## STP data-2

| Block | Function | Task |
|-------|----------|------|
| EWT | Event wait table | JSH |
| GRT | Generic resource table | JCM, DQM |
| ILT | ISP link table | IQM |
| IPT | Define path table | IQM |
| IST | ISPMAIN status table | IQM |
| JSQBM | JSQ bit map | JSH |
| JXT | Job execution table | JSH |
| LCT | Link configuration table | SCP |
| LIT | Link interface table | SCP |
| LXT | Link extension table | SCP |
| MST | Memory segment table | JSH |
| PDI | PDM information table | PDM |
| PDS | Permanent dataset table | PDM, EXP |
| PHR | Physical request table | DQM, TQM, IQM |
| PXT | Processor execution table | JSH |
| QDT | Queued dataset table | JSH |
| REPLY | Inter task commun. reply header | all |
| REQST | Inter task commun. request header | all |
| RIT | Registered ID table | IQM |
| RQT | Disk queue manager request table | DQM |
| SAC | SSD active table | DQM, EXEC |
| SDT | System dataset table | SCP, EXP, JSH |
| SDR | System directory | EXP |
| SQP | SSD queued packet table | DQM |
| STS | System task suspend table | all |
| SXT | System execution table | all |
| TDT | Tape device table | TQM |
| TNT | Table name table | EXP |
| TQP | Task queued packet for EXEC | all |
| TXT | User task execution table | JSH, EXP |
| UDT | User channel driver table | IQM |
| VPT | ISP virtual circuit pointer | ISP |

| Trace | Function | |
|-------|----------|---|
| DQMT | Disk queue manager trace | DQM |
| ITCT | Inter task communication trace | all |
| TQMT | Tape queue manager trace | TQM |

| Pool | Function | |
|------|----------|---|
| POOL1 | Working storage | SPM, JCM, SCP, LOG |
| POOL2 | Not used | |
| POOL3 | Tape labels | TQM |
| POOL4 | | IQM |
| POOL5 | | IQM |

| | | |
|---|---|---|
| Installation parameters | | all |
| Task software stacks | | all |
| Task IDs | | all |
| STP stop buffer | | all |
| STP-EXEC communication area | | all |
| DSC/DXT buffers in high mem. – static | | PDM |
| Station buffers in high mem. – dynamic | | JSH, SCP |

## Common subroutines-1

| Block | Entry | Function |
|-------|-------|----------|
| BFMAN | BFMAN | Manage system buffers |
| BTAD | BTAD | Binary to ASCII decimal conversion |
| CCOPY | CCOPY | Character copy routine |
| CHAINS | (J)CHAIN(F) | (JTA) chain (front) routine |
| CHKSM | CHKSUM | Compute check sum |
| CLEAR | CLEAR | Clear an area of memory |
| CONFIG | CONFIG | Process operator CONFIG command |
| CONVTS | CONVTS | Convert Rtc to timestamps |
| COPY | COPY | Copy a block of memory |
| CRACKER | IND | Analyse JOB statement |
| DCJXT | DCJXT | Decrement no. of jobs awaiting JXT |
| FIXJXPR | FIXJXO | Convert JXT offset to ordinal |
| FNDJB | FNDJB | Find a job on an SDT queue |
| GETOWN | GETOWN | Get owner of a PDN |
| GETPARM | GETPARM | Generalized parameter routine |
| GTMEM | GTMEM | Allocate memory from an STP pool |
| GTTCB | GTTCB | Allocate TCB in the JTA |
| GTTXT | GTTXT | Allocate TXT in STP |
| IJTSUB | | Inter job transfer subroutines |
| | IJDQD | Drop queued data |
| | IJFQM | Find queued messages |
| | IJFRB | Find receiving buffer |
| | IJPQM | Process queued message |
| | IJSMT | Send message to target |
| JMEM | JMEM(DE)AL | (De)allocate JTA memory |
| JTADNT | (REL)GETDNT | Release/allocate a DNT in the JTA |
| LGMSG | LGMSG | Enter message in log file |
| MSGQUE | MSGQUE | Enqueue a station message |
| PACKETQ | (PT)GTPKT | (De)enqueue MIOP packet |
| QMSG | | Interactive message management |
| | ENQMSG | Enqueue a message |
| | FREEMSG | Free a message string |
| | NXTMSG | Fetch next message |
| QUEUES | (EQ)DESD2 | (En)dequeue SDT entry |
| REQRPL | | Inter-task communication |
| | GETREPLY | Get asynchronous reply |
| | GETREQ | Get asynchronous request |
| | PUTREPLY | Put asynchronous reply |
| | PUTREQ | Put asynchronous request |
| | TSKREQ | Synchronous task request/reply |
| | REREPLY | Intermediate reply |
| RLMEM | RLMEM | Release memory from STP pool |
| RLTXT | RLTXT | Release TXT from STP and TCB from JTA |
| SD2PDD | SD2PD | Move SDT fields to PDD |
| STPDAT | (GET)RELDAT | (Get)release DAT pages from STP |
| STPERR | ERROR(0)1 | STP error conditions (hangs COS) |
| STPMEM | (P)MEM(DE)AL | (Partial De)allocate memory from STP pool |
| STPTIME | | Time conversion routines |
| | RT2JD | Rtc value to Julian date |
| | JD2RT | Julian date to Rtc |

