DOMAIN ENGINEERING HANDBOOK

Order No. 002398

Revision 04

Software Release 9.5

THIS DOCUMENT CONTAINS CONFIDENTIAL AND PROPRIETARY INFORMATION WHICH IS THE SOLE PROPERTY OF APOLLO COMPUTER INC. ITS USE IS RESTRICTED TO EMPLOYEES OF APOLLO COMPUTER INC.

APOLLO COMPUTER INC.
330 Billerica Road
Chelmsford, MA 01824

First printing: September, 1981
Last printing: January, 1987

This document was formatted using the FMT tool
distributed with the Apollo Computer system.

PREFACE

The Apollo DOMAIN Engineering Handbook contains system information for the Apollo DOMAIN nodes and related peripherals. References are made throughout the Handbook to reverse-mapped machines and forward-mapped machines. The reverse-mapped machines include the DN300/320/300, DN400/420/600, DN550/560/570/580, and the DSP80/80A/90. The forward-mapped machines include the DNx60, DN3000, DSP3000, and the DN5xx-T.

Audience

This handbook is for Apollo employees only.

Structure of This Document

This manual contains 12 chapters, two appendices, and an index.

Chapter 1 describes principal control blocks associated with AEGIS internal data structures.

Chapter 2 describes formats of file system data structures.

Chapter 3 describes miscellaneous information useful to those programming in the DOMAIN environment.

Chapter 4 describes system error (status) codes and messages.

Chapter 5 describes system debugging tools, including the mnemonic debugger, extensions to the HLL debugger and system dumps.

Chapter 6 describes characteristics of peripheral I/O devices, including the format of control registers and I/O commands.

Chapter 7 describes the hardware architecture of the DN300/320/330 nodes, including the basic processor, Memory Management Unit (MMU), display hardware, FPU, ring, and serial I/O.

Chapter 8 describes the hardware architecture of the DN400/420/600 nodes, including the basic processor, Memory Management Unit (MMU), display hardware, PEB, ring/disk, and serial I/O.

Chapter 9 describes the hardware architecture of the DN460/660 and DSP160 nodes, including the basic processor, Memory Management Unit (MMU), display hardware, ring/disk, and serial I/O.

Chapter 10 describes the hardware architecture of the DN550/560/570/580 nodes, including the basic processor, Memory Management Unit (MMU), display hardware, FPU, ring, serial I/O, MULTIBUS, and VME.

Chapter 11 describes the hardware architecture of the DSP80/80A/90 nodes, including the basic processor, Memory Management Unit (MMU), ring, serial I/O, and MULTIBUS.

Chapter 12 describes the hardware architecture of the DN3000 nodes, including the basic processor, Memory Management Unit (MMU), display hardware, ring/disk, and serial I/O.

Appendix A describes the ASCII character set.

Appendix B is a powers of 2 table.

Related Documents

For hardware architecture information on the DN5xx-T nodes, refer to the DN570-T/DN580-T Workstations and DSP500-T Server Hardware Architecture Handbook (009490).

For hardware architecture information on the DN3000 and DSP3000 nodes, refer to the DOMAIN Series 3000 Hardware Architecture Handbook (007861).

For information about system calls and the program environment, see the Programming With General System Calls (005506) and Progamming With System Calls for Interprocess Communication (005696).

For information about peripheral driver routines, see Writing Device Drivers with GPIO, (000959).

For Shell and DM commands see, DOMAIN System Command Reference, (002547).

Conventions

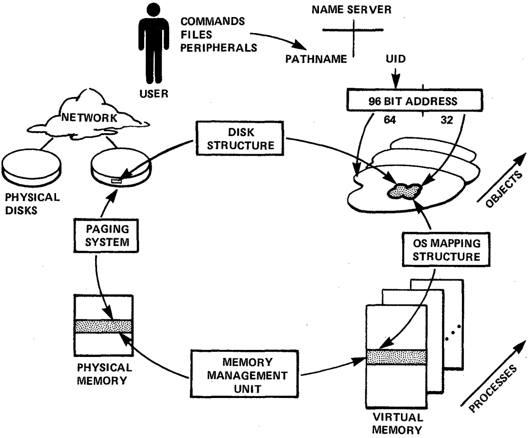
Symbol	Meaning
+00	Offset, in hexadecimal
-	Unused bit
- - - -	Range of unused bits
....	Continuation
..ltame	Name for this field
N = 1 =>	If bit N is set, then ...
N..N	All bits labeled N
=	Equal to
=>	Implies
-->	Pointer to
->	Pointer to
^	Pointer to
set of 0..31	32 bits
1..32	32 values; 5 bits
<a>	Variable
[pa | va]	Physical address | Virtual address
UPPERCASE	Uppercase words => literal commands or keywords
lowercase	Lowercase words => command values that you must supply
[ ]	Square brackets enclose optional items in formats and commands
{ }	Braces enclose a list from which you must choose an item in formats and command descriptions.
|	A vertical bar separates items in a list of choices
< >	Angle brackets enclose the name of a key on the keyboard

TABLE OF CONTENTS

CHAPTER 1

AEGIS

AEGIS SYSTEM RELATIONSHIPS



ACTIVE OBJECT TABLE (AOT) AND ACTIVE SEGMENT TABLE (AST)

The AOT/AST is a wired system-wide table that describes the current state of active objects and their associated active segments. Each ASTE has enough information to identify the file segment and, through the associated AST_PMAPS, the location of each page of the segment. All virtual segments mapped to the same file segment will use the same ASTE entry. The AOTE acts as a cache of the VTOC entry. The table is dynamically allocated at startup based on physical memory size.

ACTIVE OBJECT TABLE

(type "aote_t" in vm.ins.pas)

```
     +-----------------------------------+
+00  |                                   |
     /     CACHED VTOCE HEADER           /  .vtoce_hdr
     |                                   |
     +-----------------------------------+
+40  |          VTOCE POINTER            |  .vtoce_addr
     +-----------------+-----------------+
+44  |   FRONT LINK    |    BACK LINK    |  .flink,
     +-----------------+-----------------+    .blink
+48  |LINK TO FIRST AST|NUM SEGS. ACTIVE |  .aste_link,
     +-----------------+--------+--------+    .nsa
+4C  |AOTE EVENT NUMBER|VIUGXXXX|DEEEEEEE|  .event
+50  +-----------------+--------+--------+
```

```
V    - vtoce header needs updating (.vhdr_mod)
I    - aote in transition (.in_trans)
U    - aote usage bit (.used)
G    - global transparent mod switch (.gtms)
XXXX - volume index into dvt (.volx)
D    - dtm must be updated (.dtm_fl)
E..E - aote-hold-count (.ehcnt)
```

ACTIVE SEGMENT TABLE (AST)

AST: ARRAY[1..N] OF ASTE_T
N is dependent on physical memory size

(type "aste_t" in vm.ins.pas)

```
     +-----------------+-----------------+
 +00 |BACK LINK TO AOT |FILE SEGMENT NUM.|  .aote_link,
     +-----------------+-----------------+    .fsegno
 +04 |   FRONT LINK    |    BACK LINK    |  .flink,
     +-----------------+-----------------+    .blink
 +08 |         POINTER TO FILE MAP       |  .fm_addr
     +-----------------+--------+--------+
 +0C | ASTE EVENT NUM  |IUFRVVVV|PCCCCCCC|  .event
     +-----------------+--------+--------+
 +10 |QQQQQQQQ|SSNNNNNN|
 +12 +-----------------+
```

Forward mapped machines also include:

```
     +-----------------------------------+
 +12 | ASTE TO MSTE BACK THREAD CHAIN    |  .mst_thread
     +-----------------------------------+
 +16 |       PHYSICAL ADDR OF PMAP       |  .pmap_phadd
 +1A +-----------------------------------+
```

```
I    - aste in transition (.in_trans)
U    - aste usage bit (.used)
F    - file-map modified (.fm_mod_
R    - true if object is remote (.remote)
VVVV - index in dvt (.volx)
P    - padding (.pad1)
C..C - aste-hold-count (.ehcnt)
Q..Q - padding (.pad2)
SS   - padding (.pad3)
N..N - number of pages resident (.npr)
```

ACTIVE OBJECT TABLE HEADER

```
        31               16 15              0
       +-----------------+-----------------+
   +00 |                 |                 | .hashtb[0..60]
       +                 +                 +       or
       |           HASH TABLE              |       [0..250]
       /                                   /        for forward
       |         START POINTERS            |       mapped
       +                 +                 +
       |                 |                 |
       +-----------------+-----------------+
   +7A |       AOTE   FREE POINTER         | .free_aote_ptr
       +-----------------+-----------------+
   +7E |  AOTE FREE LIST |                   .lru_aote
       +-----------------+-----------------+
   +80 |          ASTE FREE POINTER        | .free_aste_ptr
       +-----------------+-----------------+
   +84 |  ASTE FREE LIST |-------------DDDD| .lru_aste,
       +-----------------+-----------------+    .dm_req
   +88 |        PROCESSES PER VOLUME       | .vm_cnt[1..4]
       +-----------------------------------+
   +90 |        AOT SEQUENCE NUMBER        | .aot_seq_num
       +-----------------------------------+
   +94 |        AST SEQUENCE NUMBER        | .ast_seq_num
       +-----------------------------------+
   +98 |         NUMBER OF DISMOUNTS       + .dism_seqn
       +-----------------------------------+
   +9C |     EVENT COUNT FOR DISMOUNTER    | .vm_ec
       +-----------------------------------+
   +A0 |          GRACE RPLC PTR           | .lrug
   +A2 +-----------------------------------+

      DDDD - Dismount request flags (.dm_req)
             (bit 16 = volume 1)
```

AST PAGE MAPS - REVERSE MAPPED

```
AST_PMAPS: ARRAY[1..N] OF PMAP_T
PAGE MAP ENTRY (PMAPE):
(type "pmape_t" in vm.ins.pas)

If page is resident:
     +----------------+-----------------+
 +00 |WWTRN-----HHHHHH|PHYS PAGE NUMBER | R=1
     +----------------------------------+

If page is NOT resident:
     +----------+-----------------------+
 +00 |WWTRN-----|   FULL DISK ADDRESS   | R=0
     +----------+-----------------------+
WW   - Wired count (.wired)
T    - Page is in transition (.in_trans)
R    - Page is resident in memory (.resident)
N    - Page is null (no copy on disk) (.null)
```

AST PAGE MAPS - FORWARD MAPPED

```
AST_PMAPS: ARRAY[1..N] OF PMAP_T

For DNx60, DN5xx-T if page is resident:
     +----------------+-----------------+
 +00 |VUMIXTWWRNHHHHHH| PHYS PAGE NUM   | R=1
     +----------------+-----------------+

For DNx60, DN5xx-T if page is NOT resident:
     +----------+-----------------------+
 +00 |VUMIXTWWRN|   FULL DISK ADDRESS   | R=0
     +----------+-----------------------+
V    - Entry is valid (.valid)
U    - Page has been used, set by Hardware (.used)
M    - Page has been modified, set by hardware (.pmod)
I    - Indirect entry (.indirect)
X    - Page may be written (.write)
H..H - Disk address high bits
Others as above.

For machines with PMMU (Series 3000) if the page is resident:
     +--------+----------------+----------+
 +00 |TNRHHHHH| PHYS PAGE NUM  |WW---MUPIV| R=1
     +--------+----------------+----------+
For machines with PMMU if the page is not resident:
     +---+-----------------------+--------+
 +00 |TNR|   FULL DISK ADDRESS   |---MUPIV| R=0
     +---+-----------------------+--------+
P    - Write protect (.write_prot)
Others as above.
```

In forward mapped systems, these are the page tables used by the address translation hardware. AST_PMAPS are dynamically allocated one-for-one with ASTEs.

CLOCK

(type "clock_t" in base.ins.pas)

```
|<- - - - - - -.high- - - - - ->|<- -.low - ->|

 47           33 32           16 15          0
+--------------+--------------+--------------+
|              |              |              |
+--------------+--------------+--------------+

|<- .high16- ->|<- - - - - .low32 - - - - ->|
```

Low-order bit represents four microseconds.

Clock.high is type clockh_t (base.ins.pas);
low-order bit represents approx. four milliseconds.

Hardware clock is only 16 bits. The upper 32 bits are maintained by clock interrupt routine. There are three hardware clocks (not counting calendar clock):

```
Clock 1:   4 microseconds (real-time clock)
Clock 2:   8 microseconds (process timer)
Clock 3: 128 microseconds (real-time intervals)
```

DISK CONTROLLER TABLE ENTRY

(type dcte_t in io.ins.pas)

```
      31                            0
     +--------------+--------------+
+00  |CONTROLLER TYP| CONTROLLER # | .ctype, .cnum
     +--------------+--------------+
+04  |      CONTROLLER STATUS      | .cstatus
     +-----------------------------+
+08  |        LOCK NUMBER          | .lock_no
     +-----------------------------+
+0A  |    BLOCK HEADER POINTER     | .blk_hdr_ptr
     +-----------------------------+
+0E  |      BLOCK HEADER PAGE      | .blk_hdr_pa
     +-----------------------------+
+12  |  I/O REGISTER PAGE POINTER  | .csrs_ptr
     +-----------------------------+
+22  | PAGE ZERO INTERRUPT VECTOR  | .vector_ptr
     +-----------------------------+
+26  |   ACTUAL INTERRUPT ENTRY    | .int_entry
     +-----------------------------+
+2A  | DEV DEPENDENT INTERRUPT RTN | .int_routine
     +-----------------------------+
+2E  |         DISK DINIT          | .disk_dinit
     +-----------------------------+
+32  |        DISK I/O ROUTINE     | .disk_do_io
     +-----------------------------+
+36  |      DISK ERROR RECOVERY    | .disk_error_que
     +--------------+--------------+
+3A  |NCDL----------| UNIT NUMBER  | .dflags, .d_unit_irq
     +--------------+--------------+
+3E  |    PYSICAL DVTE INDEXES     | .pdvte_index
+44  +-----------------------------+
```

DISK FLAGS:
   N - 1 = no headers on device
   C - 1 = do checksumming for this controller
   D - 1 = driver supports multi-read requests (temp)
   L - 1 = liberty (smd) type winchester

DISK VOLUME TABLE ENTRY

```
        (type dvte_t in disk.pvt.pas)

             15                           0
            +------------------------------+
        +00 |            STATE             | .state
            +------------------------------+
        +02 |       BLOCKS PRE VOLUME      | .blocks_per_vol
            +------------------------------+
        +06 |       OWNER PROCESS          | .owner_proc
            +------------------------------+
        +08 |         BASE DADDR           | .lv_base
            +------------------------------+
        +0C |     UID OF PV OR LV          | .uid
            +------------------------------+
        +14 |     PTR TO DEVICE CTR TABLE  | .dcte
            +------------------------------+
        +18 |        DVTE INDEX            | .pdvtex
            +------------------------------+
        +1A |     DISK UNIT NUMBER         | .unit
            +------------------------------+
        +1C |        DISK TYPE             | .dtype
            +------------------------------+
        +1E |       BLOCKS PER TRACK       | .blocks_per_track
            +------------------------------+
        +20 |     TRACKS PER CYLINDER      | .tracks_per_cyl
            +--------------+---------------+
        +22 | DISK FACTOR  |            UNW| .bat_step, .flags
        +26 +--------------+---------------+

        state: 0 - free                   flags: U - use_caller_blkhdr
               1 - being_mounted                 N - no_crc_retry
               2 - assigned                      W - write_protect
               3 - mounted
```

EVENT COUNT

```
        (type eventcount_t of base.ins.pas)

           31                   0
           +--------------------+
        +0 |  CURRENT EC VALUE  | .value
           +--------------------+
        +4 |    NEXT PROCESS    | .nnext
           +--------------------+
        +8 | PREVIOUS PROCESS   | .nprev
           +--------------------+
```

FAULT DIAGNOSTIC RECORD

```
        (type "fault_$diag_t" in fault.ins.pas)

             15               0
            +-----------------+
        +00 |1101111111011111|  .pattern  (#DFDF)
            +-----------------+
        +02 |      STATUS     |  .status
            +                 +
            |       WORD      |
            +-----------------+
        +06 |                 |  .registers
            /    REGISTERS    /
            |                 |
            +-----------------+
        +46 |- - - - - -RNFFF|  .bus_info
            +-----------------+
        +48 | ACCESS ADDRESS |  .access_address
            +-----------------+
        +4C | INTRUCTION REG |  .inst_register
            +-----------------+
        +4E |      FLAGS      |
            +-----------------+

        R   - Read operation (.write_op)
        N   - Not instruction reference (.not_inst)
        FFF - Instruction function code (.function_code):
              001  User data
              010  User program
              101  Supervisor data
              110  Supervisor program
              111  Interrupt acknowledge
```

MAPPED SEGMENT TABLE (MST)

The MST is a wired array indexed by asid and va. Each mste describes one segment of virtual memory.

```
MST: array[0..(asid_tn*mst_pages_tn)-1] of pinteger
     mst_entry_free = 0
```

mst_pages_tn is the number of mst entries needed to describe one asid.

```
mstes_per_page = page_size div sizeof mste_t
mst_pages_tn = mst_seg_tn div mstes_per_page
```

MAPPED SEGMENT TABLE ENTRY (MSTE)

(type "mste_t" in vm.ins.pas)

```
     31             16 15             0
    +----------------+----------------+
+00 |           UID OF FILE           |  .uid
    +----------------+----------------+
+08 | FILE SEGMENT # |EGxxxxxPPPPPPPPP|  .fsegno
    +----------------+----------------+
+0C |   INDEX OF VTOC ENTRY (VTOCX)   |  .locx
    |               OR                |
    |RCCCCCCSQQQQQMMMMMMMMMMMMMMMMMMMM|
+10 +---------------------------------+
```

```
E = 1 => File extension allowed (.ext_ok)
G = 1 => Interrrupt on ref (.guard)
xxxxx = for DNx60   => --PPP
        for DN5xx-T => PPPPP
        for DN3000  => ----P
        for others  => AAAAA
AAAAA - Access (.access):
    00000   Nil (access_$nil)
    00010   Read (access_$r)
    00011   Read-execute (access_$rx)
    00110   Write-read (access_$wr)
    00111   Write-read-execute (access_$wrx)
    10010   Supervisor read (access_$sr)
    10011   Supervisor read-execute (access_$srx)
    10110   Supervisor write-read (access_$swr)
    10111   Supervisor write-read-execute (access_$swrx)
P - Probable ASTEX (.pastex)
R - Object is remote (.mste_remote)
C - Touch ahead count (.mste_touch_cnt)
S - Sequential access
Q - Next sequential page
M - Mste-node
```

MAPPED SEGMENT TABLE ENTRY PAGES:
(defined in /os/kins/mst.pvt.pas) (type "mste_pages" in kins/mst.pvt.pas)

```
MSTE_PAGES : ARRAY[1..(ppn_tn DIV mst_page_percentage] of mst_page_t
             mst_page_t : array[0..mstes_per_page - 1] of mste_t

    For systems with a virtual address space greater than 1Gb:
             ARRAY[1..(mst_sm_config * mst_page_percentage
                    DIV 100 + ((ppn_tn - mst_sm_config) *
                    mst_l_config_percent DIV 100))] of mst_page_t
```

MSTE_PAGES is an array of pages that are wired as needed to support mappings in the mst. When a segment is mapped and the corresponding mst entry = mst_entry_free, a free page is found in the mste_pages array and wired. It's number is put into mst[entry]. The number of pages that can be used is limited at boot time to be some percentage of physical memory. See mst_$init.

MEMORY MAP (MMAP)

For all machine types an MMAPE (mmap entry) is 12 (decimal) bytes in size.

