */ 
2 history_reg_data (64) fixed bin (71), /* user id for current process */ 
2 process_group_id char (32), /* number that when subtracted from clock reading gives 
virtual cpu time */ 
2 cpu_time fixed bin (52), /* temporary used in calculating VCPU time */ 
2 virtual_delta fixed bin (52), /* temporary used in calculating VCPU time */ 
2 virtual_time_at_eligibility fixed bin (52), /* temporary used in calculating 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_np ptr, /* stack pointer at getw time */ 
2 apt_ptr ptr, /* pointer to this process's AST 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_arg_invalid bit (36) aligned, /* used by pxss masking/arg copying code */ 
2 process_id 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 pds_valid bit (0) unaligned, /* (added segdef for dseg) */ 
2 date 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_inhibit 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_vails fixed bin, /* page faults */ 
2 number_of_pages_in_use fixed bin, /* used in calculating memory units */ 
2 post_purred 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 (35), /* count of bounds faults taken by this process */ 
2 vtoo_reads fixed bin (35), /* vtoo read I/Os done for this process */ 
2 vtoo_writes fixed bin (35), /* vtoo write I/Os done for this process */ 
2 mo_trace_seg fixed bin, /* seg number of object segment being traced */ 
2 mo_trace_sw bit (2) aligned, /* switch for M. G. Tracing "11"b => trace on */ 
2 stack_0_sdwp ptr aligned, /* ptr to stack sdw in dseg */
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>stack_0_ptr</td>
<td>ptr aligned, ptr to base of ring 0 stack (wired for esd)</td>
</tr>
<tr>
<td>tc_argp</td>
<td>ptr, arg ptr used by tc</td>
</tr>
<tr>
<td>tc_mask</td>
<td>bit (72), save to mask</td>
</tr>
<tr>
<td>exp_undfl_rest</td>
<td>bit (2), fim restarts underflow: '1'b = binary, '01'b = hex</td>
</tr>
<tr>
<td>exp_ovfl_rest</td>
<td>bit (2), fim restarts exp overflow: '1'b = binary, '01'b = hex</td>
</tr>
<tr>
<td>sovfl_value</td>
<td>bit (72), fim restarts exp overflow: value DFLD'ed by fim on restart binary overflow</td>
</tr>
<tr>
<td>hex_sovfl_value</td>
<td>bit (72), fim on restart hex overflow</td>
</tr>
<tr>
<td>cpar_err_data</td>
<td>bit (72), cache parity error data (from cache)</td>
</tr>
<tr>
<td>cpar_mem_data</td>
<td>bit (72), cache parity error data (from memory)</td>
</tr>
<tr>
<td>cpar_info</td>
<td>bit (36), diagnose flag, cache level and absaddr</td>
</tr>
<tr>
<td>hfp_exponent_enabled</td>
<td>bit (1), user allowed to set IR hex exp bit</td>
</tr>
<tr>
<td>pre_empr_poll_return_pointer</td>
<td>bit (1), limit bandwidth of segment state covert channels</td>
</tr>
<tr>
<td>block_lock_count</td>
<td>fixed bin, count of locks held</td>
</tr>
<tr>
<td>throttle_segment_state_changes</td>
<td>bit (1), limit bandwidth of segment state covert channels</td>
</tr>
<tr>
<td>first_covert_event_time</td>
<td>fixed bin (52), per-rings start time for the process</td>
</tr>
<tr>
<td>pad_for_trace_mod16</td>
<td>fixed bin (52), packed ptr to mc_trace buffer</td>
</tr>
<tr>
<td>trace</td>
<td>(306), fixed bin (71), system trace data</td>
</tr>
<tr>
<td>timer_time_out</td>
<td>fixed bin (52), time out for the process</td>
</tr>
<tr>
<td>timer_channel</td>
<td>fixed bin (71), event channel for time out event</td>
</tr>
<tr>
<td>term_channel</td>
<td>fixed bin (71), channel used to signal process termination</td>
</tr>
<tr>
<td>term_proc</td>
<td>bit (36), process ID of process to signal target process</td>
</tr>
<tr>
<td>pli_machine</td>
<td>fixed bin, nonzero if we do pli-like things</td>
</tr>
<tr>
<td>validation_level</td>
<td>fixed bin (3), ACC string for condition name</td>
</tr>
<tr>
<td>condition_name</td>
<td>fixed bin (8), unaligned, IPS masks</td>
</tr>
<tr>
<td>len</td>
<td>char (32), unaligned, array of automatic masks for IPS signals</td>
</tr>
<tr>
<td>chars</td>
<td>bit (36), used in checking validation level changes</td>
</tr>
<tr>
<td>pad_obsolete</td>
<td>bit (36), UID used in some locking</td>
</tr>
<tr>
<td>ips_mask</td>
<td>bit (35), packed ptr to mc_trace wired buffer</td>
</tr>
<tr>
<td>auto_mask</td>
<td>bit (35), pathnam associative memory</td>
</tr>
<tr>
<td>ring_alarm_val</td>
<td>bit (36), first procedure executed in a new process</td>
</tr>
<tr>
<td>lock_id</td>
<td>bit (36), not used yet</td>
</tr>
<tr>
<td>mo_trace_buf</td>
<td>ptr unaligned, alternate form of process group id</td>
</tr>
<tr>
<td>pad_end_of_page_0</td>
<td>bit (0), pathnames associative memory</td>
</tr>
<tr>
<td>pathname_am</td>
<td>aligned like pam, first procedure executed in a new process</td>
</tr>
<tr>
<td>initial_procedure</td>
<td>ptr, not used yet</td>
</tr>
<tr>
<td>account_id</td>
<td>char (32), alternate form of process group id</td>
</tr>
<tr>
<td>access_name</td>
<td>char (32), top of present ITT list</td>
</tr>
<tr>
<td>user</td>
<td>char (32), pointers to per-ring working directories</td>
</tr>
<tr>
<td>project</td>
<td>char (32), UID of per-ring working directories</td>
</tr>
<tr>
<td>tag</td>
<td>char (32), UID of per-ring working directories</td>
</tr>
<tr>
<td>home_dir</td>
<td>char (168), transparent usage, mod, pd switch</td>
</tr>
<tr>
<td>process_dir_name</td>
<td>char (32), top of present ITT list</td>
</tr>
<tr>
<td>wdir</td>
<td>ptr, max authorization this user can attain</td>
</tr>
<tr>
<td>wdir_uid</td>
<td>bit (36), max authorization this user can attain</td>
</tr>
<tr>
<td>transparent</td>
<td>bit (36), per-ring mask for special channels</td>
</tr>
<tr>
<td>itt_head</td>
<td>bit (18), initial ring of execution for the process</td>
</tr>
<tr>
<td>max_access_authorization</td>
<td>aligned like aim_template, per-ring stack pointers</td>
</tr>
<tr>
<td>stacks</td>
<td>(0:7) ptr, pointer to start of KST</td>
</tr>
<tr>
<td>kstp</td>
<td>ptr, special wakeups pending</td>
</tr>
<tr>
<td>events_pending</td>
<td>bit (36), special channels assigned</td>
</tr>
<tr>
<td>special_channels</td>
<td>bit (36), per-ring mask for special channels</td>
</tr>
<tr>
<td>event_masks</td>
<td>(7) bit (36), initial ring of execution for the process</td>
</tr>
</tbody>
</table>
*/ lowest ring in which IPS interrupts are allowed */
*/ highest ring in which process can run */
*/ bit(1) is ON if ring (1) is prelinked */
*/ used to tag MC */
*/ size of lot in stack (0 -> lot not in stack) */
*/ size of CLR in stack */
*/ histograms of linkage faults */
*/ histogram of linkage fault times */
*/ histogram of linkage faults PF's */
*/ ptr to temp segment into which dirs are copied */
*/ pvtd of volume being dumped */
*/ OH until leave ring zero once */
*/ rel pointer to start of saved MC area */
*/ ptr to next MC save place */
*/ max address where MC can be saved */
*/ indicates whether lot can be referenced */
*/ per-ring indicator that ltt messages copied to ect */
*/ limit on force-writing */
*/ Following must be doubleword aligned */
*/ holds state of fast_ho_ipc at block */
*/ for initializer */
*/ There is a 1 here for each privilege that must be reset on exit from ring 1 */
*/ Ring 1 has asked to turn off ring 0 auditing */
*/ to mark end of PDS for MC save area */
*/ Returns ZEROS */

