88.12

8-20

NewTypes

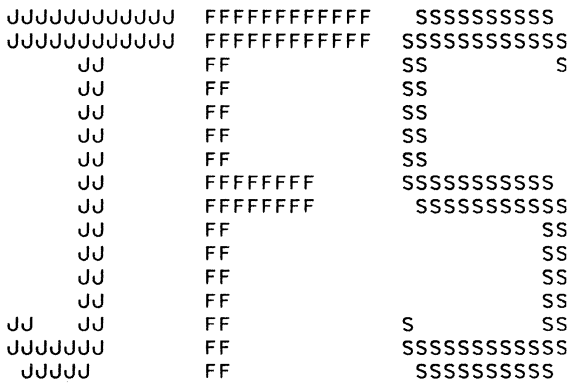
CDC NOS/VE Listing
August 1988

via:

CONTROL DATA

```
OPERATING SYSTEM =  NOS 739HX/22R2/1C.   89/08/09. PRINTED = 89/08/21. 19.48.08.

UJN          =  JFS         FAMILY    = AQUA          JOB ORIGIN    = BATCH.
CREATING JSN =  AFMR        USER NAME = JFS           SERVICE CLASS = BATCH.
```



Common Deck Listing

Common Deck Listing

```
o      4 MODULE type_declarations;


o      6 {------------------------------------  DFT$SERVER_DESCRIPTOR            ----------------------------
o      7
o      8 { DECK: DFT$SERVER_DESCRIPTOR
o      9 { This describes the data structure that is used on the client mainframe to
o     10 { describe the server file.  This is used as the served file variant portion of
o     11 { the MEDIA field in a File_Descriptor_Entry (FDE).
o     12
o     13   TYPE
o     14     dft$server_descriptor_p = ^dft$server_descriptor,
o     15
o     16     dft$server_descriptor = record
o     17       header: dft$server_descriptor_header,
o     18     recend,
o     19
o     20     dft$server_descriptor_header = record
o     21       server_mainframe_id: pmt$binary_mainframe_id,
o     22       served_family_table_index: dft$served_family_table_index,
o     23       server_lifetime: dft$lifetime,
o     24       read_write_count: 0 .. 0ffff(16),
o     25       purged: boolean,
o     26       highest_offset_allocated: amt$file_byte_address,
o     27       bytes_per_allocation: 0 .. dmc$max_bytes_per_allocation,
o     28       case file_state: dft$server_state of
o     29       = dfc$active =
o     30         { The file state is never really changed to dfc$inactive,
o     31         { or dfc$deactivated despite the fact the queues are in that state.
o     32         total_allocated_length: amt$file_byte_address,
o     33         remote_sfid: gft$system_file_identifier,
o     34         allow_other_mainframe_writer: boolean,
o     35         allocation_info: dft$server_allocation_info,
o     36         requested_transfer_size: dmt$transfer_size,
o     37       = dfc$terminated, dfc$awaiting_recovery =
o     38         { The information needs to be re-obtained from the server mainframe.
o     39         { The file state is never changed to dfc$recovering, or
o     40         { dfc$deleted
o     41       casend,
o     42     recend;
o     43
o     45 {*copyc amt$file_byte_address
o     46 {*copyc amt$file_limit
o     47 {*copyc amt$preset_value
o     48 {*copyc dft$lifetime
o     49 {*copyc dft$served_family_table_index
o     50 {*copyc dft$server_allocation_info
o     51 {*copyc dft$server_state
o     52 {*copyc dmt$allocation_size
o     53 {*copyc dmt$minimum_allocation_unit
o     54 {*copyc dmt$transfer_size
o     55 {*copyc gft$system_file_identifier
o     56 {*copyc ost$clear_file_space
o     57 {*copyc pmt$binary_mainframe_id
o     59
```

```
o     61 {----------------------------------  DMT$DISK_FILE_DESCRIPTOR  ----------------------------
o     62 {
o     63 { dmt$disk_file_descriptor
o     64 {
o     65
o     66
o     67   TYPE
o     68     dmt$disk_file_descriptor = record
o     69       read_write_count: 0 .. 0ffff(16),
o     70       delete_count: dmt$delete_count,
o     71       purged: boolean,
o     72       restricted_attach: boolean,
o     73       bytes_per_allocation: 0 .. dmc$max_bytes_per_allocation,
o     74       file_allocation_table: ^dmt$level_1_table,
o     75       fat_upper_bound: dmt$level_1_index,
o     76       current_fmd_index: dmt$fmd_index,
o     77       highest_offset_allocated: amt$file_byte_address,
o     78       bytes_per_level_2: amt$file_byte_address,
o     79       dfd_modified: boolean,
o     80       overflow_allowed: boolean,
o     81       requested_allocation_size: dmt$allocation_size,
o     82       requested_class: dmt$class_member,
o     83       requested_class_ordinal: dmt$class_ordinal,
o     84       requested_transfer_size: dmt$transfer_size,
o     85       requested_volume: dmt$requested_volume,
o     86       number_of_fmds: dmt$fmd_index,
o     87       p_fmd: ^dmt$file_medium_descriptor,
o     88       file_damaged: boolean,
o     89       damaged_detection_enabled: boolean,
o     90     recend;
o     91
o     92   TYPE
o     93
o     94     dmt$delete_count = 0 .. 0ffffff(16);
o     95
o     96   TYPE
o     97     dmt$file_hash_thread = ^gft$file_descriptor_entry,
o     98
o     99     dmt$active_file_hash_threads = array [0 .. dmc$max_file_hash] of dmt$file_hash_thread,
o    100
o    101     dmt$active_fde_lock = array [0 .. dmc$max_file_hash] of ost$signature_lock;
o    102
o    104 {*copyc amt$file_byte_address
o    105 {*copyc amt$file_limit
o    106 {*copyc amt$preset_value
o    107 {*copyc dmt$file_attributes
o    108 {*copyc dmt$global_file_name
o    109 {*copyc dmt$locked_file
o    110 {*copyc dmt$queue_status
o    111 {*copyc dmt$sparse_allocation
o    112 {*copyc dmt$system_file_id
o    113 {*copyc dmt$usage_count
o    114 {*copyc gft$file_descriptor_entry
```



```
o    115 {*copyc jmt$ijl_ordinal
o    116 {*copyc mmt$eoi_state
o    117 {*copyc osd$virtual_address
o    118 {*copyc ost$clear_file_space
o    119 {*copyc ost$hardware_subranges
o    120 {*copyc ost$signature_lock


o    123 {----------------------------------  DMT$FILE_ALLOCATION_STATUS  ----------------------------
o    124
o    125   TYPE
o    126     dmt$file_allocation_status = (
o    127         dmc$fas_account_limit_exceeded,
o    128         dmc$fas_file_allocated,
o    129         dmc$fas_job_mode_work_required,
o    130         dmc$fas_temp_reject);


o    132 {----------------------------------  DMT$FILE_ALLOCATION_UNIT  ----------------------------
o    133 {
o    134 { dmt$file_allocation_unit
o    135 {
o    136
o    137   TYPE
o    138     dmt$file_allocation_units = record
o    139       faus: array [ * ] of dmt$file_allocation_unit,
o    140     recend,
o    141     dmt$file_allocation_unit = record
o    142       dau_address: dmt$dau_address,
o    143       state: dmt$fau_states,
o    144       fmd_index: dmt$fmd_index,
o    145     recend;
o    146
o    147   TYPE
o    148     dmt$default_number_fau_entries = 1 .. dmc$default_number_fau_entries,
o    149     dmt$fau_entries = 0 .. dmc$max_fau_entries,
o    150     dmt$fau_states = (dmc$fau_free, dmc$fau_invalid_data,
o    151       dmc$fau_invalid_and_flawed, dmc$fau_initialized,
o    152       dmc$fau_initialized_and_flawed, dmc$fau_initialization_in_prog);
o    153
o    154   CONST
o    155     dmc$default_number_fau_entries = 39,
o    156     dmc$max_fau_entries = 134880;
o    157
o    158 {*copyc dmt$device_allocation_unit
o    159 {*copyc dmt$fmd_index


o    161 {----------------------------------  DMT$FILE_MEDIUM_DESCRIPTOR  ----------------------------
o    162
o    163 {
o    164 { dmt$file_medium_descriptor
o    165 {
```

```
o   166
o   167   TYPE
o   168     dmt$file_medium_descriptor = record
o   169       in_use: boolean,
o   170       system_file_id: dmt$system_file_id,
o   171       avt_index: dmt$active_volume_table_index,
o   172       dfl_index: dmt$device_file_list_index,
o   173       delete_logging_count: dmt$delete_logging_count,
o   174       volume_assigned: boolean,
o   175       fmd_allocated_length: amt$file_byte_address,
o   176       bytes_per_mau: dmt$bytes_per_mau,
o   177       daus_per_cylinder: dmt$daus_per_position,
o   178       daus_per_allocation_unit: dmt$daus_per_allocation,
o   179       internal_vsn: dmt$internal_vsn,
o   180       maus_per_dau: dmt$maus_per_dau,
o   181       maus_per_transfer_unit: dmt$maus_per_transfer,
o   182       p_next_fmd: ^dmt$file_medium_descriptor,
o   183       allocation_style: dmt$allocation_styles,
o   184     recend;
o   185
o   186   TYPE
o   187     dmt$delete_logging_count = 0 .. 0ffff(16);
o   188
o   189   TYPE
o   190     dmt$fmd_attributes = record
o   191       fmd_index: dmt$fmd_index,
o   192       attributes: array [1 .. * ] of dmt$fmd_attribute,
o   193     recend;
o   194
o   195   TYPE
o   196     dmt$fmd_attribute = record
o   197       case keyword: dmt$file_attribute_keywords of
o   198       = dmc$allocated_length =
o   199         fmd_allocated_length: amt$file_byte_address,
o   200       = dmc$device_file_list_index =
o   201         device_file_list_index: dmt$device_file_list_index,
o   202       = dmc$internal_vsn =
o   203         internal_vsn: dmt$internal_vsn,
o   204       = dmc$recorded_vsn =
o   205         recorded_vsn: rmt$recorded_vsn,
o   206       casend,
o   207     recend;
o   208
o   209   CONST
o   210     dmc$max_fmd_attribute = 7;
o   211
o   212 {*copyc amt$file_byte_address
o   213 {*copyc dmt$allocation_size
o   214 {*copyc dmt$active_volume_table_index
o   215 {*copyc dmt$device_allocation_unit
o   216 {*copyc dmt$device_file_list_index
o   217 {*copyc dmt$fad_index
o   218 {*copyc dmt$file_attributes
o   219 {*copyc dmt$global_file_name
o   220 {*copyc dmt$internal_vsn
o   221 {*copyc dmt$minimum_allocation_unit
```



```
o   222 {*copyc dmt$subfile_index
o   223 {*copyc dmt$system_file_id
o   224 {*copyc rmd$volume_declarations


o   226 {------------------------------------  DMT$FMD_INDEX                  ----------------------------
o   227 {
o   228 { dmt$fmd_index
o   229 {
o   230
o   231   TYPE
o   232     dmt$fmd_index = 0 .. 255;


o   234 {------------------------------------  GFC$CONSTANTS                  ----------------------------
o   235
o   236 { Define constants used for calculating addresses of FDE entries.
o   237 {   The FDE array is in mainframe wired or job fixed at a large address defined
o   238 {   by GFC$FDE_TABLE_BASE. Each entry is GFC$FDE_SIZE bytes long. NOTE that the actual
o   239 {   CYBIL type definition must be exactly this size. There is a check in
o   240 {   SYM$DEADSTART_INITIALIZATION to verify this size.
o   241
o   242   CONST
o   243     gfc$fde_table_base = 7f00000(16), {133169152 (Big and 0 mod 16384)}
o   244     gfc$fde_control_table_base = gfc$fde_table_base - 16384,
o   245     gfc$fde_size = 104;   {Must be 0 mod 8 and >= than actual FDE size
o   246
o   247


o   249 {------------------------------------  GFC$MONITOR_INTERLOCKS         ----------------------------
o   250
o   251 { This constant controls whether the GFP$ rotines actually set/clear the
o   252 { monitor interlock field in the FDE. Until 4 CPU design is complete, interlocking
o   253 { of FDEs is not required because all serailization is done at the entry to monitor.
o   254
o   255   CONST
o   256     gfc$monitor_interlocks = FALSE;


o   258 {------------------------------------  GFT$ALLOCATION_UNIT_SIZE       ----------------------------
o   259
o   260 { Define the maximum allowed size (in bytes) for an allocation unit.
o   261
o   262   TYPE
o   263     gft$allocation_unit_size = 0 .. 1000000;


o   265 {------------------------------------  GFT$ATTACH_COUNT               ----------------------------
o   266
o   267 { Define maximum number of jobs that can have a file attached simultaneously.
```

```
o   268
o   269   TYPE
o   270     gft$attach_count = 0 .. 65535;
o   271


o   273 {----------------------------------  GFT$FILE_DESCRIPTOR_CONTROL  ----------------------------
o   274
o   275 [ The following definition is used to manage the file descriptor entry array.
o   276
o   277   TYPE
o   278     gft$file_descriptor_control = RECORD
o   279       lock: ost$signature_lock,
o   280       index1: bool64,
o   281       index2: array [0 .. gfc$max_level_1_index] of bool64,
o   282       CASE 0 .. 2 OF
o   283       = 0 =
o   284         in_use: array [0 .. gfc$max_level_2_index] of bool64,
o   285       = 1 =
o   286         in_use_bits: packed array [0 .. gfc$max_level_2_bit_index] of boolean,
o   287       = 2 =
o   288         in_use_words: array [0 .. gfc$max_level_2_index] of integer,
o   289       CASEND,
o   290     RECEND,
o   291
o   292     bool64 = packed array [0 .. 63] of boolean;
o   293
o   294   CONST
o   295     gfc$max_level_1_index = 15,
o   296     gfc$max_level_2_index = 1023,
o   297     gfc$max_level_2_bit_index = 65535;
o   298
o   299
o   300
o   301 {*copyc ost$signature_lock


o   303 {----------------------------------  GFT$FILE_DESCRIPTOR_ENTRY  ----------------------------
o   304
o   305 { File Decription Table Entry (FDE)
o   306 { NOTE global_file_name may be moved to another table. It will contain an array
o   307 {   that define gfn/sfid pairs.
o   308
o   309   TYPE
o   310     gft$file_descriptor_entry = RECORD
o   311       job_lock: gft$signature_lock,
o   312       monitor_lock: mtt$monitor_interlock,
o   313       flags: gft$fde_flags,
o   314       global_file_name: ost$binary_unique_name,          {?????}
o   315       file_hash_thread: ^gft$file_descriptor_entry,
o   316       attached_in_write_count: gft$attach_count,
o   317       attach_count: gft$attach_count,
o   318       open_count: gft$open_count,
o   319       file_kind: gft$file_kind,
```



```
o   320       file_hash: 0 .. 255,
o   321       segment_lock: gft$segment_lock_info,
o   322       asti: mmt$ast_index,
o   323       eoi_byte_address: amt$file_byte_address,
o   324       eoi_state: mmt$eoi_state,
o   325       allocation_unit_size: gft$allocation_unit_size,
o   326       transfer_unit_size: gft$transfer_unit_size,
o   327       file_limit: amt$file_limit,
o   328       queue_status: gft$queue_status,
o   329       preset_value: pmt$initialization_value,
o   330       time_last_modified: ost$free_running_clock,
o   331       last_segment_number: ost$segment,          { Stack or last reference.
o   332       global_task_id: ost$global_task_id,        { Stack or last GTID that assigned ASID.
o   333       stack_for_ring: 0 .. 15,                   {    used mainly for stack mgmt
o   334
o   335       CASE media: gft$file_media OF
o   336       = gfc$fm_mass_storage_file =
o   337         disk_file_descriptor_p: ost$valid_relative_pointer,
o   338
o   339       = gfc$fm_served_file =
o   340         served_file_descriptor_p: ost$valid_relative_pointer,
o   341
o   342       CASEND,
o   343     RECEND,
o   344
o   345     gft$fde_flags = PACKED RECORD
o   346       eoi_modified: boolean,
o   347       wire_eoi_page: boolean,
o   348       active_file: boolean,
o   349       global_template_file: boolean,
o   350       fde_spare_4: boolean,
o   351       fde_spare_5: boolean,
o   352       fde_spare_6: boolean,
o   353       fde_spare_7: boolean,
o   354     RECEND;
o   355
o   356 {*copyc amt$file_byte_address
o   357 {*copyc amt$file_limit
o   358 {*copyc gft$allocation_unit_size
o   359 {*copyc gft$attach_count
o   360 {*copyc gft$transfer_unit_size
o   361 {*copyc gft$file_descriptor_index
o   362 {*copyc gft$file_kind
o   363 {*copyc gft$file_media
o   364 {*copyc gft$signature_lock
o   365 {*copyc gft$open_count
o   366 {*copyc gft$queue_status
o   367 {*copyc gft$segment_lock_info
o   368 {*copyc gft$transfer_unit_size
o   369 {*copyc mmt$ast_index
o   370 {*copyc mmt$eoi_state
o   371 {*copyc mtt$monitor_interlock
o   372 {*copyc osd$virtual_address
o   373 {*copyc ost$binary_unique_name
o   374 {*copyc ost$free_running_clock
o   375 {*copyc ost$global_task_id
```

```
o   376 {*copyc ost$signature_lock
o   377 {*copyc pmt$initialization_value


o   379 {------------------------------------  GFT$FILE_DESCRIPTOR_INDEX          ----------------------------
o   380
o   381   TYPE
o   382     gft$file_descriptor_index = 0 .. 65535;


o   384 {------------------------------------  GFT$FILE_DESC_ENTRY_P              ----------------------------
o   385
o   386   TYPE
o   387     gft$file_desc_entry_p = ^gft$file_descriptor_entry;
o   388
o   389 {*copyc gft$file_descriptor_entry


o   391 {------------------------------------  GFT$FILE_KIND                      ----------------------------
o   392
o   393 {
o   394 { !!!! WARNING - The order of ordinals in this type CANNOT be changed for file
o   395 {                 compatibility reasons. Although there is plenty of room to
o   396 {                 grow, the size of this type is also restricted to 1 byte for
o   397 {                 file compatibility reasons.
o   398 {
o   399
o   400   TYPE
o   401     gft$file_kind = (
o   402       gfc$fk_job_permanent_file,    {Permanent files.
o   403       gfc$fk_device_file,           {Device Files
o   404       gfc$fk_save_2,                {Reserved for future use.
o   405       gfc$fk_save_3,                {Reserved for future use.
o   406       gfc$fk_catalog,               {Permanent file catalogs.
o   407             { - - - dividing line between perm and temp files - - - }
o   408       gfc$fk_job_local_file,        {BAM named job local files.
o   409       gfc$fk_unnamed_file,          {File tables exist in Job Fixed.
o   410       gfc$fk_global_unnamed,        {File tables exist in mainframe wired.
o   411       gfc$fk_monitor_only_unnamed); {Files that are accessible in monitor ONLY.
o   412
o   413   TYPE
o   414     gft$file_kind_set = set of gft$file_kind;
o   415
o   416   CONST
o   417     gfc$fk_first_temporary_file = gfc$fk_job_local_file,
o   418     gfc$fk_last_permanent_file = gfc$fk_catalog;


o   420 {------------------------------------  GFT$FILE_MEDIA                     ----------------------------
o   421
o   422   TYPE
o   423     gft$file_media = (gfc$fm_transient_segment, gfc$fm_mass_storage_file,
```



```
o   424         gfc$fm_served_file);
o   425
o   426


o   428 {------------------------------------  GFT$LOCKED_FILE_DESC_ENTRY_P       ----------------------------
o   429
o   430   TYPE
o   431     gft$locked_file_desc_entry_p = ^gft$file_descriptor_entry;
o   432
o   433 {*copyc gft$file_descriptor_entry


o   435 {------------------------------------  GFT$PAGE_STATUS                    ----------------------------
o   436
o   437 { Define page status values returned from GFP$FETCH_PAGE_STATUS.
o   438
o   439
o   440 TYPE
o   441   gft$page_status = (
o   442       gfc$ps_page_doesnt_exist,
o   443       gfc$ps_page_on_disk,
o   444       gfc$ps_page_on_server,
o   445       gfc$ps_job_mode_work_required,
o   446       gfc$ps_temp_reject,
o   447       gfc$ps_volume_unavailable,
o   448       gfc$ps_account_limit_exceeded,
o   449       gfc$ps_server_terminated,
o   450       gfc$ps_server_allocate_required);
o   451
o   452
o   453


o   455 {------------------------------------  GFT$QUEUE_STATUS                   ----------------------------
o   456
o   457 { This parameter specifies where to keep pages of permanent files.
o   458 {       gfc$qs_global_shared -  pages ALWAYS in global queues
o   459 {       gfc$qs_job_working_set - pages ALWAYS in job working set
o   460 {       gfc$qs_job_shared - pages in shared queue only if file is attached by multiple
o   461 {                           jobs. File must be attach only for READ access.
o   462
o   463
o   464   TYPE
o   465     gft$queue_status = (gfc$qs_global_shared, gfc$qs_job_shared, gfc$qs_job_working_set);
o   466


o   468 {------------------------------------  GFT$SEGMENT_LOCK_INFO              ----------------------------
o   469
o   470   TYPE
o   471     gft$segment_lock_info = RECORD
```

GFT$SEGMENT_LOCK_INFO

```
o   472        locked_for_read: gft$open_count,
o   473        locked_for_write: boolean,
o   474        task_queue: tmt$task_queue_link,
o   475      RECEND;
o   476
o   477 {*copyc gft$open_count
o   478 {*copyc tmt$task_queue_link


o   480 {------------------------------------  GFT$SYSTEM_FILE_IDENTIFIER  ----------------------------
o   481
o   482    TYPE
o   483     gft$system_file_identifier = RECORD
o   484       file_entry_index: gft$file_descriptor_index,
o   485       residence: gft$table_residence,
o   486       file_hash: 0 .. 255,
o   487     recend;
o   488
o   489   CONST
o   490     gfc$null_file_hash = 255;
o   491
o   492 {*copyc gft$file_descriptor_index
o   493 {*copyc gft$table_residence


o   495 {------------------------------------  GFT$TABLE_RESIDENCE  ----------------------------
o   496
o   497    TYPE
o   498      gft$table_residence = (gfc$tr_null_residence, gfc$tr_system, gfc$tr_job, gfc$tr_system_wait_recovery);
o   499


o   501 {------------------------------------  JMC$SPECIAL_DISPATCH_PRIORITIES  ----------------------------
o   502 {
o   503 { Define priorities for special system tasks and tasks in special states.
o   504 {
o   505
o   506   CONST
o   507     jmc$priority_system_job          = jmc$priority_p10,
o   508     jmc$priority_job_scheduler       = jmc$priority_p11,
o   509     jmc$priority_split_alloc         = jmc$priority_p12,
o   510     jmc$priority_administer_log      = jmc$priority_p11,
o   511     jmc$priority_volume_space_mgr    = jmc$priority_p10,
o   512     jmc$priority_mli_helper          = jmc$priority_p13,
o   513
o   514     jmc$prior_system_tbls_locked     = jmc$priority_p10,
o   515     jmc$prior_subsystem_tbls_locked  = jmc$priority_p9;
o   516
o   517 {*copyc jmt$dispatching_priority


o   519 {------------------------------------  JMT$ACTIVE_JOB_LIST  ----------------------------
```


JMT$ACTIVE_JOB_LIST

```
o   520 {AJL - This deck defines the AJL. An entry in the AJL exists for each job
o   521 {that is in memory, in the process of being swapped in, or is swapped out
o   522 {but is being accessed temporarily by the system.  The AJL
o   523 {is also used to control the mapping of job fixed segments in the
o   524 {address space of monitor.
o   525
o   526 {
o   527 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o   528 {   make the appropriate changes in the corresponding display procedures in the
o   529 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o   530 {
o   531   TYPE
o   532     jmt$active_job_list_entry = RECORD
o   533       in_use: ALIGNED [0 MOD 8] 0 .. 0ffffff(16),
o   534       ijl_ordinal: jmt$ijl_ordinal,
o   535       ijle_p: ^jmt$initiated_job_list_entry,
o   536       job_is_good_swap_candidate: boolean,
o   537       time_freed: ost$free_running_clock,
o   538     RECEND,
o   539
o   540     jmt$active_job_list = ARRAY [0 .. *] of jmt$active_job_list_entry;
o   541
o   542 {*copyc jmt$ijl_ordinal
o   543 {*copyc jmt$initiated_job_list_entry
o   544 {*copyc ost$hardware_subranges


o   546 {------------------------------------  JMT$DELAYED_SWAPIN_WORK  ----------------------------
o   547 {Define list of special work that must be done to a job environment when
o   548 {the job is next swapped in.
o   549 {
o   550 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o   551 {   make the appropriate changes in the corresponding display procedures in the
o   552 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o   553 {
o   554
o   555   TYPE
o   556     jmt$delayed_swapin_work = SET OF (jmc$dsw_job_recovery, jmc$dsw_update_debug_lists,
o   557         jmc$dsw_update_keypoint_masks, jmc$dsw_job_asid_changed, jmc$dsw_job_shared_asid_changed,
o   558         jmc$dsw_update_job_task_enviro, jmc$dsw_recovery_swap_io_error, jmc$dsw_update_server_files,
o   559         jmc$dsw_adjust_cpu_selections, jmc$dsw_io_error_while_swapped, jmc$dsw_unused_10, jmc$dsw_unused_11,
o   560         jmc$dsw_unused_12, jmc$dsw_unused_13, jmc$dsw_unused_14, jmc$dsw_unused_15);
o   561
o   562    TYPE
o   563     jmt$delayed_swapin_work_record = record
o   564       delayed_swapin_work: jmt$delayed_swapin_work,
o   565       { The inhibit_access_work and terminate_access_work are only used when
o   566       { update server files is include in the set.
o   567       inhibit_access_work: dft$mainframe_set,
o   568       terminate_access_work: dft$mainframe_set,
o   569     recend;
o   570
o   571 {*copyc dft$mainframe_set
```

```
o   573 {------------------------------------  JMT$IJL_ORDINAL                 ----------------------------
o   574 {  Define IJL ordinal. An IJL ordinal is packed record that consists of
o   575 {     two parts:
o   576 {     block_number - an index into an array of pointers to small arrays
o   577 {                    of IJL entries
o   578 {     block_index  - an index into the small array of IJL entries
o   579 {
o   580 {  For optimum efficiency, the number of bits in an IJL ordinal should be
o   581 {  a multiple of 8 bits and each component of the ordinal should be
o   582 {  0 .. 2**n-1.
o   583
o   584   TYPE
o   585     jmt$ijl_ordinal = packed RECORD
o   586       block_number: jmt$ijl_block_number,
o   587       block_index: jmt$ijl_block_index,
o   588     RECEND,
o   589
o   590     jmt$ijl_block_number = 0 .. 2047,
o   591     jmt$ijl_block_index = 0 .. 31;
o   592
o   593   CONST
o   594     jmc$max_ijl_entries = 2048 * 32,
o   595     jmc$max_ijl_index_count = 32;


o   597 {------------------------------------  JMT$IJL_STATISTICS              ----------------------------
o   598 {Define statistics record that is kept in the IJL.
o   599 {
o   600 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o   601 {   make the appropriate changes in the corresponding display procedures in the
o   602 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o   603 {
o   604
o   605   TYPE
o   606     jmt$ijl_statistics = record
o   607       cp_time: ost$cp_time,
o   608       paging_statistics: ost$paging_statistics,
o   609       perm_file_space: sft$counter,
o   610       temp_file_space: sft$counter,
o   611       ready_task_count: 0 .. 0ffff(16),
o   612       tasks_not_in_long_wait: 0 .. 0ffff(16),
o   613     recend;
o   614
o   615 {*copyc OST$CP_TIME
o   616 {*copyc OST$PAGING_STATISTICS
o   617 {*copyc sft$counter


o   619 {------------------------------------  JMT$IJL_SWAP_STATUS             ----------------------------
o   620 {Define swap status field used in IJL (initiated job list entry).
o   621
o   622   TYPE
o   623     jmt$ijl_swap_status = (jmc$iss_null,
```



```
o   624           jmc$iss_executing,
o   625           jmc$iss_idle_tasks_initiated,
o   626           jmc$iss_job_idle_tasks_complete,
o   627           jmc$iss_swapped_no_io,
o   628           jmc$iss_flush_am_pages,
o   629           jmc$iss_job_allocate_swap_file,
o   630           jmc$iss_wait_allocate_swap_file,
o   631           jmc$iss_allocate_swap_file,
o   632           jmc$iss_wait_job_io_complete,
o   633           jmc$iss_job_io_complete,
o   634           jmc$iss_wait_allocate_sfd,
o   635           jmc$iss_allocate_sfd,
o   636           jmc$iss_swapped_io_cannot_init,
o   637           jmc$iss_initiate_swapout_io,
o   638           jmc$iss_wait_swapout_io_init,
o   639           jmc$iss_swapout_io_initiated,
o   640           jmc$iss_swapout_io_complete,
o   641           jmc$iss_swapped_io_complete,
o   642           jmc$iss_free_swapped_memory,
o   643           {Note: jmc$iss_swapout_complete is used by syp$get_job_swap_status
o   644           {to determine if JWS pages were recovered (or not) by DM file recovery
o   645           jmc$iss_swapout_complete,
o   646           jmc$iss_swapin_requested,
o   647           jmc$iss_swapin_resource_claimed,
o   648           jmc$iss_wait_swapin_io_init,
o   649           jmc$iss_swapin_io_initiated,
o   650           jmc$iss_swapin_io_complete),
o   651
o   652       jmt$swapout = jmc$iss_idle_tasks_initiated .. jmc$iss_swapout_complete,
o   653       jmt$swapin = jmc$iss_swapin_requested .. jmc$iss_swapin_io_complete;
o   654
o   655 { The following constants are used to inhibit access to jobs that are in
o   656 { the process of being swapped.  Memory manager io is inhibited if swap status
o   657 { is greater than jmc$inhibit_memory_manager_io (MMP$GET_INHIBIT_IO_STATUS).
o   658 { XCB access is inhibited if swap status is greater than jmc$inhibit_xcb_access
o   659 { (TMP$GET_XCB_ACESS_STATUS).
o   660
o   661   CONST
o   662     jmc$inhibit_memory_manager_io = jmc$iss_swapped_no_io,
o   663     jmc$inhibit_xcb_access = jmc$iss_swapped_io_cannot_init;
o   664


o   666 {------------------------------------  JMT$INITIATED_JOB_LIST_ENTRY    ----------------------------
o   667 { IJL - (Initiated Job List) An entry exists in this table for each job that
o   668 {       has been initiated and has not terminated.
o   669 {
o   670 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o   671 {   make the appropriate changes in the corresponding display procedures in the
o   672 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o   673 {
o   674
o   675
o   676
o   677   TYPE
```

JMT$INITIATED_JOB_LIST_ENTRY

```
o   678      jmt$initiated_job_list_entry = record
o   679        system_supplied_name: ALIGNED [0 MOD 8] jmt$system_supplied_name,
o   680        job_name: string (8),
o   681        entry_status: jmt$ijl_entry_status,
o   682        ajl_ordinal: jmt$ajl_ordinal,
o   683        kjl_ordinal: jmt$kjl_index,
o   684        swap_status: jmt$ijl_swap_status,
o   685        next_swap_status: jmt$ijl_swap_status,
o   686        last_swap_status: jmt$ijl_swap_status,
o   687        inhibit_swap_count: 0 .. 0ffffff(16),
o   688        active_io_page_count: 0 .. 0ffffff(16),
o   689        active_io_requests: 0 .. 0ffff(16),
o   690        swap_queue_link: jst$ijl_swap_queue_link,
o   691        job_fixed_asid: ost$asid,
o   692        long_wait_aging_complete: boolean,
o   693        notify_swapper_when_io_complete: boolean,
o   694        scheduling_dispatching_priority: jmt$dispatching_priority,
o   695        dispatching_control: jmt$ijl_dispatching_control,
o   696        job_monitor_taskid: ost$global_task_id,
o   697        job_mode: jmt$job_mode,
o   698        executing_task_count: 0 .. 0ff(16),
o   699        multiprocessing_allowed: boolean,
o   700        memory_reserve_request: mmt$memory_reserve_request,
o   701        swapin_candidate_queue: jmt$ijl_ordinal,
o   702        estimated_ready_time: ost$free_running_clock,
o   703        last_think_time: ost$free_running_clock,
o   704        age_purge_timestamp: ost$free_running_clock,
o   705        sfd_purge_timestamp: ost$free_running_clock,
o   706        job_scheduler_data: jmt$scheduling_data,
o   707        job_page_queue_list: mmt$job_page_queue_list,
o   708        swap_data: jmt$swap_data,
o   709        swap_io_control: jst$io_control_information,
o   710        sfd_p: ^jst$swap_file_descriptor,
o   711        system_breakpoint_selected: boolean,
o   712        delayed_swapin_work: jmt$delayed_swapin_work,
o   713        inhibit_access_work: dft$mainframe_set,
o   714        terminate_access_work: dft$mainframe_set,
o   715        statistics: jmt$ijl_statistics,
o   716        service_class_statistics: jmt$ijl_service_class_stats,
o   717        job_fixed_contiguous_pages: 0 .. 0ff(16),
o   718        hung_task_in_job: boolean,
o   719        job_damaged_during_recovery: boolean,
o   720        maxws_aio_slowdown_display: 0 .. 0ff(16),
o   721        unable_to_swap_idle_flag: boolean,
o   722        queue_file_information: jmt$queue_file_ijl_information,
o   723        relative_priority_enabled: boolean,
o   724        task_created_after_last_swap: boolean,
o   725        interactive_task_gtid: ost$global_task_id,
o   726      RECEND,
o   727
o   728      jmt$scheduling_data = RECORD
o   729        ready_task_link: jmt$ijl_ordinal,
o   730        service_accumulator: jmt$service_accumulator,
o   731        service_accumulator_since_swap: jmt$service_accumulator,
o   732        guaranteed_service_remaining: jmt$service_accumulator,
o   733        last_cptime: ost$cp_time_value,
```


JMT$INITIATED_JOB_LIST_ENTRY

```
o   734        last_page_fault_count: 0 .. 0fffffffff(16),
o   735        job_swap_counts: jmt$ijl_swap_counts,
o   736        swapout_reason: jmt$swapout_reasons,
o   737        priority: jmt$job_priority,
o   738        unaged_swap_queue_priority: jmt$job_priority,
o   739        swapin_q_priority_timestamp: ost$free_running_clock,
o   740        job_class: jmt$job_class,
o   741        service_class: jmt$service_class_index,
o   742      RECEND,
o   743
o   744      jmt$swap_data = record
o   745        swap_file_sfid: dmt$system_file_id,
o   746        swapping_io_error: iot$io_error,
o   747        swapped_job_page_count: 0 .. osc$max_page_frames,
o   748        swap_file_length_in_pages: 0 .. osc$max_page_frames,
o   749        asid_reassigned_timestamp: ost$free_running_clock,
o   750        timestamp: ost$free_running_clock,
o   751        swapout_timestamp: ost$free_running_clock,
o   752        reassigned_job_fixed_asti: mmt$ast_index,
o   753        swapped_job_entry: jmt$swapped_job_entry,
o   754      RECEND,
o   755
o   756      jst$swap_direction = (jsc$sd_in, jsc$sd_out),
o   757
o   758      jmt$service_counts = 0 .. 0fffffffffff(16);
o   759
o   760  {*copyc dft$mainframe_set
o   761  {*copyc dmt$system_file_id
o   762  {*copyc jmt$ajl_ordinal
o   763  {*copyc jmt$delayed_swapin_work
o   764  {*copyc jmt$dispatching_priority
o   765  {*copyc jmt$ijl_dispatching_control
o   766  {*copyc jmt$ijl_entry_status
o   767  {*copyc jmt$ijl_swap_status
o   768  {*copyc jmt$ijl_ordinal
o   769  {*copyc jmt$ijl_service_class_stats
o   770  {*copyc jmt$ijl_statistics
o   771  {*copyc jmt$job_class
o   772  {*copyc jmt$job_mode
o   773  {*copyc jmt$job_priority
o   774  {*copyc jmt$kjl_index
o   775  {*copyc jmt$queue_file_ijl_information
o   776  {*copyc jmt$service_accumulator
o   777  {*copyc jmt$service_class_index
o   778  {*copyc jmt$swapout_reasons
o   779  {*copyc jmt$swapped_job_entry
o   780  {*copyc jmt$system_supplied_name
o   781  {*copyc jst$ijl_swap_queue_link
o   782  {*copyc jst$io_control_information
o   783  {*copyc jst$swap_file_descriptor
o   784  {*copyc mmt$ast_index
o   785  {*copyc mmt$memory_reserve_request
o   786  {*copyc mmt$page_queue_list
o   787  {*copyc ost$global_task_id
o   788  {*copyc ost$cp_time
o   789  {*copyc ost$hardware_subranges
```

```
o   790 {*copyc jmt$dispatching_control
o   791 {*copyc iot$io_error


o   793 {------------------------------------  JMT$INITIATED_JOB_LIST_P          ----------------------------
o   794 {  Define structures used to manage the IJL arrays.
o   795 {  Force word alignment for performance.
o   796
o   797   TYPE
o   798     jmt$initiated_job_list_block = RECORD
o   799       in_use_count: ALIGNED [0 MOD 8] 0 .. jmc$max_ijl_index_count,
o   800       terminated_job: boolean,
o   801       index_p: ^ARRAY [jmt$ijl_block_index] OF jmt$initiated_job_list_entry,
o   802     RECEND,
o   803
o   804     jmt$initiated_job_list_p = ^ARRAY [0 .. *] OF jmt$initiated_job_list_block;
o   805
o   807 {*copyc jmt$ijl_ordinal
o   808 {*copyc jmt$initiated_job_list_entry


o   811 {------------------------------------  JMT$JOB_CLASS                    ----------------------------
o   812
o   813 { This deck defines the type for job classes.  Any time the job class
o   814 { table (jmv$job_class_table_p) needs to be scanned, the scan should
o   815 { be from jmc$system_job_class (the first defined class) to
o   816 { jmv$maximum_job_class_in_use (the index of the highest defined class).
o   817
o   818   CONST
o   819     jmc$null_job_class = 0,
o   820     jmc$system_job_class = 1,
o   821     jmc$maintenance_job_class = 2,
o   822     jmc$unassigned_job_class = 3,
o   823     jmc$lowest_site_job_class = 4,
o   824     jmc$minimum_job_classes = 3,
o   825     jmc$maximum_job_classes = 255;
o   826
o   827   TYPE
o   828     jmt$job_class = 0 .. jmc$maximum_job_classes;


o   830 {------------------------------------  JMT$JOB_CONTROL_BLOCK            ----------------------------
o   831
o   832 { This common deck contains the type declarations for the }
o   833 { JOB CONTROL BLOCK (JCB). }
o   834
o   835 { The JCB is used to maintain information on a job. }
o   836
o   837   { * * * * WARNING - If the length of this table grows * * * *}
o   838 {     * * * *     beyond 256 bytes, the constant <jrootsiz>* * * *}
o   839 {     * * * *     in the common deck ASMBCOM must be changed.* * *}
o   840 {     * * * *     There also be other changes required!!!     * * *}
o   841
```



```
o   842 TYPE
o   843   jmt$job_control_block = record
o   844
o   845 { The jcb_identifier MUST be the first field in the JCB.
o   846
o   847     jcb_identifier: 0 .. 0ffff(16),
o   848
o   849 { The following fields are referenced by assembly code; the offsets must not change unless
o   850 { the deck tma$task_switch is changed.
o   851
o   852     last_lpid_for_job: 0 .. 0ff(16),
o   853
o   854 { End of fields referenced by assembly code.
o   855
o   856     system_name: jmt$system_supplied_name,
o   857     jobname: jmt$user_supplied_name,
o   858     job_id: jmt$job_system_id,
o   859     user_id: ost$user_identification,
o   860     job_monitor_id: ost$global_task_id,
o   861     ijle_p: ^jmt$initiated_job_list_entry,
o   862     ijl_ordinal: jmt$ijl_ordinal,
o   863     server_mainframe_id: pmt$binary_mainframe_id,
o   864     last_execution_time: ost$free_running_clock,
o   865     cptime_next_age_working_set: ost$cp_time_value,
o   866     cptime_signal_last_sent: ost$cp_time_value,
o   867     signal_interval:  0 .. 0FFFFFFFF(16),
o   868     max_working_set_size: jmt$working_set_size,
o   869     min_working_set_size: jmt$working_set_size,
o   870     page_aging_interval: ost$aging_interval,
o   871     cyclic_aging_interval: ost$aging_interval,
o   872     detached_job_wait_time: jmt$detached_job_wait_time,
o   873     next_cyclic_aging_time: ost$free_running_clock,
o   874     sense_switches: pmt$sense_switches,
o   875     perm_file_job_warning_limit: sft$counter,
o   876     perm_file_job_warning_checking: boolean,
o   877     perm_file_job_maximum_limit: sft$counter,
o   878     temp_file_job_warning_limit: sft$counter,
o   879     temp_file_job_warning_checking: boolean,
o   880     temp_file_job_maximum_limit: sft$counter,
o   881     swapped_job_entry: jmt$swapped_job_entry,
o   882     account_project_specified: boolean,
o   883   recend;
o   884
o   885 {*copyc jmt$detached_job_wait_time
o   886 {*copyc jmt$ijl_ordinal
o   887 {*copyc jmt$initiated_job_list_entry
o   888 {*copyc jmt$job_system_id
o   889 {*copyc jmt$swapped_job_entry
o   890 {*copyc jmt$system_supplied_name
o   891 {*copyc jmt$user_supplied_name
o   892 {*copyc jmt$working_set_size
o   893 {*copyc ost$aging_interval
o   894 {*copyc ost$cp_time
o   895 {*copyc ost$global_task_id
o   896 {*copyc ost$hardware_subranges
o   897 {*copyc ost$user_identification
```

JMT$JOB_CONTROL_BLOCK

```
o   898 {*copyc pmt$binary_mainframe_id
o   899 {*copyc pmt$sense_switches
o   900 {*copyc sft$counter


o   902 {------------------------------------  JMT$JOB_MODE                    ----------------------------
o   903
o   904   TYPE
o   905     jmt$job_mode = (jmc$batch, jmc$interactive_connected,
o   906       jmc$interactive_cmnd_disconnect, jmc$interactive_line_disconnect,
o   907       jmc$interactive_sys_disconnect);


o   909 {------------------------------------  JMT$JOB_PRIORITY                ----------------------------
o   910
o   911   TYPE
o   912     jmt$job_priority = 0 .. 0ffffff(16);
o   913
o   914 { The following constants define the range of values permitted on SCL
o   915 { parameter definitions.
o   916
o   917   CONST
o   918     jmc$lowest_job_priority = 0,
o   919     jmc$highest_job_priority = 16000000;
o   920


o   922 {------------------------------------  JMT$JOB_STATE                   ----------------------------
o   923
o   924   TYPE
o   925     jmt$job_state = (jmc$deferred_job, jmc$queued_job, jmc$initiated_job,
o   926           jmc$terminating_job, jmc$completed_job);
o   927
o   928


o   930 {------------------------------------  JMT$JOB_STATISTICS              ----------------------------
o   931 {Define job statistics record. This record is not kept in any system table but is constructed
o   932 {when required from information kept in several system tables.
o   933
o   934   TYPE
o   935     jmt$job_statistics = record
o   936       cp_time: ost$cp_time,
o   937       paging_statistics: ost$paging_statistics,
o   938       working_set_size: 0 .. 0ffff(16),
o   939       ready_task_count: 0 .. 0ffff(16),
o   940     recend;
o   941
o   942 {*copyc OST$CP_TIME
o   943 {*copyc OST$PAGING_STATISTICS
```


JMT$SWAPPED_JOB_ENTRY

```
o   945 {------------------------------------  JMT$SWAPPED_JOB_ENTRY           ----------------------------
o   946
o   947
o   948 {  Type definition for swapped job entry.
o   949
o   950   TYPE
o   951     jmt$swapped_job_entry = record
o   952       available_modified_page_count: 0 .. osc$max_page_frames,
o   953       job_page_queue_count: ARRAY [mmt$job_page_queue_index]
o   954         OF 0 .. osc$max_page_frames,
o   955       swap_file_descriptor_page_count: 0 .. 65535,
o   956     recend;
o   957
o   958 {*copyc mmt$page_frame_queue_id
o   959 {*copyc ost$page_table
o   960 {*copyc ost$hardware_subranges


o   962 {------------------------------------  JSE$CONDITION_CODES             ----------------------------
o   963
o   964
```

```
JSE$CONDITION_CODES
ERROR CODES FOR JOB SWAPPER : 'JS' 0 .. 7

  966 ?? FMT (FORMAT := OFF) ??
  967   CONST
  968     jsc$min_ecc = (($INTEGER ('J') * 100(16)) + $INTEGER ('S')) * 1000000(16),
  969     jsc$min_ecc_js {base error condition code} = jsc$min_ecc {***KLUDGE***},
  970
  971
  972     jse$not_enough_mem_for_swap_in = jsc$min_ecc_js + 0,
  973     {E+ Not enough memory in free and available queue to swap job in.}
  974
  975     jse$unimplemented_subfunction = jsc$min_ecc_js + 1,
  976     {E+ Unimplemented job swapping monitor subfunction.}
  977
  978     jse$pt_full_on_swap_in = jsc$min_ecc_js + 2,
  979     {E+ Page table full on swap in.}
  980
  981     jse$unable_to_idle_all_tasks = jsc$min_ecc_js + 3,
  982     {E+ Unable to idle all tasks in the job.}
  983
  984     jse$job_terminated = jsc$min_ecc_js + 4,
  985     {E+ Attempted to swap a job that has terminated.}
  986
  987     jse$job_not_in_long_wait = jsc$min_ecc_js + 5,
  988     {E+ Attempted a conditional swap of a job not in long wait.}
  989
  990     jse$job_executing_non_swappable = jsc$min_ecc_js + 6,
  991     {E+ Job is executing and not swappable.}
  992
  993     jse$swapin_rejected_pages_freed = jsc$min_ecc_js + 7,
  994     {E+ Swapin rejected because swapout still active and pages have been freed.}
  995
  996     jse$swapout_and_job_swapped_out = jsc$min_ecc_js + 8,
  997     {E+ Job mode swapout request for job already swapped out.}
  998
  999     jse$swap_file_not_allocated = jsc$min_ecc_js + 9,
 1000     {E+ Swap file not allocated--job mode will be called to allocate.}
 1001
 1002     jse$swap_file_volume_unavail = jsc$min_ecc_js + 10,
 1003     {E+ Swap file volume is unavailable.}
 1004
 1005     jse$bad_swap_file_data_detected = jsc$min_ecc_js + 11;
 1006     {E+ Bad data was detected in the swap file--the job is dead.}
 1007
 1008 ?? FMT (FORMAT := ON) ??


 1011 {------------------------------------  JST$IO_CONTROL_INFORMATION        ----------------------------
 1012 {Define information kept in the AJL to control swap IO initiation
 1013
 1014   TYPE
 1015     jst$io_control_information = record
 1016         spd_index: mmt$page_frame_index,
 1017         next_queue_id: mmt$page_frame_queue_id,
 1018         next_pfti: mmt$page_frame_index,
 1019         stop_pfti: mmt$page_frame_index,
```



```
JST$IO_CONTROL_INFORMATION

 1020         swap_file_descriptor_pfti: mmt$page_frame_index,
 1021     RECEND;
 1022
 1023 {*copyc mmt$page_frame_queue_id
 1024 {*copyc mmt$page_frame_index


 1026 {------------------------------------  JST$SWAP_FILE_DESCRIPTOR          ----------------------------
 1027 {  Type definition for swap file descriptor.  This is part of the
 1028 {  swapped job image on mass storage and describes the swap file contents.
 1029
 1030   TYPE
 1031     jst$swap_file_descriptor = record
 1032       ijl_entry: jmt$initiated_job_list_entry,
 1033       swapped_job_entry: jmt$swapped_job_entry,
 1034       swapped_page_descriptors: jst$swapped_page_descriptors,
 1035     recend,
 1036
 1037     jst$swapped_page_descriptors = array [0 .. *] of
 1038       jst$swapped_page_descriptor,
 1039
 1040     jst$swapped_page_descriptor = record
 1041       pft_entry: mmt$page_frame_table_entry,
 1042       page_table_entry: ost$page_table_entry,
 1043       ast_entry: mmt$active_segment_table_entry,
 1044       entry_updated: boolean,
 1045       old_asid {used on swap in if asid reassigned} : ost$asid,
 1046       changed_asid: jst$changed_asid_entry,
 1047     recend,
 1048
 1049
 1050 {  The changed asid list is NOT logically part of the swapped page descriptor
 1051 {  but is imbedded in the same record entry since CYBIL does not
 1052 {  currently support record definitions that contain multiple adaptable
 1053 {  components.
 1054
 1055     jst$changed_asid_entry = RECORD
 1056       old_asid: ost$asid,
 1057       new_asid: ost$asid,
 1058       new_asti: mmt$ast_index,
 1059     RECEND;
 1060
 1061 {*copyc jmt$initiated_job_list_entry
 1062 {*copyc jmt$swapped_job_entry
 1063 {*copyc mmt$active_segment_table
 1064 {*copyc mmt$ast_index
 1065 {*copyc mmt$page_frame_index
 1066 {*copyc mmt$page_frame_table
 1067 {*copyc ost$hardware_subranges


 1069 {------------------------------------  JST$SWAP_STATE_STATISTICS         ----------------------------
 1070 {Define statistics kept for time spent in each swap state
 1071
```

JST$SWAP_STATE_STATISTICS

```
o  1072   TYPE
o  1073     jst$swap_state_statistics = ARRAY [jmt$ijl_swap_status] OF ARRAY [jmt$ijl_swap_status] OF
o  1074       jst$swap_state_statistics_entry,
o  1075     jst$swap_state_statistics_entry = RECORD
o  1076       total_time: integer,
o  1077       maximum_time: 0 .. 0ffffffff(16),
o  1078       count: 0 .. 0ffffffff(16),
o  1079     RECEND;
o  1080 {*copyc jmt$ijl_swap_status


o  1082 {------------------------------------  JSV$SWAP_STATUS_ID_ARRAY          ----------------------------
o  1083 {Define 2 character id used for identifying swap states.
o  1084
o  1085   VAR
o  1086       jsv$swap_status_id_array: [static, oss$job_paged_literal, read]
o  1087         array [jmt$ijl_swap_status] of string (2) := [
o  1088                 '  ',    { jmc$iss_null}
o  1089                 ' R',    { jmc$iss_executing}
o  1090                 'TI',    { jmc$iss_idle_tasks_initiated}
o  1091                 'TJ',    { jmc$iss_job_idle_tasks_complete}
o  1092                 'SO',    { jmc$iss_swapped_no_io}
o  1093                 'FA',    { jmc$iss_flush_am_pages}
o  1094                 'AJ',    { jmc$iss_job_allocate_swap_file}
o  1095                 'AW',    { jmc$iss_wait_allocate_swap_file}
o  1096                 'AF',    { jmc$iss_allocate_swap_file}
o  1097                 'JW',    { jmc$iss_wait_job_io_complete}
o  1098                 'JC',    { jmc$iss_job_io_complete}
o  1099                 'DW',    { jmc$iss_wait_allocate_sfd}
o  1100                 'AD',    { jmc$iss_allocate_sfd}
o  1101                 'SD',    { jmc$iss_swapped_io_cannot_init}
o  1102                 'OS',    { jmc$iss_initiate_swapout_io}
o  1103                 'OW',    { jmc$iss_wait_swapout_io_init}
o  1104                 'OI',    { jmc$iss_swapout_io_initiated}
o  1105                 'OC',    { jmc$iss_swapout_io_complete}
o  1106                 'S2',    { jmc$iss_swapped_io_complete}
o  1107                 'FM',    { jmc$iss_free_swapped_memory}
o  1108                 ' S',    { jmc$iss_swapout_complete}
o  1109                 'IR',    { jmc$iss_swapin_requested}
o  1110                 'IS',    { jmc$iss_swapin_resource_claimed}
o  1111                 'IW',    { jmc$iss_wait_swapin_io_init}
o  1112                 'II',    { jmc$iss_swapin_io_initiated}
o  1113                 'IC'];   { jmc$iss_swapin_io_complete}
o  1115 {*copyc jmt$ijl_swap_status
o  1116 {*copyc oss$job_paged_literal


o  1119 {------------------------------------  MMC$DEFAULT_SDT_LENGTH           ----------------------------
o  1120 {  {*copyc mmc$default_sdt_length
o  1121 {  This common deck defines constants used by memory manager in job mode.
o  1122 {  Also used as length of bit map ( osv$system_privilege_map ) set by
o  1123 {  system core segment manager.
o  1124
o  1125   CONST
```


MMC$DEFAULT_SDT_LENGTH

```
o  1126     mmc$default_sdt_length = 100;


o  1128 {------------------------------------  MMC$SEGMENT_MANAGER_DEFAULTS     ----------------------------
o  1129
o  1130
o  1131 {  Define constants that are global to the segment manager.
o  1132
o  1133   CONST
o  1134     mmc$default_preset_value = 0,
o  1135     mmc$default_current_seg_length = 0,
o  1136     mmc$default_maximum_seg_length = 7fffffff(16),
o  1137     mmc$default_clear_space = FALSE;


o  1139 {------------------------------------  MMD$SEGMENT_ACCESS_CONDITION     ----------------------------
o  1140 {
o  1141 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  1142 {   make the appropriate changes in the corresponding display procedures in the
o  1143 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  1144 {
o  1145
o  1146   CONST
o  1147     mmc$sac_read_beyond_eoi  = 1,
o  1148     mmc$sac_read_write_beyond_msl  = 2,
o  1149     mmc$sac_segment_access_error = 3,
o  1150     mmc$sac_key_lock_violation = 4,
o  1151     mmc$sac_ring_violation = 5,
o  1152     mmc$sac_io_read_error  = 6,
o  1153     mmc$sac_no_append_permission = 7,
o  1154     mmc$sac_tape_system_failure = 8,
o  1155     mmc$sac_file_server_terminated = 9,
o  1156     mmc$sac_pf_space_limit_exceeded = 10,
o  1157     mmc$sac_tf_space_limit_exceeded = 11;
o  1158
o  1159   TYPE
o  1160     mmt$segment_access_condition = record
o  1161       identifier: pmt$condition_identifier,
o  1162       segment: ^cell,
o  1163     recend;
o  1164
o  1165 {*copyc PMT$CONDITION_IDENTIFIER


o  1167 {------------------------------------  MME$CONDITION_CODES              ----------------------------
o  1168
o  1169   CONST
o  1170     mmc$ = (($INTEGER ('M') * 100(16)) + $INTEGER ('M')) * 1000000(16);
o  1171
```

```
MME$CONDITION_CODES
MMDERR : ERROR CODES FOR MEM MGR  0 .. 5999

o  1173 ?? FMT (FORMAT := OFF) ??
o  1174
o  1175   CONST
o  1176     mme$invalid_sfid = mmc$ + 1,
o  1177     {F The specified SFID is invalid.}
o  1178
o  1179     mme$page_table_full = mmc$ + 2,
o  1180     {F Page table is full.}
o  1181
o  1182     mme$no_free_pages = mmc$ + 3,
o  1183     {F There are no free pages.}
o  1184
o  1185     mme$job_file_tables_full = mmc$ + 4,
o  1186     {F There are not any free entries in the job file table.}
o  1187
o  1188     mme$page_not_in_page_table = mmc$ + 5,
o  1189     {F Page was not found in the page table.}
o  1190
o  1191     mme$invalid_pva = mmc$ + 6,
o  1192     {F Invalid PVA specified on the request.}
o  1193
o  1194     mme$unable_to_get_fde_fomp = mmc$ + 7,
o  1195     {F Unable to get the file descriptor entry during the
o  1196     {   FETCH_OFFSET_MODIFIED_PAGES request.}
o  1197
o  1198     mme$page_frame_not_assigned = mmc$ + 8,
o  1199     {F Page frame was not assigned to a page being locked.}
o  1200
o  1201     mme$read_beyond_eoi = mmc$ + 9,
o  1202     {F Tried to read beyond EOI on read only segment - PVA = +P, P = +P.}
o  1203
o  1204     mme$io_read_error = mmc$ + 10,
o  1205     {F Uncorrected IO error - PVA = +P, P = +P.}
o  1206
o  1207     mme$read_write_beyond_msl = mmc$ + 11,
o  1208     {F Tried to read/write beyond maximum segment length - PVA = +P, P = +P.}
o  1209
o  1210     mme$segment_access_error  = mmc$ + 12,
o  1211     {F Attempt to access segment with access not granted - PVA = +P, P = +P.}
o  1212
o  1213     mme$computed_asti_out_of_range = mmc$ + 13,
o  1214     {F The computed ASTI was out of the range of the image AST; procedure = +P}
o  1215
o  1216     mme$ring_violation = mmc$ + 14,
o  1217     {F Not validated to access a segment from the present ring,
o  1218     { PVA = +P, P = +P.}
o  1219
o  1220     mme$invalid_ring_brackets = mmc$ + 15,
o  1221     {F Invalid ring brackets in the SDTE.}
o  1222
o  1223     mme$binding_attribute_invalid = mmc$ + 16,
o  1224     {F Attempt to set the binding attribute from above ring 3.}
o  1225
o  1226     mme$execute_global_invalid = mmc$ + 17,
o  1227     {F Execute_global attribute can not be specified.}
```



```
MME$CONDITION_CODES
MMDERR : ERROR CODES FOR MEM MGR  0 .. 5999

o  1228
o  1229     mme$software_attribute_invalid = mmc$ + 18,
o  1230     {F Attempt to specify software attribute from beyond ring 3.}
o  1231
o  1232     mme$unused_error_condition_19 = mmc$ + 19,
o  1233     {F Unused}
o  1234
o  1235     mme$invalid_close_segment_req = mmc$ + 20,
o  1236     {F Attempt to close/delete segment that is not in callers write bracket
o  1237     {  or segment that is not a user segment.}
o  1238
o  1239     mme$caller_not_in_read_bracket = mmc$ + 21,
o  1240     {F Caller not in read bracket of segment specified.}
o  1241
o  1242     mme$caller_not_in_write_bracket = mmc$ + 22,
o  1243     {F Caller not in write bracket of segment specified.}
o  1244
o  1245     mme$execute_local_invalid = mmc$ + 23,
o  1246     {F Attempt to set EXECUTE_LOCAL attribute from above ring 3.}
o  1247
o  1248     mme$set_unmodifiable_attribute = mmc$ + 24,
o  1249     {F Attempt to change segment attribute that can not be modified.}
o  1250
o  1251     mme$segment_table_is_full = mmc$ + 25,
o  1252     {F Segment table is full.}
o  1253
o  1254     mme$segment_number_is_in_use = mmc$ + 26,
o  1255     {F Segment number specified is already in use.}
o  1256
o  1257     mme$segment_number_not_in_use = mmc$ + 27,
o  1258     {F Segment number specified is not in use.}
o  1259
o  1260     mme$segment_number_too_big = mmc$ + 28,
o  1261     {F Segment number is beyond the range of the segment table.}
o  1262
o  1263     mme$unsupported_keyword = mmc$ + 29,
o  1264     {F Keyword specified for the segment attributes is invalid.}
o  1265
o  1266     mme$sdt_or_sdtx_exist = mmc$ + 30,
o  1267     {F Attempt to create inherited SDT and SDT or SDTX already exists.}
o  1268
o  1269     mme$no_pages_found_for_move = mmc$ + 31,
o  1270     {F No pages which could be moved were found. }
o  1271
o  1272     mme$asid_specified = mmc$ + 32,
o  1273     {F ASID specified as attribute and segment manager not called from ring 1.}
o  1274
o  1275     mme$invalid_asid_specified = mmc$ + 33,
o  1276     {F The specified ASID is not one of the reserved ASID's.}
o  1277
o  1278     mme$unused_error_condition_34 = mmc$ + 34,
o  1279     {F Unused}
o  1280
o  1281     mme$page_already_locked = mmc$ + 35,
o  1282     {F Attempt to lock a page that is already locked.}
```

```
MME$CONDITION_CODES
MMDERR : ERROR CODES FOR MEM MGR  0 .. 5999

o  1283
o  1284      mme$segment_not_assigned_device = mmc$ + 36,
o  1285      {F Attempt to do IO on a segment with no backing file.}
o  1286
o  1287      mme$unused_error_condition_37 = mmc$ + 37,
o  1288      {F Unused}
o  1289
o  1290      mme$not_valid_in_page_table = mmc$ +38,
o  1291      {F Page in memory but valid bit not set in page table.}
o  1292
o  1293      mme$stack_overflow_on_push = mmc$ + 39,
o  1294      {F Stack overflow on PUSH.}
o  1295
o  1296      mme$invalid_task_id = mmc$ + 40,
o  1297      {F The taskid specified on the call is invalid.}
o  1298
o  1299      mme$no_matching_offset = mmc$ + 41,
o  1300      {F On subsequent MMP$FETCH_UNWRITTEN_PAGES request, the matching
o  1301      {  offset could not be found in the page frame table.}
o  1302
o  1303      mme$invalid_request = mmc$ + 42,
o  1304      {F Invalid request code.}
o  1305
o  1306      mme$io_write_error = mmc$ + 43,
o  1307      {F Io error on trying to write a page to disk.}
o  1308
o  1309      mme$segment_not_pageable = mmc$ + 44,
o  1310      {F Attempt to age out a page in a non pageable segment.}
o  1311
o  1312      mme$segment_origin_invalid = mmc$ + 45,
o  1313      {F Attempt to set segment origin from above ring 1.}
o  1314
o  1315      mme$segment_origin_change = mmc$ + 46,
o  1316      {F Attempt to modify segment origin.}
o  1317
o  1318      mme$lock_unlock_invalid_length = mmc$ + 47,
o  1319      {F Lock/unlock request and length + offset > maximum segment length.}
o  1320
o  1321      mme$unused_error_condition_48 = mmc$ + 48,
o  1322      {F Unused}
o  1323
o  1324      mme$page_not_locked = mmc$ + 49,
o  1325      {F Request to clear lock and page not locked.}
o  1326
o  1327      mme$exceeds_max_lock_page_count = mmc$ + 50,
o  1328      {F Lock page request will exceed maximum locked pages allowed.}
o  1329
o  1330      mme$no_write_access = mmc$ + 51,
o  1331      {F Set segment length on segment without write access.}
o  1332
o  1333      mme$lock_ring_1_stack_from_r1 = mmc$ + 52,
o  1334      {F Monitor request to lock ring 1 stack issued from ring 1.}
o  1335
o  1336      mme$write_beyond_eoi_no_append = mmc$ + 53,
o  1337      {F Write beyond segment eoi with no append permission - PVA = +P, P = +P.}
```



```
MME$CONDITION_CODES
MMDERR : ERROR CODES FOR MEM MGR  0 .. 5999

o  1338
o  1339      mme$unused_error_condition_54 = mmc$ + 54,
o  1340      {F Unused}
o  1341
o  1342      mme$segment_locked_by_task = mmc$ + 55,
o  1343      {F Segment already locked by current task}
o  1344
o  1345      mme$segment_locked_another_task = mmc$ + 56,
o  1346      {F Segment locked by another task}
o  1347
o  1348      mme$segment_not_locked = mmc$ + 57,
o  1349      {F Segment not locked by current task}
o  1350
o  1351      mme$temporary_reject = mmc$ + 58,
o  1352      {F Resources required to process request are temporarily unavailable}
o  1353
o  1354      mme$nil_io_control_block = mmc$ + 59,
o  1355      {F The io control block has not been allocated.}
o  1356
o  1357      mme$full_io_control_block = mmc$ + 60,
o  1358      {F The io control block is full.}
o  1359
o  1360      mme$page_found_in_memory = mmc$ + 61,
o  1361      {F Page found in memory.}
o  1362
o  1363      mme$pf_space_limit_exceeded = mmc$ + 62,
o  1364      {F The maximum permanent file space limit has been exceeded -
o  1365      { PVA = +P, P = +P.}
o  1366
o  1367      mme$tf_space_limit_exceeded = mmc$ + 63,
o  1368      {F The maximum temporary file space limit has been exceeded -
o  1369      { PVA = +P, P = +P.}
o  1370
o  1371      mme$disk_flaws = mmc$ + 64,
o  1372      {F Page could not be written to disk because of disk flaws.}
o  1373
o  1374      mme$invalid_io_status_ptrs = mmc$ + 65,
o  1375      {F Attempt to check status of io with pointers for which no io
o  1376      { has been requested.}
o  1377
o  1378      mme$write_status_complete = mmc$ + 66,
o  1379      {F Status of the write request is complete.}
o  1380
o  1381      mme$stack_overflow = mmc$ + 67,
o  1382      {F Stack overflow - PVA = +P, P = +P.}
o  1383
o  1384      mme$request_length_too_long = mmc$ + 68,
o  1385      {F Length on read/write request exceeds 65536 bytes.}
o  1386
o  1387      mme$invalid_pva_formed = mmc$ + 69,
o  1388      {F Offset plus length created an invalid pva.}
o  1389
o  1390      mme$ref_to_unrecovered_file = mmc$ + 70,
o  1391      {F The file backing the segment being referenced has not been recovered.}
o  1392
```