MEMORY MAP ENTRY (MMAPE) (REVERSE MAPPED MACHINES)

```
 (type "mmape" in mmap.pvt.pas)
   15                0
   +----+------------+
+00|IARO|  PTT INDEX |  .pttx
   +----+------------+
+02|PREV LST|CUR LIST|  .prev_lst,
   +--------+--------+   .list
+04|  PPN NEXT PAGE  |  .flink
   +-----------------+
+06|  PPN PREV PAGE  |  .blink
   +-----------------+
+08|  LOW DISK ADDR  |  .daddr_1
   +--+--------------+
+0A|D-| ASTE INDEX   |  .astex
   +--+--------------+

I     = 1 => Entry in use (.inuse)
A     - 0 => Page is wired or in transit (.avail)
R     - 1 => Page mod'ed but dtm shouldn't
             be updated (.rmod)
O     = 1 => Page is on some memory page list (.onlist)
D     = 1 => This page is holding data (as opposed
             to code) (.d_pg)
```

MMAPEs are organized onto a number of lists, each list is described by a list header (type ws_hdr_t in mmap.pvt.pas) the headers are collected into the mmap_$wsl array (type wsl_t in mmap.pvt.pas)

List name	List index	Use
mmap_$xfree	0	Pages not pointed at by a PMAPE ... contain no useful data
mmap_$xhi_pure	1	Pages not in use ... have already been purified
mmap_$xpure	2	Pages not in use ... are pure, don't need to be written
mmap_$ximpure_l	3	Pages not in use ... are impure, need to be written to local disk
mmap_$ximpure_r	4	Pages not in use ... are impure, need to be written to remote disk
mmap_$xos_shared	5	Pages in use by any of the L1 Aegis processes
mmap_$xnet_pager	6	Pages being 'cached' by one of the network paging servers ... replaces the remote paging pool
mmap_$xinit_proc	7	Pages in use by process 1 (DM)

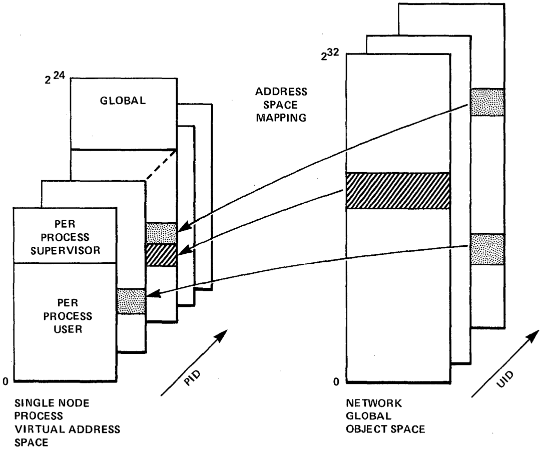
Lists after this hold pages in use by some process, assigned at process creation.

List header = ws_hdr_t =

```
    32             16 15             0
   +----------------+----------------+
+00|IP--------------|  .cur_size     |
   +----------------+----------------+
+04|    UNUSED      |   .flink       |
   +----------------+----------------+
+08|   .max_size    |   UNUSED       |
   +----------------+----------------+
+0C|              .ws_pri            |
   +---------------------------------+
+10|           .last_ws_scan         |
   +----------------+----------------+
+14|  .interr_cnt   |  .prot_size    |
   +----------------+----------------+
```

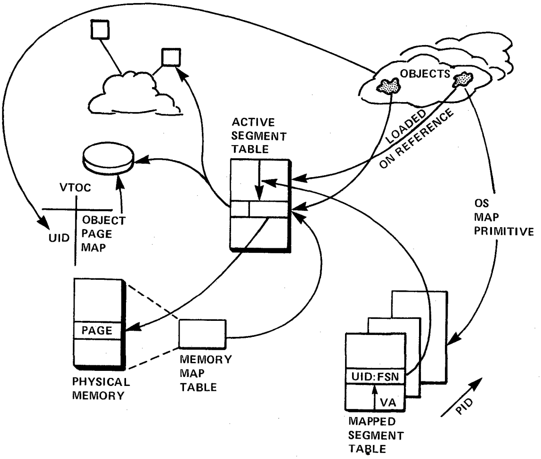
```
I              : 1 => list is in use  (.inuse)
P              : 1 => scan list periodically in real
                      time (.p_scan)
 .cur_size     : # of pages (MMAPEs) on this list
 .flink        : PPN of first page on this list
                      (if .cur_size > 0) ... anchors
                        a doubly threaded list of MMAPEs
 .max_size     : maximum   of  pages that this list can hold
 .ws_pri       : time (time_$clockh) the process 'owning'
                       this last last ran
 .last_ws_scan: time (time_$clockh) this last was last
                scanned
 .interr_cnt   : used in scheduling scans on this list
 .prot_size    : list is always allowed to have this many pgs
                on it (i.e., list size won't be forced below
                this  )
```

OS MAPPING



2^{24}
GLOBAL
PER PROCESS SUPERVISOR
PER PROCESS USER
0
PID
SINGLE NODE PROCESS VIRTUAL ADDRESS SPACE
ADDRESS SPACE MAPPING
2^{32}
0
UID
NETWORK GLOBAL OBJECT SPACE

PAGING SYSTEM



PROCESS CONTROL BLOCK (PCB)

```
        (type "procl_t" in procl.pvt.asm)

             31                           0
            +-----------------------------+
        +00 |      --> NEXT READY PCB     | .nextp
            +-----------------------------+
        +04 |    --> PREVIOUS READY PCB   | .prevp
            +-----------------------------+
        +08 |             d2              | .save_d2
            +-----------------------------+
                          ...
            +-----------------------------+
        +1C |             d7              | .save_d7
            +-----------------------------+
        +20 |             a2              | .save_a2
            +-----------------------------+
                          ...
            +-----------------------------+
        +28 |             a4              | .save_a4
            +-----------------------------+
        +2c |             a5              | .save_db
            +-----------------------------+
        +30 |             a6              | .save_sb
            +-----------------------------+
        +34 |             a7              | .save_sp
            +-----------------------------+
        +38 |            USP              | .save_usp
            +-----------------------------+
        +3C |CLOCKH_T AT START OF EC_$WAIT| .wait_start
            +-----------------------------+
        +40 |       RESOURCE LOCK WORD    | .rlock
            +--------------+--------------+
        +44 |     PID      |     ASID     | .mypid, .asid
            +--------------+--------------+
        +48 |T.S. REM.*8MIC|FLPT.PROC.TYPE| .vtimer .fpptype
            +--------------+--------------+
        +4C |           CPU TIME          | .cpu_total
            +--------------+--------------+
        +50 |   CPU TIME   |  priority    |         .priority
            +--------------+--------------+
        +54 | STATE: TBPSW |  PRI Minimum | .state, .pri_min
            +--------------+--------------+
        +58 | PRI Maximum  |  Inhibit crt | .pri_max .inh_count
            +--------------+--------------+
        +58 | bus status   |  pad         | .sw_bsr  .pcb_pad
            +--------------+--------------+

        State: T - Time slice end with rlock <> [] (tse_onb)
               B - Bound (bound)
               P - Suspension pending (susp_pending)
               S - Suspended (suspended)
               W - Waiting (waiting)
```

```
procl_$state_t = SET OF (waiting, suspended, susp_pending,
                         bound, tse_onb);
```

(State is type "procl_$state_t" in procl.ins.pas)

PROCESSES

PID	ASID	Description
1	1	Initial user process and DM
2	0	Null process
3	0	Wired DXM process
4	0	Page purifier local
5	0	Page purifier remote
6	0	Unwired DXM process
7	0	Network-receive server
8	0	Network-paging server
9	0	Network-request server
10	0	Terminal helper
11	2	First user process
12	3	Second user process
13	4	.
14	5	.
15	6	.
16	7	.
64	56	Last user process

Note: Processes with an ASID of 0 run entirely in global space.

Note: If the debugger's Lights command has been given, there will be a Lights process with PID 11 or greater with a zero ASID.

Note: If the netsvc -servers option has been used there may be more net-paging and net-request servers.

Note: There can also be IIC guardian Process Ethernet router for Bridge nodes.

RING PACKET FORMAT

Message Header

```
         15               0
        +-----------------+
  +00   |   TO ADDRESS    |
        |                 |
        +-----------------+
  +04   |   TYPE FIELD    |
        +--------+--------+
  +06   |   0s   |EARLYACK|
        +--------+--------+
  +08   |  FROM ADDRESS   |
        |                 |
        +-----------------+
  +0C   |     HEADER      |
        |      DATA       |
        /                 /

        /(0 - 1024 BYTES) /
        |                 |
  +40C  +-----------------+
```

Type Field

```
   15    14    13    12    11    10    9     8
+-----+-----+-----+-----+-----+-----+-----+-----+
| BST |  T  |  T  |  T  |  T  |  T  |  T  |  T  |
+-----+-----+-----+-----+-----+-----+-----+-----+

   7     6     5     4     3     2     1     0
+-----+-----+-----+-----+-----+-----+-----+-----+
|  T  |  T  |  T  |  T  |  T  |  T  |  T  |  T  |
+-----+-----+-----+-----+-----+-----+-----+-----+
```

Bit 15 - BST (Broadcast)- This bit is set in a packet intended to be broadcast to all receivers. If it is set, the To Address field is ignored.

Bit 14 thru 0 - T (Type)- This field determines whether a packet is to be received. Each 1 bit in the received Type field is compared to the corresponding bit in the controller Type register. If any bit selected in the Type register is a 1, the message is received. If all bits selected in the Type register are 0, the message is ignored. The APOLLO I only implements Bits 14 through 8 only; bits 7 through 0 never match.

Early Acknowledge Field

The Early Acknowledge field is inserted by the transmitter and modified by the receivers. For the purposes of the CRC calculation the EARLY ACK field is treated like a byte of zeros. This allows receivers to modify the ACK field without having to recompute the CRC checksum.

```
   7     6     5     4     3     2     1     0
+-----+-----+-----+-----+-----+-----+-----+------+
|  0  |  X  |  X  |  0  | ICP |  X  | PAR |  0   |
+-----+-----+-----+-----+-----+-----+-----+------+
```

Bit 7-0 - This bit is inserted to prevent the remaining bits in the late-acknowledge byte from being modified by the bit-stuffing protocol.

Bit 6-X - Don't care - This bit is not used.

Bit 5-X - Don't care - This bit is not used.

Bit 4-0 - This bit is inserted to prevent the remaining bits in the late-acknowledge byte from being modified by the bit-stuffing protocol.

Bit 3-ICP - Intend to Copy - This bit is set by an addressed receiver if it was set up to copy a message and the type field matched. A NAK (negative acknowledge) condition is indicated when no receiver sets this bit.

Bit 2-X - Don't care - This bit is not used.

Bit 1-PAR - Parity - This parity bit is set so that there are an odd number of bits in the late acknowledge byte.

Bit 0-0 - This bit is inserted to prevent the remaining bits in the late-acknowledge byte from being modified by the bit-stuffing protocol.

Message Data

```
 <----------- 8 bits--------------->
+-----------------------------------+
|                                   |
|    Message Data 0 - 1024 bytes    |
|                                   |
+-----------------------------------+
```

The Message Data field varies in length from 0 to 1024 bytes but must always be an even number of bytes(on the Apollo I the controller always uses word DMA). The contents of Message Data field is determined by the software; it is transmitted verbatim.

RESOURCE LOCK

```
Defined in base.ins.pas
network_$serv_lock,{ 00 1 }
mt_$lock,          { 01 2 }
xpd_$lock,         { 02 4 }
term_$lock,        { 03 8 }
proc2_$lock,       { 04 10 }
file_$lock_lock,   { 05 20 }
ec2_$lock,         { 06 40 }
smd_$respond_lock, { 07 80 }
smd_$request_lock, { 08 100 }
pbv_$lock,         { 09 200 }
acl_$lock,         { 10 400 }
proc1_$create_lock,{ 11 800 }
onb_$lock,         { 12 1000 faulted to CPUB }
bok_$lock,         { 13 2000 runnable on B }
disk_$mnt_lock,    { 14 4000 }
vtoc_$lock,        { 15 8000 }
bat_$lock,         { 16 10000 }
ast_$lock,         { 17 20000 }
pag_$lock,         { 18 40000 }
sm_$lock,          { 19 80000 }
flp_$lock,         { 20 100000 }
win_$lock,         { 21 200000 }
ring_$xmit_lock,   { 22 400000 }
ml_$free7,         { 23 800000 }
                   { the next two locks are highest}
time_$proc_lock,   { 24 1000000 clock process only }
time_$lock         { 25 2000000 clock process data base }
```

SYSTEM BOOT FILES

```
          FILES REQUIRED DURING BOOT

REQUESTING AGENT               FILE

prom       /sysboot (records 2-b on track 0)

(if DNx60  /saun/wcs.uc      (microcode file)
or DN5xx-T)      dcode.uc    (instr. decode ram contents)
                 spad.uc(DNx60 only)(scratchpad constants, temps)
                 uload       (program to load the above)

sysboot    /saun/aegis       (aegis load file)
           /saun/salvol      (only if salvage required)

aegis      [os paging file]  (uncatalogued)
           //                (uids found and saved by name_$init)
           /
           /com
           /sys/node_data
           /sys/peb_microcode or peb2_microcode(1)
           `node_data/boot_shell(2)   (mapped by proc2_$init)

shell      /sys/apollo_logo(3)
           `node_data/startup_shell(3)   (cmd file to override dflts)
           /sys/env         (shell tells him what to run)

env        /lib/?*
           /sys/dm/dm       "go" command or normal boot      -or-
           /sys/boot        "sh" or boot from sio line       -or-
           /sys/spm/spm     "spm" or normal boot on server node

dm         `node_data/dev/sio
           /sys/dm/fonts
           `node_data/startup[.191,.color,.1280color,.1280bw](3)
           /sys/boot

boot       /registry/registry(4)    (+ppo,account files pointed to)
                     local_registry
                     local_site/?*
           /com/sh
```

NOTES:
 (1) PEB is disabled if microcode file not found.
 (2) If booted from cartridge tape, the tape is first searched for BSCOM/RBAK_SHELL.
 (3) Optional -- system will manage without it.
 (4) If no registries are available, you can login only as USER.NONE.NONE.

SYSTEM DIRECTORIES

/bscom	Boot shell command directory
/cc	C compiler (optional)
/com	Shell commands directory
/core	Core graphics (optional)
/dev	Peripheral I/O device definitions
/install	Installation scripts directory
/lib	System libraries directory
/sau1	Stand-alone utilities for DN4xx/600
/sau2	Stand-alone utilities for DN300/320/330
/sau3	Stand-alone utilities for DSP80/80A/90
/sau4	Stand-alone utilities for DNx60,DSP160
/sau5	Stand-alone utilities for DN550/560/570/580
/sau6	Stand-alone utilities for DN5xx-T
/sau8	Stand-alone utilities for DN3000,DSP3000
/sys	
/sys/alarm	Alarm server
/sys/boot	Boot shell file
/sys/dm	Display manager programs
/sys/dm/fonts	Display manager font definitions
/sys/env	Process 1 initializer
/sys/hasp	Hasp support
/sys/help	Help text files
/sys/ins	User insert files
/sys/mbx	Mailbox helper
/sys/net	Netman (diskless node support)
/sys/node_data[.nn]	Per node read/write data files
/sys/sf	Store and foward files
/sys/siologin	Sioline login support
/sys/source	Selected source files
/sys/spm	Server process manager
/sys/stream_$sfcbs	Stream mgr control blocks
/sys/subsys	Protected system support
/sys/sysdev	Stream device files
/sysboot	System boot file
/systest	On-line system tests directory

TRAP CODES

0	SVC -- 0 arguments
1	SVC -- 1 arguments
2	SVC -- 2 arguments
3	SVC -- 3 arguments
4	SVC -- 4 arguments
5	SVC -- any number of arguments
6	SVC -- from GPIO interrupt routines
7	Undefined (reflected to user space)
8	Undefined (reflected to user space)
9	Undefined (reflected to user space)
A	Undefined (reflected to user space)
B	Undefined (reflected to user space)
C	Undefined (reflected to user space)
D	Undefined (reflected to user space)
E	Software-generated fault (pfm_$error_trap)
F	Undefined (traps to PROM debugger)

CHAPTER 2

FILE SYSTEM

ACLS STRUCTURE

ACL Header Record

```
(type acl_$hdr in acls.pvt.pas)
     15              0
    +------------------+
+0  | VERSION OF ACL   | .version
    +------------------+
+02 | TYPE OF OBJECT   | .atype
    \ FOR WHICH THIS   \
    | CAN BE ACL       |
    +------------------+
+OA |                  | .def_nil
    |DEFAULT NODE LIST|
    | FOR NEW ENTRIES  |
    +------------------+
+OE |NUMBER OF ENTRIES| .nents
    +------------------+
+10 |NO. OF NODE LISTS| .nlists
    +------------------+
+12 |    NOT USED      | .spare
    +------------------+
+14 |                  | .extra
    \    NOT USED      \
    |                  |
+34 +------------------+
```

ACL Record

```
(type acl_$rep in acls.pvt.pas)
     15              0
    +-----------------+
+0  |                 | .acl_$hdr
    \    HEADER       \
    |                 |
    +-----------------+
+34 |   ENTRIES       | .entries
    |    array        |
    | [1.acl_$entmax]|
    | of acl_$entry   |
    +-----------------+
```

ACL Entry

(type acl_$entry in acl.ins.pas)

```
        64                              0
       +--------------------------------+
  +00  |          PERSON UID            |  .pers in acl_$sid
       +--------------------------------+
  +08  |          PROJECT UID           |  .proj in acl_$sid
       +--------------------------------+
  +10  |        ORGANIZATION UID        |  .org in acl_$sid
       +--------------------------------+
  +18  |          SUBSYSTEM UID         |  .subs in acl_$sid
       +----------------+---------------+
  +20  |    NODE ID     |    EXP DATE   |  .node_t, .exp_date
       +----------------+---------------+
  +28  |  ACL RIGHTS    |                  .rights
       +----------------+
```

BLOCK AVAILABILITY TABLE (BAT)

(type "bat_blk" in vol.ins.pas)

type bat_blk_t= array[0..255] of bat_lword_t

```
          31                            0
         +----------------------------+
   +00   |        BAT WORD [0]        |
         +----------------------------+
   +04   |        BAT WORD [1]        |
         +----------------------------+
         |                            |
         /                            /
         |                            |
         +----------------------------+
  +3FC   |        BAT WORD [255]      |
  +400   +----------------------------+
```

First BAT block pointed to by BAT header in logical volume label (pv_label). BAT resides in contiguous records.

Bit 0, BAT WORD[0] corresponds to first block (bat_hdr.base_add) in the logical volume. BAT bit = 1 if block is available.

BLOCK AVAILABILITY TABLE HEADER

(type "bat_hdr_t" in vol.ins.pas)

```
      31                            0
     +-----------------------------+
 +2C |NUMBER OF BLOCKS REPRESENTED|   .n_blk
     +-----------------------------+
 +30 |   NUMBER OF FREE BLOCKS     |   .n_free
     +-----------------------------+
 +34 |  DADDR OF FIRST BAT BLOCK   |   .addr
     +-----------------------------+
 +38 |BLK # REP BY 1st BIT IN BAT |   .base_add
     +-------------+---------------+
 +3C |VCB----------|     BAT_      |
     +-------------+---------------+
 +40 |    STEP     |               |   .bat_step
     +-------------+               |
     |                             |
     /           UNUSED            /
     |                             |
 +4C +-----------------------------+
```

```
V - Volume trouble, set by OS if volume needs
    salvaging, cleared by SALVOL. (.vol_trouble)
C - Volume CHUVOLed (.vol_chuvoled)
B - Volume being CHUVOLed (.vol_being_chuvoled)
```

BAT header lives in logical volume label.
Offsets given are from start of label.