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.quota_inhib = 1;
*/ initializer ignore rqover until it is enabled */
pds.vtime_count = -1;

*/ * pds.cds
pds.process_id = (36)*1b;
pds.lock_id = (36)*1b;
pds.pl1_machine = 1;
pds.ip_mask (*) = (35)*1b;
pds.force_write_limit (*) = 1;

pds.save_history_regs = "1"b;
pds.hreg_saved = "0"b;
pds.history_reg_data (*) = 0;

pds.mo_trace_buf = null;
pds.mo_trace_sw = "0"b;
pds.mo_trace_reg = 0;

pds.eovfl_value = unspec (Default_exponent_control_overflow_value);
pds.hex_eovfl_value = unspec (Default_hex_exponent_control_overflow_value);

pds.exp_eovfl_rest, pds.exp_undfl_rest = "0"b;

pds.stack_0_swp = 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 = "e";

pds.home_dir = ">system_control_I";
pds.process_dir_name = ">process_dir_dir>!zzzzzzbBBBBBBB";

pds.wdir (*) = null;
pds.wdir_uid (*) = "0"b;

pds.stacks (*) = null;
pds.dmpr_pvid = "0"b;
pds.dmpr_ptrx = 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$filename_file (get_wdir_ (), "pds", ", "template_pds", code);
if code ~= 0
then if code ~= error_table$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, pdaname, "The variable ^a is "d words away from being aligned on a "d-word boundary.",
                                message, (modulo - remainder), modulo);
    end check;

Xpage; Xinclude aim_template;
Xpage; Xinclude cds_args;
Xpage; Xinclude exponent_control_info;
Xpage; Xinclude mc;
Xpage; Xinclude pathname_am;
Xpage; Xinclude 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
 6/13/77 by M. Weaver to null signal and set 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 1 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,
 2 interrupt_data aligned like mc,
 2 fin_data aligned like mc,
 2 syscall_data aligned like mc,
 2 ignore_data aligned like scr,
 2 ftemp fixed bin (71),
 2 last_recorded_time fixed bin (71),
 2 idle_ptr ptr,
 2 simulated_mask fixed bin (71),
 2 am_data bit (0),
 2 ptw_am_regs (4*16) fixed bin (35),
 2 ptw_am_ptr (4*16) fixed bin (35),
 2 adw_am_regs (4*16) fixed bin (71),
 2 adw_am_ptr (4*16) fixed bin (35),
 2 processor_pattern bit (8) aligned,
 2 processor_tag fixed bin (3),
 2 last_timer_setting bit (27) aligned,
 2 depth fixed bin,
 2 mode_reg bit (36) aligned,
 2 fault_reg bit (72) aligned,
 2 apt_ptr ptr,
 2 idle_temp fixed bin (71),

/* standard stack header */
/* MC for interrupts */
/* MC for connect faults, timer runouts */
/* MC for saved syscall data */
/* for SCU data to be ignored at certain times */
/* temporary used by il (surprise) */
/* used by traffic control */
/* pointer to idle process APT for this processor */
/* simulated system controller mask register */
/* to get addr of associative memory data block */
/* page table regs (4 sets of 16 for dps8/70m) */
/* page table pointers (4 sets of 16 for dps8/70m) */
/* segment desc. regs (4 sets of 16 for dps8/70m) */
/* segment desc. pointers (4 sets of 16 for dps8/70m) */
/* 1 bit ON for this processor */
/* CPU tag from maintenance panel */
/* last timer value loaded for this CPU */
/* depth in eligible queue for running process */
/* mode register for this processor */
/* cache mode register for this CPU */
/* place to store the fault register */
/* CPU tag from maintenance panel */
/* first stack frame location */

/* 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,
2 pad_mod_8 (2) fixed bin,
2 ignore_pl (8) bit (36) aligned,
2 pad_mod_16 (16) bit (36) aligned,
2 processor_stack aligned like stack_frame;
/* marker for fast_connect_init */
/* used to set mode register */
/* used by wired fin to spl/lpl */
/* 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 database */

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 = l;
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);

del where ptr;
del message char (*);
del modulo fixed bin;

    if mod (bin (rel (where), 18), modulo) ^= 0
    then call com_err_ (0, prdname, "The variable "^a is not aligned on a "^d-word boundary.", message, modulo);

end check;
% include cds_args;
X include stack_header;
X include stack_frame;
X include me;

end prds;
/* BEGIN INCLUDE FILE ... ptw.168.incl.pll ... 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),
  2 frame fixed bin (14) unsigned unaligned,
  2 padl bit (4) unaligned,
  2 flags unaligned like 168_ptw_flags;

/* In-core page descriptor */
/* Core frame number */
dcl 1 168_ptw aligned based (ptp),
  2 add bit (18) unaligned,
  2 flags like 168_ptw_flags unaligned;

/* General declaration for out-of-core PTW */
dcl 1 168_special_ptw aligned based (ptp) like 168_ptw;
dcl 1 168_real_disk_ptw aligned based (ptp) like 168_ptw;
dcl 1 168_null_disk_ptw aligned based (ptp) like 168_ptw;

dcl 1 168_ptw_flags unaligned based,
  (2 add_type bit (4),
  2 first bit (1),
  2 er bit (1),
  2 padl bit (1),
  2 unusablel bit (1),
  2 phu bit (1),
  2 phml bit (1),
  2 nypdl bit (1),
  2 phml bit (1),
  2 dfl bit (1),
  2 wired bit (1),
  2 os bit (1),
  2 valid bit (1),
  2 df_no bit (2)) unaligned;

/* Page is somewhere peculiar -- add_type = "Ol"b */
/* Page for page actually on disk -- add_type = "10"b */
/* Page for page not yet on disk -- add_type = "11"b */

/* Various software/hardware flags */
/* 0000=nu11, 1000=core, 0100=disk, 0010=pd, 0001=swap */
/* the page has not yet been written out */
/* error on last page 1/0 (also used by post-purge as temp) */
/* can't be used because hardware resets this bit */
/* page has been used bit */
/* Cumulative OR of hardware phm's */
/* must be moved to paging device */
/* page has been modified bit */
/* page has been used in the quantum */
/* page is to remain in core */
/* page is out-of-service (I/O in progress) */
/* directed fault if this is 0 (page not in core) */
/* directed fault number for page faults */

/* END INCLUDE FILE ... ptw.168.incl.pll */
/* 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 spterp bit(18) unaligned;

/* END INCLUDE FILE ... pv_holdt.incl.pl1 ... */
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.