MME$CONDITION_CODES
MMDERR : ERROR CODES FOR MEM MGR 0 .. 5999

```
o  1393      mme$length_not_0_mod_16384 = mmc$ + 71,
o  1394      {F The length of the shadow file must be a multiple of 16384.}
o  1395
o  1396      mme$address_not_0_mod_16384 = mmc$ + 72,
o  1397      {F The starting address of the shadow file must be a multiple of 16384.}
o  1398
o  1399      mme$invalid_shadow_segment = mmc$ + 73,
o  1400      {F The specified shadow file is not valid.}
o  1401
o  1402      mme$init_shadow_improper_seg = mmc$ + 74,
o  1403      {F The segment specified is not appropriate for shadowing.}
o  1404
o  1405      mme$unused_error_condition_75 = mmc$ + 75,
o  1406      {F Unused}
o  1407
o  1408      mme$wired_or_fixed_segs_illegal = mmc$ + 76,
o  1409      {F The use of wired or fixed segments prohibited with this request.}
o  1410
o  1411      mme$unused_error_condition_77 = mmc$ + 77,
o  1412      {F Unused}
o  1413
o  1414      mme$memory_not_avail_for_assign = mmc$ + 78,
o  1415      {F Memory is not currently available for assign_pages request.}
o  1416
o  1417      mme$dm_assign_active = mmc$ + 79,
o  1418      {F Backing file is being assigned for the segment.}
o  1419
o  1420      mme$assign_length_too_long = mmc$ + 80,
o  1421      {F The length requested would cause the working set to get too large.}
o  1422
o  1423      mme$length_must_be_positive = mmc$ + 81,
o  1424      {F The requested length must be positive.}
o  1425
o  1426      mme$wait_so_other_tasks_can_run = mmc$ + 82,
o  1427      {F Cause task to wait so other tasks can execute.}
o  1428
o  1429      mme$cannot_wait_for_memory = mmc$ + 83,
o  1430      {F Job is non swappable -- cannot wait to assign memory. }
o  1431
o  1432      mme$illegal_segment_origin_chg = mmc$ + 84,
o  1433      {F An illegal attempt was made to change the segment origin.}
o  1434
o  1435      mme$invalid_shared_taskid = mmc$ + 85,
o  1436      {F An illegal taskid was found either opening or closing a
o  1437      { shared stack segment.}
o  1438
o  1439      mme$volume_unavailable = mmc$ + 86,
o  1440      {F A reference has been made to a segment on a volume that is
o  1441      { not available.}
o  1442
o  1443      mme$unused_error_condition_87 = mmc$ + 87,
o  1444      {F Unused}
o  1445
o  1446      mme$wired_seg_length_too_large = mmc$ + 88,
o  1447      {F The requested length of the wired segment exceeds 65536 bytes.}
```


MME$CONDITION_CODES
MMDERR : ERROR CODES FOR MEM MGR 0 .. 5999

```
o  1448
o  1449      mme$length_not_page_size_mult = mmc$ + 90,
o  1450      {F The requested length on mmp$move_pages must be a page size multiple.}
o  1451
o  1452      mme$pva_not_on_page_boundary = mmc$ + 91,
o  1453      {F The specified pvas on mmp$move_pages must be on a page boundary.}
o  1454
o  1455      mme$unused_error_condition_92 = mmc$ + 92,
o  1456      {F Unused}
o  1457
o  1458      mme$modified_source_page_reject = mmc$ + 93,
o  1459      {F Source page was modified on mmp$move_pages_request.}
o  1460
o  1461      mme$source_page_not_in_memory = mmc$ + 94,
o  1462      {F Source page not in memory on mmp$move_pages request.}
o  1463
o  1464      mme$invalid_length_requested = mmc$ + 95,
o  1465      {F Length is greater than maximum allowed or less than minimum allowed.}
o  1466
o  1467      mme$io_active_on_move_page = mmc$ + 96,
o  1468      {F Source page had io active om mmp$move_pages request.}
o  1469
o  1470      mme$unsupported_segment_kind = mmc$ + 97,
o  1471      {F Interface does not support segments of this kind--must be mmc$sk_file.}
o  1472
o  1473      mme$unused_error_condition_98 = mmc$ + 98,
o  1474      {F Unused}
o  1475
o  1476      mme$invalid_seg_for_prealloc = mmc$ + 99,
o  1477      {F The segment must be assigned a file for preallocation to occur.}
o  1478
o  1479      mme$contig_mem_seg_violation = mmc$ + 100,
o  1480      {F Segment must be either wired or job_fixed to assign contiguous memory.}
o  1481
o  1482      mme$unable_to_assign_contig_mem = mmc$ + 101,
o  1483      {F Unable to allocate the requested amount of contiguous memory.}
o  1484
o  1485      mme$pages_already_assigned = mmc$ + 102,
o  1486      {F Pages within the range (PVA-->PVA+length) are already assigned.}
o  1487
o  1488      mme$update_req_write_permission = mmc$ + 103,
o  1489      {F Updating the passive segment requires write permission.}
o  1490
o  1491      mme$unused_error_condition_104 = mmc$ + 104,
o  1492      {F Unused}
o  1493
o  1494      mme$cant_shadow_transient_segs = mmc$ + 105,
o  1495      {F Transient segments can not be shadowed.}
o  1496
o  1497      mme$file_server_terminated = mmc$ + 106,
o  1498      {F The segment is located on a terminated file_server and therefore it..
o  1499      {  cannot be accessed - PVA = +P, P = +P.}
o  1500
o  1501      mme$preallocate_failed = mmc$ + 107,
o  1502      {F The mmp$preallocate_file_space request could not be completed normally.
```

MME$CONDITION_CODES
MMDERR : ERROR CODES FOR MEM MGR 0 .. 5999

```
o  1503
o  1504      mme$unable_to_assign_fde = mmc$ + 108,
o  1505      {F A file descriptor entry could not be assigned for this segment.
o  1506
o  1507      mme$last_error_code = mmc$ + 5998;
o  1508      {F Dummy error code to eliminate feature conflicts. }
o  1509 ?? FMT (FORMAT := ON) ??


o  1512 {------------------------------------  MMT$ACTIVE_SEGMENT_TABLE          ----------------------------
o  1513
o  1514 { Define Active Segment Table - (AST)}
o  1515
o  1516   TYPE
o  1517     mmt$active_segment_table_entry = record
o  1518       pft_link: mmt$link,
o  1519       pages_in_memory: 0 .. osc$max_page_frames,
o  1520       ijl_ordinal: jmt$ijl_ordinal,
o  1521       case in_use: boolean of
o  1522       = FALSE =
o  1523         time_freed: ost$free_running_clock,
o  1524         asid: ost$asid,          {???}
o  1525       = TRUE =
o  1526         queue_id: mmt$page_frame_queue_id,
o  1527         sfid: gft$system_file_identifier,
o  1528         include_pages_in_dump: boolean,
o  1529       casend,
o  1530     recend,
o  1531
o  1532     mmt$active_segment_table = array [0 .. * ] of
o  1533       mmt$active_segment_table_entry;
o  1534
o  1535 {*copyc gft$system_file_identifier
o  1536 {*copyc jmt$ijl_ordinal
o  1537 {*copyc mmt$link
o  1538 {*copyc mmt$page_frame_queue_id
o  1539 {*copyc ost$free_running_clock
o  1540


o  1542 {------------------------------------  MMT$AGING_STATISTICS             ----------------------------
o  1543 {This deck defines the record used for keeping memory manager AGING statistics.
o  1544
o  1545   TYPE
o  1546     mmt$aging_statistics = record
o  1547       force_aggressive_aging: integer,
o  1548       aggressive_age_shared_queue: integer,
o  1549       aggressive_age_job_queues: integer,
o  1550       aggressive_aging_failed: integer,
o  1551       age_cp_bound_job: integer,
o  1552       remove_unmodified_page_from_ws: integer,
o  1553       remove_modified_page_from_ws: integer,
o  1554       page_written_to_disk: integer,
```


MMT$AGING_STATISTICS

```
o  1555       multiple_pages_written_to_disk: integer,
o  1556       calls_to_age_jws: integer,
o  1557       age_exceeds_aif: integer,
o  1558       age_exceeds_aic: integer,
o  1559       age_unused_page_in_shared_queue: integer,
o  1560       age_sys_shared_queue: ARRAY [mmc$pq_shared_first .. mmc$pq_shared_last_sys] of integer,
o  1561       write_aged_out_page: integer,
o  1562       write_forced_out_page: integer,
o  1563       write_pt_full_page: integer,
o  1564       write_avail_mod_page: integer,
o  1565       write_page_failed: integer,
o  1566     recend;
o  1567
o  1568 {*copyc mmt$page_frame_queue_id


o  1570 {------------------------------------  MMT$AST_INDEX                    ----------------------------
o  1571 {Define index to AST - table used for managing ASIDs.
o  1572
o  1573   TYPE
o  1574     mmt$ast_index = 0 .. 0ffff(16);


o  1576 {------------------------------------  MMT$ASYNC_WORK_LIST              ----------------------------
o  1577 {Define record used to pass requests to the Memory Manager periodic routine
o  1578
o  1579   TYPE
o  1580     mmt$async_work_list = RECORD
o  1581       reclaim_astes: boolean,
o  1582       pt_full: boolean,
o  1583       pt_full_sva: ost$system_virtual_address,
o  1584       pt_full_aste_p: ^mmt$active_segment_table_entry,
o  1585     RECEND;
o  1586 {*copyc OST$HARDWARE_SUBRANGES
o  1587 {*copyc MMT$ACTIVE_SEGMENT_TABLE


o  1589 {------------------------------------  MMT$ATTRIBUTE_KEYWORD            ----------------------------
o  1590
o  1591 {
o  1592 {    Type definitions for memory management.
o  1593 {
o  1594 {
o  1595 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  1596 {   make the appropriate changes in the corresponding display procedures in the
o  1597 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  1598 {
o  1599
o  1600   TYPE
o  1601     mmt$attribute_keyword = (mmc$kw_null_keyword, mmc$kw_ring_numbers,
o  1602       mmc$kw_segment_number, mmc$kw_current_segment_length,
o  1603       mmc$kw_max_segment_length, mmc$kw_clear_space,
o  1604       mmc$kw_error_exit_procedure, mmc$kw_software_attributes, mmc$kw_gl_key,
```

```
o  1605       mmc$kw_preset_value, mmc$kw_segment_access_control, mmc$kw_asid,
o  1606       mmc$kw_inheritance, mmc$kw_hardware_attributes,
o  1607       mmc$kw_shadow_segment, mmc$kw_wired_segment, mmc$kw_ps_transfer_size ),
o  1608
o  1609 { * * *  WARNING - mmc$kw_hardware_attributes and mmc$kw_ring_numbers
o  1610 { * * *           will be deleted. Replace with mmc$kw_access_control.
o  1611
o  1612
o  1613 {Define the record used to describe an attribute of a segment.}
o  1614
o  1615     mmt$attribute_descriptor = record
o  1616       case keyword: mmt$attribute_keyword of
o  1617       = mmc$kw_ring_numbers =
o  1618         r1: ost$ring,
o  1619         r2: ost$ring,
o  1620       = mmc$kw_segment_number =
o  1621         segnum: ost$segment,
o  1622       = mmc$kw_current_segment_length =
o  1623         current_length: ost$segment_length,
o  1624       = mmc$kw_max_segment_length =
o  1625         max_length: ost$segment_length,
o  1626       = mmc$kw_gl_key =
o  1627         gl_key: ost$key_lock,
o  1628       = mmc$kw_clear_space =
o  1629         clear_space: boolean,
o  1630       = mmc$kw_preset_value =
o  1631         preset_value: pmt$initialization_value,
o  1632       = mmc$kw_error_exit_procedure = {? ? may use conditions ? ?}
o  1633         err_exit_proc: ^procedure (pva: ^cell;
o  1634           VAR status: ost$status),
o  1635       = mmc$kw_hardware_attributes =
o  1636         hardware_attri_set: mmt$hardware_attribute_set,
o  1637       = mmc$kw_software_attributes =
o  1638         software_attri_set: mmt$software_attribute_set,
o  1639       = mmc$kw_segment_access_control =
o  1640         access_control: ost$segment_access_control,
o  1641       = mmc$kw_asid =
o  1642         asid: ost$asid,
o  1643       = mmc$kw_inheritance =
o  1644         inheritance: mmt$segment_inheritance,
o  1645       = mmc$kw_shadow_segment =
o  1646         shadow_p: ^cell,
o  1647         shadow_length: ost$segment_length,
o  1648       = mmc$kw_wired_segment =
o  1649         wired_segment_length: ost$segment_length,
o  1650         contiguous_real_memory: boolean,          {NOT SUPPORTED FOR 1.2.2}
o  1651       = mmc$kw_ps_transfer_size =
o  1652         ps_transfer_size: ost$segment_length,
o  1653       casend,
o  1654     recend;
o  1655
o  1656
o  1657   TYPE
o  1658     mmt$hardware_attributes = (mmc$ha_read, mmc$ha_read_key_lock,
o  1659       mmc$ha_binding, mmc$ha_write, mmc$ha_write_key_lock, mmc$ha_execute,
o  1660       mmc$ha_execute_local, mmc$ha_execute_global, mmc$ha_cache_bypass),
```



```
o  1661
o  1662
o  1663 {  The software attributes from mmc$sa_wired to mmc$sa_stack can not be set
o  1664 {  from above ring 3.
o  1665
o  1666     mmt$software_attributes = (mmc$sa_wired, mmc$sa_fixed,
o  1667       mmc$sa_stack, mmc$sa_read_transfer_unit, mmc$sa_free_behind,
o  1668       mmc$sa_no_append, mmc$sa_job_shared),
o  1669
o  1670     mmt$hardware_attribute_set = set of mmt$hardware_attributes,
o  1671
o  1672     mmt$software_attribute_set = set of mmt$software_attributes;
o  1673
o  1674
o  1675
o  1676 {Define type declarations for specifying  pointers.}
o  1677
o  1678   TYPE
o  1679     mmt$segment_pointer_kind = (mmc$cell_pointer, mmc$sequence_pointer,
o  1680       mmc$heap_pointer),
o  1681
o  1682     mmt$segment_pointer = record
o  1683       case kind: mmt$segment_pointer_kind of
o  1684       = mmc$cell_pointer =
o  1685         cell_pointer: ^cell,
o  1686       = mmc$sequence_pointer =
o  1687         seq_pointer: ^SEQ ( * ),
o  1688       = mmc$heap_pointer =
o  1689         heap_pointer: ^HEAP ( * ),
o  1690       casend,
o  1691     recend;
o  1692
o  1693
o  1694 {*copyc MMT$SEGMENT_INHERITANCE
o  1695 {*copyc OSD$VIRTUAL_ADDRESS
o  1696 {*copyc OST$HARDWARE_SUBRANGES
o  1697 {*copyc OST$SEGMENT_ACCESS_CONTROL
o  1698 {*copyc OST$STATUS
o  1699 {*copyc PMT$INITIALIZATION_VALUE


o  1701 {------------------------------------  MMT$BUFFER_DESCRIPTOR  ----------------------------
o  1702 {  Define the type definitions of the buffer descriptor for locking and
o  1703 {  unlocking pages when doing physical io.
o  1704
o  1705   TYPE
o  1706     mmt$buffer_descriptor = record
o  1707       page_count: mmt$rma_list_length,
o  1708       case buffer_descriptor_type: mmt$buffer_descriptor_type of
o  1709       = mmc$bd_paging_io, mmc$bd_explicit_io =
o  1710         sva: ost$system_virtual_address,
o  1711       = mmc$bd_job_swapping_io =
o  1712         ijl_ordinal: jmt$ijl_ordinal,
o  1713       casend,
o  1714     recend,
```

```
o  1715
o  1716      mmt$buffer_descriptor_type = (mmc$bd_paging_io, mmc$bd_job_swapping_io,
o  1717        mmc$bd_explicit_io);
o  1718
o  1719 (*copyc jmt$ijl_ordinal
o  1720 (*copyc MMT$RMA_LIST
o  1721 (*copyc OST$HARDWARE_SUBRANGES


o  1723 (------------------------------------  MMT$IMAGE_PAGE_DESCRIPTION  ----------------------------
o  1724
o  1725
o  1726 ( This deck defines the record that describes pages recovered from the image
o  1727 ( file.
o  1728
o  1729    TYPE
o  1730      mmt$image_page_description = record
o  1731        valid_desc_count: 0 .. 7fffffff(16),
o  1732        pagesize: ost$page_size,
o  1733        page_desc: array [ * ] of mmt$page_descriptor,
o  1734      recend,
o  1735
o  1736      mmt$page_descriptor = record
o  1737        image_pva: ^cell,
o  1738        file_offset: ost$segment_offset,
o  1739      recend;
o  1740
o  1741 (*copyc osd$virtual_address
o  1742 (*copyc ost$hardware_subranges
o  1743 (*copyc ost$page_size


o  1745 (------------------------------------  MMT$IO_TYPE  -------------------------------------------
o  1746 (Define the ordinal list that defines the type of IO taking place into
o  1747 (a page frame.
o  1748
o  1749    TYPE
o  1750      mmt$io_type = (mmc$it_none, mmc$it_explicit_input, mmc$it_explicit_output,
o  1751        mmc$it_implicit_input, mmc$it_implicit_output, mmc$it_swap_out, mmc$it_swap_in);


o  1753 (------------------------------------  MMT$LINK  ----------------------------------------------
o  1754
o  1755 (Define types for managing the circular lists that link PFT entries)
o  1756 (together.)
o  1757
o  1758    TYPE
o  1759      mmt$link = record
o  1760        bkw,
o  1761        fwd: mmt$page_frame_index,
o  1762      recend;
o  1763
o  1764 (*copyc MMT$PAGE_FRAME_INDEX
```



```
o  1766 (------------------------------------  MMT$LOCKED_PAGE  ----------------------------------------
o  1767
o  1768
o  1769 (  Type definition for locked page types.
o  1770
o  1771    TYPE
o  1772      mmt$locked_page = (mmc$lp_not_locked, mmc$lp_aging_lock,
o  1773        mmc$lp_write_protected_lock, mmc$lp_page_in_lock,
o  1774        mmc$lp_server_allocate_lock);


o  1776 (------------------------------------  MMT$LUS_DECLARATIONS  ----------------------------------
o  1777
o  1778 ( This decks defines type for the mmp$lock_segment and  mmp$unlock_segment
o  1779 ( requests.
o  1780 (
o  1781 ( NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  1782 (   make the appropriate changes in the corresponding display procedures in the
o  1783 (   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  1784 (
o  1785
o  1786    TYPE
o  1787      mmt$lus_lock_type = (mmc$lus_no_lock, mmc$lus_lock_for_read,
o  1788            mmc$lus_lock_for_write),
o  1789      mmt$lus_page_disposition = (mmc$lus_none, mmc$lus_protected_write,
o  1790            mmc$lus_write, mmc$lus_remove_from_working_set, mmc$lus_free);
o  1791


o  1793 (------------------------------------  MMT$MAKE_PT_ENTRY_STATUS  -------------------------------
o  1794 (Define status values for mmp$make_pt_entry.
o  1795
o  1796    TYPE
o  1797      mmt$make_pt_entry_status = (mmc$mpt_done, mmc$mpt_page_table_full,
o  1798        mmc$mpt_page_already_exists);


o  1800 (------------------------------------  MMT$PAGE_FRAME_INDEX  ----------------------------------
o  1801    TYPE
o  1802      mmt$page_frame_index = 0 .. osc$max_page_frames - 1;
o  1803 (*copyc OST$PAGE_TABLE


o  1805 (------------------------------------  MMT$PAGE_FRAME_QUEUE_ID  -------------------------------
o  1806
o  1807 (Define queue ids for the threads that run thru the Page Frame Table.)
o  1808 ( NOTE:  There are places in memory manager--mmp$process_assign_pages for one--
o  1809 ( that assumes that if a page table entry is NOT VALID, the page is in either
o  1810 ( the available or available modified queue ONLY.  If there are other queues
```

```
o  1811 { added that can have non-valid pages, memory manager must be changed.
o  1812
o  1813
o  1814   CONST
o  1815     mmc$pq_free = 0,
o  1816     mmc$pq_avail = 1,
o  1817     mmc$pq_avail_modified = 2,
o  1818     mmc$pq_wired = 3,
o  1819
o  1820     mmc$pq_shared_task_service    = 4,
o  1821     mmc$pq_shared_pf_execute      = 5,
o  1822     mmc$pq_shared_pf_non_execute  = 6,
o  1823     mmc$pq_shared_device_file     = 7,
o  1824     mmc$pq_shared_file_server     = 8,
o  1825     mmc$pq_shared_other           = 9,
o  1826
o  1827     mmc$pq_shared_site_01         = 10,
o  1828     mmc$pq_shared_site_02         = 11,
o  1829     mmc$pq_shared_site_03         = 12,
o  1830     mmc$pq_shared_site_04         = 13,
o  1831     mmc$pq_shared_site_05         = 14,
o  1832     mmc$pq_shared_site_06         = 15,
o  1833     mmc$pq_shared_site_07         = 16,
o  1834     mmc$pq_shared_site_08         = 17,
o  1835     mmc$pq_shared_site_09         = 18,
o  1836     mmc$pq_shared_site_10         = 19,
o  1837     mmc$pq_shared_site_11         = 20,
o  1838     mmc$pq_shared_site_12         = 21,
o  1839     mmc$pq_shared_site_13         = 22,
o  1840     mmc$pq_shared_site_14         = 23,
o  1841     mmc$pq_shared_site_15         = 24,
o  1842     mmc$pq_shared_site_16         = 25,
o  1843     mmc$pq_shared_site_17         = 26,
o  1844     mmc$pq_shared_site_18         = 27,
o  1845     mmc$pq_shared_site_19         = 28,
o  1846     mmc$pq_shared_site_20         = 29,
o  1847     mmc$pq_shared_site_21         = 30,
o  1848     mmc$pq_shared_site_22         = 31,
o  1849     mmc$pq_shared_site_23         = 32,
o  1850     mmc$pq_shared_site_24         = 33,
o  1851     mmc$pq_shared_site_25         = 34,
o  1852
o  1853     mmc$pq_shared_io_error        = 35,
o  1854     mmc$pq_swapped_io_error       = 36,
o  1855
o  1856     mmc$pq_job_fixed              = 37,
o  1857     mmc$pq_job_io_error           = 38,
o  1858     mmc$pq_job_working_set        = 39,
o  1859
o  1860
o  1861     mmc$pq_first_valid_in_pt = mmc$pq_wired,
o  1862     mmc$pq_last_reassignable = mmc$pq_avail,
o  1863     mmc$pq_job_base = mmc$pq_job_fixed,
o  1864     mmc$pq_shared_first      = mmc$pq_shared_task_service,
o  1865     mmc$pq_shared_last_sys   = mmc$pq_shared_other,
o  1866     mmc$pq_shared_first_site= mmc$pq_shared_site_01,
```



```
o  1867     mmc$pq_shared_num_sites = 25,
o  1868                 {Warning, coding assumes that there is at least one site queue}
o  1869     mmc$pq_shared_last_site = mmc$pq_shared_first_site
o  1870       + mmc$pq_shared_num_sites -1,
o  1871     mmc$pq_shared_last       = mmc$pq_shared_site_25;
o  1872
o  1873   TYPE
o  1874     mmt$global_page_queue_index = mmc$pq_free .. mmc$pq_swapped_io_error,
o  1875     mmt$job_page_queue_index = mmc$pq_job_fixed .. mmc$pq_job_working_set,
o  1876     mmt$page_frame_queue_id = 0 .. mmc$pq_job_working_set;
o  1877


o  1879 {------------------------------------  MMT$PAGE_FRAME_TABLE  ----------------------------
o  1880 {Page Frame Table. This table is used to:}
o  1881 {       . manage assignment of free pages}
o  1882 {       . maintain threads for page aging}
o  1883
o  1884   TYPE
o  1885     mmt$page_frame_table_entry = record
o  1886       link:  mmt$link,
o  1887       segment_link: mmt$link,
o  1888       cyclic_age: 0 .. 255,
o  1889       ijl_ordinal: jmt$ijl_ordinal,
o  1890       queue_id: mmt$page_frame_queue_id,
o  1891       active_io_count: 0 .. 0ff(16),
o  1892       locked_page: mmt$locked_page,
o  1893       pti: ost$page_table_index,
o  1894       task_queue: tmt$task_queue_link,
o  1895       age: mmt$page_age,
o  1896       aste_p: ^mmt$active_segment_table_entry,
o  1897       io_error: iot$io_error,
o  1898       sva:  ost$system_virtual_address,
o  1899     recend,
o  1900
o  1901     mmt$page_frame_table = array [ * ] of mmt$page_frame_table_entry;
o  1902
o  1903 {*copyc MMT$LINK
o  1904 {*copyc MMT$LOCKED_PAGE
o  1905 {*copyc MMT$UNUSED_AGE_TABLE_ENTRY
o  1906 {*copyc OST$PAGE_TABLE
o  1907 {*copyc OST$HARDWARE_SUBRANGES
o  1908 {*copyc MMT$ACTIVE_SEGMENT_TABLE
o  1909 {*copyc TMT$TASK_QUEUE_LINK
o  1910 {*copyc MMT$PAGE_FRAME_QUEUE_ID
o  1911 {*copyc jmt$ijl_ordinal
o  1912 {*copyc iot$io_error
o  1913


o  1915 {------------------------------------  MMT$PAGE_QUEUE_LIST  ----------------------------
o  1916
o  1917 {  This deck (mmt$page_queue_list) defines the page queue list entry that
o  1918 {  contains the heads of the chains that run thru the Page Frame Table.}
```

MMT$PAGE_QUEUE_LIST

```
o  1919 {
o  1920 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  1921 {   make the appropriate changes in the corresponding display procedures in the
o  1922 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  1923 {
o  1924
o  1925    TYPE
o  1926      mmt$page_queue_list_entry = record
o  1927        link: mmt$link,
o  1928        count: 0 .. osc$max_page_frames,
o  1929      recend,
o  1930      mmt$global_page_queue_list_ent = record
o  1931        pqle : mmt$page_queue_list_entry,
o  1932        age_interval: 0 .. 255,
o  1933        minimum : 0 .. 0ffffffff(16),  {temporary definition so that SETSA can set this value
o  1934        maximum : 0 .. osc$max_page_frames,
o  1935      recend,
o  1936
o  1937
o  1938 {Define the global and local page queue lists.
o  1939
o  1940      mmt$global_page_queue_list = array [mmt$global_page_queue_index] of mmt$global_page_queue_list_ent,
o  1941      mmt$job_page_queue_list = array [mmt$job_page_queue_index] of mmt$page_queue_list_entry;
o  1942
o  1943 {*copyc OST$HARDWARE_SUBRANGES
o  1944 {*copyc MMT$LINK
o  1945 {*copyc OST$PAGE_TABLE
o  1946 {*copyc mmt$page_frame_queue_id


o  1948 {------------------------------------  MMT$PAGE_Q_COUNTS                ---------------------------
o  1949
o  1950 { If the following type must be expanded, make sure the ARRAY is the last field in the record.
o  1951
o  1952    TYPE
o  1953      mmt$page_q_counts = RECORD
o  1954        long_wait_count: 0 .. 0ffffffff(16),
o  1955        swap_resident_count: 0 .. 0ffffffff(16),
o  1956        site_defined_queues_active: 0..255,
o  1957        q_counts: ARRAY [mmt$page_frame_queue_id] OF 0 .. 0ffffffff(16),
o  1958      RECEND;
o  1960 {*copyc mmt$page_frame_queue_id


o  1963 {------------------------------------  MMT$PF_STATISTICS                ---------------------------
o  1964
o  1965    TYPE
o  1966      mmt$pf_statistics = array [0 .. 18] of integer;


o  1968 {------------------------------------  MMT$RB_ADVISE                    ---------------------------
o  1969
o  1970    TYPE
```


MMT$RB_ADVISE

```
o  1971      mmt$rb_advise = record
o  1972        reqcode: ALIGNED [0 MOD 8] syt$monitor_request_code,
o  1973        status: syt$monitor_status,
o  1974        out_pva: ^cell,
o  1975        out_length: ost$segment_length,
o  1976        in_pva: ^cell,
o  1977        in_length: ost$segment_length,
o  1978      recend;
o  1979
o  1980 {*copyc OSD$VIRTUAL_ADDRESS
o  1981 {*copyc SYT$MONITOR_REQUEST_CODE
o  1982 {*copyc SYC$MONITOR_REQUEST_CODES


o  1984 {------------------------------------  MMT$RB_FREE_FLUSH                ---------------------------
o  1985
o  1986 {Init_new_io is set to true in job mode and cleared in monitor before
o  1987 {a reissue of the monitor request if wait is true.  This is to prevent
o  1988 {mmp$mm_write_modified_pages from initiating new writes.
o  1989
o  1990    TYPE
o  1991      mmt$rb_free_flush = record
o  1992        reqcode: ALIGNED [0 MOD 8] syt$monitor_request_code,
o  1993        status: syt$monitor_status,
o  1994        pva: ^cell,
o  1995        length: ost$segment_length,
o  1996        waitopt: ost$wait,
o  1997        init_new_io: boolean,              {Used only for write_modified_pages processing.}
o  1998      recend;
o  1999
o  2000 {*copyc SYT$MONITOR_REQUEST_CODE
o  2001 {*copyc OST$WAIT
o  2002 {*copyc SYC$MONITOR_REQUEST_CODES
o  2003 {*copyc OSD$VIRTUAL_ADDRESS


o  2005 {------------------------------------  MMT$RB_LOCK_RING_1_STACK         ---------------------------
o  2006
o  2007 { Define the type definition for the monitor request to change the calling job's
o  2008 { ring 1 stack to a transient segment during job termination.
o  2009
o  2010    TYPE
o  2011      mmt$rb_lock_ring_1_stack = record
o  2012        request_code: ALIGNED [0 MOD 8] syt$monitor_request_code,
o  2013        status: syt$monitor_status,
o  2014        disk_file_descriptor_offset: ost$valid_relative_pointer,
o  2015      recend;
o  2016
o  2017 {*copyc osd$virtual_address
o  2018 {*copyc syc$monitor_request_codes
o  2019 {*copyc syt$monitor_request_code
```

```
o  2021 {------------------------------------  MMT$RB_LOCK_UNLOCK_PAGES  ----------------------------
o  2022
o  2023
o  2024 {  Define type definition for lock/unlock pages request block.
o  2025
o  2026   TYPE
o  2027     mmt$rb_lock_unlock_pages = record
o  2028       reqcode: ALIGNED [0 MOD 8] syt$monitor_request_code,
o  2029       status: syt$monitor_status,
o  2030       lock_page_type: mmt$locked_page,
o  2031       pva: ^cell,
o  2032       length: ost$byte_count,
o  2033     recend;
o  2034
o  2035 {*copyc MMT$LOCKED_PAGE
o  2036 {*copyc OST$HARDWARE_SUBRANGES
o  2037 {*copyc SYT$MONITOR_REQUEST_CODE
o  2038 {*copyc SYC$MONITOR_REQUEST_CODES


o  2040 {------------------------------------  MMT$RB_LOCK_UNLOCK_SEGMENT  ----------------------------
o  2041
o  2042 {The following defines the request block for issuing lock/unlock segment
o  2043 {requests to memory manager.
o  2044
o  2045 {Init_new_io is set to true in job mode and cleared in monitor before
o  2046 {a reissue of the monitor request if wait is true.  This is to prevent
o  2047 {mmp$mm_write_modified_pages from initiating new writes.
o  2048
o  2049   TYPE
o  2050     mmt$rb_lock_unlock_segment = RECORD
o  2051       reqcode: ALIGNED [0 MOD 8] syt$monitor_request_code,
o  2052       status: syt$monitor_status,
o  2053       pva: ^cell,
o  2054       wait: ost$wait,
o  2055       init_new_io: boolean,                    {Used only for write_modified_pages processing.}
o  2056       CASE request: (mmc$lus_lock_segment, mmc$lus_unlock_segment) OF
o  2057       = mmc$lus_lock_segment =
o  2058         access: mmt$lus_lock_type,
o  2059       = mmc$lus_unlock_segment =
o  2060         page_disposition: mmt$lus_page_disposition,
o  2061       CASEND,
o  2062     RECEND;
o  2063
o  2064 {*copyc MMT$LUS_DECLARATIONS
o  2065 {*copyc OST$WAIT
o  2066 {*copyc SYC$MONITOR_REQUEST_CODES
o  2067 {*copyc SYT$MONITOR_REQUEST_CODE


o  2069 {------------------------------------  MMT$RB_RING1_SEGMENT_REQUEST  ----------------------------
o  2070
o  2071 {Init_new_io is set to true in job mode and cleared in monitor before
o  2072 {a reissue of the monitor request if wait is true.  This is to prevent
```



```
o  2073 {mmp$mm_write_modified_pages from initiating new writes.
o  2074
o  2075   TYPE
o  2076     mmt$rb_ring1_segment_request = record
o  2077       reqcode: ALIGNED [0 MOD 8] syt$monitor_request_code,
o  2078       status: syt$monitor_status,
o  2079       wait_for_io_complete: boolean, {Used only on FLUSH requests.}
o  2080       io_active: boolean,            {OUTPUT parameter - returned only on FLUSH requests.}
o  2081       init_new_io: boolean,          {Used only for write_modified_pages processing.}
o  2082       case request: (mmc$sr1_delete_seg_segnum,
o  2083                      mmc$sr1_delete_seg_sfid,
o  2084                      mmc$sr1_free_image_pages,
o  2085                      mmc$sr1_commit_memory,
o  2086                      mmc$sr1_detach_file,
o  2087                      mmc$sr1_flush_delete_seg_sfid,
o  2088                      mmc$sr1_flush_seg_segnum,
o  2089                      mmc$sr1_replace_sfid,
o  2090                      mmc$sr1_end_job_recovery,
o  2091                      mmc$sr1_make_mfw_cache,
o  2092                      mmc$sr1_remove_job_shared_pages,
o  2093                      mmc$sr1_change_swap_file_queue,
o  2094                      mmc$sr1_get_highest_offset,
o  2095                      mmc$sr1_delete_job_seg_by_sfid,
o  2096                      mmc$sr1_remove_detached_pages,
o  2097                      mmc$sr1_flush_avail_modified) OF
o  2098
o  2099       = mmc$sr1_delete_seg_sfid, mmc$sr1_detach_file, mmc$sr1_flush_delete_seg_sfid,
o  2100           mmc$sr1_flush_seg_segnum, mmc$sr1_change_swap_file_queue,
o  2101           mmc$sr1_delete_job_seg_by_sfid, mmc$sr1_remove_detached_pages,
o  2102           mmc$sr1_flush_avail_modified =
o  2103         sfid: dmt$system_file_id,
o  2104       = mmc$sr1_delete_seg_segnum =
o  2105         segnum: ost$segment,
o  2106       = mmc$sr1_replace_sfid =
o  2107         old_sfid: dmt$system_file_id,
o  2108         new_sfid: dmt$system_file_id,
o  2109         asti: mmt$ast_index,
o  2110       = mmc$sr1_end_job_recovery =
o  2111         unrecovered_pages: integer,
o  2112         unrecovered_files: integer,
o  2113       = mmc$sr1_make_mfw_cache =
o  2114         ,
o  2115       = mmc$sr1_remove_job_shared_pages =
o  2116         system_file_id: dmt$system_file_id,
o  2117         segment_number: ost$segment,
o  2118         server_file: boolean,
o  2119       = mmc$sr1_get_highest_offset =
o  2120         file_sfid: dmt$system_file_id,
o  2121         highest_offset: amt$file_byte_address,
o  2122       casend,
o  2123     recend;
o  2124
o  2125 {*copyc amt$file_byte_address
o  2126 {*copyc SYT$MONITOR_REQUEST_CODE
o  2127 {*copyc SYC$MONITOR_REQUEST_CODES
o  2128 {*copyc DMT$SYSTEM_FILE_ID
```

```
o  2129 {*copyc mmt$ast_index
o  2130 {*copyc OSD$VIRTUAL_ADDRESS


o  2132 {------------------------------------  MMT$RB_SET_GET_SEGMENT_LENGTH    ----------------------------
o  2133
o  2134 {  Define type definition for monitor request to set or get the current
o  2135 {  segment length.
o  2136 {  NOTE: this request is available in ring 1 ONLY. It assumes the FDE_P is valid.
o  2137 {        no status is returned from this request.
o  2138
o  2139    TYPE
o  2140      mmt$rb_set_get_segment_length = RECORD
o  2141        request_code: syt$monitor_request_code,
o  2142        fde_p: gft$file_desc_entry_p,
o  2143        segment_length {input, output} : ost$segment_length,
o  2144        subfunction_code: mmt$set_get_subfunction_codes,
o  2145      RECEND,
o  2146
o  2147      mmt$set_get_subfunction_codes = (mmc$sf_get_segment_length_fde_p,
o  2148        mmc$sf_set_segment_length_fde_p);
o  2149
o  2150 {*copyc gft$file_desc_entry_p
o  2151 {*copyc gft$system_file_identifier
o  2152 {*copyc syc$monitor_request_codes
o  2153 {*copyc syt$monitor_request_code


o  2155 {------------------------------------  MMT$RMA_LIST                     ----------------------------
o  2156 {Define the RMA LIST used by memory manager.
o  2157
o  2158    CONST
o  2159      mmc$max_rma_list_length = 2048;
o  2160
o  2161    TYPE
o  2162      mmt$rma_list = array [mmt$rma_list_index] of mmt$rma_list_entry,
o  2163
o  2164      mmt$rma_list_index = 1 .. mmc$max_rma_list_length,
o  2165      mmt$rma_list_length = 1 .. mmc$max_rma_list_length,
o  2166
o  2167      mmt$rma_list_entry = record
o  2168        fill: ALIGNED [0 MOD 8] 0 .. 0ffff(16),
o  2169        length: 0 .. 0ffff(16),
o  2170        rma: 0 .. 0ffffffff(16),
o  2171      recend;


o  2173 {------------------------------------  MMT$SEGMENT_ACCESS_RIGHTS        ----------------------------
o  2174 {Define types for specifying ACCESS RIGHTS of a user to a segment.}
o  2175 {  *** note - used only in monitor mode. ***}
o  2176 {
o  2177 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  2178 {   make the appropriate changes in the corresponding display procedures in the
```



```
o  2179 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  2180 {
o  2181
o  2182    TYPE
o  2183      mmt$segment_access_rights = (mmc$sar_none, mmc$sar_read, mmc$sar_modify,
o  2184        mmc$sar_write_extend);


o  2186 {------------------------------------  MMT$SEGMENT_ACCESS_TYPE          ----------------------------
o  2187 {Define ordinal type used in MMP$VERIFY_PVA request.
o  2188
o  2189    TYPE
o  2190      mmt$segment_access_type = (mmc$sat_none, mmc$sat_read, mmc$sat_write, mmc$sat_read_or_write);


o  2192 {------------------------------------  MMT$SEGMENT_ATTRIB_DESCRIPTOR    ----------------------------
o  2193
o  2194    TYPE
o  2195      mmt$segment_attrib_descriptor = record
o  2196        validating_ring_number: ost$valid_ring,
o  2197        file_limits_to_enforce: sft$file_space_limit_kind,
o  2198        pointer_kind: mmt$segment_pointer_kind,
o  2199        sfid: gft$system_file_identifier,
o  2200        user_attributes: ^array [ * ] of mmt$attribute_descriptor,
o  2201      recend;
o  2202
o  2203 {*copyc gft$system_file_identifier
o  2204 {*copyc mmt$attribute_keyword
o  2205 {*copyc osd$virtual_address
o  2206 {*copyc sft$file_space_limit_kind


o  2208 {------------------------------------  MMT$SEGMENT_DESCRIPTOR_TABLE     ----------------------------
o  2209
o  2210 { This common deck contains system constants and type declarations
o  2211 { for tables defined for SEGMENT MANAGEMENT routines.
o  2212 {
o  2213 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  2214 {   make the appropriate changes in the corresponding display procedures in the
o  2215 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  2216 {
o  2217
o  2218
o  2219
o  2220 { Segment Descriptor Table - SDT. This table is the hardware defined
o  2221 { Segment Descriptor Table.}
o  2222
o  2223    TYPE
o  2224      mmt$segment_descriptor = record
o  2225        ste: ost$segment_descriptor,
o  2226        fill1: 0 .. 0ff(16),
o  2227        asti: mmt$ast_index,
o  2228      recend,
```

```
o  2229
o  2230 { For performance, the adaptable size segment table will be used only for allocation.
o  2231 { All other references to the segment table will use the pointer to the fixed size array.
o  2232
o  2233     mmt$segment_descriptor_table = record
o  2234       st: ALIGNED [0 MOD 8192] array [0 .. * ] of mmt$segment_descriptor,
o  2235     recend,
o  2236
o  2237     mmt$max_sdt = record
o  2238       st: ALIGNED [0 MOD 8192] array [0 .. 4095] of mmt$segment_descriptor,
o  2239     recend,
o  2240
o  2241     mmt$max_sdt_p = ^mmt$max_sdt;
o  2242
o  2243 {*copyc OST$SEGMENT_DESCRIPTOR
o  2244 {*copyc MMT$AST_INDEX


o  2246 {------------------------------------  MMT$SEGMENT_DESCRIPTOR_TABLE_EX  ---------------------------
o  2247
o  2248
o  2249 {  This deck defines the SDTX used by Segment Manager.
o  2250 {
o  2251 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  2252 {   make the appropriate changes in the corresponding display procedures in the
o  2253 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  2254 {
o  2255
o  2256
o  2257   TYPE
o  2258     mmt$segment_descriptor_extended = record
o  2259       open_validating_ring_number: ost$ring,
o  2260       access_state: mmt$segment_access_state,
o  2261       sfid: sft$system_file_identifier,
o  2262       inheritance: mmt$segment_inheritance,
o  2263       segment_reservation_state: mmt$segment_reservation_state,
o  2264       software_attribute_set: mmt$software_attribute_set,
o  2265       access_rights: mmt$segment_access_rights,
o  2266       segment_lock: mmt$lock_segment_status,
o  2267       shadow_info: mmt$shadow_info,
o  2268       file_limits_enforced: sft$file_space_limit_kind,
o  2269       stream: mmt$sdtx_stream_data,
o  2270       assign_active: 0 .. osc$max_segment_length + 1,
o  2271     recend,
o  2272
o  2273     mmt$sdtx_stream_data = PACKED RECORD
o  2274       last_page_fault: ost$segment_offset,
o  2275       sequential_accesses: 0..255,
o  2276       transfer_size: 0..15, {size= (2**(transfer_size)) * osv$page_size)
o  2277       random_faults: 0..15,
o  2278       streaming: boolean,
o  2279       transfer_size_specified: boolean,
o  2280       preset_streaming: boolean,
o  2281     RECEND,
o  2282
```



```
o  2283 { For performance, the adaptable size table will be used only for allocation.
o  2284 { All other references to the sdtx will use the fixed size array.
o  2285
o *2286     mmt$segment_descriptor_table_ex = record
o  2287       sdtx_table: array [0 .. * ] of mmt$segment_descriptor_extended,
o  2288     recend,
o  2289
o  2290     mmt$max_sdtx = record
o  2291       sdtx_table: array [0 .. 4095] of mmt$segment_descriptor_extended,
o  2292     recend,
o  2293
o  2294     mmt$max_sdtx_p = ^mmt$max_sdtx;
o  2295
o  2296
o  2297 { Constants for referencing SDTX.ASSIGN_ACTIVE. Note a value < mmc$assign_active_null is a valid
o  2298 { assign for the offset specified by <assign_sctive>.
o  2299
o  2300   CONST
o  2301     mmc$assign_active_null = osc$max_segment_length,
o  2302     mmc$assign_active_escaped = mmc$assign_active_null + 1;
o  2303
o  2304
o  2305 {*copyc gft$system_file_identifier
o  2306 {*copyc mmt$segment_inheritance
o  2307 {*copyc mmt$attribute_keyword
o  2308 {*copyc MMT$LUS_DECLARATIONS
o  2309 {*copyc MMT$SEGMENT_ACCESS_RIGHTS
o  2310 {*copyc MMT$SEGMENT_ACCESS_STATE
o  2311 {*copyc mmt$shadow_info
o  2312 {*copyc MMT$LOCK_SEGMENT_STATUS
o  2313 {*copyc mmt$segment_access_state
o  2314 {*copyc mmt$segment_reservation_state
o  2315 {*copyc MMT$SHADOW_SEGMENT_KIND
o  2316 {*copyc OSD$VIRTUAL_ADDRESS
o  2317 {*copyc SFT$FILE_SPACE_LIMIT_KIND


o  2319 {------------------------------------  MMT$SEGMENT_INHERITANCE  ----------------------------
o  2320
o  2321   TYPE
o  2322
o  2323     mmt$segment_inheritance = (mmc$si_none, mmc$si_share_segment,
o  2324          mmc$si_transfer_segment, mmc$si_new_segment,
o  2325          mmc$si_copy_on_write);


o  2327 {------------------------------------  MMT$SEGMENT_ORIGIN  ----------------------------


o  2329 {------------------------------------  MMT$SHADOW_INFO  ----------------------------
o  2330 {
o  2331 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  2332 {   make the appropriate changes in the corresponding display procedures in the
```

MMT$SHADOW_INFO

```
o  2333 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  2334 {
o  2335
o  2336   TYPE
o  2337     mmt$shadow_info = RECORD
o  2338       shadow_start_page_number: 0 .. 0ffffff(16),
o  2339       shadow_length_page_count: 0 .. 0ffffff(16),
o  2340       shadow_sfid: gft$system_file_identifier,
o  2341       CASE shadow_segment_kind: mmt$shadow_segment_kind OF
o  2342       = mmc$ssk_segment_number =
o  2343         shadow_segment_number: ost$segment,
o  2344       = mmc$ssk_none =
o  2345         passive_for_shadow_by_segnum: boolean,
o  2346       CASEND,
o  2347     RECEND;
o  2348
o  2349
o  2350 {*copyc gft$system_file_identifier
o  2351 {*copyc osd$virtual_address


o  2353 {------------------------------------  MMT$UNUSED_AGE_TABLE_ENTRY       ----------------------------
o  2354 {Define the UNUSED AGE TABLE - contains constants for working}
o  2355 {set calculations.}
o  2356
o  2357   TYPE
o  2358     mmt$page_age = 0 .. 0ffff(16),
o  2359     mmt$unused_age_table_entry = record
o  2360       aii: 0 .. 0ffffffff(16),
o  2361       aif,
o  2362       aic: mmt$page_age,
o  2363     recend;


o  2365 {------------------------------------  MMT$UPDATE_EOI_REASON            ----------------------------
o  2366
o  2367 { The following constants are used in the call to MMP$UPDATE_EOI to indicate
o  2368 { the reason for the EOI update.
o  2369
o  2370   TYPE
o  2371     mmt$update_eoi_reason = (mmc$uer_set_exact_eoi, mmc$uer_page_assigned,
o  2372         mmc$uer_multiple_pages_assigned, mmc$uer_page_written);


o  2374 {------------------------------------  MMT$VA_ACCESS_MODE               ----------------------------
o  2375 {Definitions for interface to segment manager requests.
o  2376
o  2377   TYPE
o  2378     mmt$va_access_mode = (mmc$va_read, mmc$va_write, mmc$va_read_write,
o  2379           mmc$va_execute, mmc$va_pointer_to_procedure, mmc$va_read_execute,
o  2380           mmc$va_binding);
o  2381
```