DIRECTORY STRUCTURE

Directory Overview

(type dir_t in name.pvt.pas)

```
+------------------------+
|         HEADER         |     Directory configuration
+------------------------+       information
|       LINEAR LIST      |     Sequentially used directory
+------------------------+       entries
|       INFO BLOCK       |     ACL manager's intial ACL
+------------------------+       description block
|       HASH THREADS     |     Pointers to linked lists of
+------------------------+       hashed entries
|         ENTRY          |     Holding blocks for hashed
|         BLOCKS         |       entries and/or link text
+------------------------+
```

Directory Info Block

(type infoblk_hdr_t in name.pvt.pas)

```
     +------------------------+
+00  |  VERSION  |   M B Z    | Info block version number
     +------------------------+
+02  |    INFO BLOCK LENGTH   | Total length of info block
     +------------------------+
+04  |  INFO BLOCK HDR LENGTH | Length of info blk hdr (8)
     +------------------------+
+06  |         M B Z          | Reserved for future use
     +------------------------+
+08  |   DEFAULT ACL UID      | UID of ACL to be applied to
     |   FOR DIRECTORIES      |  directories catalogued in
     +------------------------+  this directory
+0C  |    DEFAULT ACL UID     | UID of ACL to be applied to
     |      FOR FILES         |  files catalogued in this
     +------------------------+  directory
+10  |    24 UNUSED BYTES     | Reserved for future use
     \                        \
+30  +------------------------+
```

Directory Entry

(type dir_entry_t in name.pvt.pas)

```
    +------------------------+
+00 |      ENTRY NAME        | 32 bytes of entry name
    +------------------------+
+20 |         UNUSED         | Reserved
    +------------------------+
+22 |         UNUSED         | Reserved
    +------------------------+
+24 |         UNUSED         | Reserved
    +-----------+------------+
    |           |            | Name length - # of useful
+26 |NAME LENGTH| ENTRY TYPE |    characters in entry name
    |           |            | Entry type - 0 = not in use
    |           |            |              1 = name/UID pair
    +-----------+------------+              3 = name/link-data
    |                        |                  pair
+28 |       4 WORDS OF       | If entry type = 1, UID
    |       ENTRY DATA       | Entry type = 3, =>
    |  (EITHER UID OR LINK   | Link text: link text len,
    |    TEXT DESCRIPTION)   | Blk holds lnk text chrs, 1-144
    |                        | Blk holds lnk text chrs, 145-256
    +------------------------+ Reserved for future use
```

Directory Entry Block

(type entry_block_t in name.pvt.pas)
        total length - 150 bytes

```
    +------------------------+
+00 |   NEXT BLOCK NUMBER    | Forward thread - doubly linked
    +------------------------+   list
+02 |   PREV BLOCK NUMBER    | Backward thread - doubly linked
    +-----------+------------+   list
    |           |            | Use count - # of used entries
+04 | USE COUNT | BLOCK TYPE |   in this block
    |           |            | Blk type -   0 = not in use
    |           |            |  -1 = hash blk with 3 dir entrs
    +-----------+------------+  -3 = link text holding block
+06 |      ENTRY BLOCK       | Either 3 dir entries or
    |         DATA           | Up to 144 chars of link text
    +------------------------+
```

Directory Header

(first part of type "dir_t" in name.pvt.pas)

```
     +-------------------------+
+00  |        VERSION          |version number of this directory (1)
     +-------------------------+
+02  |      HASH VALUE         | # of hash threads used for entry
     +-------------------------+    name hashing
+04  |      LIST SIZE          | # of entries configured into
     +-------------------------+    linear list (18)
+06  |      POOL SIZE          | # of entry blocks in this
     +-------------------------+    directory (429)
+08  |   ENTRIES PER BLOCK     | # of entries that fit in an
     +-------------------------+    entry block (3)
+0A  |   HIGH BLOCK NUMBER     | # of the highest entry block
     +-------------------------+    used so far
+0C  |   FREE BLOCK THREAD     | # of the first block on the free
     +-------------------------+    block list
+0E  |        UNUSED           | Reserved for future use
     +-------------------------+
+10  |        UNUSED           | Reserved for future use
     +-------------------------+
+12  |        UNUSED           | Reserved for future use
     +-------------------------+
+14  |        UNUSED           | Reserved for future use
     +-------------------------+
+16  |     ENTRY COUNT         | # of entries currently catalogued
     +-------------------------+    in this directory
+18  |    MAXIMUM COUNT        | # of entries this directory can
     +-------------------------+     hold (1300)
```

Notes on directories

1. To add an entry to a directory:

   A. Look for an unused entry in the linear list. If you find one, use it and you're done.

   B. Hash the name you want to add:

```
- name is:
     name: array [1..32] of CHAR
- lnth is useful lnth of name
   sum: =0;
   For i : = 1 to lnth DO
      sum : = ord(name[i])+2*sum;
   HASH_VAL : = sum mod HASH_VALUE;
```

   C. Get the hash thread for the specified hash value and call that value the found block.

   D. If the found block number is 0 then we need a new entry block, so:

a) See if there are any blocks threaded through the free block list and if so, take one of those. Otherwise, bump the high block number and use that.

b) Initialize the newly obtained block, add it to the end of the apprpriate hash chain, add the new entry as the first entry in the new entry block and you're done.

E. If there is an unused entry in the found block, use it and you're done.

F. Change the found block value to the number in the current found block's NEXT BLOCK field and go to step D.

2. The searching rule for a directory is:

A. Look in the linear list.

B. Hash the name you're searching for.

C. Follow the hash thread for the specified hash value to the first entry block with that hash synonym.

D. Search all (3) of the entries in the found entry block

E. Follow the "next block number" in the found entry block to get a NEW found entry block. If the next block number is zero, then return NOT FOUND.

F. Go to step D with the newly found block.

DISK BLOCK HEADER

```
(type "blk_hdr_t" in base.ins.pas)

     31             16 15             0
    +-----------------+----------------+
+00 |      UID OF OBJECT TO WHICH      |  .uid
    +                                  +
    |          BLOCK BELONGS           |
    +----------------------------------+
+08 |       PAGE NUMBER IN FILE        |  .page
    +----------------------------------+
+0C |    TIME WRITTEN (clock.high32)   |  .dtm
    +--------+--------+----------------+
+10 | BLKTYP | SYSTYP |                |  .blk_type,
    +--------+--------+                +    .sys_type
+14 |              UNUSED              |
    +                 +----------------+
+18 |                 |    CHECKSUM    |  .chksum
    +-----------------+----------------+
+1C |           DISK ADDRESS           |  .daddr
+20 +----------------------------------+

BLKTYP: 0 - Data block
        1 - Level 1 index block in file map
        2 -   "   2   "     "    "    "   "
        3 -   "   3   "     "    "    "   "
SYSTYP: 0 - File
        1 - Directory
        2 - System directory
```

DISK $ERROR INFO

```
     31                             0
    +---------------------------------+
+00 |          TIME OF R/W            |
    +---------------------------------+
+04 |            DADDR                |
    +----------------+----------------+
+08 |  B_PR_TRK      |   TRK_PR_CYL   |
    +----------------+----------------+
+0C |           REQUESTED             | UID, page,
    |             HEADER              |    daddr
    +---------------------------------+
+2C |             READ                |
    |            HEADER               |
    +----------------+----------------+
+4C |     VOLX       |      PPN       |
    +----------------+----------------+
+50 |           STATUS                |
    +---------------------------------+
```

DISK/VOLUME FORMAT

```
     +=================+
     |                 | BLOCK 00 OF PHYSICAL VOLUME
+----|     PHYSICAL    |  (1 BLOCK)
|+---|   VOLUME LABEL  |
||+--|                 |
|||  +=================+
||+->|                 | BLOCK 00 OF LOGICAL VOLUME (1 BLOCK)
||   |     LOGICAL     |-----+
||   |   VOLUME LABEL  |---+ |
||   |                 |   | |
||   +-----------------+   | |
||   |                 |   | | BLOCK 01 OF LV (10 BLOCKS)
||   |       BOOT      |   | |
||   |       FILE      |   | |
||   |                 |   | |
||   +-----------------+   | |
||   |                 |   | | BLOCK 0B OF LV
||   |       FREE      |   | |
||   /                 /   | |
||   |      BLOCKS     |   | |
||   |                 |   | |
||   +-----------------+   | |
||   |                 |<--+ |
||   |       VTOC      |     |
||   |                 |     |
||   +-----------------+     |
||   |                 |<----+
||   |       BAT       |
||   |                 |
||   +-----------------+
||   |                 |
||   |       FREE      |  NOTE: all disk addresses
||   /                 /  (DADDRs) in a logical volume
||   |      BLOCKS     |  are relative to the start
||   |                 |  of a logical volume.
||   +-----------------+
|+-->|                 | LAST BLOCK OF LOGICAL VOLUME
|    | LOGICAL VOLUME  |
|    |LABEL (2nd COPY) |
|    |                 |
|    +=================+
+--->|  NEXT LOGICAL   |
     |     VOLUME      |
     |       ...       |
```

Badspots may cause the VTOC to be non-contiguous and not adjacent to BAT. There may be dead space between logical volumes. Bad spots may not be added to the VTOC once it has been INVOLed.

FILE MAP

```
              +---------+
         +00  |         |----> DATA BLOCK 1
              +---------+
         +04  |         |----> DATA BLOCK 2
              +---------+
         +08  |         |
              /         /      ...
              |         |
              +---------+
         +7C  |         |----> DATA BLOCK 32
              +---------+
              +---------+       LEVEL  1
         +80  |         |----> +---------+
              +---------+      |         |----> DATA BLOCK 33
         +84  |         |--+   +---------+
         +88  +---------+  |   |         |----> DATA BLOCK 34
          +---|         |  |   +---------+
          |   +---------+  |   |         |
          |                |   |         |
          |                |   +---------+
          |                |   |         |----> DATA BLOCK 288
          |                |   +---------+
          |                |
          |    LEVEL  3    |    LEVEL  2
          +-->+---------+  +-->+---------+
              |         |      |         |
              |  256    |      |  256    |
              |POINTERS |      |POINTERS |
              |   TO    |      |   TO    |
              |LEVEL  2 |      |LEVEL  1 |
              | BLOCKS  |      | BLOCKS  |
              |         |      |         |
              +---------+      +---------+
```

```
Maximum file size = (32+256+256**2+256**3)*1024 bytes
                  = 17,247,300,000 bytes.
```

File map resides in VTOCE and AST entries.

REGISTRY FORMAT

Header Record

(type ppo_$header_t in ppo.ins.pas)

```
      +--------------------------------+
+0    | TRANSACTION UID FOR SALVAGING |  .ppo_$xact_uid
      +---+---+-------+-------+-------+
+8    | C | F |HDR LEN|NUMBER | READ  |
      +---+---+-------+-------+-------+
+10   | WRITE |       | PW LEN|REC LEN|  .ppo_$rec_len
      +-------+-------+---------------+
+18   |   UNUSED      |   UNUSED      |  .ppo_$space,
      +---------------+---------------+  .ppo_$space2
```

```
C = 1 => committed (.ppo_$committed)
F = 1 => local     (.ppo_$local_flag)
NUMBER - number of entries (.ppo_$num_entries)
READ   - oldest software that can read this (.ppo_$r_vers)
WRITE  - oldest software that can write new (.ppo_$w_vers)
PW LEN - minimum password length (.ppo_$min_plen)
```

PPO Record

(type ppo_$record_t in ppo.ins.pas)

```
      +--------------------------------+
+00   |                                |
      +                                +
+08   |          PPO NAME              |  .ppo_$name
      +                                +
+10   |                                |
      +                                +
+18   |                                |
      +---------+----------------------+
+20   | NAMLEN  |       UID ...        |  .ppo_$namlen
      +--------------------------------+  .ppo_$uid
+28   | ...     |----------------------|
      +---------+----------------------+
```

Account Header

(type acct_$header in acct.ins.pas)

```
    +-------------------------------+
+00 | TRANSACTION UID FOR SALVAGING | .acct_$xact_uid
    +---+---+-------+-------+-------+
+08 | C | F |HDR LEN|NUM ENT| READ  |
    +---+---+-------+-------+-------+
+10 | WRITE |-------|PW LEN |REC LEN|
    +-------+-------+-------+-------+
+18 |CLOCKH TIME PER|   UNUSED      | .acct_$exp_period
    +-------------------------------+
```

```
C = 1 => committed (.acct_$committed)
F = 1 => local (.acct_$local_flag)
HDR LEN - header length (.acct_$hdr_len)
NUM ENT - number of entries (.acct_$num_entries)
READ    - oldest software that can read this (.acct_$r_vers)
WRITE   - oldest software that can write new (.acct_$w_vers)
PW LEN  - minimum password length (.acct_$min_plen)
REC LEN - record length (.acct_$rec_len)
```

Account Record

(type acct_$record_t in acct.ins.pas)

```
    +-------------------------------+
+00 |          PERSON UID           | .acct_$pers_uid
    +-------------------------------+
+08 |          PROJECT UID          | .acct_$proj_uid
    +-------------------------------+
+10 |        ORGANIZATION UID       | .acct_$org_uid
    +-------------------------------+
+18 |          ACCT PW              | .acct_$pwd
    +               +---------------+
+20 |               |   EXP DATE    | .acct_$exp_date*
    +---------------+---------------+
+28 |-------| LAST LOGIN    |-------| .acct_$last_login*
    +-------+---------------+-------+
+30 | FLAGS |     NODE      | HM LN |
    +-------+---------------+-------+
+38 |                               |
    \           HOME                \ .acct_$home
    |                               |
+138+-------------------------------+
```

```
* local registries only
FLAGS - set of acct_$invalid (local registries only)
           (.acct_$flags)
NODE  - node type (local registries only) (.acct_$node)
HM LN - home length (.acct_$home_len)
```

Registry Record

(type rgy_$registry_t in rgy.ins.pas)

```
      +--------------+----------------+
+00   |  FLAGS       | NUM PNAMES     |  .rgy_$count
      +--------------+----------------+
+04   |  LENGTH OF REGISTRY NAME      |  .rgy_$nlen
      +-------------------------------+
+08   |  FIRST PNAME                  |  .rgy_$ent_name
      \                               \
      +-------------------------------+
      |   MORE LENGTHS AND NAMES AS   |
      |   INDICATED BY COUNT FIELD    |
      +-------------------------------+
```

First pname is path of original registry; it is used as lock.

STREAM FILE HEADER

(type "stream_$hdr_rec_t" in sbase.ins.pas)

```
     31              16 15              0
    +-----------------+-----------------+
+00 |  HEADER LENGTH  |     VERSION     | .hdr_lgth,
    +-----------------+-----------------+   .version
+04 |               C R C               | .crc
    +-----------------------------------+
+08 |     RECORD LENGTH (SEE BELOW)     + .rec_lgth
    +-----------------------------------+
+0C |     FILE LENGTH (INC. HEADER)     | .file_length
    +-----------------------------------+
+10 |--RRAESC|--CO----|--------|--------|
    +--------+--------+--------+--------+
+14 |                                   |
    /          R E S E R V E D          /
    |                                   |
+20 +-----------------------------------+
```

CRC = -(integer sum of (1 + 3 thru n longwords))
RR - Record type (.rec_type, stream_$rtype_t):
     00 - var len w/counts (stream_$v1)
     01 - fixed length (stream_$f2)
     10 - no record structure (stream_$undef)

A = 0 => Binary, 1 => ASCII (.elb_flag)
E = 1 => No automatic type change (.explicit_type)
S = 1 => File may have holes (.sparse)
C = 1 => Carriage control (ASCII only) (.cc)

CO - Concurrency (.conc, stream_$fconc_t):
     00 - N readers or 1 writer (stream_$n_or_1)
     01 - N readers and 1 writer (stream_$n_and_1)
     10 - N readers and N writers (steam_$n_and_n)

RECORD LENGTH: Record length for fixed
               Maximum length for variable
               0 for undefined record length

Stream file header is the first 32 bytes of a file to be accessed by the stream interface.

UNIQUE IDENTIFIER (UID)

(type "uid_t" in base.ins.pas)

```
 31               16 15              0
+-----------------+----------------+
|CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC|  .clock
+--------+--------+----------------+
|iiii----|----NNNN|NNNNNNNNNNNNNNNN|  .node
+--------+--------+----------------+
```

C..C - Top 32-bits of clock (4 mSec units)

iiii - A counter if more than one UID is generated
       in one four-millisecond interval

N..N - Node ID

UID Hash Algorithm

X = the four words of the UID XORed together

INDEX = X mod TABLE_SIZE

where TABLE_SIZE is the size of the table into which the UID is being hashed (e.g., vtoc_hdr.vtoc_size).

UIDS -- System

(from /os/nuc/uid_list.asm)

uid_$nil	00000000,0
acl_$nil	00000100,0

disk structure canned UIDs (000002xx series)

pv_label_$uid	00000200,0
lv_label_$uid	00000201,0
vtoc_$uid	00000202,0
bat_$uid	00000203,0

canned object type UIDs (000003xx series)

records_$uid	00000300,0
hdr_undef_$uid	00000301,0
object_file_$uid	00000302,0
UNDEF_$uid	00000304,0
pad_$uid	00000305,0
input_pad_$uid	00000309,0
sio_$uid	0000030A,0
ddf_$uid	0000030B,0
mbx_$uid	0000030C,0
nulldev_$uid	0000030D,0
D3M_area_$uid	0000030E,0
D3M_sch_$uid	0000030F,0
pipe_$uid	00000310,0
uasc_$uid	00000311,0
directory_$uid	00000312,0
unix_directory_$uid	00000313,0
mt_$uid	00000314,0
sysboot_$uid	00000315,0

canned objects UIDs (000004xx series)

display1_$uid	00000400,0
display2_$uid	00000401,0
name_$canned_root_uid	00000308,0
special_seg_$uid	00000402,0
diskless_$uid	00000403,0
name_$canned_rep_root_uid	00000404,0
display3_$uid	00000405,0
os_wired_$uid	00000406,0

canned person. project, organization
and subsystem UIDs (005xx series)

canned persons (0000050x series)

acl_$sys_user_uid	00000500,0

canned projects (0000054x series)

acl_$sys_proj_uid	00000540,0
acl_$login_uid	00000541,0
acl_$locksmith_uid	00000542,0

canned organizations (0000058x series)

acl_$sys_org_uid	00000580,0

canned subsystems (000005Cx series)

acl_$nil_subs_uid	000005C0,0

canned ACL type UIDs (000006xx series)

acl_$file_acl	00000600,0
acl_$dir_acl	00000601,0

canned ACL UIDs (file ACLs) (0001xxxx series)

acl_$fnil	00010000,0
acl_$fndwrx	0001800F,0
acl_$file_nwrx	00018007,0

canned ACL UIDs (directory ACLs) (0002xxxx series)

acl_$dnil	00020000,0
acl_$dndcal	0002801F,0
acl_$dir_ncal	0002800F,0

VOLUME LABEL -- LOGICAL

(type "lv_label_t" in vol.ins.pas)

```
      31            16 15            0
    +---------------+---------------+
+00 |    VERSION    |    UNUSED     | .version
    +---------------+---------------+
+04 |                               | .name
    /      LOGICAL VOLUME NAME      /
    |                               |
    +-------------------------------+
+24 |          UNIQUE ID OF         | .id
    +                               +
    |         LOGICAL VOLUME        |
    +-------------------------------+
+2C |                               | .bat_hdr
    /           BAT HEADER          /
    |                               |
    +-------------------------------+
+4C |                               | .vtoc_hdr
    /          VTOC HEADER          /
    |                               |
    +-------------------------------+
+B0 |      TIME LABEL WRITTEN       | .label_write_time
    +-------------------------------+
+B4 |----------| LAST MOUNTED NODE  | .last_mounted_node
    +-------------------------------+
+B8 |    TIME SYSTEM WAS BOOTED     | .node_boot_time
    +-------------------------------+
+BC | TIME THIS VOLUME WAS MOUNTED  | .mounted_time
    +-------------------------------+
+C0 |TIME THIS VOLUME WAS DISMOUNTED| .dismounted_time
    +-------------------------------+
+C4 |----------|NODE OF LAST SALVAGE| .salvage_node
    +-------------------------------+
+C8 |    TIME SALVAGE COMPLETED     | .salvage_time
    +---------------+---------------+
+CC |MODE OF SALVAGE|SHUTDOWN STATE | .salvage_mode,
    +---------------+---------------+   .sys_shut_state
+D0 |       TIME DUMP STARTED       | .dump_start_time
    +-------------------------------+
+D4 |       TIME DUMP FINISHED      | .dump_end_time
    +-------------------------------+
+D8 |        UID OF CURRENT         | .dump_cur_uid
    +                               +
    |       ITEM BEING DUMPED       |
    +---------------+---------------+
    |    CONTINUED ON NEXT PAGE     |
       |                                |
 +E0   |# MINS FROM UTC|  NAME OF ...   |  .utc_delta
       +---------------+----------------+    .timezone_name
 +E4   |  ... TIMEZONE |   LAST ...     |  .last_valid_time
       +--------------------------------+
 +E8   | ... VALID TIME|    UNUSED      |
       +---------------+----------------+
 +EC   |        BAD SPOT BARRIER*       |  .bad_spot_barrier
       +--------------------------------+
 +F0   |        BAD SPOT LIST [60]      |  .bad_spot_list[60]
       +--------------------------------+
       |               -                |
       /               -                /
       |               -                |
       +--------------------------------+
+3FC   |        BAD SPOT LIST [255]     |
+400   +--------------------------------+
```

```
salvage_mode: currently unused; always = 1

sys_shut_state:    lv_dismounted = 0
                   lv_mounted    = 1
                   lv_salvaged   = 2
```

bad_spot_list allocated from end of list.

```
* FFFFFFFF -> no badspot list overflow
  FFFFFFFE -> +FO is DADDR of overflow block
```

The LV label is the first block of a logical volume. The alternate LV label is a copy of the LV label and lives at or near the end of the logical volume.