/*
   BEGIN INCLUDE FILE ... pvt.incl.pl1 ... last modified January 1982 */

/*
   number of PVT entries */
2 n_entries fixed bin (17),
1 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, /* VTOC index for root (» */
2 disk_table_vtocx fixed bin, /* VTOC 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.pll */
/* 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_pvts
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 (va to 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 physical 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 ho_part_used
bit (1), /* HC part set up by init_pvt */

/* */
2 volmap_lock_notify bit (1) unaligned, /* TRUE if notify required when volmap lock is unlocked */
2 volmap_idle_notify bit (1) unaligned, /* TRUE if notify required when volmap state is idle */
2 vtoc_map_lock_notify bit (1) unaligned, /* TRUE if notify required when vtoc map lock is unlocked */

2 n_free_vtoc 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 dbmp (2) bit (18), /* rel ptr to number 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 curm_dmpr_vtocx fixed bin unaligned, /* current vtocx being dumped */
2 n_vtoc fixed bin unaligned, /* number of vtocx on this volume */
2 baseaddr fixed bin (18) unaligned, /* Base of paging region */
2 pad2 bit (18) unaligned, /* */
2 volmap_segsdw fixed bin (71), /* SDW describing volmap_seg */
2 volmap_astepptr unaligned, /* Packed pointer to ASTE for volmap_seg */
2 volmap_offset bit (18) unaligned, /* Offset in volmap_seg of volume map */
2 vtoc_offset bit (18) unaligned, /* 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 unaligned, /* Packed pointer to record stock */
2 vtoc_map_stock_ptr ptr unaligned, /* Packed pointer to VTOC 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 */

dol
(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.pll

* * * * * * * * * * * * * * * * * *
/* ***********************************************************
* *
* 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$end_clock_init
Modified 01/16/81 W. Olin Sibert, to add scs$port_data
Modified January 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 dcl_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 odsa aligned like cfa_args;
dcl 1 fixed bin;

/* Builtins */

dcl (addr, bin, bit, null, size, string, unspec) builtin;

/* Entries */

dcl create_data_segment_entry (ptr, fixed bin (35));
}xpage;
dcl 1 scs aligned,
    2 start_of_scs fixed bin (71),
    2 controller_data (0:7) aligned like scs$controller_data,
2 processor_data (0:7) aligned like scs$processor_data,
2 port_data (0:7) like scs$port_data aligned,
2 cow (0:7) like scs$cow,
2 cow_ptrs (0:7) aligned like scs$cow_ptrs,
2 reconfig_general_cow aligned like scs$reconfig_general_cow,
2 sys_level aligned bit (72),
2 open_level aligned bit (72),
2 processor_start_mask aligned bit (72),
2 cpu_test_mask aligned bit (72),
2 number_of_masks fixed bin,
2 processor_start_pattern bit (36) aligned,
2 cpu_test_pattern bit (36) aligned,
2 expanded_ports bit (1) unaligned dim (0:7),
2 cam_pair fixed bin (71),
2 cam_wait bit (8) aligned,
2 pad1 fixed bin,
2 set_mask (0:7) bit (36) aligned,
2 read_mask (0:7) bit (36) aligned,
2 mask_ptr (0:7) ptr unaligned,
2 idle_step (0:7) ptr unaligned,
2 connect_lock bit (36) aligned,
2 reconfig_lock bit (36) aligned,
2 trouble_flags bit (8) aligned,
2 bos_restart_flags bit (8) aligned,
2 nprocessors fixed bin,
2 bos_processor_tag fixed bin (3),
2 faults_initialised bit (3) aligned,
2 sys_trouble_pending bit (1) aligned,
2 fast_cam_pending (0:7) bit (36) aligned,
2 interrupt_controller fixed bin (3),
2 cycle_priority_template bit (7) aligned,
2 set_cycle_switches bit (1) aligned,
2 processor_start_int_no fixed bin (5),
2 processor bit (8) aligned,
2 processor_start_wait bit (8) aligned,
2 trouble_processid bit (36) aligned,
2 drl_message_pointer ptr unaligned,
2 processor_test_data aligned like scs$processor_test_data,
2 pad2 fixed bin,
2 trouble_drbs (0:7) fixed bin (71),
2 port_addressing_word (0:7) bit (3) aligned,
2 cfg_data (0:7) fixed bin (71),
2 cfg_data_save fixed bin (71),
2 processor_switch_data (0:4) bit (36) aligned,
2 processor_switch_template (0:4) bit (36) aligned,
2 processor_switch_compare (0:4) bit (36) aligned,
2 processor_switch_mask (0:4) bit (36) aligned,
2 processor_data_switch_value bit (36) aligned,
2 controller_config_size (0:7) fixed bin (14) aligned,/* Data used by init_set and collect_free_core, from config cards. */
2 reconfig_locker_id char (32) aligned,/* config card-stated size of controller */
2 scas_page_table (0:31) bit (36) aligned, /* process group ID of process doing reconfiguration */
2 end_of_scs fixed bin; /* Page table for SCAS */

Xpage:
unspec (scs) = "0"b1
/* For initialization */

/* 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"b1;
cdsa.switches.have_text = "1"b1;

call create_data_segment_ (addr (cdsa), code);
return;

Xpage;
Xinclude scs;
Xinclude cdsa_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),
    2 rings,
      3 r1 bit (3),
      3 r2 bit (3),
      3 r3 bit (3),
    2 valid bit (1),
    2 df_no bit (2),
    2 pad1 bit (1),
    2 bound bit (14),
    2 access,
      3 read bit (1),
      3 execute bit (1),
      3 write bit (1),
      3 privileged bit (1),
    2 unpaged bit (1),
    2 not_a_gate bit (1),
    2 cache bit (1),
    2 entry_bound bit (14)) unaligned;
   /* main memory address of page table */
   /* ring brackets for the segment */
   /* directed fault bit (0 -> fault) */
   /* directed fault number */
   /* boundary field (in 16 word blocks) */
   /* access bits */
   /* read permission bit */
   /* execute permission bit */
   /* write permission bit */
   /* privileged bit */
   /* segment is unpaged if this is 1 */
   /* if this is 0 the entry bound is checked by hardware */
   /* cache enable bit */
   /* entry bound */

END INCLUDE FILE ... sdw.168.incl.pl1 */
/* BEGIN INCLUDE FILE ... sst.incl.pl ... 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 segment info, Benson Margulies 84-01-03 */
/* Added covert channel meters, Keith Loepere 85-01-08. */

#include "incl.pl1"

January 1985

Note: The included file has an ALM counterpart made with cif.

Deleted paging device info and added pc segment info, Benson Margulies 84-01-03.
Added covert channel meters, Keith Loepere 85-01-08.