MMT$WRITE_MODIFIED_PAGES_STATUS

```
o  2383 {------------------------------------  MMT$WRITE_MODIFIED_PAGES_STATUS  ----------------------------
o  2384 {Error codes from mmp$mm_write_modified_pages.
o  2385
o  2386   TYPE
o  2387     mmt$write_modified_pages_status = (mmc$wmp_io_initiation_reject, mmc$wmp_io_complete, mmc$wmp_io_active,
o  2388           mmc$wmp_volume_unavailable, mmc$wmp_io_errors, mmc$wmp_server_terminated);


o  2390 {------------------------------------  MMT$WRITE_PAGE_TO_DISK_STATUS    ----------------------------
o  2391   TYPE
o  2392     mmt$write_page_to_disk_status = (ws_ok, ws_physical_io_reject, ws_no_file_assigned, ws_disk_flaws,
o  2393     ws_device_manager_reject, ws_volume_unavailable, ws_server_terminated);
o  2394


o  2396 {------------------------------------  MMT$XCB_PAGE_WAIT_INFO           ----------------------------
o  2397
o  2398 {This deck defines the information kept in the XCB for a task that is in
o  2399 {'page_wait' status.
o  2400
o  2401   TYPE
o  2402     mmt$xcb_page_wait_info = RECORD
o  2403       pva: ^cell,
o  2404     recend;
o  2405
o  2406


o  2408 {------------------------------------  MTT$SMU_COMMUNICATIONS_BLOCK     ----------------------------
o  2409 { This deck defines the format of the SMU Communications Block (SCB). The SCB is
o  2410 { used by the the 180 CPU Monitor to maintain information about the status of the
o  2411 { 180 OS and hardware.
o  2412
o  2413   TYPE
o  2414     mtt$smu_communications_block = record
o  2415       hardware_status: ALIGNED [0 MOD 8] mtt$scb_hardware_status,
o  2416       kill_180: ALIGNED [0 MOD 8] boolean,
o  2417       processors_logically_on: ost$processor_id_set,
o  2418       processors_with_state_changed: ost$processor_id_set,
o  2419       vector_simulation_control: ost$vector_simulation_control,
o  2420       nos_180_status: ALIGNED [0 MOD 8] mtt$scb_180_status,
o  2421       nos_service_flag: ALIGNED [0 MOD 8] integer,
o  2422       critical_message_time_stamp: ALIGNED [0 MOD 8] integer,
o  2423       hardware_status_messages: mtt$scb_hardware_status_msgs,
o  2424     RECEND,
o  2425
o  2426 { The following field is maintained by the CPU and contains the dynamic status
o  2427 { of the 180 Operating system.
o  2428
o  2429     mtt$scb_180_status = record
o  2430       system_status: mtt$system_status_block,
o  2431       idle_code: syt$180_idle_code,
```

```
0  2432        fill_1: 0..0ffff(16),
0  2433        cause_of_idle: syt$180_idle_code,
0  2434      recend,
0  2435
0  2436  { The following record contains the requested and actual status' of the
0  2437  { system for IDLE and STEP.
0  2438
0  2439      mtt$system_status_block = RECORD
0  2440        idle_status_block: mtt$idle_status_block,
0  2441        step_status_block: mtt$step_status_block,
0  2442      RECEND,
0  2443
0  2444  { Describe the layout of the IDLE and STEP status blocks. When the REQUESTED
0  2445  { and ACTUAL fields are not the same, the system will be driven to the state
0  2446  { found in the REQUESTED fields.
0  2447
0  2448      mtt$step_status_block = RECORD
0  2449        requested_status,
0  2450        actual_status: mtt$system_step_update_request,
0  2451      RECEND,
0  2452
0  2453      mtt$idle_status_block = RECORD
0  2454        requested_status,
0  2455        actual_status: mtt$system_idle_update_request,
0  2456      RECEND;
0  2457
0  2458  {*copyc mtt$scb_hardware_status
0  2459  {*copyc mtt$system_update_requests
0  2460  {*copyc ost$processor_id_set
0  2461  {*copyc ost$vector_simulation_control
0  2462  {*copyc syt$180_idle_code


0  2464  {------------------------------------  OST$CPU_STATE_TABLE  ----------------------------
0  2465
0  2466  {          WARNING! WARNING! WARNING! WARNING! WARNING! WARNING! WARNING!
0  2467  { If this type is changed the type MTA$CPU_STATE_TABLE must reflect a
0  2468  { corresponding change!
0  2469  {
0  2470  { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
0  2471  {   make the appropriate changes in the corresponding display procedures in the
0  2472  {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
0  2473  {
0  2474
0  2475    TYPE
0  2476      ost$state_tables = array [0 .. osc$max_number_of_processors - 1] of ost$cpu_state_table;
0  2477
0  2478    TYPE
0  2479      ost$cpu_state_table = record
0  2480        fill: ALIGNED [0 MOD 8] 0 .. 0ff(16),
0  2481        dispatching_priority: jmt$dispatching_priority,
0  2482        dual_state_prior_subpriority: tmt$dual_state_priority_entry,
0  2483        memory_port_mask: ost$cpu_memory_port_mask,
0  2484        cst_index: ost$logical_processor_id,
0  2485        processor_state: cmt$element_state,
```



```
0  2486        next_processor_state: cmt$element_state,
0  2487        cpu_alive_flag: ALIGNED [0 MOD 8] integer,
0  2488        taskid: ALIGNED [0 MOD 8] ost$global_task_id,
0  2489        ajlo: jmt$ajl_ordinal,
0  2490        dual_state_jps: 0 .. 0ffffffff(16),
0  2491        jcb_p: ALIGNED [0 MOD 8] ^jmt$job_control_block,
0  2492        cpu_state: ost$cpu_state,
0  2493        xcb_p: ALIGNED [0 MOD 8] ^ost$execution_control_block,
0  2494        xcb_rma: ALIGNED [0 MOD 8] integer,
0  2495        dispatch_control: ALIGNED [0 MOD 8] tmt$dispatch_control,
0  2496        max_cptime: ALIGNED [0 MOD 8] integer,
0  2497        accumulated_job_cptime: ALIGNED [0 MOD 8] integer,
0  2498        accumulated_monitor_cptime: ALIGNED [0 MOD 8] integer,
0  2499        ext_int_request: ALIGNED [0 MOD 8] ost$external_interrupt_request,
0  2500        idle_code: syt$180_idle_code,
0  2501        cst_index_x_8: 0 .. 255,
0  2502        time_last_cache_purge: ALIGNED [0 MOD 8] integer,
0  2503        time_last_map_request: ALIGNED [0 MOD 8] integer,
0  2504        monitor_mps: ALIGNED [0 MOD 8] ost$real_memory_address,
0  2505        aborted_task_count: ost$parcel,
0  2506        due_count: ost$parcel,
0  2507        element_id: ALIGNED [0 MOD 8] ost$cpu_element_id,
0  2508        ijl_ordinal: ALIGNED [0 MOD 8] jmt$ijl_ordinal,
0  2509        ijle_p: ^jmt$initiated_job_list_entry,
0  2510        cpu_idle_statistics: ALIGNED [0 MOD 8] ost$cpu_idle_statistics,
0  2511        trace_control: ALIGNED [0 MOD 8] ost$cst_trace_control,
0  2512        termination_message: ALIGNED [0 MOD 8] string (80),
0  2513        reason_for_current_state: ALIGNED [0 MOD 8] ost$cpu_state_reason,
0  2514        pre_processed_for_reconfig: ost$pre_processed_for_reconfig,
0  2515        cpu_should_spin_indefinitely: boolean,
0  2516        previous_processor_state: cmt$element_state,
0  2517        log_cpu_state_change: boolean,
0  2518        next_ptlo_to_dispatch: ost$task_index,
0  2519        fill_ff: 0 .. 0ff(16),
0  2520        dispatching_priority_integer: ALIGNED [0 MOD 8] integer,
0  2521        dummy_4: ALIGNED [0 MOD 8] integer,
0  2522      recend;
0  2523
0  2524  {*copyc cmt$element_state
0  2525  {*copyc jmt$ajl_ordinal
0  2526  {*copyc jmt$ijl_ordinal
0  2527  {*copyc jmt$initiated_job_list_entry
0  2528  {*copyc jmt$job_control_block
0  2529  {*copyc ost$cpu_definitions
0  2530  {*copyc ost$cpu_idle_statistics
0  2531  {*copyc ost$cpu_state
0  2532  {*copyc ost$cpu_state_reason
0  2533  {*copyc ost$execution_control_block
0  2534  {*copyc ost$external_interrupt_request
0  2535  {*copyc ost$global_task_id
0  2536  {*copyc ost$hardware_subranges
0  2537  {*copyc ost$pre_processed_for_reconfig
0  2538  {*copyc ost$segment_descriptor
0  2539  {*copyc ost$cst_trace_control
0  2540  {*copyc syt$180_idle_code
0  2541  {*copyc tmt$dispatch_control
```

```
o  2542 {*copyc tmt$dual_state_dispatch_prior


o  2544 {----------------------------------  OST$CP_TIME                          ----------------------------
o  2545
o  2546 {Define cp time statistics record.
o  2547 {
o  2548 {  NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  2549 {    make the appropriate changes in the corresponding display procedures in the
o  2550 {    module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  2551 {
o  2552
o  2553     TYPE
o  2554       ost$cp_time_value = 0 .. 0ffffffffff(16),
o  2555
o  2556       ost$cp_time = record
o  2557         time_spent_in_job_mode: ost$cp_time_value,
o  2558         time_spent_in_mtr_mode: ost$cp_time_value,
o  2559       recend;


o  2561 {----------------------------------  OST$EXCHANGE_PACKAGE                 ----------------------------
o  2562
o  2563
o  2564
o  2565 {CYBER 180 processor exchange package.
o  2566
o  2567     TYPE
o  2568       ost$exchange_package = packed record
o  2569         p_register: ost$p_register,
o  2570         undefined1: 0 .. 0f(16),
o  2571         vmid: ost$virtual_machine_identifier,
o  2572         undefined2: 0 .. 0f(16),
o  2573         uvmid: ost$virtual_machine_identifier,
o  2574         a0_dynamic_space_pointer: ^cell,
o  2575         flags: ost$flags,
o  2576         undefined3: 0 .. 03ff(16),
o  2577         trap_enable: ost$trap_enable,
o  2578         a1_current_stack_frame: ^cell,
o  2579         user_mask: ALIGNED ost$user_conditions,
o  2580         a2_previous_save_area: ^ost$minimum_save_area,
o  2581         monitor_mask: ALIGNED ost$monitor_conditions,
o  2582         a3: ^cell,
o  2583         user_condition_register: ost$user_conditions,
o  2584         a4: ^cell,
o  2585         monitor_condition_register: ost$monitor_conditions,
o  2586         a5: ^cell,
o  2587         undefined4: 0 .. 0f(16),
o  2588         keypoint_class_number: ost$keypoint_class,
o  2589         last_processor_id: 0 .. 0ff(16),
o  2590         a6: ^cell,
o  2591         keypoint_mask: ALIGNED ost$keypoint_mask,
o  2592         a7: ^cell,
o  2593         keypoint_code_1: 0 .. 0ffff(16),
```



```
o  2594         a8: ^cell,
o  2595         keypoint_code_2: 0 .. 0ffff(16),
o  2596         a9: ^cell,
o  2597         process_interval_timer_1: 0 .. 0ffff(16),
o  2598         aa: ^cell,
o  2599         process_interval_timer_2: 0 .. 0ffff(16),
o  2600         ab: ^cell,
o  2601         base_constant_1: 0 .. 0ffff(16),
o  2602         ac: ^cell,
o  2603         base_constant_2: 0 .. 0ffff(16),
o  2604         ad: ^cell,
o  2605         model_dependent_flags: 0 .. 0ffff(16),
o  2606         ae: ^cell,
o  2607         undefined5: 0 .. 0f(16),
o  2608         segment_table_length: ost$segment,
o  2609         af: ^cell,
o  2610         x_registers: array [ost$register_number] of ost$x_register,
o  2611         model_dependent_word: integer {ost$word} ,
o  2612         segment_table_address_1: 0 .. 0ffff(16),
o  2613         untranslatable_pointer: ost$pva,
o  2614         segment_table_address_2: 0 .. 0ffff(16),
o  2615         trap_pointer: ^cell,
o  2616         debug_index: 0 .. 63,
o  2617         undefined6: 0 .. 7,
o  2618         debug_mask_register: ost$debug_mask,
o  2619         debug_list_pointer: ^ost$debug_list,
o  2620         tos_registers: array [ost$valid_ring] of ost$top_of_stack_pointer,
o  2621       recend;
o  2622
o  2623     TYPE
o  2624       ost$flags = set of (osc$critical_frame, osc$on_condition,
o  2625             osc$keypoint_enable, osc$process_not_damaged);
o  2626
o  2627     TYPE
o  2628       ost$top_of_stack_pointer = packed record
o  2629         undefined: 0 .. 0fff(16),
o  2630         largest_ring_number: ost$ring, {only present in ring 1 TOS}
o  2631         pva: ost$pva,
o  2632       recend;
o  2633
o  2634 {*copyc osd$virtual_address
o  2635 {*copyc osd$registers
o  2636 {*copyc ost$virtual_machine_identifier
o  2637 {*copyc ost$keypoint_class
o  2638 {*copyc osd$conditions
o  2639 {*copyc ost$trap_enable
o  2640 {*copyc ost$stack_frame_save_area
o  2641 {*copyc ost$debug_list
o  2642 {*copyc ost$debug_mask
o  2643 {*copyc ost$debug_code


o  2645 {----------------------------------  OST$EXECUTION_CONTROL_BLOCK          ----------------------------
o  2646 {Declaration for the EXECUTION CONTROL BLOCK
o  2647 {      *** If you add fields to this TYPE, make sure the constant in      ***
```

```
2648 {     *** in SYA$CONSTANTS for XCBSIZE is >= actual size of this record  ***
2649
2650
2651
2652   TYPE
2653     ost$execution_control_block = record
2654       xp: ALIGNED [0 MOD 416] ost$exchange_package,
2655       monitor_flags: syt$monitor_flags,
2656       processor_selections: ost$processor_id_set,
2657       last_lpid_for_task: ost$processor_id,
2658 { End of fields referenced in assembly language
2659       system_task_id: tmt$system_task_id,
2660       critical_task: boolean,
2661       task_has_terminated: boolean,
2662       stlc_allocation: boolean,
2663       special_trap_count: 0 .. 0ff(16),
2664       global_task_id: ost$global_task_id,
2665       parent_global_task_id: ost$global_task_id,
2666       wait_inhibited: boolean,
2667       system_table_lock_count: ALIGNED [0 MOD 8] ost$cs_lock,
2668       system_flags: ALIGNED [0 MOD 8] tmt$system_flags,
2669       received_message_list: ALIGNED [0 MOD 8] nat$received_message_list,
2670       task_is_terminating: boolean,
2671       task_has_been_rethreaded: boolean,
2672       system_give_up_cpu: boolean,
2673       subsystem_give_up_cpu: boolean,
2674       subsystem_lock_priority_count: 0 .. 0ff(16),
2675       dispatching_priority: jmt$dispatching_priority,
2676       dispatching_priority_bias_id: (jmc$dpb_positive, jmc$dpb_negative, jmc$dpb_absolute),
2677       dispatching_priority_bias: jmt$dispatching_priority,
2678       system_error_count: 0 .. 0ff(16),
2679       link: ^ost$execution_control_block,
2680       task_control_block: ^cell,
2681       task_id: pmt$task_id,
2682       stack_pages_saved: PACKED ARRAY [0 .. 15] of boolean,
2683       sdt_offset: ost$valid_relative_pointer,
2684       sdtx_offset: ost$valid_relative_pointer,
2685       pit_count: ost$free_running_clock,
2686       cp_time: ost$cp_time,
2687       page_wait_info: mmt$xcb_page_wait_info,
2688       timeslice: jmt$time_slice_values,
2689       relative_task_priority: 0 .. 255,
2690       ring1_termination_reason: ost$ring1_termination_reason,
2691       maxws_aio_slowdown: 0..0ffffff(16),
2692       monitor_faults: tmt$monitor_fault_buffer,
2693       signals: tmt$signal_buffer,
2694       paging_statistics: ost$paging_statistics,
2695       save9: string (31), { * * * currently used for task name * * * }
2696       iocb_p: ^cell,
2697       keypoint_enable: boolean,
2698       keypoint_register_enable: boolean,
2699       time_last_due: ost$cp_time_value,
2700       proc_malf_count: 0 .. 255,
2701       shadow_reference_info: mmt$shadow_reference_info,
2702       assign_active_sfid: gft$system_file_identifier
2703     recend;
```



```
2704
2705 {*copyc gft$system_file_identifier
2706 {*copyc jmt$dispatching_priority
2707 {*copyc jmt$service_class_attributes
2708 {*copyc mmt$segment_descriptor_table
2709 {*copyc mmt$segment_descriptor_table_ex
2710 {*copyc mmt$shadow_reference_info
2711 {*copyc mmt$xcb_page_wait_info
2712 {*copyc nat$received_message_list
2713 {*copyc osd$conditions
2714 {*copyc osd$virtual_address
2715 {*copyc ost$cp_time
2716 {*copyc ost$exchange_package
2717 {*copyc ost$hardware_subranges
2718 {*copyc ost$name
2719 {*copyc ost$paging_statistics
2720 {*copyc ost$processor_id_set
2721 {*copyc ost$quantum
2722 {*copyc ost$ring1_termination_reason
2723 {*copyc ost$signature_lock
2724 {*copyc ost$task_id
2725 {*copyc pmt$task_id
2726 {*copyc syt$monitor_flags
2727 {*copyc tmt$monitor_fault_buffer
2728 {*copyc tmt$signal_buffer
2729 {*copyc tmt$system_flags
2730 {*copyc tmt$system_task_id


2732 {-------------------------------------  OST$GLOBAL_TASK_ID  ----------------------------
2733
2734   TYPE
2735     ost$global_task_id = record
2736       index: ost$task_index,
2737       seqno: 0 .. 255,
2738     recend;
2739
2740   TYPE
2741     ost$task_index = 0 .. osc$max_tasks;
2742
2743   CONST
2744     osc$max_tasks = 4095;


2746 {-------------------------------------  OST$HARDWARE_SUBRANGES  ----------------------------
2747 {Hardware defined subranges and container sizes.}
2748
2749   TYPE
2750     ost$real_memory_address = 0 .. 0ffffffff(16),
2751     ost$real_memory_word_address = 0 .. 0fffffff(16),
2752     ost$byte_count = 0 .. 7fffffff(16);
2753
2754 {Define SVA.}
2755
```

```
o  2756    TYPE
o  2757      ost$system_virtual_address = packed record
o  2758        asid: ost$asid,
o  2759        offset: ost$segment_offset,
o  2760      recend,
o  2761
o  2762      ost$asid = 0 .. 0ffff(16);
o  2763
o  2764  {*copyc OSD$VIRTUAL_ADDRESS
o  2765  {*copyc ost$byte
o  2766  {*copyc ost$free_running_clock
o  2767  {*copyc ost$halfword
o  2768  {*copyc ost$parcel
o  2769  {*copyc ost$word


o  2771  {----------------------------------  OST$HEAP                          ----------------------------
o  2772
o  2773  {Define segment numbers for system heaps and reserved segments.
o  2774
o  2775    CONST
o  2776      osc$segnum_page_table = 0,
o  2777      osc$segnum_mainframe_wired = 1,
o  2778      osc$segnum_mainframe_wired_cb = 12(16),
o  2779      osc$segnum_mainframe_paged = 2,
o  2780      osc$segnum_job_fixed_heap = 3,
o  2781      osc$segnum_job_pageable_heap = 4,
o  2782      osc$segnum_task_private_heap = 5,
o  2783      osc$segnum_task_shared_heap = 6,
o  2784      osc$segnum_task_private_ring_11 = 7,
o  2785      osc$segnum_system_dayfile = 8,
o  2786      osc$segnum_job_dayfile = 9,
o  2787
o  2788  { The following constant defines the segment number of the first global log
o  2789  {  FOLLOWING the system dayfile.  The segment numbers are sequential starting
o  2790  {  there
o  2791
o  2792      osc$segnum_first_global_log = 20(16);
o  2793
o  2794  {Define Type declaration for the OS heaps.
o  2795
o  2796    TYPE
o  2797      ost$heap = HEAP (REP 3ffffff(16) of cell);
o  2798


o  2800  {----------------------------------  OST$MONITOR_FAULT                  ----------------------------
o  2801
o  2802  { Declarations for a MONITOR FAULT
o  2803  {
o  2804  { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  2805  {   make the appropriate changes in the corresponding display procedures in the
o  2806  {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  2807  {
```



```
o  2808
o  2809    TYPE
o  2810      ost$monitor_fault = record
o  2811        pva: ost$pva,
o  2812        a0,
o  2813        a1: ^cell,
o  2814        a2: ^ost$minimum_save_area,
o  2815        CASE identifier: tmt$monitor_fault_identifiers  OF
o  2816        = tmc$broken_task_fault_id =
o  2817          broken_task_fault: tmt$broken_task_monitor_fault,
o  2818        = tmc$mcr_fault =
o  2819          mcr_fault: tmt$mcr_faults,
o  2820        = mmc$segment_fault_processor_id =
o  2821          segment_access_fault: mmt$segment_access_condition,
o  2822        = tmc$dummy_fault =
o  2823          contents: ost$monitor_fault_contents,
o  2824        CASEND,
o  2825      recend,
o  2826
o  2827      ost$monitor_fault_contents = array [1 .. osc$max_fault_contents] of 0 .. 0ff(16);
o  2828
o  2829    CONST
o  2830      osc$max_fault_id = 63;
o  2831
o  2832    CONST
o  2833      osc$max_fault_contents = 24;
o  2834
o  2835  {*copyc ost$stack_frame_save_area
o  2836  {*copyc osd$registers
o  2837  {*copyc tmt$monitor_fault_buffer
o  2838


o  2840  {----------------------------------  OST$PAGING_STATISTICS              ----------------------------
o  2841
o  2842  { OSDPFST - Type declarations for paging statistics record. }
o  2843
o  2844  { ****************************************************************************
o  2845  { NOTE:  This type is used by the ASSEMBLER deck PMM$INTERCEPT_PROCEDURES.  If
o  2846  { this type is changed PMM$INTERCEPT_PROCEDURES may need to be upgraded too or
o  2847  { ANALYZE_PROGRAM_DYNAMICS will break.
o  2848  { ****************************************************************************
o  2849  {
o  2850  { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  2851  {   make the appropriate changes in the corresponding display procedures in the
o  2852  {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  2853  {
o  2854
o  2855    TYPE
o  2856      ost$paging_statistics = record
o  2857        page_in_count: 0 .. 0ffffffff(16),
o  2858        pages_reclaimed_from_queue: 0 .. 0ffffffff(16),
o  2859        new_pages_assigned: 0 .. 0ffffffff(16),
o  2860        pages_from_server:  0 .. 0ffffffff(16),
o  2861        page_fault_count:   0 .. 0fffffffffff(16),
```

```
o  2862       working_set_max_used: 0 .. 0ffff(16),
o  2863     recend;


o  2865 {------------------------------------  OST$SEGMENT_ACCESS_CONTROL        ----------------------------
o  2866
o  2867   TYPE
o  2868     ost$segment_access_control = record
o  2869       cache_bypass: boolean,
o  2870       execute_privilege: ost$execute_privilege,
o  2871       read_privilege: ost$read_privilege,
o  2872       write_privilege: ost$write_privilege,
o  2873     recend,
o  2874
o  2875     ost$execute_privilege = (osc$non_executable, osc$non_privileged,
o  2876       osc$local_privilege, osc$global_privilege),
o  2877
o  2878     ost$read_privilege = (osc$non_readable, osc$read_key_lock_controlled,
o  2879       osc$read_uncontrolled, osc$binding_segment),
o  2880
o  2881     ost$write_privilege = (osc$non_writable, osc$write_key_lock_controlled,
o  2882       osc$write_uncontrolled, osc$wp_reserved);


o  2884 {------------------------------------  OST$SEGMENT_DESCRIPTOR           ----------------------------
o  2885 {Segment descriptor word : describes a single segment.}
o  2886 {
o  2887 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  2888 {   make the appropriate changes in the corresponding display procedures in the
o  2889 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  2890 {
o  2891
o  2892   TYPE
o  2893     ost$segment_descriptor = packed record
o  2894       v1: (osc$v1_invalid_entry, osc$v1_reserved, osc$v1_regular_segment, osc$v1_cache_bypass),
o  2895       xp: ost$execute_privilege,
o  2896       rp: ost$read_privilege,
o  2897       wp: ost$write_privilege,
o  2898       r1: ost$ring,
o  2899       r2: ost$ring,
o  2900       asid: ost$asid,
o  2901       key_lock: ost$key_lock,
o  2902     recend;
o  2903
o  2904 {*copyc OSD$VIRTUAL_ADDRESS
o  2905 {*copyc OST$HARDWARE_SUBRANGES
o  2906 {*copyc OST$SEGMENT_ACCESS_CONTROL


o  2908 {------------------------------------  OST$SIGNATURE_LOCK_STATUS        ----------------------------
o  2909 {  Define status values for compare-swap operations.
o  2910 {  NOTE: this deck use to define types for OST$SIGNATURE_LOCK. These
o  2911 {  definitions were removed from this deck and put in a non
```



```
o  2912 {  program-interface deck OST$SIGNATURE_LOCK. The declarations have been
o  2913 { changed
o  2914 {  for NOSVE internal use and were removed from this deck to minimize the
o  2915 { chance
o  2916 {  of unexpected breakages. The original declarations are commented out and
o  2917 { given
o  2918 {  below. Products that need these definitions should copy the following
o  2919 { declarations
o  2920 {  into their own decks.
o  2921 {
o  2922 {             ost$compare_swap_lock = integer,
o  2923 {             ost$cs_lock = integer, {* * HCS compatibility * * *}
o  2924 {             ost$signature_lock = record
o  2925 {               lock_id: ALIGNED [0 MOD 8] ost$cs_lock,
o  2926 {               lock_count: integer,
o  2927 {               reject_count: integer,
o  2928 {             recend,
o  2929 {
o  2930
o  2931
o  2932   CONST
o  2933     osc$cs_successful = 0,
o  2934     osc$cs_failed = 1,
o  2935     osc$cs_variable_locked = 2;
o  2936
o  2937   TYPE
o  2938     ost$signature_lock_status = (osc$sls_not_locked,
o  2939           osc$sls_locked_by_another_task, osc$sls_locked_by_current_task);
o  2940


o  2942 {------------------------------------  SYC$MONITOR_REQUEST_CODES        ----------------------------
o  2943
o  2944 {  This common deck defines monitor request codes.
o  2945 {
o  2946 {  NOTE: Unused request codes have 'unimplemented' as part of the name, when
o  2947 {  defining a new code these codes should be used first.
o  2948
o  2949   TYPE
o  2950     syt$monitor_request_code = 0 .. 255;
o  2951
o  2952
o  2953   CONST
o  2954
o  2955
o  2956 {  NOTE: Any changes to this common deck will require a change to
o  2957 {        the static array 'request_id' in procedure 'format_system_mr_data'
o  2958 {        in module 'clm$display_system_data' in deck CLMDSS.
o  2959 {        The dump procedure in deck dum$create_monitor_func_file  should
o  2960 {        also be changed.  If syc$rc_maximum_value is changed then deck
o  2961 {         dum$display_active_tasks should be changed to reflect this value.
o  2962
o  2963     syc$rc_maximum_value = 85;  {*** See note above ****
o  2964
o  2965
```

```
o  2966    CONST
o  2967      syc$rc_cycle = 1,
o  2968      syc$rc_delay = 2,
o  2969      syc$rc_unused_request_3 = 3,
o  2970      syc$rc_device_io = 4,
o  2971      syc$rc_advise_in = 5,
o  2972      syc$rc_advise_out = 6,
o  2973      syc$rc_advise_out_in = 7,
o  2974      syc$rc_initiate_task = 8,
o  2975      syc$rc_page_fault = 9,                {Monitor use only}
o  2976      syc$rc_initiate_job = 10,
o  2977      syc$rc_exit_job = 11,
o  2978      syc$rc_free_pages = 12,
o  2979      syc$rc_write_modified_pages = 13,
o  2980      syc$rc_change_segment_table = 14,
o  2981      syc$rc_unused_request_15 = 15,
o  2982      syc$rc_unused_request_16 = 16,
o  2983      syc$rc_unused_request_17 = 17,
o  2984      syc$rc_job_swapping_functions = 18,
o  2985      syc$rc_idle_system = 19,
o  2986      syc$rc_mcr_ucr_fault = 20,            {Monitor use only}
o  2987      syc$rc_system_error = 21,
o  2988      syc$rc_fetch_task_statistics = 22,
o  2989      syc$rc_unused_request_23 = 23,
o  2990      syc$rc_unused_request_24 = 24,
o  2991      syc$rc_ready_task = 25,
o  2992      syc$rc_set_system_flag = 26,
o  2993      syc$rc_wait = 27,
o  2994      syc$rc_lock_ring_1_stack = 28,
o  2995      syc$rc_mtr_send_signal = 29,
o  2996      syc$rc_set_get_segment_length = 30,
o  2997      syc$rc_memory_manager_io = 31,
o  2998      syc$rc_job_recovery_requests = 32,
o  2999      syc$rc_ring1_segment_request = 33,
o  3000      syc$rc_task_exit = 34,
o  3001      syc$rc_unused_request_35 = 35,
o  3002      syc$rc_update_job_task_enviro = 36,
o  3003      syc$rc_segment_request = 37,
o  3004      syc$rc_lock_pages = 38,
o  3005      syc$rc_unlock_pages = 39,
o  3006      syc$rc_fetch_pva_unwritten_pgs = 40,
o  3007      syc$rc_allocate_front_end = 41,
o  3008      syc$rc_deallocate_front_end = 42,
o  3009      syc$rc_apply_mat_changes = 43,
o  3010      syc$rc_tape_io = 44,
o  3011      syc$rc_translate_byte_address = 45,
o  3012      syc$rc_config_mgmt_request = 46,
o  3013      syc$rc_manage_system_tasks = 47,
o  3014      syc$rc_lock_unlock_segment = 48,
o  3015      syc$rc_issue_dft_request = 49,
o  3016      syc$rc_wait_io_completion = 50,
o  3017      syc$rc_switch_task = 51,              {Monitor use only}
o  3018      syc$rc_process_short_warning = 52,    {Monitor use only}
o  3019      syc$rc_monitor_system_status = 53,    {Monitor use only}
o  3020      syc$rc_process_io_completions = 54,   {Monitor use only}
o  3021      syc$rc_update_system_display = 55,
```



```
o  3022      syc$rc_process_scd_block = 56,        {Monitor use only}
o  3023      syc$rc_keypoint = 57,
o  3024      syc$rc_periodic_call = 58,            {Monitor use only}
o  3025      syc$rc_process_due = 59,              {Monitor use only}
o  3026      syc$rc_unused_request_60 = 60,
o  3027      syc$rc_swap_job = 61,                 {Monitor use only}
o  3028      syc$rc_monitor_mode_ei = 62,          {Monitor use only}
o  3029      syc$rc_unused_request_63 = 63,
o  3030      syc$rc_subsystem_request = 64,
o  3031      syc$rc_logging_request = 65,
o  3032      syc$rc_process_dft_block = 66,        {Monitor use only}
o  3033      syc$rc_job_scheduler_request = 67,
o  3034      syc$rc_fetch_offset_mod_pages = 68,
o  3035      syc$rc_assign_pages = 69,
o  3036      syc$rc_conditional_free = 70,
o  3037      syc$rc_queue_rhfam_request = 71,
o  3038      syc$rc_namve_io = 72,
o  3039      syc$rc_file_server_request = 73,
o  3040      syc$rc_move_pages = 74,
o  3041      syc$rc_assign_contig_memory = 75,
o  3042      syc$rc_reallocate_front_end = 76,
o  3043      syc$rc_ring1_server_seg_request = 77,
o  3044      syc$rc_monitor_cpu_self_state = 78,   {Monitor use only}
o  3045      syc$rc_stats_facility_request = 79,
o  3046      syc$rc_system_deadstart_status = 80,
o  3047      syc$rc_service_class_statistics = 81,
o  3048      syc$rc_unused_request_82 = 82,
o  3049      syc$rc_unused_request_83 = 83,
o  3050      syc$rc_unused_request_84 = 84,
o  3051      syc$inject_hardware_fault = 85;
o  3052


o  3054 {----------------------------------  SYT$MONITOR_FLAG  ----------------------------
o  3055 { This deck defines system core flag numbers.
o  3056 { Please note that the order of these flags is very important.
o  3057 { The flags are processed from lowest to highest number.
o  3058 { Any changes made to this deck, must also be reflected in
o  3059 { the assembly language version- SYA$MONITOR_FLAG_HANDLERS.
o  3060 {
o  3061 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  3062 {   make the appropriate changes in the corresponding display procedures in the
o  3063 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  3064 {
o  3065
o  3066
o  3067    TYPE
o  3068      syt$monitor_flag = (tmc$mf_cause_job_free_flag_trap, syc$mf_hang_task,
o  3069         syc$mf_cause_job_recovery, mmc$mf_volume_unavailable, syc$mf_invoke_sysdebug,
o  3070         syc$mf_system_debugger, syc$mf_dump_job_environment, mmc$mf_segment_mgr_flag,
o  3071         syc$mf_cpu_configuration_change, mmc$mf_shadow_file_reference,
o  3072         syc$mf_for_keypoint_traceback, syc$mf_spare_11, syc$mf_spare_12,
o  3073         syc$mf_spare_13, syc$mf_spare_14, syc$mf_spare_15);
o  3074
```

```
o  3076 {------------------------------------  SYT$MONITOR_FLAGS              ----------------------------
o  3077 {Define Monitor Flags.
o  3078
o  3079   TYPE
o  3080     syt$monitor_flags = set of syt$monitor_flag;
o  3081
o  3082 {*copyc SYT$MONITOR_FLAG


o  3084 {------------------------------------  SYT$MONITOR_STATUS             ----------------------------
o  3085 {This deck defines the types for monitor request blocks.
o  3086
o  3087   TYPE
o  3088     syt$monitor_status = record
o  3089       normal: boolean,
o  3090       condition: ost$status_condition,
o  3091     recend;
o  3092
o  3093 {*copyc OST$STATUS


o  3095 {------------------------------------  TMC$BROKEN_TASK_FAULT_ID       ----------------------------


o  3097 {------------------------------------  TMC$EXECUTION_RING_CONSTANTS   ----------------------------
o  3098
o  3099   {Values for allocating handler execution rings - tmt$handler_execution_ring}
o  3100
o  3101   CONST
o  3102     tmc$unallocated = 0,
o  3103     tmc$task_monitor2_ring = osc$tmtr_ring,
o  3104     tmc$task_services_ring = osc$tsrv_ring,
o  3105     tmc$delay_allocation = 0f(16);
o  3106
o  3107   CONST
o  3108     tmc$lowest_signal_flag_ring = tmc$task_monitor2_ring,
o  3109     tmc$highest_signal_flag_ring = tmc$task_services_ring,
o  3110     tmc$highest_recognition_ring = tmc$highest_signal_flag_ring + 1;
o  3111
o  3112 {*copyc OSD$VIRTUAL_ADDRESS


o  3114 {------------------------------------  TMC$FREE_FLAG_ID               ----------------------------
o  3115
o  3116 { TMDFFID - This comdeck defines the SYSTEM FLAG VALUE for the free flag.
o  3117
o  3118   CONST
o  3119     tmc$free_flag_id = 0;
```



```
o  3121 {------------------------------------  TMC$LAST_FAULT_ID_ASSIGNED     ----------------------------


o  3123 {------------------------------------  TMC$MCR_FAULT                  ----------------------------


o  3125 {------------------------------------  TMC$UNKNOWN_SYSTEM_REQ_FAULT   ----------------------------


o  3127 {------------------------------------  TMT$ALLOCATED_EXECUTION_RINGS  ----------------------------
o  3128
o  3129   TYPE
o  3130     tmt$allocated_execution_rings = set of tmt$handler_execution_ring;
o  3131
o  3132 {*copyc TMT$HANDLER_EXECUTION_RING


o  3134 {------------------------------------  TMT$BROKEN_TASK_CONDITION      ----------------------------
o  3135
o  3136 {   Define conditions for which monitor mode processing decides a task is
o  3137 { broken.
o  3138
o  3139   TYPE
o  3140     tmt$broken_task_condition = (tmc$btc_mntr_fault_buffer_full,
o  3141       tmc$btc_mf_traps_disabled, tmc$btc_invalid_a0, tmc$btc_invalid_p,
o  3142       tmc$btc_mcr_traps_disabled, tmc$btc_ucr_traps_disabled,
o  3143       tmc$btc_system_error);


o  3145 {------------------------------------  TMT$BROKEN_TASK_MONITOR_FAULT  ----------------------------
o  3146
o  3147 {   Define monitor fault buffer contents for monitor fault sent when task is
o  3148 { determined to be broken.
o  3149 {
o  3150 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  3151 {   make the appropriate changes in the corresponding display procedures in the
o  3152 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  3153 {
o  3154
o  3155   TYPE
o  3156     tmt$broken_task_monitor_fault = record
o  3157       trap_enable: ost$trap_enable,
o  3158       case broken_task_condition: tmt$broken_task_condition of
o  3159       = tmc$btc_system_error =
o  3160         caller_p_register: ost$p_register,
o  3161         status_p: ^ost$status,
o  3162         text_p: ^string (*),
o  3163       = tmc$btc_mntr_fault_buffer_full, tmc$btc_mf_traps_disabled,
o  3164           tmc$btc_invalid_a0, tmc$btc_invalid_p,
o  3165           tmc$btc_mcr_traps_disabled, tmc$btc_ucr_traps_disabled =
o  3166         p: ost$p_register,
```

```
o  3167          a0: ^cell,
o  3168          monitor_condition_register: ost$monitor_conditions,
o  3169          user_condition_register: ost$user_conditions,
o  3170          monitor_fault_id: tmt$monitor_fault_identifiers,
o  3171        casend,
o  3172      recend;
o  3173
o  3174 {*copyc TMT$BROKEN_TASK_CONDITION
o  3175 {*copyc OSD$CONDITIONS
o  3176 {*copyc OSD$REGISTERS
o  3177 {*copyc OST$TRAP_ENABLE
o  3178 {*copyc OST$MONITOR_FAULT
o  3179 {*copyc OST$STATUS


o  3181 {------------------------------------  TMT$DISPATCH_CONTROL  ----------------------------
o  3182 {This deck defines the DISPATCHER control information that is in the CST.
o  3183 {The 1st 4 bytes are  private to the dispatcher except that CPU monitor will cause
o  3184 {the dispatcher to be called whenever the value of this record is not
o  3185 {equal to all zeros. (Requires knowledge of CYBIL data mapping.)
o  3186 {* * * Assembly language modules set the first byte of this record to 'TRUE'
o  3187 {      to force a call to the dispatcher.
o  3188 {The right 4 bytes of the record are used mainly by assembly decks.
o  3189
o  3190   TYPE
o  3191     tmt$dispatch_control = record
o  3192       call_dispatcher: boolean,
o  3193         rethread_current_task: boolean,
o  3194       new_task_status: tmt$task_status,
o  3195       fill: boolean,
o  3196       asynchronous_interrupts_pending: boolean,
o  3197     recend;
o  3198
o  3199 {*copyc TMT$TASK_STATUS


o  3201 {------------------------------------  TMT$DISPATCH_CONTROL_TABLE  ----------------------
o  3202
o  3203 { This common deck contains the type declarations for the DCT.
o  3204
o  3205
o  3206   TYPE
o  3207     tmt$dct_entry = RECORD
o  3208       queue_head: ALIGNED [0 MOD 8] 0 .. 0ffff(16),
o  3209       minor_priority: 0 .. 0ffff(16),
o  3210       major_priority: 0 .. 0ffff(16),
o  3211       queue_tail: 0 .. 0ffff(16),
o  3212     RECEND,
o  3213
o  3214     tmt$dispatch_control_table = array [jmt$dispatching_priority] of tmt$dct_entry;
o  3215
o  3216
o  3217
o  3218 {*copyc jmt$dispatching_priority
```



```
o  3219 {*copyc OST$GLOBAL_TASK_ID


o  3221 {------------------------------------  TMT$MCR_FAULTS  ----------------------------------
o  3222 {
o  3223 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  3224 {   make the appropriate changes in the corresponding display procedures in the
o  3225 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  3226 {
o  3227
o  3228   TYPE
o  3229     tmt$mcr_faults = record
o  3230       faults: ost$monitor_conditions,
o  3231       untranslatable_pointer: ost$pva,
o  3232     recend;
o  3233
o  3234 {*copyc OSD$VIRTUAL_ADDRESS
o  3235 {*copyc OSD$CONDITIONS


o  3237 {------------------------------------  TMT$MONITOR_FAULT_BUFFER  ------------------------
o  3238
o  3239   {Internal declarations for MONITOR FAULT BUFFER}
o  3240
o  3241   TYPE
o  3242     tmt$monitor_fault_buffer = record
o  3243       present: packed array [tmt$monitor_fault_buffers] of boolean,
o  3244       reserved: packed array [tmt$monitor_fault_buffers] of boolean,
o  3245       buffer: array [tmt$monitor_fault_buffers] of ost$monitor_fault,
o  3246     recend,
o  3247
o  3248     tmt$monitor_fault_buffers = 1 .. tmc$maximum_monitor_faults;
o  3249
o  3250   TYPE
o  3251     tmt$monitor_fault_identifiers = (tmc$null_fault,
o  3252       tmc$broken_task_fault_id, tmc$mcr_fault, mmc$segment_fault_processor_id,
o  3253       tmc$unknown_system_req_fault, syc$system_core_condition, tmc$dummy_fault);
o  3254
o  3255   CONST
o  3256     tmc$last_fault_id_assigned = 6,
o  3257     tmc$maximum_monitor_faults = 4;
o  3258
o  3259 {*copyc mmd$segment_access_condition
o  3260 {*copyc ost$monitor_fault
o  3261 {*copyc ost$name
o  3262 {*copyc ost$trap_enable
o  3263 {*copyc ost$status
o  3264 {*copyc syc$system_core_cond_constants
o  3265 {*copyc syt$system_core_condition
o  3266 {*copyc tmt$broken_task_condition
o  3267 {*copyc tmt$broken_task_monitor_fault
o  3268 {*copyc tmt$mcr_faults
```

```
o  3270 {------------------------------------  TMT$MONITOR_FAULT_HANDLER  ----------------------------
o  3271
o  3272   TYPE
o  3273     tmt$monitor_fault_handler = ^procedure (fault: ost$monitor_fault;
o  3274           save_area: ^ost$stack_frame_save_area);
o  3275
o  3276 {*copyc OST$MONITOR_FAULT
o  3277 {*copyc OST$STACK_FRAME_SAVE_AREA


o  3279 {------------------------------------  TMT$PRIMARY_TASK_LIST  ----------------------------
o  3280
o  3281 { This common deck contains the type declarations for the PTL.
o  3282
o  3283   CONST
o  3284     tmc$initial_ptl_size = 255,
o  3285     tmc$maximum_ptl = 0ffff(16),
o  3286     tmc$ptl_increment = 256;
o  3287
o  3288
o  3289
o  3290 {Define Primary Task List (PTL)}
o  3291
o  3292   TYPE
o  3293     tmt$primary_task_list_entry = record
o  3294       ptl_thread: ALIGNED [0 MOD 32] ost$task_index,
o  3295       sequence_number: 0 .. 255,
o  3296       xcb_offset: tmt$xcb_offset_size,
o  3297       ijl_ordinal: jmt$ijl_ordinal,
o  3298       status: tmt$task_status,
o  3299       new_task_status: tmt$task_status,
o  3300       idle_status: tmt$idle_status,
o  3301       queue_link: tmt$task_queue_link,
o  3302       monitor_flags: syt$monitor_flags,
o  3303       system_flags: tmt$system_flags,
o  3304       ptl_flags: tmt$ptl_flags,
o  3305       dispatching_priority: jmt$dispatching_priority,
o  3306       readying_task_priority: jmt$dispatching_priority,
o  3307       ijl_thread: ost$task_index,
o  3308       end_of_wait_time: 0 .. 0ffffffffffff(16),
o  3309     recend,
o  3310
o  3311     tmt$primary_task_list = array [0 .. * ] of tmt$primary_task_list_entry,
o  3312
o  3313     tmt$option = (tmc$opt_stop, tmc$opt_return),
o  3314
o  3315     tmt$xcb_offset_size = 0 .. 0ffffff(16),
o  3316
o  3317     tmt$ptl_flags = PACKED RECORD
o  3318        subsystem_locks_set: boolean,
o  3319        wait_inhibited: tmt$wait_inhibited,
o  3320     recend;
o  3321
o  3322 {*copyc jmt$ijl_ordinal
```



```
o  3323 {*copyc jmt$dispatching_priority
o  3324 {*copyc SYT$MONITOR_FLAGS
o  3325 {*copyc OST$GLOBAL_TASK_ID
o  3326 {*copyc TMT$SYSTEM_FLAGS
o  3327 {*copyc TMT$TASK_QUEUE_LINK
o  3328 {*copyc TMT$TASK_STATUS
o  3329 {*copyc TMT$WAIT_INHIBITED
o  3330


o  3332 {------------------------------------  TMT$SIGNAL  ----------------------------
o  3333
o  3334   TYPE
o  3335     tmt$signal = record
o  3336       originator: ost$global_task_id,
o  3337       signal: pmt$signal,
o  3338     recend;
o  3339
o  3340 {*copyc PMT$SIGNAL
o  3341 {*copyc OST$GLOBAL_TASK_ID


o  3343 {------------------------------------  TMT$SIGNAL_BUFFER  ----------------------------
o  3344
o  3345   {Internal declarations for the SIGNAL BUFFER}
o  3346
o  3347   TYPE
o  3348     tmt$signal_buffer = record
o  3349       present: packed array [tmt$signal_buffers] of boolean,
o  3350       reserved: packed array [tmt$signal_buffers] of boolean,
o  3351       buffer: array [tmt$signal_buffers] of tmt$signal,
o  3352     recend;
o  3353
o  3354 {*copyc TMT$SIGNAL
o  3355 {*copyc TMT$SIGNAL_BUFFERS


o  3357 {------------------------------------  TMT$SIGNAL_BUFFERS  ----------------------------
o  3358
o  3359   TYPE
o  3360     tmt$signal_buffers = 1 .. tmc$maximum_signals;
o  3361
o  3362   CONST
o  3363     tmc$maximum_signals = 4;


o  3365 {------------------------------------  TMT$SIGNAL_HANDLER  ----------------------------
o  3366
o  3367   TYPE
o  3368     tmt$signal_handler = ^procedure (originator: ost$global_task_id;
o  3369           signal: pmt$signal);
o  3370
```

```
o  3371 {*copyc OST$GLOBAL_TASK_ID
o  3372 {*copyc PMT$SIGNAL


o  3374 {------------------------------------  TMT$SIGNAL_STATUS                 ----------------------------
o  3375
o  3376   TYPE
o  3377     tmt$signal_status = (tmc$normal_signal_status, tmc$no_signal_present,
o  3378       tmc$invalid_buffer_index);


o  3380 {------------------------------------  TMT$SYSTEM_FLAGS                  ----------------------------
o  3381
o  3382   TYPE
o  3383     tmt$system_flags = set of ost$system_flag;
o  3384
o  3385 {*copyc OST$SYSTEM_FLAG


o  3387 {------------------------------------  TMT$SYSTEM_TASK_ID                ----------------------------
o  3388 { tmdstid - system task id definitions.
o  3389
o  3390   CONST
o  3391     tmc$maximum_system_task_id = 30;
o  3392
o  3393   TYPE
o  3394     tmt$system_task_id = tmc$stid_null_task .. tmc$maximum_system_task_id;
o  3395
o  3396   CONST
o  3397     tmc$stid_null_task = 0,
o  3398     tmc$stid_memory_link_helper = 1,
o  3399     tmc$stid_task_id_2 = 2,
o  3400     tmc$stid_administer_log = 3,
o  3401     tmc$stid_dm_split_al = 4,
o  3402     tmc$stid_volume_space_managemnt = 5,
o  3403     tmc$stid_job_scheduler = 6,
o  3404     tmc$stid_job_monitor = 7,
o  3405     tmc$stid_task_id_8 = 8,
o  3406     tmc$stid_completed_output = 9,
o  3407     tmc$stid_intranet_layer_mgmt = 10,
o  3408     tmc$stid_namve_system_input = 11,
o  3409     tmc$stid_tape_scanner = 12,
o  3410     tmc$stid_task_id_13 = 13,
o  3411     tmc$stid_task_id_14 = 14,
o  3412     tmc$stid_task_id_15 = 15,
o  3413     tmc$stid_task_id_16 = 16,
o  3414     tmc$stid_task_id_17 = 17,
o  3415     tmc$stid_task_id_18 = 18,
o  3416     tmc$stid_task_id_19 = 19,
o  3417     tmc$stid_task_id_20 = 20,
o  3418     tmc$stid_task_id_21 = 21,
o  3419     tmc$stid_task_id_22 = 22,
o  3420     tmc$stid_task_id_24 = 24,
```



```
o  3421     tmc$stid_task_id_25 = 25,
o  3422     tmc$stid_task_id_26 = 26,
o  3423     tmc$stid_task_id_27 = 27,
o  3424     tmc$stid_task_id_28 = 28,
o  3425     tmc$stid_task_id_29 = 29,
o  3426     tmc$stid_task_id_30 = 30;


o  3428 {------------------------------------  TMT$TASK_STATUS                   ----------------------------
o  3429
o  3430 { This common deck contains the task status codes used by the
o  3431 { task management procedures that run in monitor mode.
o  3432
o  3433
o  3434 {Define task status values. Task status' are divided into 3 groups depending on where tasks that
o  3435 {are in the status are threaded in the dispatch tables.
o  3436 {   GROUP 1A- tasks are linked into the DCT.
o  3437 {   GROUP 1B- tasks are ready and have been selected to execute on a specific processor.
o  3438 {   GROUP 2A- tasks are threaded into a timed wait queue.
o  3439 {   GROUP 2B- tasks are in a timed wait, but will not be threaded into a timed wait queue until
o  3440 {             closer to the end of wait time.
o  3441 {   GROUP 3A- tasks are not in any queue. Tasks are waiting for a 'conditional' ready
o  3442 {             task request.
o  3443 {   GROUP 3B- tasks are not in any queue. However, tasks can only be made ready
o  3444 {             by an 'unconditional' ready task request.
o  3445 {   GROUP 3C- tasks are not in a DCT queue but are queued on some external event in a wait queue.
o  3446 {             Tasks can only be placed or removed in/from these status via the
o  3447 {             tmp$queue_task and tmp$dequeue_task requests.
o  3448
o  3449   CONST
o  3450     tmc$ts_last_status_in_dct = tmc$ts_ready,
o  3451     tmc$ts_first_status_in_wait_q = tmc$ts_timeout_reqexp_shortshrt,
o  3452     tmc$ts_last_status_in_wait_q = tmc$ts_timeout_reqexp_longvlong,
o  3453     tmc$last_timed_wait_status = tmc$ts_timed_wait_not_queued,
o  3454     tmc$ts_first_ready_uncond = tmc$ts_io_wait_not_queued,
o  3455     tmc$ts_first_external_queue = tmc$ts_page_wait;
o  3456
o  3457
o  3458   TYPE
o  3459     tmt$task_status = (tmc$ts_null, tmc$ts_ready,
o  3460 {                                                                    END GROUP 1A}
o  3461           tmc$ts_ready_and_selected,
o  3462 {                                                                    END GROUP 1B}
o  3463           tmc$ts_timeout_reqexp_shortshrt,
o  3464           tmc$ts_timeout_reqexp_longlong, tmc$ts_timeout_reqexp_longvlong,
o  3465 {                                                                    END GROUP 2A}
o  3466           tmc$ts_timed_wait_not_queued,
o  3467 {                                                                    END GROUP 2B}
o  3468           tmc$ts_executing,
o  3469           tmc$ts_timeout_reqexp_inflong, tmc$ts_timeout_reqexp_infvlong,
o  3470           tmc$ts_ready_but_swapped,
o  3471 {                                                                    END GROUP 3A}
o  3472           tmc$ts_io_wait_not_queued,
o  3473 {                                                                    END GROUP 3B}
o  3474           tmc$ts_page_wait, tmc$ts_memory_wait, tmc$ts_segment_lock_wait,
```

```
TMT$TASK_STATUS

        o  3475            tmc$ts_job_event_queue, tmc$ts_io_wait_queued, tmc$ts_volume_unavailable),
        o  3476 {                                                                    END GROUP 3C}
        o  3477      tmt$idle_status = (tmc$is_not_idled, tmc$is_idle_initiated, tmc$is_idled,
        o  3478        tmc$is_idled_sched_notified),
        o  3479
        o  3480      tmt$ready_condition = (tmc$rc_ready_conditional_wi, tmc$rc_ready_conditional);
```



```
Decks referenced by selected decks

        o  3483  PROCEDURE p; VAR pp: ^cell;
        4  3484    VAR t0:^ DFT$SERVER_DESCRIPTOR           ;
        4  3485    VAR t1:^ DMT$DISK_FILE_DESCRIPTOR        ;
        4  3486    VAR t2:^ DMT$FILE_ALLOCATION_STATUS      ;
        4  3487    VAR t3:^ DMT$FILE_MEDIUM_DESCRIPTOR      ;
        4  3488    VAR t4:^ GFT$FILE_DESCRIPTOR_ENTRY       ;
        4  3489    VAR t5:^ JMT$ACTIVE_JOB_LIST_ENTRY       ;
        4  3490    VAR t6:^ JMT$INITIATED_JOB_LIST_ENTRY    ;
        4  3491    VAR t7:^ JMT$JOB_CONTROL_BLOCK           ;
        4  3492    VAR t8:^ MMT$ACTIVE_SEGMENT_TABLE_ENTRY  ;
        4  3493    VAR t9:^ MMT$PAGE_FRAME_TABLE_ENTRY      ;
        4  3494    VAR t10:^ MMT$SEGMENT_DESCRIPTOR_TABLE_EX;
        4  3495    VAR t11:^ MTT$SMU_COMMUNICATIONS_BLOCK   ;
        4  3496    VAR t12:^ OST$CPU_STATE_TABLE            ;
        4  3497    VAR t13:^ OST$EXECUTION_CONTROL_BLOCK    ;
        4  3498    VAR t14:^ TMT$PRIMARY_TASK_LIST_ENTRY    ;
        4  3499     pp := ^t0;
        C  3500     pp := ^t1;
       14  3501     pp := ^t2;
       1C  3502     pp := ^t3;
       24  3503     pp := ^t4;
       2C  3504     pp := ^t5;
       34  3505     pp := ^t6;
       3C  3506     pp := ^t7;
       44  3507     pp := ^t8;
       4C  3508     pp := ^t9;
       54  3509     pp := ^t10;
       5C  3510     pp := ^t11;
       64  3511     pp := ^t12;
       6C  3512     pp := ^t13;
       74  3513     pp := ^t14;
       7C  3514  procend;
        o  3516
        o  3517
        o  3518    CONST
        o  3519      amc$file_byte_limit = 4398046511103 (2**42 - 1 bytes) ;
        o  3520
        o  3521    TYPE
        o  3522      amt$file_byte_address = 0 .. amc$file_byte_limit;
        o  3523   TYPE
        o  3524      amt$file_limit = 0 .. amc$file_byte_limit;
        o  3525
        o  3526
        o  3527    TYPE
        o  3528      amt$preset_value = integer;
        o  3529
        o  3530    TYPE
        o  3531      dft$lifetime = 0 .. dfc$maximum_lifetime;
        o  3532
        o  3533    CONST
        o  3534      dfc$maximum_lifetime = 0FFFF(16);
        o  3535
        o  3536   TYPE
        o  3537      dft$served_family_table_index = record
        o  3538        pointers_index: dft$family_pointer_index,
        o  3539        family_list_index: dft$served_family_list_index,
```

Decks referenced by selected decks

```
o  3540      recend;
o  3541
o  3542    CONST
o  3543      dfc$served_family_list_size = 16;
o  3544
o  3545    TYPE
o  3546      dft$served_family_list_index = 1 .. dfc$served_family_list_size;
o  3547
o  3548    CONST
o  3549      dfc$max_family_ptr_array_size = 0ff(16);
o  3550
o  3551    TYPE
o  3552      dft$family_pointer_index = 1 .. dfc$max_family_ptr_array_size;
o  3553
o  3554    TYPE
o  3555      dft$server_allocation_info = RECORD
o  3556        CASE allocation_needed_on_server: boolean OF
o  3557        = FALSE =
o  3558          invalid_data: 0 .. 3fffffffffff(16),
o  3559        = TRUE =
o  3560          bytes_to_allocate: amt$file_byte_address,
o  3561        CASEND,
o  3562      RECEND;
o  3563
o  3566 { DECK: DFT$SERVER_STATE
o  3567
o  3568    TYPE
o  3569      dft$server_state = (dfc$active, dfc$deactivated, dfc$inactive,
o  3570           dfc$awaiting_recovery, dfc$recovering, dfc$terminated, dfc$deleted),
o  3571
o  3572     dft$server_states = set of dft$server_state;
o  3573 {
o  3574 {                  common deck dmdaloc
o  3575 {
o  3576
o  3577    TYPE
o  3578      dmt$allocation_size = 0 .. dmc$max_bytes_per_allocation,
o  3579      dmt$bytes_per_allocation = dmc$min_bytes_per_allocation ..
o  3580        dmc$max_bytes_per_allocation,
o  3581      dmt$byte_offset_within_au = 0 .. dmc$max_bytes_per_allocation;
o  3582
o  3583    CONST
o  3584      dmc$min_bytes_per_allocation = 4096,
o  3585      dmc$max_bytes_per_allocation = 0ffffff(16),
o  3586      dmc$maximum_page_size = 16384;
o  3587
o  3588    TYPE
o  3589      dmt$allocation_styles = (dmc$a0, dmc$a1, dmc$a2, dmc$a3, dmc$a4, dmc$a5,
o  3590        dmc$a6, dmc$a7, dmc$a8, dmc$acyl);
o  3591
o  3592    CONST
o  3593      dmc$default_allocation_style = dmc$a2,
o  3594      dmc$default_req_alloc_size = 16384,    {delete this ?}
o  3595      dmc$unspecified_allocation_size = 0;
o  3596 {
o  3597 {  dmt$minimum_allocation_unit
```


Decks referenced by selected decks

```
o  3598 {
o  3599
o  3600    TYPE
o  3601      dmt$bytes_per_mau = 0 .. dmc$max_bytes_per_mau,
o  3602      dmt$maus_per_allocation = dmc$min_maus_per_allocation ..
o  3603        dmc$max_maus_per_allocation,
o  3604      dmt$mau_offset_within_au = 0 .. dmc$max_maus_per_allocation,
o  3605      dmt$maus_per_dau = dmc$min_maus_per_dau .. dmc$max_maus_per_dau,
o  3606      dmt$mau_offset_within_dau = 0 .. dmc$max_maus_per_dau,
o  3607      dmt$maus_per_transfer = dmc$min_maus_per_transfer ..
o  3608        dmc$max_maus_per_transfer,
o  3609      dmt$mau_offset_within_tu = 0 .. dmc$max_maus_per_transfer,
o  3610      dmt$maus_per_position = 0 .. dmc$max_maus_position,
o  3611      dmt$mau_address = dmc$min_mau_address .. dmc$max_mau_address;
o  3612
o  3613    CONST
o  3614      dmc$min_bytes_per_mau = 1,
o  3615      dmc$max_bytes_per_mau = 4096,
o  3616      dmc$min_maus_per_allocation = 2,
o  3617      dmc$max_maus_per_allocation = 392,
o  3618      dmc$min_maus_per_dau = 2,
o  3619      dmc$max_maus_per_dau = 48,
o  3620      dmc$min_maus_per_transfer = 1,
o  3621      dmc$max_maus_per_transfer = 392,
o  3622      dmc$min_mau_address = 0,
o  3623      dmc$max_mau_address = dmc$max_dau_address * dmc$max_maus_per_dau,
o  3624      dmc$min_maus_position = 1,
o  3625      dmc$max_maus_position = dmc$max_daus_position * dmc$max_maus_per_dau;
o  3626
o  3627 {
o  3628 {                      common deck dmddau
o  3629 {
o  3630
o  3631    TYPE
o  3632      dmt$bytes_per_dau = dmc$min_bytes_per_dau .. dmc$max_bytes_per_dau,
o  3633      dmt$daus_per_allocation = 0 .. dmc$max_daus_allocation,
o  3634      dmt$daus_per_position = 0 .. dmc$max_daus_position,
o  3635      dmt$daus_per_transfer = 0 .. dmc$max_daus_transfer,
o  3636      dmt$dau_address = dmc$min_dau_address .. dmc$max_dau_address;
o  3637
o  3638    CONST
o  3639      dmc$min_bytes_per_dau = 1,
o  3640      dmc$max_bytes_per_dau = 16384,
o  3641      dmc$min_dau_address = 0,
o  3642      dmc$max_dau_address = 134880,
o  3643      dmc$min_daus_allocation = 1,
o  3644      dmc$max_daus_allocation = 160,
o  3645      dmc$min_daus_position = 1,
o  3646      dmc$max_daus_position = 160,
o  3647      dmc$min_daus_transfer = 1,
o  3648      dmc$max_daus_transfer = 160,
o  3649      dmc$min_dau_size = 1,
o  3650      dmc$max_dau_size = 16384;
o  3651 {
o  3652 {              common deck dmdtran
o  3653 {
```

Decks referenced by selected decks

```
o  3654
o  3655    TYPE
o  3656      dmt$transfer_size = 0 .. dmc$max_transfer_size;
o  3657
o  3658    CONST
o  3659      dmc$default_transfer_size = 8192,                {is this old value still valid?
o  3660      dmc$default_req_transfer_size = 16384,
o  3661      dmc$max_transfer_size = 0ffffff(16),
o  3662      dmc$unspecified_transfer_size = 0;
o  3663
o  3664
o  3665  { Secure memory/file parameter }
o  3666
o  3667  TYPE
o  3668    ost$clear_file_space = boolean;
o  3669
o  3670    TYPE
o  3671      pmt$binary_mainframe_id = record
o  3672        model_number: ost$processor_model_number,
o  3673        serial_number: ost$processor_serial_number,
o  3674      recend;
o  3675
o  3676
o  3677    TYPE
o  3678      ost$processor_model_number = 0 .. 0ff(16);
o  3679
o  3680    CONST
o  3681      osc$cyber_180_model_unknown = 000(16),
o  3682      osc$cyber_180_model_810 = 014(16),
o  3683      osc$cyber_180_model_815 = 011(16),
o  3684      osc$cyber_180_model_825 = 012(16),
o  3685      osc$cyber_180_model_830 = 013(16),
o  3686      osc$cyber_180_model_835 = 020(16),
o  3687      osc$cyber_180_model_840 = 034(16),
o  3688      osc$cyber_180_model_840s = 037(16),
o  3689      osc$cyber_180_model_845 = 031(16),
o  3690      osc$cyber_180_model_845s = 035(16),
o  3691      osc$cyber_180_model_850 = 033(16),
o  3692      osc$cyber_180_model_855 = 030(16),
o  3693      osc$cyber_180_model_855s = 036(16),
o  3694      osc$cyber_180_model_860 = 032(16),
o  3695      osc$cyber_180_model_990 = 040(16),
o  3696      osc$cyber_180_model_990e = 041(16),
o  3697      osc$cyber_180_model_9301 = 053(16),
o  3698      osc$cyber_180_model_9303 = 052(16),
o  3699      osc$cyber_180_model_930a = 05B(16),
o  3700      osc$cyber_180_model_930b = 05D(16),
o  3701      osc$cyber_180_model_930c = 05E(16),
o  3702      osc$cyber_180_model_930d = 051(16),
o  3703      osc$cyber_900_model_9321 = 055(16),
o  3704      osc$cyber_900_model_9323 = 054(16),
o  3705      osc$cyber_900_model_932a = 05C(16),
o  3706      osc$cyber_900_model_932b = 05F(16),
o  3707      osc$cyber_900_model_9601 = 03b(16),
o  3708      osc$cyber_900_model_9603 = 03a(16),
o  3709      osc$cyber_900_model_992 = 042(16),
```


Decks referenced by selected decks

```
o  3710      osc$cyber_900_model_994 = 044(16);
o  3711
o  3712  { The following constants are retained for compatability only.
o  3713
o  3714    CONST
o  3715      osc$cyber_180_model_992 = 042(16),
o  3716      osc$cyber_180_model_994 = 044(16),
o  3717      osc$cyber_180_model_9321 = 055(16),
o  3718      osc$cyber_180_model_9323 = 054(16),
o  3719      osc$cyber_180_model_932a = 05C(16),
o  3720      osc$cyber_180_model_932b = 05F(16),
o  3721      osc$cyber_180_model_9601 = 03b(16),
o  3722      osc$cyber_180_model_9603 = 03a(16);
o  3723
o  3724
o  3725    TYPE
o  3726      pmt$processor_attributes = record
o  3727        model_number: pmt$cpu_model_number,
o  3728        serial_number: pmt$cpu_serial_number,
o  3729        page_size: ost$page_size,
o  3730      recend,
o  3731
o  3732      pmt$processor = record
o  3733        serial_number: pmt$cpu_serial_number,
o  3734        model_number: pmt$cpu_model_number,
o  3735      recend;
o  3736
o  3737    TYPE
o  3738      pmt$cpu_model_number = (pmc$cpu_model_p1, pmc$cpu_model_p2,
o  3739        pmc$cpu_model_p3, pmc$cpu_model_p4),
o  3740
o  3741      pmt$cpu_serial_number = 0 .. 0ffff(16);
o  3742
o  3743
o  3744    TYPE
o  3745
o  3746      {page size in bytes}
o  3747
o  3748      ost$page_size = osc$min_page_size .. osc$max_page_size;
o  3749
o  3750    CONST
o  3751      osc$min_page_size = 512,
o  3752      osc$max_page_size = 65536;
o  3753
o  3754
o  3755    TYPE
o  3756      ost$processor_serial_number = 0 .. 0ffff(16);
o  3757  {
o  3758  { dmt$file_attributes
o  3759  {
o  3760
o  3761    TYPE
o  3762      dmt$file_attribute = record
o  3763        case keyword: dmt$file_attribute_keywords of
o  3764        = dmc$allocated_length =
o  3765          allocated_length: amt$file_byte_address,
```

Decks referenced by selected decks

```
o  3766         = dmc$asid =
o  3767           asid: ost$asid,
o  3768         = dmc$byte_address =
o  3769           byte_address: amt$file_byte_address,
o  3770         = dmc$bytes_per_allocation =
o  3771           bytes_per_allocation: dmt$allocation_size,
o  3772         = dmc$class =
o  3773           class: dmt$class_member,
o  3774         = dmc$class_ordinal =
o  3775           ordinal: dmt$class_ordinal,
o  3776         = dmc$clear_space =
o  3777           required: ost$clear_file_space,
o  3778         = dmc$device_file_list_index =
o  3779           device_file_list_index: dmt$device_file_list_index,
o  3780         = dmc$eof_byte_address =
o  3781           eof_address: amt$file_byte_address,
o  3782         = dmc$eoi_byte_address =
o  3783           eoi_address: amt$file_byte_address,
o  3784         = dmc$file_hash =
o  3785           file_hash: dmt$file_hash,
o  3786         = dmc$file_limit =
o  3787           limit: amt$file_limit,
o  3788         = dmc$file_status =
o  3789           file_modified: boolean,
o  3790         = dmc$file_kind =
o  3791           file_kind: gft$file_kind,
o  3792         = dmc$global_file_name =
o  3793           global_file_name: dmt$global_file_name,
o  3794         = dmc$internal_vsn =
o  3795           internal_vsn: dmt$internal_vsn,
o  3796         = dmc$locked_file =
o  3797           file_lock: dmt$locked_file,
o  3798         = dmc$logical_length =
o  3799           logical_length: amt$file_byte_address,
o  3800         = dmc$master_volume_required =
o  3801           master_volume_required: boolean,
o  3802         = dmc$overflow =
o  3803           overflow_allowed: boolean,
o  3804         = dmc$owner =
o  3805           file_space_limit: sft$file_space_limit_kind,
o  3806         = dmc$preset_value =
o  3807           preset_value: amt$preset_value,
o  3808         = dmc$recorded_vsn =
o  3809           recorded_vsn: rmt$recorded_vsn,
o  3810         = dmc$requested_allocation_size =
o  3811           requested_allocation_size: dmt$allocation_size,
o  3812         = dmc$requested_transfer_size =
o  3813           requested_transfer_size: dmt$transfer_size,
o  3814         = dmc$requested_volume =
o  3815           requested_volume: dmt$requested_volume,
o  3816         = dmc$setname =
o  3817           setname: stt$set_name,
o  3818         = dmc$chapter_length =
o  3819           chapter_length: ost$segment_length,
o  3820         = dmc$write_mode =
o  3821           attached_in_write_mode: boolean,
```


Decks referenced by selected decks

```
o  3822         = dmc$queue_status =
o  3823           queue_status: gft$queue_status,
o  3824         casend,
o  3825       recend,
o  3826       dmt$file_attribute_keywords = (dmc$allocated_length, dmc$asid,
o  3827         dmc$byte_address, dmc$bytes_per_allocation, dmc$class, dmc$class_ordinal,
o  3828         dmc$clear_space, dmc$device_file_list_index, dmc$eof_byte_address,
o  3829         dmc$eoi_byte_address, dmc$file_hash, dmc$file_limit, dmc$file_status,
o  3830         dmc$file_kind, dmc$global_file_name, dmc$internal_vsn, dmc$locked_file,
o  3831         dmc$logical_length, dmc$master_volume_required, dmc$null_attribute, dmc$overflow,
o  3832         dmc$owner, dmc$preset_value, dmc$recorded_vsn, dmc$requested_allocation_size,
o  3833         dmc$requested_transfer_size, dmc$requested_volume, dmc$setname,
o  3834         dmc$chapter_length, dmc$write_mode, dmc$queue_status);
o  3835
o  3836
o  3837
o  3838 { NOS/VE address constants. }
o  3839
o  3840
o  3841   CONST
o  3842
o  3843
o  3844     { Ring names. }
o  3845
o  3846     osc$min_ring = 1, { Lowest ring number (most privileged). }
o  3847     osc$max_ring = 15, { Highest ring number (least privileged). }
o  3848     osc$invalid_ring = 0,
o  3849     osc$os_ring_1 = 1, { Reserved for Operating System. }
o  3850     osc$tmtr_ring = 2, { Task Monitor. }
o  3851     osc$tsrv_ring = 3, { Task services. }
o  3852     osc$sj_ring_1 = 4, { Reserved for system job. }
o  3853     osc$sj_ring_2 = 5,
o  3854     osc$sj_ring_3 = 6,
o  3855     osc$application_ring_1 = 7, { Reserved for application subsystems.}
o  3856     osc$application_ring_2 = 8,
o  3857     osc$application_ring_3 = 9,
o  3858     osc$application_ring_4 = 10,
o  3859     osc$user_ring = 11, { Standard user task. }
o  3860     osc$user_ring_1 = 12, { Reserved for user...O.S. requests available.}
o  3861     osc$user_ring_2 = 13,
o  3862     osc$user_ring_3 = 14, { Reserved for user...O.S. requests not available. }
o  3863     osc$user_ring_4 = 15;
o  3864
o  3865
o  3866 { Virtual address space dimensions. }
o  3867
o  3868   CONST
o  3869     osc$maximum_segment = 0fff(16),
o  3870     osc$maximum_offset = 7fffffff(16),
o  3871     osc$max_segment_length = osc$maximum_offset + 1;
o  3872
o  3873
o  3874 { Global-local key lock definition. }
o  3875
o  3876   TYPE
o  3877     ost$key_lock = packed record
```

Decks referenced by selected decks

```
o  3878        global: boolean, { True if value is global key. }
o  3879        local: boolean, { True if value is local key. }
o  3880        value: ost$key_lock_value, { Key or lock value. }
o  3881      recend,
o  3882
o  3883      ost$key_lock_value = 0 .. 3f(16),
o  3884
o  3885
o  3886      { CYBER 180 forty eight bit PVA definition. }
o  3887
o  3888      ost$ring = osc$invalid_ring .. osc$max_ring, { Ring number. }
o  3889      ost$valid_ring = osc$min_ring .. osc$max_ring, { Valid Ring Number. }
o  3890      ost$segment = 0 .. osc$maximum_segment, { Segment number. }
o  3891      ost$segment_offset = - (osc$maximum_offset + 1) .. osc$maximum_offset,
o  3892      ost$valid_segment_offset = 0 .. osc$maximum_offset,
o  3893
o  3894      ost$segment_length = 0 .. osc$max_segment_length,
o  3895
o  3896      ost$relative_pointer = - 7fffffff(16) .. 7fffffff(16),
o  3897      ost$valid_relative_pointer = 0 .. 7fffffff(16),
o  3898
o  3899      ost$pva = packed record
o  3900        ring: ost$ring,
o  3901        seg: ost$segment,
o  3902        offset: ost$segment_offset,
o  3903      recend;
o  3904    TYPE
o  3905      amt$global_file_position = amt$file_position,
o  3906      amt$label_options = set of (amc$vol1, amc$uv1, amc$hdr1, amc$hdr2,
o  3907        amc$eov1, amc$eov2, amc$uh1, amc$eof1, amc$eof2, amc$ut1),
o  3908      amt$return_option = (amc$return_at_close, amc$return_at_task_exit,
o  3909        amc$return_at_job_exit);
o  3910
o  3911   TYPE
o  3912      amt$file_access_selections = ^array [1 .. * ] of amt$access_selection,
o  3913
o  3914      amt$access_selection = amt$file_item;
o  3915
o  3916   TYPE
o  3917      amt$file_attributes = array [1 .. * ] of amt$file_item,
o  3918      amt$file_item = record
o  3919        case key {input} : amt$file_attribute_keys of {input}
o  3920  {}
o  3921  { The caller of amp$file must initialize the tag field selector (key) }
o  3922  { and store the indicated attribute value into this record before}
o  3923  { calling amp$file.}
o  3924  {}
o  3925        = amc$access_mode =
o  3926          access_mode: pft$usage_selections,
o  3927        = amc$block_type =
o  3928          block_type: amt$block_type,
o  3929        = amc$character_conversion =
o  3930          character_conversion: boolean,
o  3931        = amc$clear_space =
o  3932          clear_space: ost$clear_file_space,
o  3933        = amc$error_exit_name =
```


Decks referenced by selected decks

```
o  3934          error_exit_name: pmt$program_name,
o  3935        = amc$error_options =
o  3936          error_options: amt$tape_error_options,
o  3937        = amc$file_access_procedure =
o  3938          file_access_procedure: pmt$program_name,
o  3939        = amc$file_contents =
o  3940          file_contents: amt$file_contents,
o  3941        = amc$file_limit =
o  3942          file_limit: amt$file_limit,
o  3943        = amc$file_organization =
o  3944          file_organization: amt$file_organization,
o  3945        = amc$file_processor =
o  3946          file_processor: amt$file_processor,
o  3947        = amc$file_structure =
o  3948          file_structure: amt$file_structure,
o  3949        = amc$forced_write =
o  3950          forced_write: amt$forced_write,
o  3951        = amc$internal_code =
o  3952          internal_code: amt$internal_code,
o  3953        = amc$label_exit_name =
o  3954          label_exit_name: pmt$program_name,
o  3955        = amc$label_options =
o  3956          label_options: amt$label_options,
o  3957        = amc$label_type =
o  3958          label_type: amt$label_type,
o  3959        = amc$line_number =
o  3960          line_number: amt$line_number,
o  3961        = amc$max_block_length =
o  3962          max_block_length: amt$max_block_length,
o  3963        = amc$max_record_length =
o  3964          max_record_length: amt$max_record_length,
o  3965        = amc$min_block_length =
o  3966          min_block_length: amt$min_block_length,
o  3967        = amc$min_record_length =
o  3968          min_record_length: amt$min_record_length,
o  3969        = amc$null_attribute =
o  3970          ,
o  3971        = amc$open_position =
o  3972          open_position: amt$open_position,
o  3973        = amc$padding_character =
o  3974          padding_character: amt$padding_character,
o  3975        = amc$page_format =
o  3976          page_format: amt$page_format,
o  3977        = amc$page_length =
o  3978          page_length: amt$page_length,
o  3979        = amc$page_width =
o  3980          page_width: amt$page_width,
o  3981        = amc$preset_value =
o  3982          preset_value: amt$preset_value,
o  3983        = amc$record_type =
o  3984          record_type: amt$record_type,
o  3985        = amc$return_option =
o  3986          return_option: amt$return_option,
o  3987        = amc$ring_attributes =
o  3988          ring_attributes: amt$ring_attributes,
o  3989        = amc$statement_identifier =
```