VOLUME LABEL -- PHYSICAL

(type "pv_label_t" in vol.ins.pas)

```
      31            16 15             0
     +---------------+----------------+
+00  |    VERSION    |  "A"     "P"   |   .version,
     +---------------+----------------+     .apollo
     |    "O"     "L"     "L"     "O" |
     +--------------------------------+
+08  |                                |   .name
     /          VOLUME NAME           /
     |                                |
     +--------------------------------+
+28  |            UNIQUE ID           |   .id
     |            OF VOLUME           |
     +---------------+----------------+
+30  |     UNUSED    |  DISK  TYPE    +   .dtype
     +---------------+----------------+
+34  |    TOTAL BLOCKS IN VOLUME      |   .blocks_per_pvol
     +---------------+----------------+
+38  |BLKS PER TRACK |TRACKS PER CYL  |   .blocks_per_track
     +---------------+----------------+   .tracks_per_cyl
+3C  |  DADDR OF LOGICAL VOLUME [1]   |   .lv_list[1]
     +--------------------------------+
     |                                |
     /            . . .               /
     |                                |
     +--------------------------------+
+60  |  DADDR OF LOGICAL VOLUME [10]  |   .lv_list[10]
     +--------------------------------+
+64  |   ALTERNATE LABEL DADDR [1]    |   .alt_lv_list[1]
     +--------------------------------+
+68  |                                |
     /            . . .               /
     |                                |
     +--------------------------------+
+88  |   ALTERNATE LABEL DADDR [10]   |   .alt_lv_list[10]
     +--------------------------------+
+8C  |   START OF BADSPOT CYLINDER    |   .phys_badspot_daddr
     +--------------------------------+
+90  |    START OF DIAGNOSTIC CYL     |   .phys_diag_daddr
     +--------------+-----------------+
+94  | SECTOR START |   SECTOR SIZE   |   .phys_sector_start,
     +--------------+-----------------+      .size
+98  | PRE-COMP CYL |                     .pre_comp
+9A  +--------------+
```

The DISK TYPE field describes variants of the physical disk, e.g., double density. Today there are none and the field contains 0.

The PV label is the first block (cylinder 0, track 0, block 0) of a physical volume.

VTOC BLOCK

```
        (type "vtoc_blk_t" in vol.ins.pas)
              31                               0
             +----------------------------------+
         +00 |  --> NEXT BLOCK IN HASH BUCKET  |   .next_add
             +----------------------------------+
         +04 |          VTOC ENTRY[0]           |   .vtoc[0]
             +----------------------------------+
         +D0 |          VTOC ENTRY[1]           |
             +----------------------------------+
        +19C |          VTOC ENTRY[2]           |
             +----------------------------------+
        +268 |          VTOC ENTRY[3]           |
             +----------------------------------+
        +334 |          VTOC ENTRY[4]           |   .vtoc[4]
        +400 +----------------------------------+

                           -or-

             +----------------------------------+
         +00 |           FILE MAP[0]            |   .fm[0]
             +----------------------------------+
         +80 |           FILE MAP[1]            |
             +----------------------------------+
        +100 |           FILE MAP[2]            |
             +----------------------------------+
        +180 |           FILE MAP[3]            |
             +----------------------------------+
        +200 |           FILE MAP[4]            |
             +----------------------------------+
        +280 |           FILE MAP[5]            |
             +----------------------------------+
        +300 |           FILE MAP[6]            |
             +----------------------------------+
        +380 |           FILE MAP[255]          |   .fm[255]
        +400 +----------------------------------+
```

When the VTOC block contains a file map, the block is pointed to by vtoce.fm2[1-3] (see VTOC ENTRY).

VTOC ENTRY

(types "vtoce_hdr_t" and "vtoce" in vol.ins.pas)

```
     31    24 23    16 15               0
    +--------+--------+-----------------+
+00 | VERSION|SYS_TYPE|UCCPIFLB---------|  .version,
    +--------+--------+-----------------+    .sys_type
+04 |              OBJECT               |  .uid
    |               UID                 |
    +-----------------------------------+
+12 |   UID OF TYPE DEFINITION OBJECT   |  .type_uid
    |          FOR THIS OBJECT          |
    +-----------------------------------+
+20 |         UID OF ACL OBJECT         |  .acl_uid
    |          FOR THIS OBJECT          |
    +-----------------------------------+
+28 |       CURRENT LENGTH (BYTES)      |  .cur_len
    +-----------------------------------+
+32 |       BLOCKS USED FOR FILE        |  .blocks_used
    +-----------------------------------+
+36 |        DATE-TIME LAST USED        |  .dtu
    +-----------------------------------+
+40 |      DATE-TIME LAST MODIFIED      |  .dtm
    +-----------------------------------+
+44 |         UID OF DIRECTORY          |  .dir_uid
    |    WHERE OBJECT IS CATALOGUED     |
    +-----------------+-----------------+
+52 | EXTRA DTM BITS  |# OF REFS TO OBJ |  .extdtm,
    +-----------------+-----------------+    .ref_cnt
+56 |           FILE LOCK KEY           |  .lock_key
    +-----------------------------------+
+60 |              PAD3                 |  .pad3
+64 +-----------------------------------+
     U  - VTOC entry in use (.inuse)
     CC - Concurrency control (.con_ctrl):
          00 - None
          01 - Shared
          10 - Exclusive
     P  - Permanent (.permanent)
     I  - Immutable (.immutable)
     F  - File needs salvaging (.trouble)
     L  - Local access only (.local_acc_only)
     B  - (.pad1) (0..511)
```

VTOC HEADER

(type "vtoc_hdr_t" in vol.ins.pas)

```
     31             16 15              0
    +----------------+----------------+
+4C | VERSION NUMBER |# BLKS FOR HASH |  .version,
    +----------------+----------------+     .vtoc_size
+50 |      NUMBER VTOC BLOCKS USED     |  .vtoc_blocks
    +----------------------------------+
+54 |       VTOCX OF NETWORK ROOT      |  .net_x
    +----------------------------------+
+58 | VTOCX OF ROOT DIR OF THIS VOLUME|  .root_x
    +----------------------------------+
+5C |  VTOCX OF PAGING FILE FOR AEGIS |  .os_x
    +----------------------------------+
+60 |         VTOCX OF BOOT FILE       |  .boot_x
    +----------------------------------+
+64 |                                  |  .map
    /             VTOC MAP             /
    /       (8 VTOC MAP ENTRIES)       /
    |                                  |
    +----------------------------------+
+94 |                                  |  .pad
    /              UNUSED              /
    |                                  |
+B0 +----------------------------------+
```

The VTOC header lives in the logical volume label. Offsets given are from the start of the label.

VTOC MAP ENTRY

(type "vtoc_mape" in vol.ins.pas)

```
     15              0
    +----------------+
+00 |# CONSEC. BLOCKS|  .lt_blk
    +----------------+
+02 |  DISK ADDRESS  |  .blk_add
    +                +
    | OF FIRST EXTENT|
+06 +----------------+
```

Each VTOC map entry describes one set of contiguous VTOC blocks ("extent"). VTOC extents are preallocated by INVOL to be near the middle of the logical volume and to avoid badspots.

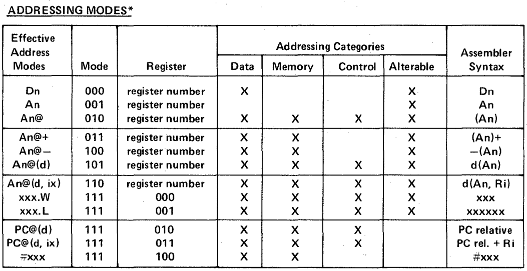
VTOC INDEX

```
(type "vtocx_t" in base.ins.pas)

 31                         4 3  0
+----------------------------+-----+
|0DADDR OF VTOC BLK OF OBJECT|INDX|   (local object)
+----------------------------+-----+

              -or-

              19
+------------+--------------------+
|1TTTTTNNNNNN|       NODE ID      |   (remote object)
+------------+--------------------+

                -or-

+----------------------------+-----+
|0                           |VOLX|   (local, but DADDR
+----------------------------+-----+      is unknown)

INDX - Index of VTOC entry in VTOC block (0-4)
          or File Map index (0-7)
TTTTT - Touch ahead count
NNNNNN - Network ID  (netx_t)
VOLX - Logical volume number
```

CHAPTER 3

PROGRAMMING INFORMATION

ADDRESSING MODES



ADDRESSING MODES*

Effective Address Modes	Mode	Register	Addressing Categories				Assembler Syntax
			Data	Memory	Control	Alterable	
Dn	000	register number	X			X	Dn
An	001	register number				X	An
An@	010	register number	X	X	X	X	(An)
An@+	011	register number	X	X		X	(An)+
An@–	100	register number	X	X		X	–(An)
An@(d)	101	register number	X	X	X	X	d(An)
An@(d, ix)	110	register number	X	X	X	X	d(An, Ri)
xxx.W	111	000	X	X	X	X	xxx
xxx.L	111	001	X	X	X	X	xxxxxx
PC@(d)	111	010	X	X	X		PC relative
PC@(d, ix)	111	011	X	X	X		PC rel. + Ri
#xxx	111	100	X	X			#xxx

*Reprinted from MC68000, page B-1.

ADDRESS SPACE

Physical Address Space

```
non-DNx60
(except DN3000,DN5xx-T)       DNx60
        0 +-----------------+ 0
          |    TRAP PAGE    |
      400 +-----------------+ 400
          |       PROM      |
     4000 +-----------------+ 8000
          |        I/O      |
          |      OBJECTS    |
    20000 +-----------------+ 20000
          | DISPLAY BITMAP  |
    40000 +-----------------+ 400000
          |DISPLAY2 BITMAP  |
    80000 +-----------------+ --
          | OPT. 1/2MB MEM  |
   100000 +-----------------+ 200000
          | PROM DATA AREA  |
   100400 +-----------------+ 200800
          |                 |
          |       MAIN      |
          |                 |
          /                 /
          |      MEMORY     |
          |                 |
   400000 +-----------------+ 1000000
```

Refer to Chapter 12, DN3000 for information about address space for the DN3000.

Virtual memory

```
   Reverse Mapped Mach.       DNx60
        -- +----------------+ --
           |   TRAP PAGE    |
       400 +----------------+ 400
           |      PROM      |
      8000 +----------------+ 8000
           /  USER PROCESS  /
           |  PRIVATE DATA  |
    940000 +----------------+ E780000
           | GUARD SEGMENT  |
           |                |
    948000 +----------------+ E788000
           | USER STACK     |
           |                |
    988000 +----------------+ E7C8000
           | GUARD SEGMENT  |
    990000 +----------------+ E7D0000
           | PM STATIC DATA |             PRIVATE (USER MODE)
           |                |
    9C0000 +----------------+ E800000<---- GLOBAL A BOUNDARY
           |GLOBAL LIBRARIES|
           |                |
    BC0000 +----------------+ F000000 <---- PRIVATE (SUPER-
           |  PER PROCESS   |               VISOR MODE)
           |SUPERVISOR SPACE|
    C00000 +----------------+ --
           |     UNUSED     |
    D00000 +----------------+ --      <---- GLOBAL B BOUNDARY
           | unwired stacks |
           |     etc        |
    E00000 +----------------+ F800000
           |                |
           |     AEGIS      |
           |                |
    FA0000 +----------------+ FFA0000
           | DISPLAY BITMAPS|
    FE0000 +----------------+ FFE0000
           |   I/O CONTROL  |
           | PAGES & I/O MAP|
   1000000 +----------------+ 10000000
```

Refer to the appropriate node chapter for information about virtual address space for forward mapping machines.

CALLING SEQUENCE - PIC

Caller:

```
    PEA       ARGn          PUSH ADDRESS OF LAST ARG
    ...
    PEA       ARG1          PUSH ADDRESS OF FIRST ARG
    MOVE.L    ECBADR,A0     GET ADDRESS OF ENTRY POINT
    JSR       (A0)          JUMP AND PUSH PC
    ADD.W     #4*n,SP       POP ARG PTRS OFF STACK
```

Impure Subroutine Entrypoint (PIC only):

```
    LEA       static_data,a0
    JMP.L     subroutine_entry
```

Subroutine entry:

```
    LINK      SB,#autosize LOAD MY STACK BASE
    ...                     & DEFINE AUTOMATIC STORAGE
    MOVEM.L   reg_save_mask,-(SP)  SAVE CALLER'S REGs
    MOVE.L    A0,DB         LOAD MY DATA BASE
    MOVE.L    8(SB),WORK   GET ADDR FIRST ARG
    ...
```

Subroutine return:

```
    MOVEM.L   -autosize(SB),reg_save_mask RESTORE CALLER'S REGs
    UNLK      SB            RELOAD CALLER'S SB
    RTS                     RETURN TO CALLER
```

See also STACK FRAME, ECB.

Note: Registers other than D0, D1, A0, A1 are preserved.

CONDITION CODES*

Operations	X	N	Z	V	C	Special Definition
ABCD	*	U	?	U	?	C = Decimal Carry $Z = Z \cdot \overline{Rm} \cdot \ldots \cdot \overline{R0}$
ADD, ADDI, ADDQ	*	*	*	?	?	$V = Sm \cdot Dm \cdot \overline{Rm} + \overline{Sm} \cdot \overline{Dm} \cdot Rm$ $C = Sm \cdot Dm + \overline{Rm} \cdot Dm + Sm \cdot \overline{Rm}$
ADDX	*	*	?	?	?	$V = Sm \cdot Dm \cdot \overline{Rm} + \overline{Sm} \cdot \overline{Dm} \cdot Rm$ $C = Sm \cdot Dm + \overline{Rm} \cdot Dm + Sm \cdot \overline{Rm}$ $Z = Z \cdot \overline{Rm} \cdot \ldots \cdot \overline{R0}$
AND, ANDI, EOR, EORI, MOVEQ, MOVE, OR, ORI, CLR, EXT, NOT, TAS, TST	–	*	*	0	0	
CHK	–	*	U	U	U	
SUB, SUBI SUBQ	*	*	*	?	?	$V = \overline{Sm} \cdot Dm \cdot \overline{Rm} + Sm \cdot \overline{Dm} \cdot Rm$ $C = Sm \cdot \overline{Dm} + Rm \cdot \overline{Dm} + Sm \cdot Rm$
SUBX	*	*	?	?	?	$V = \overline{Sm} \cdot Dm \cdot \overline{Rm} + Sm \cdot \overline{Dm} \cdot Rm$ $C = Sm \cdot \overline{Dm} + Rm \cdot \overline{Dm} + Sm \cdot Rm$ $Z = Z \cdot \overline{Rm} \cdot \ldots \cdot \overline{R0}$
CMP, CMPI, CMPM	–	*	*	?	?	$V = \overline{Sm} \cdot Dm \cdot \overline{Rm} + Sm \cdot \overline{Dm} \cdot Rm$ $C = Sm \cdot \overline{Dm} + Rm \cdot \overline{Dm} + Sm \cdot Rm$
DIVS, DIVU	–	*	*	?	0	V = Division Overflow
MULS, MULU	–	*	*	0	0	
SBCD, NBCD	*	U	?	U	?	C = Decimal Borrow $Z = Z \cdot \overline{Rm} \cdot \ldots \cdot \overline{R0}$
NEG NEGX	* *	* *	* ?	? ?	? ?	$V = Dm \cdot Rm, C = Dm + Rm$ $V = Dm \cdot Rm, C = Dm + Rm$ $Z = Z \cdot \overline{Rm} \cdot \ldots \cdot \overline{R0}$
BTST, BCHG, BSET, BCLR	–	–	?	–	–	$Z = \overline{Dn}$
ASL	*	*	*	?	?	$V = Dm \cdot (\overline{D_{m-1}} + \ldots + \overline{D_{m-r}})$ $+ \overline{Dm} \cdot (D_{m-1} + \ldots + D_{m-r})$ $C = D_{m-r+1}$
ASL (r = 0)	–	*	*	0	0	
LSL, ROXL	*	*	*	0	?	$C = D_{m-r+1}$
LSR (r = 0)	–	*	*	0	0	
ROXL (r = 0)	–	*	*	0	?	C = X
ROL	–	*	*	0	?	$C = D_{m-r+1}$
ROL (r = 0)	–	*	*	0	0	
ASR, LSR, ROXR	*	*	*	0	?	$C = D_{r-1}$
ASR, LSR (r = 0)	–	*	*	0	0	
ROXR (r = 0)	–	*	*	0	?	C = X
ROR	–	*	*	0	?	$C = D_{r-1}$
ROR (r = 0)	–	*	*	0	0	

– Not affected
U Undefined
? Other – see Special Definition

* General Case:
X = C
N = Rm
$Z = \overline{Rm} \cdot \ldots \cdot \overline{R0}$

Sm – Source operand most significant bit
Dm – Destination operand most significant bit
Rm – Result bit most significant bit
n – bit number
r – shift amount

*Reprinted from MC68000, page A-4.

CONDITIONAL TESTS*

Mnemonic	Condition	Encoding	Test
T	true	0000	1
F	false	0001	0
HI	high	0010	$\overline{C} \cdot \overline{Z}$
LS	low or same	0011	$C + Z$
CC	carry clear	0100	$\overline{C}$
CS	carry set	0101	C
NE	not equal	0110	$\overline{Z}$
EQ	equal	0111	Z
VC	overflow clear	1000	$\overline{V}$
VS	overflow set	1001	V
PL	plus	1010	$\overline{N}$
MI	minus	1011	N
GE	greater or equal	1100	$N \cdot V + \overline{N} \cdot \overline{V}$
LT	less than	1101	$N \cdot \overline{V} + \overline{N} \cdot V$
GT	greater than	1110	$N \cdot V \cdot \overline{Z} + \overline{N} \cdot \overline{V} \cdot \overline{Z}$
LE	less or equal	1111	$Z + N \cdot \overline{V} + \overline{N} \cdot V$

*Reprinted from MC68000, page A-4.

FILENAME SUFFIXES

SUFFIX	MEANING	RECOGNIZED BY
.ASM	Assembler source	Assembler (input)
.BAK	Backup file	Display manager (output)
.BIN	Binary file	Compilers (output)
.BND	Binder	
.BS	Boot Shell command	Boot Shell
.C	C source	CC (input)
.DATA	Data file	
.FTN	FORTRAN source	FTN (input)
.HLP	Help text	HELP command (input)
.INS	Insert file	
.LST	Listing file	Compilers (output)
.MAP	Map file	Binder (output)
.PAS	Pascal source	PAS (input)
.RFC	Run file converter	RFC command (output), /sysboot (input)

D3M SUFFIX	MEANING
.DDL	Schema, subschema, aggregate schema DDL
.FMT	Output from the RDL
.LST	ASCII listing of the schema, subschema, aggregate schema compiler
.RPT	Output from the D3M/FORMATTER
.CMD	Executable D3M/DATAVIEW commands
.RDL	Source for report writer
.UWA.xxx	UWA definition generated by SSCH

DOMAIN/IX SUFFIX	MEANING
.c	C compiler
.o	Binary file from compiler
.h	C insert file

SCRIBE SUFFIX	MEANING
.MSS	Manuscript
.OTC	Outline
.LPT	Lineprinter
.AUX	Auxillary
.ERR	Error listing

WPS SUFFIX	MEANING
.doc	Document (file)
.fdr	Folder (directory)
.drw	Drawer (directory)
.cab	Cabinet (directory)
.imp	Impress document (file)
.clp	Clipboard (directory)

These conventions are not requirements; you can give a file any name you like, within the syntax rules. The operating system does not check a file's contents against its name. However, some programs assume that the names of input files end with a particular suffix.

PATHNAME SYNTAX

Symbol	Starting Point
//	Network root directory
/	Node entry directory
~	Naming directory
\	Parent directory
. or none	Working directory
'node_data	/sys/node_data[.nn]

Legal characters in names:

A-Z	
a-z	
0-9	
$	(dollar sign)
_	(underscore)
.	(period)

Valid pathnames:

```
/PASCAL
\MISC/SAU_SOURCE
//US/INS/STREAMS.INS.FTN
~com
~link_name
```

STACK FRAME

```
                    no floating point registers
                   +------------------+
        SP--->     |  Save registers  |
                   +------------------+
                   |                  |
                   |    AUTOMATIC     |
                   |     STORAGE      |
                   |                  |
                   +------------------+
        SB --> +00 |   CALLER'S SB    | --------+
                   +------------------+         |
               +04 |  opt FCW1 PTR    | -----------------+
                   +---  . . .  ------+         |        |
             +04*m |  opt FCWm PTR M  |         |        |
                   +------------------+         |        V
          +04+04*m |  PC FOR RETURN   |         |    Frame control
                   +------------------+         |      word ptr + 1
          +08+04*m |      --> ARG1    |         V
                   +------------------+      PREVIOUS
          +0c+04*m |      --> ARG2    |     STACK FRAME
                   +------------------+
                 . |                  |
                 . /                  /
                 . |                  |
                   +------------------+
        +4+4*n+4*m |      --> ARGn    |
                   +------------------+
                   |                  |
                   |                  |
```

```
            FCW PTR = nil        NOP   Non-nil FCWs are distinguished
            FCW PTR = addr+1 of FCW    from return PCs by the fact
                                       that they are odd-addresses.
```

```
                   +------------------+
        FCW        | type    |  mask  |
                   +------------------+
               +04 |  offset from SB  |------> pointer to register
                   +------------------+            save area

                type = 0 = unknown
                       1 = 68881 floating point registers
                       2 = DNx60 (tern) floating point registers

                mask = bit mask used in:
                   movem.l f2-f7,-(sp)
                      instruction

               +-------------------------------+
               |fp0|fp1|fp2|fp3|fp4|fp5|fp6|fp7|
               +-------------------------------+
```

STATUS WORD

(type "status_t" in base.ins.pas)

```
 31     24 23     16 15              0
+--------+--------+----------------+
|FSSSSSSS|AMMMMMMM|CCCCCCCCCCCCCCCC|
+--------+--------+----------------+

F    = 1 => module couldn't handle error (fail bit)
S..S - Subsystem identification
A    = 1 => asynchronous fault; only set during delivery
            of fault (.async)
M..M - Module identification
C..C - Module-specific error code
```

See Chapter 4, Error Codes and Messages.