# Cray Research (UK) Ltd

# Dump analysis work aid

This work aid has been produced to assist analysts in the analysis of COS dumps. It is hoped that it will also prove to be a useful tool for every day use on site as it gives information which is otherwise only available in the standard reference documents which are not conveniently portable.

The information to make up the aid was derived primarily from the reference manuals SM 0040, SM 0042, SM 0043, SM 0044, SM 0045 and also from consulting the COS listings.

The aid has been specifically put together for the dump analysis work shop to be held in Mendota Heights from August 19-23, 1985, and it is expected that it will be reviewed in the light of comments obtained from users at the workshop. Any other user is welcome to write to the author:

Alistair B Mills
Head of Software Training
Cray Research (UK) Ltd
Gingers Court
36a High Street
Bracknell
Berkshire
England
Tel: + 44 344 422 774

Although considerable effort has been expended to ensure that the aid is accurate and in line with the COS 1.14 release, it must be emphasized that this is not a company supported training or reference document, and that any reliance on the aid is at the discretion of the analyst and that no undertaking is implied to fix errors or maintain this aid for future or earlier releases of the COS operating system.

The aid has pages on each of the major components of COS software

The minor items of COS software such as DEC, MEP, SPM, JCM, FVD are not discussed. TQM and IQM have been omitted from this version, because the author has insufficient experience of these components to provide a summary. Contributions of pages on these topics would be most welcome.

In a document of this sort space is clearly at a premium, and therefore to keep it short, many documentation features are used without explanation, as the experienced COS analyst should recognize them immediately. However, as the aid will probably be used by less experienced COS analysts, where space permits, some key facts about parts of the operating system have been stated.

# System Task Data

| Task | ID | XA | Pr | Description |
|------|----|----|----|-------------|
| Z | 00 | 4740 | 77 | Startup task |
| SCP | 01 | 4760 | 07 | Station call processor |
| EXP (UEP) | 02 | 5000 | 11 | User exchange processor |
| PDM | 03 | 5020 | 14 | Permanent dataset manager |
| DEC | 04 | 5040 | 20 | Disk error corrector |
| DQM | 05 | 5060 | 02 | Disk queue manager |
| LOG (MSG) | 06 | 5100 | 05 | Log manager |
| MEP | 07 | 5120 | 10 | Exec message processor |
| SPM | 10 | 5140 | 24 | System performance monitor |
| JSH | 11 | 5160 | 13 | User task scheduler (Job) |
| JCM | 12 | 5200 | 12 | Job class manager |
| TQM | 13 | 5220 | 03 | Tape queue manager |
| STG | 14 | 5240 | 06 | Stager |
| FVD | 15 | 5260 | 15 | Flush volatile device |
| IQM | 16 | 5300 | 04 | ISP queue manager |

System tasks are interruptable programs which have higher priority than user tasks and which communicate with EXEC via EXEC requests. Their function is to provide management services to EXEC, while allowing EXEC to service interrupts. The priority order indicates which can pre-empt others following an exchange. This order inhibits user service requests from pre-empting physical I/O management for example.