Decks referenced by selected decks

```
o  3990          statement_identifier: amt$statement_identifier,
o  3991        : amc$user_info :
o  3992          user_info: amt$user_info,
o  3993        : amc$vertical_print_density :
o  3994          vertical_print_density: amt$vertical_print_density,
o  3995 {}
o  3996 { The following attributes are only used to describe files which}
o  3997 { are accessed with the Advanced Access Method (AAM). The}
o  3998 { documentation of the AAM attributes are found in the AAM ERS.}
o  3999 {}
o  4000        : amc$average_record_length :
o  4001          average_record_length: amt$average_record_length,
o  4002        : amc$collate_table_name :
o  4003          collate_table_name: pmt$program_name,
o  4004        : amc$compression_procedure_name :
o  4005          compression_procedure_name: [input,output]
o  4006            ^amt$compression_procedure_name,
o  4007        : amc$data_padding :
o  4008          data_padding: amt$data_padding,
o  4009        : amc$dynamic_home_block_space :
o  4010          dynamic_home_block_space: amt$dynamic_home_block_space,
o  4011        : amc$embedded_key :
o  4012          embedded_key: boolean,
o  4013        : amc$error_limit :
o  4014          error_limit: amt$error_limit,
o  4015        : amc$estimated_record_count :
o  4016          estimated_record_count: amt$estimated_record_count,
o  4017        : amc$hashing_procedure_name :
o  4018          hashing_procedure_name: [input,output] ^amt$hashing_procedure_name,
o  4019        : amc$index_levels :
o  4020          index_levels: amt$index_levels,
o  4021        : amc$index_padding :
o  4022          index_padding: amt$index_padding,
o  4023        : amc$initial_home_block_count :
o  4024          initial_home_block_count: amt$initial_home_block_count,
o  4025        : amc$key_length :
o  4026          key_length: amt$key_length,
o  4027        : amc$key_position :
o  4028          key_position: amt$key_position,
o  4029        : amc$key_type :
o  4030          key_type: amt$key_type,
o  4031        : amc$loading_factor :
o  4032          loading_factor: amt$loading_factor,
o  4033        : amc$lock_expiration_time :
o  4034          lock_expiration_time: amt$lock_expiration_time,
o  4035        : amc$logging_options :
o  4036          logging_options: amt$logging_options,
o  4037        : amc$log_residence :
o  4038          log_residence: [input,output] ^amt$log_residence,
o  4039        : amc$message_control :
o  4040          message_control: amt$message_control,
o  4041        : amc$record_limit :
o  4042          record_limit: amt$record_limit,
o  4043        : amc$records_per_block :
o  4044          records_per_block: amt$records_per_block,
o  4045        casend
```


Decks referenced by selected decks

```
o  4046      recend;
o  4047
o  4048
o  4049    TYPE
o  4050      amt$block_header_type = (amc$tapemark_block, amc$data_block),
o  4051      amt$block_status = (amc$no_error, amc$unrecovered_error),
o  4052      amt$pack_block_header = record
o  4053        header_type: amt$block_header_type,
o  4054        block_length: amt$max_block_length,
o  4055        block_number: amt$block_number,
o  4056        unused_bit_count: amt$unused_bit_count,
o  4057      recend,
o  4058      amt$unpack_block_header = record
o  4059        header_type: amt$block_header_type,
o  4060        block_length_as_read: amt$max_block_length,
o  4061        block_length_as_written: amt$max_block_length,
o  4062        block_number: amt$block_number,
o  4063        unused_bit_count: amt$unused_bit_count,
o  4064        block_status: amt$block_status,
o  4065      recend;
o  4066
o  4067
o  4068    CONST
o  4069      amc$max_block_number = 0ffffffff[16];
o  4070
o  4071    TYPE
o  4072      amt$block_number = 1 .. amc$max_block_number;
o  4073   TYPE
o  4074      amt$max_block_length = 1 .. amc$maximum_block;
o  4075
o  4076   CONST
o  4077      amc$maximum_block = osc$max_segment_length - 32;
o  4078
o  4079
o  4080
o  4081    TYPE
o  4082      amt$unused_bit_count = 0 .. 7;
o  4083
o  4084    CONST
o  4085 {    Use constants in this deck with amp$open.
o  4086 {    Use constants in deck fsc$file_contents with fsp$open_file.
o  4087      amc$unknown_contents = 'UNKNOWN',
o  4088      amc$legible = 'LEGIBLE',
o  4089      amc$list = 'LIST',
o  4090      amc$object = 'OBJECT',
o  4091      amc$screen = 'SCREEN';
o  4092
o  4093
o  4094    TYPE
o  4095      amt$file_contents = ost$name;
o  4096
o  4097 {    Use constants in deck amd$file_contents with amp$open}
o  4098 {    Use constants in deck fsc$file_contents with fsp$open_file}
o  4099
o  4100
o  4101 { The following are NOS/VE system conventions for referring to the}
```

Decks referenced by selected decks

```
o  4102 { contents of a file:
o  4103
o  4104 ?? FMT (FORMAT := OFF) ??
o  4105  CONST
o  4106     fsc$ascii_log                 = 'ASCII_LOG                    ',
o  4107     fsc$binary_log                = 'BINARY_LOG                   ',
o  4108     fsc$data                      = 'DATA                         ',
o  4109     fsc$file_backup               = 'FILE_BACKUP                  ',
o  4110     fsc$legible_data              = 'LEGIBLE_DATA                 ',
o  4111     fsc$legible_library           = 'LEGIBLE_LIBRARY              ',
o  4112     fsc$legible_scl_include       = 'LEGIBLE_SCL_INCLUDE          ',
o  4113     fsc$legible_scl_job           = 'LEGIBLE_SCL_JOB              ',
o  4114     fsc$legible_scl_procedure     = 'LEGIBLE_SCL_PROCEDURE        ',
o  4115     fsc$list                      = 'LIST                         ',
o  4116     fsc$object_data               = 'OBJECT_DATA                  ',
o  4117     fsc$object_library            = 'OBJECT_LIBRARY               ',
o  4118     fsc$screen_form               = 'SCREEN_FORM                  ',
o  4119     fsc$source_map                = 'SOURCE_MAP                   ',
o  4120     fsc$unknown_contents          = 'UNKNOWN                      ';
o  4121 ?? FMT (FORMAT := ON) ??
o  4122
o  4123
o  4124   CONST
o  4125     osc$max_name_size = 31,
o  4126     osc$null_name = '                               ';
o  4127
o  4128   TYPE
o  4129     ost$name_size = 1 .. osc$max_name_size;
o  4130
o  4131   TYPE
o  4132     ost$name = string (osc$max_name_size);
o  4133
o  4134
o  4135
o  4136 {   Use constants in this deck with amp$open.
o  4137 {   Use constants in deck fsc$file_processor with fsp$open_file.
o  4138
o  4139   CONST
o  4140     amc$unknown_processor = 'UNKNOWN',
o  4141     amc$apl = 'APL',
o  4142     amc$basic = 'BASIC',
o  4143     amc$cobol = 'COBOL',
o  4144     amc$cybil = 'CYBIL',
o  4145     amc$debugger = 'DEBUGGER',
o  4146     amc$fortran = 'FORTRAN',
o  4147     amc$pascal = 'PASCAL',
o  4148     amc$pli = 'PLI',
o  4149     amc$scl = 'SCL',
o  4150     amc$scu = 'SCU',
o  4151     amc$assembler = 'ASSEMBLER',
o  4152     amc$ppu_assembler = 'PPU_ASSEMBLER';
o  4153
o  4154
o  4155   TYPE
o  4156     amt$file_processor = ost$name;
o  4157
```


Decks referenced by selected decks

```
o  4158 {   Use constants in deck amd$file_processor with amp$open}
o  4159 {   Use constants in deck fsc$file_processor with fsp$open_file}
o  4160
o  4161
o  4162 { The following are NOS/VE system conventions for referring to the}
o  4163 { processor of a file:
o  4164
o  4165 ?? FMT (FORMAT := OFF) ??
o  4166  CONST
o  4167     fsc$unknown_processor         = 'UNKNOWN                      ',
o  4168     fsc$Ada                       = 'ADA                          ',
o  4169     fsc$apl                       = 'APL                          ',
o  4170     fsc$assembler                 = 'ASSEMBLER                    ',
o  4171     fsc$basic                     = 'BASIC                        ',
o  4172     fsc$c                         = 'C                            ',
o  4173     fsc$cobol                     = 'COBOL                        ',
o  4174     fsc$cybil                     = 'CYBIL                        ',
o  4175     fsc$debugger                  = 'DEBUGGER                     ',
o  4176     fsc$fortran                   = 'FORTRAN                      ',
o  4177     fsc$lisp                      = 'LISP                         ',
o  4178     fsc$pascal                    = 'PASCAL                       ',
o  4179     fsc$pli                       = 'PLI                          ',
o  4180     fsc$ppu_assembler             = 'PPU_ASSEMBLER                ',
o  4181     fsc$prolog                    = 'PROLOG                       ',
o  4182     fsc$scl                       = 'SCL                          ',
o  4183     fsc$scu                       = 'SCU                          ',
o  4184     fsc$vx                        = 'VX                           ';
o  4185 ?? FMT (FORMAT := ON) ??
o  4186
o  4187
o  4188   CONST
o  4189
o  4190 {   Use the following constants with amp$open and other interfaces that
o  4191 {   externalize a separate file_structure attribute.  Use the constants in
o  4192 {   deck fsc$file_contents with fsp$open_file and other interfaces that
o  4193 {   externalize the file_contents attribute and do not externalize
o  4194 {   file_structure.
o  4195
o  4196     amc$unknown_structure = 'UNKNOWN',
o  4197     amc$data = 'DATA',
o  4198     amc$library = 'LIBRARY',
o  4199     amc$form = 'FORM';
o  4200
o  4201   TYPE
o  4202     amt$file_structure = ost$name;
o  4203
o  4204  TYPE
o  4205     amt$page_format = (amc$continuous_form, amc$burstable_form,
o  4206       amc$non_burstable_form, amc$untitled_form);
o  4207
o  4208  TYPE
o  4209     amt$page_length = 1 .. amc$file_byte_limit;
o  4210
o  4211  CONST
o  4212     amc$max_page_width = 65535;
o  4213
```

Decks referenced by selected decks

```
0  4214   TYPE
0  4215     amt$page_width = 1 .. amc$max_page_width;
0  4216   TYPE
0  4217     amt$compression_procedure_name = amt$entry_point_reference;
0  4218
0  4219   TYPE
0  4220     amt$entry_point_reference = record
0  4221       name: pmt$program_name,
0  4222       object_library: amt$path_name,
0  4223     recend;
0  4224
0  4225   CONST
0  4226     amc$max_path_name_size = 256;
0  4227
0  4228    TYPE
0  4229     amt$path_name = string (amc$max_path_name_size);
0  4230
0  4231
0  4232  TYPE
0  4233    pmt$program_name = ost$name;
0  4234
0  4235   TYPE
0  4236     amt$dynamic_home_block_space = boolean;
0  4237   TYPE
0  4238     amt$hashing_procedure_name = amt$entry_point_reference;
0  4239
0  4240
0  4241    CONST
0  4242     amc$max_index_level = 15;
0  4243
0  4244    TYPE
0  4245     amt$index_levels = 0 .. amc$max_index_level;
0  4246   CONST
0  4247     amc$max_home_blocks = amc$file_byte_limit;
0  4248
0  4249    TYPE
0  4250     amt$initial_home_block_count = 1 .. amc$max_home_blocks;
0  4251
0  4252
0  4253    TYPE
0  4254     amt$line_number = record
0  4255       length: amt$line_number_length,
0  4256       location: amt$line_number_location,
0  4257     recend;
0  4258
0  4259   CONST
0  4260     amc$max_line_number = 6;
0  4261
0  4262    TYPE
0  4263     amt$line_number_length = 1 .. amc$max_line_number;
0  4264   TYPE
0  4265     amt$line_number_location = amt$page_width;
0  4266
0  4267   TYPE
0  4268     amt$loading_factor = 0 .. 100;
0  4269   TYPE
```


Decks referenced by selected decks

```
0  4270     amt$lock_expiration_time = 0 .. 604800000 {milliseconds} ;
0  4271   TYPE
0  4272     amt$log_residence = amt$path_name;
0  4273
0  4274   TYPE
0  4275     amt$logging_options = set of amt$logging_possibilities;
0  4276
0  4277   TYPE
0  4278     amt$logging_possibilities = (amc$enable_parcels, amc$enable_media_recovery,
0  4279       amc$enable_request_recovery);
0  4280
0  4281
0  4282    CONST
0  4283     amc$maximum_record = amc$file_byte_limit;
0  4284
0  4285    TYPE
0  4286     amt$max_record_length = 0 .. amc$maximum_record;
0  4287
0  4288
0  4289    TYPE
0  4290     amt$open_position = (amc$open_no_positioning, amc$open_at_boi,
0  4291       amc$open_at_bop, amc$open_at_eoi);
0  4292
0  4293    TYPE
0  4294     amt$ring_attributes = record
0  4295       r1: ost$valid_ring,
0  4296       r2: ost$valid_ring,
0  4297       r3: ost$valid_ring,
0  4298     recend;
0  4299
0  4300   TYPE
0  4301     amt$statement_identifier = record
0  4302       length: amt$statement_id_length,
0  4303       location: amt$statement_id_location,
0  4304     recend;
0  4305
0  4306   CONST
0  4307     amc$max_statement_id_length = 17;
0  4308
0  4309    TYPE
0  4310     amt$statement_id_length = 1 .. amc$max_statement_id_length;
0  4311   TYPE
0  4312     amt$statement_id_location = amt$page_width;
0  4313
0  4314   TYPE
0  4315     amt$tape_error_options = record
0  4316       perform_failure_recovery: boolean,
0  4317       error_action: amt$tape_error_action,
0  4318     recend;
0  4319
0  4320
0  4321    TYPE
0  4322     amt$tape_error_action = (amc$accept_erroneous_block,
0  4323       amc$ignore_erroneous_block, amc$terminate_file_access);
0  4324
```

Decks referenced by selected decks

```
o  4326 {Permanent File Attributes}
o  4327
o  4328
o  4329   TYPE
o  4330     pft$application_info = string (osc$max_name_size),
o  4331
o  4332     pft$permit_options = (pfc$read, pfc$shorten, pfc$append, pfc$modify,
o  4333       pfc$execute, pfc$cycle, pfc$control),
o  4334
o  4335     pft$usage_options = pfc$read .. pfc$execute,
o  4336     pft$usage_selections = set of pft$usage_options,
o  4337
o  4338     pft$share_options = pfc$read .. pfc$execute,
o  4339     pft$share_selections = set of pft$share_options;
o  4342   TYPE
o  4343     amt$access_level = (amc$physical, amc$record, amc$segment);
o  4344
o  4345   TYPE
o  4346     amt$attribute_source = (amc$undefined_attribute,
o  4347       amc$local_file_information, amc$change_file_attributes, amc$open_request,
o  4348       amc$file_reference, amc$file_command, amc$file_request,
o  4349       amc$add_to_file_description, amc$access_method_default,
o  4350       amc$store_request);
o  4351   TYPE
o  4352     amt$average_record_length = 1 .. amc$maximum_record;
o  4353
o  4354   TYPE
o  4355     amt$block_type = (amc$system_specified, amc$user_specified);
o  4356   TYPE
o  4357     amt$collate_table = array [char] of amt$collation_value;
o  4358
o  4359   TYPE
o  4360     amt$collation_value = 0 .. 255;
o  4361   TYPE
o  4362
o  4363     amt$data_padding = 0 .. 99 {expressed as a percentage} ;
o  4364   TYPE
o  4365     amt$error_exit_procedure = ^procedure (file_identifier:
o  4366       amt$file_identifier;
o  4367       VAR status: ost$status);
o  4368
o  4369
o  4370   CONST
o  4371     amc$max_file_id_ordinal = 4095;
o  4372
o  4373   TYPE
o  4374     amt$file_identifier = record
o  4375       ordinal: amt$file_id_ordinal,
o  4376       sequence: amt$file_id_sequence,
o  4377     recend,
o  4378     amt$file_id_ordinal = 0 .. amc$max_file_id_ordinal,
o  4379     amt$file_id_sequence = 1 .. 4095;
o  4380
o  4381   TYPE
o  4382     ost$status = record
```


Decks referenced by selected decks

```
o  4383       case normal: boolean of
o  4384       = FALSE =
o  4385         condition: ost$status_condition_code,
o  4386         text: ost$string,
o  4387       = TRUE =
o  4388         ,
o  4389       casend,
o  4390     recend;
o  4391
o  4392
o  4393   CONST
o  4394     osc$max_condition = osc$max_status_condition_code;
o  4395
o  4396
o  4397   CONST
o  4398     osc$max_status_condition_code = 0fffffffff(16);
o  4399
o  4400
o  4401   CONST
o  4402     osc$status_parameter_delimiter = $CHAR (31) {Unit Separator} ;
o  4403
o  4404
o  4405   TYPE
o  4406     ost$status_condition = ost$status_condition_code;
o  4407
o  4408
o  4409   TYPE
o  4410     ost$status_condition_code = 0 .. osc$max_status_condition_code;
o  4411
o  4412
o  4413   CONST
o  4414     osc$max_string_size = 256;
o  4415
o  4416   TYPE
o  4417     ost$string_size = 0 .. osc$max_string_size;
o  4418
o  4419   TYPE
o  4420     ost$string_index = 1 .. osc$max_string_size + 1;
o  4421
o  4422   TYPE
o  4423     ost$string = record
o  4424       size: ost$string_size,
o  4425       value: string (osc$max_string_size),
o  4426     recend;
o  4427
o  4428
o  4429   TYPE
o  4430     amt$error_limit = 0 .. 0ffff(16);
o  4431
o  4432
o  4433   CONST
o  4434     amc$max_error_count = 0ffff(16);
o  4435
o  4436   TYPE
o  4437     amt$error_count = 0 .. amc$max_error_count;
o  4438   TYPE
```

Decks referenced by selected decks

```
0  4439      amt$estimated_record_count = integer;
0  4440    CONST
0  4441      amc$access_level = 1,
0  4442      amc$access_mode = 2,
0  4443      amc$application_info = 3,
0  4444      amc$average_record_length = 4,
0  4445      amc$block_type = 5,
0  4446      amc$character_conversion = 6,
0  4447      amc$clear_space = 7,
0  4448      amc$collate_table = 8,
0  4449      amc$collate_table_name = 9,
0  4450      amc$data_padding = 12,
0  4451      amc$embedded_key = 13,
0  4452      amc$error_exit_name = 14,
0  4453      amc$error_exit_procedure = 15,
0  4454      amc$error_limit = 16,
0  4455      amc$error_options = 17,
0  4456      amc$estimated_record_count = 18,
0  4457      amc$file_access_procedure = 19,
0  4458      amc$file_contents = 20,
0  4459      amc$file_length = 21,
0  4460      amc$file_limit = 22,
0  4461      amc$file_organization = 24,
0  4462      amc$file_processor = 25,
0  4463      amc$file_structure = 26,
0  4464      amc$forced_write = 27,
0  4465      amc$global_access_mode = 28,
0  4466      amc$global_file_address = 29,
0  4467      amc$global_file_position = 30,
0  4468      amc$global_file_name = 31,
0  4469      amc$global_share_mode = 32,
0  4470      amc$index_levels = 33,
0  4471      amc$index_padding = 34,
0  4472      amc$internal_code = 35,
0  4473      amc$key_length = 36,
0  4474      amc$key_position = 37,
0  4475      amc$key_type = 38,
0  4476      amc$label_exit_name = 39,
0  4477      amc$label_exit_procedure = 40,
0  4478      amc$label_options = 41,
0  4479      amc$label_type = 42,
0  4480      amc$line_number = 44,
0  4481      amc$max_block_length = 45,
0  4482      amc$max_record_length = 46,
0  4483      amc$message_control = 47,
0  4484      amc$min_block_length = 48,
0  4485      amc$min_record_length = 49,
0  4486      amc$null_attribute = 50,
0  4487      amc$open_position = 51,
0  4488      amc$padding_character = 52,
0  4489      amc$page_format = 53,
0  4490      amc$page_length = 54,
0  4491      amc$page_width = 55,
0  4492      amc$permanent_file = 56,
0  4493      amc$preset_value = 57,
0  4494      amc$record_limit = 59,
```


Decks referenced by selected decks

```
0  4495      amc$record_type = 60,
0  4496      amc$records_per_block = 61,
0  4497      amc$return_option = 62,
0  4498      amc$ring_attributes = 63,
0  4499      amc$statement_identifier = 64,
0  4500      amc$user_info = 66,
0  4501      amc$vertical_print_density = 67,
0  4502      amc$compression_procedure_name = 68,
0  4503      amc$dynamic_home_block_space = 69,
0  4504      amc$hashing_procedure_name = 70,
0  4505      amc$initial_home_block_count = 71,
0  4506      amc$loading_factor = 72,
0  4507      amc$lock_expiration_time = 73,
0  4508      amc$logging_options = 74,
0  4509      amc$log_residence = 75,
0  4510      amc$input_device_classes = 76,
0  4511      amc$output_device_classes = 77,
0  4512      amc$device_class = 78,
0  4513      amc$initial_open = 79,
0  4514      amc$private_read = 80,
0  4515      amc$record_delimiting_character = 81,
0  4516      amc$open_attached_file = 82,
0  4517      amc$open_created_file = 83,
0  4518      amc$open_deleted_data = 84,
0  4519      amc$open_share_modes = 85,
0  4520
0  4521
0  4522  {}
0  4523      amc$concatenated_key_portion = 100,
0  4524      amc$duplicate_keys = 101,
0  4525      amc$group_name = 102,
0  4526      amc$null_suppression = 103,
0  4527      amc$repeating_group = 104,
0  4528      amc$sparse_keys = 105,
0  4529      amc$variable_length_key = 106,
0  4530      amc$actual_block_length = 107,
0  4531      amc$keyed_file_bkup_for_logging = 108,
0  4532
0  4533      amc$alternate_key_count = 129,
0  4534      amc$alternate_key_information = 130,
0  4535      amc$block_count = 131,
0  4536      amc$creation_date = 132,
0  4537      amc$record_count = 133,
0  4538      amc$delete_count = 134,
0  4539      amc$get_count = 135,
0  4540      amc$get_next_count = 136,
0  4541      amc$open_count = 137,
0  4542      amc$put_count = 138,
0  4543      amc$putrep_count = 139,
0  4544      amc$replace_count = 140,
0  4545      amc$index_level_overflow = 141,
0  4546      amc$home_block_count = 142,
0  4547      amc$overflow_block_count = 143,
0  4548      amc$overflow_record_count = 144,
0  4549  {}
0  4550      amc$max_attribute = 511 {01ff(16)} ;
```

Decks referenced by selected decks

```
o  4551
o  4552
o  4553   TYPE
o  4554      amt$file_attribute_keys = 1 .. amc$max_attribute;
o  4555   TYPE
o  4556      amt$file_length = 0 .. amc$file_byte_limit;
o  4557
o  4558   TYPE
o  4559
o  4560      amt$file_organization = (amc$sequential, amc$byte_addressable,
o  4561        amc$indexed_sequential, amc$direct_access, amc$system_key);
o  4562   TYPE
o  4563      amt$file_position = (amc$boi, amc$bop, amc$mid_record, amc$eor, amc$eop,
o  4564        amc$eoi, amc$end_of_key_list);
o  4565   TYPE
o  4566      amt$forced_write = (amc$forced, amc$forced_if_structure_change,
o  4567        amc$unforced);
o  4568   TYPE
o  4569
o  4570      amt$index_padding = 0 .. 99 {expressed as a percentage} ;
o  4571   TYPE
o  4572      amt$internal_code = (amc$as6, amc$as8, amc$ascii, amc$d64,
o  4573        amc$ebcdic, amc$bcd, amc$d63);
o  4574   CONST
o  4575      amc$max_key_length = 255;
o  4576
o  4577    TYPE
o  4578
o  4579      amt$key_length = 1 .. amc$max_key_length;
o  4580   TYPE
o  4581      amt$key_position = 0 .. amc$max_key_position;
o  4582
o  4583   CONST
o  4584      amc$max_key_position = amc$maximum_keyed_record - 1;
o  4585
o  4586   CONST
o  4587      amc$maximum_keyed_record = 65497;
o  4588   TYPE
o  4589
o  4590      amt$key_type = (amc$collated_key, amc$integer_key, amc$uncollated_key);
o  4591   TYPE
o  4592      amt$label_exit_procedure = ^procedure (file_identifier:
o  4593        amt$file_identifier;
o  4594        VAR status: ost$status);
o  4595
o  4596   TYPE
o  4597      amt$label_type = (amc$labelled, amc$non_standard_labelled, amc$unlabelled);
o  4598
o  4599    TYPE
o  4600      amt$local_file_name = ost$name;
o  4601
o  4602
o  4603    CONST
o  4604      amc$mau_length = 2048 (bytes) ;
o  4605   TYPE
o  4606      amt$message_control = set of (amc$trivial_errors, amc$messages,
```


Decks referenced by selected decks

```
o  4607        amc$statistics);
o  4608   TYPE
o  4609      amt$min_block_length = 1 .. amc$maximum_block;
o  4610
o  4611   TYPE
o  4612
o  4613      amt$min_record_length = 0 .. amc$maximum_record;
o  4614
o  4615   TYPE
o  4616      amt$padding_character = char;
o  4617   TYPE
o  4618      amt$record_limit = 1 .. amc$file_byte_limit;
o  4619
o  4620   TYPE
o  4621
o  4622      amt$record_type = (amc$variable {V} , amc$undefined {U} ,
o  4623        amc$ansi_fixed {F} , amc$ansi_spanned {S} , amc$ansi_variable {D} ,
o  4624        amc$trailing_char_delimited {T} );
o  4625   CONST
o  4626      amc$max_records_per_block = 0ffff(16);
o  4627
o  4628    TYPE
o  4629
o  4630      amt$records_per_block = 1 .. amc$max_records_per_block;
o  4631
o  4632    TYPE
o  4633      amt$user_info = string (amc$max_user_info);
o  4634
o  4635
o  4636   CONST
o  4637      amc$max_user_info = 32;
o  4638   TYPE
o  4639      amt$vertical_print_density = 6 .. amc$max_lines_per_inch;
o  4640
o  4641   CONST
o  4642      amc$max_lines_per_inch = 12;
o  4643 {
o  4644 {                 common deck dmdclas
o  4645 {
o  4646
o  4647    TYPE
o  4648      dmt$class = set of dmt$class_member,
o  4649      dmt$class_member = 'A' .. 'Z',
o  4650      dmt$system_class = (dmc$swap_files, dmc$critical_files,
o  4651          dmc$temporary_files, dmc$system_class_spare_1, dmc$system_class_spare_2);
o  4652
o  4653    TYPE
o  4654      dmt$class_ordinal = 0 .. dmc$max_class_ordinal;
o  4655
o  4656    CONST
o  4657      dmc$max_class_ordinal = 63,
o  4658      dmc$default_class = rmc$unspecified_file_class,
o  4659      dmc$swap_file_class = 'C',
o  4660      dmc$transient_segment_class = 'B',
o  4661      dmc$default_class_ordinal = 0;
o  4662
```

Decks referenced by selected decks

```
o  4663
o  4664   CONST
o  4665     rmc$unspecified_file_class = 'A';
o  4666
o  4667 {
o  4668 { dmt$device_file_list_index
o  4669 {
o  4670
o  4671   TYPE
o  4672     dmt$device_file_list_index = 0 .. dmc$max_device_file_list_index;
o  4673
o  4674   CONST
o  4675     dmc$max_device_file_list_index = 65535;
o  4676
o  4677 {
o  4678 {                    common deck dmdfhsh
o  4679 {
o  4680
o  4681   TYPE
o  4682     dmt$file_hash = 0 .. dmc$max_file_hash;
o  4683
o  4684   CONST
o  4685     dmc$max_file_hash = 0ff(16);
o  4686
o  4687   TYPE
o  4688     dmt$global_file_name = ost$binary_unique_name;
o  4689
o  4691
o  4692 { Note: Changing the size (in bytes) of this structure will result in a
o  4693 {       permanent file breakage and a reload will be required.
o  4694
o  4695   TYPE
o  4696     ost$binary_unique_name = packed record
o  4697       serial_number: ost$processor_serial_number,
o  4698       model_number: ost$processor_model_number,
o  4699       year: 1980 .. 2047,
o  4700       month: 1 .. 12,
o  4701       day: 1 .. 31,
o  4702       hour: 0 .. 23,
o  4703       minute: 0 .. 59,
o  4704       second: 0 .. 59,
o  4705       sequence_number: 0 .. 9999999,
o  4706
o  4707 { the field fill pads this value to an even 11 bytes - this is for performance
o  4708
o  4709       fill: 0 .. 7,
o  4710     recend;
o  4711
o  4713 {
o  4714 {                     common deck dmdivsn
o  4715 {
o  4716
o  4717   TYPE
o  4718     dmt$internal_vsn = ost$binary_unique_name;
o  4719
o  4720 {
```


Decks referenced by selected decks

```
o  4721 {                              common deck dmdlofi
o  4722 {
o  4723
o  4724   TYPE   *
o  4725     dmt$access_kind = (dmc$read_access, dmc$write_access),
o  4726     dmt$lock_file_status = (dmc$file_locked_for_caller, dmc$file_already_busy),
o  4727 dmt$write_lock = (dmc$no_write_lock,dmc$write_lock,dmc$write_flush_lock),
o  4728     dmt$locked_file = record
o  4729       case required: boolean of
o  4730       = TRUE =
o  4731         locks: dmt$access_kind,
o  4732         read_lock_count: 0 .. 0ff(16),
o  4733         write_lock: dmt$write_lock,
o  4734       = FALSE =
o  4735         ,
o  4736       casend,
o  4737     recend;
o  4738 {
o  4739 {                  common deck dmdreqv
o  4740 {
o  4741
o  4742   TYPE
o  4743     dmt$request_volume_attribute = record
o  4744       case keyword: dmt$file_attribute_keywords of
o  4745       = dmc$class =
o  4746         class: dmt$class_member,
o  4747       = dmc$class_ordinal =
o  4748         class_ordinal: dmt$class_ordinal,
o  4749       = dmc$recorded_vsn =
o  4750         recorded_vsn: rmt$recorded_vsn,
o  4751       = dmc$file_kind =
o  4752         file_kind: gft$file_kind,
o  4753       = dmc$master_volume_required =
o  4754         master_requested: boolean,
o  4755       = dmc$requested_allocation_size =
o  4756         requested_allocation_size: dmt$allocation_size,
o  4757     =dmc$setname=
o  4758        setname:stt$set_name,
o  4759       casend,
o  4760     recend,
o  4761     dmt$requested_volume = record
o  4762       recorded_vsn: rmt$recorded_vsn,
o  4763       setname: stt$set_name,
o  4764     recend;
o  4765
o  4766
o  4767
o  4768   CONST
o  4769     rmc$external_vsn_size = 6;
o  4770
o  4771   CONST
o  4772     rmc$recorded_vsn_size = 6;
o  4773
o  4774   TYPE
o  4775     rmt$external_vsn = string (rmc$external_vsn_size);
o  4776
```

Decks referenced by selected decks

```
o  4777
o  4778   CONST
o  4779     rmc$unspecified_vsn = '       ';
o  4780
o  4781   TYPE
o  4782     rmt$recorded_vsn = string (rmc$recorded_vsn_size);
o  4783
o  4784
o  4785
o  4786   TYPE
o  4787     rmt$volume_descriptor = record
o  4788       recorded_vsn: rmt$recorded_vsn,
o  4789       external_vsn: rmt$external_vsn,
o  4790     recend;
o  4791
o  4792
o  4793   TYPE
o  4794     rmt$volume_list = array [ * ] of rmt$volume_descriptor;
o  4795
o  4796
o  4797
o  4798 { deck is  STDNAME                                                    }
o  4799
o  4800   TYPE
o  4801     stt$set_name = ost$name,
o  4802     stt$unique_set_name = ost$binary_unique_name;
o  4803
o  4805
o  4806   TYPE
o  4807     ost$unique_name = record
o  4808       case boolean of
o  4809       = TRUE =
o  4810         value: ost$name,
o  4811       = FALSE =
o  4812         dollar_sign: string (1),
o  4813         sequence_number: string (7),
o  4814         p: string (1),
o  4815         processor_model_number: string (1),
o  4816         s: string (1),
o  4817         processor_serial_number: string (4),
o  4818         d: string (1),
o  4819         year: string (4),
o  4820         month: string (2),
o  4821         day: string (2),
o  4822         t: string (1),
o  4823         hour: string (2),
o  4824         minute: string (2),
o  4825         second: string (2),
o  4826       casend,
o  4827     recend;
o  4828
o  4830 { end deck STDNAME }
o  4831 {
o  4832 { dmt$segment_file_information
o  4833 {
o  4834
```


Decks referenced by selected decks

```
o  4835   TYPE
o  4836     dmt$segment_file_info = record
o  4837       asid: ost$asid,
o  4838       preset_value: amt$preset_value,
o  4839       clear_space: boolean,
o  4840       chapter_limit: amt$file_limit,
o  4841       segment_queue_status: dmt$queue_status,
o  4842       usage_count: dmt$usage_count,
o  4843       global_file_name: dmt$global_file_name,
o  4844       allocation_size: dmt$allocation_size,
o  4845       transfer_size: dmt$transfer_size,
o  4846     recend;
o  4847
o  4849 {
o  4850 { dmt$queue_status
o  4851 {
o  4852
o  4853   TYPE
o  4854     dmt$queue_status = (dmc$global_shared, dmc$job_shared, dmc$job_working_set);
o  4855
o  4856 {
o  4857 { dmt$usage_count
o  4858 {
o  4859
o  4860   TYPE
o  4861     dmt$usage_count = 0 .. 0ffff(16);
o  4863
o  4864 {
o  4865 {                   common deck dmdover
o  4866 {
o  4867
o  4868   TYPE
o  4869     dmt$ms_overflow_allowed = boolean,
o  4870     dmt$ms_overflow_indicator = record
o  4871       occured: boolean,
o  4872       byte_address: amt$file_byte_address,
o  4873     recend;
o  4874
o  4875 {
o  4876 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  4877 {   make the appropriate changes in the corresponding display procedures in the
o  4878 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  4879 {
o  4880
o  4881   TYPE
o  4882     sft$file_space_limit_kind = (sfc$no_limit, sfc$perm_file_space_limit,
o  4883           sfc$temp_file_space_limit);
o  4884   CONST
o  4885
o  4886     {Increase this constant to decrease size of level 2 tables
o  4887
o  4888     dmc$level_1_table_size = 4096,
o  4889
o  4890     {Assume 2gb is maximum file size
o  4891
o  4892     dmc$bytes_per_level_2 = 2147483648 DIV dmc$level_1_table_size;
```

Decks referenced by selected decks

```
o  4893
o  4894    TYPE
o  4895      dmt$level_1_index = 0 .. dmc$level_1_table_size - 1,
o  4896
o  4897      {Specified using minimum allocation size -
o  4898      {1 table entry per AU
o  4899
o  4900      dmt$level_2_index = 0 .. dmc$bytes_per_level_2 DIV 16384,
o  4901
o  4902      {A level 1 table consists of an array of offsets.  An offset
o  4903      {refers to either mainframe wired or job fixed, depending on
o  4904      {file residence.  A level 1 table is (kind of) adaptable, in that
o  4905      {the table will expand, not by using CYBIL adaptable pointers but
o  4906      {by using a pointer to fixed type (maximum size) and a current size
o  4907      {value both kept in the fde.
o  4908
o  4909      dmt$level_1_table = array [dmt$level_1_index] of amt$file_byte_address,
o  4910      dmt$level_1_adapt = array [ * ] of amt$file_byte_address,
o  4911
o  4912      {A level 2 table is also (kind of) adaptable, in that it appears
o  4913      {to be an array of a fixed size, but is in fact allocated
o  4914      {adaptably, based on the allocation size.  Each level 2 table
o  4915      {represents the same number of bytes, but it takes fewer table
o  4916      {entries for larger allocation sizes.
o  4917      {No bound information is required as each level 2 table is
o  4918      {initially allocated to it's maximum required size based on allocation size.
o  4919
o  4920      dmt$level_2_table = array [dmt$level_2_index] of dmt$file_allocation_unit,
o  4921      dmt$level_2_adapt = array [ * ] of dmt$file_allocation_unit;
o  4924 {
o  4925 {  DMT$SYSTEM_FILE_ID is an obsolete deck. All references to  it should be replaced
o  4926 {  with references to GFT$SYSTEM_FILE_IDENTIFIER. To ease conversion, however, both names
o  4927 {  will be supported for several releases.
o  4928
o  4929    TYPE
o  4930      dmt$system_file_id = gft$system_file_identifier;
o  4931
o  4932
o  4933 { mmc$eoi_actual - User has made a set segment length request.
o  4934 { mmc$eoi_rounded - This is the default.  Eoi is at a page boundary.
o  4935 { mmc$eoi_uncertain - Page fault processor has assigned extra pages that
o  4936 {                     may or may not have been used.  Eoi is still at the
o  4937 {                     end of the page that faulted.
o  4938
o  4939    TYPE
o  4940      mmt$eoi_state = (mmc$eoi_actual, mmc$eoi_rounded, mmc$eoi_uncertain);
o  4941
o  4942 {Define lock word for COMPARE SWAP operations.}
o  4943
o  4944    TYPE
o  4945      ost$compare_swap_lock = integer,
o  4946      ost$cs_lock = integer, {* * HCS compatibility * * *}
o  4947      ost$signature_lock = record
o  4948        lock_id: ALIGNED [0 MOD 8] integer,
o  4949      recend,
o  4950
```


Decks referenced by selected decks

```
o  4951
o  4952 {Define lock byte for TEST_SET bit operations.}
o  4953
o  4954      ost$byte_lock = packed array [0 .. 7] of boolean;
o  4955
o  4956 {
o  4957 { dmt$active_volume_table_index
o  4958 {
o  4959
o  4960    TYPE
o  4961      dmt$active_volume_table_index = 0 .. ioc$max_unit_number;
o  4962
o  4963
o  4964 { DECK: IOT$LOGICAL_UNIT
o  4965
o  4966    CONST
o  4967      ioc$max_unit_number = 0ffff(16);
o  4968
o  4969    TYPE
o  4970      iot$logical_unit = 0 .. ioc$max_unit_number;
o  4971 {
o  4972 { dmt$subfile_index
o  4973 {
o  4974
o  4975    TYPE
o  4976      dmt$subfile_index = 0 .. 255;
o  4977
o  4978
o  4979 { Define transfer unit size (in bytes).
o  4980
o  4981    TYPE
o  4982      gft$transfer_unit_size = 0 .. 10000000;
o  4983
o  4984    TYPE
o  4985      gft$signature_lock = RECORD
o  4986        locked:  boolean,
o  4987        count: 0 .. 255,
o  4988        gtid: ost$global_task_id,
o  4989        p_register: integer,              {!!!!##### debug only}
o  4990        p_register_2: integer,            {!!!!##### debug only}
o  4991      RECEND;
o  4992
o  4993
o  4994 { Define maximum number of instances of segment access OPENs for a file.
o  4995
o  4996    TYPE
o  4997      gft$open_count = 0 .. 0ffffffff(16);
o  4998
o  4999
o  5000 { Define byte used for monitor mode interlocks. The left most bit is used by
o  5001 { #TEST_SET_BIT as the interlock bit. If monitor is compiled with debug code active,
o  5002 { then bits 1 thru 7 will contain the CPU# that has the interlock.
o  5003
o  5004    TYPE
o  5005      mtt$monitor_interlock = PACKED RECORD
o  5006        CASE boolean OF
```

Decks referenced by selected decks

```
o  5007         = FALSE =
o  5008           byte: 0 .. 255,
o  5009         = TRUE =
o  5010           locked: boolean,
o  5011           id: 0 .. 7f(16),
o  5012         CASEND,
o  5013       RECEND;
o  5014
o  5015 { This represents the values attainable by the cyber 180 microsecond clock.
o  5016
o  5017   TYPE
o  5018     ost$free_running_clock = 0 .. osc$free_running_clock_maximum;
o  5019
o  5020   CONST
o  5021     osc$free_running_clock_maximum = 0fffffffffff(16);
o  5022
o  5023
o  5024   TYPE
o  5025     pmt$initialization_value = (pmc$initialize_to_zero,
o  5026       pmc$initialize_to_alt_ones, pmc$initialize_to_indefinite,
o  5027       pmc$initialize_to_infinity);
o  5028
o  5029
o  5030 { TMDTQLK - contains the type declaration for task queue link.
o  5031
o  5032   TYPE
o  5033     tmt$task_queue_link = record
o  5034       head: ost$task_index,
o  5035       tail: ost$task_index,
o  5036     recend;
o  5037
o  5038
o  5039 { Define 180 job and task cpu priorities. For purposes of sharing the CPU in
o  5040 { dual state, the equivalent 170 priorities are also shown in comments.
o  5041
o  5042   TYPE
o  5043     jmt$dispatching_priority = 0 .. jmc$max_dispatching_priority,
o  5044     jmt$user_dispatching_priority = jmc$priority_p1 .. jmc$priority_p8,
o  5045     jmt$system_dispatching_priority = jmc$priority_p10 .. jmc$priority_p14,
o  5046     jmt$dispatching_priority_bias = -jmc$max_dispatching_priority ..
o  5047           jmc$max_dispatching_priority;
o  5048
o  5049
o  5050
o  5051 {  jmc$dp_conversion is used to change a dispatching priority to a bit number
o  5052 {  that is used in dispatching priority sets.  Because task switch uses
o  5053 {  #unchecked_conversion to select elements from the dispatching control sets
o  5054 {  rather than the CYBIL IN set manipulator, the bit stored in the sets has to
o  5055 {  be a "converted" dispatching priority.  The leftmost bit in the set
o  5056 {  represents the highest dispatching priority.  The following table relates
o  5057 {  priorities and bits:
o  5058 {-------------------------------------------------------------------------------
o  5059 {     Dispatching                          Dispatching       Conversion
o  5060 {      Priority          Dispatching         Priority         Function
o  5061 {        Name             Priority           Bit             Returns
o  5062 {     -----------       -----------       -----------       ----------
```


Decks referenced by selected decks

```
o  5063 {         P1                  2                 14                  2
o  5064 {         P2                  3                 13                  3
o  5065 {         P3                  4                 12                  4
o  5066 {         P4                  5                 11                  5
o  5067 {         P5                  6                 10                  6
o  5068 {         P6                  7                  9                  7
o  5069 {         P7                  8                  8                  8
o  5070 {         P8                  9                  7                  9
o  5071 {         P9                 10                  6                 10
o  5072 {         P10                11                  5                 11
o  5073 {         P11                12                  4                 12
o  5074 {         P12                13                  3                 13
o  5075 {         P13                14                  2                 14
o  5076 {         P14                15                  1                 15
o  5077 {-------------------------------------------------------------------------------
o  5078
o  5079   CONST
o  5080
o  5081     jmc$dp_conversion = 16,
o  5082
o  5083     jmc$max_dispatching_priority = 15,
o  5084     jmc$min_dispatching_priority = 2, { Minimum dispatchable priority
o  5085     jmc$null_dispatching_priority = 0;
o  5086
o  5087 { The following constants define the range of values permitted on SCL
o  5088 { parameter definitions. The values are specified internally as one number
o  5089 { greater than the external values.  The following internal range of 2 .. 11
o  5090 { is specified as 1 .. 10 externally.
o  5091
o  5092   CONST
o  5093     jmc$lowest_dispatching_priority = 2,
o  5094     jmc$highest_dispatch_priority = 11;
o  5095
o  5096   CONST
o  5097     jmc$priority_p1 = 2, { 170 - 10 }
o  5098     jmc$priority_p2 = 3, { 170 - 10 }
o  5099     jmc$priority_p3 = 4, { 170 - 20 }
o  5100     jmc$priority_p4 = 5, { 170 - 20 }
o  5101     jmc$priority_p5 = 6, { 170 - 30 }
o  5102     jmc$priority_p6 = 7, { 170 - 30 }
o  5103     jmc$priority_p7 = 8, { 170 - 40 }
o  5104     jmc$priority_p8 = 9, { 170 - 40 }
o  5105     jmc$priority_p9 = 10, { 170 - 50 }
o  5106     jmc$priority_p10 = 11, { 170 - 50 }
o  5107     jmc$priority_p11 = 12, { 170 - 60 }
o  5108     jmc$priority_p12 = 13, { 170 - 60 }
o  5109     jmc$priority_p13 = 14, { 170 - 70 }
o  5110     jmc$priority_p14 = 15; { 170 - 70 }
o  5111
o  5112
o  5113 { Deck:  DFT$MAINFRAME_SET
o  5114
o  5115   TYPE
o  5116     dft$mainframe_set = set of 1 .. dfc$max_number_of_mainframes;
o  5117
o  5118 { DECK: DFC$ESM_ALLOCATION_CONSTANTS
```

Decks referenced by selected decks

```
o  5119
o  5120 { These values are in ESM/STORNET memory words.
o  5121   CONST
o  5122     dfc$division_overwrite_words = 16,
o  5123     dfc$esm_maintenance_buf_size = 1000,
o  5124     dfc$max_esm_memory_size = 16777216,
o  5125     dfc$min_esm_memory_size = 1048576;
o  5126
o  5127 { These values are also declared as constants in File Server's PP driver.
o  5128   CONST
o  5129     dfc$esm_memory_base_shift = 100(8),
o  5130     dfc$esm_division_chwrds_shift = 100(8),
o  5131     dfc$max_number_of_mainframes = 8,
o  5132     dfc$max_esm_divisions = 16,
o  5133     dfc$max_rma_list_entries = 64,
o  5134     dfc$header_record_bytes = 24,
o  5135     dfc$command_record_bytes = 4096;
o  5136
o  5137   CONST
o  5138     dfc$max_data_record_bytes = 262144,
o  5139     dfc$min_data_record_bytes = 16384;
o  5140
o  5141 { These values are expressed in number of 60 bit ESM words rounded up by 10(8).
o  5142   CONST
o  5143     dfc$esm_command_record_size = ((((dfc$command_record_bytes * 8) DIV 60) + 7) DIV 8) * 8,
o  5144     dfc$esm_header_record_size = ((((dfc$header_record_bytes * 8) DIV 60) + 7) DIV 8) * 8;
o  5145
o  5146 { These values are expressed as number of 60 bit ESM words.
o  5147   CONST
o  5148     dfc$max_driver_formed_esm_adrs = 37777777(8),
o  5149     dfc$min_esm_division_size =
o  5150      (((((((dfc$min_data_record_bytes * 8) DIV 60) + 7 + dfc$division_overwrite_words) DIV 8) * 8) +
o  5151            (dfc$esm_command_record_size + dfc$esm_header_record_size)) + dfc$esm_memory_base_shift-1)
o  5152              DIV dfc$esm_memory_base_shift) * dfc$esm_memory_base_shift,
o  5153     dfc$max_esm_memory_base = ((dfc$max_esm_memory_size - dfc$min_esm_division_size
o  5154                                 - dfc$esm_maintenance_buf_size) DIV 1000(8)) * 1000(8);
o  5155
o  5156
o  5157
o  5158
o  5159
o  5160   TYPE
o  5161     sft$counter = integer;
o  5162
o  5163 { * * * * * common deck JMDAJLO: Active Job List Ordinal * * * * * }
o  5164
o  5165 TYPE
o  5166   jmt$ajl_ordinal = 0 .. jmc$max_ajl_ord;
o  5167
o  5168 { * * * * * end of deck jmdajlo. . . . . * * * * * }
o  5169
o  5170
o  5171   CONST
o  5172     jmc$max_active_jobs = jmc$max_ajl_ord - jmc$reserved_ajls,
o  5173     jmc$max_ajl_ord = 255,
o  5174     jmc$max_ijl_ord = jmc$kjl_maximum_entries,
```


Decks referenced by selected decks

```
o  5175     jmc$max_kjl_ord = jmc$kjl_maximum_entries,
o  5176     jmc$max_kol_ord = jmc$kol_maximum_entries,
o  5177     jmc$reserved_ajls = 5;
o  5178
o  5179
o  5180   CONST
o  5181     jmc$kjl_maximum_entries = jmc$maximum_job_count;
o  5182
o  5183
o  5184 { This constant represents the maximum number of jobs that can be known
o  5185 {   by the NOS/VE operating system.
o  5186
o  5187   CONST
o  5188     jmc$maximum_job_count = 65535;
o  5189
o  5190   CONST
o  5191     jmc$kol_maximum_entries = jmc$maximum_output_count;
o  5192
o  5193
o  5194 { This constant represents the maximum number of output files
o  5195 {   that NOS/VE is capable of knowing at one time.
o  5196
o  5197   CONST
o  5198     jmc$maximum_output_count = 65535;
o  5199
o  5200 {
o  5201 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  5202 {   make the appropriate changes in the corresponding display procedures in the
o  5203 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  5204 {
o  5205
o  5206   TYPE
o  5207     jmt$ijl_dispatching_control = record
o  5208       dispatching_control_index: jmt$dispatching_control_index,
o  5209       dispatching_priority: jmt$dispatching_priority,
o  5210       user_requested_dispatching_prio: jmt$dispatching_priority,
o  5211       operator_set_dispatching_prio: jmt$dispatching_priority,
o  5212       service_remaining: ost$free_running_clock,
o  5213       cp_service_at_class_switch: integer,
o  5214     RECEND;
o  5215
o  5216
o  5217   TYPE
o  5218     jmt$dispatching_control= ARRAY [jmt$dispatching_control_index] OF
o  5219           jmt$dispatching_controls,
o  5220
o  5221     jmt$dispatching_controls = RECORD
o  5222       set_defined: boolean,
o  5223       dispatching_priority: jmt$dispatching_priority,
o  5224       service_limit: ost$free_running_clock, { microseconds
o  5225       dispatching_timeslice: jmt$time_slice_values,
o  5226     RECEND;
o  5227
o  5228 { The following constants define the range of values permitted on SCL
o  5229 { parameter definitions and the internal representation for the keyword,
o  5230 { UNLIMITED, for the service limit element of the record.  Service limit
```

Decks referenced by selected decks

```
o  5231 { has a unit of microseconds internally but is specified in milliseconds
o  5232 { externally.
o  5233
o  5234   CONST
o  5235     jmc$lowest_service_limit = 1000, { microseconds
o  5236     jmc$highest_service_limit = 3600000000, { microseconds
o  5237     jmc$dc_maximum_service_limit = 0fffffffffff(16); { microseconds
o  5238
o  5239   TYPE
o  5240     jmt$time_slice_values = RECORD
o  5241       minor: jmt$task_time_slice,
o  5242       major: jmt$task_time_slice,
o  5243     RECEND;
o  5244
o  5245
o  5246   CONST
o  5247     jmc$min_dispatching_control = 1,
o  5248     jmc$max_dispatching_control = 5;
o  5249
o  5250   TYPE
o  5251     jmt$dispatching_control_index = jmc$min_dispatching_control ..
o  5252           jmc$max_dispatching_control;
o  5253
o  5254
o  5255
o  5256
o  5257
o  5258 { The task time slice has a unit of microseconds.
o  5259
o  5260   TYPE
o  5261     jmt$task_time_slice = ost$task_time_slice;
o  5262
o  5263 { The following constants define the range of values permitted on SCL
o  5264 { parameter definitions.
o  5265
o  5266   CONST
o  5267     jmc$lowest_task_time_slice = 1,
o  5268     jmc$highest_task_time_slice = 100;
o  5269
o  5270
o  5271   CONST
o  5272     osc$task_time_slice_maximum = 0ffffffff(16);
o  5273
o  5274   TYPE
o  5275     ost$task_time_slice = 0 .. osc$task_time_slice_maximum;
o  5276
o  5277
o  5278 { NOTE:  The ijl entry statuses are order dependant for swap direction checking.  The constants
o  5279 { jmc$ies_swapped_in and jmc$ies_swapped_job are defined for swap direction checking in swapper.
o  5280 { If a job's entry status is less than jmc$ies_swapped_out, then the swap direction is IN.
o  5281 { If the entry status is greater than jmc$ies_swapped_in, then the swap direction is OUT.
o  5282
o  5283   TYPE
o  5284     jmt$ijl_entry_status = (jmc$ies_entry_free,
o  5285                             jmc$ies_job_terminating,
o  5286                             jmc$ies_job_in_memory_non_swap,
```


Decks referenced by selected decks

```
o  5287                             jmc$ies_job_in_memory,
o  5288                             jmc$ies_swapin_in_progress,
o  5289                             jmc$ies_job_swapped,
o  5290                             jmc$ies_operator_force_out,
o  5291                             jmc$ies_system_force_out,
o  5292                             jmc$ies_job_damaged,
o  5293                             jmc$ies_ready_task,
o  5294                             jmc$ies_swapin_candidate);
o  5295
o  5296   CONST
o  5297     jmc$ies_swapped_in = jmc$ies_swapin_in_progress,
o  5298     jmc$ies_swapped_out = jmc$ies_job_swapped;
o  5299
o  5300   TYPE
o  5301     jmt$ijl_service_class_stats = record
o  5302       cp_time: ost$cp_time,
o  5303       page_faults: jmt$ijl_page_stats,
o  5304       swapouts: jmt$ijl_swap_counts,
o  5305     recend,
o  5306
o  5307     jmt$ijl_page_stats = record
o  5308       disk: jmt$ijl_page_fault_count,
o  5309       reclaimed: jmt$ijl_page_fault_count,
o  5310       assigned: jmt$ijl_page_fault_count,
o  5311     recend,
o  5312
o  5313     jmt$ijl_page_fault_count = 0 .. 0ffffffff(16);
o  5314
o  5315
o  5316   TYPE
o  5317     jmt$ijl_swap_counts = record
o  5318       long_wait: jmt$ijl_swap_count,
o  5319       job_mode: jmt$ijl_swap_count,
o  5320     recend,
o  5321
o  5322     jmt$ijl_swap_count = 0 .. 0ffffff(16);
o  5323
o  5324   TYPE
o  5325     jmt$kjl_index = 0 .. jmc$kjl_maximum_entries;
o  5326
o  5327   CONST
o  5328     jmc$kjl_undefined_index = 0;
o  5329
o  5330 {
o  5331 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  5332 {   make the appropriate changes in the corresponding display procedures in the
o  5333 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  5334 {
o  5335
o  5336   TYPE
o  5337     jmt$queue_file_ijl_information = record
o  5338       job_abort_disposition: jmt$job_abort_disposition,
o  5339       job_recovery_disposition: jmt$job_recovery_disposition,
o  5340       input_file_location: jmt$input_file_location,
o  5341     recend;
o  5342
```

Decks referenced by selected decks

```
o  5343
o  5344    TYPE
o  5345      jmt$input_file_location = 0 .. 255;
o  5346
o  5347    CONST
o  5348      jmc$ifl_no_input_file_exists = 0,
o  5349      jmc$ifl_system_input_queue = 1,
o  5350      jmc$ifl_store_and_forward_queue = 2,
o  5351      jmc$ifl_login_family_queue = 3;
o  5352
o  5353    TYPE
o  5354      jmt$job_abort_disposition = (jmc$restart_on_abort, jmc$terminate_on_abort);
o  5355
o  5356    TYPE
o  5357      jmt$job_recovery_disposition = (jmc$continue_on_recovery,
o  5358            jmc$restart_on_recovery, jmc$terminate_on_recovery);
o  5359
o  5360
o  5361    TYPE
o  5362      jmt$service_accumulator = 0 .. jmc$service_accumulator_maximum;
o  5363
o  5364    CONST
o  5365      jmc$service_accumulator_maximum = 0ffffffffffff(16);
o  5366
o  5367  { The following constants define the range of values permitted on SCL
o  5368  { parameter definitions and the internal representation for the keyword,
o  5369  { UNLIMITED.
o  5370
o  5371    CONST
o  5372      jmc$lowest_service_accumulator = 0,
o  5373      jmc$highest_service_accumulator = 100000000000,
o  5374      jmc$unlimited_service_accum = jmc$highest_service_accumulator +
o  5375            jmc$unlimited_offset;
o  5376
o  5377
o  5378  { Define the numerical offsets which represent the keywords, UNLIMITED,
o  5379  { UNSPECIFIED, REQUIRED, and SYSTEM_DEFAULT, used on SCL command parameter
o  5380  { values for scheduling attributes.  These offsets are used by the
o  5381  { Manage_Active_Scheduling Utility to facilitate the coding required to
o  5382  { translate the command parameter values to their internal format.
o  5383
o  5384
o  5385    CONST
o  5386      jmc$unlimited_offset = 1,
o  5387      jmc$unspecified_offset = 2,
o  5388      jmc$required_offset = 3,
o  5389      jmc$system_default_offset = 4,
o  5390      jmc$keyword_offset_maximum = jmc$system_default_offset;
o  5391
o  5392
o  5393  { This deck defines the type for service classes.  Any time the service
o  5394  { class table (jmv$service_class_table_p) needs to be scanned, the scan
o  5395  { should be from jmc$system_service_class (the first defined class) to
o  5396  { jmv$max_service_class_in_use (the index of the highest defined class).
o  5397
o  5398    CONST
```


Decks referenced by selected decks

```
o  5399      jmc$null_service_class = 0,
o  5400      jmc$unspecified_service_class = jmc$null_service_class,
o  5401      jmc$system_service_class = 1,
o  5402      jmc$maintenance_service_class = 2,
o  5403      jmc$unassigned_service_class = 3,
o  5404      jmc$lowest_site_service_class = 4,
o  5405      jmc$minimum_service_classes = 3,
o  5406      jmc$maximum_service_classes = 255;
o  5407
o  5408    TYPE
o  5409      jmt$service_class_index = 0 .. jmc$maximum_service_classes;
o  5410
o  5411    TYPE
o  5412      jmt$swapout_reasons = (jmc$sr_null,
o  5413                             jmc$sr_operator_request,
o  5414                             jmc$sr_thrashing,
o  5415                             jmc$sr_lower_priority,
o  5416                             jmc$sr_idling_system_swapout,
o  5417                             jmc$sr_long_wait,
o  5418                             jmc$sr_memory_reserve_request,
o  5419                             jmc$sr_idle_dispatching,
o  5420                             jmc$sr_job_damaged);
o  5421
o  5422
o  5423  { The system supplied name is of the form $MMMM_NNNN_SSS_CCCC }
o  5424
o  5425    TYPE
o  5426      jmt$system_supplied_name = string (jmc$system_supplied_name_size);
o  5427
o  5428    CONST
o  5429      jmc$system_supplied_name_size = 19,
o  5430      jmc$long_ssn_size = 9,
o  5431      jmc$short_ssn_size = 5,
o  5432      jmc$full_system_supplied_name = '$0000_0000_AAA_0000',
o  5433      jmc$long_system_supplied_name = '$AAA_0000',
o  5434      jmc$short_system_supplied_name = '$0000',
o  5435      jmc$blank_system_supplied_name = '                   ';
o  5436  {
o  5437  { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  5438  {   make the appropriate changes in the corresponding display procedures in the
o  5439  {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  5440  {
o  5441   TYPE
o  5442      jst$ijl_swap_queue_link = record
o  5443        queue_id: jst$ijl_swap_queue_id,
o  5444        backward_link: jmt$ijl_ordinal,
o  5445        forward_link: jmt$ijl_ordinal,
o  5446      recend,
o  5447
o  5448      jst$ijl_swap_queue_id = (jsc$isqi_null, jsc$isqi_swapping, jsc$isqi_swapped_io_not_init,
o  5449        jsc$isqi_swapped_io_cannot_init, jsc$isqi_swapped_io_completed, jsc$isqi_swapped_out),
o  5450
o  5451      jst$swapped_but_still_in_memory = jsc$isqi_swapped_io_not_init .. jsc$isqi_swapped_io_completed;
o  5452
o  5453
o  5454    TYPE
```

Decks referenced by selected decks

```
0  5455      mmt$memory_reserve_request = RECORD
0  5456        swapout_job: boolean,
0  5457        requested_page_count: mmt$page_frame_index,
0  5458        reserved_page_count: mmt$page_frame_index,
0  5459      RECEND;
0  5460
0  5463    TYPE
0  5464      iot$io_error = (ioc$no_error, ioc$allocate_file_space, ioc$media_error, ioc$unrecovered_error,
0  5465        ioc$unrecovered_error_unit_down, ioc$server_allocation_error, ioc$server_has_terminated,
0  5466        ioc$error_on_init, ioc$unit_down_on_init);
0  5467
0  5468    TYPE
0  5469      jmt$detached_job_wait_time = 0..jmc$detached_job_wait_time_max;
0  5470
0  5471    CONST
0  5472      jmc$detached_job_wait_time_max = jmc$highest_det_job_wait_time +
0  5473            jmc$unlimited_offset;
0  5474
0  5475  { The following constants define the range of values permitted on SCL
0  5476  { parameter definitions and the internal representation for the keyword,
0  5477  { UNLIMITED.
0  5478
0  5479    CONST
0  5480
0  5481      jmc$lowest_det_job_wait_time = 0,
0  5482      jmc$highest_det_job_wait_time = 36000,
0  5483      jmc$unlimited_det_job_wait_time = jmc$highest_det_job_wait_time +
0  5484            jmc$unlimited_offset,
0  5485
0  5486
0  5487    TYPE
0  5488      jmt$job_system_id = jmt$kjl_index;
0  5489
0  5490
0  5491    TYPE
0  5492      jmt$user_supplied_name = ost$name;
0  5493
0  5494
0  5495    TYPE
0  5496      jmt$working_set_size = 0 .. jmc$working_set_size_maximum;
0  5497
0  5498    CONST
0  5499      jmc$working_set_size_maximum = jmc$highest_working_set_size +
0  5500            jmc$keyword_offset_maximum;
0  5501
0  5502  { The following constants define the range of values permitted on SCL
0  5503  { parameter definitions and the internal representation for the keywords,
0  5504  { UNLIMITED, UNSPECIFIED, REQUIRED, and SYSTEM_DEFAULT.
0  5505
0  5506    CONST
0  5507      jmc$lowest_working_set_size = 20,
0  5508      jmc$highest_working_set_size = 65000,
0  5509      jmc$unlimited_working_set_size = jmc$highest_working_set_size +
0  5510            jmc$unlimited_offset,
0  5511      jmc$unspecified_work_set_size = jmc$highest_working_set_size +
0  5512            jmc$unspecified_offset,
```


Decks referenced by selected decks

```
0  5513      jmc$required_working_set_size = jmc$highest_working_set_size +
0  5514            jmc$required_offset,
0  5515      jmc$system_default_work_set_siz = jmc$highest_working_set_size +
0  5516            jmc$system_default_offset;
0  5517
0  5518
0  5519    CONST
0  5520      osc$aging_interval_maximum = 0ffffffff(16);
0  5521
0  5522    TYPE
0  5523      ost$aging_interval = 0..osc$aging_interval_maximum;
0  5524
0  5525    TYPE
0  5526      ost$user_identification = record
0  5527        user: ost$user_name,
0  5528        family: ost$family_name,
0  5529      recend,
0  5530
0  5531      ost$user_name = ost$name,
0  5532
0  5533      ost$family_name = ost$name;
0  5534
0  5535
0  5536
0  5537    { Sense switch definition
0  5538
0  5539    TYPE
0  5540      pmt$sense_switches = set OF 1 .. 8;
0  5541
0  5542  {Page Table definitions - (hardware defined)]
0  5543
0  5544    CONST
0  5545      osc$max_page_frames = 0ffff(16),
0  5546      osc$max_page_table_entries = 131072 * 2;
0  5547
0  5548    TYPE
0  5549      ost$page_table_index = 0 .. osc$max_page_table_entries - 1,
0  5550
0  5551      ost$page_id = packed record
0  5552        asid: 0 .. 0ffff(16),
0  5553        pagenum: 0 .. 3fffff(16),
0  5554      recend,
0  5555
0  5556      ost$page_table_entry = packed record
0  5557        v: boolean,
0  5558        c: boolean,
0  5559        u: boolean,
0  5560        m: boolean,
0  5561        pageid: ost$page_id,
0  5562        rma: 0 .. 3fffff(16),
0  5563      recend,
0  5564
0  5565      ost$page_table = array [ost$page_table_index] of ost$page_table_entry;
0  5566
0  5567    SECTION
0  5568      oss$job_paged_literal: READ;
```