CHAPTER 4

ERROR CODES AND MESSAGES

AEGIS ERROR CODES

```
 31     24 23     16 15              0
+--------+--------+----------------+
|FSSSSSSS|AMMMMMMM|CCCCCCCCCCCCCCCC|
+--------+--------+----------------+

F    = 1 => module couldn't handle error
S..S - Subsystem identification
A    = 1 => asynchronous fault; only set during delivery
             of fault
M..M - Module identification
C..C - Module-specific error code:

         0        = OK status
         negative = warning
         positive = error

(type "status_$t" in base.ins.pas)
```

00000000	status_$ok
	OS / BAT manager:
(00010001)	attempt to free already-freed block
(00010002)	disk is full
(00010003)	attempt to free illegal disk address
(00010004)	BAT not mounted
(00010005)	disk needs salvaging
	OS / VTOC manager:
(00020001)	VTOC not mounted
(00020002)	VTOC is bad
(00020003)	no file map
(00020004)	no UID
(00020005)	not found
(00020006)	UID not found
(00020007)	duplicate UID
(00020008)	uid mismatch
(00020009)	only local access allowed
	OS / AST manager:
(00030001)	attempted reference to out-of-bounds address

(00030003)	no replaceable aste's
(00030004)	segment is not deactivatable
(00030005)	write concurrency violation
(00030006)	incompatible request
(00030007)	reference count says unused
(00030008)	segment not found in bst
(00030009)	segment thread error in bst
(0003000A)	only local access allowed
	OS / MST manager:
(00040001)	object not found
(00040002)	invalid length
(00040003)	no space available
(00040004)	reference to illegal address
(00040005)	reference to out-of-bounds address
(00040006)	no asid is available
(00040007)	object is not mapped
(00040008)	no rights
(00040009)	insufficient rights
(0004000A)	guard fault
(0004000B)	wrong type - can't map system objects
(0004000C)	ppn list overflow
(0004000D)	uid mismatch
(0004000E)	virtual memory resources exhausted
(0004000F)	invalid va for install of io page
(00040010)	invalid segment count
(00040011)	asid 0 is illegal for this mapping
	OS / PMAP manager:
(00050001)	not allocated
(00050002)	already allocated
(00050003)	mismatch
(00050004)	bad wire
(00050005)	bad unwire
(00050006)	bad assoc
(00050007)	pages wired
(00050008)	page null
(00050009)	bad disk address
(0005000A)	read concurrency violation
(0005000B)	changed pmods
(0005000C)	invalid pmape
(0005000D)	attempt to map i/o page over real page
(0005000E)	bst threads yielded invalid va
(00050010)	illegal pid argument from dxm callback
(00050011)	illegal wsl index
	OS / MMAP manager:
(00060004)	bad avail
(00060005)	bad free
(00060006)	bad unavail
(00060008)	inconsistent mmape
(00060009)	illegal wsl index
(0006000A)	illegal pid
(0006000B)	ws lists exhsusted
(0006000C)	bad install
(0006000D)	bad reclaim

(0006000E)	contiguous pages unavailable
	OS / MMU manager:
(00070001)	mmu miss
(00070002)	va not in valid mmu manager range
(00070003)	va does not have os_pmap
(00070004)	ptt parity error
(00070005)	pft parity error
(00070006)	mmu timeout
(00070007)	unknown mmu status
(00070008)	mmu parity error
(00070009)	data cache parity error
(0007000A)	unexpected virtual timeout
	OS / disk manager:
(00080001)	disk not ready
(00080002)	disk controller busy
(00080003)	disk controller time-out
(00080004)	disk controller error
(00080005)	disk equipment check
(00080006)	floppy is not 2-sided
(00080007)	disk write protected
(00080008)	bad disk format
(00080009)	disk data check
(0008000A)	DMA overrun
(0008000B)	volume in use
(0008000C)	volume table full
(0008000D)	volume not properly mounted or assigned
(0008000E)	operation requires a physical volume
(0008000F)	invalid volume index
(00080010)	logical volume not found
(00080011)	disk block header error
(00080012)	invalid disk address
(00080013)	disk buffer is not page aligned
(00080014)	invalid logical volume index or list
(00080015)	disk seek error
(00080016)	drive timed out before operation completed
(00080017)	bus error occurred during disk DMA transfer
(00080018)	invalid unit number
(00080019)	unknown status returned by hardware
(0008001A)	invalid physical volume label
(0008001B)	floppy door has been opened or storage module has been stopped
(0008001C)	read after write failed
(0008001D)	dma not at end of range
(0008001E)	disk already mounted
(0008001F)	software detected checksum error
(00080020)	checksum error in read after write
(00080021)	too many wired pages -- storage module manager
(00080022)	disk driver logic error
(00080023)	unknown error status from drive
(00080024)	unrecognized drive id
(00080025)	memory parity error during disk write
(00080026)	unrecognized interrupt from disktape controller
(00080027)	ecc error in sector id field
(00080028)	disk subsystem detected a DC powerfail

	OS / eventcount manager:
(00090001)	bad wait list on eventcount
	OS / level 1 process manager:
(000A0001)	illegal process id
(000A0002)	illegal lock
(000A0003)	process not suspended
(000A0004)	process already suspended
(000A0005)	process not bound
(000A0006)	process already bound
(000A0007)	bad atomic operation
(000A0008)	no pcb is available
(000A0009)	no stack space is available
(000A000A)	process not suspendable
(000A000B)	ready list is out of order
	OS / terminal manager:
(000B0001)	buffer too small
(000B0002)	end of file entered from keyboard
(000B0003)	invalid output length
(000B0004)	invalid option passed to term_$control
(000B0005)	input buffer overrun - characters lost
(000B0006)	asynchronous fault occurred while waiting for input
(000B0007)	invalid line number supplied
(000B0008)	manual stop: type G<ret>G *+2<ret> to continue
(000B0009)	character framing error
(000B000A)	character parity error
(000B000B)	data carrier detect (dcd) changed
(000B000C)	clear to send (cts) changed
(000B000D)	requested line or operation not implemented
(000B000E)	hangup fault
(000B000F)	speed incompatible with partner SIO line
	OS / DBUF manager:
(000C0001)	bad ptr
(000C0002)	bad free
	OS / time manager:
(000D0001)	no timer queue entry
(000D0002)	entry to be cancelled not found
(000D0003)	quit while waiting for event
(000D0004)	bad timer interrupt
(000D0005)	bad timer key
(000D0006)	alarm fault
(000D0007)	real interval timer fault
(000D0008)	virtual interval timer fault
(000D0009)	queue element not in use
(000D000A)	queue element not found
	OS / naming server:
(000E0002)	directory is full
(000E0003)	name already exists
(000E0004)	invalid pathname
(000E0005)	invalid link

(000E0006)	not a link
(000E0007)	name not found
(000E000A)	invalid link operation
(000E000B)	invalid leaf
(000E000C)	node is unavailable
(000E000D)	bad directory
(000E000E)	branch is not a directory
(000E000F)	directory is not empty
(000E0010)	name is not a file
(000E0011)	illegal directory operation
(000E0012)	bad type
(000E0013)	no rights
(000E0014)	insufficient rights
(000E0015)	unable to delete system bootstrap (sysboot)
(000E0016)	directory is in use (locked)
(000E0017)	name server helpers clocks are skewed
(000E0018)	illegal request made of name server helper
(000E0019)	cannot find entry in replicated root
(000E001A)	last entry in replicated root returned
(000E001B)	name server helper is shutdown
(000E001C)	name server helper sent packet with errors
(000E001D)	clocks skewed
(000E001E)	cant find name server helper
(000E001F)	directory must be root
(000E0020)	directory not found in pathname
(000E0021)	too many components in pathname
(000E0022)	cache entry is stale
(000E0023)	cache entry was stale and was updated
(000E0024)	name server helper is uninitialized

OS / file server:

(000F0001)	object not found
(000F0002)	object is remote
(000F0003)	bad reply received from remote node
(000F0004)	communications problem with remote node
(000F0005)	object is not locked by this process
(000F0006)	object is in use
(000F0007)	illegal lock request
(000F0008)	lock violation detected
(000F0009)	local lock table is full
(000F000A)	remote lock table is full
(000F000B)	operation cannot be done from here
(000F000C)	no more lock table entries
(000F000D)	volume uid is unavailable
(000F000E)	locking files is blocked for this volume
(000F000F)	locking is already blocked for this volume
(000F0010)	no rights
(000F0011)	insufficient rights
(000F0012)	wrong type - can't operate on system objects
(000F0013)	objects are on different volumes

OS / I/O manager:

(00100001)	dcte not found
(00100002)	controller not in system

OS / network:

(00110001)	buffer error
(00110002)	out of pages
(00110003)	out of blocks
(00110004)	transmit failed
(00110005)	no available socket
(00110006)	buffer queue is empty
(00110007)	remote node failed to respond to request
(00110008)	unable to route
(00110009)	network hardware error
(0011000A)	msg header too big
(0011000B)	unexpected reply type
(0011000C)	no more free sockets
(0011000D)	unknown request type
(0011000E)	request denied by local node
(0011000F)	request denied by remote node
(00110010)	bad checksum
(00110011)	too many transmit retries
(00110012)	socket not open
(00110013)	receive bus error
(00110014)	transmit bus error
(00110015)	bad asknode version number
(00110016)	memory parity error during transmit
(00110017)	unknown network
(00110018)	too many networks in internet
(00110019)	conflict with another node listing
(0011001A)	quit fault during node listing
(0011001B)	waited too long for more nodes to respond
(0011001C)	data length too large
(0011001D)	operation not defined on network hardware
(0011001E)	header length + data length exceeds max msg size
(0011001F)	no nodeid prom on this system

OS / fault handler:

(00120001)	odd address error
(00120002)	illegal instruction
(00120003)	integer divide by zero
(00120004)	CHK instruction trapped - value out of range?
(00120005)	arithmetic overflow
(00120006)	privileged instruction violation
(00120007)	invalid SVC code
(00120008)	invalid SVC procedure name
(00120009)	undefined TRAP instruction
(0012000A)	unimplemented instruction
(0012000B)	protection boundary violation
(0012000C)	bus time-out
(0012000D)	invalid user stack pointer
(0012000E)	correctable memory error detected
(0012000F)	uncorrectable memory error detected
(00120010)	process quit
(00120011)	access violation
(00120012)	CPU B enabled with MMU valid bit reset
(00120013)	null process running on CPU B
(00120014)	OS-internal quit (with display return)
(00120015)	single step completed

(00120016)	invalid user-generated fault (subsystem code = 0)
(00120017)	fault in user-space interrupt handler for pbu device
(00120018)	process stop
(00120019)	process BLAST
(0012001A)	PEB cache parity error
(0012001B)	PEB WCS parity error
(0012001C)	unimplemented SVC
(0012001D)	invalid stack format
(0012001E)	memory parity error
(0012001F)	process interrupt
(00120020)	supervisor fault while resource lock(s) set
(00120021)	spurious parity error
(00120022)	floating point inexact result (loss of significance)
(00120023)	floating point divide by zero
(00120024)	floating point underflow
(00120025)	floating point operand error
(00120026)	floating point overflow
(00120027)	process suspend fault
(00120028)	process suspend from keyboard
(00120029)	process suspend due to background read
(0012002A)	process suspend due to background write
(0012002B)	process continue fault
(0012002C)	fault(s) lost; process suspended or pfm_$enable/pfm_$inhibit mismatch?
(0012002D)	coprocessor protocol violation
(0012002E)	floating point branch/set on unordered condition
(0012002F)	floating point signalling not-a-number
(00120030)	invalid thread during parity error check
(00120031)	illegal page fault in user gpio interrupt routine
(00120032)	bus error while running on cpu B
(00120033)	spurious interrupt
(00120034)	unexpected bus error during system initialization
(00120035)	cleanup handler set
(00120036)	cleanup handler released out of order
(00120037)	ac power failure
(00120038)	fpx parity error
(00120039)	unknown fpa vector exception
(0012003A)	VME bus error on bus error
	OS / display driver:
(00130001)	invalid display unit number
(00130002)	specified font not loaded
(00130003)	internal font table full
(00130004)	invalid use of display driver procedure
(00130005)	font too large
(00130006)	error unloading internal (hdmt) table
(00130007)	invalid direction from SM
(00130008)	unexpected BLT in use
(00130009)	internal protocol violation
(0013000A)	too many pages to be wired

(0013000B)	unsupported font version
(0013000C)	invalid buffer size
(0013000D)	error mapping display memory
(0013000E)	error borrowing display from screen manager
(0013000F)	unable to borrow - display in use
(00130010)	display borrow request denied by screen manager
(00130011)	error returning display to screen manager
(00130012)	can't return - display not borrowed
(00130013)	can't borrow both displays simultaneously
(00130014)	display already borrowed by this process
(00130015)	invalid position argument
(00130016)	invalid window limits argument
(00130017)	invalid length argument
(00130018)	invalid direction argument
(00130019)	invalid scroll displacement argument
(0013001A)	invalid blt mode register
(0013001B)	invalid blt control register
(0013001C)	invalid blt-done interrupt
(0013001D)	invalid interrupt routine state
(0013001E)	invalid screen coordinates in blt request
(0013001F)	font associated with specified id not mapped
(00130020)	display memory is already mapped
(00130021)	display memory is not mapped
(00130022)	quit while waiting
(00130023)	invalid cursor number
(00130024)	hidden display memory is full
(00130025)	quit while waiting
(00130026)	invalid eventcount key
(00130027)	operation not implemented on color display
(00130028)	non-conforming and main memory blts not implemented
(00130029)	invalid DM window id
(0013002A)	acquire denied because window is obscured
(0013002B)	no more direct mode window ID's available
(0013002C)	process not found
(0013002D)	pad/stream operations not allowed while display acquired
(0013002E)	display already acquired
(0013002F)	display acquire timed out
(00130030)	bad tracking rectangle
(00130031)	tracking list full
	OS / volume manager:
(0014FFFF)	Warning: disk is write protected
(00140001)	entry directory problems on logical volume
(00140002)	unable to dismount the boot volume
(00140003)	logical volume is not mounted
(00140004)	entry directory is not on specified logical volume
(00140005)	physical volume replaced since mount
	OS / calendar manager:
(00150001)	invalid syntax for date or time specification
(00150002)	date or time specification invalid
(00150003)	an empty string was passed to a decode routine
(00150004)	timezone specified is unknown

(00150005)	invalid time-zone difference
	OS / cross-process debug manager:
(00160001)	locate target spans multiple objects
(00160002)	locate target spans discontiguous segments
(00160003)	invalid state argument
(00160005)	not a debugger
(00160006)	debugger not found
(00160007)	debugger table full
(00160008)	requested state inapplicable to machine type
(00160009)	already a debugger
(0016000A)	target process not found
(0016000B)	no event posted for target - not suspended
(0016000C)	invalid ec key
(0016000D)	locate target has variable mmu access
(0016000E)	state unavailable for this event
(0016000F)	invalid control/inquire option
	OS / deferred execution module manager:
(00170001)	no more deferred execution queue slots
(00170002)	datum too large for deferred execution
(00170003)	wired dxm helper not currently supported
	OS / level 2 eventcount manager:
(00180001)	internal table exhausted
(00180002)	internal error
(00180003)	asynchronous fault occurred while waiting
(00180004)	bad eventcount
(00180005)	unable to allocate level 1 eventcount
(00180006)	level 1 eventcount not allocated
	OS / level 2 process manager:
(00190001)	process not found
(00190002)	not a level two process
(00190003)	bad stack base
(00190004)	request is for current process
(00190005)	suspend request timed out
(00190006)	process not suspended
(00190007)	process already suspended
(00190008)	child process terminated
(00190009)	another fault is pending for this process
(0019000A)	invalid process name
(0019000B)	bad eventcount key
(0019000C)	attempt to complete vfork on non-vforked process
	OS / import/export manager:
(001A0001)	entry directory is not cataloged in the namespace
(001A0002)	files are locked on this volume
(001A0003)	specified entry directory not on this volume
(001A0004)	volume is not mounted
	OS / startup/shutdown:
(001B0001)	node ID mismatch
(001B0002)	checksumming already enabled

(001B0003)	no os paging file -- please run invol option 8
(001B0004)	no calendar on system -- please boot over network
	OS / vfmt:
(001C0001)	unterminated control string
(001C0002)	invalid control string
(001C0003)	too few arguments supplied for read/decode
(001C0004)	field width missing on "(" designator
(001C0005)	encountered end of string where more text expected
(001C0006)	encountered null token where numeric token expected
(001C0007)	non-numeric character found where numeric was expected
(001C0008)	sign encountered in unsigned field
(001C0009)	value out of range in text string
(001C000A)	character in text string does not match control string
(001C000B)	terminator in text string does not match specified terminator
	OS / circular buffer manager:
(001D0001)	invalid block size requested
(001D0002)	quit while waiting
(001D0003)	buffer wrap-around error
	OS / pbu manager:
(001E0001)	ddf is larger than one page
(001E0002)	ddf has wrong version
(001E0003)	invalid unit number in ddf
(001E0004)	invalid csr page address in ddf
(001E0005)	csr page is in use
(001E0006)	initialization routine not in library
(001E0007)	cleanup routine not in library
(001E0008)	interrupt library too large
(001E0009)	interrupt routine not in library
(001E000A)	pbu not present
(001E000B)	too many pbu manager pages wired
(001E000C)	invalid unit number
(001E000D)	unit in use
(001E000E)	unit not acquired
(001E000F)	unit already acquired
(001E0010)	bad parameter
(001E0011)	no room in iomap
(001E0012)	requested iomap in use
(001E0013)	iomap already allocated
(001E0014)	iomap not allocated
(001E0015)	invalid iova
(001E0016)	buffer too large
(001E0017)	buffer page not wired
(001E0018)	buffer not mapped
(001E0019)	page already wired
(001E001A)	page wired too many times
(001E001B)	page not wired
(001E001C)	reference to csr page caused bus timeout

(001E001D)	trap 6 executed outside of interrupt routine
(001E001E)	invalid trap 6 code
(001E001F)	invalid usp at trap 6
(001E0020)	protection violation
(001E0021)	unexpected interrupt from pbu device
(001E0022)	ddf has wrong file type
(001E0023)	too many wired pages
(001E0024)	csr not in device's csr page
(001E0025)	controller already mapped
(001E0026)	bad controller memory length
(001E0027)	bad buffer address
(001E0028)	interrupt library not found
(001E0029)	device library not found
(001E002A)	device is not a shared controller
(001E002B)	device not mapped
(001E002C)	pbu device got bus timeout on multibus
(001E002D)	all pbu units in use
(001E002E)	wrong version of /lib/pbulib in use
(001E002F)	interrupt level in use
(001E0030)	operation valid only for VME device
(001E0031)	physical address list too small
(001E0032)	function not supported for this device type
(001E0033)	illegal dma channel number
(001E0034)	bad dma direction specified
(001E0035)	requested dma channel in use
(001E0036)	requested dma channel not in use
(001E0037)	dma channel not at end of range
(001E0038)	no more eventcounts available
(001E0039)	eventcount not allocated to this unit
(001E003A)	unit already in use as a global device
(001E003B)	unit is publicly owned
(001E003C)	buffer pages not physically contiguous
(001E003D)	contiguous buffer not page aligned
	OS / line printer module:
(001F0001)	pna board not installed in system
(001F0002)	invalid string length
(001F0003)	invalid string termination
(001F0004)	line printer not acquired
(001F0005)	line printer already acquired
(001F0006)	internal error
(001F0007)	ppn list overflow - internal error
(001F0008)	line printer not assigned
(001F0009)	no line printer on system
	OS / OS info supplier:
(00200001)	array too small for complete table
	OS / badspot manager:
(00210001)	bad checksum in physical badspot block
(00210002)	bad count in physical badspot block
(00210003)	missing minus-one in physical badspot block
(00210004)	badspot list too small
(00210005)	no physical badspot blocks read or written
(00210006)	physical badspot list partially read or written
(00210007)	duplicate entry in badspot list

(00210008)	no physical badspot information on disk
(00210009)	bad daddr for lv label badspot extension block
(0021000A)	too many extensions to lv badspot list
(0021000B)	badspot extension uid <> logical volume uid