/* SST In:ADER *
2 pre_page_time fixed bin (71),
2 post_purge_time fixed bin (71),
2 post_in_core fixed bin,
2 thrashing fixed bin,
2 npfs_misses fixed bin,
2 salv fixed bin,
2 ptrl bit (36),
2 astl bit (36),
2 astl_event bit (36),
2 astl_notify_requested bit (1) aligned,
2 used fixed bin,
2 probase fixed bin (24),
2 tfree ptr,
2 sstp ptr,
2 pctl_wait_ct fixed bin,
2 astsize fixed bin,
2 cmaxsize fixed bin,
2 root_astep ptr,
2 pnts (0:3) fixed bin,
2 level (0:3),
3 (ausedp, no_aste) bit (18) unaligned,
2 (atempp, atempp1) bit (18) unal,
2 dm_enabled bit (1) aligned,
2 (mine, minip) bit (18) unal,
2 strsize fixed bin,
2 sstp ptr,
2 utf bit (18),
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 startp bit (18),
2 cmp ptr,
2 usedp bit (18),
2 wctr fixed bin,
2 remove 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 set 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 fo_skips_pinned fixed bin (35),
/* number of skips over pinned page in find_core */
2 ol_skips_pinned fixed bin (35),
/* number of skips over pinned page in claim_mod_core */
2 ast_ht_ptr ptr,
/* AST hash 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 pdir_page_faults fixed bin */
2 level1_page_faults fixed bin,
/* total page faults in sys libs */
2 dir_page_faults fixed bin,
/* Total page faults on directories */
2 ring0_page_faults fixed bin,
/* page faults in ring 0 */
2 rqover fixed bin (35),
/* ercode for record quota overflow */
2 pc_io_waits fixed bin,
/* Number of times pc had to wait for 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 needs 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 skipu fixed bin,
/* number of pages skipped because they were wired */
2 skipm fixed bin,
/* because of being modified */
2 skipos fixed bin,
/* because out of service */
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,  
2 writes (8) fixed bin,  
2 short_pf_count fixed bin,  
2 loop_locks fixed bin,  
2 loop_lock_time fixed bin (71),  
2 cpu_sf_time fixed bin (71),  
2 total_sf_pf fixed bin,  
2 total_sf fixed bin,  
2 pre_page_size fixed bin,  
2 post_list_size fixed bin,  
2 post_purgings fixed bin,  
2 post_purge_calls fixed bin,  
2 pre_page_calls fixed bin,  
2 pre_page_list_size fixed bin,  
2 pre_page_misses fixed bin,  
2 pre_pagings fixed bin,  

/* number of reads for each did */  
/* number of writes for each did */  
/* count of page faults on out of service pages */  
/* count of times locked PTL */  
/* time spent looping on PTL */  
/* cpu time spent in seg_fault */  
/* total page faults while in seg_fault */  
/* total number of seg_faults */  
/* total pre-pagings expected */  
/* total number of post-purgings */  
/* total number of calls to post-purge */  
/* total number of calls to pre-page */  
/* total number of misses in pre-page list */  
/* 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,  
2 system_type fixed bin,  
2 recopies fixed bin,  
2 first_core_block fixed bin,  
2 last_core_block fixed bin,  
2 fv_retires fixed bin (35),  
2 pvhtp ptr unaligned,  

/* count of abs-wired pages */  
/* ADP_SYSTEM or L68_SYSTEM */  
/* number of times a wired page was copied */  
/* number of times recopied because modified */  
/* core map index for first block of core */  
/* core map index for last block of core */  
/* force_write retries due to ASTE move */  
/* ptr to PV hold table for debugging */  

/* AST METERS */  
2 askipsiz (0: 3) fixed bin,  
2 aneedsize (0: 3) fixed bin,  
2 steps fixed bin,  
2 askipsz fixed bin,  
2 assearches fixed bin,  
2 askiplevel fixed bin,  
2 askipinit fixed bin,  
2 acost fixed bin,  
2 askiplock fixed bin,  
2 askipdius fixed bin,  
2 alaps fixed bin,  
2 updates fixed bin,  
2 setfaults_all fixed bin,  
2 setfaults_acc fixed bin,  
2 total_bf fixed bin,  
2 total_bf_pf fixed bin,  
2 cpu_bf_time fixed bin (71),  

/* array of skips because wrong AST size */  
/* array of times needed each size */  
/* count of steps taken looking for an AST entry */  
/* count of skips because EHS was ON */  
/* count of full searches made */  
/* count of skips because pages were in core */  
/* count of times turned OFF init switch */  
/* cumulative cost of deactivations */  
/* count of skips because couldn't lock parent */  
/* count of skips because DIUS was ON */  
/* lap counter for AST list */  
/* calls to update */  
/* setfaults done to the entire SDW */  
/* setfaults done to the access field */  
/* count of bound faults */  
/* page faults during bound faults */  
/* 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 that ast 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$cleanup */
2 cleanup_real_time fixed bin, /* total real time in pc$cleanup */
2 force_activations fixed bin, /* Data for pre-page decisions */
2 force_updatevs fixed bin, /* counters for pre-page decisions */
2 fw_none fixed bin, /* counters for measuring pre-page success */
2 force_updates fixed bin,
2 force_writes fixed bin, /* Relative cmem to next cmem for writing */
2 force_write bit (18) aligned, /* Times claim_mod_core invoked */
2 pf_unlock_ptrl_time fixed bin (71), /* Times unacceptable cmem found by o_m_c */
2 pf_unlock_ptrl_meterings fixed bin, /* Times free cmem passed by o_m_c */
2 makeknown_activations fixed bin (35), /* Times o_m_c passed pure page */
2 backup_activations fixed bin (35), /* Times used page seen */
2 dir_activations fixed bin, /* Times o_m_c saw unacceptable ptw */
2 claim_runs fixed bin, /* Writes queued by o_m_c */
2 activations fixed bin, /* Steps passed in core claiming */
2 activations fixed bin, /* counter of times quick find_core_filed */
2 activations fixed bin, /* null addresses reinstated */
2 activations fixed bin, /* Pseudo-page faults on volmap_seg */
2 bailout_resurrections fixed bin, /* out-of-physical-volume page faults */
2 bailout_resurrections fixed bin, /* addresses resurrected by double-writing */
2 volmap_seg_page_faults fixed bin (35), /* Time (VCPU) in seg mover */
2 ocpu fixed bin, /* Page faults in seg moving */
2 sgm_time fixed bin (71), /* Seg moves that failed */
2 sgm_time fixed bin, /* Seg faults in seg moves */
2 sgm_time fixed bin, /* Seg moves that completed */
2 sgm_time fixed bin, /* Times claim_mod_core had to run */
2 good_sgms fixed bin, /* total count of activations */
2 good_sgms fixed bin, /* count of directory activations */
2 good_sgms fixed bin, /* call-in updates */
2 sgm_time fixed bin, /* call in core flush writes */
2 claim_runs fixed bin, /* see evict_page.sln */
2 activations fixed bin, /* ptp of page being moved */
2 activations fixed bin, /* N/Z if page was mod */
2 activations fixed bin, /* Data for metering force_write facility 08/19/78 */
2 activations fixed bin, /* Calls on segments to force write */
2 activations fixed bin, /* Mod pages so written */
2 activations fixed bin, /* Force write wrote none */
2 activations fixed bin, /* Update's so forced */
2 activations fixed bin, /* Time unlocking ptrl page faults */
2 activations fixed bin, /* activations at makeknown time */
2 activations fixed bin, /* activations for backup */
2 activations fixed bin,
2 activations fixed bin,
2 metering_flags aligned, 3 activate activated bit (1) unaligned, 3 pad bit (35) unaligned, 2 seg_fault_calls fixed bin (35), /* small chunks of misc. information */ /* ON => last call to activate entry actually activated something */ /* number calls to seg_fault for explicit activation */

/* METERS FOR STACK TRUNCATION */

2 (stk_truncate_should_didnt, 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), /* counts */

/* 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_to_limit fixed bin, /* max read aheads */ 2 segmove_found_synch fixed bin (35), /* one 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_asetp_ptr unaligned, 2 segmove_vtx pos fixed bin, 2 segmove_vtoocx fixed bin, 2 segmove_old_addr_asetp ptr unaligned, 2 segmove_new_addr_asetp ptr unaligned, /* if non-null, addresses to be rescued from old_addr_asetp */ /* if segmove_asetp nonnull, valid */ /* ditto */ /* ditto */ /* if non-null, the addresses must be deposited. */