Decks referenced by selected decks

```
o  5569 {  *copyc mmc$default_sdt_length
o  5570
o  5571   TYPE
o  5572     pmt$condition_identifier = 0 .. 255;
o  5573 {!!!!!! users of this deck should change to copy SYT$MONITOR_STATUS.
o  5574
o  5575
o  5576 { Asynchronous request parameter: used by all NOS/180 requests that }
o  5577 { can be performed asynchronously to indicate whether the caller }
o  5578 { wishes to execute the request synchronously or asynchronously. }
o  5579
o  5580 TYPE
o  5581   ost$wait = (osc$wait, osc$nowait);
o  5582 { Define how access may proceed to the segment.
o  5583
o  5584   TYPE
o  5585     mmt$segment_access_state = (
o  5586       mmc$sas_allow_access,      { - The normal state. All access is allowed.
o  5587       mmc$sas_inhibit_access,    { - Indicates that any access
o  5588                                  { to this segment from the task should wait.
o  5589                                  { This happens because the job needs to perform
o  5590                                  { recovery for this file.
o  5591       mmc$sas_terminate_access);{ - Indicates that a segment access condition
o  5592                                  { should be raised On any access to the segment.
o  5593
o  5594
o  5595     TYPE
o  5596       mmt$lock_segment_status = (mmc$lss_none, mmc$lss_queued_for_lock_user, mmc$lss_queued_for_lock_r3,
o  5597               mmc$lss_lock_for_read_user, mmc$lss_lock_for_read_r3, mmc$lss_lock_for_write_user,
o  5598               mmc$lss_lock_for_write_r3);
o  5599 {
o  5600 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  5601 {   make the appropriate changes in the corresponding display procedures in the
o  5602 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  5603 {
o  5604
o  5605   TYPE
o  5606     mmt$segment_reservation_state = (mmc$srs_not_reserved, mmc$srs_reserved,
o  5607          mmc$srs_reserved_shared_stack);
o  5608 {
o  5609 { NOTE: If TYPE declarations or record fields are added/changed/deleted, please
o  5610 {   make the appropriate changes in the corresponding display procedures in the
o  5611 {   module(s) for the System Core Debugger: SYM$DEBUG, SYM$DEBUG1
o  5612 {
o  5613
o  5614
o  5615     TYPE
o  5616       mmt$shadow_segment_kind = (mmc$ssk_none, mmc$ssk_read_write_file, mmc$ssk_read_only_file,
o  5617          mmc$ssk_read_only_trans_file, mmc$ssk_segment_number);
o  5618
o  5619 { TYPE deck MTT$SCB_HARDWARE_STATUS
o  5620
o  5621 { WARNING!!!!!
o  5622 { If the following TYPE (subrange) is modified, then the TYPE
o  5623 { mtt$scb_trick_variant_record MUST be changed.  See the comments
o  5624 { above the variant record definition further on.
```


Decks referenced by selected decks

```
o  5625
o  5626   TYPE
o  5627     mtt$scb_hardware_status_options = (mtc$scb_short_warning_step,
o  5628           mtc$scb_hardware_failure_step, mtc$scb_long_warning_idle,
o  5629           mtc$scb_hardware_failure_idle, mtc$scb_170_status);
o  5630
o  5631   TYPE
o  5632     mtt$scb_hardware_status_actions = (mtc$scb_hsa_set, mtc$scb_hsa_clear);
o  5633
o  5634   TYPE
o  5635     mtt$scb_hardware_status_count = 0 .. mtc$scb_max_hardware_status,
o  5636
o  5637     mtt$scb_hardware_status = ARRAY [mtt$scb_hardware_status_options] OF
o  5638           mtt$scb_hardware_status_count;
o  5639
o  5640   TYPE
o  5641     mtt$scb_hardware_status_msg = RECORD
o  5642       message_read: boolean,
o  5643       message: dpt$top_line_message,
o  5644     RECEND,
o  5645
o  5646     mtt$scb_hardware_status_msgs = ARRAY [mtt$scb_hardware_status_options]
o  5647           OF mtt$scb_hardware_status_msg;
o  5648
o  5649 { WARNING!!!!!
o  5650 { The following record MUST be modified if the TYPE mtt$scb_hardware_status
o  5651 { changes.  The ERRORS_PRESENT variant must be the same length as the
o  5652 { HARDWARE_STATUS variant; i.e. there must be as many bytes in the
o  5653 { ERRORS_PRESENT field as there are ordinals in the TYPE
o  5654 { mtt$scb_hardware_status_options (above).
o  5655
o  5656   TYPE
o  5657     mtt$scb_trick_variant_record = RECORD
o  5658       CASE 0..1 OF
o  5659       = 0 =
o  5660         hardware_status: mtt$scb_hardware_status,
o  5661       = 1 =
o  5662         errors_present: 0 .. 0ffffffffff(16),
o  5663       CASEND,
o  5664     RECEND;
o  5665
o  5666
o  5667   TYPE
o  5668     dpt$top_line_message = string (dpc$top_line_message_size);
o  5669
o  5671
o  5672   CONST
o  5673     dpc$top_line_message_size = (dpc$console_row_size - 9);
o  5674
o  5675
o  5676   { This constant describes the length, in characters, of a row on the system console.
o  5677
o  5678   CONST
o  5679     dpc$console_row_size = 80;
o  5681
o  5682 { CONST deck MTC$SCB_MAX_HARDWARE_STATUS
```

Decks referenced by selected decks

```
o  5683
o  5684    CONST
o  5685      mtc$scb_max_hardware_status = 0ff(16);
o  5686
o  5687
o  5688    TYPE
o  5689      mtt$system_step_update_request = (mtc$unstepped_system, mtc$stepped_system),
o  5690      mtt$system_idle_update_request = (mtc$running_system, mtc$idled_system);
o  5691
o  5692  { TYPE declaration: ost$processor_id_set
o  5693
o  5694   TYPE
o  5695     ost$processor_id_set = SET OF ost$processor_id;
o  5696
o  5697
o  5698  { TYPE declaration: ost$processor_id
o  5699
o  5700    TYPE
o  5701      ost$processor_id = 0 .. osc$maximum_processor_id;
o  5702
o  5703
o  5704    CONST
o  5705      osc$maximum_processor_id = 7;
o  5706
o  5707
o  5708
o  5709  { * * * * WARNING - This record is contained in the SCB and is referenced * * *
o  5710  { * * * *           from assembly language trap handlers. Dont change     * * *
o  5711  { * * * *           this record without understanding the implications.   * * *
o  5712
o  5713    TYPE
o  5714      ost$vector_simulation_control = record
o  5715        vector_simulation_attribute: pmt$vector_simulation,
o  5716        vector_divide_degraded: ost$processor_id_set,
o  5717        all_vector_divides_degraded: boolean,
o  5718      recend;
o  5719
o  5720    TYPE
o  5721      pmt$vector_simulation = 0 .. 255;
o  5722
o  5723    CONST
o  5724      pmc$vectors_simulated = 0,
o  5725      pmc$vectors_suspended = 1,
o  5726      pmc$vectors_aborted = 2;
o  5727
o  5728
o  5729  {This deck defines values for the idle status of NOS/VE.
o  5730
o  5731    TYPE
o  5732      syt$180_idle_code = (syc$ic_null,
o  5733           syc$ic_system_terminated,
o  5734
o  5735  {  System is ABORTED for the next two codes}
o  5736
o  5737           syc$ic_fatal_hardware_error,
o  5738           syc$ic_fatal_software_error,
```


Decks referenced by selected decks

```
o  5739
o  5740  {  System is IDLED for the next three codes}
o  5741
o  5742           syc$ic_long_power,
o  5743           syc$ic_hardware_idle,
o  5744           syc$ic_idle_command,
o  5745
o  5746  {  System is STEPPED for all following codes}
o  5747
o  5748           syc$ic_step_command,
o  5749           syc$ic_short_power,
o  5750           syc$ic_disk_error,
o  5751           syc$ic_software_breakpoint);
o  5752
o  5753
o  5754    TYPE
o  5755      cmt$element_state = (cmc$on, cmc$off, cmc$down);
o  5756
o  5757
o  5758  { declarations for cpus }
o  5759
o  5760    TYPE
o  5761      ost$cpu_element_id = ost$processor_element_id,
o  5762      ost$cpu_memory_port_number = 0 .. 1f(16),
o  5763      ost$cpu_memory_port_mask = 0 .. 1f(16),
o  5764      ost$logical_processor_id = 0 .. osc$maximum_processor_number,
o  5765      ost$next_processor_state = (osc$null, osc$off, osc$down);
o  5766
o  5768    CONST
o  5769      osc$maximum_processor_number = osc$maximum_processors - 1;
o  5770
o  5771
o  5772    CONST
o  5773      osc$maximum_processors = 2;
o  5774
o  5775
o  5776   CONST
o  5777     osc$max_number_of_processors = osc$maximum_processors;
o  5778
o  5779    TYPE
o  5780      ost$processor_element_id = record
o  5781        fill: ost$halfword,
o  5782        element_number: ost$processor_element_number,
o  5783        model_number: ost$processor_model_number,
o  5784        serial_number: ost$processor_serial_number,
o  5785      recend;
o  5786
o  5787    TYPE
o  5788      ost$halfword = 0 .. 0ffffffff(16);
o  5789
o  5790    TYPE
o  5791      ost$processor_element_number = 0 .. 0ff(16);
o  5793   CONST
o  5794      osc$max_idle_count = 0fffffffffffff(16);
o  5795
o  5796   TYPE
```

Decks referenced by selected decks

```
0  5797     ost$cpu_idle_statistics = record
0  5798       idle_no_io_active: integer,
0  5799       idle_io_active: integer,
0  5800       idle_start_time: integer,
0  5801       idle_type: ost$idle_type,
0  5802       idle_count: 0 .. osc$max_idle_count,
0  5803     recend;
0  5804
0  5805   TYPE
0  5806     ost$idle_type = (osc$not_idle, osc$idle_with_io_active,
0  5807       osc$idle_no_io_active);
0  5808
0  5809    TYPE
0  5810      ost$cpu_state = RECORD
0  5811        current_state,
0  5812        next_state: ost$cpu_running_or_stepped,
0  5813      RECEND,
0  5814
0  5815      ost$cpu_running_or_stepped = (osc$cpu_running, osc$cpu_stepped);
0  5816
0  5817
0  5818 { TYPE declaration: OST$CPU_STATE_REASON
0  5819
0  5820    TYPE
0  5821      ost$cpu_state_reason = (osc$csr_state_at_cti_time, osc$csr_changed_by_dft, osc$csr_cpu_saw_too_many_dues,
0  5822            osc$csr_cpu_not_alive_recently, osc$csr_changed_by_operator)
0  5823
0  5824
0  5825 { OSDEIR - define external interrupt request for multi processors
0  5826 {           if these definitions are changed, code in MTACST
0  5827 {           must also be changed.
0  5828
0  5829    TYPE
0  5830      ost$external_interrupt_request = packed record
0  5831        task_switch: boolean,
0  5832        purge_cache: boolean,
0  5833        purge_map: boolean,
0  5834        step_processor: boolean,
0  5835      recend;
0  5836
0  5837    TYPE
0  5838      ost$pre_processed_for_reconfig = (osc$ppfr_not_processed, osc$ppfr_processing_in_progress,
0  5839            osc$ppfr_processing_complete);
0  5840
0  5841   TYPE
0  5842      ost$cst_trace_control = packed record
0  5843        fill: 0 .. 0ffff(16),
0  5844        buffer_p: ^cell,
0  5845      recend;
0  5846
0  5847    TYPE
0  5848      tmt$dual_state_dispatch_prior = ARRAY
0  5849         [jmc$priority_p1 .. jmc$priority_p14] OF
0  5850         tmt$dual_state_priority_entry,
0  5851
0  5852
```


Decks referenced by selected decks

```
0  5853     tmt$dual_state_priority_entry = record
0  5854        dual_state_priority: 0 .. 7,
0  5855        subpriority: 0 .. 15,
0  5856     recend;
0  5857
0  5858
0  5859    TYPE
0  5860      ost$register_number = 0 .. 0f(16),
0  5861      ost$x_register = integer;
0  5862
0  5863    TYPE
0  5864      ost$p_register = PACKED record
0  5865        undefined1: 0 .. 3(16),
0  5866        global_key: ost$key_lock_value,
0  5867        undefined2: 0 .. 3(16),
0  5868        local_key: ost$key_lock_value,
0  5869        pva: ost$pva,
0  5870      recend;
0  5871
0  5872
0  5873    TYPE
0  5874      ost$virtual_machine_identifier = (osc$cyber_180_mode, osc$cyber_170_mode,
0  5875        osc$50_reserved, osc$51_reserved, osc$52_reserved, osc$53_reserved,
0  5876        osc$54_reserved, osc$55_reserved, osc$56_reserved, osc$57_reserved,
0  5877        osc$58_reserved, osc$59_reserved, osc$60_reserved, osc$61_reserved,
0  5878        osc$62_reserved, osc$63_reserved);
0  5879
0  5880    TYPE
0  5881      ost$keypoint_class = 0 .. 0f(16),
0  5882
0  5883      ost$keypoint_mask = set of ost$keypoint_class;
0  5884
0  5885
0  5886    TYPE
0  5887      ost$monitor_condition = (osc$detected_uncorrected_err,
0  5888        osc$not_assigned, osc$short_warning, osc$instruction_spec,
0  5889        osc$address_specification, osc$exchange_request, osc$access_violation,
0  5890        osc$environment_spec, osc$external_interrupt, osc$page_fault,
0  5891        osc$system_call, osc$system_interval_timer, osc$invalid_segment_ring_0,
0  5892        osc$out_call_in_return, osc$soft_error, osc$trap_exception),
0  5893
0  5894      ost$monitor_conditions = set OF ost$monitor_condition;
0  5895
0  5896    TYPE
0  5897      ost$user_condition = (osc$privileged_instruction,
0  5898        osc$unimplemented_instruction, osc$free_flag, osc$process_interval_timer,
0  5899        osc$inter_ring_pop, osc$critical_frame_flag, osc$keypoint,
0  5900        osc$divide_fault, osc$debug, osc$arithmetic_overflow,
0  5901        osc$exponent_overflow, osc$exponent_underflow, osc$fp_significance_loss,
0  5902        osc$fp_indefinite, osc$arithmetic_significance, osc$invalid_bdp_data),
0  5903
0  5904      ost$user_conditions = set OF ost$user_condition;
0  5905
0  5906    TYPE
0  5907      ost$trap_enable = (osc$traps_disabled, osc$traps_undefined,
0  5908            osc$traps_enabled, osc$traps_enabled_delay);
```

Decks referenced by selected decks

```
0  5909
0  5910
0  5911   TYPE
0  5912     ost$stack_frame_save_area = record
0  5913       minimum_save_area: ost$minimum_save_area,
0  5914       undefined: 0 .. 0ffff[16],
0  5915       a3: ^cell,
0  5916       user_condition_register: ost$user_conditions,
0  5917       a4: ^cell,
0  5918       monitor_condition_register: ost$monitor_conditions,
0  5919       a5: ^cell,
0  5920       a_registers: array[6 .. 0f[16]] OF record
0  5921         undefined: 0 .. 0ffff[16],
0  5922         a_register: ^cell,
0  5923       recend,
0  5924       x_registers: array [ost$register_number] OF ost$x_register,
0  5925     recend;
0  5926
0  5927   TYPE
0  5928     ost$frame_descriptor = packed record
0  5929       critical_frame_flag: boolean,
0  5930       on_condition_flag: boolean,
0  5931       undefined: 0 .. 3[16],
0  5932       x_starting: ost$register_number,
0  5933       a_terminating: ost$register_number,
0  5934       x_terminating: ost$register_number,
0  5935     recend;
0  5936
0  5937   TYPE
0  5938     ost$minimum_save_area = packed record
0  5939       p_register: ost$p_register,
0  5940       vmid: ost$virtual_machine_identifier,
0  5941       undefined: 0 .. 0fff[16],
0  5942       a0_dynamic_space_pointer: ^cell,
0  5943       frame_descriptor: ost$frame_descriptor,
0  5944       a1_current_stack_frame: ^cell,
0  5945       user_mask: ost$user_conditions,
0  5946       a2_previous_save_area: ^ost$stack_frame_save_area,
0  5947     recend;
0  5948
0  5949
0  5950   TYPE
0  5951     ost$debug_list_entry = packed record
0  5952       debug_code: ALIGNED[0 MOD 16] packed array [ost$debug_code] of boolean,
0  5953       low_fill: 0 .. 0fff[16],
0  5954       seg: ost$segment,
0  5955       low_bn: ost$segment_offset,
0  5956       high_fill: 0 .. 0ffffffff[16],
0  5957       high_bn: ost$segment_offset,
0  5958     recend,
0  5959
0  5960     ost$debug_list = array [0 .. 31] of ost$debug_list_entry;
0  5961
0  5962
0  5963   TYPE
0  5964     ost$debug_code = (osc$data_read, osc$data_write, osc$instruction_fetch,
```


Decks referenced by selected decks

```
0  5965       osc$branching_instruction, osc$call_instruction, osc$end_of_list,
0  5966       osc$dc_6, osc$dc_7);
0  5967
0  5968
0  5969   TYPE
0  5970     ost$debug_mask = packed record
0  5971       end_of_list_seen_flag: boolean,
0  5972       scan_in_progress: boolean,
0  5973       codes: packed array [osc$data_read .. osc$call_instruction] of boolean,
0  5974     recend;
0  5975
0  5976
0  5977   TYPE
0  5978     jmt$service_class_attributes = RECORD
0  5979
0  5980 { Define the Definition group attributes.
0  5981
0  5982       defined: boolean,
0  5983       index: jmt$service_class_index,
0  5984       profile_identification: ost$name,
0  5985       name: jmt$service_class_name,
0  5986       abbreviation: jmt$service_class_name,
0  5987
0  5988 { Define the Control group attributes.
0  5989
0  5990       class_service_threshold: jmt$service_accumulator,
0  5991       guaranteed_service_quantum: jmt$service_accumulator,
0  5992       maximum_active_jobs: jmt$maximum_active_jobs,
0  5993       next_service_class_index: jmt$service_class_index,
0  5994       service_factors: ARRAY [jmt$service_factors] OF jmt$service_factor_value,
0  5995
0  5996 { Define the Priority group attributes.
0  5997
0  5998       dispatching_control: jmt$dispatching_control,
0  5999       scheduling_priority: jmt$scheduling_priority,
0  6000       swap_age_interval: jmt$priority_aging_interval, { microseconds
0  6001
0  6002     RECEND;
0  6003
0  6004   TYPE
0  6005
0  6006     jmt$scheduling_priority = RECORD
0  6007       minimum: jmt$job_priority,
0  6008       maximum: jmt$job_priority,
0  6009       swap_age_increment: jmt$job_priority,
0  6010       ready_task_increment: jmt$job_priority,
0  6011     RECEND;
0  6012
0  6013
0  6014   TYPE
0  6015     jmt$maximum_active_jobs = 0 .. jmc$max_active_jobs;
0  6016
0  6017 { The following constants define the range of values permitted on SCL
0  6018 { parameter definitions and the internal representation for the keyword,
0  6019 { UNLIMITED.
0  6020
```

Decks referenced by selected decks

```
o  6021   CONST
o  6022     jmc$lowest_maximum_active_jobs = 0,
o  6023     jmc$highest_maximum_active_jobs = jmc$max_active_jobs,
o  6024     jmc$unlimited_max_active_jobs = jmc$max_active_jobs;
o  6025
o  6026
o  6027 { The priority aging interval has a unit of microseconds.
o  6028
o  6029   TYPE
o  6030     jmt$priority_aging_interval = 0 .. jmc$priority_aging_interval_max;
o  6031
o  6032   CONST
o  6033     jmc$priority_aging_interval_max = jmc$highest_prio_age_interval +
o  6034           jmc$keyword_offset_maximum;
o  6035
o  6036 { The following constants define the range of values permitted on SCL
o  6037 { parameter definitions and the internal representation for the keyword,
o  6038 { UNLIMITED.
o  6039
o  6040   CONST
o  6041     jmc$lowest_prio_age_interval = 1 * 1000000, { microseconds
o  6042     jmc$highest_prio_age_interval = 36000 * 1000000, { microseconds
o  6043     jmc$unlimited_prio_age_interval = jmc$highest_prio_age_interval +
o  6044           jmc$unlimited_offset;
o  6045
o  6046
o  6047   TYPE
o  6048     jmt$service_class_name = ost$name;
o  6049
o  6050
o  6051   TYPE
o  6052     jmt$service_factors = (jmc$sf_cpu, jmc$sf_memory, jmc$sf_residence,
o  6053           jmc$sf_io);
o  6054
o  6055   TYPE
o  6056     jmt$service_factor_value = 0 .. jmc$service_factor_value_max;
o  6057
o  6058   CONST
o  6059     jmc$service_factor_value_max = jmc$highest_service_factor_valu;
o  6060
o  6061 { The following constants define the range of values permitted on SCL
o  6062 { parameter definitions.
o  6063
o  6064   CONST
o  6065     jmc$lowest_service_factor_value = 0,
o  6066     jmc$highest_service_factor_valu = 100;
o  6067
o  6068   TYPE
o  6069     mmt$shadow_reference_info = RECORD
o  6070       source_pva: ^cell,
o  6071       destination_pva: ^cell,
o  6072       page_count: 0 .. 255,
o  6073     RECEND;
o  6074
o  6076
o  6077   TYPE
```


Decks referenced by selected decks

```
o  6078     nat$received_message_list = record
o  6079       next_received_message: ALIGNED [0 MOD 8] ^nat$received_message_descriptor,
o  6080       fill: 0 .. 0ffff(16),
o  6081     recend;
o  6082
o  6083   TYPE
o  6084   {  The size of this record must be a multiple of 8 bytes.
o  6085
o  6086     nat$received_message_descriptor = record
o  6087       next_received_message: ALIGNED [0 MOD 8]
o  6088         ^nat$received_message_descriptor,
o  6089       device_id: nlt$device_identifier,
o  6090       transfer_count: iot$transfer_count,
o  6091       fill: 0 .. 0ffffffff(16),
o  6092       CASE pdu_type: nlt$pdu_type OF
o  6093       = nlc$channel_connection_pdu =
o  6094         sequence#_or_connect_timestamp: nlt$cc_seq#_or_connect_time,
o  6095       = nlc$channelnet_pdu =
o  6096         ,
o  6097       CASEND,
o  6098     recend;
o  6099
o  6100
o  6101   TYPE
o  6102     iot$transfer_count = 0 .. 0ffffffff(16);
o  6103
o  6104 TYPE
o  6105   nlt$cc_seq#_or_connect_time = record
o  6106     CASE nlt$cc_pdu_kind OF
o  6107     = nlc$cc_connect_request =
o  6108       time_connect_request_received: integer,
o  6109     = nlc$cc_connect_confirm .. nlc$cc_expedited_data =
o  6110       sequence_number: nlt$cc_sequence_number,
o  6111     CASEND,
o  6112   recend;
o  6113
o  6114 {  NLT$CC_PDU_KIND }
o  6115
o  6116 CONST
o  6117   nlc$cc_connect_request = 1,
o  6118   nlc$cc_connect_confirm = 2,
o  6119   nlc$cc_disconnect_request = 3,
o  6120   nlc$cc_disconnect_confirm = 4,
o  6121   nlc$cc_credit_allocation = 5,
o  6122   nlc$cc_data = 6,
o  6123   nlc$cc_expedited_data = 7,
o  6124   nlc$cc_global_window = 8,
o  6125   nlc$cc_max_pdu_kind = 0ff(16);
o  6126
o  6127 TYPE
o  6128   nlt$cc_pdu_kind = 0 .. nlc$cc_max_pdu_kind;
o  6129
o  6130 TYPE
o  6131   nlt$cc_sequence_number = integer;
o  6132
o  6133   CONST
```

Decks referenced by selected decks

```
0  6134      nlc$null_device_identifier = 0,
0  6135      nlc$maximum_network_devices = 64;
0  6136
0  6137    TYPE
0  6138      nlt$device_identifier = 0 .. 0ff(16);
0  6139
0  6140  TYPE
0  6141    nlt$pdu_type = (nlc$channelnet_pdu, nlc$channel_connection_pdu);
0  6142
0  6143    TYPE
0  6144      ost$quantum = 0 .. 7fffffff(16);
0  6145
0  6146    TYPE
0  6147      ost$ring1_termination_reason = (osc$rtr_non, osc$rtr_sft_full);
0  6148
0  6149    TYPE
0  6150      ost$task_id = record
0  6151        global: ost$global_task_id,
0  6152        local: pmt$task_id,
0  6153      recend;
0  6154
0  6155
0  6156    TYPE
0  6157      pmt$task_id = 0 .. pmc$max_task_id;
0  6158
0  6159    CONST
0  6160      pmc$max_task_id = 0ffffffff(16);
0  6161    TYPE
0  6162      ost$byte = 0 .. 0ff(16);
0  6163    TYPE
0  6164      ost$parcel = 0 .. 0ffff(16);
0  6165    TYPE
0  6166      ost$word = integer;
0  6167
0  6168    TYPE
0  6169      tmt$handler_execution_ring = ost$ring;
0  6170
0  6171
0  6172   CONST
0  6173      syc$ucr_condition = 0,
0  6174      syc$user_defined_condition = 1;
0  6175
0  6176    CONST
0  6177      syc$volume_unavailable = 'SYC$VOLUME_UNAVAILABLE          ',
0  6178      syc$no_file_space = 'SYC$NO_FILE_SPACE              ';
0  6179
0  6180    TYPE
0  6181      syt$system_core_condition = record
0  6182        sfsa: ^ost$stack_frame_save_area,
0  6183        case condition: 0 .. 0ff(16) of
0  6184        = syc$ucr_condition =
0  6185          ucr: ost$user_conditions,
0  6186        = syc$user_defined_condition =
0  6187          name: ost$name,
0  6188        casend,
0  6189      recend,
```


Decks referenced by selected decks

```
0  6190      syt$continue_option = (syc$condition_processed,
0  6191        syc$condition_ignored);
0  6192
0  6193
0  6194    TYPE
0  6195       tmt$wait_inhibited = (tmc$wi_null, tmc$wi_wait_inhibited,
0  6196           tmc$wi_wait_selected, tmc$wi_wait_selected_r3);
0  6197
0  6198    {Declarations for a SIGNAL}
0  6199    TYPE
0  6200      pmt$signal = record
0  6201        identifier: pmt$signal_id,
0  6202        contents: pmt$signal_contents,
0  6203      recend,
0  6204
0  6205      pmt$signal_id = (tmc$signal_available_0, ofc$signal,
0  6206          mlc$signal_id, ifc$signal_id, pmc$ss_child_terminated,
0  6207          jmc$timesharing_signal_id, tmc$signal_available_6,
0  6208          jmc$sense_switch_signal_id, tmc$signal_available_8,
0  6209          jmc$job_resource_signal_id, tmc$signal_available_10,
0  6210          tmc$signal_available_11, nac$network_device_error,
0  6211          tmc$signal_available_13, tmc$signal_available_14,
0  6212          pmc$multi_task_condition, nac$gt_deliver_data,
0  6213          nac$gt_send_data, nac$gt_deliver_connect_request,
0  6214          nac$se_deliver_data_signal, nac$se_send_data_signal,
0  6215          nac$se_disconnect_signal, tmc$signal_available_22,
0  6216          tmc$signal_available_23, tmc$signal_available_24,
0  6217          tmc$signal_available_25, tmc$signal_available_26,
0  6218          tmc$signal_available_27, tmc$signal_available_28,
0  6219          tmc$signal_available_29, tmc$signal_available_30,
0  6220          tmc$signal_available_31, tmc$signal_available_32,
0  6221          tmc$signal_available_33, tmc$signal_available_34,
0  6222          tmc$signal_available_35, tmc$signal_available_36,
0  6223          tmc$signal_available_37, tmc$signal_available_38,
0  6224          tmc$signal_available_39, tmc$signal_available_40,
0  6225          tmc$signal_available_41, tmc$signal_available_42,
0  6226          tmc$signal_available_43, tmc$signal_available_44,
0  6227          tmc$signal_available_45, tmc$signal_available_46,
0  6228          tmc$signal_available_47, tmc$signal_available_48,
0  6229          tmc$signal_available_49, tmc$signal_available_50,
0  6230          tmc$signal_available_51, tmc$signal_available_52,
0  6231          tmc$signal_available_53, tmc$signal_available_54,
0  6232          tmc$signal_available_55, tmc$signal_available_56,
0  6233          tmc$signal_available_57, tmc$signal_available_58,
0  6234          tmc$signal_available_59, tmc$signal_available_60,
0  6235          tmc$signal_available_61, tmc$signal_available_62,
0  6236          tmc$signal_available_63),
0  6237
0  6238      pmt$signal_contents = array [1 .. pmc$max_signal_contents] of 0 .. 0ff(16);
0  6239
0  6240    CONST
0  6241      pmc$max_signal_id = 63;
0  6242
0  6243    CONST
0  6244      pmc$max_signal_contents = 32;
0  6245
```

Decks referenced by selected decks

```
o  6246   CONST
o  6247     tmc$last_signal_id_assigned = tmc$signal_available_63;
o  6248
o  6249
o  6250   TYPE
o  6251     ost$system_flag = (pmc$kill_task_flag, avc$monitor_statistics_flag,
o  6252           pmc$sf_terminate_task, jmc$terminate_job_flag,
o  6253           tmc$mainframe_linked_signals, jmc$logout_flag_id, jmc$kill_job_flag,
o  6254           dsc$retrieve_system_message, nac$network_input_received,
o  6255           nlc$xt_work_list_flag, nac$xi_local_event,
o  6256           nac$channelnet_local_event, nac$notify_routing_me,
o  6257           syc$job_recovery_flag, ioc$subsystem_io_completed,
o  6258           dsc$log_dft_flag_id, ofc$operator_break_flag,
o  6259           osc$system_unstep_resume_flag, nlc$cc_work_list_flag,
o  6260           rfc$pp_response_available, mmc$failed_file_alloc_flag,
o  6261           mmc$volume_unavailable_flag, jmc$message_waiting_flag_id,
o  6262           tmc$flag_available_23, tmc$flag_available_24, tmc$flag_available_25,
o  6263           tmc$flag_available_26, tmc$flag_available_27, tmc$flag_available_28,
o  6264           tmc$flag_available_29, tmc$flag_available_30, tmc$flag_available_31);
o  6265
o  6266   CONST
o  6267     tmc$first_system_flag = pmc$kill_task_flag,
o  6268     tmc$last_system_flag = tmc$flag_available_31,
o  6269     tmc$last_flag_id_assigned = 31;
o  6270
o  6271   CONST
o  6272     osc$maximum_system_flag = 31;
o  6273
o  6274 MODEND
```

**** I=CC L=ZZXXLIST B=LGO DA=(DS,DT) LO=RX RC=NONE OPT=LOW EL=F LF=CS612 PAD=0

**** NO DIAGNOSTICS

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES						
amc$access_mode	4442	3925						
amc$average_record_length	4444	4000						
amc$block_type	4445	3927						
amc$character_conversion	4446	3929						
amc$clear_space	4447	3931						
amc$collate_table_name	4449	4002						
amc$compression_procedure_name	4502	4004						
amc$data_padding	4450	4007						
amc$dynamic_home_block_space	4503	4009						
amc$embedded_key	4451	4011						
amc$error_exit_name	4452	3933						
amc$error_limit	4454	4013						
amc$error_options	4455	3935						
amc$estimated_record_count	4456	4015						
amc$file_access_procedure	4457	3937						
amc$file_byte_limit	3519	3522	3524	4209	4247	4283	4556	4618
amc$file_contents	4458	3939						
amc$file_limit	4460	3941						
amc$file_organization	4461	3943						
amc$file_processor	4462	3945						
amc$file_structure	4463	3947						
amc$forced_write	4464	3949						
amc$hashing_procedure_name	4504	4017						
amc$index_levels	4470	4019						
amc$index_padding	4471	4021						
amc$initial_home_block_count	4505	4023						
amc$internal_code	4472	3951						
amc$key_length	4473	4025						
amc$key_position	4474	4027						
amc$key_type	4475	4029						
amc$label_exit_name	4476	3953						
amc$label_options	4478	3955						
amc$label_type	4479	3957						
amc$line_number	4480	3959						
amc$loading_factor	4506	4031						
amc$lock_expiration_time	4507	4033						
amc$log_residence	4509	4037						
amc$logging_options	4508	4035						
amc$max_attribute	4550	4554						
amc$max_block_length	4481	3961						
amc$max_block_number	4069	4072						
amc$max_error_count	4434	4437						
amc$max_file_id_ordinal	4371	4378						
amc$max_home_blocks	4247	4250						
amc$max_index_level	4242	4245						
amc$max_key_length	4575	4579						
amc$max_key_position	4584	4581						
amc$max_line_number	4260	4263						
amc$max_lines_per_inch	4642	4639						
amc$max_page_width	4212	4215						
amc$max_path_name_size	4226	4229						
amc$max_record_length	4482	3963						

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES						
amc$max_records_per_block	4626	4630						
amc$max_statement_id_length	4307	4310						
amc$max_user_info	4637	4633						
amc$maximum_block	4077	4074	4609					
amc$maximum_keyed_record	4587	4584						
amc$maximum_record	4283	4286	4352	4613				
amc$message_control	4483	4039						
amc$min_block_length	4484	3965						
amc$min_record_length	4485	3967						
amc$null_attribute	4486	3969						
amc$open_position	4487	3971						
amc$padding_character	4488	3973						
amc$page_format	4489	3975						
amc$page_length	4490	3977						
amc$page_width	4491	3979						
amc$preset_value	4493	3981						
amc$record_limit	4494	4041						
amc$record_type	4495	3983						
amc$records_per_block	4496	4043						
amc$return_option	4497	3985						
amc$ring_attributes	4498	3987						
amc$statement_identifier	4499	3989						
amc$user_info	4500	3991						
amc$vertical_print_density	4501	3993						
amt$access_selection	3914	3912						
amt$average_record_length	4352	4001						
amt$block_header_type	4050	4053	4059					
amt$block_number	4072	4055	4062					
amt$block_status	4051	4064						
amt$block_type	4355	3928						
amt$collation_value	4360	4357						
amt$compression_procedure_name	4217	4006						
amt$data_padding	4363	4008						
amt$dynamic_home_block_space	4236	4010						
amt$entry_point_reference	4220	4217	4238					
amt$error_limit	4430	4014						
amt$estimated_record_count	4439	4016						
amt$file_attribute_keys	4554	3919						
amt$file_byte_address	3522	26	32	77	78	175	199	323	2121
		3560	3765	3769	3781	3783	3799	4872	4909
		4910							
amt$file_contents	4095	3940						
amt$file_id_ordinal	4378	4375						
amt$file_id_sequence	4379	4376						
amt$file_identifier	4374	4366	4593					
amt$file_item	3918	3914	3917					
amt$file_limit	3524	327	3787	3942	4840			
amt$file_organization	4560	3944						
amt$file_position	4563	3905						
amt$file_processor	4156	3946						
amt$file_structure	4202	3948						
amt$forced_write	4566	3950						

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES			
amt$hashing_procedure_name	4238	4018			
amt$index_levels	4245	4020			
amt$index_padding	4570	4022			
amt$initial_home_block_count	4250	4024			
amt$internal_code	4572	3952			
amt$key_length	4579	4026			
amt$key_position	4581	4028			
amt$key_type	4590	4030			
amt$label_options	3907	3956			
amt$label_type	4597	3958			
amt$line_number	4254	3960			
amt$line_number_length	4263	4255			
amt$line_number_location	4265	4256			
amt$loading_factor	4268	4032			
amt$lock_expiration_time	4270	4034			
amt$log_residence	4272	4038			
amt$logging_options	4275	4036			
amt$logging_possibilities	4278	4275			
amt$max_block_length	4074	3962	4054	4060	4061
amt$max_record_length	4286	3964			
amt$message_control	4607	4040			
amt$min_block_length	4609	3966			
amt$min_record_length	4613	3968			
amt$open_position	4290	3972			
amt$padding_character	4616	3974			
amt$page_format	4205	3976			
amt$page_length	4209	3978			
amt$page_width	4215	3980	4265	4312	
amt$path_name	4229	4222	4272		
amt$preset_value	3528	3807	3982	4838	
amt$record_limit	4618	4042			
amt$record_type	4622	3984			
amt$records_per_block	4630	4044			
amt$return_option	3908	3986			
amt$ring_attributes	4294	3988			
amt$statement_id_length	4310	4302			
amt$statement_id_location	4312	4303			
amt$statement_identifier	4301	3990			
amt$tape_error_action	4322	4317			
amt$tape_error_options	4315	3936			
amt$unused_bit_count	4082	4056	4063		
amt$user_info	4633	3992			
amt$vertical_print_density	4639	3994			
boo164	292	280	281	284	
cmt$element_state	5755	2485	2486	2516	
dfc$active	3569	29			
dfc$awaiting_recovery	3570	37			
dfc$command_record_bytes	5135	5143			
dfc$division_overwrite_words	5122	5150			

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES				
dfc$esm_command_record_size	5143	5151				
dfc$esm_header_record_size	5144	5151				
dfc$esm_maintenance_buf_size	5123	5154				
dfc$esm_memory_base_shift	5129	5151	5152	5152		
dfc$header_record_bytes	5134	5144				
dfc$max_esm_memory_size	5124	5153				
dfc$max_family_ptr_array_size	3549	3552				
dfc$max_number_of_mainframes	5131	5116				
dfc$maximum_lifetime	3534	3531				
dfc$min_data_record_bytes	5139	5150				
dfc$min_esm_division_size	5149	5153				
dfc$served_family_list_size	3543	3546				
dfc$terminated	3570	37				
dft$family_pointer_index	3552	3538				
dft$lifetime	3531	23				
dft$mainframe_set	5116	567	568	713	714	
dft$served_family_list_index	3546	3539				
dft$served_family_table_index	3537	22				
dft$server_allocation_info	3555	35				
dft$server_descriptor	16	14	3484			
dft$server_descriptor_header	20	17				
dft$server_state	3569	28	3572			
dmc$a2	3589	3593				
dmc$allocated_length	3826	198	3764			
dmc$asid	3826	3766				
dmc$byte_address	3827	3768				
dmc$bytes_per_allocation	3827	3770				
dmc$bytes_per_level_2	4892	4900				
dmc$chapter_length	3834	3818				
dmc$class	3827	3772	4745			
dmc$class_ordinal	3827	3774	4747			
dmc$clear_space	3828	3776				
dmc$default_number_fau_entries	155	148				
dmc$device_file_list_index	3828	200	3778			
dmc$eof_byte_address	3828	3780				
dmc$eoi_byte_address	3829	3782				
dmc$file_hash	3829	3784				
dmc$file_kind	3830	3790	4751			
dmc$file_limit	3829	3786				
dmc$file_status	3829	3788				
dmc$global_file_name	3830	3792				
dmc$internal_vsn	3830	202	3794			
dmc$level_1_table_size	4888	4892	4895			
dmc$locked_file	3830	3796				
dmc$logical_length	3831	3798				
dmc$master_volume_required	3831	3800	4753			
dmc$max_bytes_per_allocation	3585	27	73	3578	3580	3581
dmc$max_bytes_per_dau	3640	3632				
dmc$max_bytes_per_mau	3615	3601				
dmc$max_class_ordinal	4657	4654				
dmc$max_dau_address	3642	3623	3636			
dmc$max_daus_allocation	3644	3633				

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES				
dmc$max_daus_position	3646	3625	3634			
dmc$max_daus_transfer	3648	3635				
dmc$max_device_file_list_index	4675	4672				
dmc$max_fau_entries	156	148				
dmc$max_file_hash	4685	99	101	4682		
dmc$max_mau_address	3623	3611				
dmc$max_maus_per_allocation	3617	3603	3604			
dmc$max_maus_per_dau	3619	3605	3606	3623	3625	
dmc$max_maus_per_transfer	3621	3608	3609			
dmc$max_maus_position	3625	3610				
dmc$max_transfer_size	3661	3656				
dmc$min_bytes_per_allocation	3584	3579				
dmc$min_bytes_per_dau	3639	3632				
dmc$min_dau_address	3641	3636				
dmc$min_mau_address	3622	3611				
dmc$min_maus_per_allocation	3616	3602				
dmc$min_maus_per_dau	3618	3605				
dmc$min_maus_per_transfer	3620	3607				
dmc$overflow	3831	3802				
dmc$owner	3832	3804				
dmc$preset_value	3832	3806				
dmc$queue_status	3834	3822				
dmc$recorded_vsn	3832	204	3808	4749		
dmc$requested_allocation_size	3832	3810	4755			
dmc$requested_transfer_size	3833	3812				
dmc$requested_volume	3833	3814				
dmc$setname	3833	3816	4757			
dmc$write_mode	3834	3820				
dmt$access_kind	4725	4731				
dmt$active_volume_table_index	4961	171				
dmt$allocation_size	3578	81	3771	3811	4756	4844
dmt$allocation_styles	3589	183				
dmt$bytes_per_mau	3601	176				
dmt$class_member	4649	82	3773	4648	4746	
dmt$class_ordinal	4654	83	3775	4748		
dmt$dau_address	3636	142				
dmt$daus_per_allocation	3633	178				
dmt$daus_per_position	3634	177				
dmt$delete_count	94	70				
dmt$delete_logging_count	187	173				
dmt$device_file_list_index	4672	172	201	3779		
dmt$disk_file_descriptor	68	3485				
dmt$fau_states	150	143				
dmt$file_allocation_status	127	3486				
dmt$file_allocation_unit	141	139	4920	4921		
dmt$file_attribute_keywords	3826	197	3763	4744		
dmt$file_hash	4682	3785				
dmt$file_hash_thread	97	99				
dmt$file_medium_descriptor	168	87	182	3487		
dmt$fmd_attribute	196	192				
dmt$fmd_index	232	76	86	144	191	
dmt$global_file_name	4688	3793	4843			

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES					
dmt$internal_vsn	4718	179	203	3795			
dmt$level_1_index	4895	75	4909				
dmt$level_1_table	4909	74					
dmt$level_2_index	4900	4920					
dmt$locked_file	4728	3797					
dmt$maus_per_dau	3605	180					
dmt$maus_per_transfer	3607	181					
dmt$queue_status	4854	4841					
dmt$requested_volume	4761	85	3815				
dmt$system_file_id	4930	170	745	2103	2107	2108	2116	2120
dmt$transfer_size	3656	36	84	3813	4845			
dmt$usage_count	4861	4842					
dmt$write_lock	4727	4733					
dpc$console_row_size	5679	5673					
dpc$top_line_message_size	5673	5668					
dpt$top_line_message	5668	5643					
gfc$fde_table_base	243	244					
gfc$fk_catalog	406	418					
gfc$fk_job_local_file	408	417					
gfc$fm_mass_storage_file	423	336					
gfc$fm_served_file	424	339					
gfc$max_level_1_index	295	281					
gfc$max_level_2_bit_index	297	286					
gfc$max_level_2_index	296	284	288				
gft$allocation_unit_size	263	325					
gft$attach_count	270	316	317				
gft$fde_flags	345	313					
gft$file_desc_entry_p	387	2142					
gft$file_descriptor_entry	310	97	315	387	431	3488	
gft$file_descriptor_index	382	484					
gft$file_kind	402	319	414	3791	4752		
gft$file_media	423	335					
gft$open_count	4997	318	472				
gft$queue_status	465	328	3823				
gft$segment_lock_info	471	321					
gft$signature_lock	4985	311					
gft$system_file_identifier	483	33	1527	2199	2261	2340	2703	4930
gft$table_residence	498	485					
gft$transfer_unit_size	4982	326					
ioc$max_unit_number	4967	4961	4970				
iot$io_error	5464	746	1897				
iot$transfer_count	6102	6090					
jmc$detached_job_wait_time_max	5472	5469					
jmc$highest_det_job_wait_time	5482	5472	5483				
jmc$highest_prio_age_interval	6042	6033	6043				
jmc$highest_service_accumulator	5373	5374					
jmc$highest_service_factor_valu	6066	6059					
jmc$highest_working_set_size	5508	5499	5509	5511	5513	5515	
jmc$ies_job_swapped	5289	5298					

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES							
jmc$ies_swapin_in_progress	5288	5297							
jmc$iss_idle_tasks_initiated	625	652							
jmc$iss_swapin_io_complete	650	653							
jmc$iss_swapin_requested	646	653							
jmc$iss_swapout_complete	645	652							
jmc$iss_swapped_io_cannot_init	636	663							
jmc$iss_swapped_no_io	627	662							
jmc$keyword_offset_maximum	5390	5500	6034						
jmc$kjl_maximum_entries	5181	5174	5175	5325					
jmc$kol_maximum_entries	5191	5176							
jmc$max_active_jobs	5172	6015	6023	6024					
jmc$max_ajl_ord	5173	5166	5172						
jmc$max_dispatching_control	5248	5252							
jmc$max_dispatching_priority	5083	5043	5046	5047					
jmc$max_ijl_index_count	595	799							
jmc$maximum_job_classes	825	828							
jmc$maximum_job_count	5188	5181							
jmc$maximum_output_count	5198	5191							
jmc$maximum_service_classes	5406	5409							
jmc$min_dispatching_control	5247	5251							
jmc$null_service_class	5399	5400							
jmc$priority_aging_interval_max	6033	6030							
jmc$priority_p1	5097	5044	5849						
jmc$priority_p10	5106	507	511	514	5045				
jmc$priority_p11	5107	508	510						
jmc$priority_p12	5108	509							
jmc$priority_p13	5109	512							
jmc$priority_p14	5110	5045	5849						
jmc$priority_p8	5104	5044							
jmc$priority_p9	5105	515							
jmc$required_offset	5388	5514							
jmc$reserved_ajls	5177	5172							
jmc$service_accumulator_maximum	5365	5362							
jmc$service_factor_value_max	6059	6056							
jmc$system_default_offset	5389	5390	5516						
jmc$system_supplied_name_size	5429	5426							
jmc$unlimited_offset	5386	5375	5473	5484	5510	6044			
jmc$unspecified_offset	5387	5512							
jmc$working_set_size_maximum	5499	5496							
jmt$active_job_list_entry	532	540	3489						
jmt$ajl_ordinal	5166	682	2489						
jmt$delayed_swapin_work	560	564	712						
jmt$detached_job_wait_time	5469	872							
jmt$dispatching_control	5218	5998							
jmt$dispatching_control_index	5251	5208	5218						
jmt$dispatching_controls	5221	5219							
jmt$dispatching_priority	5043	694	2481	2675	2677	3214	3305	3306	5209
		5210	5211	5223					
jmt$ijl_block_index	591	587	801						
jmt$ijl_block_number	590	586							
jmt$ijl_dispatching_control	5207	695							
jmt$ijl_entry_status	5284	681							

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES							
jmt$ijl_ordinal	585	534	701	729	862	1520	1712	1889	2508
		3297	5444	5445					
jmt$ijl_page_fault_count	5313	5308	5309	5310					
jmt$ijl_page_stats	5307	5303							
jmt$ijl_service_class_stats	5301	716							
jmt$ijl_statistics	606	715							
jmt$ijl_swap_count	5322	5318	5319						
jmt$ijl_swap_counts	5317	735	5304						
jmt$ijl_swap_status	623	684	685	686	1073	1073	1087		
jmt$initiated_job_list_block	798	804							
jmt$initiated_job_list_entry	678	535	801	861	1032	2509	3490		
jmt$input_file_location	5345	5340							
jmt$job_abort_disposition	5354	5338							
jmt$job_class	828	740							
jmt$job_control_block	843	2491	3491						
jmt$job_mode	905	697							
jmt$job_priority	912	737	738	6007	6008	6009	6010		
jmt$job_recovery_disposition	5357	5339							
jmt$job_system_id	5488	858							
jmt$kjl_index	5325	683	5488						
jmt$maximum_active_jobs	6015	5992							
jmt$priority_aging_interval	6030	6000							
jmt$queue_file_ijl_information	5337	722							
jmt$scheduling_data	728	706							
jmt$scheduling_priority	6006	5999							
jmt$service_accumulator	5362	730	731	732	5990	5991			
jmt$service_class_index	5409	741	5983	5993					
jmt$service_class_name	6048	5985	5986						
jmt$service_factor_value	6056	5994							
jmt$service_factors	6052	5994							
jmt$swap_data	744	708							
jmt$swapout_reasons	5412	736							
jmt$swapped_job_entry	951	753	881	1033					
jmt$system_supplied_name	5426	679	856						
jmt$task_time_slice	5261	5241	5242						
jmt$time_slice_values	5240	2688	5225						
jmt$user_supplied_name	5492	857							
jmt$working_set_size	5496	868	869						
jsc$isqi_swapped_io_completed	5449	5451							
jsc$isqi_swapped_io_not_init	5448	5451							
jsc$min_ecc	968	969							
jsc$min_ecc_js	969	972	975	978	981	984	987	990	993
		996	999	1002	1005				
jst$changed_asid_entry	1055	1046							
jst$ijl_swap_queue_id	5448	5443							
jst$ijl_swap_queue_link	5442	690							
jst$io_control_information	1015	709							
jst$swap_file_descriptor	1031	710							
jst$swap_state_statistics_entry	1075	1074							
jst$swapped_page_descriptor	1040	1038							
jst$swapped_page_descriptors	1037	1034							

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES							
mmc$	1170	1176	1179	1182	1185	1188	1191	1194	1198
		1201	1204	1207	1210	1213	1216	1220	1223
		1226	1229	1232	1235	1239	1242	1245	1248
		1251	1254	1257	1260	1263	1266	1269	1272
		1275	1278	1281	1284	1287	1290	1293	1296
		1299	1303	1306	1309	1312	1315	1318	1321
		1324	1327	1330	1333	1336	1339	1342	1345
		1348	1351	1354	1357	1360	1363	1367	1371
		1374	1378	1381	1384	1387	1390	1393	1396
		1399	1402	1405	1408	1411	1414	1417	1420
		1423	1426	1429	1432	1435	1439	1443	1446
		1449	1452	1455	1458	1461	1464	1467	1470
		1473	1476	1479	1482	1485	1488	1491	1494
		1497	1501	1504	1507				
mmc$assign_active_null	2301	2302							
mmc$bd_explicit_io	1717	1709							
mmc$bd_job_swapping_io	1716	1711							
mmc$bd_paging_io	1716	1709							
mmc$cell_pointer	1679	1684							
mmc$heap_pointer	1680	1688							
mmc$kw_asid	1605	1641							
mmc$kw_clear_space	1603	1628							
mmc$kw_current_segment_length	1602	1622							
mmc$kw_error_exit_procedure	1604	1632							
mmc$kw_gl_key	1604	1626							
mmc$kw_hardware_attributes	1606	1635							
mmc$kw_inheritance	1606	1643							
mmc$kw_max_segment_length	1603	1624							
mmc$kw_preset_value	1605	1630							
mmc$kw_ps_transfer_size	1607	1651							
mmc$kw_ring_numbers	1601	1617							
mmc$kw_segment_access_control	1605	1639							
mmc$kw_segment_number	1602	1620							
mmc$kw_shadow_segment	1607	1645							
mmc$kw_software_attributes	1604	1637							
mmc$kw_wired_segment	1607	1648							
mmc$lus_lock_segment	2056	2057							
mmc$lus_unlock_segment	2056	2059							
mmc$max_rma_list_length	2159	2164	2165						
mmc$pq_avail	1816	1862							
mmc$pq_free	1815	1874							
mmc$pq_job_fixed	1856	1863	1875						
mmc$pq_job_working_set	1858	1875	1876						
mmc$pq_shared_first	1864	1560							
mmc$pq_shared_first_site	1866	1870							
mmc$pq_shared_last_sys	1865	1560							
mmc$pq_shared_num_sites	1867	1870							
mmc$pq_shared_other	1825	1865							
mmc$pq_shared_site_01	1827	1866							
mmc$pq_shared_site_25	1851	1871							
mmc$pq_shared_task_service	1820	1864							
mmc$pq_swapped_io_error	1854	1874							

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES							
mmc$pq_wired	1818	1861							
mmc$segment_fault_processor_id	3252	2820							
mmc$sequence_pointer	1679	1686							
mmc$sr1_change_swap_file_queue	2093	2100							
mmc$sr1_delete_job_seg_by_sfid	2095	2101							
mmc$sr1_delete_seg_segnum	2082	2104							
mmc$sr1_delete_seg_sfid	2083	2099							
mmc$sr1_detach_file	2086	2099							
mmc$sr1_end_job_recovery	2090	2110							
mmc$sr1_flush_avail_modified	2097	2102							
mmc$sr1_flush_delete_seg_sfid	2087	2099							
mmc$sr1_flush_seg_segnum	2088	2100							
mmc$sr1_get_highest_offset	2094	2119							
mmc$sr1_make_mfw_cache	2091	2113							
mmc$sr1_remove_detached_pages	2096	2101							
mmc$sr1_remove_job_shared_pages	2092	2115							
mmc$sr1_replace_sfid	2089	2106							
mmc$ssk_none	5616	2344							
mmc$ssk_segment_number	5617	2342							
mmt$active_segment_table_entry	1517	1043	1533	1584	1896	3492			
mmt$ast_index	1574	322	752	1058	2109	2227			
mmt$attribute_descriptor	1615	2200							
mmt$attribute_keyword	1601	1616							
mmt$buffer_descriptor_type	1716	1708							
mmt$eoi_state	4940	324							
mmt$global_page_queue_index	1874	1940							
mmt$global_page_queue_list_ent	1930	1940							
mmt$hardware_attribute_set	1670	1636							
mmt$hardware_attributes	1658	1670							
mmt$job_page_queue_index	1875	953	1941						
mmt$job_page_queue_list	1941	707							
mmt$link	1759	1518	1886	1887	1927				
mmt$lock_segment_status	5596	2266							
mmt$locked_page	1772	1892	2030						
mmt$lus_lock_type	1787	2058							
mmt$lus_page_disposition	1789	2060							
mmt$max_sdt	2237	2241							
mmt$max_sdtx	2290	2294							
mmt$memory_reserve_request	5455	700							
mmt$page_age	2358	1895	2362	2362					
mmt$page_descriptor	1736	1733							
mmt$page_frame_index	1802	1016	1018	1019	1020	1761	1761	5457	5458
mmt$page_frame_queue_id	1876	1017	1526	1890	1957				
mmt$page_frame_table_entry	1885	1041	1901	3493					
mmt$page_queue_list_entry	1926	1931	1941						
mmt$rma_list_entry	2167	2162							
mmt$rma_list_index	2164	2162							
mmt$rma_list_length	2165	1707							
mmt$sdtx_stream_data	2273	2269							
mmt$segment_access_condition	1160	2821							
mmt$segment_access_rights	2183	2265							
mmt$segment_access_state	5586	2260							

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES							
mmt$segment_descriptor	2224	2234	2238						
mmt$segment_descriptor_extended	2258	2287	2291						
mmt$segment_descriptor_table_ex	2286	3494							
mmt$segment_inheritance	2323	1644	2262						
mmt$segment_pointer_kind	1679	1683	2198						
mmt$segment_reservation_state	5606	2263							
mmt$set_get_subfunction_codes	2147	2144							
mmt$shadow_info	2337	2267							
mmt$shadow_reference_info	6069	2701							
mmt$shadow_segment_kind	5616	2341							
mmt$software_attribute_set	1672	1638	2264						
mmt$software_attributes	1666	1672							
mmt$xcb_page_wait_info	2402	2687							
mtc$scb_max_hardware_status	5685	5635							
mtt$idle_status_block	2453	2440							
mtt$monitor_interlock	5005	312							
mtt$scb_180_status	2429	2420							
mtt$scb_hardware_status	5637	2415	5660						
mtt$scb_hardware_status_count	5635	5638							
mtt$scb_hardware_status_msg	5641	5647							
mtt$scb_hardware_status_msgs	5646	2423							
mtt$scb_hardware_status_options	5627	5637	5646						
mtt$smu_communications_block	2414	3495							
mtt$step_status_block	2448	2441							
mtt$system_idle_update_request	5690	2455	2455						
mtt$system_status_block	2439	2430							
mtt$system_step_update_request	5689	2450	2450						
nat$received_message_descriptor	6086	6079	6088						
nat$received_message_list	6078	2669							
nlc$cc_connect_confirm	6118	6109							
nlc$cc_connect_request	6117	6107							
nlc$cc_expedited_data	6123	6109							
nlc$cc_max_pdu_kind	6125	6128							
nlc$channel_connection_pdu	6141	6093							
nlc$channelnet_pdu	6141	6095							
nlt$cc_pdu_kind	6128	6106							
nlt$cc_seq#_or_connect_time	6105	6094							
nlt$cc_sequence_number	6131	6110							
nlt$device_identifier	6138	6089							
nlt$pdu_type	6141	6092							
osc$aging_interval_maximum	5520	5523							
osc$call_instruction	5965	5973							
osc$data_read	5964	5973							
osc$free_running_clock_maximum	5021	5018							
osc$invalid_ring	3848	3888							
osc$max_fault_contents	2833	2827							
osc$max_idle_count	5794	5802							
osc$max_name_size	4125	4129	4132	4330					
osc$max_number_of_processors	5777	2476							
osc$max_page_frames	5545	747	748	952	954	1519	1802	1928	1934

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES						
osc$max_page_size	3752	3748						
osc$max_page_table_entries	5546	5549						
osc$max_ring	3847	3888	3889					
osc$max_segment_length	3871	2270	2301	3894	4077			
osc$max_status_condition_code	4398	4394	4410					
osc$max_string_size	4414	4417	4420	4425				
osc$max_tasks	2744	2741						
osc$maximum_offset	3870	3871	3891	3891	3892			
osc$maximum_processor_id	5705	5701						
osc$maximum_processor_number	5769	5764						
osc$maximum_processors	5773	5769	5777					
osc$maximum_segment	3869	3890						
osc$min_page_size	3751	3748						
osc$min_ring	3846	3889						
osc$task_time_slice_maximum	5272	5275						
osc$tmtr_ring	3850	3103						
osc$tsrv_ring	3851	3104						
ost$aging_interval	5523	870	871					
ost$asid	2762	691	1045	1056	1057	1524	1642	2758	2900
		3767	4837					
ost$binary_unique_name	4696	314	4688	4718	4802			
ost$byte_count	2752	2032						
ost$clear_file_space	3668	3777	3932					
ost$cp_time	2556	607	936	2686	5302			
ost$cp_time_value	2554	733	865	866	2557	2558	2699	
ost$cpu_element_id	5761	2507						
ost$cpu_idle_statistics	5797	2510						
ost$cpu_memory_port_mask	5763	2483						
ost$cpu_running_or_stepped	5815	5812	5812					
ost$cpu_state	5810	2492						
ost$cpu_state_reason	5821	2513						
ost$cpu_state_table	2479	2476	3496					
ost$cs_lock	4946	2667						
ost$cst_trace_control	5842	2511						
ost$debug_code	5964	5952						
ost$debug_list	5960	2619						
ost$debug_list_entry	5951	5960						
ost$debug_mask	5970	2618						
ost$exchange_package	2568	2654						
ost$execute_privilege	2875	2870	2895					
ost$execution_control_block	2653	2493	2679	3497				
ost$external_interrupt_request	5830	2499						
ost$family_name	5533	5528						
ost$flags	2625	2575						
ost$frame_descriptor	5928	5943						
ost$free_running_clock	5018	330	537	702	703	704	705	739	749
		750	751	864	873	1523	2685	5212	5224
ost$global_task_id	2735	332	696	725	860	2488	2664	2665	3336
		3368	4988	6151					
ost$halfword	5788	5781						
ost$idle_type	5806	5801						
ost$key_lock	3877	1627	2901					

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES						
ost$key_lock_value	3883	3880	5866	5868				
ost$keypoint_class	5881	2588	5883					
ost$keypoint_mask	5883	2591						
ost$logical_processor_id	5764	2484						
ost$minimum_save_area	5938	2580	2814	5913				
ost$monitor_condition	5887	5894						
ost$monitor_conditions	5894	2581	2585	3168	3230	5918		
ost$monitor_fault	2810	3245	3273					
ost$monitor_fault_contents	2827	2823						
ost$name	4132	4095	4156	4202	4233	4600	4801	4810	5492
		5531	5533	5984	6048	6187		
ost$p_register	5864	2569	3160	3166	5939			
ost$page_id	5551	5561						
ost$page_size	3748	1732	3729					
ost$page_table_entry	5556	1042	5565					
ost$page_table_index	5549	1893	5565					
ost$paging_statistics	2856	608	937	2694				
ost$parcel	6164	2505	2506					
ost$pre_processed_for_reconfig	5838	2514						
ost$processor_element_id	5780	5761						
ost$processor_element_number	5791	5782						
ost$processor_id	5701	2657	5695					
ost$processor_id_set	5695	2417	2418	2656	5716			
ost$processor_model_number	3678	3672	4698	5783				
ost$processor_serial_number	3756	3673	4697	5784				
ost$pva	3899	2613	2631	2811	3231	5869		
ost$read_privilege	2878	2871	2896					
ost$real_memory_address	2750	2504						
ost$register_number	5860	2610	5924	5932	5933	5934		
ost$ring	3888	1618	1619	2259	2630	2898	2899	3900	6169
ost$ring1_termination_reason	6147	2690						
ost$segment	3890	331	1621	2105	2117	2343	2608	3901	5954
ost$segment_access_control	2868	1640						
ost$segment_descriptor	2893	2225						
ost$segment_length	3894	1623	1625	1647	1649	1652	1975	1977	1995
		2143	3819					
ost$segment_offset	3891	1738	2274	2759	3902	5955	5957		
ost$signature_lock	4947	101	279					
ost$stack_frame_save_area	5912	3274	5946	6182				
ost$status	4382	1634	3161	4367	4594			
ost$status_condition	4406	3090						
ost$status_condition_code	4410	4385	4406					
ost$string	4423	4386						
ost$string_size	4417	4424						
ost$system_flag	6251	3383						
ost$system_virtual_address	2757	1583	1710	1898				
ost$task_index	2741	2518	2736	3294	3307	5034	5035	
ost$task_time_slice	5275	5261						
ost$top_of_stack_pointer	2628	2620						
ost$trap_enable	5907	2577	3157					
ost$user_condition	5897	5904						
ost$user_conditions	5904	2579	2583	3169	5916	5945	6185	