DQM > TQM > IQM > LOG > STG > SCP > MEP > EXP > JCM > JSH > PDM > DEC > SPM > Z

| Task | Req. to task | Rep. to task | Comment |
|------|--------------|--------------|---------|
| Z | 00 | 01 | Only EXEC starts Z |
| | | | Z does one thing at a time |
| SCP | 30 | 30 | Number of stations |
| EXP | 00 | NE@JXT+20 | Only EXEC starts EXP |
| | | | Can handle 1 request/job+ |
| PDM | NE@JXT+20 | 01 | Can handle 1 request/job+ |
| | | | PDM does one thing at a time |
| DEC | NE@EQT | NE@EQT | Can handle errors on all equip. |
| DQM | NE@EQT | NE@EQT | Can handle a request for all equip |
| MSG | NE@JXT | NE@JXT | Can handle 1 request/job |
| MEP | 00 | 24 | Only EXEC starts MEP |
| SPM | 00 | 14 | Only EXEC starts SPM |
| JSH | 40 | NE@JXT+5 | Can handle 1 reply/job+ |
| JCM | 12 | 05 | Somewhat arbitrary |
| TQM | NE@TDT+5 | NE@TDT+5 | Can handle 1 request/tape+ |
| STG | 40 | 40 | See SCP |
| FVD | 02 | 02 | Number of volatile devices |
| IQM | 40 | 40 | See SCP |

Each system task can accept a maximum number of requests queued to it concurrently, or put a maximum number of replies concurrently. Most tasks are apparently multi-threaded and can start another request, while waiting for another one to complete. Others can only do one thing at a time (Z, PDM). Some do not accept requests from other tasks and are started by EXEC (EXP, MEP, SPM).

# STP data-1

System tasks have a large common data area before their code. This data is generally accessible to all tasks although in general although many task may read the data, only one is reponsible for the altering of the data. Some of the data indeed is read only, and if changed at assembly time, allows the COS analyst to change system defaults without having to change executable code.

If STP causes a COS hang (as is often the case), an analysis of this data area will normally lead the analyst to the code causing the problem. It is essential however that the data structures be examined together with the data structures in EXEC. Only by considerable study of dumps of COS data structures, can the analyst become familiar with the normal state of these data structures, and hence be able to recognize abnormal situations.

In the list which follows, the names of the major tables in the STP data area are given in alphabetical order. However the tasks using the data structures is also given.

The most significant of the data structures in determining problems is often the most complicated structures ie the memory pools, the SDT or the DSC/DXT. Note that the DSC/DXT are disk resident data structures and that there are no supported tools for dumping these structures. The author can provide an unsupported tool (FDSC). If the problem is user related, or EXP related, a dump of the user JTA is almost always required.

The most useful data in building up a mental picture of activity in the system immediately prior to the crash are the traces. The inter task communication trace (a 1.14 feature) enables the analyst to quickly build up a picture of STP activity. This feature is not documented in the reference documentation associated with the 1.14 release but there are notes available from software training on this topic. The DQM trace, and the TQM trace may also be useful, if the analyst has determined that the problem lies in one of these tasks.

The first data structure to look at following the EXEC stop buffer is the STP STOP buffer. If STP has stopped, this will point you at the code in the task or common subroutine being executed at the time of the crash. Locating this piece of code in the listing is a simple and sometimes sufficient measure to solving the problem. Unfortunately this invariably points to a further data abnormality which has to be explained by further research.

| Block | Function | Task |
|-------|----------|------|
| AUT | Active user table (interactive use) | SCP, EXP, Z |
| CALL | EXP call table | EXP (read only) |
| CNT | Configuration table | Z |
| CSD | Job class structure definition | JCM, JSH |
| DAT | STP dataset allocation table | EXP, PDM, SCP, DQM |
| DCT | Device channel table | Z |
| DRT | Device reservation table | DQM |
| DXI | Dataset extension table map | PDM |
| EQP | EXEC reply packet table for STP | DQM, EXEC |
| EQT | Equipment table | DQM |