2 mod_during_write fixed bin, 2 zero_pages fixed bin, 2 trace_sw aligned, 3 pad_trace bit (32) unaligned, 3 pc_trace_pf bit (1) unaligned, 3 try_trace bit (1) unaligned, 3 pc_trace bit (1) unaligned, 3 sc_trace bit (1) unaligned, 2 new_pages fixed bin, 2 ast_track bit (1) aligned, 2 dislock_writebehind fixed bin, 2 write_limit fixed bin, /* times a page was modified while it was being written */ /* count of pages truncated because all zero */ /* tracing control flags */ /* flag used by page control primitives */ /* flag used by segment control primitives */ /* newly created pages */ /* "1"b => keep SST name table */ /* "1"b => keep SST name table */ /* Max # of outstanding writes by page control */ /* crash in mid-segmove */ /* count of times a process was delayed in affecting a seg stat */ /* count of times a process was audited for excessive seg state changes */ /* total times processes were delayed for covert channels */ /* number of events over which we determine covert channel bandwidth */ /* max bps for covert channel before we delay */ /* maximum bps for covert channel before we audit */ /* access_operation_value for excessive seg state_chg */ /* padding to 512 words (1000)8 */

/ * END INCLUDE FILE sst.incl.pli */
/* Begin include file astnt.incl.pl1 */

/* Created 10/03/74 by Bernard Greenberg */
/* modified 08/24/79 by J. A. Bush for easier calculation of size of astnt */
/* Modified 08/27/84 by Keith Loepere to purge BOS */

dcl    sst_names $ ext;

dcl    sstnp ptr;

dcl 1 sstnt based (sstnp) aligned,
    2 valid bit (1) aligned,
    2 multics_or_bce char (4) aligned,
    2 nentries fixed bin,
    2 pad (5) fixed bin,
    2 (ast_sizes,
    ast_name_offsets,
    ast_offsets,
    pad2) (0 : 2) 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 */

/* Segment containing ast name table */
/* Pointer to ast name segment */
/* Major structure */
/* 1 => structure filled by Multics */
/* Origin of data in table */
/* number of entries in the astnt */
/* Sizes of ASTE's at each level */
/* Starting index for names at each level */
/* Starting rel addr of each AST region */

/* End include file astnt.incl.pl1 */
begin include file ...

/* begin include file ... stack_0_data.incl.pl1 */

/* Created 790509 by Mike Grady */

dcl stack_0_data$ fixed bin ext;
dcl stack_0_data_init_number_of_stacks fixed bin;
dcl sdtp ptr;

dcl 1 sdt aligned based (sdtp),
    2 lock bit (36),
    2 num_stacks fixed bin,
    2 free bit (18),
    2 pad fixed bin,
    2 stacks (stack_0_data_init_number_of_stacks
       refer (sdt.num_stacks)) like sdt;

dcl sdtep ptr;

dcl 1 sdte aligned based (sdtep),
    2 nextp bit (18) unal,
    2 pad bit (18) unal,
    2 stack bit (18) unal,
    2 aptep bit (18) unal,
    2 sdtv bit (72);

/* end include file ... stack_0_data.incl.pl1 */
dcl RETURN_PTR_MASK bit (72) int static options (constant) /* mask to be AND'd with stack_frame.return_prt */
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,`  /* serves as both */
`2 prev_sp pointer,`  /* Translator ID */
`2 next_sp pointer,`  /* 0 => PL/I version II */
`2 return_ptr pointer,`  /* 1 => ALM */
`2 entry_ptr pointer,`  /* 2 => PL/I version I */
`2 operator_and ip_ptr ptr,`  /* 3 => signal caller frame */
`2 arg_ptr pointer,`  /* 1 => on_unit_relp1 bit(18) unaligned, */
`2 static_ptr ptr unaligned,`  /* 2 => on_unit_relp2 bit(18) unaligned, */
`2 support_ptr ptr unaligned,`  /* 3 => on if frame belongs to a main procedure */
`2 on_unit_relp3 bit(18) unaligned,`  /* 4 => signaller frame */
`2 operator_return_offset bit(18) unaligned,`  /* 2 operator return offset */
`2 x(0: 7) bit(18) unaligned,`  /* 2 index registers */
`2 a bit(36),`  /* 2 accumulator */
`2 q bit(36),`  /* 2 q-register */
`2 e bit(36),`  /* 2 exponent */
`2 timer bit(27) unaligned,`  /* 2 timer */
`2 pad bit(6) unaligned,`  /* 2 pad */
`2 ring_alarm_reg bit(3) unaligned;`  /* 2 ring alarm */

`dcl 1 stack_frame_flags based(sp) aligned,`  /* stack_frame.incl.pl1 */
`2 psa(0 : 7) bit(72),`  /* 2 skip over prs */
`2 xs0 bit(22) unaligned,`  /* 2 on if frame belongs to a main procedure */
`2 main_proc bit(1) unaligned;`  /* 2 main proc */
2 run_unit_manager bit(1) unal,
2 signal bit(1) unal,
2 crawl_out bit(1) unal,
2 signaler bit(1) unal,
2 link_trap bit(1) unal,
2 support bit(1) unal,
2 condition bit(1) unal,
2 xx08 bit(6) unal,
2 xx1 fixed bin,
2 xx1 fixed bin,
2 xx1 bit(25) unal,
2 old_crawl_out bit (1) unal,
2 old_signaler bit(1) unal,
2 xx1a bit(9) unaligned,
2 xx1(9) bit(72) aligned,
2 v2_pll_op_ret_base ptr,
2 xx3 bit(72) aligned,
2 pll_ps_ptr ptr;

/* on if frame belongs to run unit manager */
/* on if frame belongs to logical signal */
/* on if this is a signal caller frame */
/* on if next frame is signaler's */
/* on if this frame was made by the linker */
/* on if frame belongs to a support proc */
/* on if condition established in this frame */

/* on this is a signal caller frame */
/* on if next frame is signaler's */

/* 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) */
/* ptr to ps for this frame; also used by fio. */
BEGIN INCLUDE FILE ... stack_header.incl.pl1 . . 3/72 Bill Silver */
modified 7/76 by H. Weaver for "system links and more system use of areas */
modified 3/77 by H. Weaver to add rnt_ptr */
Modified April 1983 by G. Hornig for tasking. (the trace stuff is temporary - MBW) */