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES						
ost$user_identification	5526	859						
ost$user_name	5531	5527						
ost$valid_relative_pointer	3897	337	340	2014	2683	2684		
ost$valid_ring	3889	2196	2620	4295	4296	4297		
ost$vector_simulation_control	5714	2419						
ost$virtual_machine_identifier	5874	2571	2573	5940				
ost$wait	5581	1996	2054					
ost$write_privilege	2881	2872	2897					
ost$x_register	5861	2610	5924					
pfc$execute	4333	4335	4338					
pfc$read	4332	4335	4338					
pft$share_options	4338	4339						
pft$usage_options	4335	4336						
pft$usage_selections	4336	3926						
pmc$kill_task_flag	6251	6267						
pmc$max_signal_contents	6244	6238						
pmc$max_task_id	6160	6157						
pmt$binary_mainframe_id	3671	21	863					
pmt$condition_identifier	5572	1161						
pmt$cpu_model_number	3738	3727	3734					
pmt$cpu_serial_number	3741	3728	3733					
pmt$initialization_value	5025	329	1631					
pmt$program_name	4233	3934	3938	3954	4003	4221		
pmt$sense_switches	5540	874						
pmt$signal	6200	3337	3369					
pmt$signal_contents	6238	6202						
pmt$signal_id	6205	6201						
pmt$task_id	6157	2681	6152					
pmt$vector_simulation	5721	5715						
pp	3483	3499/M	3500/M	3501/M	3502/M	3503/M	3504/M	3505/M	3506/M
		3507/M	3508/M	3509/M	3510/M	3511/M	3512/M	3513/M	
rmc$external_vsn_size	4769	4775						
rmc$recorded_vsn_size	4772	4782						
rmc$unspecified_file_class	4665	4658						
rmt$external_vsn	4775	4789						
rmt$recorded_vsn	4782	205	3809	4750	4762	4788		
rmt$volume_descriptor	4787	4794						
sft$counter	5161	609	610	875	877	878	880	
sft$file_space_limit_kind	4882	2197	2268	3805				
stt$set_name	4801	3817	4758	4763				
syc$ucr_condition	6173	6184						
syc$user_defined_condition	6174	6186						
syt$180_idle_code	5732	2431	2433	2500				
syt$monitor_flag	3068	3080						
syt$monitor_flags	3080	2655	3302					
syt$monitor_request_code	2950	1972	1992	2012	2028	2051	2077	2141
syt$monitor_status	3088	1973	1993	2013	2029	2052	2078	
to	3484	3499						

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES
t1	3485	3500
t10	3494	3509
t11	3495	3510
t12	3496	3511
t13	3497	3512
t14	3498	3513
t2	3486	3501
t3	3487	3502
t4	3488	3503
t5	3489	3504
t6	3490	3505
t7	3491	3506
t8	3492	3507
t9	3493	3508
tmc$broken_task_fault_id	3252	2816
tmc$btc_invalid_a0	3141	3164
tmc$btc_invalid_p	3141	3164
tmc$btc_mcr_traps_disabled	3142	3165
tmc$btc_mf_traps_disabled	3141	3163
tmc$btc_mntr_fault_buffer_full	3140	3163
tmc$btc_system_error	3143	3159
tmc$btc_ucr_traps_disabled	3142	3165
tmc$dummy_fault	3253	2822
tmc$flag_available_31	6264	6268
tmc$highest_signal_flag_ring	3109	3110
tmc$maximum_monitor_faults	3257	3248
tmc$maximum_signals	3363	3360
tmc$maximum_system_task_id	3391	3394
tmc$mcr_fault	3252	2818
tmc$signal_available_63	6236	6247
tmc$stid_null_task	3397	3394
tmc$task_monitor2_ring	3103	3108
tmc$task_services_ring	3104	3109
tmc$ts_io_wait_not_queued	3472	3454
tmc$ts_page_wait	3474	3455
tmc$ts_ready	3459	3450
tmc$ts_timed_wait_not_queued	3466	3453
tmc$ts_timeout_reqexp_longvlong	3464	3452
tmc$ts_timeout_reqexp_shortshrt	3463	3451
tmt$broken_task_condition	3140	3158
tmt$broken_task_monitor_fault	3156	2817
tmt$dct_entry	3207	3214
tmt$dispatch_control	3191	2495
tmt$dual_state_priority_entry	5853	2482 5850
tmt$handler_execution_ring	6169	3130
tmt$idle_status	3477	3300
tmt$mcr_faults	3229	2819
tmt$monitor_fault_buffer	3242	2692
tmt$monitor_fault_buffers	3248	3243 3244 3245
tmt$monitor_fault_identifiers	3251	2815 3170
tmt$primary_task_list_entry	3293	3311 3498
tmt$ptl_flags	3317	3304

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

Decks referenced by selected decks

IDENTIFIER	DEFINED ON LINE	REFERENCES		
tmt$signal	3335	3351		
tmt$signal_buffer	3348	2693		
tmt$signal_buffers	3360	3349	3350	3351
tmt$system_flags	3383	2668	3303	
tmt$system_task_id	3394	2659		
tmt$task_queue_link	5033	474	1894	3301
tmt$task_status	3459	3194	3298	3299
tmt$wait_inhibited	6195	3319		
tmt$xcb_offset_size	3315	3296		