OS / magtape manager:

(0022FFFE)	warning: tape not at load-point
(0022FFFF)	warning: tape unit is offline
(00220001)	invalid mt unit number
(00220002)	invalid mode field
(00220003)	invalid buffer length
(00220004)	invalid parameter
(00220005)	no PNA board installed in system
(00220006)	magtape unit is not connected
(00220007)	magtape not acquired
(00220008)	magtape unit is not ready
(00220009)	unit will not fit thru 25" hatch
(0022000A)	magtape unit in use
(0022000B)	magtape not initialized
(0022000C)	magtape already acquired
(0022000D)	invalid option
(0022000E)	too many outstanding operations
(0022000F)	invalid buffer address
(00220010)	invalid count for erase or space operation
(00220011)	tape drive is hung
(00220012)	ppn list overflow - internal error
(00220013)	config page in use - internal error
(00220014)	release problems - internal error
(00220015)	unexpected interrupt
(00220016)	operation attempted before waiting
(00220017)	wait attempted before go issued
(00220018)	go command issued while not in batch mode
(00220019)	header or buffer misalignment on chained r/w
(0022001A)	user quit while in mt_$wait
(0022001B)	timeout during wait or release
(0022001C)	header buffer not on header page
(0022001D)	no room from mt_$write - internal error
(0022001E)	info array (passed to mt_$wait) too small
(0022001F)	too many pages wired
(00220020)	too many pbu devices in use
(00220021)	buffer already wired
(00220022)	buffer not wired

OS / ACL manager:

(00230001)	no right to perform operation
(00230002)	insufficient rights to perform operation
(00230003)	exit_super called more often than enter_super
(00230004)	wrong type - operation illegal on system objects
(00230005)	entry already exists
(00230006)	ACL is remote
(00230007)	ACL is on different volume than object
(00230008)	ACL protects wrong type of object
(00230009)	insufficient address space to open ACL
(0023000A)	Unused ACL status code
(0023000B)	no entry - entry number too large

(0023000C)	image buffer too small or incorrect size
(0023000D)	ACL object not found
(0023000E)	ACL would be unchangeable
(0023000F)	object may not be readable by backup procedure
(00230010)	no right to set subsystem data or subsystem manager
(00230011)	project list is full - no more entries may be added
(00230012)	project list is too big - it cannot be added to object
(00230013)	ACL is full - no more entries may be added
	OS / PEB manager:
(00240001)	fpu is hung
(00240002)	PEB interrupt
(00240003)	floating point overflow
(00240004)	floating point underflow
(00240005)	divide by zero
(00240006)	floating point loss of significance
(00240007)	floating point hardware error
(00240008)	attempted use of unimplemented opcode
(00240009)	wcs verify failed
	OS / network logging manager:
(00250001)	ppn list overflow
	OS / color display manager:
(00260001)	illegal caller
(00260002)	too many wired pages
(00260003)	virtual address not page aligned in color_$map
(00260004)	pages unmapped out of order
(00260005)	parameter value out of range
(00260006)	color display not available
(00260007)	instruction queue done wait timed out
	OS / vme bus manager:
(00270001)	undefined vme interrupt
(00270002)	vme bus error
	OS / cartridge tape manager:
(0028FFFA)	warning: tape in write mode
(0028FFFB)	warning: tape in read mode
(0028FFFC)	warning: tape not at load-point
(0028FFFD)	tape at load point
(0028FFFE)	warning: tape unit is offline
(0028FFFF)	tape power on/reset
(00280001)	invalid ct unit number
(00280002)	unit not acquired
(00280003)	unit already acquired
(00280004)	unit in use
(00280005)	no tape controller on system
(00280006)	invalid buffer length
(00280007)	bad buffer alignment
(00280008)	invalid buffer address
(00280009)	unrecognized action type
(0028000A)	invalid operation count

(0028000B)	unit not ready
(0028000C)	unexpected ct interrupt
(0028000D)	quit waiting for i/o
(0028000E)	timeout waiting for i/o
(0028000F)	too many wired pages
(00280010)	no cartridge in drive
(00280011)	drive does not exist
(00280012)	tape is write protected
(00280013)	end of tape
(00280014)	read/write abort
(00280015)	read block error
(00280016)	read filler error
(00280017)	read no data
(00280018)	read no data and end of tape
(00280019)	read no data and load point
(0028001A)	filemark detected
(0028001B)	illegal drive command
(0028001C)	marginal block detected
(0028001D)	unrecognized drive status
(0028001E)	dma not at end of range
(0028001F)	dma underrun/overrun
(00280020)	memory parity error during dma
(00280021)	illegal controller command
(00280022)	controller timeout
(00280023)	controller diagnostic failed
(00280024)	unrecognized controller status
(00280025)	operation already in progress
(00280026)	operation not in progress
	OS / msg manager:
(00290001)	socket out of range
(00290002)	to deep
(00290003)	socket error
(00290004)	no more sockets
(00290005)	no owner
(00290006)	too_much_data
(00290007)	socket empty
(00290008)	socket in use
(00290009)	time out
(0029000A)	quit fault
	OS / symbolic link manager:
(002A0001)	file not symbolic link type
(002A0002)	bad symbolic link file
	OS / internet routing:
(002B0001)	network port not open
(002B0002)	buffer queue for user port is full
(002B0003)	unknown network port
(002B0004)	can not create/delete that port type
(002B0005)	max number of ports already open
(002B0006)	routing service type not recognized
(002B0007)	port belongs to another process
(002B0008)	routing through-traffic queue overflow
(002B0009)	operation not legal on this port type
(002B000A)	unknown network device type

(002B000B)	no more buffer queues for user networks
(002B000C)	user network checksum failed
(002B000D)	bad packet length from user network
(002B000E)	unable to create through-traffic queue
(002B000F)	max number of USER ports already open
(002B0010)	bad request type asking for service change
(002B0011)	routing not allowed at port with 0 network ID

OS / internet interface controller:

(002C0001)	IIC: dma got multibus read timeout error
(002C0002)	IIC: not initialized prior to operation
(002C0003)	IIC: trasnsmitter underrun error
(002C0004)	IIC: undocumented interrupt raised
(002C0005)	IIC: hardware reset operation timed out
(002C0006)	IIC: self test failure reported by board init
(002C0007)	device already acquired
(002C0008)	device not acquired
(002C0009)	operation aborted
(002C000A)	device not in system
(002C000B)	remote device not acquired
(002C000C)	could get expected packet from receive socket
(002C000D)	could not allocate receive socket for device
(002C000E)	never got expected command completion interrupt
(002C000F)	wrong board revision level

OS / graphics processor manager:

(002D0001)	device not present in system
(002D0002)	device not available
(002D0003)	package not initialized
(002D0004)	package already initialized
(002D0005)	device not ready for PIO
(002D0006)	device timeout
(002D0007)	wait terminated by process fault
(002D0008)	error condition reported by GPU
(002D0009)	page fault interrupt
(002D000A)	illegal values for physical page use limits
(002D000B)	Programmed I/O command error
(002D000C)	DMA command execution error
(002D000D)	buffer already wired
(002D000E)	buffer too large
(002D000F)	no GPU microcode loaded
(002D0010)	error reported by draw processor

OS / DMA manager:

(002E0001)	illegal channel
(002E0002)	illegal byte count
(002E0003)	channel in use
(002E0004)	channel not allocated for operation
(002E0005)	operation did not finish

OS / Ethernet:

(002F0001)	internal driver error
(002F0002)	feature is not implemented
(002F0003)	driver version mismatch
(002F0004)	device is off-line
(002F0005)	device is already on-line

(002F0006)	adapter hardware error
(002F0007)	transmit operation failed
(002F0008)	invalid unit number
(002F0009)	illegal packet length
(002F000A)	invalid statistics block
(002F000B)	packet type is already in use
(002F000C)	no channels are available
(002F000D)	no packet available for receive
(002F000E)	invalid packet type
(002F000F)	channel is not open
	OS / audit trail manager:
(00300001)	could not create Event Server Process

BOOT ERRORS (PROM)

error: boot not found - The SYSBOOT read from records 2 thru B did not have a good boot header.

```
disk init error  <SC>  <RCD>  <UNIT>  <W/F/S/C>
disk read error  <SC>  <RCD>  <UNIT>  <W/F/S/C>
```

SC = Status Code
RCD = Record Address
Unit = Disk Unit No.
W/F/S/C = Winchester/Floppy/SMD/Cartridge Tape

Winchester Status Codes:

1	- not responding
2	- not ready
11	- seek not complete
12	- CRC, timeout, buserr, overrun
13	- drive faults

Floppy Status Codes:

1	- wrong no. of status bytes
2	- seek not complete
3	- equipment check
11	- insufficient rights
12	- seek not complete
13	- equipment check
14	- bad seek
15	- insufficient status
16	- abnormal termination
17	- '' ''
18	- device not ready
19	- CRC error

Disk/Tape Status Codes:

11	- controller diagnostic failed
12	- contrlloer timed out
13	- illegal controller command
15	- memory parity during dma
16	- dma overrun/underrun
17	- dma not at end of range
1E	- disk still busy
1F	- controller still busy
21	- seek did not complete
22	- write fault
23	- unit not present
24	- sector not found
25	- no index pulse
26	- drive not ready
27	- no track 0 on restore
28	- address mark not found
29	- ECC error in sector ID field
30	- illegal tape command
31	- filemark encountered
32	- read error - no data and BOM
33	- read error - no data and BOM
34	- read error - no data
35	- read error - filler block transfer
36	- read error - bad block transferred
37	- read or write abort
38	- end of media
39	- drive not present
3A	- no cartridge in drive
FF	- timeout wating for controller done

BOOT PROM DIAGNOSTIC ERROR CODES (EXCEPT 020 MACHINES)

```
error:
<test no.><detected at><object addr><data is><data sb>
```

Steady State

0	first instruction at "init"
1	memory passed tests at init
2	state saved
3	parity cleared in first 2 pages
4	sio's have been initialized
5	clr_disp called, returned
6	disp_init called, returned
7	display init got bus error
8	we're in service mode
9	character received from keyboard
A	character received from line 1
B	character received from line 2
C	just printed MD's banner msg
D	ptt enabled (map routine)
E	mmu initialized (map routine)
F	mmu enabled (map routine)

```
Digit 1 Digit 2
0 ------ PROM Wait Conditions
      2      booting: waiting for disk I/O to complete
      3      booting: waiting for network transmit to complete
      4      booting: waiting for partner to respond
      5      booting: waiting for network receive to complete
      C      waiting for command input (service mode only)
1 ------ DIAGNOSTIC 1: Validate PROM Checksum
      1     PROM checksum did not match calculated checksum
2 ------ DIAGNOSTIC 2: Test PFT
      1     verify failed with data = 0000... (first time)
      2     verify failed with data = FFFF...
      3     verify failed with data = AAAA...
      4     verify failed with data = 0000... (second time)
      5     verify failed for address uniqueness
3 ------ DIAGNOSTIC 3: Test PTT
      1     verify failed with data = 0000... (first time)
      2     verify failed with data = FFFF...
      3     verify failed with data = AAAA...
      4     verify failed with data = 0000... (second time)
      5     verify failed for address uniqueness
4 ------ DIAGNOSTIC 4: Test IOMAP
      1     verify failed with data = 0000... (first time)
      2     verify failed with data = FFFF...
      3     verify failed with data = AAAA...
      4     verify failed with data = 5555...
      5     verify failed with data = 0000... (second time)
      6     verify failed for address uniqueness
5 ------ DIAGNOSTIC 5: Test PFT, PTT IOMAP Interaction
      1     writing PFT affected PTT (should be 0's and wasn't)
      2     writing IOMAP affected PTT (should be 0's and wasn't)
      3     writing PFT affected PTT (should be 1's and wasn't)
```

```
       4      writing IOMAP affected PTT (should be 1's and wasn't)
 6 ------ DIAGNOSTIC 6: Test Physical/Virtual Memory
       1      verify failed during byte test with 0's
       2      verify failed during word test with 1's
       3      verify failed during long word test with 5's
       4      verify failed during long word test with A's
       5      verify failed during long word test with addresses
 7 ------ DIAGNOSTIC 7: Test Virtual Memory
       1     verify failed during long word test with addresses
 8 ------ DIAGNOSTIC 8: Retest Memory in Physical Mode
       1     verify failed during long word test with addresses
 9 ------ DIAGNOSTIC 9: Timers, MULTIBUS, Calendar
       1      timer  1 failed
       2      timer  2 failed
       3      timer  3 failed
       4      MULTIBUS map test failed pattern test
       5      calendar hardware clock not incrementing seconds
       6      DMA continuity test for channel 0 failed
       7      DMA continuity test for channel 1 failed
       8      DMA continuity test for channel 2 failed
       9      DMA continuity test for channel 3 failed
       A      DMA test with all channels activated failed
 4 ------
       7      MULTIBUS loopback test failed
       8      MULTIBUS timeout test failed
 9 ------ DIAGNOSTIC 10: Test Ring Board
       B      ring board loopback test failed
 4 ------ DIAGNOSTIC 11: Test VME Interface
       9      VME register test failed
       A      open VMEbus test failed
       B      bus arbiter logic test failed
       C      VME-BPORT logic test failed
 A ------ DISK ERROR
       1       not responding
       2       not ready
       B       seek not complete
       C       CRC, timeout, bus error, overrun
       D       drive fault
 B ------ NETWORK ERROR
 C ------ SYSBOOT ERRORS
       1       bad command line (cmd not found, missing filename)
       2       unable to read physical volume label
       3       volume "n" not found
       4       unable to read logical volume label
       5       salvaging boot volume
       6       auto salvage failed
       7       unable to read root directory
       8       SAUn not found in root directory
       9       SAUn uid not found
       A       unable to restore SAUn directory
       B       <program> not found
       C       <program> uid not found
       D       <program> unreadable
       E       <program> not a file
       F       <program> has wrong machine type
 D ------ NETBOOT ERRORS
```

	1	bad command line (cmd not found, missing filename)
	2	ring initialization failed
	3	unable to send load request
	4	no response to load request
	5	unexpected packet (get_reply)
	6	<program> not found
	7	bad pathname
	8	insufficient rights
	9	<program> has wrong machine type
	A	(other) non-zero status from netman
	B	unable to send uid request
	C	no response to uid request
E ------ RESERVED		
F ------ AEGIS CRASH		
	1	aegis crash
	F	normal shutdown complete

BOOT PROM DIAGNOSTIC ERROR CODES (DN330)

Upper LEDs	Lower LEDs	Test Type - Board - Test
0	0	Core - CPU - Turn off LEDs
	1	Core - CPU - Checksum PROM
	2	Core - CPU - Critical Memory
	3	Core - CPU - Bus Error
	4	Core - CPU - Enable Instruction Cache
	5	Console Integrity - CPU - SBUS Loopback
	6	Console Integrity - CPU - Keyboard
	7	Load Path - CPU - Parity Memory
	8	Load Path - CPU - DMA
	9	Non-critical HW - CPU - Timers
	A	Non-critical HW - CPU - MMU
	B	Non-critical HW - CPU - Coprocessor
	C	Non-critical HW - CPU - Speaker
1	0	Console Integrity - Display - Existence
	1	Console Integrity - Display - Fill mode
	2	Console Integrity - Display - Display memory
2	0	Load Path - Mem Ext - Parity Memory
3	0	Load Path - Win Disk - Existence
	1	Load Path - Win Disk - Initialize
	2	Load Path - Win Disk - Read block 0
	3	Non-critical HW - Win Disk - Calendar
4	0	Load Path - Ring - Single node transmit
-	F	Software detected error
-	Arrow	Unexpected exception

BOOT PROM DIAGNOSTIC ERROR CODES (DSP90)

Upper LEDs	Lower LEDs	Test Type - Board - Test
0	0	Core - CPU - Turn off LEDs
	1	Core - CPU - Checksum PROM
	2	Core - CPU - Critical Memory
	3	Core - CPU - Bus Error
	4	Core - CPU - Enable Instruction Cache
	5	Console Integrity - CPU - SBUS Loopback
	6	Load Path - CPU - Parity Memory
	7	Load Path - CPU - DMA
	8	Non-critical HW - CPU - Timers
	9	Non-critical HW - CPU - MMU
	A	Non-critical HW - CPU - Coprocessor
	B	Non-critical HW - CPU - Speaker
1	0	Load Path - Mem Ext - Parity Memory
2	0	Load Path - CPU Ext - MULTIBUS
	1	Non-critical HW - CPU Ext - Calendar
3	0	Load Path - SMD - Existence
	1	Load Path - SMD - Initialize
	2	Load Path - SMD - Read block 0
4	0	Load Path - FSD - Existence
	1	Load Path - FSD - Controller diagnostics
5	0	Load Path - Ring - Single node transmit
-	F	Software detected error
-	Arrow	Unexpected exception

BOOT PROM DIAGNOSTIC ERROR CODES (DN5xx-T)

Lower LEDs	Middle LEDs	Test Type - Board - Test
0	0	Core - CPU - Turn off LEDs
	1	Core - CPU - Checksum PROM
	2	Core - CPU - Data Cache Data
	3	Core - CPU - Data Cache Address
	4	Core - CPU - Data Cache Byte Steering
	5	Core - CPU - Stack Initialization
	6	Core - CPU - Greeting on SIO
	7	Console Integrity - CPU - Registers
	8	Console Integrity - CPU - VME bus
	9	Console Integrity - CPU - Bus Arbiter Logic
	A	Console Integrity - CPU - Bus Interrupt
	B	Console Integrity - CPU - Bus Error
1	0	Non-critical HW - CPU - Timers
	1	Non-critical HW - CPU - Calendar
	2	Non-critical HW - CPU - SIOus arbiter logic
	3	Non-critical HW - CPU - MMU/Data cache

	4	Non-critical HW - CPU - Coprocessor (68881)
	5	Non-critical HW - CPU - Beep
2	0	Console Integrity - Display - Existence
	1	Console Integrity - Display - Display Self Test
	2	Console Integrity - Display - Clear Display
	3	Console Integrity - Display - Display memory
	4	Console Integrity - Display - Load Font
3	0	Console Integrity - Memory - Memory Existence
	1	Console Integrity - Memory - Memory Data
	2	Console Integrity - memory - Memory Address
4	0	Load Path - Mass Stor - Existence
	1	Load Path - Mass Stor - Self Test
	2	Load Path - Mass Stor - Extended Self Tests
5	0	Load Path - MULTIBUS - Existence
	1	Load Path - MULTIBUS - Registers
	2	Load Path - MULTIBUS - I/O map
	3	Load Path - MULTIBUS - Loopback
6	0	Load Path - SMD - Existence
	1	Load Path - SMD - Initialize
	2	Load Path - SMD - Read block 0
7	0	Load Path - FSD - Existence
	1	Load path - FSD - Controller diagnostics
8	0	Console Integrity - Ring - Ring Test
	1	Load Path - Ring - Single node transmit
9	0	Non-critical HW - FPX - Existence
	1	Non-critical HW - FPX - Interface
A	0	Non-critical HW - GPU - Existence
	1	Non-critical HW - GPU - Reset
	2	Non-critical HW - GPU - Self Test
B	0	Console Integrity - Keyboard - Model II Existence

BOOT PROM DIAGNOSTIC ERROR CODES (DN560/570/580)

Upper LEDs	Lower LEDs	Test Type - Board - Test
0	0	Core - CPU - Turn off LEDs
	1	Core - CPU - Checksum PROM
	2	Core - CPU - Critical Memory
	3	Core - CPU - Bus Error
	4	Core - CPU - Enable Instruction Cache
	5	Console Integrity - CPU - SBUS Loopback
	6	Console Integrity - CPU - Keyboard
	7	Load Path - CPU - Parity Memory
	8	Non-critical HW - CPU - Timers
	9	Non-critical HW - CPU - MMU
	A	Non-critical HW - CPU - Coprocessor
	B	Non-critical HW - CPU - Speaker
1	0	Console Integrity - VME - Registers
	1	Console Integrity - VME - Open VME bus
	2	Console Integrity - VME - Bus arbiter logic
	3	Console Integrity - VME - VME to B-port logic
2	0	Console Integrity - Display - Existence
	1	Console Integrity - Display - Load microcode
	2	Console Integrity - Display - Run microcode

	3	Console Integrity - Display - Display memory
	4	Console Integrity - Display - Clear the screen
	5	Console Integrity - Display - Load the fonts
3	0	Load Path - Mem Exp - Parity Memory
4	0	Load Path - Win Disk - Disk Controller diagnostic
	1	Non-critical HW - Win Disk - Calendar
5	0	Load Path - MULTIBUS - Existence
	1	Load Path - MULTIBUS - Registers
	2	Load Path - MULTIBUS - I/O map
	3	Load Path - MULTIBUS - Loopback
6	0	Load Path - SMD - Existence
	1	Load Path - SMD - Initialize
	2	Load Path - SMD - Read block 0
7	0	Load Path - FSD - Existence
	1	Load path - FSD - Controller diagnostics
8	0	Load Path - Ring - Single node transmit
-	F	Software detected error
-	Arrow	Unexpected exception

BOOT PROM DIAGNOSTIC ERROR CODES (DN3000)

Ext. LEDs	Int. LEDs	Test
0	0	Turn off LEDs
	1	Checksum PROM
	2	Refresh circuitry
	3	Bus Error
	4	Enable Instruction Cache
	5	Keyboard SIO
	6	Parity circuitry
	7	MMU
	8	Interrupt
	9	Timers
	A	DMA pare register
	B	DMA controller 1
	C	DMA controller 2
	D	Calendar and configuration
1	0	Critical memory (Megabyte 1)
2	1	Display controller existence
	2	8255A test
	3	Pixel test
	4	Horizontal sync counter
	5	Vertical sync counter
	6	Frame buffer
	7	Video output
	8	Red, blue high level output
	9	Green output
	A	LUT red, blue high level output
	B	BLTs
	C	ROP logic
	F	A/D converter error
3	0	Keyboard self-test
	1	Keyboard speaker

4	0	Megabyte 2
5	0	Megabyte 3
6	0	Megabyte 4
7	0	Disk existence
	1	Disk controller self-test
	2	Read disk block 0
8	0	Digital loopback ring controller
	1	Analog loopback ring controller

MNEMONIC DEBUGGER ERROR CODES (PROM)

Printed on system crash or other entry to MD. See also System Crash Analysis under Operational Procedures.
For DNx60 CPIO, mnemonic debugger codes are printed just like non-DNx60, except the qualifier "(CPIO)" is also printed. For DNx60 CPU, the qualifier
"(CPU)" is also printed to identify the environments.