/* format: style2 */
do 1 stack_header based (sb) aligned,
  2 pad1 (4) fixed bin,
  2 old_lot_ptr ptr,
  2 combined_stat_ptr ptr,
  2 cllr_ptr ptr,
  2 max_lot_size fixed bin (17) unal,
  2 main_proc_invoked fixed bin (11) unal,
  2 have_static_vlas bit (1) unal,
  2 pad4 bit (2) unal,
  2 run_unit_depth fixed bin (2) unal,
  2 our_lot_size fixed bin (17) unal,
  2 pad2 bit (18) unal,
  2 system_free_ptr ptr,
  2 user_free_ptr ptr,
  2 null_ptr ptr,
  2 stack_begin_ptr ptr,
  2 stack_end_ptr ptr,
  2 lot_ptr ptr,
  2 signal_ptr ptr,
  2 bar_mode_sp ptr,
  2 pill_operators_ptr ptr,
  2 call_op_ptr ptr,
  2 push_op_ptr ptr,
  2 return_op_ptr ptr,
  2 return_no_pop_op_ptr ptr,
  2 entry_op_ptr ptr,
  2 trans_op_tw_ptr ptr,
  2 isot_ptr ptr,
  2 act_ptr ptr,
  2 unminder_ptr ptr,
  2 sys_link_info_ptr ptr,
  2 rnt_ptr ptr,
  2 scd_ptr ptr,
  2 assign_linkage_ptr ptr,
  2 task_data_ptr ptr,
  2 trace,
  3 frames,
  4 count fixed bin,
  4 top_ptr ptr unal,
  3 in_trace bit (36) aligned,
  2 pad3 (3) bit (36) aligned;
/* the main pointer to the stack header */
/* (0) also used as arg list by outward_call_handler */
/* (4) pointer to the lot for current ring (obsolete) */
/* (6) pointer to area containing separate static */
/* (8) pointer to area containing linkage sections */
/* (10) DU number of words allowed in lot */
/* (10) DL nonzero if main procedure invoked in run unit */
/* (10) DL "lb if (very) large arrays are being used in static */
/* (10) DL number of active run units stacked */
/* (11) number of words (entries) in lot */
/* (11) reserved */
/* (12) pointer to system storage area */
/* (14) pointer to user storage area */
/* (16) */
/* (18) pointer to first stack frame on the stack */
/* (20) pointer to next useable stack frame */
/* (22) pointer to the lot for the current ring */
/* (24) pointer to signal procedure for current ring */
/* (25) value of sp before entering bar mode */
/* (28) pointer to pill_operators$operator_table */
/* (30) pointer to standard call operator */
/* (32) pointer to standard push operator */
/* (34) pointer to standard return operator */
/* (36) pointer to standard return / no pop operator */
/* (38) pointer to standard entry operator */
/* (40) pointer to translator operator ptrs */
/* (42) pointer to ISOT */
/* (44) pointer to System Condition Table */
/* (46) pointer to unminder for current ring */
/* (48) pointer to *system link name table */
/* (50) pointer to Reference Name Table */
/* (52) pointer to event channel table */
/* (54) pointer to storage for (obsolete) hcs_assign_linkage */
/* (56) for possible tasking (experimental) */
/* expected to be temporary */
/* number of trace frames */
/* pointer to last trace frame */
/* trace antirecursion flag */
/* for future expansion */
The following offset refers to a table within the pll 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.pll */
/* BEGIN INCLUDE FILE ... str.incl.pli ... 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 ass trailer rel pointer */
     2 bp bit (18),
     /* backward ass trailer rel pointer */
     2 segno bit (18),
     /* segment number */
     2 dstep bit (18)) unaligned;
    /* rel pointer to ring 0 date */
dcl stra (0:8000) bit (72) based (strp) aligned;
    /* END INCLUDE FILE ... str.incl.pli */
dcl tcmp ptr;

dcl 1 tcm aligned based (tcmp),
 2 tc_suspend_lock like lock,
 2 cld2 fixed bin (18),
 2 cld3 fixed bin (18),
 2 cld4 fixed bin (18),
 2 depth_count fixed bin (18),
 2 loadings fixed bin (18),
 2 blocks fixed bin (18),
 2 wakups fixed bin (18),
 2 waits fixed bin (18),
 2 notifies fixed bin (18),
 2 scheduling fixed bin (18),
 2 interactions fixed bin (18),
 2 squeue fixed bin (35, 18),
 2 tewait fixed bin (18),
 2 te_block fixed bin (18),
 2 te_i_stop fixed bin (18),
 2 te_preempt fixed bin (18),
 2 p_interactions fixed bin,
 2 idle fixed bin (71),
 2 mp_idle fixed bin (71),
 2 nmp_idle fixed bin (71),
 2 zero_idle fixed bin (71),
 2 last_time fixed bin (71),
 2 loop_locks fixed bin (18),
 2 loop_lock_time fixed bin (18),
 2 squeue fixed bin (35, 18),
 2 sort_to_elhead fixed bin (18),
 2 processor_time fixed bin (71),
 2 response_time fixed bin (71),
 2 eligible_time fixed bin (71),
 2 response_count fixed bin,
 2 eligible_count fixed bin,
 2 quit_counts (0:5) fixed bin,
 2 loading_idle fixed bin (71),
 2 delta_vcpu fixed bin (71),
 2 post_purge_switch fixed bin,
/* when locked, te is suspended */
/* number of calls to block */
/* number of calls to wakeup */
/* number of calls to notify */
/* number of calls to block */
/* number of calls updated from pre_empt */
/* number of interactive schedulings */
/* recent time average of number in queue */
/* times te called from wait */
/* times te updated from block */
/* times te updated from i_stop */
/* times te updated from pre_empt */
/* times interaction bit turned off because of high priority */
/* multi-programming idle */
/* non-multi-programming idle time */
/* zero idle time */
/* times looped on the APT lock */
/* time looping on the APT lock */
/* average length of eligible queue */
/* 0 => no one, 1 => int've only, 2 => everybody */
/* estimate of response time */
/* estimate of eligible time */
/* count of response meters */
/* count of eligible meters */
/* array of buckets indexed by state */
/* loading_idle time */
/* delta virtual CPU time for the system */
/* ON if post purging is to be done */

*/

* * *
2 time_out_severity fixed bin,
2 notify_check fixed bin,
2 quit_priority fixed bin,
2 ibm_polling_time fixed bin (71),
2 end_of_time fixed bin (71),
2 gp_at_notify fixed bin (18),
2 gp_at_pnotify fixed bin (18),
2 int_q_enabled fixed bin (18),
2 fnp_buffer_threshold fixed bin (18),

/* 100 octal */
2 depths (8) fixed bin (18),
2 tdepths (8) fixed bin (71),
2 pfdepth (8) fixed bin (18),
2 pti_not_waits fixed bin (18),
2 gv_gp_window_count fixed bin (18),
2 metering_lock fixed bin (18),
2 gp_start_count fixed bin (18),
2 gp_done_count fixed bin (18),
2 nto_check_time fixed bin (71),
2 nto_delta fixed bin (35),
2 nto_count fixed bin (18),
2 tcpu_scheduling fixed bin (18),
2 nto_event bit (36),
2 page_notifies fixed bin (18),
2 notify_nobody_count fixed bin (18),
2 notify_nobody_event bit (36),
2 system_type fixed bin,
2 stat (0:15) fixed bin (18),

/* 200 octal */
2 wait (8),
3 time fixed bin (18),
3 count fixed bin (18),
2 ready (8),
3 time fixed bin (18),
3 count fixed bin (18),
2 total_pf_time fixed bin (71),
2 total_pf_count fixed bin (18),
2 auto_tune_ws fixed bin (18),
2 occrs delta fixed bin (18),
2 ws_sum fixed bin (18),
2 nonidle_force_count fixed bin (18),
2 itt_list_lock bit (36) aligned,
2 cpu_pf_time fixed bin (71),
2 cpu_pf_count fixed bin (18),
2 special_offsets unaligned,

/* syserr first arg for notify time outs */
/* obsolete */
/* factor for scheduler quit response */
/* time to poll lobm */
/* very large time */
/* 0 => just do get_idle_processor */
/* 0 => just do get_idle_processor */
/* 0 => no intv q in percent mode */
/* if fewer free buffs then stingy alloc strategy */
/* set this to >= half n_ttylines/fnp for safety */

/* histogram of run depths */
/* histogram of times run per depth */
/* histogram of page faults per depth */
/* times pti_wait noticed pti was unlocked */
/* times window noticed */
/* 0=locked, else unlocked */
/* num calls to pti_wait */
/* to detect gv_gp window lossage */
/* next time at which nto code will be called */
/* microsec between nto checks */
/* number of times nto detected */
/* obsolete */
/* last event which NTO'd */
/* used to be tcm.inter */
/* num apes in each state */