*** REFERENCE ABBREVIATIONS : M=modify, A=attribute, S=subscript, I=I/O ref, R=read, W=write, P=parameter

```
DFT$SERVER_DESCRIPTOR                Type is record  Length =  41
 HEADER                               Type is field of a record  Length =  41(bytes)  Offset =  0(16) : 0
   DFT$SERVER_DESCRIPTOR_HEADER        Type is record  Length =  41
     SERVER_MAINFRAME_ID                Type is field of a record  Length =  3(bytes)  Offset =  0(16) : 0
       PMT$BINARY_MAINFRAME_ID            Type is record  Length =  3
         MODEL_NUMBER                       Type is field of a record  Length =  1(bytes)  Offset =  0(16) : 0
                                             OST$PROCESSOR_MODEL_NUMBER       Type is subrange  0 ..  255
         SERIAL_NUMBER                      Type is field of a record  Length =  2(bytes)  Offset =  1(16) : 0
                                             OST$PROCESSOR_SERIAL_NUMBER      Type is subrange  0 ..  65535
     SERVED_FAMILY_TABLE_INDEX          Type is field of a record  Length =  2(bytes)  Offset =  3(16) : 0
       DFT$SERVED_FAMILY_TABLE_INDEX      Type is record  Length =  2
         POINTERS_INDEX                     Type is field of a record  Length =  1(bytes)  Offset =  3(16) : 0
                                             DFT$FAMILY_POINTER_INDEX         Type is subrange  1 ..  255
         FAMILY_LIST_INDEX                  Type is field of a record  Length =  1(bytes)  Offset =  4(16) : 0
                                             DFT$SERVED_FAMILY_LIST_INDEX     Type is subrange  1 ..  16
     SERVER_LIFETIME                    Type is field of a record  Length =  2(bytes)  Offset =  5(16) : 0
                                         DFT$LIFETIME                    Type is subrange  0 ..  65535
     READ_WRITE_COUNT                   Type is field of a record  Length =  2(bytes)  Offset =  7(16) : 0
                                          Type is subrange  0 ..  65535
     PURGED                             Type is field of a record  Length =  1(bytes)  Offset =  9(16) : 0
                                          Type is boolean
     HIGHEST_OFFSET_ALLOCATED           Type is field of a record  Length =  6(bytes)  Offset =  A(16) : 0
                                         AMT$FILE_BYTE_ADDRESS           Type is subrange  0 ..  4398046511103
     BYTES_PER_ALLOCATION               Type is field of a record  Length =  3(bytes)  Offset =  10(16) : 0
                                          Type is subrange  0 ..  16777215
     FILE_STATE                         Type is field of a record  Length =  1(bytes)  Offset =  13(16) : 0
                                         DFT$SERVER_STATE                Type is ordinal
                                           DFC$DELETED                     Type is constant  =  6
                                           DFC$TERMINATED                  Type is constant  =  5
                                           DFC$RECOVERING                  Type is constant  =  4
                                           DFC$AWAITING_RECOVERY           Type is constant  =  3
                                           DFC$INACTIVE                    Type is constant  =  2
                                           DFC$DEACTIVATED                 Type is constant  =  1
                                           DFC$ACTIVE                      Type is constant  =  0
      --- Variant --DFC$ACTIVE
       TOTAL_ALLOCATED_LENGTH           Type is field of a record  Length =  6(bytes)  Offset =  14(16) : 0
                                         AMT$FILE_BYTE_ADDRESS           Type is subrange  0 ..  4398046511103
       REMOTE_SFID                      Type is field of a record  Length =  4(bytes)  Offset =  1A(16) : 0
         GFT$SYSTEM_FILE_IDENTIFIER       Type is record  Length =  4
           FILE_ENTRY_INDEX                 Type is field of a record  Length =  2(bytes)  Offset =  1A(16) : 0
                                             GFT$FILE_DESCRIPTOR_INDEX       Type is subrange  0 ..  65535
           RESIDENCE                        Type is field of a record  Length =  1(bytes)  Offset =  1C(16) : 0
                                             GFT$TABLE_RESIDENCE             Type is ordinal
                                               GFC$TR_SYSTEM_WAIT_RECOVERY     Type is constant  =  3
                                               GFC$TR_JOB                      Type is constant  =  2
                                               GFC$TR_SYSTEM                   Type is constant  =  1
                                               GFC$TR_NULL_RESIDENCE           Type is constant  =  0
           FILE_HASH                        Type is field of a record  Length =  1(bytes)  Offset =  1D(16) : 0
                                             Type is subrange  0 ..  255
       ALLOW_OTHER_MAINFRAME_WRITER     Type is field of a record  Length =  1(bytes)  Offset =  1E(16) : 0
                                          Type is boolean
       ALLOCATION_INFO                  Type is field of a record  Length =  7(bytes)  Offset =  1F(16) : 0
         DFT$SERVER_ALLOCATION_INFO       Type is record  Length =  7
           ALLOCATION_NEEDED_ON_SERVER      Type is field of a record  Length =  1(bytes)  Offset =  1F(16) : 0
                                             Type is boolean
            --- Variant --FALSE
             INVALID_DATA                    Type is field of a record  Length =  6(bytes)  Offset =  20(16) : 0
                                              Type is subrange  0 ..  70368744177663
            --- Variant --TRUE
             BYTES_TO_ALLOCATE               Type is field of a record  Length =  6(bytes)  Offset =  20(16) : 0
                                              AMT$FILE_BYTE_ADDRESS           Type is subrange  0 ..  4398046511103
       REQUESTED_TRANSFER_SIZE          Type is field of a record  Length =  3(bytes)  Offset =  26(16) : 0
                                         DMT$TRANSFER_SIZE               Type is subrange  0 ..  16777215
      --- Variant --DFC$AWAITING_RECOVERY
      --- Variant --DFC$TERMINATED
```

```
DMT$DISK_FILE_DESCRIPTOR          Type is record  Length =  87
 READ_WRITE_COUNT                  Type is field of a record  Length =  2(bytes)  Offset =  0(16) : 0
                                     Type is subrange  0 ..  65535
 DELETE_COUNT                      Type is field of a record  Length =  3(bytes)  Offset =  2(16) : 0
                                      DMT$DELETE_COUNT                Type is subrange  0 ..  16777215
 PURGED                            Type is field of a record  Length =  1(bytes)  Offset =  5(16) : 0
                                     Type is boolean
 RESTRICTED_ATTACH                 Type is field of a record  Length =  1(bytes)  Offset =  6(16) : 0
                                     Type is boolean
 BYTES_PER_ALLOCATION              Type is field of a record  Length =  3(bytes)  Offset =  7(16) : 0
                                     Type is subrange  0 ..  16777215
 FILE_ALLOCATION_TABLE             Type is field of a record  Length =  6(bytes)  Offset =  A(16) : 0
                                     Type is pointer  Ptr object length  24576
                                     Pointer to DMT$LEVEL_1_TABLE              llc$cybil_array_kind
 FAT_UPPER_BOUND                   Type is field of a record  Length =  2(bytes)  Offset =  10(16) : 0
                                      DMT$LEVEL_1_INDEX               Type is subrange  0 ..  4095
 CURRENT_FMD_INDEX                 Type is field of a record  Length =  1(bytes)  Offset =  12(16) : 0
                                      DMT$FMD_INDEX                   Type is subrange  0 ..  255
 HIGHEST_OFFSET_ALLOCATED          Type is field of a record  Length =  6(bytes)  Offset =  13(16) : 0
                                      AMT$FILE_BYTE_ADDRESS           Type is subrange  0 ..  4398046511103
 BYTES_PER_LEVEL_2                 Type is field of a record  Length =  6(bytes)  Offset =  19(16) : 0
                                      AMT$FILE_BYTE_ADDRESS           Type is subrange  0 ..  4398046511103
 DFD_MODIFIED                      Type is field of a record  Length =  1(bytes)  Offset =  1F(16) : 0
                                     Type is boolean
 OVERFLOW_ALLOWED                  Type is field of a record  Length =  1(bytes)  Offset =  20(16) : 0
                                     Type is boolean
 REQUESTED_ALLOCATION_SIZE         Type is field of a record  Length =  3(bytes)  Offset =  21(16) : 0
                                      DMT$ALLOCATION_SIZE             Type is subrange  0 ..  16777215
 REQUESTED_CLASS                   Type is field of a record  Length =  1(bytes)  Offset =  24(16) : 0
                                      DMT$CLASS_MEMBER                Type is subrange  65 ..  90
 REQUESTED_CLASS_ORDINAL           Type is field of a record  Length =  1(bytes)  Offset =  25(16) : 0
                                      DMT$CLASS_ORDINAL               Type is subrange  0 ..  63
 REQUESTED_TRANSFER_SIZE           Type is field of a record  Length =  3(bytes)  Offset =  26(16) : 0
                                      DMT$TRANSFER_SIZE               Type is subrange  0 ..  16777215
 REQUESTED_VOLUME                  Type is field of a record  Length =  37(bytes)  Offset =  29(16) : 0
   DMT$REQUESTED_VOLUME              Type is record  Length =  37
     RECORDED_VSN                      Type is field of a record  Length =  6(bytes)  Offset =  29(16) : 0
                                        RMT$RECORDED_VSN              Type is string  String length  6
     SETNAME                           Type is field of a record  Length =  31(bytes)  Offset =  2F(16) : 0
                                        OST$NAME                      Type is string  String length  31
 NUMBER_OF_FMDS                    Type is field of a record  Length =  1(bytes)  Offset =  4E(16) : 0
                                      DMT$FMD_INDEX                   Type is subrange  0 ..  255
 P_FMD                             Type is field of a record  Length =  6(bytes)  Offset =  4F(16) : 0
                                     Type is pointer  Ptr object length  43
                                     Pointer to DMT$FILE_MEDIUM_DESCRIPTOR     llc$record_kind
 FILE_DAMAGED                      Type is field of a record  Length =  1(bytes)  Offset =  55(16) : 0
                                     Type is boolean
 DAMAGED_DETECTION_ENABLED         Type is field of a record  Length =  1(bytes)  Offset =  56(16) : 0
                                     Type is boolean
Cant find - DMT$FILE_ALLOCATION_STATUS
```

```
MT$FILE_MEDIUM_DESCRIPTOR       Type is record  Length =  43
```

```
 IN_USE                            Type is field of a record  Length =  1(bytes)  Offset =  0(16) : 0
                                     Type is boolean
 SYSTEM_FILE_ID                    Type is field of a record  Length =  4(bytes)  Offset =  1(16) : 0
   GFT$SYSTEM_FILE_IDENTIFIER        Type is record  Length =  4
     FILE_ENTRY_INDEX                  Type is field of a record  Length =  2(bytes)  Offset =  1(16) : 0
                                        GFT$FILE_DESCRIPTOR_INDEX     Type is subrange  0 ..  65535
     RESIDENCE                         Type is field of a record  Length =  1(bytes)  Offset =  3(16) : 0
                                        GFT$TABLE_RESIDENCE           Type is ordinal
                                          GFC$TR_SYSTEM_WAIT_RECOVERY    Type is constant  =  3
                                          GFC$TR_JOB                     Type is constant  =  2
                                          GFC$TR_SYSTEM                  Type is constant  =  1
                                          GFC$TR_NULL_RESIDENCE          Type is constant  =  0
     FILE_HASH                       Type is field of a record  Length =  1(bytes)  Offset =  4(16) : 0
                                       Type is subrange  0 ..  255
 AVT_INDEX                         Type is field of a record  Length =  2(bytes)  Offset =  5(16) : 0
                                      DMT$ACTIVE_VOLUME_TABLE_INDEX   Type is subrange  0 ..  65535
 DFL_INDEX                         Type is field of a record  Length =  2(bytes)  Offset =  7(16) : 0
                                      DMT$DEVICE_FILE_LIST_INDEX      Type is subrange  0 ..  65535
 DELETE_LOGGING_COUNT              Type is field of a record  Length =  2(bytes)  Offset =  9(16) : 0
                                      DMT$DELETE_LOGGING_COUNT        Type is subrange  0 ..  65535
 VOLUME_ASSIGNED                   Type is field of a record  Length =  1(bytes)  Offset =  B(16) : 0
                                     Type is boolean
 FMD_ALLOCATED_LENGTH              Type is field of a record  Length =  6(bytes)  Offset =  C(16) : 0
                                      AMT$FILE_BYTE_ADDRESS           Type is subrange  0 ..  4398046511103
 BYTES_PER_MAU                     Type is field of a record  Length =  2(bytes)  Offset =  12(16) : 0
                                      DMT$BYTES_PER_MAU               Type is subrange  0 ..  4096
 DAUS_PER_CYLINDER                 Type is field of a record  Length =  1(bytes)  Offset =  14(16) : 0
                                      DMT$DAUS_PER_POSITION           Type is subrange  0 ..  160
 DAUS_PER_ALLOCATION_UNIT          Type is field of a record  Length =  1(bytes)  Offset =  15(16) : 0
                                      DMT$DAUS_PER_ALLOCATION         Type is subrange  0 ..  160
 INTERNAL_VSN                      Type is field of a record  Length =  11(bytes)  Offset =  16(16) : 0
   OST$BINARY_UNIQUE_NAME            Type is packed record  Length =  11
     SERIAL_NUMBER                     Type is field of a record  Length =  2(bytes)  Offset =  16(16) : 0
                                        OST$PROCESSOR_SERIAL_NUMBER   Type is subrange  0 ..  65535
     MODEL_NUMBER                      Type is field of a record  Length =  1(bytes)  Offset =  18(16) : 0
                                        OST$PROCESSOR_MODEL_NUMBER    Type is subrange  0 ..  255
     YEAR                              Type is field of a record  Length =  11 (bits)  Offset =  19(16) : 0
                                       Type is subrange  1980 ..  2047
     MONTH                             Type is field of a record  Length =  4 (bits)  Offset =  1A(16) : 3
                                       Type is subrange  1 ..  12
     DAY                               Type is field of a record  Length =  5 (bits)  Offset =  1A(16) : 7
                                       Type is subrange  1 ..  31
     HOUR                              Type is field of a record  Length =  5 (bits)  Offset =  1B(16) : 4
                                       Type is subrange  0 ..  23
     MINUTE                            Type is field of a record  Length =  6 (bits)  Offset =  1C(16) : 1
                                       Type is subrange  0 ..  59
     SECOND                            Type is field of a record  Length =  6 (bits)  Offset =  1C(16) : 7
                                       Type is subrange  0 ..  59
     SEQUENCE_NUMBER                   Type is field of a record  Length =  24 (bits)  Offset =  1D(16) : 5
                                       Type is subrange  0 ..  9999999
     FILL                              Type is field of a record  Length =  3 (bits)  Offset =  20(16) : 5
                                       Type is subrange  0 ..  7
 MAUS_PER_DAU                      Type is field of a record  Length =  1(bytes)  Offset =  21(16) : 0
                                      DMT$MAUS_PER_DAU                Type is subrange  2 ..  48
```

```
MAUS_PER_TRANSFER_UNIT          Type is field of a record  Length =  2(bytes)  Offset =  22(16) : 0
                                 DMT$MAUS_PER_TRANSFER          Type is subrange  1 ..  392
P_NEXT_FMD                      Type is field of a record  Length =  6(bytes)  Offset =  24(16) : 0
                                  Type is pointer  Ptr object length  43
                                  Pointer to DMT$FILE_MEDIUM_DESCRIPTOR     11c$record_kind
ALLOCATION_STYLE                Type is field of a record  Length =  1(bytes)  Offset =  2A(16) : 0
                                 DMT$ALLOCATION_STYLES          Type is ordinal
                                   DMC$ACYL                       Type is constant  =  9
                                   DMC$A8                         Type is constant  =  8
                                   DMC$A7                         Type is constant  =  7
                                   DMC$A6                         Type is constant  =  6
                                   DMC$A5                         Type is constant  =  5
                                   DMC$A4                         Type is constant  =  4
                                   DMC$A3                         Type is constant  =  3
                                   DMC$A2                         Type is constant  =  2
                                   DMC$A1                         Type is constant  =  1
                                   DMC$A0                         Type is constant  =  0
```

```
GFT$FILE_DESCRIPTOR_ENTRY       Type is record  Length =  99
 JOB_LOCK                        Type is field of a record  Length =  21(bytes)  Offset =  0(16) : 0
   GFT$SIGNATURE_LOCK              Type is record  Length =  21
     LOCKED                          Type is field of a record  Length =  1(bytes)  Offset =  0(16) : 0
                                       Type is boolean
     COUNT                           Type is field of a record  Length =  1(bytes)  Offset =  1(16) : 0
                                       Type is subrange  0 ..  255
     GTID                            Type is field of a record  Length =  3(bytes)  Offset =  2(16) : 0
       OST$GLOBAL_TASK_ID              Type is record  Length =  3
         INDEX                           Type is field of a record  Length =  2(bytes)  Offset =  2(16) : 0
                                          OST$TASK_INDEX                 Type is subrange  0 ..  4095
         SEQNO                           Type is field of a record  Length =  1(bytes)  Offset =  4(16) : 0
                                           Type is subrange  0 ..  255
     P_REGISTER                      Type is field of a record  Length =  8(bytes)  Offset =  5(16) : 0
                                       Type is integer
     P_REGISTER_2                    Type is field of a record  Length =  8(bytes)  Offset =  D(16) : 0
                                       Type is integer
 MONITOR_LOCK                    Type is field of a record  Length =  1(bytes)  Offset =  15(16) : 0
   MTT$MONITOR_INTERLOCK           Type is packed record  Length =  1
      Length =  0(bytes)  Offset =  15(16) : 0
                                     Type is boolean
      --- Variant -- 0
       BYTE                          Type is field of a record  Length =  1(bytes)  Offset =  15(16) : 0
                                       Type is subrange  0 ..  255
      --- Variant -- 1
       LOCKED                        Type is field of a record  Length =  1 (bits)  Offset =  15(16) : 0
                                       Type is boolean
       ID                            Type is field of a record  Length =  7 (bits)  Offset =  15(16) : 1
                                       Type is subrange  0 ..  127
 FLAGS                           Type is field of a record  Length =  1(bytes)  Offset =  16(16) : 0
   GFT$FDE_FLAGS                   Type is packed record  Length =  1
     EOI_MODIFIED                    Type is field of a record  Length =  1 (bits)  Offset =  16(16) : 0
                                       Type is boolean
     WIRE_EOI_PAGE                   Type is field of a record  Length =  1 (bits)  Offset =  16(16) : 1
                                       Type is boolean
     ACTIVE_FILE                     Type is field of a record  Length =  1 (bits)  Offset =  16(16) : 2
                                       Type is boolean
     GLOBAL_TEMPLATE_FILE            Type is field of a record  Length =  1 (bits)  Offset =  16(16) : 3
                                       Type is boolean
     FDE_SPARE_4                     Type is field of a record  Length =  1 (bits)  Offset =  16(16) : 4
                                       Type is boolean
     FDE_SPARE_5                     Type is field of a record  Length =  1 (bits)  Offset =  16(16) : 5
                                       Type is boolean
     FDE_SPARE_6                     Type is field of a record  Length =  1 (bits)  Offset =  16(16) : 6
                                       Type is boolean
     FDE_SPARE_7                     Type is field of a record  Length =  1 (bits)  Offset =  16(16) : 7
                                       Type is boolean
 GLOBAL_FILE_NAME                Type is field of a record  Length =  11(bytes)  Offset =  17(16) : 0
   OST$BINARY_UNIQUE_NAME          Type is packed record  Length =  11
     SERIAL_NUMBER                   Type is field of a record  Length =  2(bytes)  Offset =  17(16) : 0
                                      OST$PROCESSOR_SERIAL_NUMBER      Type is subrange  0 ..  65535
     MODEL_NUMBER                    Type is field of a record  Length =  1(bytes)  Offset =  19(16) : 0
                                      OST$PROCESSOR_MODEL_NUMBER       Type is subrange  0 ..  255
     YEAR                            Type is field of a record  Length =  11 (bits)  Offset =  1A(16) : 0
                                       Type is subrange  1980 ..  2047
     MONTH                           Type is field of a record  Length =  4 (bits)  Offset =  1B(16) : 3
                                       Type is subrange  1 ..  12
     DAY                             Type is field of a record  Length =  5 (bits)  Offset =  1B(16) : 7
                                       Type is subrange  1 ..  31
     HOUR                            Type is field of a record  Length =  5 (bits)  Offset =  1C(16) : 4
                                       Type is subrange  0 ..  23
     MINUTE                          Type is field of a record  Length =  6 (bits)  Offset =  1D(16) : 1
                                       Type is subrange  0 ..  59
     SECOND                          Type is field of a record  Length =  6 (bits)  Offset =  1D(16) : 7
```

```
                                    Type is subrange  0 ..  59
    SEQUENCE_NUMBER                 Type is field of a record  Length =  24 (bits)  Offset =  1E(16) : 5
                                      Type is subrange  0 ..  9999999
    FILL                            Type is field of a record  Length =  3 (bits)  Offset =  21(16) : 5
                                      Type is subrange  0 ..  7
FILE_HASH_THREAD                  Type is field of a record  Length =  6(bytes)  Offset =  22(16) : 0
                                    Type is pointer  Ptr object length  99
                                    Pointer to GFT$FILE_DESCRIPTOR_ENTRY      11c$record_kind
ATTACHED_IN_WRITE_COUNT           Type is field of a record  Length =  2(bytes)  Offset =  28(16) : 0
                                    GFT$ATTACH_COUNT                Type is subrange  0 ..  65535
ATTACH_COUNT                      Type is field of a record  Length =  2(bytes)  Offset =  2A(16) : 0
                                    GFT$ATTACH_COUNT                Type is subrange  0 ..  65535
OPEN_COUNT                        Type is field of a record  Length =  4(bytes)  Offset =  2C(16) : 0
                                    GFT$OPEN_COUNT                  Type is subrange  0 ..  4294967295
FILE_KIND                         Type is field of a record  Length =  1(bytes)  Offset =  30(16) : 0
                                    GFT$FILE_KIND                   Type is ordinal
                                      GFC$FK_MONITOR_ONLY_UNNAMED     Type is constant  =  8
                                      GFC$FK_GLOBAL_UNNAMED           Type is constant  =  7
                                      GFC$FK_UNNAMED_FILE             Type is constant  =  6
                                      GFC$FK_JOB_LOCAL_FILE           Type is constant  =  5
                                      GFC$FK_CATALOG                  Type is constant  =  4
                                      GFC$FK_SAVE_3                   Type is constant  =  3
                                      GFC$FK_SAVE_2                   Type is constant  =  2
                                      GFC$FK_DEVICE_FILE              Type is constant  =  1
                                      GFC$FK_JOB_PERMANENT_FILE       Type is constant  =  0
FILE_HASH                         Type is field of a record  Length =  1(bytes)  Offset =  31(16) : 0
                                    Type is subrange  0 ..  255
SEGMENT_LOCK                      Type is field of a record  Length =  9(bytes)  Offset =  32(16) : 0
  GFT$SEGMENT_LOCK_INFO             Type is record  Length =  9
    LOCKED_FOR_READ                   Type is field of a record  Length =  4(bytes)  Offset =  32(16) : 0
                                      GFT$OPEN_COUNT                Type is subrange  0 ..  4294967295
    LOCKED_FOR_WRITE                  Type is field of a record  Length =  1(bytes)  Offset =  36(16) : 0
                                      Type is boolean
    TASK_QUEUE                        Type is field of a record  Length =  4(bytes)  Offset =  37(16) : 0
      TMT$TASK_QUEUE_LINK               Type is record  Length =  4
        HEAD                              Type is field of a record  Length =  2(bytes)  Offset =  37(16) : 0
                                          OST$TASK_INDEX                Type is subrange  0 ..  4095
        TAIL                              Type is field of a record  Length =  2(bytes)  Offset =  39(16) : 0
                                          OST$TASK_INDEX                Type is subrange  0 ..  4095
ASTI                              Type is field of a record  Length =  2(bytes)  Offset =  3B(16) : 0
                                    MMT$AST_INDEX                 Type is subrange  0 ..  65535
EOI_BYTE_ADDRESS                  Type is field of a record  Length =  6(bytes)  Offset =  3D(16) : 0
                                    AMT$FILE_BYTE_ADDRESS         Type is subrange  0 ..  4398046511103
EOI_STATE                         Type is field of a record  Length =  1(bytes)  Offset =  43(16) : 0
                                    MMT$EOI_STATE                 Type is ordinal
                                      MMC$EOI_UNCERTAIN             Type is constant  =  2
                                      MMC$EOI_ROUNDED               Type is constant  =  1
                                      MMC$EOI_ACTUAL                Type is constant  =  0
ALLOCATION_UNIT_SIZE              Type is field of a record  Length =  3(bytes)  Offset =  44(16) : 0
                                    GFT$ALLOCATION_UNIT_SIZE      Type is subrange  0 ..  1000000
TRANSFER_UNIT_SIZE                Type is field of a record  Length =  3(bytes)  Offset =  47(16) : 0
                                    GFT$TRANSFER_UNIT_SIZE        Type is subrange  0 ..  10000000
FILE_LIMIT                        Type is field of a record  Length =  6(bytes)  Offset =  4A(16) : 0
                                    AMT$FILE_LIMIT                Type is subrange  0 ..  4398046511103
QUEUE_STATUS                      Type is field of a record  Length =  1(bytes)  Offset =  50(16) : 0
                                    GFT$QUEUE_STATUS              Type is ordinal
                                      GFC$QS_JOB_WORKING_SET        Type is constant  =  2
                                      GFC$QS_JOB_SHARED             Type is constant  =  1
                                      GFC$QS_GLOBAL_SHARED          Type is constant  =  0
PRESET_VALUE                      Type is field of a record  Length =  1(bytes)  Offset =  51(16) : 0
                                    PMT$INITIALIZATION_VALUE      Type is ordinal
                                      PMC$INITIALIZE_TO_INFINITY    Type is constant  =  3
                                      PMC$INITIALIZE_TO_INDEFINITE  Type is constant  =  2
                                      PMC$INITIALIZE_TO_ALT_ONES    Type is constant  =  1
                                      PMC$INITIALIZE_TO_ZERO        Type is constant  =  0
TIME_LAST_MODIFIED                Type is field of a record  Length =  6(bytes)  Offset =  52(16) : 0
                                    OST$FREE_RUNNING_CLOCK        Type is subrange  0 ..  281474976710655
LAST_SEGMENT_NUMBER               Type is field of a record  Length =  2(bytes)  Offset =  58(16) : 0
                                    OST$SEGMENT                   Type is subrange  0 ..  4095
GLOBAL_TASK_ID                    Type is field of a record  Length =  3(bytes)  Offset =  5A(16) : 0
  OST$GLOBAL_TASK_ID                Type is record  Length =  3
    INDEX                             Type is field of a record  Length =  2(bytes)  Offset =  5A(16) : 0
                                      OST$TASK_INDEX                Type is subrange  0 ..  4095
    SEQNO                             Type is field of a record  Length =  1(bytes)  Offset =  5C(16) : 0
                                      Type is subrange  0 ..  255
STACK_FOR_RING                    Type is field of a record  Length =  1(bytes)  Offset =  5D(16) : 0
                                    Type is subrange  0 ..  15
MEDIA                             Type is field of a record  Length =  1(bytes)  Offset =  5E(16) : 0
                                    GFT$FILE_MEDIA                Type is ordinal
                                      GFC$FM_SERVED_FILE            Type is constant  =  2
                                      GFC$FM_MASS_STORAGE_FILE      Type is constant  =  1
                                      GFC$FM_TRANSIENT_SEGMENT      Type is constant  =  0
 --- Variant --GFC$FM_MASS_STORAGE_FILE
  DISK_FILE_DESCRIPTOR_P          Type is field of a record  Length =  4(bytes)  Offset =  5F(16) : 0
                                    OST$VALID_RELATIVE_POINTER    Type is subrange  0 ..  2147483647
 --- Variant --GFC$FM_SERVED_FILE
  SERVED_FILE_DESCRIPTOR_P        Type is field of a record  Length =  4(bytes)  Offset =  5F(16) : 0
                                    OST$VALID_RELATIVE_POINTER    Type is subrange  0 ..  2147483647
```

```
JMT$ACTIVE_JOB_LIST_ENTRY       Type is record  Length =  18
 IN_USE                          Type is field of a record  Length =  3(bytes)  Offset =  0(16) : 0
                                   Type is subrange  0 ..  16777215
 IJL_ORDINAL                     Type is field of a record  Length =  2(bytes)  Offset =  3(16) : 0
   JMT$IJL_ORDINAL                 Type is packed record  Length =  2
     BLOCK_NUMBER                    Type is field of a record  Length =  11 (bits)  Offset =  3(16) : 0
                                      JMT$IJL_BLOCK_NUMBER          Type is subrange  0 ..  2047
     BLOCK_INDEX                     Type is field of a record  Length =  5 (bits)  Offset =  4(16) : 3
                                      JMT$IJL_BLOCK_INDEX           Type is subrange  0 ..  31
 IJLE_P                          Type is field of a record  Length =  6(bytes)  Offset =  5(16) : 0
                                   Type is pointer  Ptr object length  342
                                   Pointer to JMT$INITIATED_JOB_LIST_ENTRY   11c$record_kind
 JOB_IS_GOOD_SWAP_CANDIDATE      Type is field of a record  Length =  1(bytes)  Offset =  B(16) : 0
                                   Type is boolean
 TIME_FREED                      Type is field of a record  Length =  6(bytes)  Offset =  C(16) : 0
                                  OST$FREE_RUNNING_CLOCK          Type is subrange  0 ..  281474976710655
```

```
JMT$INITIATED_JOB_LIST_ENTRY    Type is record  Length =  342
 SYSTEM_SUPPLIED_NAME            Type is field of a record  Length =  19(bytes)  Offset =  0(16) : 0
                                  JMT$SYSTEM_SUPPLIED_NAME        Type is string  String length  19
 JOB_NAME                        Type is field of a record  Length =  8(bytes)  Offset =  13(16) : 0
                                   Type is string  String length  8
 ENTRY_STATUS                    Type is field of a record  Length =  1(bytes)  Offset =  1B(16) : 0
                                  JMT$IJL_ENTRY_STATUS            Type is ordinal
                                    JMC$IES_SWAPIN_CANDIDATE        Type is constant  =  10
                                    JMC$IES_READY_TASK              Type is constant  =  9
                                    JMC$IES_JOB_DAMAGED             Type is constant  =  8
                                    JMC$IES_SYSTEM_FORCE_OUT        Type is constant  =  7
                                    JMC$IES_OPERATOR_FORCE_OUT      Type is constant  =  6
                                    JMC$IES_JOB_SWAPPED             Type is constant  =  5
                                    JMC$IES_SWAPIN_IN_PROGRESS      Type is constant  =  4
                                    JMC$IES_JOB_IN_MEMORY           Type is constant  =  3
                                    JMC$IES_JOB_IN_MEMORY_NON_SWAP  Type is constant  =  2
                                    JMC$IES_JOB_TERMINATING         Type is constant  =  1
                                    JMC$IES_ENTRY_FREE              Type is constant  =  0
 AJL_ORDINAL                     Type is field of a record  Length =  1(bytes)  Offset =  1C(16) : 0
                                  JMT$AJL_ORDINAL                 Type is subrange  0 ..  255
 KJL_ORDINAL                     Type is field of a record  Length =  2(bytes)  Offset =  1D(16) : 0
                                  JMT$KJL_INDEX                   Type is subrange  0 ..  65535
 SWAP_STATUS                     Type is field of a record  Length =  1(bytes)  Offset =  1F(16) : 0
                                  JMT$IJL_SWAP_STATUS             Type is ordinal
                                    JMC$ISS_SWAPIN_IO_COMPLETE      Type is constant  =  25
                                    JMC$ISS_SWAPIN_IO_INITIATED     Type is constant  =  24
                                    JMC$ISS_WAIT_SWAPIN_IO_INIT     Type is constant  =  23
                                    JMC$ISS_SWAPIN_RESOURCE_CLAIMED Type is constant  =  22
                                    JMC$ISS_SWAPIN_REQUESTED        Type is constant  =  21
                                    JMC$ISS_SWAPOUT_COMPLETE        Type is constant  =  20
                                    JMC$ISS_FREE_SWAPPED_MEMORY     Type is constant  =  19
                                    JMC$ISS_SWAPPED_IO_COMPLETE     Type is constant  =  18
                                    JMC$ISS_SWAPOUT_IO_COMPLETE     Type is constant  =  17
                                    JMC$ISS_SWAPOUT_IO_INITIATED    Type is constant  =  16
                                    JMC$ISS_WAIT_SWAPOUT_IO_INIT    Type is constant  =  15
                                    JMC$ISS_INITIATE_SWAPOUT_IO     Type is constant  =  14
                                    JMC$ISS_SWAPPED_IO_CANNOT_INIT  Type is constant  =  13
                                    JMC$ISS_ALLOCATE_SFD            Type is constant  =  12
                                    JMC$ISS_WAIT_ALLOCATE_SFD       Type is constant  =  11
                                    JMC$ISS_JOB_IO_COMPLETE         Type is constant  =  10
                                    JMC$ISS_WAIT_JOB_IO_COMPLETE    Type is constant  =  9
                                    JMC$ISS_ALLOCATE_SWAP_FILE      Type is constant  =  8
                                    JMC$ISS_WAIT_ALLOCATE_SWAP_FILE Type is constant  =  7
                                    JMC$ISS_JOB_ALLOCATE_SWAP_FILE  Type is constant  =  6
                                    JMC$ISS_FLUSH_AM_PAGES          Type is constant  =  5
                                    JMC$ISS_SWAPPED_NO_IO           Type is constant  =  4
                                    JMC$ISS_JOB_IDLE_TASKS_COMPLETE Type is constant  =  3
                                    JMC$ISS_IDLE_TASKS_INITIATED    Type is constant  =  2
                                    JMC$ISS_EXECUTING               Type is constant  =  1
                                    JMC$ISS_NULL                    Type is constant  =  0
 NEXT_SWAP_STATUS                Type is field of a record  Length =  1(bytes)  Offset =  20(16) : 0
                                  JMT$IJL_SWAP_STATUS             Type is ordinal
                                    JMC$ISS_SWAPIN_IO_COMPLETE      Type is constant  =  25
                                    JMC$ISS_SWAPIN_IO_INITIATED     Type is constant  =  24
                                    JMC$ISS_WAIT_SWAPIN_IO_INIT     Type is constant  =  23
                                    JMC$ISS_SWAPIN_RESOURCE_CLAIMED Type is constant  =  22
                                    JMC$ISS_SWAPIN_REQUESTED        Type is constant  =  21
                                    JMC$ISS_SWAPOUT_COMPLETE        Type is constant  =  20
                                    JMC$ISS_FREE_SWAPPED_MEMORY     Type is constant  =  19
                                    JMC$ISS_SWAPPED_IO_COMPLETE     Type is constant  =  18
                                    JMC$ISS_SWAPOUT_IO_COMPLETE     Type is constant  =  17
                                    JMC$ISS_SWAPOUT_IO_INITIATED    Type is constant  =  16
                                    JMC$ISS_WAIT_SWAPOUT_IO_INIT    Type is constant  =  15
                                    JMC$ISS_INITIATE_SWAPOUT_IO     Type is constant  =  14
```

```
                                JMC$ISS_SWAPPED_IO_CANNOT_INIT  Type is constant  =  13
                                JMC$ISS_ALLOCATE_SFD            Type is constant  =  12
                                JMC$ISS_WAIT_ALLOCATE_SFD       Type is constant  =  11
                                JMC$ISS_JOB_IO_COMPLETE         Type is constant  =  10
                                JMC$ISS_WAIT_JOB_IO_COMPLETE    Type is constant  =  9
                                JMC$ISS_ALLOCATE_SWAP_FILE      Type is constant  =  8
                                JMC$ISS_WAIT_ALLOCATE_SWAP_FILE Type is constant  =  7
                                JMC$ISS_JOB_ALLOCATE_SWAP_FILE  Type is constant  =  6
                                JMC$ISS_FLUSH_AM_PAGES          Type is constant  =  5
                                JMC$ISS_SWAPPED_NO_IO           Type is constant  =  4
                                JMC$ISS_JOB_IDLE_TASKS_COMPLETE Type is constant  =  3
                                JMC$ISS_IDLE_TASKS_INITIATED    Type is constant  =  2
                                JMC$ISS_EXECUTING               Type is constant  =  1
                                JMC$ISS_NULL                    Type is constant  =  0
LAST_SWAP_STATUS                Type is field of a record  Length =  1(bytes)  Offset =  21(16) : 0
                                 JMT$IJL_SWAP_STATUS              Type is ordinal
                                   JMC$ISS_SWAPIN_IO_COMPLETE     Type is constant  =  25
                                   JMC$ISS_SWAPIN_IO_INITIATED    Type is constant  =  24
                                   JMC$ISS_WAIT_SWAPIN_IO_INIT    Type is constant  =  23
                                   JMC$ISS_SWAPIN_RESOURCE_CLAIMED Type is constant  =  22
                                   JMC$ISS_SWAPIN_REQUESTED       Type is constant  =  21
                                   JMC$ISS_SWAPOUT_COMPLETE       Type is constant  =  20
                                   JMC$ISS_FREE_SWAPPED_MEMORY    Type is constant  =  19
                                   JMC$ISS_SWAPPED_IO_COMPLETE    Type is constant  =  18
                                   JMC$ISS_SWAPOUT_IO_COMPLETE    Type is constant  =  17
                                   JMC$ISS_SWAPOUT_IO_INITIATED   Type is constant  =  16
                                   JMC$ISS_WAIT_SWAPOUT_IO_INIT   Type is constant  =  15
                                   JMC$ISS_INITIATE_SWAPOUT_IO    Type is constant  =  14
                                   JMC$ISS_SWAPPED_IO_CANNOT_INIT Type is constant  =  13
                                   JMC$ISS_ALLOCATE_SFD           Type is constant  =  12
                                   JMC$ISS_WAIT_ALLOCATE_SFD      Type is constant  =  11
                                   JMC$ISS_JOB_IO_COMPLETE        Type is constant  =  10
                                   JMC$ISS_WAIT_JOB_IO_COMPLETE   Type is constant  =  9
                                   JMC$ISS_ALLOCATE_SWAP_FILE     Type is constant  =  8
                                   JMC$ISS_WAIT_ALLOCATE_SWAP_FILE Type is constant  =  7
                                   JMC$ISS_JOB_ALLOCATE_SWAP_FILE Type is constant  =  6
                                   JMC$ISS_FLUSH_AM_PAGES         Type is constant  =  5
                                   JMC$ISS_SWAPPED_NO_IO          Type is constant  =  4
                                   JMC$ISS_JOB_IDLE_TASKS_COMPLETE Type is constant  =  3
                                   JMC$ISS_IDLE_TASKS_INITIATED   Type is constant  =  2
                                   JMC$ISS_EXECUTING              Type is constant  =  1
                                   JMC$ISS_NULL                   Type is constant  =  0
INHIBIT_SWAP_COUNT              Type is field of a record  Length =  3(bytes)  Offset =  22(16) : 0
                                  Type is subrange  0 ..  16777215
ACTIVE_IO_PAGE_COUNT            Type is field of a record  Length =  3(bytes)  Offset =  25(16) : 0
                                  Type is subrange  0 ..  16777215
ACTIVE_IO_REQUESTS              Type is field of a record  Length =  2(bytes)  Offset =  28(16) : 0
                                  Type is subrange  0 ..  65535
SWAP_QUEUE_LINK                 Type is field of a record  Length =  5(bytes)  Offset =  2A(16) : 0
  JST$IJL_SWAP_QUEUE_LINK         Type is record  Length =  5
    QUEUE_ID                        Type is field of a record  Length =  1(bytes)  Offset =  2A(16) : 0
                                     JST$IJL_SWAP_QUEUE_ID          Type is ordinal
                                       JSC$ISQI_SWAPPED_OUT           Type is constant  =  5
                                       JSC$ISQI_SWAPPED_IO_COMPLETED  Type is constant  =  4
                                       JSC$ISQI_SWAPPED_IO_CANNOT_INIT Type is constant  =  3
                                       JSC$ISQI_SWAPPED_IO_NOT_INIT   Type is constant  =  2
                                       JSC$ISQI_SWAPPING              Type is constant  =  1
                                       JSC$ISQI_NULL                  Type is constant  =  0
    BACKWARD_LINK                   Type is field of a record  Length =  2(bytes)  Offset =  2B(16) : 0
      JMT$IJL_ORDINAL                 Type is packed record  Length =  2
        BLOCK_NUMBER                    Type is field of a record  Length =  11 (bits)  Offset =  2B(16) : 0
                                         JMT$IJL_BLOCK_NUMBER           Type is subrange  0 ..  2047
        BLOCK_INDEX                     Type is field of a record  Length =  5 (bits)  Offset =  2C(16) : 3
                                         JMT$IJL_BLOCK_INDEX            Type is subrange  0 ..  31
    FORWARD_LINK                    Type is field of a record  Length =  2(bytes)  Offset =  2D(16) : 0
      JMT$IJL_ORDINAL                 Type is packed record  Length =  2
        BLOCK_NUMBER                    Type is field of a record  Length =  11 (bits)  Offset =  2D(16) : 0
                                         JMT$IJL_BLOCK_NUMBER           Type is subrange  0 ..  2047
        BLOCK_INDEX                     Type is field of a record  Length =  5 (bits)  Offset =  2E(16) : 3
                                         JMT$IJL_BLOCK_INDEX            Type is subrange  0 ..  31
JOB_FIXED_ASID                  Type is field of a record  Length =  2(bytes)  Offset =  2F(16) : 0
                                 OST$ASID                       Type is subrange  0 ..  65535
LONG_WAIT_AGING_COMPLETE        Type is field of a record  Length =  1(bytes)  Offset =  31(16) : 0
                                  Type is boolean
NOTIFY_SWAPPER_WHEN_IO_COMPLETE Type is field of a record  Length =  1(bytes)  Offset =  32(16) : 0
                                  Type is boolean
SCHEDULING_DISPATCHING_PRIORITY Type is field of a record  Length =  1(bytes)  Offset =  33(16) : 0
                                 JMT$DISPATCHING_PRIORITY       Type is subrange  0 ..  15
DISPATCHING_CONTROL             Type is field of a record  Length =  18(bytes)  Offset =  34(16) : 0
  JMT$IJL_DISPATCHING_CONTROL     Type is record  Length =  18
    DISPATCHING_CONTROL_INDEX       Type is field of a record  Length =  1(bytes)  Offset =  34(16) : 0
                                     JMT$DISPATCHING_CONTROL_INDEX  Type is subrange  1 ..  5
    DISPATCHING_PRIORITY            Type is field of a record  Length =  1(bytes)  Offset =  35(16) : 0
                                     JMT$DISPATCHING_PRIORITY       Type is subrange  0 ..  15
    USER_REQUESTED_DISPATCHING_PRIO Type is field of a record  Length =  1(bytes)  Offset =  36(16) : 0
                                     JMT$DISPATCHING_PRIORITY       Type is subrange  0 ..  15
    OPERATOR_SET_DISPATCHING_PRIO   Type is field of a record  Length =  1(bytes)  Offset =  37(16) : 0
                                     JMT$DISPATCHING_PRIORITY       Type is subrange  0 ..  15
    SERVICE_REMAINING               Type is field of a record  Length =  6(bytes)  Offset =  38(16) : 0
                                     OST$FREE_RUNNING_CLOCK         Type is subrange  0 ..  281474976710655
    CP_SERVICE_AT_CLASS_SWITCH      Type is field of a record  Length =  8(bytes)  Offset =  3E(16) : 0
                                      Type is integer
JOB_MONITOR_TASKID              Type is field of a record  Length =  3(bytes)  Offset =  46(16) : 0
  OST$GLOBAL_TASK_ID              Type is record  Length =  3
    INDEX                           Type is field of a record  Length =  2(bytes)  Offset =  46(16) : 0
                                     OST$TASK_INDEX                 Type is subrange  0 ..  4095
    SEQNO                           Type is field of a record  Length =  1(bytes)  Offset =  48(16) : 0
                                      Type is subrange  0 ..  255
JOB_MODE                        Type is field of a record  Length =  1(bytes)  Offset =  49(16) : 0
                                 JMT$JOB_MODE                   Type is ordinal
                                   JMC$INTERACTIVE_SYS_DISCONNECT  Type is constant  =  4
                                   JMC$INTERACTIVE_LINE_DISCONNECT Type is constant  =  3
                                   JMC$INTERACTIVE_CMND_DISCONNECT Type is constant  =  2
                                   JMC$INTERACTIVE_CONNECTED       Type is constant  =  1
                                   JMC$BATCH                       Type is constant  =  0
EXECUTING_TASK_COUNT            Type is field of a record  Length =  1(bytes)  Offset =  4A(16) : 0
                                  Type is subrange  0 ..  255
MULTIPROCESSING_ALLOWED         Type is field of a record  Length =  1(bytes)  Offset =  4B(16) : 0
                                  Type is boolean
MEMORY_RESERVE_REQUEST          Type is field of a record  Length =  5(bytes)  Offset =  4C(16) : 0
  MMT$MEMORY_RESERVE_REQUEST      Type is record  Length =  5
    SWAPOUT_JOB                     Type is field of a record  Length =  1(bytes)  Offset =  4C(16) : 0
                                      Type is boolean
    REQUESTED_PAGE_COUNT            Type is field of a record  Length =  2(bytes)  Offset =  4D(16) : 0
                                     MMT$PAGE_FRAME_INDEX           Type is subrange  0 ..  65534
    RESERVED_PAGE_COUNT             Type is field of a record  Length =  2(bytes)  Offset =  4F(16) : 0
                                     MMT$PAGE_FRAME_INDEX           Type is subrange  0 ..  65534
SWAPIN_CANDIDATE_QUEUE          Type is field of a record  Length =  2(bytes)  Offset =  51(16) : 0
  JMT$IJL_ORDINAL                 Type is packed record  Length =  2
    BLOCK_NUMBER                    Type is field of a record  Length =  11 (bits)  Offset =  51(16) : 0
                                     JMT$IJL_BLOCK_NUMBER           Type is subrange  0 ..  2047
    BLOCK_INDEX                     Type is field of a record  Length =  5 (bits)  Offset =  52(16) : 3
                                     JMT$IJL_BLOCK_INDEX            Type is subrange  0 ..  31
ESTIMATED_READY_TIME            Type is field of a record  Length =  6(bytes)  Offset =  53(16) : 0
                                 OST$FREE_RUNNING_CLOCK         Type is subrange  0 ..  281474976710655
LAST_THINK_TIME                 Type is field of a record  Length =  6(bytes)  Offset =  59(16) : 0
                                 OST$FREE_RUNNING_CLOCK         Type is subrange  0 ..  281474976710655
AGE_PURGE_TIMESTAMP             Type is field of a record  Length =  6(bytes)  Offset =  5F(16) : 0
                                 OST$FREE_RUNNING_CLOCK         Type is subrange  0 ..  281474976710655
SFD_PURGE_TIMESTAMP             Type is field of a record  Length =  6(bytes)  Offset =  65(16) : 0
                                 OST$FREE_RUNNING_CLOCK         Type is subrange  0 ..  281474976710655
JOB_SCHEDULER_DATA              Type is field of a record  Length =  51(bytes)  Offset =  6B(16) : 0
```

```
JMT$SCHEDULING_DATA                Type is record  Length =  51
  READY_TASK_LINK                    Type is field of a record  Length =  2(bytes)  Offset =  6B(16) : 0
    JMT$IJL_ORDINAL                    Type is packed record  Length =  2
      BLOCK_NUMBER                       Type is field of a record  Length =  11 (bits)  Offset =  6B(16) : 0
                                          JMT$IJL_BLOCK_NUMBER              Type is subrange  0 ..  2047
      BLOCK_INDEX                        Type is field of a record  Length =  5 (bits)  Offset =  6C(16) : 3
                                          JMT$IJL_BLOCK_INDEX               Type is subrange  0 ..  31
  SERVICE_ACCUMULATOR                Type is field of a record  Length =  6(bytes)  Offset =  6D(16) : 0
                                      JMT$SERVICE_ACCUMULATOR           Type is subrange  0 ..  281474976710655
  SERVICE_ACCUMULATOR_SINCE_SWAP     Type is field of a record  Length =  6(bytes)  Offset =  73(16) : 0
                                      JMT$SERVICE_ACCUMULATOR           Type is subrange  0 ..  281474976710655
  GUARANTEED_SERVICE_REMAINING       Type is field of a record  Length =  6(bytes)  Offset =  79(16) : 0
                                      JMT$SERVICE_ACCUMULATOR           Type is subrange  0 ..  281474976710655
  LAST_CPTIME                        Type is field of a record  Length =  5(bytes)  Offset =  7F(16) : 0
                                      OST$CP_TIME_VALUE                 Type is subrange  0 ..  1099511627775
  LAST_PAGE_FAULT_COUNT              Type is field of a record  Length =  5(bytes)  Offset =  84(16) : 0
                                        Type is subrange  0 ..  1099511627775
  JOB_SWAP_COUNTS                    Type is field of a record  Length =  6(bytes)  Offset =  89(16) : 0
    JMT$IJL_SWAP_COUNTS                Type is record  Length =  6
      LONG_WAIT                          Type is field of a record  Length =  3(bytes)  Offset =  89(16) : 0
                                          JMT$IJL_SWAP_COUNT                Type is subrange  0 ..  16777215
      JOB_MODE                           Type is field of a record  Length =  3(bytes)  Offset =  8C(16) : 0
                                          JMT$IJL_SWAP_COUNT                Type is subrange  0 ..  16777215
  SWAPOUT_REASON                     Type is field of a record  Length =  1(bytes)  Offset =  8F(16) : 0
                                      JMT$SWAPOUT_REASONS               Type is ordinal
                                        JMC$SR_JOB_DAMAGED                Type is constant  =  8
                                        JMC$SR_IDLE_DISPATCHING           Type is constant  =  7
                                        JMC$SR_MEMORY_RESERVE_REQUEST     Type is constant  =  6
                                        JMC$SR_LONG_WAIT                  Type is constant  =  5
                                        JMC$SR_IDLING_SYSTEM_SWAPOUT      Type is constant  =  4
                                        JMC$SR_LOWER_PRIORITY             Type is constant  =  3
                                        JMC$SR_THRASHING                  Type is constant  =  2
                                        JMC$SR_OPERATOR_REQUEST           Type is constant  =  1
                                        JMC$SR_NULL                       Type is constant  =  0
  PRIORITY                           Type is field of a record  Length =  3(bytes)  Offset =  90(16) : 0
                                      JMT$JOB_PRIORITY                  Type is subrange  0 ..  16777215
  UNAGED_SWAP_QUEUE_PRIORITY         Type is field of a record  Length =  3(bytes)  Offset =  93(16) : 0
                                      JMT$JOB_PRIORITY                  Type is subrange  0 ..  16777215
  SWAPIN_Q_PRIORITY_TIMESTAMP        Type is field of a record  Length =  6(bytes)  Offset =  96(16) : 0
                                      OST$FREE_RUNNING_CLOCK            Type is subrange  0 ..  281474976710655
  JOB_CLASS                          Type is field of a record  Length =  1(bytes)  Offset =  9C(16) : 0
                                      JMT$JOB_CLASS                     Type is subrange  0 ..  255
  SERVICE_CLASS                      Type is field of a record  Length =  1(bytes)  Offset =  9D(16) : 0
                                      JMT$SERVICE_CLASS_INDEX           Type is subrange  0 ..  255
JOB_PAGE_QUEUE_LIST                Type is field of a record  Length =  18(bytes)  Offset =  9E(16) : 0
                                    MMT$JOB_PAGE_QUEUE_LIST         Type is cybil array  Array element length  6
  MMT$PAGE_QUEUE_LIST_ENTRY          Type is record  Length =  6
    LINK                               Type is field of a record  Length =  4(bytes)  Offset =  9E(16) : 0
      MMT$LINK                           Type is record  Length =  4
        BKW                                Type is field of a record  Length =  2(bytes)  Offset =  9E(16) : 0
                                            MMT$PAGE_FRAME_INDEX            Type is subrange  0 ..  65534
        FWD                                Type is field of a record  Length =  2(bytes)  Offset =  A0(16) : 0
                                            MMT$PAGE_FRAME_INDEX            Type is subrange  0 ..  65534
    COUNT                              Type is field of a record  Length =  2(bytes)  Offset =  A2(16) : 0
                                        Type is subrange  0 ..  65535
SWAP_DATA                          Type is field of a record  Length =  39(bytes)  Offset =  B0(16) : 0
  JMT$SWAP_DATA                      Type is record  Length =  39
    SWAP_FILE_SFID                     Type is field of a record  Length =  4(bytes)  Offset =  B0(16) : 0
      GFT$SYSTEM_FILE_IDENTIFIER         Type is record  Length =  4
        FILE_ENTRY_INDEX                   Type is field of a record  Length =  2(bytes)  Offset =  B0(16) : 0
                                            GFT$FILE_DESCRIPTOR_INDEX       Type is subrange  0 ..  65535
        RESIDENCE                          Type is field of a record  Length =  1(bytes)  Offset =  B2(16) : 0
                                            GFT$TABLE_RESIDENCE             Type is ordinal
                                              GFC$TR_SYSTEM_WAIT_RECOVERY     Type is constant  =  3
                                              GFC$TR_JOB                      Type is constant  =  2
                                              GFC$TR_SYSTEM                   Type is constant  =  1
                                              GFC$TR_NULL_RESIDENCE           Type is constant  =  0
        FILE_HASH                          Type is field of a record  Length =  1(bytes)  Offset =  B3(16) : 0
                                            Type is subrange  0 ..  255
    SWAPPING_IO_ERROR                  Type is field of a record  Length =  1(bytes)  Offset =  B4(16) : 0
                                        IOT$IO_ERROR                    Type is ordinal
                                          IOC$UNIT_DOWN_ON_INIT             Type is constant  =  8
                                          IOC$ERROR_ON_INIT                 Type is constant  =  7
                                          IOC$SERVER_HAS_TERMINATED         Type is constant  =  6
                                          IOC$SERVER_ALLOCATION_ERROR       Type is constant  =  5
                                          IOC$UNRECOVERED_ERROR_UNIT_DOWN   Type is constant  =  4
                                          IOC$UNRECOVERED_ERROR             Type is constant  =  3
                                          IOC$MEDIA_ERROR                   Type is constant  =  2
                                          IOC$ALLOCATE_FILE_SPACE           Type is constant  =  1
                                          IOC$NO_ERROR                      Type is constant  =  0
    SWAPPED_JOB_PAGE_COUNT             Type is field of a record  Length =  2(bytes)  Offset =  B5(16) : 0
                                        Type is subrange  0 ..  65535
    SWAP_FILE_LENGTH_IN_PAGES          Type is field of a record  Length =  2(bytes)  Offset =  B7(16) : 0
                                        Type is subrange  0 ..  65535
    ASID_REASSIGNED_TIMESTAMP          Type is field of a record  Length =  6(bytes)  Offset =  B9(16) : 0
                                        OST$FREE_RUNNING_CLOCK            Type is subrange  0 ..  281474976710655
    TIMESTAMP                          Type is field of a record  Length =  6(bytes)  Offset =  BF(16) : 0
                                        OST$FREE_RUNNING_CLOCK            Type is subrange  0 ..  281474976710655
    SWAPOUT_TIMESTAMP                  Type is field of a record  Length =  6(bytes)  Offset =  C5(16) : 0
                                        OST$FREE_RUNNING_CLOCK            Type is subrange  0 ..  281474976710655
    REASSIGNED_JOB_FIXED_ASTI          Type is field of a record  Length =  2(bytes)  Offset =  CB(16) : 0
                                        MMT$AST_INDEX                     Type is subrange  0 ..  65535
    SWAPPED_JOB_ENTRY                  Type is field of a record  Length =  10(bytes)  Offset =  CD(16) : 0
      JMT$SWAPPED_JOB_ENTRY              Type is record  Length =  10
        AVAILABLE_MODIFIED_PAGE_COUNT      Type is field of a record  Length =  2(bytes)  Offset =  CD(16) : 0
                                            Type is subrange  0 ..  65535
        JOB_PAGE_QUEUE_COUNT               Type is field of a record  Length =  6(bytes)  Offset =  CF(16) : 0
                                            Type is cybil array  Array element length  2
                                              Type is subrange  0 ..  65535
        SWAP_FILE_DESCRIPTOR_PAGE_COUNT Type is field of a record  Length =  2(bytes)  Offset =  D5(16) : 0
                                            Type is subrange  0 ..  65535
SWAP_IO_CONTROL                    Type is field of a record  Length =  9(bytes)  Offset =  D7(16) : 0
  JST$IO_CONTROL_INFORMATION         Type is record  Length =  9
    SPD_INDEX                          Type is field of a record  Length =  2(bytes)  Offset =  D7(16) : 0
                                        MMT$PAGE_FRAME_INDEX              Type is subrange  0 ..  65534
    NEXT_QUEUE_ID                      Type is field of a record  Length =  1(bytes)  Offset =  D9(16) : 0
                                        MMT$PAGE_FRAME_QUEUE_ID           Type is subrange  0 ..  39
    NEXT_PFTI                          Type is field of a record  Length =  2(bytes)  Offset =  DA(16) : 0
                                        MMT$PAGE_FRAME_INDEX              Type is subrange  0 ..  65534
    STOP_PFTI                          Type is field of a record  Length =  2(bytes)  Offset =  DC(16) : 0
                                        MMT$PAGE_FRAME_INDEX              Type is subrange  0 ..  65534
    SWAP_FILE_DESCRIPTOR_PFTI          Type is field of a record  Length =  2(bytes)  Offset =  DE(16) : 0
                                        MMT$PAGE_FRAME_INDEX              Type is subrange  0 ..  65534
SFD_P                              Type is field of a record  Length =  18(bytes)  Offset =  E0(16) : 0
                                    Type is pointer  Ptr object length  352
                                    Pointer to JST$SWAP_FILE_DESCRIPTOR       11c$record_kind
SYSTEM_BREAKPOINT_SELECTED         Type is field of a record  Length =  1(bytes)  Offset =  F2(16) : 0
                                    Type is boolean
DELAYED_SWAPIN_WORK                Type is field of a record  Length =  2(bytes)  Offset =  F3(16) : 0
                                    JMT$DELAYED_SWAPIN_WORK         Type is set  Set length  16
                                      Type is ordinal
                                        JMC$DSW_UNUSED_15                 Type is constant  =  15
                                        JMC$DSW_UNUSED_14                 Type is constant  =  14
                                        JMC$DSW_UNUSED_13                 Type is constant  =  13
                                        JMC$DSW_UNUSED_12                 Type is constant  =  12
                                        JMC$DSW_UNUSED_11                 Type is constant  =  11
                                        JMC$DSW_UNUSED_10                 Type is constant  =  10
                                        JMC$DSW_IO_ERROR_WHILE_SWAPPED    Type is constant  =  9
                                        JMC$DSW_ADJUST_CPU_SELECTIONS     Type is constant  =  8
                                        JMC$DSW_UPDATE_SERVER_FILES       Type is constant  =  7
                                        JMC$DSW_RECOVERY_SWAP_IO_ERROR    Type is constant  =  6
                                        JMC$DSW_UPDATE_JOB_TASK_ENVIRO    Type is constant  =  5
```

```
                                JMC$DSW_JOB_SHARED_ASID_CHANGED Type is constant  =  4
                                JMC$DSW_JOB_ASID_CHANGED        Type is constant  =  3
                                JMC$DSW_UPDATE_KEYPOINT_MASKS   Type is constant  =  2
                                JMC$DSW_UPDATE_DEBUG_LISTS      Type is constant  =  1
                                JMC$DSW_JOB_RECOVERY            Type is constant  =  0
INHIBIT_ACCESS_WORK           Type is field of a record  Length =  1(bytes)  Offset =  F5(16) : 0
                               DFT$MAINFRAME_SET                Type is set  Set length  8
                                 Type is subrange  1 ..  8
TERMINATE_ACCESS_WORK         Type is field of a record  Length =  1(bytes)  Offset =  F6(16) : 0
                               DFT$MAINFRAME_SET                Type is set  Set length  8
                                 Type is subrange  1 ..  8
STATISTICS                    Type is field of a record  Length =  54(bytes)  Offset =  F7(16) : 0
  JMT$IJL_STATISTICS            Type is record  Length =  54
    CP_TIME                       Type is field of a record  Length =  10(bytes)  Offset =  F7(16) : 0
      OST$CP_TIME                   Type is record  Length =  10
        TIME_SPENT_IN_JOB_MODE        Type is field of a record  Length =  5(bytes)  Offset =  F7(16) : 0
                                       OST$CP_TIME_VALUE              Type is subrange  0 ..  1099511627775
        TIME_SPENT_IN_MTR_MODE        Type is field of a record  Length =  5(bytes)  Offset =  FC(16) : 0
                                       OST$CP_TIME_VALUE              Type is subrange  0 ..  1099511627775
    PAGING_STATISTICS             Type is field of a record  Length =  24(bytes)  Offset =  101(16) : 0
      OST$PAGING_STATISTICS         Type is record  Length =  24
        PAGE_IN_COUNT                 Type is field of a record  Length =  4(bytes)  Offset =  101(16) : 0
                                       Type is subrange  0 ..  4294967295
        PAGES_RECLAIMED_FROM_QUEUE    Type is field of a record  Length =  4(bytes)  Offset =  105(16) : 0
                                       Type is subrange  0 ..  4294967295
        NEW_PAGES_ASSIGNED            Type is field of a record  Length =  4(bytes)  Offset =  109(16) : 0
                                       Type is subrange  0 ..  4294967295
        PAGES_FROM_SERVER             Type is field of a record  Length =  4(bytes)  Offset =  10D(16) : 0
                                       Type is subrange  0 ..  4294967295
        PAGE_FAULT_COUNT              Type is field of a record  Length =  6(bytes)  Offset =  111(16) : 0
                                       Type is subrange  0 ..  281474976710655
        WORKING_SET_MAX_USED          Type is field of a record  Length =  2(bytes)  Offset =  117(16) : 0
                                       Type is subrange  0 ..  65535
    PERM_FILE_SPACE               Type is field of a record  Length =  8(bytes)  Offset =  119(16) : 0
                                   Type is integer
    TEMP_FILE_SPACE               Type is field of a record  Length =  8(bytes)  Offset =  121(16) : 0
                                   Type is integer
    READY_TASK_COUNT              Type is field of a record  Length =  2(bytes)  Offset =  129(16) : 0
                                   Type is subrange  0 ..  65535
    TASKS_NOT_IN_LONG_WAIT        Type is field of a record  Length =  2(bytes)  Offset =  12B(16) : 0
                                   Type is subrange  0 ..  65535
SERVICE_CLASS_STATISTICS      Type is field of a record  Length =  28(bytes)  Offset =  12D(16) : 0
  JMT$IJL_SERVICE_CLASS_STATS   Type is record  Length =  28
    CP_TIME                       Type is field of a record  Length =  10(bytes)  Offset =  12D(16) : 0
      OST$CP_TIME                   Type is record  Length =  10
        TIME_SPENT_IN_JOB_MODE        Type is field of a record  Length =  5(bytes)  Offset =  12D(16) : 0
                                       OST$CP_TIME_VALUE              Type is subrange  0 ..  1099511627775
        TIME_SPENT_IN_MTR_MODE        Type is field of a record  Length =  5(bytes)  Offset =  132(16) : 0
                                       OST$CP_TIME_VALUE              Type is subrange  0 ..  1099511627775
    PAGE_FAULTS                   Type is field of a record  Length =  12(bytes)  Offset =  137(16) : 0
      JMT$IJL_PAGE_STATS            Type is record  Length =  12
        DISK                          Type is field of a record  Length =  4(bytes)  Offset =  137(16) : 0
                                       JMT$IJL_PAGE_FAULT_COUNT       Type is subrange  0 ..  4294967295
        RECLAIMED                     Type is field of a record  Length =  4(bytes)  Offset =  13B(16) : 0
                                       JMT$IJL_PAGE_FAULT_COUNT       Type is subrange  0 ..  4294967295
        ASSIGNED                      Type is field of a record  Length =  4(bytes)  Offset =  13F(16) : 0
                                       JMT$IJL_PAGE_FAULT_COUNT       Type is subrange  0 ..  4294967295
    SWAPOUTS                      Type is field of a record  Length =  6(bytes)  Offset =  143(16) : 0
      JMT$IJL_SWAP_COUNTS           Type is record  Length =  6
        LONG_WAIT                     Type is field of a record  Length =  3(bytes)  Offset =  143(16) : 0
                                       JMT$IJL_SWAP_COUNT             Type is subrange  0 ..  16777215
        JOB_MODE                      Type is field of a record  Length =  3(bytes)  Offset =  146(16) : 0
                                       JMT$IJL_SWAP_COUNT             Type is subrange  0 ..  16777215
JOB_FIXED_CONTIGUOUS_PAGES    Type is field of a record  Length =  1(bytes)  Offset =  149(16) : 0
                               Type is subrange  0 ..  255
HUNG_TASK_IN_JOB              Type is field of a record  Length =  1(bytes)  Offset =  14A(16) : 0
                               Type is boolean
JOB_DAMAGED_DURING_RECOVERY   Type is field of a record  Length =  1(bytes)  Offset =  14B(16) : 0
                               Type is boolean
MAXWS_AIO_SLOWDOWN_DISPLAY    Type is field of a record  Length =  1(bytes)  Offset =  14C(16) : 0
                               Type is subrange  0 ..  255
UNABLE_TO_SWAP_IDLE_FLAG      Type is field of a record  Length =  1(bytes)  Offset =  14D(16) : 0
                               Type is boolean
QUEUE_FILE_INFORMATION        Type is field of a record  Length =  3(bytes)  Offset =  14E(16) : 0
  JMT$QUEUE_FILE_IJL_INFORMATION  Type is record  Length =  3
    JOB_ABORT_DISPOSITION         Type is field of a record  Length =  1(bytes)  Offset =  14E(16) : 0
                                   JMT$JOB_ABORT_DISPOSITION      Type is ordinal
                                     JMC$TERMINATE_ON_ABORT         Type is constant  =  1
                                     JMC$RESTART_ON_ABORT           Type is constant  =  0
    JOB_RECOVERY_DISPOSITION      Type is field of a record  Length =  1(bytes)  Offset =  14F(16) : 0
                                   JMT$JOB_RECOVERY_DISPOSITION   Type is ordinal
                                     JMC$TERMINATE_ON_RECOVERY      Type is constant  =  2
                                     JMC$RESTART_ON_RECOVERY        Type is constant  =  1
                                     JMC$CONTINUE_ON_RECOVERY       Type is constant  =  0
    INPUT_FILE_LOCATION           Type is field of a record  Length =  1(bytes)  Offset =  150(16) : 0
                                   JMT$INPUT_FILE_LOCATION        Type is subrange  0 ..  255
RELATIVE_PRIORITY_ENABLED     Type is field of a record  Length =  1(bytes)  Offset =  151(16) : 0
                               Type is boolean
TASK_CREATED_AFTER_LAST_SWAP  Type is field of a record  Length =  1(bytes)  Offset =  152(16) : 0
                               Type is boolean
INTERACTIVE_TASK_GTID         Type is field of a record  Length =  3(bytes)  Offset =  153(16) : 0
  OST$GLOBAL_TASK_ID            Type is record  Length =  3
    INDEX                         Type is field of a record  Length =  2(bytes)  Offset =  153(16) : 0
                                   OST$TASK_INDEX               Type is subrange  0 ..  4095
    SEQNO                         Type is field of a record  Length =  1(bytes)  Offset =  155(16) : 0
                                   Type is subrange  0 ..  255
```

```
JMT$JOB_CONTROL_BLOCK                Type is record  Length =  217
 JCB_IDENTIFIER                       Type is field of a record  Length =  2(bytes)  Offset =  0(16) : 0
                                        Type is subrange  0 ..  65535
 LAST_LPID_FOR_JOB                    Type is field of a record  Length =  1(bytes)  Offset =  2(16) : 0
                                        Type is subrange  0 ..  255
 SYSTEM_NAME                          Type is field of a record  Length =  19(bytes)  Offset =  3(16) : 0
                                       JMT$SYSTEM_SUPPLIED_NAME        Type is string  String length  19
 JOBNAME                              Type is field of a record  Length =  31(bytes)  Offset =  16(16) : 0
                                       OST$NAME                        Type is string  String length  31
 JOB_ID                               Type is field of a record  Length =  2(bytes)  Offset =  35(16) : 0
                                       JMT$KJL_INDEX                   Type is subrange  0 ..  65535
 USER_ID                              Type is field of a record  Length =  62(bytes)  Offset =  37(16) : 0
   OST$USER_IDENTIFICATION              Type is record  Length =  62
     USER                               Type is field of a record  Length =  31(bytes)  Offset =  37(16) : 0
                                         OST$NAME                        Type is string  String length  31
     FAMILY                             Type is field of a record  Length =  31(bytes)  Offset =  56(16) : 0
                                         OST$NAME                        Type is string  String length  31
 JOB_MONITOR_ID                       Type is field of a record  Length =  3(bytes)  Offset =  75(16) : 0
   OST$GLOBAL_TASK_ID                   Type is record  Length =  3
     INDEX                              Type is field of a record  Length =  2(bytes)  Offset =  75(16) : 0
                                         OST$TASK_INDEX                  Type is subrange  0 ..  4095
     SEQNO                              Type is field of a record  Length =  1(bytes)  Offset =  77(16) : 0
                                          Type is subrange  0 ..  255
 IJLE_P                               Type is field of a record  Length =  6(bytes)  Offset =  78(16) : 0
                                        Type is pointer  Ptr object length  342
                                        Pointer to JMT$INITIATED_JOB_LIST_ENTRY   11c$record_kind
 IJL_ORDINAL                          Type is field of a record  Length =  2(bytes)  Offset =  7E(16) : 0
   JMT$IJL_ORDINAL                      Type is packed record  Length =  2
     BLOCK_NUMBER                       Type is field of a record  Length =  11 (bits)  Offset =  7E(16) : 0
                                         JMT$IJL_BLOCK_NUMBER            Type is subrange  0 ..  2047
     BLOCK_INDEX                        Type is field of a record  Length =  5 (bits)  Offset =  7F(16) : 3
                                         JMT$IJL_BLOCK_INDEX             Type is subrange  0 ..  31
 SERVER_MAINFRAME_ID                  Type is field of a record  Length =  3(bytes)  Offset =  80(16) : 0
   PMT$BINARY_MAINFRAME_ID              Type is record  Length =  3
     MODEL_NUMBER                       Type is field of a record  Length =  1(bytes)  Offset =  80(16) : 0
                                         OST$PROCESSOR_MODEL_NUMBER      Type is subrange  0 ..  255
     SERIAL_NUMBER                      Type is field of a record  Length =  2(bytes)  Offset =  81(16) : 0
                                         OST$PROCESSOR_SERIAL_NUMBER     Type is subrange  0 ..  65535
 LAST_EXECUTION_TIME                  Type is field of a record  Length =  6(bytes)  Offset =  83(16) : 0
                                       OST$FREE_RUNNING_CLOCK          Type is subrange  0 ..  281474976710655
 CPTIME_NEXT_AGE_WORKING_SET          Type is field of a record  Length =  5(bytes)  Offset =  89(16) : 0
                                       OST$CP_TIME_VALUE               Type is subrange  0 ..  1099511627775
 CPTIME_SIGNAL_LAST_SENT              Type is field of a record  Length =  5(bytes)  Offset =  8E(16) : 0
                                       OST$CP_TIME_VALUE               Type is subrange  0 ..  1099511627775
 SIGNAL_INTERVAL                      Type is field of a record  Length =  4(bytes)  Offset =  93(16) : 0
                                        Type is subrange  0 ..  4294967295
 MAX_WORKING_SET_SIZE                 Type is field of a record  Length =  2(bytes)  Offset =  97(16) : 0
                                       JMT$WORKING_SET_SIZE            Type is subrange  0 ..  65004
 MIN_WORKING_SET_SIZE                 Type is field of a record  Length =  2(bytes)  Offset =  99(16) : 0
                                       JMT$WORKING_SET_SIZE            Type is subrange  0 ..  65004
 PAGE_AGING_INTERVAL                  Type is field of a record  Length =  4(bytes)  Offset =  9B(16) : 0
                                       OST$AGING_INTERVAL              Type is subrange  0 ..  4294967295
 CYCLIC_AGING_INTERVAL                Type is field of a record  Length =  4(bytes)  Offset =  9F(16) : 0
                                       OST$AGING_INTERVAL              Type is subrange  0 ..  4294967295
 DETACHED_JOB_WAIT_TIME               Type is field of a record  Length =  2(bytes)  Offset =  A3(16) : 0
                                       JMT$DETACHED_JOB_WAIT_TIME      Type is subrange  0 ..  36001
 NEXT_CYCLIC_AGING_TIME               Type is field of a record  Length =  6(bytes)  Offset =  A5(16) : 0
                                       OST$FREE_RUNNING_CLOCK          Type is subrange  0 ..  281474976710655
 SENSE_SWITCHES                       Type is field of a record  Length =  1(bytes)  Offset =  AB(16) : 0
                                       PMT$SENSE_SWITCHES              Type is set  Set length  8
                                         Type is subrange  1 ..  8
 PERM_FILE_JOB_WARNING_LIMIT          Type is field of a record  Length =  8(bytes)  Offset =  AC(16) : 0
                                        Type is integer
 PERM_FILE_JOB_WARNING_CHECKING       Type is field of a record  Length =  1(bytes)  Offset =  B4(16) : 0
                                        Type is boolean
 PERM_FILE_JOB_MAXIMUM_LIMIT          Type is field of a record  Length =  8(bytes)  Offset =  B5(16) : 0
                                        Type is integer
 TEMP_FILE_JOB_WARNING_LIMIT          Type is field of a record  Length =  8(bytes)  Offset =  BD(16) : 0
                                        Type is integer
 TEMP_FILE_JOB_WARNING_CHECKING       Type is field of a record  Length =  1(bytes)  Offset =  C5(16) : 0
                                        Type is boolean
 TEMP_FILE_JOB_MAXIMUM_LIMIT          Type is field of a record  Length =  8(bytes)  Offset =  C6(16) : 0
                                        Type is integer
 SWAPPED_JOB_ENTRY                    Type is field of a record  Length =  10(bytes)  Offset =  CE(16) : 0
   JMT$SWAPPED_JOB_ENTRY                Type is record  Length =  10
     AVAILABLE_MODIFIED_PAGE_COUNT      Type is field of a record  Length =  2(bytes)  Offset =  CE(16) : 0
                                          Type is subrange  0 ..  65535
     JOB_PAGE_QUEUE_COUNT               Type is field of a record  Length =  6(bytes)  Offset =  D0(16) : 0
                                          Type is cybil array  Array element length  2
                                            Type is subrange  0 ..  65535
     SWAP_FILE_DESCRIPTOR_PAGE_COUNT Type is field of a record  Length =  2(bytes)  Offset =  D6(16) : 0
                                          Type is subrange  0 ..  65535
 ACCOUNT_PROJECT_SPECIFIED            Type is field of a record  Length =  1(bytes)  Offset =  D8(16) : 0
                                        Type is boolean
```

```
MMT$ACTIVE_SEGMENT_TABLE_ENTRY  Type is record  Length =  17
 PFT_LINK                       Type is field of a record  Length =  4(bytes)  Offset =  0(16) : 0
   MMT$LINK                       Type is record  Length =  4
     BKW                            Type is field of a record  Length =  2(bytes)  Offset =  0(16) : 0
                                     MMT$PAGE_FRAME_INDEX           Type is subrange  0 ..  65534
     FWD                            Type is field of a record  Length =  2(bytes)  Offset =  2(16) : 0
                                     MMT$PAGE_FRAME_INDEX           Type is subrange  0 ..  65534
 PAGES_IN_MEMORY                Type is field of a record  Length =  2(bytes)  Offset =  4(16) : 0
                                  Type is subrange  0 ..  65535
 IJL_ORDINAL                    Type is field of a record  Length =  2(bytes)  Offset =  6(16) : 0
   JMT$IJL_ORDINAL                Type is packed record  Length =  2
     BLOCK_NUMBER                   Type is field of a record  Length =  11 (bits)  Offset =  6(16) : 0
                                     JMT$IJL_BLOCK_NUMBER           Type is subrange  0 ..  2047
     BLOCK_INDEX                    Type is field of a record  Length =  5 (bits)  Offset =  7(16) : 3
                                     JMT$IJL_BLOCK_INDEX            Type is subrange  0 ..  31
 IN_USE                         Type is field of a record  Length =  1(bytes)  Offset =  8(16) : 0
                                  Type is boolean

  --- Variant --FALSE
   TIME_FREED                     Type is field of a record  Length =  6(bytes)  Offset =  9(16) : 0
                                   OST$FREE_RUNNING_CLOCK        Type is subrange  0 ..  281474976710655
   ASID                           Type is field of a record  Length =  2(bytes)  Offset =  F(16) : 0
                                   OST$ASID                      Type is subrange  0 ..  65535
  --- Variant --TRUE
   QUEUE_ID                       Type is field of a record  Length =  1(bytes)  Offset =  9(16) : 0
                                   MMT$PAGE_FRAME_QUEUE_ID       Type is subrange  0 ..  39
   SFID                           Type is field of a record  Length =  4(bytes)  Offset =  A(16) : 0
     GFT$SYSTEM_FILE_IDENTIFIER     Type is record  Length =  4
       FILE_ENTRY_INDEX               Type is field of a record  Length =  2(bytes)  Offset =  A(16) : 0
                                       GFT$FILE_DESCRIPTOR_INDEX     Type is subrange  0 ..  65535
       RESIDENCE                      Type is field of a record  Length =  1(bytes)  Offset =  C(16) : 0
                                       GFT$TABLE_RESIDENCE           Type is ordinal
                                         GFC$TR_SYSTEM_WAIT_RECOVERY     Type is constant  =  3
                                         GFC$TR_JOB                      Type is constant  =  2
                                         GFC$TR_SYSTEM                   Type is constant  =  1
                                         GFC$TR_NULL_RESIDENCE           Type is constant  =  0
       FILE_HASH                      Type is field of a record  Length =  1(bytes)  Offset =  D(16) : 0
                                       Type is subrange  0 ..  255
   INCLUDE_PAGES_IN_DUMP          Type is field of a record  Length =  1(bytes)  Offset =  E(16) : 0
                                    Type is boolean
```

```
MMT$PAGE_FRAME_TABLE_ENTRY      Type is record  Length =  36
 LINK                           Type is field of a record  Length =  4(bytes)  Offset =  0(16) : 0
   MMT$LINK                       Type is record  Length =  4
     BKW                            Type is field of a record  Length =  2(bytes)  Offset =  0(16) : 0
                                     MMT$PAGE_FRAME_INDEX           Type is subrange  0 ..  65534
     FWD                            Type is field of a record  Length =  2(bytes)  Offset =  2(16) : 0
                                     MMT$PAGE_FRAME_INDEX           Type is subrange  0 ..  65534
 SEGMENT_LINK                   Type is field of a record  Length =  4(bytes)  Offset =  4(16) : 0
   MMT$LINK                       Type is record  Length =  4
     BKW                            Type is field of a record  Length =  2(bytes)  Offset =  4(16) : 0
                                     MMT$PAGE_FRAME_INDEX           Type is subrange  0 ..  65534
     FWD                            Type is field of a record  Length =  2(bytes)  Offset =  6(16) : 0
                                     MMT$PAGE_FRAME_INDEX           Type is subrange  0 ..  65534
 CYCLIC_AGE                     Type is field of a record  Length =  1(bytes)  Offset =  8(16) : 0
                                  Type is subrange  0 ..  255
 IJL_ORDINAL                    Type is field of a record  Length =  2(bytes)  Offset =  9(16) : 0
   JMT$IJL_ORDINAL                Type is packed record  Length =  2
     BLOCK_NUMBER                   Type is field of a record  Length =  11 (bits)  Offset =  9(16) : 0
                                     JMT$IJL_BLOCK_NUMBER           Type is subrange  0 ..  2047
     BLOCK_INDEX                    Type is field of a record  Length =  5 (bits)  Offset =  A(16) : 3
                                     JMT$IJL_BLOCK_INDEX            Type is subrange  0 ..  31
 QUEUE_ID                       Type is field of a record  Length =  1(bytes)  Offset =  B(16) : 0
                                 MMT$PAGE_FRAME_QUEUE_ID       Type is subrange  0 ..  39
 ACTIVE_IO_COUNT                Type is field of a record  Length =  1(bytes)  Offset =  C(16) : 0
                                  Type is subrange  0 ..  255
 LOCKED_PAGE                    Type is field of a record  Length =  1(bytes)  Offset =  D(16) : 0
                                 MMT$LOCKED_PAGE               Type is ordinal
                                   MMC$LP_SERVER_ALLOCATE_LOCK     Type is constant  =  4
                                   MMC$LP_PAGE_IN_LOCK             Type is constant  =  3
                                   MMC$LP_WRITE_PROTECTED_LOCK     Type is constant  =  2
                                   MMC$LP_AGING_LOCK               Type is constant  =  1
                                   MMC$LP_NOT_LOCKED               Type is constant  =  0
 PTI                            Type is field of a record  Length =  3(bytes)  Offset =  E(16) : 0
                                 OST$PAGE_TABLE_INDEX          Type is subrange  0 ..  262143
 TASK_QUEUE                     Type is field of a record  Length =  4(bytes)  Offset =  11(16) : 0
   TMT$TASK_QUEUE_LINK            Type is record  Length =  4
     HEAD                           Type is field of a record  Length =  2(bytes)  Offset =  11(16) : 0
                                     OST$TASK_INDEX                Type is subrange  0 ..  4095
     TAIL                           Type is field of a record  Length =  2(bytes)  Offset =  13(16) : 0
                                     OST$TASK_INDEX                Type is subrange  0 ..  4095
 AGE                            Type is field of a record  Length =  2(bytes)  Offset =  15(16) : 0
                                 MMT$PAGE_AGE                  Type is subrange  0 ..  65535
 ASTE_P                         Type is field of a record  Length =  6(bytes)  Offset =  17(16) : 0
                                  Type is pointer  Ptr object length  17
                                  Pointer to MMT$ACTIVE_SEGMENT_TABLE_ENTRY llc$record_kind
 IO_ERROR                       Type is field of a record  Length =  1(bytes)  Offset =  1D(16) : 0
                                 IOT$IO_ERROR                  Type is ordinal
                                   IOC$UNIT_DOWN_ON_INIT           Type is constant  =  8
                                   IOC$ERROR_ON_INIT               Type is constant  =  7
                                   IOC$SERVER_HAS_TERMINATED       Type is constant  =  6
                                   IOC$SERVER_ALLOCATION_ERROR     Type is constant  =  5
                                   IOC$UNRECOVERED_ERROR_UNIT_DOWN Type is constant  =  4
                                   IOC$UNRECOVERED_ERROR           Type is constant  =  3
                                   IOC$MEDIA_ERROR                 Type is constant  =  2
                                   IOC$ALLOCATE_FILE_SPACE         Type is constant  =  1
                                   IOC$NO_ERROR                    Type is constant  =  0
 SVA                            Type is field of a record  Length =  6(bytes)  Offset =  1E(16) : 0
   OST$SYSTEM_VIRTUAL_ADDRESS     Type is packed record  Length =  6
     ASID                           Type is field of a record  Length =  2(bytes)  Offset =  1E(16) : 0
                                     OST$ASID                      Type is subrange  0 ..  65535
     OFFSET                         Type is field of a record  Length =  4(bytes)  Offset =  20(16) : 0
                                     OST$SEGMENT_OFFSET            Type is subrange -2147483648 ..  2147483647
```

```
MMT$SEGMENT_DESCRIPTOR_TABLE_EX Type is record  Length =  0
 SDTX_TABLE                     Type is field of a record  Length =  18(bytes)  Offset =  0(16) : 0
                                  Type is cybil array  Array element length  36
     MMT$SEGMENT_DESCRIPTOR_EXTENDED Type is record  Length =  36
       OPEN_VALIDATING_RING_NUMBER     Type is field of a record  Length =  1(bytes)  Offset =  0(16) : 0
                                        OST$RING                          Type is subrange  0 ..  15
       ACCESS_STATE                    Type is field of a record  Length =  1(bytes)  Offset =  1(16) : 0
                                        MMT$SEGMENT_ACCESS_STATE          Type is ordinal
                                          MMC$SAS_TERMINATE_ACCESS          Type is constant  =  2
                                          MMC$SAS_INHIBIT_ACCESS            Type is constant  =  1
                                          MMC$SAS_ALLOW_ACCESS              Type is constant  =  0
       SFID                            Type is field of a record  Length =  4(bytes)  Offset =  2(16) : 0
         GFT$SYSTEM_FILE_IDENTIFIER      Type is record  Length =  4
           FILE_ENTRY_INDEX                Type is field of a record  Length =  2(bytes)  Offset =  2(16) : 0
                                            GFT$FILE_DESCRIPTOR_INDEX       Type is subrange  0 ..  65535
           RESIDENCE                       Type is field of a record  Length =  1(bytes)  Offset =  4(16) : 0
                                            GFT$TABLE_RESIDENCE             Type is ordinal
                                              GFC$TR_SYSTEM_WAIT_RECOVERY     Type is constant  =  3
                                              GFC$TR_JOB                      Type is constant  =  2
                                              GFC$TR_SYSTEM                   Type is constant  =  1
                                              GFC$TR_NULL_RESIDENCE           Type is constant  =  0
           FILE_HASH                       Type is field of a record  Length =  1(bytes)  Offset =  5(16) : 0
                                            Type is subrange  0 ..  255
       INHERITANCE                     Type is field of a record  Length =  1(bytes)  Offset =  6(16) : 0
                                        MMT$SEGMENT_INHERITANCE           Type is ordinal
                                          MMC$SI_COPY_ON_WRITE              Type is constant  =  4
                                          MMC$SI_NEW_SEGMENT                Type is constant  =  3
                                          MMC$SI_TRANSFER_SEGMENT           Type is constant  =  2
                                          MMC$SI_SHARE_SEGMENT              Type is constant  =  1
                                          MMC$SI_NONE                       Type is constant  =  0
       SEGMENT_RESERVATION_STATE       Type is field of a record  Length =  1(bytes)  Offset =  7(16) : 0
                                        MMT$SEGMENT_RESERVATION_STATE     Type is ordinal
                                          MMC$SRS_RESERVED_SHARED_STACK     Type is constant  =  2
                                          MMC$SRS_RESERVED                  Type is constant  =  1
                                          MMC$SRS_NOT_RESERVED              Type is constant  =  0
       SOFTWARE_ATTRIBUTE_SET          Type is field of a record  Length =  1(bytes)  Offset =  8(16) : 0
                                        MMT$SOFTWARE_ATTRIBUTE_SET        Type is set  Set length  7
                                          MMT$SOFTWARE_ATTRIBUTES           Type is ordinal
                                            MMC$SA_JOB_SHARED                 Type is constant  =  6
                                            MMC$SA_NO_APPEND                  Type is constant  =  5
                                            MMC$SA_FREE_BEHIND                Type is constant  =  4
                                            MMC$SA_READ_TRANSFER_UNIT         Type is constant  =  3
                                            MMC$SA_STACK                      Type is constant  =  2
                                            MMC$SA_FIXED                      Type is constant  =  1
                                            MMC$SA_WIRED                      Type is constant  =  0
       ACCESS_RIGHTS                   Type is field of a record  Length =  1(bytes)  Offset =  9(16) : 0
                                        MMT$SEGMENT_ACCESS_RIGHTS         Type is ordinal
                                          MMC$SAR_WRITE_EXTEND              Type is constant  =  3
                                          MMC$SAR_MODIFY                    Type is constant  =  2
                                          MMC$SAR_READ                      Type is constant  =  1
                                          MMC$SAR_NONE                      Type is constant  =  0
       SEGMENT_LOCK                    Type is field of a record  Length =  1(bytes)  Offset =  A(16) : 0
                                        MMT$LOCK_SEGMENT_STATUS           Type is ordinal
                                          MMC$LSS_LOCK_FOR_WRITE_R3         Type is constant  =  6
                                          MMC$LSS_LOCK_FOR_WRITE_USER       Type is constant  =  5
                                          MMC$LSS_LOCK_FOR_READ_R3          Type is constant  =  4
                                          MMC$LSS_LOCK_FOR_READ_USER        Type is constant  =  3
                                          MMC$LSS_QUEUED_FOR_LOCK_R3        Type is constant  =  2
                                          MMC$LSS_QUEUED_FOR_LOCK_USER      Type is constant  =  1
                                          MMC$LSS_NONE                      Type is constant  =  0
       SHADOW_INFO                     Type is field of a record  Length =  13(bytes)  Offset =  B(16) : 0
         MMT$SHADOW_INFO                 Type is record  Length =  13
           SHADOW_START_PAGE_NUMBER        Type is field of a record  Length =  3(bytes)  Offset =  B(16) : 0
                                            Type is subrange  0 ..  16777215
           SHADOW_LENGTH_PAGE_COUNT        Type is field of a record  Length =  3(bytes)  Offset =  E(16) : 0
                                            Type is subrange  0 ..  16777215
           SHADOW_SFID                     Type is field of a record  Length =  4(bytes)  Offset =  11(16) : 0
             GFT$SYSTEM_FILE_IDENTIFIER      Type is record  Length =  4
               FILE_ENTRY_INDEX                Type is field of a record  Length =  2(bytes)  Offset =  11(16) : 0
                                                GFT$FILE_DESCRIPTOR_INDEX       Type is subrange  0 ..  65535
               RESIDENCE                       Type is field of a record  Length =  1(bytes)  Offset =  13(16) : 0
                                                GFT$TABLE_RESIDENCE             Type is ordinal
                                                  GFC$TR_SYSTEM_WAIT_RECOVERY     Type is constant  =  3
                                                  GFC$TR_JOB                      Type is constant  =  2
                                                  GFC$TR_SYSTEM                   Type is constant  =  1
                                                  GFC$TR_NULL_RESIDENCE           Type is constant  =  0
               FILE_HASH                       Type is field of a record  Length =  1(bytes)  Offset =  14(16) : 0
                                                Type is subrange  0 ..  255
           SHADOW_SEGMENT_KIND             Type is field of a record  Length =  1(bytes)  Offset =  15(16) : 0
                                            MMT$SHADOW_SEGMENT_KIND           Type is ordinal
                                              MMC$SSK_SEGMENT_NUMBER            Type is constant  =  4
                                              MMC$SSK_READ_ONLY_TRANS_FILE      Type is constant  =  3
                                              MMC$SSK_READ_ONLY_FILE            Type is constant  =  2
                                              MMC$SSK_READ_WRITE_FILE           Type is constant  =  1
                                              MMC$SSK_NONE                      Type is constant  =  0
            --- Variant --MMC$SSK_NONE
            PASSIVE_FOR_SHADOW_BY_SEGNUM    Type is field of a record  Length =  1(bytes)  Offset =  16(16) : 0
                                             Type is boolean
            --- Variant --MMC$SSK_SEGMENT_NUMBER
            SHADOW_SEGMENT_NUMBER           Type is field of a record  Length =  2(bytes)  Offset =  16(16) : 0
                                             OST$SEGMENT                       Type is subrange  0 ..  4095
       FILE_LIMITS_ENFORCED            Type is field of a record  Length =  1(bytes)  Offset =  18(16) : 0
                                        SFT$FILE_SPACE_LIMIT_KIND         Type is ordinal
                                          SFC$TEMP_FILE_SPACE_LIMIT         Type is constant  =  2
                                          SFC$PERM_FILE_SPACE_LIMIT         Type is constant  =  1
                                          SFC$NO_LIMIT                      Type is constant  =  0
       STREAM                          Type is field of a record  Length =  7(bytes)  Offset =  19(16) : 0
         MMT$SDTX_STREAM_DATA            Type is packed record  Length =  7
           LAST_PAGE_FAULT                 Type is field of a record  Length =  4(bytes)  Offset =  19(16) : 0
                                            OST$SEGMENT_OFFSET                Type is subrange -2147483648 ..  2147483647
           SEQUENTIAL_ACCESSES             Type is field of a record  Length =  1(bytes)  Offset =  1D(16) : 0
                                            Type is subrange  0 ..  255
           TRANSFER_SIZE                   Type is field of a record  Length =  4 (bits)  Offset =  1E(16) : 0
                                            Type is subrange  0 ..  15
           RANDOM_FAULTS                   Type is field of a record  Length =  4 (bits)  Offset =  1E(16) : 4
                                            Type is subrange  0 ..  15
           STREAMING                       Type is field of a record  Length =  1 (bits)  Offset =  1F(16) : 0
                                            Type is boolean
           TRANSFER_SIZE_SPECIFIED         Type is field of a record  Length =  1 (bits)  Offset =  1F(16) : 1
                                            Type is boolean
           PRESET_STREAMING                Type is field of a record  Length =  1 (bits)  Offset =  1F(16) : 2
                                            Type is boolean
       ASSIGN_ACTIVE                   Type is field of a record  Length =  4(bytes)  Offset =  20(16) : 0
                                        Type is subrange  0 ..  2147483649
```

```
MTT$SMU_COMMUNICATIONS_BLOCK     Type is record  Length =  400
 HARDWARE_STATUS                  Type is field of a record  Length =  5(bytes)  Offset =  0(16) : 0
                                   MTT$SCB_HARDWARE_STATUS         Type is cybil array  Array element length  1
                                     MTT$SCB_HARDWARE_STATUS_COUNT   Type is subrange  0 ..  255
 KILL_180                         Type is field of a record  Length =  1(bytes)  Offset =  8(16) : 0
                                    Type is boolean
 PROCESSORS_LOGICALLY_ON          Type is field of a record  Length =  1(bytes)  Offset =  9(16) : 0
                                   OST$PROCESSOR_ID_SET            Type is set  Set length  8
                                     OST$PROCESSOR_ID                Type is subrange  0 ..  7
 PROCESSORS_WITH_STATE_CHANGED    Type is field of a record  Length =  1(bytes)  Offset =  A(16) : 0
                                   OST$PROCESSOR_ID_SET            Type is set  Set length  8
                                     OST$PROCESSOR_ID                Type is subrange  0 ..  7
 VECTOR_SIMULATION_CONTROL        Type is field of a record  Length =  3(bytes)  Offset =  B(16) : 0
   OST$VECTOR_SIMULATION_CONTROL    Type is record  Length =  3
     VECTOR_SIMULATION_ATTRIBUTE      Type is field of a record  Length =  1(bytes)  Offset =  B(16) : 0
                                       PMT$VECTOR_SIMULATION           Type is subrange  0 ..  255
     VECTOR_DIVIDE_DEGRADED           Type is field of a record  Length =  1(bytes)  Offset =  C(16) : 0
                                       OST$PROCESSOR_ID_SET            Type is set  Set length  8
                                         OST$PROCESSOR_ID                Type is subrange  0 ..  7
     ALL_VECTOR_DIVIDES_DEGRADED      Type is field of a record  Length =  1(bytes)  Offset =  D(16) : 0
                                       Type is boolean
 NOS_180_STATUS                   Type is field of a record  Length =  8(bytes)  Offset =  10(16) : 0
   MTT$SCB_180_STATUS               Type is record  Length =  8
     SYSTEM_STATUS                    Type is field of a record  Length =  4(bytes)  Offset =  10(16) : 0
       MTT$SYSTEM_STATUS_BLOCK          Type is record  Length =  4