A	<PC> <SR> <IR> <FA> <FC> <PC> <Contents>	- Address Error
B	<PC> <SR> <IR> <FA> <FC> <PC> <Contents>	- Bus Error
C	<PC> <SR> <PC> <Contents>	- Floating-Point Coprocessor Error
E		- Operational Error
F	<PC> <SR> <PC> <Contents>	- Invalid stack format
I	<PC> <SR> <PC> <Contents>	- Unexpected Interrupt
J		- Spurious Interrupt
O	<PC> <SR> <FW> <FA> <PC> <Contents>	- floating point trap (DNx60 CPU)
P		- Parity error (DN300 only)
S	<PC> <SR> <PC> <Contents>	- Trap instruction or breakpoint
T	<PC> <SR> <PC> <contents>	- Trace trap
U	<PC> <SR> <PC> <Contents>	- Unimp inst trap
V	<PC> <SR> <FW> <FA> <PC> <Contents>	- access violation (DNx60 CPU)
W	<PC> <SR> <FW> <FA> <PC> <Contents>	- region fault (DNx60 CPU)
X	<PC> <SR> <FW> <FA> <PC> <Contents>	- segment fault (DNx60 CPU)
Y	<PC> <SR> <FW> <FA> <PC> <Contents>	- page fault (DNx60 CPU)
Z	<PC> <SR> <PC> <Contents>	- Divide by zero trap

PC = Program Counter
SR = Status Register
IR = Instruction Register
FA = Fault Address
FC = Fault Code
FW = Format Word identifies the trap vector

```
            System    User
          +--------+--------+
     SR:  |T-S--III|---XNZVC|
          +--------+--------+
           T -   Trace Mode        X - Extend
           S -   Supervisor Mode   N - Negative
           III - Interrupt Mask    Z - Zero
                 000 - enabled     O - Overflow
                 111 - disabled    C - Carry

           15             0
          +----------------+
    FC:   |-----------RNFFF|
          +----------------+
          R   = 1 => Read operation
          N   = 1 => Not instruction reference
          FFF - Function code:
                001  User data
                010  User program
                101  Supervisor data
                110  Supervisor program
                111  Interrupt acknowledge
```

SYSBOOT ERROR CODES

```
    boot error: unable to read pv_label
        ''      volume "N" not found
        ''      unable to read lv_label
        ''        ''   ''  ''  root_dir
        ''      SAU not found in root_dir
        ''      SAU uid not found
        ''      unable to restore SAU_dir
        ''      "FILENAME" not found
        ''          ''      uid not found
        ''          ''      unreadable
        ''          ''      not a file
        ''      missing file name
```

CHAPTER 5

SYSTEM DEBUGGING

BOOT SHELL COMMANDS

```
CF      [<pathname> | -E]           run/end command file
CHN     <pathname> <compname>       change name
CRD     <pathname>                  create directory
CRF     <pathname>                  create file
CRL     <pathname> <linkname>       add link
CTNODE  <leaf> <node_id>            add node to local copy of root
CTOB    <pathname> <uidhi> <uidlo>  catalog name with specified uid
DEBUG   <value>                     enable/disable debug mode
DLF     <pathname>                  delete file
DLL     <linkname>                  drop linkname
DM                                  invoke the dispaly manager
DMTVOL  {W | S | F} <lvno> [<pathname>] dismount a logical volume
GLOB                                list installed globals
GO                                  load as if in normal mode
H                                   prints this text
IN      <pathname> [-D] [-S | -NS]  invoke loader to install named
                                    file
LD      [<pathname>] [-A [-D]] [-U] list directory
LI      <address>                   set display lites address
LO      <pathname> [-D] [-S | -NS]  invoke loader with given file
MA      <pathname> [<l> <sz>] [-E]  map file
MTVOL   {W | S | F} <lvno> [<pathname>] mount a logical volume
ND      <pathname>                  set naming directory
REL     [-A]                        release proc-mgr assigned storage
SH                                  load single process shell
SHUT                                shutdown system
SPM                                 load DSP80 server process manager
STCODE  <status-code>               print textual definition of
                                    status code
TB                                  stack trace back
TI      {-ON | -OFF}                enable/disable timer
TR      <pathname> <sz>             truncate raw data file to
                                    given size (hex)
UCTNODE <leaf>                      drop node from local copy of root
UCTOB   <pathname>                  un-catalog pathname from
                                    namespace
UMA     {<pathname> | <l> <sz>}     unmap file by name or addr/size
WD      <pathname>                  set working directory

Key:      <l>     :=  low address
          <sz>    :=  size
          <type>  :=  { nil, rec, hdru, obj, dev, pad, undef, uasc,
                        mt, boot }
```

NOTE: Basic MD commands such as access location and dump memory are also available in the Boot Shell environment.

DEBUG COMMAND EXTENSIONS

The following commands and other items are available in the standard DEBUG, but are not advertised to the public.

Apollo DEBUG Commands

- FPREGS -- Display the floating-point registers.
- REGS -- Display the registers of the target program.
- DB -- Invoke the machine-level debugger DB (see below).

Options to DEBUG Commands

- -CDB -- This option is useful to debug C code in the C library. It prevents DEBUG from running to the "main" program when DEBUG is invoked.
- -DDD -- This option is useful only when debugging DEBUG from another DEBUG. It must be given to the target DEBUG only. When it is given, the target DEBUG will return from the fault handler with the "continue fault handling" state, so that the master DEBUG can catch breakpoints and signal steps.

Miscellaneous

More help is available on DEBUG commands in this file:

```
//us/latest/sysx/help/debug.hlp
```

Here is some useful information which IS PUBLIC, which people often forget how to do.

There are three debugger names that DEBUG looks for. If it finds them, it does some special things:

```
'cr             (macro name; made with MACRO command)
'max_array_dim  (debugger variable; made with SET command)
'max_var_len    (debugger variable; made with SET command)
```

Note that "help examine" explains the 'max_... names, and "help macro" explains the 'cr name.

DB (MACHINE LEVEL DEBUGGER)

You enter the DB debugger by entering the DB command from within a shell. The formats of the internal DB commands follow.

```
dl                   invoke emt
ef                   enter fim
fp{.s|.d} [< >]      show any or all floating point registers
                     (0-f)
fpc [<value>]        set/show the floating point control
                     register
fps [<value>]        set/show the floating point status
                     register
help
in <path>            install library
lo <path>            load program
ma <path> [<start>]  [-ex] [-r] [-nl]  map a file (starting
                     location)
pc                   display current pc
fa                   display last fault address
fc                   display last fault code
sh                   invoke new shell
ss                   single step
tb                   traceback current stack

crash analysis    commands:

a7 <value>           set a7 in dump (dflt is from 100000)
a{b|w|l}[e] <sym>    access via symbol name
am <path>            load Aegis Map
as [<asid>]          set/display current asid
aote <addr>|<aotex>  print contents of aote
aste <addr>|<astex>  print contents of aste
caot                 check aot/ast for consistency of links
clf [<addr>]         cherchez la (fault) frame (starts w/(A7)
                     if no addr)
da [<clockh>]        display date (build_$time)
dcte [<index>]       display dcte (all if no index)
df [<addr>]          display diagnostic fault frame at addr
                     (search)
dml                  dump memory log
dp [<pid>]           display pcb (first ten if no pid entered)
dpt                  disable ptt (remove from address space)
dr                   display registers at crash
ds                   display disk statistics
dsmx                 dump Xylogics queued i/o data structures
dsys                 info unique to current node type
dv <addr>            convert db address to virtual address
dvt                  print disk volume table
dwin                 dump WIN_1 queued i/o data structures
ept                  enable ptt (map into address space)
ff <addr>            print fault frame at addr
fnf [<addr>]         try to find stack frame in
```

```
                    addr - addr+1024
gd <unit>           get (pbu) dcte
ha <hi> <lo> | <addr> hash uid to ast
le                  list system error log
lvl <addr>          print lv label at addr
m [10 | 20]         enter mapped mode (select 010/020)
mm <addr>|<ppn>     print mmap entry
mr                  print mem_rec
ms <args>           mapped search (just like md's 's')
mst [<asid>|<msteaddr>] print mst for an asid (0 for gbl, omit
                    for curr)
mste <addr>         print mste for a virtual address in
                    current asid
mstp [<asid>]       print used mst entries for asid
p                   enter physical (normal) mode
pd <addr>           convert physical address to db address
pf <ppn>|<addr>     display pft entry
ph <addr>           display network packet header
pt <pttx>           display ptt entry
pv <addr>           convert ppn to virtual address
pvl <addr>          print pv label at addr
r                   enter real mode
rars [<asid>]       display rars <current asid>
regs                print locations for all A registers
rl [anything]       display ready list [just check order]
skd [<socket >]     dump socket (all)
sks [<socket >]     display socket status (all)
st                  display status at crash
tp                  display trap page
ts <pid or addr>    traceback stack
uid <hi> <lo> | <addr> interpret uid
vb <addr>           print vtoc block at <addr>
vd <addr>           convert virtual address to db address
ve <addr>           print vtoce at <addr>
vm                  verify mmu
vme [<addr>]        display saved vme bus error info
vp <addr>           convert virtual address to ppn
vr <rfc_pathname>   verify rfc file against dump
vv <addr> <data>    verify vmtest page
wh[i][p|d|e] <sym or addr> look up [init][proc|data|ecb] or
                    address in aegis map
wsl                 display state of working set lists
wt                  display last wait start times
```

Exit the DB debugger by entering the q command in response to the ! prompt as follows:

```
     ! q
```

For a description of crash analysis, refer to /os/doc/db_crash_analysis.

DB also supports all commands from the MD.

Lights Program

Execute the lights program to show network status from within the DB debugger as follows:

```
!LI  0FF9C12  (in  DN400/420)  {virtual   address   of
receive register}
!LI 0FF9C02 (in DN300/320)
or
!LI OFFF9C12 (in DN460)
```

Do not execute the lights program from a color node (unless it is a DN3000)! It will crash the node!

The transmit and receive status registers are displayed at the bottom of the screen. These register are described in chapters 7, 8, and 9. To use the lights program do the following:

- o Exit from DB with the q command.
- o Execute NETSTAT shell commands to send tokens across the network.
- o Study the status bits in the receive status register.

To exit from the lights program execute the following commands:

```
$ DB        {to re-enter DB environment}
! LI 0      {Provide address of zero to Lights program}
! q         {exit from DB environment}
<ctrl> F    {remove Lights from bottom of screen}
```

MNEMONIC DEBUGGER (PROM)

Commands:

+ = not in DNx60
- = DNx60 only
* = DN3000 only
= DNx60 and DN3000 only

+ A <location>	Access location
* AR	Access Control Register
- AS	Display current ASID
B <location>	Breakpoint
+ C <start> <end> <target>	Copy Memory
+ CA <start>	CALL Subroutine
# CB <location>	Clear breakpoints
+ D <start> <end> <items/line>	Dump Memory
+ DI <type><unit> <log vol>	Define Disk
+ DL	Down-line Loader
- DR	Dump registers
* DU	Dump system
EX <filename>	Load and Execute File
EY <filename>	Load and Execute File with trap after load
+ F <start> <end> <word>	Fill Memory
* FO	Force load
+ G <location>	Jump to Location
# H	Help
# IC	Enable/disable/show Instruction cache
+ LD	Lists SAUn Directories
+ LO <filename>	Load File
+ M	Map Address Space
+ P	Unmap Address Space
# PV	PA-to-VA
+ RE	Reset System
- RR	Access region registers
+ S <start> <end> <value> <mask>	Search Memory
+ SH <0-3>	Spindown Winchester
SK	Select keyboard
# SS	Single step
+ TE	Run boot PROM diagnostics
+ V <start> <end> <target>	Verify Memory
# VP	VA-to-PA
# XD	XON/XOFF disable
# XE	XON/XOFF enable

Commands valid in DNx60 CPU only:

DC	Enable/disable/show data cache
GB	Go Back to CPIO environment, halt CPU
FP	Access Floating Point Registers

Commands valid in DNx60 CPIO only:
Those marked with an asterisk require the micro exec to be loaded and/or running.

UC	Clock step CPU
*US	Micro step <cnt <uaddr>>
*UT	Micro trace <cnt <uaddr>>
UH	Halt the CPU and CPIO becomes master
UR	Run micro code <from uaddr>
UX	Reset micro machine
*UU	Micro trap to the micro executive
*UI	IPL micro machine at uaddr 0
UP	Display or set micro pc
UA	Load microcode file set
UL	Load one microcode file by name
UF	Set or clear microcode loaded flag
*MG	Start the CPU <at macro addr> and CPIO becomes slave
*MR	Start the CPU <at macro addr> and CPIO dies
*MS	Macro step the CPU <cnt <addr>>
*MT	Macro trace the CPU <cnt <addr>>
MX	Load and execute program using CPIO
GF	Go Forward to CPU with reset exception, halt CPIO
FR	Fill wcs with freeze micro instructions
*LP	Set loop mode for uexec commands
MM	Set CPIO in master mode
SM	Set CPIO in slave mode
DM	Set CPIO in dead mode
MO	Display CPIO master/slave mode
DS	Dump CPU state

A [<size_spec>] <location>[<base_spec>]

Accesses <location> and prints address and contents according to <size_spec> and <base_spec>.

AR <ctl_reg>

Access certain system control registers.

ctl_reg:
TC = MMU translation control
RP = MMU root pointer
DFC = CPU Destination function code
SFC = CPU Source function code
CACR = CPU Cache control
CAAR = CPU Cache address

B [<location>]

Sets/clears the breakpoint at the location specified. Breakpoint is not inserted until G command. Previous instruction is reinstalled on breakpoint entry or vector entry. For older (reverse-mapped) systems, only one breakpoint may be set at a time, and it must be cleared or moved before continuing after breaking.

C <start> <end> <target>

Copies memory defined by the bounds <start> to <end> onto memory starting at <target> through <target>+<end>-<start>.

CA <start>

Calls the subroutine which starts at <start>. All registers saved from the last entry except A0 are restored immediately prior to the call.

CB [<location>]

Clears the breakpoint at <location>, or all the breakpoints set, it you omit the argument. An error message appears if you attempt to clear breakpoints that are not set.

D [<size_spec>] <start> <end>

<items_per_line>[<base_spec>]

Dumps memory defined by the bounds <start> to <end> onto the terminal printing address followed by specified <items_per_line>. The default is one per line. The <size_spec> controls the item-size to be dumped: byte, word, long, instruction.

DI <W>|<F>|<N> [<nn>}|<S> <0-3> <1-10>

Disk defines the boot device: Winchester, Floppy, Node (nn), Storage Module unit 0-3, and logical volume 1-10. Defaults are: W, 0, 1.

DL

Transfers control to the down-line loader.

DS

Dump the CPU state. Valid only from the CPIO environment. It is useful if micromachine freezes (UPCxxx).

DU

This command initiates the system dump procedure. Requires operating system to have been booted (to provide dump address and routine). Precede with the device specification (DI N <nodeid>, .DI F or DI C), default is DI F. Will dump to multiple floppies.

EX <filename>

Execute restores the named file from the "SAU" directory of the boot device and transfers control to it. After the restore is complete, the LOW, HIGH, and START addresses are displayed.

EY <filename>

Just like EX except:

- o Just before passing control to the program, the MD will trap, and you can patch the program. Type "G", "G *+2" to continue.

- o If you are executing AEGIS, AEGIS will trap again after establishing the OS mapping and before calling OS_$INIT. AEGIS can then be patched or examined using the virtual addresses AEGIS.MAP in the appropriate SAU directory. Type "G", "G *+2" to continue bringing up AEGIS.

NOTE: Older boot PROMs may not support the EY command.

F <start> <end> [<word>]

Fills memory defined by bounds <start> to <end> with a word value <word>.

FO

This command initiates a force load sequence.

G [<location>]

Jumps to <location> after inserting breakpoint (if any), restoring all registers and SR.

GB

Go back (return control) to the boot processor. This is valid only in the main processor environment.

GF

GF start the main processor with a reset exception. You must have loaded microcode before you execute GF.

H

Calls HELP files.

IC <on> <off>

Enables or disables the instruction cache. LD

List directory displays the contents of the "SAU" directory of the boot device.

LO <filename>

Load restores the named file from the "SAU" directory of the boot device. The LOW, HIGH, and START addresses are displayed.

LP

LP sets the loop bit on any microexecutive command issued until the machine is reset. With the loop bit set, the commands operate in loop mode.

M

Maps address space and enables mmu. Memory is rearranged as shown under Address Space in Chapters 7 thorough 12.

MG [<adr>]

MG starts the main processor at the address you specify, or at the current main processor program counter (PC), if you do not give an argument. The boot processor goes into slave mode.

MR [<adr>]

Starts the main processor at the address you specify, or at the current main processor program counter (PC), if you do not give an argument. The boot processor halts.

MS [<Count> [<adr>]]

Macro steps once or for the number of counts, showing the program counter at the end of the steps. A macrostep executes one instruction from memory, on the main processor.

MT [<Count> [<adr>]]

Macro traces for the number of counts you specify, or until you hit a key. Prints the program counter at each step.

MX <filename>

Loads and executes the program <filename> on the boot processor. <filename> is a program in the SAU4 directory on the volume defined as the disk device by the DI command.

P

Turns off the mapping. MMU is assumed at FFB400.

PV <PA> <addr>

Convert physical address to all virtual addresses by searching the page tables starting at current region registers or use region registers at ADDR. PA must be long-word aligned.

RE

Reset executes the RESET instruction. If entered while running on CPU B, a second RESET instruction is executed for CPU A. The debugger will initialize and wait for terminal input. This command also enables the POWER-OFF key.

RR <region register number (0-31, decimal, 0-1F Hex>

Accesses the region register you specify in the argument, and displays its contents.

S [<size_spec>] <start> <end> <value> [<mask>][<base_spec>]

Searches memory defined by bounds <start> to <end> for <value> through optional <mask>. If <mask> is not specified it defaults to $FFFFFFFF. The <size_spec> controls the item-size to be searched: byte, word or long.

SH <0-3>

Shuts down the Winchester unit and acknowledges outstanding interrupts. This command also enables the POWER-OFF key.

SK

Forces the MD to enter its keyboard polling loop without requiring a REset.

TE

Forces execution of boot PROM diagnostics.

UA

Loads the microcode loading program (ULOAD) and the entire set of microcode files needed to boot the operating system.

UC [<count>]

Executes a clock cycle, or the number of cycles you specify in the <count> field, on the main processor. The boot processor enters slave mode.

UF [-OFF]

UF sets or clears the "microcode loaded" flag in the PROM. When the flag is set, the PROM does not load microcode when enabling the main processor. UH

Halts (disables) the main processor and puts the boot processor into master mode. The boot processor now has bus mastership and control of the system.

UI

Resets the main processor and runs it from microlocation zero. This command starts the micro executive (if it is loaded).

UL <filename> -N

Reads one microcode data file from the SAU4 directory and the micro-loading program (ULOAD) from the boot device, and loads the micrcode. By default, if the microcode file is a WCS file, UL fills the WCS with freezes. Use the -N option when you are loading two WCS files separately.