/* histogram of page fault waiting times versus did */
/* histogram of times in ready queue */
/* total time spent from start to end of all page faults */
/* total number of page faults metered */
/* 0=0 don't, otherwise compensate for quantum len */
/* number of pages reserved for int users */
/* total of eligible's ws sizes */
/* count of eligibilities forced */
/* Lock on ITT free list */
/* total cpu time spent handling page faults */
/* total count of cpu time meterings */
3 apt_offset bit (18),
3 pad bit (18),
2 getwork_offset fixed bin (71),
2 getwork_count fixed bin (18),
2 short_p_count fixed bin (18),
2 interrupt_count fixed bin (71),
2 interrupt_count fixed bin (71),
2 ocore fixed bin (35, 18),
2 preem_flag bit (36) aligned,
2 cumulative_memory_usage fixed binary (71),
2 processor_time_at_define_wc fixed bin (71),
2 boost_priority fixed bin,
2 lost_priority fixed bin,
2 total_clock_lag fixed bin (71),
2 clock_simulations fixed bin,
2 max_clock_lag fixed bin,

/* total time spent in getwork */
/* total times through getwork */
/* number of short page faults */
/* total time spent in interrupt */
/* total number of metered interrupts */
/* fraction of core for int've users */
/* controls whether preemting at done time */
/* total number of memory usage units */
/* value of processor_time when WC's last defined */
/* number of times priority process given high priority */
/* number of times priority process lost eligibility */
/* sum of all simulated clock delays */
/* number of times alarm clock interrupt was simulated */
/* largest simulated alarm clock delay */

/* 300 total */
2 pds_count fixed bin (18),
2 max_hproc_segno fixed bin,
2 pds_length fixed bin (18),
2 lock fixed bin (18),
2 id bit (36) aligned,
2 system_shutdown fixed bin (18),
2 working_set_factor fixed bin (35, 18),
2 ncpu fixed bin (18),
2 last_eligible bit (18),
2 apc_lock fixed bin (35),
2 apc_size fixed bin (18),
2 realtime_q aligned like based_sentinel,
2 shb_size fixed bin (18),
2 iuc_size fixed bin (18),
2 dat_size fixed bin (18),
2 iuc_free_list bit (18),
2 used_iuc fixed bin (18),
2 initialiser_id bit (36) aligned,
2 n_eligible fixed bin (18),
2 max_eligible fixed bin (30),
2 wait_enable fixed bin (18),
2 apc_entry_size fixed bin (18),
2 interactive_q aligned like based_sentinel,
2 dat_ptr ptr,
2 old_user_ptr,
2 initialise_time fixed bin (71),
2 init_event fixed bin (18),
2 oldt fixed bin (18),
2 newt fixed bin (18),
2 lastfirst fixed bin (30),
2 taiast fixed bin (30),
2 timax fixed bin (35),

/* amount of pds to copy for new process */
/* largest allowed hardcore segment number */
/* length of PDS */
/* length of PDS */
/* process id generator lock */
/* next uid to be added to uid_array */
/* working set factor */
/* number of processors currently being used */
/* last process to gain eligibility */
/* + write; 0 hidden -1 unlocked; +(N+1) N readers */
/* number of APT entries */
/* processes with realtime deadlines */
/* APT hash table size */
/* number of ITT entries */
/* number of allowed DST entries */
/* pointer to ITT free list */
/* number of used ITT entries */
/* process id of initialiser */
/* number of processes eligible */
/* maximum allowed number of eligible processes */
/* turned on when waiting mechanism works */
/* size of an APT entry */
/* head of interactive queue */
/* pointer to device signal table */
/* last process to run (apt ptr) */
/* time of initialization */
/* wait event during initialization */
/* timer reading from previous process */
/* timer setting for new process */
/* first eligible time */
/* last eligible time */
/* time in queue for lowest level */

*/
empty_q bit (18),
working_set_addend fixed bin (18),
ready_q_head bit (0) aligned,
eligible_q_head aligned like based_sentinel,
ready_q_tail bit (0) aligned,
eligible_q_tail aligned like based_sentinel,
idle_tail aligned like based_sentinel,
min_eligible fixed bin (30),
alarm_timer_list bit (18) aligned,
guaranteed_elig_too fixed bin (35),
priority_sched_inc fixed bin (35),
next_alarm_time fixed bin (71),
priority_sched_time fixed bin (71),
next_process_time fixed bin (71),
tape_polling_time fixed bin (71),
tape_polling_lock fixed bin (18),
max_channels fixed bin (18),

1 400 octal */

system_virtual_time fixed bin (71),
credit_bank fixed bin (71),
min_wct_index bit (18) aligned,
max_wct_index fixed bin (71),
gross_idle_time fixed bin (71),
credits_per_scatter fixed bin (35),
best_credit_value fixed bin (18),
define_wct_time fixed bin (71),
max_batch_elig fixed bin (35),
num_batch_elig fixed bin (35),
deadline_mode fixed bin (35),
credits_scattered fixed bin (35),
max_max_eligible fixed bin (30),
max_stopped_stack_0 fixed bin (35),
stopped_stack_0 fixed bin (35),
sys_polling_interval fixed bin (35),
max_polling_time fixed bin (71),
vcpu_response_bounds (VCPU_RESPONSE_BOUNDS) fixed bin (35),
vcpu_response_bounds_size fixed bin (35),
meter_response_time_calls fixed bin (35),
meter_response_time_value fixed bin (35),
meter_response_time_overhead fixed bin (71),
init_wait_time fixed bin (71),
init_wait_timeout fixed bin (71),
init_timeout_severity fixed bin,
init_timeout_recursive fixed bin,
max_timer_register fixed bin (71),
preempt_sample_time fixed bin (35),
governing_credit_bank fixed bin (35),
process_initial_quantum fixed bin (35),
default_procs_required bit (8) aligned,
work_class_idle fixed bin (71),

/* thread of empty APT entries */
/* additive working set parameter */
/* for added segdef */
/* head of eligible queue */
/* for added segdef */
/* tail of eligible queue */
/* tail of idle list */

/* next pointer to apt entry for next alarm */
/* amount of guaranteed eligibility time in microsecs */
/* amount of block time before process is given priority */
/* clock time for next alarm timer */
/* time for priority process to be given priority */
/* time to poll console DIM */
/* time to poll disk DIM */
/* time to poll tape DIM */
/* time to poll imp */
/* do not poll if lock set */
/* num special channels per process */

/* 0-> ti sorts, else deadline sorts */
/* Maximum of max */
/* Maximum stack_0's suspended by stopped procs */
/* Number stack_0's suspended by stopped procs */
/* for heals */
/* for heals */

/* used by wait/notify during initialization */
/* notify-timeout during initialization */
/* notify-timeout severity during initialization */
/* count of NTO recursion during initialization */
/* max cpu burst = $ opus x preempt_sample_time */
/* running parameter - max time between samples */
/* used for limiting eligibility on governed work classes */
/* eligibility quantum first eligibility */
/* default mask of CPUs required */
/* 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_truncate_avg_f1 fixed bin (35, 18),
2 stk_truncate_avg_f2 fixed bin (35, 18),
2 lock_error_severity fixed bin,

2 gv_integration fixed bin (35),
2 gv_integration_set bit (1) aligned,
2 pauses fixed bin (35),
2 volmap_polling_time fixed bin (71),
2 next_ring0_timer fixed bin (71),
2 realtime_io_priority_switch fixed bin,
2 realtime_io_deadline fixed bin (35),
2 realtime_io_quantum fixed bin (35),
2 realtime_priorities fixed bin (35),
2 relinquishes fixed bin (35),
2 abort_ips_mask bit (36) aligned,