         IDLE_STATUS_BLOCK                Type is field of a record  Length =  2(bytes)  Offset =  10(16) : 0
           MTT$IDLE_STATUS_BLOCK            Type is record  Length =  2
             REQUESTED_STATUS                 Type is field of a record  Length =  1(bytes)  Offset =  10(16) : 0
                                               MTT$SYSTEM_IDLE_UPDATE_REQUEST  Type is ordinal
                                                 MTC$IDLED_SYSTEM                Type is constant  =  1
                                                 MTC$RUNNING_SYSTEM              Type is constant  =  0
             ACTUAL_STATUS                    Type is field of a record  Length =  1(bytes)  Offset =  11(16) : 0
                                               MTT$SYSTEM_IDLE_UPDATE_REQUEST  Type is ordinal
                                                 MTC$IDLED_SYSTEM                Type is constant  =  1
                                                 MTC$RUNNING_SYSTEM              Type is constant  =  0
         STEP_STATUS_BLOCK                Type is field of a record  Length =  2(bytes)  Offset =  12(16) : 0
           MTT$STEP_STATUS_BLOCK            Type is record  Length =  2
             REQUESTED_STATUS                 Type is field of a record  Length =  1(bytes)  Offset =  12(16) : 0
                                               MTT$SYSTEM_STEP_UPDATE_REQUEST  Type is ordinal
                                                 MTC$STEPPED_SYSTEM              Type is constant  =  1
                                                 MTC$UNSTEPPED_SYSTEM            Type is constant  =  0
             ACTUAL_STATUS                    Type is field of a record  Length =  1(bytes)  Offset =  13(16) : 0
                                               MTT$SYSTEM_STEP_UPDATE_REQUEST  Type is ordinal
                                                 MTC$STEPPED_SYSTEM              Type is constant  =  1
                                                 MTC$UNSTEPPED_SYSTEM            Type is constant  =  0
     IDLE_CODE                        Type is field of a record  Length =  1(bytes)  Offset =  14(16) : 0
                                       SYT$180_IDLE_CODE               Type is ordinal
                                         SYC$IC_SOFTWARE_BREAKPOINT      Type is constant  =  10
                                         SYC$IC_DISK_ERROR               Type is constant  =  9
                                         SYC$IC_SHORT_POWER              Type is constant  =  8
                                         SYC$IC_STEP_COMMAND             Type is constant  =  7
                                         SYC$IC_IDLE_COMMAND             Type is constant  =  6
                                         SYC$IC_HARDWARE_IDLE            Type is constant  =  5
                                         SYC$IC_LONG_POWER               Type is constant  =  4
                                         SYC$IC_FATAL_SOFTWARE_ERROR     Type is constant  =  3
                                         SYC$IC_FATAL_HARDWARE_ERROR     Type is constant  =  2
                                         SYC$IC_SYSTEM_TERMINATED        Type is constant  =  1
                                         SYC$IC_NULL                     Type is constant  =  0
     FILL_1                           Type is field of a record  Length =  2(bytes)  Offset =  15(16) : 0
                                       Type is subrange  0 ..  65535
     CAUSE_OF_IDLE                    Type is field of a record  Length =  1(bytes)  Offset =  17(16) : 0
                                       SYT$180_IDLE_CODE               Type is ordinal
                                         SYC$IC_SOFTWARE_BREAKPOINT      Type is constant  =  10
                                         SYC$IC_DISK_ERROR               Type is constant  =  9
                                         SYC$IC_SHORT_POWER              Type is constant  =  8
                                         SYC$IC_STEP_COMMAND             Type is constant  =  7
                                         SYC$IC_IDLE_COMMAND             Type is constant  =  6
                                         SYC$IC_HARDWARE_IDLE            Type is constant  =  5
                                         SYC$IC_LONG_POWER               Type is constant  =  4
                                         SYC$IC_FATAL_SOFTWARE_ERROR     Type is constant  =  3
                                         SYC$IC_FATAL_HARDWARE_ERROR     Type is constant  =  2
                                         SYC$IC_SYSTEM_TERMINATED        Type is constant  =  1
                                         SYC$IC_NULL                     Type is constant  =  0
 NOS_SERVICE_FLAG                 Type is field of a record  Length =  8(bytes)  Offset =  18(16) : 0
                                   Type is integer
 CRITICAL_MESSAGE_TIME_STAMP      Type is field of a record  Length =  8(bytes)  Offset =  20(16) : 0
                                   Type is integer
 HARDWARE_STATUS_MESSAGES         Type is field of a record  Length =  360(bytes)  Offset =  28(16) : 0
                                   MTT$SCB_HARDWARE_STATUS_MSGS    Type is cybil array  Array element length  72
     MTT$SCB_HARDWARE_STATUS_MSG     Type is record  Length =  72
       MESSAGE_READ                    Type is field of a record  Length =  1(bytes)  Offset =  28(16) : 0
                                        Type is boolean
       MESSAGE                         Type is field of a record  Length =  71(bytes)  Offset =  29(16) : 0
                                        DPT$TOP_LINE_MESSAGE            Type is string  String length  71
```

```
OST$CPU_STATE_TABLE                Type is record  Length =  272
 FILL                               Type is field of a record  Length =  1(bytes)  Offset =  0(16) : 0
                                      Type is subrange  0 ..  255
 DISPATCHING_PRIORITY               Type is field of a record  Length =  1(bytes)  Offset =  1(16) : 0
                                     JMT$DISPATCHING_PRIORITY          Type is subrange  0 ..  15
 DUAL_STATE_PRIOR_SUBPRIORITY       Type is field of a record  Length =  2(bytes)  Offset =  2(16) : 0
   TMT$DUAL_STATE_PRIORITY_ENTRY      Type is record  Length =  2
     DUAL_STATE_PRIORITY                Type is field of a record  Length =  1(bytes)  Offset =  2(16) : 0
                                          Type is subrange  0 ..  7
     SUBPRIORITY                        Type is field of a record  Length =  1(bytes)  Offset =  3(16) : 0
                                          Type is subrange  0 ..  15
 MEMORY_PORT_MASK                   Type is field of a record  Length =  1(bytes)  Offset =  4(16) : 0
                                     OST$CPU_MEMORY_PORT_MASK          Type is subrange  0 ..  31
 CST_INDEX                          Type is field of a record  Length =  1(bytes)  Offset =  5(16) : 0
                                     OST$LOGICAL_PROCESSOR_ID          Type is subrange  0 ..  1
 PROCESSOR_STATE                    Type is field of a record  Length =  1(bytes)  Offset =  6(16) : 0
                                     CMT$ELEMENT_STATE                 Type is ordinal
                                       CMC$DOWN                          Type is constant  =  2
                                       CMC$OFF                           Type is constant  =  1
                                       CMC$ON                            Type is constant  =  0
 NEXT_PROCESSOR_STATE               Type is field of a record  Length =  1(bytes)  Offset =  7(16) : 0
                                     CMT$ELEMENT_STATE                 Type is ordinal
                                       CMC$DOWN                          Type is constant  =  2
                                       CMC$OFF                           Type is constant  =  1
                                       CMC$ON                            Type is constant  =  0
 CPU_ALIVE_FLAG                     Type is field of a record  Length =  8(bytes)  Offset =  8(16) : 0
                                      Type is integer
 TASKID                             Type is field of a record  Length =  3(bytes)  Offset =  10(16) : 0
   OST$GLOBAL_TASK_ID                 Type is record  Length =  3
     INDEX                              Type is field of a record  Length =  2(bytes)  Offset =  10(16) : 0
                                         OST$TASK_INDEX                  Type is subrange  0 ..  4095
     SEQNO                              Type is field of a record  Length =  1(bytes)  Offset =  12(16) : 0
                                          Type is subrange  0 ..  255
 AJLD                               Type is field of a record  Length =  1(bytes)  Offset =  13(16) : 0
                                     JMT$AJL_ORDINAL                   Type is subrange  0 ..  255
 DUAL_STATE_JPS                     Type is field of a record  Length =  4(bytes)  Offset =  14(16) : 0
                                      Type is subrange  0 ..  4294967295
 JCB_P                              Type is field of a record  Length =  6(bytes)  Offset =  18(16) : 0
                                      Type is pointer  Ptr object length  217
                                      Pointer to JMT$JOB_CONTROL_BLOCK          11c$record_kind
 CPU_STATE                          Type is field of a record  Length =  2(bytes)  Offset =  1E(16) : 0
   OST$CPU_STATE                      Type is record  Length =  2
     CURRENT_STATE                      Type is field of a record  Length =  1(bytes)  Offset =  1E(16) : 0
                                         OST$CPU_RUNNING_OR_STEPPED       Type is ordinal
                                           OSC$CPU_STEPPED                  Type is constant  =  1
                                           OSC$CPU_RUNNING                  Type is constant  =  0
     NEXT_STATE                         Type is field of a record  Length =  1(bytes)  Offset =  1F(16) : 0
                                         OST$CPU_RUNNING_OR_STEPPED       Type is ordinal
                                           OSC$CPU_STEPPED                  Type is constant  =  1
                                           OSC$CPU_RUNNING                  Type is constant  =  0
 XCB_P                              Type is field of a record  Length =  6(bytes)  Offset =  20(16) : 0
                                      Type is pointer  Ptr object length  956
                                      Pointer to OST$EXECUTION_CONTROL_BLOCK    11c$record_kind
 XCB_RMA                            Type is field of a record  Length =  8(bytes)  Offset =  28(16) : 0
                                      Type is integer
 DISPATCH_CONTROL                   Type is field of a record  Length =  5(bytes)  Offset =  30(16) : 0
   TMT$DISPATCH_CONTROL               Type is record  Length =  5
     CALL_DISPATCHER                    Type is field of a record  Length =  1(bytes)  Offset =  30(16) : 0
                                          Type is boolean
     RETHREAD_CURRENT_TASK              Type is field of a record  Length =  1(bytes)  Offset =  31(16) : 0
                                          Type is boolean
     NEW_TASK_STATUS                    Type is field of a record  Length =  1(bytes)  Offset =  32(16) : 0
                                         TMT$TASK_STATUS                  Type is ordinal
                                           TMC$TS_VOLUME_UNAVAILABLE        Type is constant  =  17
                                           TMC$TS_IO_WAIT_QUEUED            Type is constant  =  16
                                           TMC$TS_JOB_EVENT_QUEUE           Type is constant  =  15
                                           TMC$TS_SEGMENT_LOCK_WAIT         Type is constant  =  14
                                           TMC$TS_MEMORY_WAIT               Type is constant  =  13
                                           TMC$TS_PAGE_WAIT                 Type is constant  =  12
                                           TMC$TS_IO_WAIT_NOT_QUEUED        Type is constant  =  11
                                           TMC$TS_READY_BUT_SWAPPED         Type is constant  =  10
                                           TMC$TS_TIMEOUT_REQEXP_INFVLONG   Type is constant  =  9
                                           TMC$TS_TIMEOUT_REQEXP_INFLONG    Type is constant  =  8
                                           TMC$TS_EXECUTING                 Type is constant  =  7
                                           TMC$TS_TIMED_WAIT_NOT_QUEUED     Type is constant  =  6
                                           TMC$TS_TIMEOUT_REQEXP_LONGVLONG  Type is constant  =  5
                                           TMC$TS_TIMEOUT_REQEXP_LONGLONG   Type is constant  =  4
                                           TMC$TS_TIMEOUT_REQEXP_SHORTSHRT  Type is constant  =  3
                                           TMC$TS_READY_AND_SELECTED        Type is constant  =  2
                                           TMC$TS_READY                     Type is constant  =  1
                                           TMC$TS_NULL                      Type is constant  =  0
     FILL                               Type is field of a record  Length =  1(bytes)  Offset =  33(16) : 0
                                          Type is boolean
     ASYNCHRONOUS_INTERRUPTS_PENDING  Type is field of a record  Length =  1(bytes)  Offset =  34(16) : 0
                                          Type is boolean
 MAX_CPTIME                         Type is field of a record  Length =  8(bytes)  Offset =  38(16) : 0
                                      Type is integer
 ACCUMULATED_JOB_CPTIME             Type is field of a record  Length =  8(bytes)  Offset =  40(16) : 0
                                      Type is integer
 ACCUMULATED_MONITOR_CPTIME         Type is field of a record  Length =  8(bytes)  Offset =  48(16) : 0
                                      Type is integer
 EXT_INT_REQUEST                    Type is field of a record  Length =  1(bytes)  Offset =  50(16) : 0
   OST$EXTERNAL_INTERRUPT_REQUEST     Type is packed record  Length =  1
     TASK_SWITCH                        Type is field of a record  Length =  1 (bits)  Offset =  50(16) : 0
                                          Type is boolean
     PURGE_CACHE                        Type is field of a record  Length =  1 (bits)  Offset =  50(16) : 1
                                          Type is boolean
     PURGE_MAP                          Type is field of a record  Length =  1 (bits)  Offset =  50(16) : 2
                                          Type is boolean
     STEP_PROCESSOR                     Type is field of a record  Length =  1 (bits)  Offset =  50(16) : 3
                                          Type is boolean
 IDLE_CODE                          Type is field of a record  Length =  1(bytes)  Offset =  51(16) : 0
                                     SYT$180_IDLE_CODE                 Type is ordinal
                                       SYC$IC_SOFTWARE_BREAKPOINT        Type is constant  =  10
                                       SYC$IC_DISK_ERROR                 Type is constant  =  9
                                       SYC$IC_SHORT_POWER                Type is constant  =  8
                                       SYC$IC_STEP_COMMAND               Type is constant  =  7
                                       SYC$IC_IDLE_COMMAND               Type is constant  =  6
                                       SYC$IC_HARDWARE_IDLE              Type is constant  =  5
                                       SYC$IC_LONG_POWER                 Type is constant  =  4
                                       SYC$IC_FATAL_SOFTWARE_ERROR       Type is constant  =  3
                                       SYC$IC_FATAL_HARDWARE_ERROR       Type is constant  =  2
                                       SYC$IC_SYSTEM_TERMINATED          Type is constant  =  1
                                       SYC$IC_NULL                       Type is constant  =  0
 CST_INDEX_X_8                      Type is field of a record  Length =  1(bytes)  Offset =  52(16) : 0
                                      Type is subrange  0 ..  255
 TIME_LAST_CACHE_PURGE              Type is field of a record  Length =  8(bytes)  Offset =  58(16) : 0
                                      Type is integer
 TIME_LAST_MAP_REQUEST              Type is field of a record  Length =  8(bytes)  Offset =  60(16) : 0
                                      Type is integer
 MONITOR_MPS                        Type is field of a record  Length =  4(bytes)  Offset =  68(16) : 0
                                     OST$REAL_MEMORY_ADDRESS           Type is subrange  0 ..  4294967295
 ABORTED_TASK_COUNT                 Type is field of a record  Length =  2(bytes)  Offset =  6C(16) : 0
                                     OST$PARCEL                        Type is subrange  0 ..  65535
 DUE_COUNT                          Type is field of a record  Length =  2(bytes)  Offset =  6E(16) : 0
                                     OST$PARCEL                        Type is subrange  0 ..  65535
 ELEMENT_ID                         Type is field of a record  Length =  8(bytes)  Offset =  70(16) : 0
   OST$PROCESSOR_ELEMENT_ID           Type is record  Length =  8
     FILL                               Type is field of a record  Length =  4(bytes)  Offset =  70(16) : 0
                                         OST$HALFWORD                     Type is subrange  0 ..  4294967295
```

```
    ELEMENT_NUMBER                  Type is field of a record  Length =  1(bytes)  Offset =  74(16) : 0
                                     OST$PROCESSOR_ELEMENT_NUMBER    Type is subrange  0 ..  255
    MODEL_NUMBER                    Type is field of a record  Length =  1(bytes)  Offset =  75(16) : 0
                                     OST$PROCESSOR_MODEL_NUMBER      Type is subrange  0 ..  255
    SERIAL_NUMBER                   Type is field of a record  Length =  2(bytes)  Offset =  76(16) : 0
                                     OST$PROCESSOR_SERIAL_NUMBER     Type is subrange  0 ..  65535
IJL_ORDINAL                       Type is field of a record  Length =  2(bytes)  Offset =  78(16) : 0
  JMT$IJL_ORDINAL                   Type is packed record  Length =  2
    BLOCK_NUMBER                    Type is field of a record  Length =  11 (bits)  Offset =  78(16) : 0
                                     JMT$IJL_BLOCK_NUMBER            Type is subrange  0 ..  2047
    BLOCK_INDEX                     Type is field of a record  Length =  5 (bits)  Offset =  79(16) : 3
                                     JMT$IJL_BLOCK_INDEX             Type is subrange  0 ..  31
IJLE_P                            Type is field of a record  Length =  6(bytes)  Offset =  7A(16) : 0
                                    Type is pointer  Ptr object length  342
                                    Pointer to JMT$INITIATED_JOB_LIST_ENTRY   11c$record_kind
CPU_IDLE_STATISTICS               Type is field of a record  Length =  32(bytes)  Offset =  80(16) : 0
  OST$CPU_IDLE_STATISTICS           Type is record  Length =  32
    IDLE_NO_IO_ACTIVE               Type is field of a record  Length =  8(bytes)  Offset =  80(16) : 0
                                     Type is integer
    IDLE_IO_ACTIVE                  Type is field of a record  Length =  8(bytes)  Offset =  88(16) : 0
                                     Type is integer
    IDLE_START_TIME                 Type is field of a record  Length =  8(bytes)  Offset =  90(16) : 0
                                     Type is integer
    IDLE_TYPE                       Type is field of a record  Length =  1(bytes)  Offset =  98(16) : 0
                                     OST$IDLE_TYPE                   Type is ordinal
                                       OSC$IDLE_NO_IO_ACTIVE           Type is constant  =  2
                                       OSC$IDLE_WITH_IO_ACTIVE         Type is constant  =  1
                                       OSC$NOT_IDLE                    Type is constant  =  0
    IDLE_COUNT                      Type is field of a record  Length =  7(bytes)  Offset =  99(16) : 0
                                     Type is subrange  0 ..  72057594037927935
TRACE_CONTROL                     Type is field of a record  Length =  8(bytes)  Offset =  A0(16) : 0
  OST$CST_TRACE_CONTROL             Type is packed record  Length =  8
    FILL                            Type is field of a record  Length =  2(bytes)  Offset =  A0(16) : 0
                                     Type is subrange  0 ..  65535
    BUFFER_P                        Type is field of a record  Length =  6(bytes)  Offset =  A2(16) : 0
                                     Type is pointer  Ptr object length  1
                                     Pointer to 11c$cell_kind
TERMINATION_MESSAGE               Type is field of a record  Length =  80(bytes)  Offset =  A8(16) : 0
                                    Type is string  String length  80
REASON_FOR_CURRENT_STATE          Type is field of a record  Length =  1(bytes)  Offset =  F8(16) : 0
                                   OST$CPU_STATE_REASON            Type is ordinal
                                     OSC$CSR_CHANGED_BY_OPERATOR     Type is constant  =  4
                                     OSC$CSR_CPU_NOT_ALIVE_RECENTLY  Type is constant  =  3
                                     OSC$CSR_CPU_SAW_TOO_MANY_DUES   Type is constant  =  2
                                     OSC$CSR_CHANGED_BY_DFT          Type is constant  =  1
                                     OSC$CSR_STATE_AT_CTI_TIME       Type is constant  =  0
PRE_PROCESSED_FOR_RECONFIG        Type is field of a record  Length =  1(bytes)  Offset =  F9(16) : 0
                                   OST$PRE_PROCESSED_FOR_RECONFIG  Type is ordinal
                                     OSC$PPFR_PROCESSING_COMPLETE    Type is constant  =  2
                                     OSC$PPFR_PROCESSING_IN_PROGRESS Type is constant  =  1
                                     OSC$PPFR_NOT_PROCESSED          Type is constant  =  0
CPU_SHOULD_SPIN_INDEFINITELY      Type is field of a record  Length =  1(bytes)  Offset =  FA(16) : 0
                                    Type is boolean
PREVIOUS_PROCESSOR_STATE          Type is field of a record  Length =  1(bytes)  Offset =  FB(16) : 0
                                   CMT$ELEMENT_STATE              Type is ordinal
                                     CMC$DOWN                        Type is constant  =  2
                                     CMC$OFF                         Type is constant  =  1
                                     CMC$ON                          Type is constant  =  0
LOG_CPU_STATE_CHANGE              Type is field of a record  Length =  1(bytes)  Offset =  FC(16) : 0
                                    Type is boolean
NEXT_PTLQ_TO_DISPATCH             Type is field of a record  Length =  2(bytes)  Offset =  FD(16) : 0
                                   OST$TASK_INDEX                  Type is subrange  0 ..  4095
FILL_FF                           Type is field of a record  Length =  1(bytes)  Offset =  FF(16) : 0
                                    Type is subrange  0 ..  255
DISPATCHING_PRIORITY_INTEGER      Type is field of a record  Length =  8(bytes)  Offset =  100(16) : 0
                                    Type is integer
DUMMY_4                           Type is field of a record  Length =  8(bytes)  Offset =  108(16) : 0
                                    Type is integer
```

```
OST$EXECUTION_CONTROL_BLOCK      Type is record  Length =  956
 XP                               Type is field of a record  Length =  416(bytes)  Offset =  0(16) : 0
   OST$EXCHANGE_PACKAGE             Type is packed record  Length =  416
     P_REGISTER                       Type is field of a record  Length =  8(bytes)  Offset =  0(16) : 0
       OST$P_REGISTER                   Type is packed record  Length =  8
         UNDEFINED1                       Type is field of a record  Length =  2 (bits)  Offset =  0(16) : 0
                                            Type is subrange  0 ..  3
         GLOBAL_KEY                       Type is field of a record  Length =  6 (bits)  Offset =  0(16) : 2
                                            OST$KEY_LOCK_VALUE                Type is subrange  0 ..  63
         UNDEFINED2                       Type is field of a record  Length =  2 (bits)  Offset =  1(16) : 0
                                            Type is subrange  0 ..  3
         LOCAL_KEY                        Type is field of a record  Length =  6 (bits)  Offset =  1(16) : 2
                                            OST$KEY_LOCK_VALUE                Type is subrange  0 ..  63
         PVA                              Type is field of a record  Length =  6(bytes)  Offset =  2(16) : 0
           OST$PVA                          Type is packed record  Length =  6
             RING                             Type is field of a record  Length =  4 (bits)  Offset =  2(16) : 0
                                                OST$RING                          Type is subrange  0 ..  15
             SEG                              Type is field of a record  Length =  12 (bits)  Offset =  2(16) : 4
                                                OST$SEGMENT                       Type is subrange  0 ..  4095
             OFFSET                           Type is field of a record  Length =  4(bytes)  Offset =  4(16) : 0
                                                OST$SEGMENT_OFFSET                Type is subrange -2147483648 ..  2147483647
     UNDEFINED1                       Type is field of a record  Length =  4 (bits)  Offset =  8(16) : 0
                                        Type is subrange  0 ..  15
     VMID                             Type is field of a record  Length =  4 (bits)  Offset =  8(16) : 4
                                        OST$VIRTUAL_MACHINE_IDENTIFIER  Type is ordinal
                                          OSC$63_RESERVED                 Type is constant  =  15
                                          OSC$62_RESERVED                 Type is constant  =  14
                                          OSC$61_RESERVED                 Type is constant  =  13
                                          OSC$60_RESERVED                 Type is constant  =  12
                                          OSC$59_RESERVED                 Type is constant  =  11
                                          OSC$58_RESERVED                 Type is constant  =  10
                                          OSC$57_RESERVED                 Type is constant  =  9
                                          OSC$56_RESERVED                 Type is constant  =  8
                                          OSC$55_RESERVED                 Type is constant  =  7
                                          OSC$54_RESERVED                 Type is constant  =  6
                                          OSC$53_RESERVED                 Type is constant  =  5
                                          OSC$52_RESERVED                 Type is constant  =  4
                                          OSC$51_RESERVED                 Type is constant  =  3
                                          OSC$50_RESERVED                 Type is constant  =  2
                                          OSC$CYBER_170_MODE              Type is constant  =  1
                                          OSC$CYBER_180_MODE              Type is constant  =  0
     UNDEFINED2                       Type is field of a record  Length =  4 (bits)  Offset =  9(16) : 0
                                        Type is subrange  0 ..  15
     UVMID                            Type is field of a record  Length =  4 (bits)  Offset =  9(16) : 4
                                        OST$VIRTUAL_MACHINE_IDENTIFIER  Type is ordinal
                                          OSC$63_RESERVED                 Type is constant  =  15
                                          OSC$62_RESERVED                 Type is constant  =  14
                                          OSC$61_RESERVED                 Type is constant  =  13
                                          OSC$60_RESERVED                 Type is constant  =  12
                                          OSC$59_RESERVED                 Type is constant  =  11
                                          OSC$58_RESERVED                 Type is constant  =  10
                                          OSC$57_RESERVED                 Type is constant  =  9
                                          OSC$56_RESERVED                 Type is constant  =  8
                                          OSC$55_RESERVED                 Type is constant  =  7
                                          OSC$54_RESERVED                 Type is constant  =  6
                                          OSC$53_RESERVED                 Type is constant  =  5
                                          OSC$52_RESERVED                 Type is constant  =  4
                                          OSC$51_RESERVED                 Type is constant  =  3
                                          OSC$50_RESERVED                 Type is constant  =  2
                                          OSC$CYBER_170_MODE              Type is constant  =  1
                                          OSC$CYBER_180_MODE              Type is constant  =  0
     A0_DYNAMIC_SPACE_POINTER         Type is field of a record  Length =  6(bytes)  Offset =  A(16) : 0
                                        Type is pointer  Ptr object length  1
                                        Pointer to 11c$cell_kind
     FLAGS                            Type is field of a record  Length =  4 (bits)  Offset =  10(16) : 0
                                        OST$FLAGS                       Type is set  Set length  4
                                            Type is ordinal
                                              OSC$PROCESS_NOT_DAMAGED       Type is constant  =  3
                                              OSC$KEYPOINT_ENABLE           Type is constant  =  2
                                              OSC$ON_CONDITION              Type is constant  =  1
                                              OSC$CRITICAL_FRAME            Type is constant  =  0
     UNDEFINED3                       Type is field of a record  Length =  10 (bits)  Offset =  10(16) : 4
                                        Type is subrange  0 ..  1023
     TRAP_ENABLE                      Type is field of a record  Length =  2 (bits)  Offset =  11(16) : 6
                                        OST$TRAP_ENABLE                 Type is ordinal
                                          OSC$TRAPS_ENABLED_DELAY         Type is constant  =  3
                                          OSC$TRAPS_ENABLED               Type is constant  =  2
                                          OSC$TRAPS_UNDEFINED             Type is constant  =  1
                                          OSC$TRAPS_DISABLED              Type is constant  =  0
     A1_CURRENT_STACK_FRAME           Type is field of a record  Length =  6(bytes)  Offset =  12(16) : 0
                                        Type is pointer  Ptr object length  1
                                        Pointer to 11c$cell_kind
     USER_MASK                        Type is field of a record  Length =  2(bytes)  Offset =  18(16) : 0
                                        OST$USER_CONDITIONS             Type is set  Set length  16
                                          OST$USER_CONDITION              Type is ordinal
                                            OSC$INVALID_BDP_DATA            Type is constant  =  15
                                            OSC$ARITHMETIC_SIGNIFICANCE     Type is constant  =  14
                                            OSC$FP_INDEFINITE               Type is constant  =  13
                                            OSC$FP_SIGNIFICANCE_LOSS        Type is constant  =  12
                                            OSC$EXPONENT_UNDERFLOW          Type is constant  =  11
                                            OSC$EXPONENT_OVERFLOW           Type is constant  =  10
                                            OSC$ARITHMETIC_OVERFLOW         Type is constant  =  9
                                            OSC$DEBUG                       Type is constant  =  8
                                            OSC$DIVIDE_FAULT                Type is constant  =  7
                                            OSC$KEYPOINT                    Type is constant  =  6
                                            OSC$CRITICAL_FRAME_FLAG         Type is constant  =  5
                                            OSC$INTER_RING_POP              Type is constant  =  4
                                            OSC$PROCESS_INTERVAL_TIMER      Type is constant  =  3
                                            OSC$FREE_FLAG                   Type is constant  =  2
                                            OSC$UNIMPLEMENTED_INSTRUCTION   Type is constant  =  1
                                            OSC$PRIVILEGED_INSTRUCTION      Type is constant  =  0
     A2_PREVIOUS_SAVE_AREA            Type is field of a record  Length =  6(bytes)  Offset =  1A(16) : 0
                                        Type is pointer  Ptr object length  32
                                        Pointer to OST$MINIMUM_SAVE_AREA          11c$record_kind
     MONITOR_MASK                     Type is field of a record  Length =  2(bytes)  Offset =  20(16) : 0
                                        OST$MONITOR_CONDITIONS          Type is set  Set length  16
                                          OST$MONITOR_CONDITION           Type is ordinal
                                            OSC$TRAP_EXCEPTION              Type is constant  =  15
                                            OSC$SOFT_ERROR                  Type is constant  =  14
                                            OSC$OUT_CALL_IN_RETURN          Type is constant  =  13
                                            OSC$INVALID_SEGMENT_RING_0      Type is constant  =  12
                                            OSC$SYSTEM_INTERVAL_TIMER       Type is constant  =  11
                                            OSC$SYSTEM_CALL                 Type is constant  =  10
                                            OSC$PAGE_FAULT                  Type is constant  =  9
                                            OSC$EXTERNAL_INTERRUPT          Type is constant  =  8
                                            OSC$ENVIRONMENT_SPEC            Type is constant  =  7
                                            OSC$ACCESS_VIOLATION            Type is constant  =  6
                                            OSC$EXCHANGE_REQUEST            Type is constant  =  5
                                            OSC$ADDRESS_SPECIFICATION       Type is constant  =  4
                                            OSC$INSTRUCTION_SPEC            Type is constant  =  3
                                            OSC$SHORT_WARNING               Type is constant  =  2
                                            OSC$NOT_ASSIGNED                Type is constant  =  1
                                            OSC$DETECTED_UNCORRECTED_ERR    Type is constant  =  0
     A3                               Type is field of a record  Length =  6(bytes)  Offset =  22(16) : 0
                                        Type is pointer  Ptr object length  1
                                        Pointer to 11c$cell_kind
     USER_CONDITION_REGISTER          Type is field of a record  Length =  2(bytes)  Offset =  28(16) : 0
                                        OST$USER_CONDITIONS             Type is set  Set length  16
                                          OST$USER_CONDITION              Type is ordinal
                                            OSC$INVALID_BDP_DATA            Type is constant  =  15
```

```
                                OSC$ARITHMETIC_SIGNIFICANCE      Type is constant  =  14
                                OSC$FP_INDEFINITE                Type is constant  =  13
                                OSC$FP_SIGNIFICANCE_LOSS         Type is constant  =  12
                                OSC$EXPONENT_UNDERFLOW           Type is constant  =  11
                                OSC$EXPONENT_OVERFLOW            Type is constant  =  10
                                OSC$ARITHMETIC_OVERFLOW          Type is constant  =  9
                                OSC$DEBUG                        Type is constant  =  8
                                OSC$DIVIDE_FAULT                 Type is constant  =  7
                                OSC$KEYPOINT                     Type is constant  =  6
                                OSC$CRITICAL_FRAME_FLAG          Type is constant  =  5
                                OSC$INTER_RING_POP               Type is constant  =  4
                                OSC$PROCESS_INTERVAL_TIMER       Type is constant  =  3
                                OSC$FREE_FLAG                    Type is constant  =  2
                                OSC$UNIMPLEMENTED_INSTRUCTION    Type is constant  =  1
                                OSC$PRIVILEGED_INSTRUCTION       Type is constant  =  0
A4                          Type is field of a record  Length =  6(bytes)  Offset =  2A(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
MONITOR_CONDITION_REGISTER  Type is field of a record  Length =  2(bytes)  Offset =  30(16) : 0
                             OST$MONITOR_CONDITIONS          Type is set  Set length  16
                               OST$MONITOR_CONDITION           Type is ordinal
                                OSC$TRAP_EXCEPTION               Type is constant  =  15
                                OSC$SOFT_ERROR                   Type is constant  =  14
                                OSC$OUT_CALL_IN_RETURN           Type is constant  =  13
                                OSC$INVALID_SEGMENT_RING_0       Type is constant  =  12
                                OSC$SYSTEM_INTERVAL_TIMER        Type is constant  =  11
                                OSC$SYSTEM_CALL                  Type is constant  =  10
                                OSC$PAGE_FAULT                   Type is constant  =  9
                                OSC$EXTERNAL_INTERRUPT           Type is constant  =  8
                                OSC$ENVIRONMENT_SPEC             Type is constant  =  7
                                OSC$ACCESS_VIOLATION             Type is constant  =  6
                                OSC$EXCHANGE_REQUEST             Type is constant  =  5
                                OSC$ADDRESS_SPECIFICATION        Type is constant  =  4
                                OSC$INSTRUCTION_SPEC             Type is constant  =  3
                                OSC$SHORT_WARNING                Type is constant  =  2
                                OSC$NOT_ASSIGNED                 Type is constant  =  1
                                OSC$DETECTED_UNCORRECTED_ERR     Type is constant  =  0
A5                          Type is field of a record  Length =  6(bytes)  Offset =  32(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
UNDEFINED4                  Type is field of a record  Length =  4 (bits)  Offset =  38(16) : 0
                              Type is subrange  0 ..  15
KEYPOINT_CLASS_NUMBER       Type is field of a record  Length =  4 (bits)  Offset =  38(16) : 4
                             OST$KEYPOINT_CLASS              Type is subrange  0 ..  15
LAST_PROCESSOR_ID           Type is field of a record  Length =  1(bytes)  Offset =  39(16) : 0
                              Type is subrange  0 ..  255
A6                          Type is field of a record  Length =  6(bytes)  Offset =  3A(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
KEYPOINT_MASK               Type is field of a record  Length =  2(bytes)  Offset =  40(16) : 0
                             OST$KEYPOINT_MASK               Type is set  Set length  16
                               OST$KEYPOINT_CLASS              Type is subrange  0 ..  15
A7                          Type is field of a record  Length =  6(bytes)  Offset =  42(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
KEYPOINT_CODE_1             Type is field of a record  Length =  2(bytes)  Offset =  48(16) : 0
                              Type is subrange  0 ..  65535
A8                          Type is field of a record  Length =  6(bytes)  Offset =  4A(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
KEYPOINT_CODE_2             Type is field of a record  Length =  2(bytes)  Offset =  50(16) : 0
                              Type is subrange  0 ..  65535
A9                          Type is field of a record  Length =  6(bytes)  Offset =  52(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
PROCESS_INTERVAL_TIMER_1    Type is field of a record  Length =  2(bytes)  Offset =  58(16) : 0
                              Type is subrange  0 ..  65535
AA                          Type is field of a record  Length =  6(bytes)  Offset =  5A(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
PROCESS_INTERVAL_TIMER_2    Type is field of a record  Length =  2(bytes)  Offset =  60(16) : 0
                              Type is subrange  0 ..  65535
AB                          Type is field of a record  Length =  6(bytes)  Offset =  62(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
BASE_CONSTANT_1             Type is field of a record  Length =  2(bytes)  Offset =  68(16) : 0
                              Type is subrange  0 ..  65535
AC                          Type is field of a record  Length =  6(bytes)  Offset =  6A(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
BASE_CONSTANT_2             Type is field of a record  Length =  2(bytes)  Offset =  70(16) : 0
                              Type is subrange  0 ..  65535
AD                          Type is field of a record  Length =  6(bytes)  Offset =  72(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
MODEL_DEPENDENT_FLAGS       Type is field of a record  Length =  2(bytes)  Offset =  78(16) : 0
                              Type is subrange  0 ..  65535
AE                          Type is field of a record  Length =  6(bytes)  Offset =  7A(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
UNDEFINED5                  Type is field of a record  Length =  4 (bits)  Offset =  80(16) : 0
                              Type is subrange  0 ..  15
SEGMENT_TABLE_LENGTH        Type is field of a record  Length =  12 (bits)  Offset =  80(16) : 4
                             OST$SEGMENT                     Type is subrange  0 ..  4095
AF                          Type is field of a record  Length =  6(bytes)  Offset =  82(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
X_REGISTERS                 Type is field of a record  Length =  128(bytes)  Offset =  88(16) : 0
                              Type is cybil array  Array element length  8
                                Type is integer
MODEL_DEPENDENT_WORD        Type is field of a record  Length =  8(bytes)  Offset =  108(16) : 0
                              Type is integer
SEGMENT_TABLE_ADDRESS_1     Type is field of a record  Length =  2(bytes)  Offset =  110(16) : 0
                              Type is subrange  0 ..  65535
UNTRANSLATABLE_POINTER      Type is field of a record  Length =  6(bytes)  Offset =  112(16) : 0
  OST$PVA                     Type is packed record  Length =  6
    RING                        Type is field of a record  Length =  4 (bits)  Offset =  112(16) : 0
                                 OST$RING                    Type is subrange  0 ..  15
    SEG                         Type is field of a record  Length =  12 (bits)  Offset =  112(16) : 4
                                 OST$SEGMENT                 Type is subrange  0 ..  4095
    OFFSET                      Type is field of a record  Length =  4(bytes)  Offset =  114(16) : 0
                                 OST$SEGMENT_OFFSET          Type is subrange -2147483648 ..  2147483647
SEGMENT_TABLE_ADDRESS_2     Type is field of a record  Length =  2(bytes)  Offset =  118(16) : 0
                              Type is subrange  0 ..  65535
TRAP_POINTER                Type is field of a record  Length =  6(bytes)  Offset =  11A(16) : 0
                              Type is pointer  Ptr object length  1
                              Pointer to 11c$cell_kind
DEBUG_INDEX                 Type is field of a record  Length =  6 (bits)  Offset =  120(16) : 0
                              Type is subrange  0 ..  63
UNDEFINED6                  Type is field of a record  Length =  3 (bits)  Offset =  120(16) : 6
                              Type is subrange  0 ..  7
DEBUG_MASK_REGISTER         Type is field of a record  Length =  7 (bits)  Offset =  121(16) : 1
  OST$DEBUG_MASK              Type is packed record  Length =  1
    END_OF_LIST_SEEN_FLAG       Type is field of a record  Length =  1 (bits)  Offset =  121(16) : 1
                                  Type is boolean
    SCAN_IN_PROGRESS            Type is field of a record  Length =  1 (bits)  Offset =  121(16) : 2
                                  Type is boolean
    CODES                       Type is field of a record  Length =  5 (bits)  Offset =  121(16) : 3
                                  Type is cybil array  Array element length  1
                                    Type is boolean
DEBUG_LIST_POINTER          Type is field of a record  Length =  6(bytes)  Offset =  122(16) : 0
                              Type is pointer  Ptr object length  512
```

```
                                  Pointer to OST$DEBUG_LIST                    11c$cybil_array_kind
    TOS_REGISTERS                 Type is field of a record  Length =  120(bytes)  Offset =  128(16) : 0
                                    Type is cybil array  Array element length  8
        OST$TOP_OF_STACK_POINTER      Type is packed record  Length =  8
          UNDEFINED                     Type is field of a record  Length =  12 (bits)  Offset =  128(16) : 0
                                          Type is subrange  0 ..  4095
          LARGEST_RING_NUMBER           Type is field of a record  Length =  4 (bits)  Offset =  129(16) : 4
                                          OST$RING                              Type is subrange  0 ..  15
          PVA                           Type is field of a record  Length =  6(bytes)  Offset =  12A(16) : 0
            OST$PVA                       Type is packed record  Length =  6
              RING                          Type is field of a record  Length =  4 (bits)  Offset =  12A(16) : 0
                                              OST$RING                          Type is subrange  0 ..  15
              SEG                           Type is field of a record  Length =  12 (bits)  Offset =  12A(16) : 4
                                              OST$SEGMENT                       Type is subrange  0 ..  4095
              OFFSET                        Type is field of a record  Length =  4(bytes)  Offset =  12C(16) : 0
                                              OST$SEGMENT_OFFSET                Type is subrange -2147483648 ..  2147483647
MONITOR_FLAGS                   Type is field of a record  Length =  2(bytes)  Offset =  1A0(16) : 0
                                  SYT$MONITOR_FLAGS                 Type is set  Set length  16
                                    SYT$MONITOR_FLAG                  Type is ordinal
                                      SYC$MF_SPARE_15                   Type is constant  =  15
                                      SYC$MF_SPARE_14                   Type is constant  =  14
                                      SYC$MF_SPARE_13                   Type is constant  =  13
                                      SYC$MF_SPARE_12                   Type is constant  =  12
                                      SYC$MF_SPARE_11                   Type is constant  =  11
                                      SYC$MF_FOR_KEYPOINT_TRACEBACK     Type is constant  =  10
                                      MMC$MF_SHADOW_FILE_REFERENCE      Type is constant  =  9
                                      SYC$MF_CPU_CONFIGURATION_CHANGE   Type is constant  =  8
                                      MMC$MF_SEGMENT_MGR_FLAG           Type is constant  =  7
                                      SYC$MF_DUMP_JOB_ENVIRONMENT       Type is constant  =  6
                                      SYC$MF_SYSTEM_DEBUGGER            Type is constant  =  5
                                      SYC$MF_INVOKE_SYSDEBUG            Type is constant  =  4
                                      MMC$MF_VOLUME_UNAVAILABLE         Type is constant  =  3
                                      SYC$MF_CAUSE_JOB_RECOVERY         Type is constant  =  2
                                      SYC$MF_HANG_TASK                  Type is constant  =  1
                                      TMC$MF_CAUSE_JOB_FREE_FLAG_TRAP   Type is constant  =  0
PROCESSOR_SELECTIONS            Type is field of a record  Length =  1(bytes)  Offset =  1A2(16) : 0
                                  OST$PROCESSOR_ID_SET              Type is set  Set length  8
                                    OST$PROCESSOR_ID                  Type is subrange  0 ..  7
LAST_LPID_FOR_TASK              Type is field of a record  Length =  1(bytes)  Offset =  1A3(16) : 0
                                  OST$PROCESSOR_ID                  Type is subrange  0 ..  7
SYSTEM_TASK_ID                  Type is field of a record  Length =  1(bytes)  Offset =  1A4(16) : 0
                                  TMT$SYSTEM_TASK_ID                Type is subrange  0 ..  30
CRITICAL_TASK                   Type is field of a record  Length =  1(bytes)  Offset =  1A5(16) : 0
                                  Type is boolean
TASK_HAS_TERMINATED             Type is field of a record  Length =  1(bytes)  Offset =  1A6(16) : 0
                                  Type is boolean
STLC_ALLOCATION                 Type is field of a record  Length =  1(bytes)  Offset =  1A7(16) : 0
                                  Type is boolean
SPECIAL_TRAP_COUNT              Type is field of a record  Length =  1(bytes)  Offset =  1A8(16) : 0
                                  Type is subrange  0 ..  255
GLOBAL_TASK_ID                  Type is field of a record  Length =  3(bytes)  Offset =  1A9(16) : 0
  OST$GLOBAL_TASK_ID              Type is record  Length =  3
    INDEX                           Type is field of a record  Length =  2(bytes)  Offset =  1A9(16) : 0
                                      OST$TASK_INDEX                    Type is subrange  0 ..  4095
    SEQNO                           Type is field of a record  Length =  1(bytes)  Offset =  1AB(16) : 0
                                      Type is subrange  0 ..  255
PARENT_GLOBAL_TASK_ID           Type is field of a record  Length =  3(bytes)  Offset =  1AC(16) : 0
  OST$GLOBAL_TASK_ID              Type is record  Length =  3
    INDEX                           Type is field of a record  Length =  2(bytes)  Offset =  1AC(16) : 0
                                      OST$TASK_INDEX                    Type is subrange  0 ..  4095
    SEQNO                           Type is field of a record  Length =  1(bytes)  Offset =  1AE(16) : 0
                                      Type is subrange  0 ..  255
WAIT_INHIBITED                  Type is field of a record  Length =  1(bytes)  Offset =  1AF(16) : 0
                                  Type is boolean
SYSTEM_TABLE_LOCK_COUNT         Type is field of a record  Length =  8(bytes)  Offset =  1B0(16) : 0
                                  Type is integer
SYSTEM_FLAGS                    Type is field of a record  Length =  4(bytes)  Offset =  1B8(16) : 0
                                  TMT$SYSTEM_FLAGS                  Type is set  Set length  32
                                    OST$SYSTEM_FLAG                   Type is ordinal
                                      TMC$FLAG_AVAILABLE_31             Type is constant  =  31
                                      TMC$FLAG_AVAILABLE_30             Type is constant  =  30
                                      TMC$FLAG_AVAILABLE_29             Type is constant  =  29
                                      TMC$FLAG_AVAILABLE_28             Type is constant  =  28
                                      TMC$FLAG_AVAILABLE_27             Type is constant  =  27
                                      TMC$FLAG_AVAILABLE_26             Type is constant  =  26
                                      TMC$FLAG_AVAILABLE_25             Type is constant  =  25
                                      TMC$FLAG_AVAILABLE_24             Type is constant  =  24
                                      TMC$FLAG_AVAILABLE_23             Type is constant  =  23
                                      JMC$MESSAGE_WAITING_FLAG_ID       Type is constant  =  22
                                      MMC$VOLUME_UNAVAILABLE_FLAG       Type is constant  =  21
                                      MMC$FAILED_FILE_ALLOC_FLAG        Type is constant  =  20
                                      RFC$PP_RESPONSE_AVAILABLE         Type is constant  =  19
                                      NLC$CC_WORK_LIST_FLAG             Type is constant  =  18
                                      OSC$SYSTEM_UNSTEP_RESUME_FLAG     Type is constant  =  17
                                      OFC$OPERATOR_BREAK_FLAG           Type is constant  =  16
                                      DSC$LOG_DFT_FLAG_ID               Type is constant  =  15
                                      IOC$SUBSYSTEM_IO_COMPLETED        Type is constant  =  14
                                      SYC$JOB_RECOVERY_FLAG             Type is constant  =  13
                                      NAC$NOTIFY_ROUTING_ME             Type is constant  =  12
                                      NAC$CHANNELNET_LOCAL_EVENT        Type is constant  =  11
                                      NAC$XI_LOCAL_EVENT                Type is constant  =  10
                                      NLC$XT_WORK_LIST_FLAG             Type is constant  =  9
                                      NAC$NETWORK_INPUT_RECEIVED        Type is constant  =  8
                                      DSC$RETRIEVE_SYSTEM_MESSAGE       Type is constant  =  7
                                      JMC$KILL_JOB_FLAG                 Type is constant  =  6
                                      JMC$LOGOUT_FLAG_ID                Type is constant  =  5
                                      TMC$MAINFRAME_LINKED_SIGNALS      Type is constant  =  4
                                      JMC$TERMINATE_JOB_FLAG            Type is constant  =  3
                                      PMC$SF_TERMINATE_TASK             Type is constant  =  2
                                      AVC$MONITOR_STATISTICS_FLAG       Type is constant  =  1
                                      PMC$KILL_TASK_FLAG                Type is constant  =  0
RECEIVED_MESSAGE_LIST           Type is field of a record  Length =  8(bytes)  Offset =  1C0(16) : 0
  NAT$RECEIVED_MESSAGE_LIST       Type is record  Length =  8
    NEXT_RECEIVED_MESSAGE           Type is field of a record  Length =  6(bytes)  Offset =  1C0(16) : 0
                                      Type is pointer  Ptr object length  24
                                      Pointer to NAT$RECEIVED_MESSAGE_DESCRIPTOR11c$record_kind
    FILL                            Type is field of a record  Length =  2(bytes)  Offset =  1C6(16) : 0
                                      Type is subrange  0 ..  65535
TASK_IS_TERMINATING             Type is field of a record  Length =  1(bytes)  Offset =  1C8(16) : 0
                                  Type is boolean
TASK_HAS_BEEN_RETHREADED        Type is field of a record  Length =  1(bytes)  Offset =  1C9(16) : 0
                                  Type is boolean
SYSTEM_GIVE_UP_CPU              Type is field of a record  Length =  1(bytes)  Offset =  1CA(16) : 0
                                  Type is boolean
SUBSYSTEM_GIVE_UP_CPU           Type is field of a record  Length =  1(bytes)  Offset =  1CB(16) : 0
                                  Type is boolean
SUBSYSTEM_LOCK_PRIORITY_COUNT   Type is field of a record  Length =  1(bytes)  Offset =  1CC(16) : 0
                                  Type is subrange  0 ..  255
DISPATCHING_PRIORITY            Type is field of a record  Length =  1(bytes)  Offset =  1CD(16) : 0
                                  JMT$DISPATCHING_PRIORITY          Type is subrange  0 ..  15
DISPATCHING_PRIORITY_BIAS_ID    Type is field of a record  Length =  1(bytes)  Offset =  1CE(16) : 0
                                  Type is ordinal
                                    JMC$DPB_ABSOLUTE                  Type is constant  =  2
                                    JMC$DPB_NEGATIVE                  Type is constant  =  1
                                    JMC$DPB_POSITIVE                  Type is constant  =  0
DISPATCHING_PRIORITY_BIAS       Type is field of a record  Length =  1(bytes)  Offset =  1CF(16) : 0
                                  JMT$DISPATCHING_PRIORITY          Type is subrange  0 ..  15
SYSTEM_ERROR_COUNT              Type is field of a record  Length =  1(bytes)  Offset =  1D0(16) : 0
                                  Type is subrange  0 ..  255
LINK                            Type is field of a record  Length =  6(bytes)  Offset =  1D1(16) : 0
                                  Type is pointer  Ptr object length  956
                                  Pointer to OST$EXECUTION_CONTROL_BLOCK    11c$record_kind
```

```
TASK_CONTROL_BLOCK                Type is field of a record, Length =  6(bytes)  Offset =  1D7(16) : 0
                                    Type is pointer  Ptr object length  1
                                    Pointer to llc$cell_kind
TASK_ID                           Type is field of a record  Length =  4(bytes)  Offset =  1DD(16) : 0
                                    PMT$TASK_ID                    Type is subrange  0 ..  4294967295
STACK_PAGES_SAVED                 Type is field of a record  Length =  2(bytes)  Offset =  1E1(16) : 0
                                    Type is cybil array  Array element length  1
                                      Type is boolean
SDT_OFFSET                        Type is field of a record  Length =  4(bytes)  Offset =  1E3(16) : 0
                                    OST$VALID_RELATIVE_POINTER .    Type is subrange  0 ..  2147483647
SDTX_OFFSET                       Type is field of a record  Length =  4(bytes)  Offset =  1E7(16) : 0
                                    OST$VALID_RELATIVE_POINTER     Type is subrange  0 ..  2147483647
PIT_COUNT                         Type is field of a record  Length =  6(bytes)  Offset =  1EB(16) : 0
                                    OST$FREE_RUNNING_CLOCK         Type is subrange  0 ..  281474976710655
CP_TIME                           Type is field of a record  Length =  10(bytes)  Offset =  1F1(16) : 0
  OST$CP_TIME                       Type is record  Length =  10
    TIME_SPENT_IN_JOB_MODE            Type is field of a record  Length =  5(bytes)  Offset =  1F1(16) : 0
                                        OST$CP_TIME_VALUE              Type is subrange  0 ..  1099511627775
    TIME_SPENT_IN_MTR_MODE            Type is field of a record  Length =  5(bytes)  Offset =  1F6(16) : 0
                                        OST$CP_TIME_VALUE              Type is subrange  0 ..  1099511627775
PAGE_WAIT_INFO                    Type is field of a record  Length =  6(bytes)  Offset =  1FB(16) : 0
  MMT$XCB_PAGE_WAIT_INFO            Type is record  Length =  6
    PVA                               Type is field of a record  Length =  6(bytes)  Offset =  1FB(16) : 0
                                        Type is pointer  Ptr object length  1
                                        Pointer to llc$cell_kind
TIMESLICE                         Type is field of a record  Length =  8(bytes)  Offset =  201(16) : 0
  JMT$TIME_SLICE_VALUES             Type is record  Length =  8
    MINOR                             Type is field of a record  Length =  4(bytes)  Offset =  201(16) : 0
                                        OST$TASK_TIME_SLICE            Type is subrange  0 ..  4294967295
    MAJOR                             Type is field of a record  Length =  4(bytes)  Offset =  205(16) : 0
                                        OST$TASK_TIME_SLICE            Type is subrange  0 ..  4294967295
RELATIVE_TASK_PRIORITY            Type is field of a record  Length =  1(bytes)  Offset =  209(16) : 0
                                    Type is subrange  0 ..  255
RING1_TERMINATION_REASON          Type is field of a record  Length =  1(bytes)  Offset =  20A(16) : 0
                                    OST$RING1_TERMINATION_REASON   Type is ordinal
                                      OSC$RTR_SFT_FULL               Type is constant  =  1
                                      OSC$RTR_NON                    Type is constant  =  0
MAXWS_AIO_SLOWDOWN                Type is field of a record  Length =  3(bytes)  Offset =  20B(16) : 0
                                    Type is subrange  0 ..  16777215
MONITOR_FAULTS                    Type is field of a record  Length =  198(bytes)  Offset =  20E(16) : 0
  TMT$MONITOR_FAULT_BUFFER          Type is record  Length =  198
    PRESENT                           Type is field of a record  Length =  1(bytes)  Offset =  20E(16) : 0
                                        Type is cybil array  Array element length  1
                                          Type is boolean
    RESERVED                          Type is field of a record  Length =  1(bytes)  Offset =  20F(16) : 0
                                        Type is cybil array  Array element length  1
                                          Type is boolean
    BUFFER                            Type is field of a record  Length =  196(bytes)  Offset =  210(16) : 0
                                        Type is cybil array  Array element length  49
        OST$MONITOR_FAULT                 Type is record  Length =  49
          PVA                               Type is field of a record  Length =  6(bytes)  Offset =  210(16) : 0
            OST$PVA                           Type is packed record  Length =  6
              RING                              Type is field of a record  Length =  4 (bits)  Offset =  210(16) : 0
                                                  OST$RING                       Type is subrange  0 ..  15
              SEG                               Type is field of a record  Length =  12 (bits)  Offset =  210(16) : 4
                                                  OST$SEGMENT                    Type is subrange  0 ..  4095
              OFFSET                            Type is field of a record  Length =  4(bytes)  Offset =  212(16) : 0
                                                  OST$SEGMENT_OFFSET             Type is subrange -2147483648 ..  2147483647
          A0                                Type is field of a record  Length =  6(bytes)  Offset =  216(16) : 0
                                              Type is pointer  Ptr object length  1
                                              Pointer to llc$cell_kind
          A1                                Type is field of a record  Length =  6(bytes)  Offset =  21C(16) : 0
                                              Type is pointer  Ptr object length  1
                                              Pointer to llc$cell_kind
          A2                                Type is field of a record  Length =  6(bytes)  Offset =  222(16) : 0
                                              Type is pointer  Ptr object length  32
                                              Pointer to OST$MINIMUM_SAVE_AREA          llc$record_kind
          IDENTIFIER                        Type is field of a record  Length =  1(bytes)  Offset =  228(16) : 0
                                             TMT$MONITOR_FAULT_IDENTIFIERS    Type is ordinal
                                               TMC$DUMMY_FAULT                  Type is constant  =  6
                                               SYC$SYSTEM_CORE_CONDITION        Type is constant  =  5
                                               TMC$UNKNOWN_SYSTEM_REQ_FAULT     Type is constant  =  4
                                               MMC$SEGMENT_FAULT_PROCESSOR_ID   Type is constant  =  3
                                               TMC$MCR_FAULT                    Type is constant  =  2
                                               TMC$BROKEN_TASK_FAULT_ID         Type is constant  =  1
                                               TMC$NULL_FAULT                   Type is constant  =  0
           --- Variant --TMC$BROKEN_TASK_FAULT_ID
            BROKEN_TASK_FAULT               Type is field of a record  Length =  24(bytes)  Offset =  229(16) : 0
              TMT$BROKEN_TASK_MONITOR_FAULT   Type is record  Length =  24
                TRAP_ENABLE                     Type is field of a record  Length =  1(bytes)  Offset =  229(16) : 0
                                                  OST$TRAP_ENABLE                Type is ordinal
                                                    OSC$TRAPS_ENABLED_DELAY        Type is constant  =  3
                                                    OSC$TRAPS_ENABLED              Type is constant  =  2
                                                    OSC$TRAPS_UNDEFINED            Type is constant  =  1
                                                    OSC$TRAPS_DISABLED             Type is constant  =  0
                BROKEN_TASK_CONDITION           Type is field of a record  Length =  1(bytes)  Offset =  22A(16) : 0
                                                  TMT$BROKEN_TASK_CONDITION      Type is ordinal
                                                    TMC$BTC_SYSTEM_ERROR           Type is constant  =  6
                                                    TMC$BTC_UCR_TRAPS_DISABLED     Type is constant  =  5
                                                    TMC$BTC_MCR_TRAPS_DISABLED     Type is constant  =  4
                                                    TMC$BTC_INVALID_P              Type is constant  =  3
                                                    TMC$BTC_INVALID_AO             Type is constant  =  2
                                                    TMC$BTC_MF_TRAPS_DISABLED      Type is constant  =  1
                                                    TMC$BTC_MNTR_FAULT_BUFFER_FULL Type is constant  =  0
             --- Variant --TMC$BTC_MNTR_FAULT_BUFFER_FULL .. TMC$BTC_UCR_TRAPS_DISABLED
              P                               Type is field of a record  Length =  8(bytes)  Offset =  22B(16) : 0
                OST$P_REGISTER                  Type is packed record  Length =  8
                  UNDEFINED1                      Type is field of a record  Length =  2 (bits)  Offset =  22B(16) : 0
                                                    Type is subrange  0 ..  3
                  GLOBAL_KEY                      Type is field of a record  Length =  6 (bits)  Offset =  22B(16) : 2
                                                    OST$KEY_LOCK_VALUE             Type is subrange  0 ..  63
                  UNDEFINED2                      Type is field of a record  Length =  2 (bits)  Offset =  22C(16) : 0
                                                    Type is subrange  0 ..  3
                  LOCAL_KEY                       Type is field of a record  Length =  6 (bits)  Offset =  22C(16) : 2
                                                    OST$KEY_LOCK_VALUE             Type is subrange  0 ..  63
                  PVA                             Type is field of a record  Length =  6(bytes)  Offset =  22D(16) : 0
                    OST$PVA                         Type is packed record  Length =  6
                      RING                            Type is field of a record  Length =  4 (bits)  Offset =  22D(16) : 0
                                                        OST$RING                       Type is subrange  0 ..  15
                      SEG                             Type is field of a record  Length =  12 (bits)  Offset =  22D(16) : 4
                                                        OST$SEGMENT                    Type is subrange  0 ..  4095
                      OFFSET                          Type is field of a record  Length =  4(bytes)  Offset =  22F(16) : 0
                                                        OST$SEGMENT_OFFSET             Type is subrange -2147483648 ..  2147483647
              A0                                Type is field of a record  Length =  6(bytes)  Offset =  233(16) : 0
                                                  Type is pointer  Ptr object length  1
                                                  Pointer to llc$cell_kind
              MONITOR_CONDITION_REGISTER        Type is field of a record  Length =  2(bytes)  Offset =  239(16) : 0
                                                  OST$MONITOR_CONDITIONS         Type is set  Set length  16
                                                    OST$MONITOR_CONDITION          Type is ordinal
                                                      OSC$TRAP_EXCEPTION             Type is constant  =  15
                                                      OSC$SOFT_ERROR                 Type is constant  =  14
                                                      OSC$OUT_CALL_IN_RETURN         Type is constant  =  13
                                                      OSC$INVALID_SEGMENT_RING_0     Type is constant  =  12
                                                      OSC$SYSTEM_INTERVAL_TIMER      Type is constant  =  11
                                                      OSC$SYSTEM_CALL                Type is constant  =  10
                                                      OSC$PAGE_FAULT                 Type is constant  =  9
                                                      OSC$EXTERNAL_INTERRUPT         Type is constant  =  8
                                                      OSC$ENVIRONMENT_SPEC           Type is constant  =  7
                                                      OSC$ACCESS_VIOLATION           Type is constant  =  6
                                                      OSC$EXCHANGE_REQUEST           Type is constant  =  5
                                                      OSC$ADDRESS_SPECIFICATION      Type is constant  =  4
                                                      OSC$INSTRUCTION_SPEC           Type is constant  =  3
```

```
                                        OSC$SHORT_WARNING                Type is constant  =  2
                                        OSC$NOT_ASSIGNED                 Type is constant  =  1
                                        OSC$DETECTED_UNCORRECTED_ERR     Type is constant  =  0
        USER_CONDITION_REGISTER       Type is field of a record  Length =  2(bytes)  Offset =  23B(16) : 0
                                        OST$USER_CONDITIONS            Type is set  Set length  16
                                          OST$USER_CONDITION             Type is ordinal
                                            OSC$INVALID_BDP_DATA         Type is constant  =  15
                                            OSC$ARITHMETIC_SIGNIFICANCE  Type is constant  =  14
                                            OSC$FP_INDEFINITE            Type is constant  =  13
                                            OSC$FP_SIGNIFICANCE_LOSS     Type is constant  =  12
                                            OSC$EXPONENT_UNDERFLOW       Type is constant  =  11
                                            OSC$EXPONENT_OVERFLOW        Type is constant  =  10
                                            OSC$ARITHMETIC_OVERFLOW      Type is constant  =  9
                                            OSC$DEBUG                    Type is constant  =  8
                                            OSC$DIVIDE_FAULT             Type is constant  =  7
                                            OSC$KEYPOINT                 Type is constant  =  6
                                            OSC$CRITICAL_FRAME_FLAG      Type is constant  =  5
                                            OSC$INTER_RING_POP           Type is constant  =  4
                                            OSC$PROCESS_INTERVAL_TIMER   Type is constant  =  3
                                            OSC$FREE_FLAG                Type is constant  =  2
                                            OSC$UNIMPLEMENTED_INSTRUCTION  Type is constant  =  1
                                            OSC$PRIVILEGED_INSTRUCTION   Type is constant  =  0
        MONITOR_FAULT_ID              Type is field of a record  Length =  1(bytes)  Offset =  23D(16) : 0
                                        TMT$MONITOR_FAULT_IDENTIFIERS  Type is ordinal
                                          TMC$DUMMY_FAULT                Type is constant  =  6
                                          SYC$SYSTEM_CORE_CONDITION      Type is constant  =  5
                                          TMC$UNKNOWN_SYSTEM_REQ_FAULT   Type is constant  =  4
                                          MMC$SEGMENT_FAULT_PROCESSOR_ID  Type is constant  =  3
                                          TMC$MCR_FAULT                  Type is constant  =  2
                                          TMC$BROKEN_TASK_FAULT_ID       Type is constant  =  1
                                          TMC$NULL_FAULT                 Type is constant  =  0
      --- Variant --TMC$BTC_SYSTEM_ERROR
        CALLER_P_REGISTER             Type is field of a record  Length =  8(bytes)  Offset =  22B(16) : 0
          OST$P_REGISTER                Type is packed record  Length =  8
            UNDEFINED1                    Type is field of a record  Length =  2 (bits)  Offset =  22B(16) : 0
                                            Type is subrange  0 ..  3
            GLOBAL_KEY                    Type is field of a record  Length =  6 (bits)  Offset =  22B(16) : 2
                                            OST$KEY_LOCK_VALUE             Type is subrange  0 ..  63
            UNDEFINED2                    Type is field of a record  Length =  2 (bits)  Offset =  22C(16) : 0
                                            Type is subrange  0 ..  3
            LOCAL_KEY                     Type is field of a record  Length =  6 (bits)  Offset =  22C(16) : 2
                                            OST$KEY_LOCK_VALUE             Type is subrange  0 ..  63
            PVA                           Type is field of a record  Length =  6(bytes)  Offset =  22D(16) : 0
              OST$PVA                       Type is packed record  Length =  6
                RING                          Type is field of a record  Length =  4 (bits)  Offset =  22D(16) : 0
                                                OST$RING                       Type is subrange  0 ..  15
                SEG                           Type is field of a record  Length =  12 (bits)  Offset =  22D(16) : 4
                                                OST$SEGMENT                    Type is subrange  0 ..  4095
                OFFSET                        Type is field of a record  Length =  4(bytes)  Offset =  22F(16) : 0
                                                OST$SEGMENT_OFFSET             Type is subrange -2147483648 ..  2147483647
        STATUS_P                      Type is field of a record  Length =  6(bytes)  Offset =  233(16) : 0
                                        Type is pointer  Ptr object length  264
                                        Pointer to OST$STATUS                    11c$record_kind
        TEXT_P                        Type is field of a record  Length =  8(bytes)  Offset =  239(16) : 0
                                        Type is pointer  Ptr object length  8
                                        Pointer to 11c$string_kind
      --- Variant --TMC$MCR_FAULT
        MCR_FAULT                   Type is field of a record  Length =  8(bytes)  Offset =  229(16) : 0
          TMT$MCR_FAULTS              Type is record  Length =  8
            FAULTS                      Type is field of a record  Length =  2(bytes)  Offset =  229(16) : 0
                                          OST$MONITOR_CONDITIONS         Type is set  Set length  16
                                            OST$MONITOR_CONDITION          Type is ordinal
                                              OSC$TRAP_EXCEPTION           Type is constant  =  15
                                              OSC$SOFT_ERROR               Type is constant  =  14
                                              OSC$OUT_CALL_IN_RETURN       Type is constant  =  13
                                              OSC$INVALID_SEGMENT_RING_0   Type is constant  =  12
                                              OSC$SYSTEM_INTERVAL_TIMER    Type is constant  =  11
                                              OSC$SYSTEM_CALL              Type is constant  =  10
                                              OSC$PAGE_FAULT               Type is constant  =  9
                                              OSC$EXTERNAL_INTERRUPT       Type is constant  =  8
                                              OSC$ENVIRONMENT_SPEC         Type is constant  =  7
                                              OSC$ACCESS_VIOLATION         Type is constant  =  6
                                              OSC$EXCHANGE_REQUEST         Type is constant  =  5
                                              OSC$ADDRESS_SPECIFICATION    Type is constant  =  4
                                              OSC$INSTRUCTION_SPEC         Type is constant  =  3
                                              OSC$SHORT_WARNING            Type is constant  =  2
                                              OSC$NOT_ASSIGNED             Type is constant  =  1
                                              OSC$DETECTED_UNCORRECTED_ERR  Type is constant  =  0
        UNTRANSLATABLE_POINTER      Type is field of a record  Length =  6(bytes)  Offset =  22B(16) : 0
          OST$PVA                     Type is packed record  Length =  6
            RING                        Type is field of a record  Length =  4 (bits)  Offset =  22B(16) : 0
                                          OST$RING                       Type is subrange  0 ..  15
            SEG                         Type is field of a record  Length =  12 (bits)  Offset =  22B(16) : 4
                                          OST$SEGMENT                    Type is subrange  0 ..  4095
            OFFSET                      Type is field of a record  Length =  4(bytes)  Offset =  22D(16) : 0
                                          OST$SEGMENT_OFFSET             Type is subrange -2147483648 ..  2147483647
      --- Variant --MMC$SEGMENT_FAULT_PROCESSOR_ID
        SEGMENT_ACCESS_FAULT        Type is field of a record  Length =  7(bytes)  Offset =  229(16) : 0
          MMT$SEGMENT_ACCESS_CONDITION  Type is record  Length =  7
            IDENTIFIER                  Type is field of a record  Length =  1(bytes)  Offset =  229(16) : 0
                                          PMT$CONDITION_IDENTIFIER       Type is subrange  0 ..  255
            SEGMENT                     Type is field of a record  Length =  6(bytes)  Offset =  22A(16) : 0
                                          Type is pointer  Ptr object length  1
                                          Pointer to 11c$cell_kind
      --- Variant --TMC$DUMMY_FAULT
        CONTENTS                    Type is field of a record  Length =  24(bytes)  Offset =  229(16) : 0
                                      OST$MONITOR_FAULT_CONTENTS      Type is cybil array  Array element length  1
                                        Type is subrange  0 ..  255
SIGNALS                           Type is field of a record  Length =  146(bytes)  Offset =  2D4(16) : 0
  TMT$SIGNAL_BUFFER                 Type is record  Length =  146
    PRESENT                           Type is field of a record  Length =  1(bytes)  Offset =  2D4(16) : 0
                                        Type is cybil array  Array element length  1
                                          Type is boolean
    RESERVED                          Type is field of a record  Length =  1(bytes)  Offset =  2D5(16) : 0
                                        Type is cybil array  Array element length  1
                                          Type is boolean
    BUFFER                            Type is field of a record  Length =  144(bytes)  Offset =  2D6(16) : 0
                                        Type is cybil array  Array element length  36
      TMT$SIGNAL                          Type is record  Length =  36
        ORIGINATOR                          Type is field of a record  Length =  3(bytes)  Offset =  2D6(16) : 0
          OST$GLOBAL_TASK_ID                  Type is record  Length =  3
            INDEX                               Type is field of a record  Length =  2(bytes)  Offset =  2D6(16) : 0
                                                  OST$TASK_INDEX             Type is subrange  0 ..  4095
            SEQNO                               Type is field of a record  Length =  1(bytes)  Offset =  2D8(16) : 0
                                                  Type is subrange  0 ..  255
        SIGNAL                              Type is field of a record  Length =  33(bytes)  Offset =  2D9(16) : 0
          PMT$SIGNAL                          Type is record  Length =  33
            IDENTIFIER                          Type is field of a record  Length =  1(bytes)  Offset =  2D9(16) : 0
                                                  PMT$SIGNAL_ID                Type is ordinal
                                                    TMC$SIGNAL_AVAILABLE_63      Type is constant  =  63
                                                    TMC$SIGNAL_AVAILABLE_62      Type is constant  =  62
                                                    TMC$SIGNAL_AVAILABLE_61      Type is constant  =  61
                                                    TMC$SIGNAL_AVAILABLE_60      Type is constant  =  60
                                                    TMC$SIGNAL_AVAILABLE_59      Type is constant  =  59
                                                    TMC$SIGNAL_AVAILABLE_58      Type is constant  =  58
                                                    TMC$SIGNAL_AVAILABLE_57      Type is constant  =  57
                                                    TMC$SIGNAL_AVAILABLE_56      Type is constant  =  56
                                                    TMC$SIGNAL_AVAILABLE_55      Type is constant  =  55
                                                    TMC$SIGNAL_AVAILABLE_54      Type is constant  =  54
                                                    TMC$SIGNAL_AVAILABLE_53      Type is constant  =  53
                                                    TMC$SIGNAL_AVAILABLE_52      Type is constant  =  52
                                                    TMC$SIGNAL_AVAILABLE_51      Type is constant  =  51
```

```
                                    TMC$SIGNAL_AVAILABLE_50          Type is constant  =  50
                                    TMC$SIGNAL_AVAILABLE_49          Type is constant  =  49
                                    TMC$SIGNAL_AVAILABLE_48          Type is constant  =  48
                                    TMC$SIGNAL_AVAILABLE_47          Type is constant  =  47
                                    TMC$SIGNAL_AVAILABLE_46          Type is constant  =  46
                                    TMC$SIGNAL_AVAILABLE_45          Type is constant  =  45
                                    TMC$SIGNAL_AVAILABLE_44          Type is constant  =  44
                                    TMC$SIGNAL_AVAILABLE_43          Type is constant  =  43
                                    TMC$SIGNAL_AVAILABLE_42          Type is constant  =  42
                                    TMC$SIGNAL_AVAILABLE_41          Type is constant  =  41
                                    TMC$SIGNAL_AVAILABLE_40          Type is constant  =  40
                                    TMC$SIGNAL_AVAILABLE_39          Type is constant  =  39
                                    TMC$SIGNAL_AVAILABLE_38          Type is constant  =  38
                                    TMC$SIGNAL_AVAILABLE_37          Type is constant  =  37
                                    TMC$SIGNAL_AVAILABLE_36          Type is constant  =  36
                                    TMC$SIGNAL_AVAILABLE_35          Type is constant  =  35
                                    TMC$SIGNAL_AVAILABLE_34          Type is constant  =  34
                                    TMC$SIGNAL_AVAILABLE_33          Type is constant  =  33
                                    TMC$SIGNAL_AVAILABLE_32          Type is constant  =  32
                                    TMC$SIGNAL_AVAILABLE_31          Type is constant  =  31
                                    TMC$SIGNAL_AVAILABLE_30          Type is constant  =  30
                                    TMC$SIGNAL_AVAILABLE_29          Type is constant  =  29
                                    TMC$SIGNAL_AVAILABLE_28          Type is constant  =  28
                                    TMC$SIGNAL_AVAILABLE_27          Type is constant  =  27
                                    TMC$SIGNAL_AVAILABLE_26          Type is constant  =  26
                                    TMC$SIGNAL_AVAILABLE_25          Type is constant  =  25
                                    TMC$SIGNAL_AVAILABLE_24          Type is constant  =  24
                                    TMC$SIGNAL_AVAILABLE_23          Type is constant  =  23
                                    TMC$SIGNAL_AVAILABLE_22          Type is constant  =  22
                                    NAC$SE_DISCONNECT_SIGNAL         Type is constant  =  21
                                    NAC$SE_SEND_DATA_SIGNAL          Type is constant  =  20
                                    NAC$SE_DELIVER_DATA_SIGNAL       Type is constant  =  19
                                    NAC$GT_DELIVER_CONNECT_REQUEST   Type is constant  =  18
                                    NAC$GT_SEND_DATA                 Type is constant  =  17
                                    NAC$GT_DELIVER_DATA              Type is constant  =  16
                                    PMC$MULTI_TASK_CONDITION         Type is constant  =  15
                                    TMC$SIGNAL_AVAILABLE_14          Type is constant  =  14
                                    TMC$SIGNAL_AVAILABLE_13          Type is constant  =  13
                                    NAC$NETWORK_DEVICE_ERROR         Type is constant  =  12
                                    TMC$SIGNAL_AVAILABLE_11          Type is constant  =  11
                                    TMC$SIGNAL_AVAILABLE_10          Type is constant  =  10
                                    JMC$JOB_RESOURCE_SIGNAL_ID       Type is constant  =  9
                                    TMC$SIGNAL_AVAILABLE_8           Type is constant  =  8
                                    JMC$SENSE_SWITCH_SIGNAL_ID       Type is constant  =  7
                                    TMC$SIGNAL_AVAILABLE_6           Type is constant  =  6
                                    JMC$TIMESHARING_SIGNAL_ID        Type is constant  =  5
                                    PMC$SS_CHILD_TERMINATED          Type is constant  =  4
                                    IFC$SIGNAL_ID                    Type is constant  =  3
                                    MLC$SIGNAL_ID                    Type is constant  =  2
                                    OFC$SIGNAL                       Type is constant  =  1
                                    TMC$SIGNAL_AVAILABLE_0           Type is constant  =  0
CONTENTS                          Type is field of a record  Length =  32(bytes)  Offset =  2DA(16) : 0
                                   PMT$SIGNAL_CONTENTS                Type is cybil array  Array element length  1
                                     Type is subrange  0 ..  255
PAGING_STATISTICS                 Type is field of a record  Length =  24(bytes)  Offset =  366(16) : 0
  OST$PAGING_STATISTICS             Type is record  Length =  24
    PAGE_IN_COUNT                     Type is field of a record  Length =  4(bytes)  Offset =  366(16) : 0
                                        Type is subrange  0 ..  4294967295
    PAGES_RECLAIMED_FROM_QUEUE        Type is field of a record  Length =  4(bytes)  Offset =  36A(16) : 0
                                        Type is subrange  0 ..  4294967295
    NEW_PAGES_ASSIGNED                Type is field of a record  Length =  4(bytes)  Offset =  36E(16) : 0
                                        Type is subrange  0 ..  4294967295
    PAGES_FROM_SERVER                 Type is field of a record  Length =  4(bytes)  Offset =  372(16) : 0
                                        Type is subrange  0 ..  4294967295
    PAGE_FAULT_COUNT                  Type is field of a record  Length =  6(bytes)  Offset =  376(16) : 0
                                        Type is subrange  0 ..  281474976710655
    WORKING_SET_MAX_USED              Type is field of a record  Length =  2(bytes)  Offset =  37C(16) : 0
                                        Type is subrange  0 ..  65535
SAVE9                             Type is field of a record  Length =  31(bytes)  Offset =  37E(16) : 0
                                    Type is string  String length  31
IOCB_P                            Type is field of a record  Length =  6(bytes)  Offset =  39D(16) : 0
                                    Type is pointer  Ptr object length  1
                                    Pointer to llc$cell_kind
KEYPOINT_ENABLE                   Type is field of a record  Length =  1(bytes)  Offset =  3A3(16) : 0
                                    Type is boolean
KEYPOINT_REGISTER_ENABLE          Type is field of a record  Length =  1(bytes)  Offset =  3A4(16) : 0
                                    Type is boolean
TIME_LAST_DUE                     Type is field of a record  Length =  5(bytes)  Offset =  3A5(16) : 0
                                   OST$CP_TIME_VALUE                Type is subrange  0 ..  1099511627775
PROC_MALF_COUNT                   Type is field of a record  Length =  1(bytes)  Offset =  3AA(16) : 0
                                    Type is subrange  0 ..  255
SHADOW_REFERENCE_INFO             Type is field of a record  Length =  13(bytes)  Offset =  3AB(16) : 0
  MMT$SHADOW_REFERENCE_INFO         Type is record  Length =  13
    SOURCE_PVA                        Type is field of a record  Length =  6(bytes)  Offset =  3AB(16) : 0
                                        Type is pointer  Ptr object length  1
                                        Pointer to llc$cell_kind
    DESTINATION_PVA                   Type is field of a record  Length =  6(bytes)  Offset =  3B1(16) : 0
                                        Type is pointer  Ptr object length  1
                                        Pointer to llc$cell_kind
    PAGE_COUNT                        Type is field of a record  Length =  1(bytes)  Offset =  3B7(16) : 0
                                        Type is subrange  0 ..  255
ASSIGN_ACTIVE_SFID                Type is field of a record  Length =  4(bytes)  Offset =  3B8(16) : 0
  GFT$SYSTEM_FILE_IDENTIFIER        Type is record  Length =  4
    FILE_ENTRY_INDEX                  Type is field of a record  Length =  2(bytes)  Offset =  3B8(16) : 0
                                       GFT$FILE_DESCRIPTOR_INDEX        Type is subrange  0 ..  65535
    RESIDENCE                         Type is field of a record  Length =  1(bytes)  Offset =  3BA(16) : 0
                                       GFT$TABLE_RESIDENCE              Type is ordinal
                                         GFC$TR_SYSTEM_WAIT_RECOVERY      Type is constant  =  3
                                         GFC$TR_JOB                       Type is constant  =  2
                                         GFC$TR_SYSTEM                    Type is constant  =  1
                                         GFC$TR_NULL_RESIDENCE            Type is constant  =  0
    FILE_HASH                         Type is field of a record  Length =  1(bytes)  Offset =  3BB(16) : 0
                                        Type is subrange  0 ..  255
```

```
TMT$PRIMARY_TASK_LIST_ENTRY      Type is record  Length =  32
 PTL_THREAD                       Type is field of a record  Length =  2(bytes)  Offset =  0(16) : 0
  OST$TASK_INDEX                    Type is subrange  0 ..  4095
 SEQUENCE_NUMBER                  Type is field of a record  Length =  1(bytes)  Offset =  2(16) : 0
   Type is subrange  0 ..  255
 XCB_OFFSET                       Type is field of a record  Length =  3(bytes)  Offset =  3(16) : 0
  TMT$XCB_OFFSET_SIZE               Type is subrange  0 ..  16777215
 IJL_ORDINAL                      Type is field of a record  Length =  2(bytes)  Offset =  6(16) : 0
   JMT$IJL_ORDINAL                  Type is packed record  Length =  2
     BLOCK_NUMBER                     Type is field of a record  Length =  11 (bits)  Offset =  6(16) : 0
      JMT$IJL_BLOCK_NUMBER              Type is subrange  0 ..  2047
     BLOCK_INDEX                      Type is field of a record  Length =  5 (bits)  Offset =  7(16) : 3
      JMT$IJL_BLOCK_INDEX               Type is subrange  0 ..  31
 STATUS                           Type is field of a record  Length =  1(bytes)  Offset =  8(16) : 0
  TMT$TASK_STATUS                   Type is ordinal
    TMC$TS_VOLUME_UNAVAILABLE         Type is constant  =  17
    TMC$TS_IO_WAIT_QUEUED             Type is constant  =  16
    TMC$TS_JOB_EVENT_QUEUE            Type is constant  =  15
    TMC$TS_SEGMENT_LOCK_WAIT          Type is constant  =  14
    TMC$TS_MEMORY_WAIT                Type is constant  =  13
    TMC$TS_PAGE_WAIT                  Type is constant  =  12
    TMC$TS_IO_WAIT_NOT_QUEUED         Type is constant  =  11
    TMC$TS_READY_BUT_SWAPPED          Type is constant  =  10
    TMC$TS_TIMEOUT_REQEXP_INFVLONG    Type is constant  =  9
    TMC$TS_TIMEOUT_REQEXP_INFLONG     Type is constant  =  8
    TMC$TS_EXECUTING                  Type is constant  =  7
    TMC$TS_TIMED_WAIT_NOT_QUEUED      Type is constant  =  6
    TMC$TS_TIMEOUT_REQEXP_LONGVLONG   Type is constant  =  5
    TMC$TS_TIMEOUT_REQEXP_LONGLONG    Type is constant  =  4
    TMC$TS_TIMEOUT_REQEXP_SHORTSHRT   Type is constant  =  3
    TMC$TS_READY_AND_SELECTED         Type is constant  =  2
    TMC$TS_READY                      Type is constant  =  1
    TMC$TS_NULL                       Type is constant  =  0
 NEW_TASK_STATUS                  Type is field of a record  Length =  1(bytes)  Offset =  9(16) : 0
  TMT$TASK_STATUS                   Type is ordinal
    TMC$TS_VOLUME_UNAVAILABLE         Type is constant  =  17
    TMC$TS_IO_WAIT_QUEUED             Type is constant  =  16
    TMC$TS_JOB_EVENT_QUEUE            Type is constant  =  15
    TMC$TS_SEGMENT_LOCK_WAIT          Type is constant  =  14
    TMC$TS_MEMORY_WAIT                Type is constant  =  13
    TMC$TS_PAGE_WAIT                  Type is constant  =  12
    TMC$TS_IO_WAIT_NOT_QUEUED         Type is constant  =  11
    TMC$TS_READY_BUT_SWAPPED          Type is constant  =  10
    TMC$TS_TIMEOUT_REQEXP_INFVLONG    Type is constant  =  9
    TMC$TS_TIMEOUT_REQEXP_INFLONG     Type is constant  =  8
    TMC$TS_EXECUTING                  Type is constant  =  7
    TMC$TS_TIMED_WAIT_NOT_QUEUED      Type is constant  =  6
    TMC$TS_TIMEOUT_REQEXP_LONGVLONG   Type is constant  =  5
    TMC$TS_TIMEOUT_REQEXP_LONGLONG    Type is constant  =  4
    TMC$TS_TIMEOUT_REQEXP_SHORTSHRT   Type is constant  =  3
    TMC$TS_READY_AND_SELECTED         Type is constant  =  2
    TMC$TS_READY                      Type is constant  =  1
    TMC$TS_NULL                       Type is constant  =  0
 IDLE_STATUS                      Type is field of a record  Length =  1(bytes)  Offset =  A(16) : 0
  TMT$IDLE_STATUS                   Type is ordinal
    TMC$IS_IDLED_SCHED_NOTIFIED       Type is constant  =  3
    TMC$IS_IDLED                      Type is constant  =  2
    TMC$IS_IDLE_INITIATED             Type is constant  =  1
    TMC$IS_NOT_IDLED                  Type is constant  =  0
 QUEUE_LINK                       Type is field of a record  Length =  4(bytes)  Offset =  B(16) : 0
   TMT$TASK_QUEUE_LINK              Type is record  Length =  4
     HEAD                             Type is field of a record  Length =  2(bytes)  Offset =  B(16) : 0
      OST$TASK_INDEX                    Type is subrange  0 ..  4095
     TAIL                             Type is field of a record  Length =  2(bytes)  Offset =  D(16) : 0
      OST$TASK_INDEX                    Type is subrange  0 ..  4095
 MONITOR_FLAGS                    Type is field of a record  Length =  2(bytes)  Offset =  F(16) : 0
  SYT$MONITOR_FLAGS                 Type is set  Set length  16
    SYT$MONITOR_FLAG                  Type is ordinal
      SYC$MF_SPARE_15                   Type is constant  =  15
      SYC$MF_SPARE_14                   Type is constant  =  14
      SYC$MF_SPARE_13                   Type is constant  =  13
      SYC$MF_SPARE_12                   Type is constant  =  12
      SYC$MF_SPARE_11                   Type is constant  =  11
      SYC$MF_FOR_KEYPOINT_TRACEBACK     Type is constant  =  10
      MMC$MF_SHADOW_FILE_REFERENCE      Type is constant  =  9
      SYC$MF_CPU_CONFIGURATION_CHANGE   Type is constant  =  8
      MMC$MF_SEGMENT_MGR_FLAG           Type is constant  =  7
      SYC$MF_DUMP_JOB_ENVIRONMENT       Type is constant  =  6
      SYC$MF_SYSTEM_DEBUGGER            Type is constant  =  5
      SYC$MF_INVOKE_SYSDEBUG            Type is constant  =  4
      MMC$MF_VOLUME_UNAVAILABLE         Type is constant  =  3
      SYC$MF_CAUSE_JOB_RECOVERY         Type is constant  =  2
      SYC$MF_HANG_TASK                  Type is constant  =  1
      TMC$MF_CAUSE_JOB_FREE_FLAG_TRAP   Type is constant  =  0
 SYSTEM_FLAGS                     Type is field of a record  Length =  4(bytes)  Offset =  11(16) : 0
  TMT$SYSTEM_FLAGS                  Type is set  Set length  32
    OST$SYSTEM_FLAG                   Type is ordinal
      TMC$FLAG_AVAILABLE_31             Type is constant  =  31
      TMC$FLAG_AVAILABLE_30             Type is constant  =  30
      TMC$FLAG_AVAILABLE_29             Type is constant  =  29
      TMC$FLAG_AVAILABLE_28             Type is constant  =  28
      TMC$FLAG_AVAILABLE_27             Type is constant  =  27
      TMC$FLAG_AVAILABLE_26             Type is constant  =  26
      TMC$FLAG_AVAILABLE_25             Type is constant  =  25
      TMC$FLAG_AVAILABLE_24             Type is constant  =  24
      TMC$FLAG_AVAILABLE_23             Type is constant  =  23
      JMC$MESSAGE_WAITING_FLAG_ID       Type is constant  =  22
      MMC$VOLUME_UNAVAILABLE_FLAG       Type is constant  =  21
      MMC$FAILED_FILE_ALLOC_FLAG        Type is constant  =  20
      RFC$PP_RESPONSE_AVAILABLE         Type is constant  =  19
      NLC$CC_WORK_LIST_FLAG             Type is constant  =  18
      OSC$SYSTEM_UNSTEP_RESUME_FLAG     Type is constant  =  17
      OFC$OPERATOR_BREAK_FLAG           Type is constant  =  16
      DSC$LOG_DFT_FLAG_ID               Type is constant  =  15
      IOC$SUBSYSTEM_IO_COMPLETED        Type is constant  =  14
      SYC$JOB_RECOVERY_FLAG             Type is constant  =  13
      NAC$NOTIFY_ROUTING_ME             Type is constant  =  12
      NAC$CHANNELNET_LOCAL_EVENT        Type is constant  =  11
      NAC$XI_LOCAL_EVENT                Type is constant  =  10
      NLC$XT_WORK_LIST_FLAG             Type is constant  =  9
      NAC$NETWORK_INPUT_RECEIVED        Type is constant  =  8
      DSC$RETRIEVE_SYSTEM_MESSAGE       Type is constant  =  7
      JMC$KILL_JOB_FLAG                 Type is constant  =  6
      JMC$LOGOUT_FLAG_ID                Type is constant  =  5
      TMC$MAINFRAME_LINKED_SIGNALS      Type is constant  =  4
      JMC$TERMINATE_JOB_FLAG            Type is constant  =  3
      PMC$SF_TERMINATE_TASK             Type is constant  =  2
      AVC$MONITOR_STATISTICS_FLAG       Type is constant  =  1
      PMC$KILL_TASK_FLAG                Type is constant  =  0
 PTL_FLAGS                        Type is field of a record  Length =  1(bytes)  Offset =  15(16) : 0
   TMT$PTL_FLAGS                    Type is packed record  Length =  1
     SUBSYSTEM_LOCKS_SET              Type is field of a record  Length =  1 (bits)  Offset =  15(16) : 0
       Type is boolean
     WAIT_INHIBITED                   Type is field of a record  Length =  2 (bits)  Offset =  15(16) : 1
      TMT$WAIT_INHIBITED                Type is ordinal
        TMC$WI_WAIT_SELECTED_R3           Type is constant  =  3
        TMC$WI_WAIT_SELECTED              Type is constant  =  2
        TMC$WI_WAIT_INHIBITED             Type is constant  =  1
        TMC$WI_NULL                       Type is constant  =  0
 DISPATCHING_PRIORITY             Type is field of a record  Length =  1(bytes)  Offset =  16(16) : 0
```