500 octal */

2 uid_array (0:15) bit (36) aligned,
2 pad5 (176) fixed bin (35),

1000 octal */

2 pad7 (64) fixed bin (35),

1100 octal */

2 pad8 (8) fixed bin (35),
2 work_class_table aligned,
3 wcet (0:16) aligned like wct_entry,

3000 octal */

2 spt fixed bin;

dcl wctep ptr;

2 wcet_entry aligned based (wctep),
2 thread unaligned,
3 fp bit (18),
2 bp bit (18),
2 flags unaligned,
3 mbm bit (1),
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),
2 minf fixed bin (35),
2 pin_weight fixed bin (35),
2 eligibilities fixed bin (35),
CPU used by members

number times process entered "think" state

number times process entered "queued" state

The next three arrays correspond to the array vcpu_response_bounds

number times "processing" state

maximum fraction of cpu time

for limiting cpu resources

format of pxss-style sentinel

END INCLUDE FILE
/* BEGIN INCLUDE FILE ... vol_map.incl.pl1 */

dcl vol_map ptr;

dcl 1 vol_map based (vol_map) 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: vtoo_buffer.incl.pl1 November 1982 */

dcl vtoo_buffer_seg$ ext;

dcl vtoo_buffer_segp ptr;

dcl vtoo_buffer_desc ptr;

dcl vtoo_buffer_desc_arrayp ptr;

dcl vtoo_buffer_arrayp ptr;

dcl vtoo_buffers fixed bin;

dcl vtoo_buffer_n_buckets fixed bin;


dcl 1 vtoo_buffer aligned based (vtoo_buffer_segp),

2 lock,
3 processid bit (36) aligned,
3 wait_event bit (36) aligned,
3 notify_sw bit (1) aligned,
2 n_bufs fixed bin,
2 n_hash_buckets fixed bin,
2 hash_mask bit (36) aligned,
2 abs_addr fixed bin (24),
2 wait_event_constant fixed bin (36) uns aligned,
2 buf_desc_offset bit (18),
2 buf_offset bit (18),
2 hash_table_offset bit (18),
2 search_index fixed bin,
2 unsafe_pvtax fixed bin,
2 scavenger_free_p_clock fixed bin (35),

2 meters;
3 call_get fixed bin (35),
3 call_set fixed bin (35),
3 call_put fixed bin (35),
3 call_alloc fixed bin (35),
3 call_free fixed bin (35),
3 call_wait fixed bin (35),
3 steps fixed bin (35),
3 skip_os fixed bin (35),
3 skip_hot fixed bin (35),
3 skip_wait fixed bin (35),
3 disk_reads fixed bin (35),
3 disk_writes fixed bin (35),
3 get_buffer_calls fixed bin (35),
3 get_buffer_bits fixed bin (35),
3 wait_calls fixed bin (35),
3 wait_os fixed bin (35),
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_buffers)) aligned like vtoo_buf_desc,
2 buffer (vtoc_buf_n_buffers refer (vtoc_buffer.n_buffers)) aligned like vtoo_buffer;

dol 1 vtoo_buff_desc_array (vtoc_buffer.n_buffers) aligned based (vtoc_buffer_desc_arrayp) like vtoo_buffer_desc;

dol 1 vtoo_buff_desc aligned based (vtoc_buffer_descp),
  2 pvtx fixed bin (17) unaligned, /* PVTE index */
  2 vtoo fixed bin (17) unaligned, /* VTOCE Index */
  2 parts_used bit (3) unaligned, /* Mask of parts used or os */
  2 err bit (1) unaligned, /* ON -> I/O error on buffer */
  2 notify_sw bit (1) unaligned, /* ON -> notify required on I/O completion */
  2 write_sw bit (1) unaligned, /* ON -> write I/O */
  2 os bit (1) unaligned, /* ON -> I/O in progress */
  2 loq bit (1) unaligned, /* ON -> I/O has been requested */
  2 used bit (1) unaligned, /* ON -> this descriptor is in use */
  2 pad bit (9) unaligned,
  2 wait_index fixed bin (17) unaligned, /* Buffer index for forming wait event */
  2 ht_thread bit (18) unaligned, /* Offset of next entry in hash table */
  2 buf_rel bit (18) unaligned, /* Offset of buffer in segment */

dol 1 vtoo_buffer_array (vtoc_buffer.n_buffers) aligned based (vtoc_buffer_arrayp) like vtoo_buffer;

dol 1 vtoo_buffer aligned based (vtoc_bufferp),
  2 parts (3) aligned,
  3 words (64) bit (36) aligned;

dol N_PARTS_PER_VTOCE fixed bin int static options (constant) init (3);

dol VTOCE_PART_SIZE fixed bin int static options (constant) init (64);

dol VTOCE_BUFFER_SIZE fixed bin int static options (constant) init (3 * 64);

dol N_VTOCE_PER_RECORD fixed bin int static options (constant) init (5);

dol N_SECTOR_PER_VTOCE fixed bin int static options (constant) init (3);

/* END OF: vtoo_buffer.incl.pll */
/* BEGIN INCLUDE FILE ... vtoc_header.incl.pl1 */

dcl vtoc_header ptr;

dcl 1 vtoc_header based (vtoc_header) aligned,
   2 version fixed bin (17), /* version number. The current version number is 1. */
   2 n_vtoc 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_vtoc fixed bin (17), /* number of free vtoc entries */
   2 first_free_vtocx fixed bin (17), /* index of the first vtoc in the free list */
   2 pad (3) bit (36), /* space for dmpr bit map */
   2 dmpr_bit_map (2048 - 8) bit (36);

/* END INCLUDE ... vtoc_header */
BEGIN INCLUDE FILE ...vtoce.incl.pll ... last modified September 1982 */
/* Template for a VTOC entry. Length = 192 words. (3 * 64). */
/* NOTE: vtoce_man clears pad fields before writing a vtoce. */

dcl vtoce_ptr;

dcl 1 vtoce based (vtocep) aligned,

(2 pad_free_vtoce_chain bit (36),
 2 uid bit (36), /* Used to be pointer to next free VTOCE */
 2 msi bit (9), /* segment's uid - zero if vtoce is free */
 2 csi bit (9), /* maximum segment length in 1024 word units */
 2 records bit (9), /* current segment length - in 1024 word units */
 2 pad2 bit (9), /* number of records used by the seg in second storage */
 2 dmu bit (36), /* date and time segment was last used */
 2 dtm bit (36), /* date and time segment was last modified */
 2 ngsi bit (1), /* no quota switch - no checking for pages of this seg */
 2 decidious bit (1), /* true if ho_edw */
 2 null bit (1), /* no incremental dump switch */
 2 dmarp bit (1), /* Don't null zero pages */
 2 gtpd bit (1), /* Global transparent paging device */
 2 par_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 synchronised bit (1), /* TRUE if this is a data management synchronised segment */
 2 pad3 bit (8), /* directory switch */
 2 dirsw bit (1), /* master directory - a root for the logical volume */
 2 master_dir bit (1), /* not used */
 2 pad4 bit (16) uns aligned,

2 fm_checksum bit (36) aligned, /* Checksum of used portion of file map */

(2 quota (0:1) fixed bin (18) unsigned,
 2 used (0:1) fixed bin (18) unsigned, /* sec storage quota - (0) for non dir pages */
 2 received (0:1) fixed bin (18) unsigned, /* sec storage used - (0) for non dir pages */
 2 trp (0:1) fixed bin (71), /* total amount of storage this dir has received */
 2 trp_time (0:1) bit (36), /* time record product - (0) for non dir pages */
 2 trp_time_record_product was last calculated */