UP <uadr>

Displays or sets the microcode program counter. If the main processor is running, you cannot read or set the microcode program counter.

UR [<uadr>]

Runs the microcode on the main processor, starting at the micro address you specify. The boot processor enters slave mode. If you do not specify an address, UR uses the current microcode program counter.

US [<Count> [<uadr>]]

Micro steps once or for the number of counts you give, showing the micro program counter at the end of the steps. If you specify <uadr>, the program starts at the micro address you specify.

UT [<Count> [<uadr>]]

Micro traces for the number of counts you specify or until you hit a key. The micro program counter is displayed at each step. If you specify a micro address, the trace starts at the micro address you specify.

UU

Causes a micro trap in the main processor, and displays the microcode program counter. You can use this command to stop the main processor.

UX

Resets the main processor and sets the micro program counter to 0.

V [<size_spec>] <start> <end> <target>[<base_spec>]

Verifies equality of two memory areas defined by <start> to <end> and <target> to <target>+<end>-<start>. If a discrepancy is found, the address in the first area and the contents of each are printed in the appropriate format.

VP <VA> <addr>

Convert virtual address to physical address using current region registers at ADDR. VA must be long-word aligned.

XD

Disables the X-On protocol used for communication with dumb terminals.

XE

Enables the X-On protocol used for communication with dumb terminals.

Command Formats:

```
<command>[<size_spec>] [<parameter_list>][<base_spec>]

<command> ::= A|B|C|D|DL|F|G|S|V|<empty>

<size_spec> ::= :I|:B|:W|:L

<parameter_list> ::= <parameter> ... [up to four]

<parameter> ::= <num_exp>|Dn|An|Rn|CCR|SR|
                 (An) |<num_exp>(An) |<num_exp>(<index_spec>) |
                <num_exp>(An,<index_spec>)

<num_exp> ::= <num>|*|<num_exp>+<num>|<num_exp>-<num>|
              <num>_exp>x<num>

<num> ::= <simple_number|$<simple_number>|
          <base>$<simple_number>|-<num>|<quoted_string>

<base> ::= <simple_number>

<quoted_string> ::= '<letter> ... <letter>' [up to four]

<index_spec> ::= An.W|Dn.W|An.L|Dn.L

<base_spec> ::= :O|:D|:H|:A
```

To reference an address stored in a relocation register, prefix the register name with a zero.

Semantics:

:I ::= instr-sized items, output in mnemonic format.

:B ::= byte-sized items, output in numeric format.

:W ::= word-sized items, output in numeric format.

:L ::= longword-sized items, output in numeric format.

Parameters are evaluated to a memory location or to an MD saved register, [e.g. Dn, An] or to a location computed from a saved register [num(An)]. Up to four parameters may be required. Unspecified parameters are set to zero.

:O ::= numbers and immediate constants printed in octal.

:D ::= numbers and immediate constants printed in dec.

:H ::= numbers and immediate constants printed in hex.

:A ::= numbers and immediate constants printed in ASCII.

All numeric input defaults to hexadecimal. $num implies hexadecimal. <base>$num implies base is <base> [8$777 is octal, 2$1001 is binary]. <base_spec> and <size_spec> may be specified anywhere in the command line as well as anywhere in A command input (except in quoted strings). All addresses and offsets are printed in hexadecimal regardless of <base_spec>.

CRASH ANALYSIS

Most fatal errors recognized by AEGIS are reported by the crash_system routine, which prints a status code (see Chapter 4, Error Codes and Messages), the address of the ECB for the failing routine, and the ID (PID) of the current process. AEGIS then executes a TRAP instruction, causing entry to the PROM mnemonic debugger with an "S" code (see MNEMONIC DEBUGGER ERROR CODES in Chapter 4). A dump can then be taken as described below.

If the system appears hung, or to stop the machine at any time for debugging, make sure the NORMAL/SERVICE switch is in the SERVICE position and type CTRL/<RETURN>, to pass control to the mnemonic debugger (MD). To continue, type:

```
  > G
  > G *+2 Then CTRL/<F>.
```

If this fails, press the RESET switch. (You should first verify that the hang is not a temporary one caused by a network failure.)

Notes on DNx60 Crash Status

DNx60's are microcoded machines. They may crash as described above with a CRASH SYSTEM message, ending up in MD but still running the micromachine (the CPU - MD prompt is '>') or they may occasionally freeze, (i.e. the micromachine halts), ending up in MD but running the CPIO processor (MD prompt is '%'). When a freeze occurs, the micro PC is printed by CPIO:

```
UPC: xxx
```

You may still take a dump from CPIO exactly as described above. A freeze is a more radical failure than a normal crash and usually indicates a hardware failure in the micromachine.

SYSTEM DUMPS

AEGIS contains a memory dump routine that can be used to dump the state of physical memory and the MMU to a floppy diskette, a cartridge tape, or a file on another node. DB, which understands the format of the dump, can then be used to analyze the cause of the crash. On completion, the dump routine executes a trap instruction and returns to the Mnemonic Debugger (MD).

To use this routine, follow these steps:

1. To dump to a diskette, insert a diskette into the floppy diskette drive of the Disk Storage Option. The diskette must already be formatted (with INVOL) for 1231 blocks, and must be write-enabled. (To write-enable the diskette, cover the write-enable hole with a gummed label.) (Note: if dumping to a floppy, only the first 1Mb of memory is dumped.)

2. To dump to a catridge tape, insert a cartridge tape into the tape drive and ensure that the tape is not write-protected (arrow in the upper left should point away from "safe").

3. To dump to another node, make sure that the other node is running NETMAN and that its disk has enough space for the dump (30 blocks plus 1024 blocks for each megabyte of memory on the dumping node).

4. Execute the following MD commands.

```
MD Commands                          Comments

>RESET<RETURN>                     Reset the machine
><RETURN>

>DI F              or              To dump to a floppy
>DI C              or              To dump to a cartridge tape
>DI N nn                           To dump to node nn

***   FOR ALL NODES:    ***

>G x00C00          or              Start dump to floppy or tape
>G x00C04                          Start the dump to another node

where x = 1 for all nodes EXCEPT DNx60s or Dn5xx-Ts,
      x = 2 for DNx60s,
      x = 10 for DN5xx-Ts.

***   ALSO, FOR DN3000s and DN5xx-Ts:    ***

>DU                                Start dump to floppy or tape
```

5. When the dump routine is complete, it executes a TRAP $F instruction and returns to MD. You should then run SALVOL and reboot AEGIS.

6. To load the dump back onto a node in order to use DB, use the CPTAPE or CPFLP utility of the /SYSTEST/SSR_UTIL directory.

/SYSTEST/SSR_UTIL

The following commands are available for system debugging in /SYSTEST/SSR_UTIL.

- ALL_STCODE -- Displays a list of all the system status codes for this operating system revision.

- BCR -- BINARY_CROSS_REFERENCE produces a cross reference from a list of object modules, whose names are read from standard input. Only the global symbols referencable from outside the module are cross referenced.

  Usage: BCR

- CMB -- Compare_binary compares two binary files -- files output by any Apollo compiler or assembler.

  Usage: CMB source destination [-ERRLIM <n>] [-NODate] [-List]

  "Source" may be a wildcard. "Destination" is a derived name. Multiple source/destination pairs may be specified. -ERRLIM may be specified to control how many differences per section are displayed; the default is 1. -NODATE may be specified to ignore differences in the creation times. -List will list the names of the files being compared. Compare_binary uses the common command line handler; type HELP CL for info.

- CPFLP -- A program used to copy a file from, or to, the node's floppy disk drive. To use the program, type: Usage: CPFLP [-READ file | -WRITE file | -FMT [file]]

- CPTAPE -- A program used to copy a file from, or to, the node's cartridge tape drive. To use the program, type: Usage: CPTAPE [-READ file | -WRITE file | -FMT [file]]

- DISK_ERR -- Displays information about last recorded disk error.

  Usage: DISK_ERR

- DMPF -- DUMP_FILE dumps a file of any type to STDOUT interpreted in hexadecimal and ASCII. If hex_start_offset is given, it specifies the byte of the file at which to start dumping. If hex_end_offset is given, it specifies the byte of the file at which to stop dumping. The first byte of the file is at offset 0. By default Dump_file dumps the entire input file.

  ```
  Usage:  DMPF input_pathname [-From hex_start_offset]
                              [-To hex_end_offset]
  ```

- DMPMBX -- DMPBX formats the contents of an MBX (mailbox) file. This program gathers information to document MBX bugs or problems. You must use DMPMBX from the same node as the server of the MBX file to be dumped. Normally, you use DMPMBX to display the contents of the MBX file just before the bug is found and just after the bug takes effect. In remote cases, both the server MBX file, and the client node's 'node_data/sysmbx file should be dumped out.

  ```
  Usage: DMPMBX mbx_file [-FD] [-CHAN channel_list])
  ```

- DTCB -- Dumps the contents of the TCP control blocks associated with a particular TCP connection. The address of the TCB to be dumped may be obtained using the TCPSTAT program. Two control blocks are dumped: the UCB (user control block) which contains the send and receive queues and user-related flags, and the TCB (TCP control block) which contains the connection sequence numbers, state, flags, and out-of-sequence queues.

  ```
  Usage: dtcb control_block_address
  ```

  ARGUMENTS

  control_block_address The hex address of the control
          block to be dumped, as displayed in the
          "TCB" field of a TCPSTAT report.

- FBS -- Used to find disk blocks whose data integrity is marginal. FBS DESTROYS ALL PREVIOUS USER DATA ON THE DISK. Following the "Show failing data?" prompt, fbs will destroy all previous information on the disk except for the physical bad spot list. If second thoughts occur abort at this prompt or before.

  For the number of passes entered, fbs writes then reads patterns of bits to all blocks on the disk. For those sectors which fail, their disk address is added to the physical list of bad spots. FBS requires between one and five hours of running time for various sizes of Apollo disks.

Please refer to the appropriate appendix in the DOMAIN System Command Reference for complete information on the use of this software tool.

NOTE: The fbs utility is only valid on DN100, DN3xx, DN4xx, DN5xx, DN6xx, DSP80, DSP90, and DSP160 series machines (machines with saul, sau2, sau3, sau4 or sau5). It is not valid on any other machine, and will become obsolete in the future.

- FIXCOM -- FIXCOM fixes comments in source files so that you can use the -COMCHK option when the file is compiled. For more information, see the file FIXCOM.HLP in this directory.

- FIXVOL -- FIXVOL is a general tool for examining and repairing disk volumes. With FIXVOL you can read, write, and examine arbitrary disk blocks (like RWVOL). You can maintain a stack of such disk blocks, examining them with the db program. FIXVOL will also repair LV and PV labels, the VTOC, or disk block headers. FIXVOL is fully interactive with the default mode being to prompt before making any destructive changes.

- FMPD -- FORMAT_PROCESS_DUMP formats a process dump and writes the formatted dump to standard output. Process dumps are made by the process manager when a process terminates abnormally and appended to the file "'node_data/proc_dump". Information included is similar to that provided by the FAULT_STATUS (FST) and TRACE_BACK (TB) commands.

  Usage: FMPD [pathname...] [-LAST]

  ARGUMENTS

  The pathname(s) are the name(s) of files containing unformatted process dumps. If none is given, "'node_data/proc_dump" is assumed. The explicit dump file name facility allows dumps to easily be saved by copying the "'node_data/proc_dump" file.

  OPTION

  -L[AST] Show only the last dump in the dump file.

- GLOBMAP -- GLOBMAP (show_global_section_map) displays a section map for all or a selected portion of the global libraries. The format of this map is:

```
    section_   address   size   section_name
```

All sections for all installed libraries are listed if no arguments are supplied.

Usage: GLOBMAP [-L library_wildcard...] [-S section_wildcard...] {CL}

If the -L (library) option is specified, the sections for all libraries whose names match the wildcard are listed.

If the -S (section) option is specified, each section name that matches the wildcard is listed.

If both options are specified, both the library and section names must match the wildcard in order to be listed.

- INXLIB -- The binary file(s) with the given pathnames are installed as private libraries. Any entrypoints in the libraries that match entrypoints that already exist in the known global table are treated specially, for example: If x.bin contains an entrypoint named error_$print, following an

  ```
  inxlib x.bin
  ```

  any subsequent call to error_$print will first end up in the copy of error_$print in x.bin. When that copy "returns", the real error_$print will be called, and when the real error_$print returns, the copy in x.bin will be called again.

  Usage: INXLIB pathname... [-db]

  Specifying -db will cause the old and new entrypoints of any replaced routines to be displayed.

  For an example of how inxlib can be used, see /us/lib/mt/mtx.pas.

  "Pathname" may be a wildcard. INXLIB uses the common command line handler; type HELP CL for info.

- LLKR -- LLKR will take a pathname and, if it is locked, tell you by whom it is being locked, on what node it is presently being locked, and the locking mode.

  Usage: LLKR [pathname]

- o JUMPER -- JUMPER is an interactive program that graphically displays the locations and functions of jumpers on all system printed circuit boards. JUMPER, which is completely menu-driven, makes use of the function keys (F1 to F8). Help is available in the file /SYSTEST/SSR_UTIL/JUMPER.HLP.

- o LSYSERR -- LSYSERR writes the contents of a system error log to standard output. Help is available in the file LSYSERR.HLP in this directory.

- o LTBL -- Prints the debug line number table

  Usage: LTBL (enter object module)

- o MBD -- MEMORY_BUFFER_DUMP dumps usage information about the TCP memory buffer pools. Usage statistics on TCP memory buffers may be obtained using the "-m" option of the TCPSTAT command; MBD is intended for analyzing cases where buffers are being lost. MBD scans the entire buffer pool, finding all the chains of in-use buffers; it then prints each chain of buffers. This information may help you in discovering reasons why buffers are being lost.

  Usage: mbd [-f] [-k]

  OPTIONS

  -f - Dump the free pools as well as the chains of in-use buffers. This produces a lot of output.

  -k - Don't try to lock the mutex on the buffer pools before doing the dump. This is useful mostly when the tcp_server has crashed with the buffer pool mutex locked.

- o NETLOG -- NETWORK_LOG is a tool that can be used to monitor paging and segment activity on a system. You will need two nodes: the node that is to be monitored and a node to collect the statistics.

  The Netlog program is run only on the node gathering statistics; the "listening" node. Netlog has two subsystems: START and REPORT. For more information, invoke Netlog and type "help".

- o OBJDMP -- Dumps an object (.bin) file

  Usage: OBJDMP infile [-L[IST] [outfile|-]]

- RINGLOG -- Monitors network traffic in and out of a node

  Usage: RINGLOG [-start | -stop | -read]

  ```
  -Start   activate ring network message logging
  -Stop    deactivate ring network message logging
           (if active) and display current contents
           of the log. (This is the default!)
  -Read    display current contents of the log.
           (Different from STOP because STOP
           deactivates logging if it is active.
  ```

- RWVOL -- RWVOL reads one physical disk address at a time, and puts it into a buffer in memory. The program first asks you to specify the controller type, and then to specify whether you want to read (R) or write (W) the disk address. If you plan to write to the disk, first choose the "R" option to read the address first.

  When you choose the "R" option, the program asks you to enter the physical disk address (Daddr:) to be read; enter it in hex. The program next asks you for the memory address to be used for the buffer containing the data it reads from the disk. If you are reading only one record at a time, you can issue a <CR> to use a default location for the buffer, which RWVOL then displays. If you specified a start address, you must specify an end address when prompted for it; otherwise just type <CR>.

  After RWVOL displays the "Done" message, you can use DB (if you are running online), or the Mnemonic Debugger (if you are running offline) to look at the contents of the record that you have read, in the location the program displayed for the buffer.

  Usage: RWVOL

- TRPT -- TRANSLATE_PROTOCOL_TRACE prints the contents of the tcp debugging log in readable format. The log consists of entries for each transition of the tcp state machine for all enabled connections. The information recorded includes the old and new states of the transition, the type of the input that caused it, an identifier for the connection, and for network input and output, the tcp sequence and acknowledge- ment numbers and packet flags, and a timestamp.

  To zero out the contents of the tcp trace buffer, leaving a clean slate for future TCP protocol tracing, use the -C option.

```
Usage: trpt [-ejlnsw] [-p <TCB address>] [<file>]

OPTIONS
      -e -- exit if an invalid log entry is found
      -l -- print lapsed times between log entries
      -w -- warn on invalid log entry diagnostic
      -s -- print only tcp state info.
      -n -- print only net input and output
      -t -- do not print timestamps
      -j -- just print TCB addresses
      -C -- zero out the contents of the trace
            buffer and exit
      -p <TCB address> -- print info only for this
            connection.
      <file> -- file name to open instead of
            `node_data/tcp_data
```

- TS -- SHOW_TIME_STAMP displays the module name and time stamp stored in an Apollo object module. Shown is the time and date that the module was created by the binder or one of the compilers. The time stamp is not affected by copying an object module, so it is a reliable indicator of whether particular object modules are copies of one another.

  Usage: TS object_module_name

- UPATH -- The UPATH command will return the pathname of a uid given a specific uid. To get a uid on a directory type: 'ld (directory name) -u' .

  Usage: UPATH [uid] {CL}

- XMT -- Examines magtape. This program is interactive and will print out a list of possible commands that can be executed if help if typed once the program has been entered.

  Usage: XMT

- ZEROTR --ZEROTR zeroes out the contents of the tcp trace buffer, leaving a clean slate for future TCP protocol tracing. See TRPT for information on interpreting the TCP protocol trace information.

  Usage: zerotr [<filename>]

  OPTIONS
  <file> -- file name to use instead of `node_data/tcp_data

CHAPTER 6

PERIPHERAL I/O

AT BUS DEVICES (SERIES 3000 MACHINES)

DEVICE	IRQ LINE	DMA LINE	PHYS ADDR	I/O ADDR
Ethernet Contr.	IRQ10 (IRQ9)	DRQ6 (DRQ3)	058000-058800	300-310
SPE Controller	IRQ4 (IRQ9)		05FC00-05FF80	3F8-3FF
PC Coprocessor	IRQ11 (IRQ15)	DRQ5	C00000-CFFFFF (D00000-DFFFFF)	

DEVICE ADDRESSES (PIO)

Refer to the ADDRESS SPACE section of the appropriate chapter for each model node.

DISK PARAMETERS

DTYPE	MODEL	CYLS	HEADS	BLK/TRK	TOTAL BLOCKS
Winchester Dtype Class 000 -- 14" Ring/disk PRIAM interface					
001	PRIAM 3350	561	3	18	30294 (7656)
006	PRIAM 6650	1121	3	18	60534 (EC76)
007	PRIAM 15450	1121	7	18	141246 (227BE)
Winchester Dtype Class 100 -- 8" ANSI Interface					
103	Micropolis 1203	580(1)	5	13(2)	37700 (9344)
104	PRIAM 3450	525	5	12	31500 (7B0C)
105	PRIAM 7050	1049	5	12	62940 (F5DC)
Winchester Dtype Class 200 -- 8" Ring/disk SMD Interface					
201	NEC D2257	1024	8	18	147456 (24000)
202	NEC D2246	687	6	18	74196 (121D4)

Winchester Dtype Class 300 -- 5 1/4" ST412 Interface

301	Micropolis	50MB	830	6	8	39840	(9BA0)
302	Micro.	86MB	1024	8	8	65536	(10000)
303	Fujitsu	86MB	754	11	8	66352	(10330)
304	Maxtor	140MB	918	15	8	110160	(1AE50)
305	Maxtor	190MB	1224	15	8	146880	(23DC0)
306	Vertex	86MB	1166	7	8	65296	(FF10)

Winchester Dtype Class 400 -- 5 1/4" ST412 Interface

401	Vertex	50MB	987	5	9	44415	(AD7F)
402	Vertex	86MB	1166	7	9	73458	(11EF2)
405	Micro.	50MB	1024	5	9	44820	(AF14)
406	Micro.	86MB	1024	8	9	73728	(12000)
410	Micro.	50MB	830	6	9	44820	(AF14)
411	Maxtor	190MB	1224	15	9	165240	(28578)

Winchester Dtype Class 500 -- 5 1/4" ESDI Interface

503	Priam/ Maxtor	170MB	1224	7	18	73458	(11EF2)
504	Priam/ Maxtor	380MB	1224	15	18	73458	(11EF2)
507	Micro.	170MB	1024	8	18	73458	(11EF2)

Winchester Dtype Class 600 -- 5 1/4" ESDI Interface

603	Maxtor	170	1224	7	18	154224	(25A70)
604	Maxtor	380	1224	15	18	330480	(50AF0)
607	Microp.	170	1024	8	18	147456	(24000)

Floppy Contoller (CTYPE=1)

001	Floppy		77	2	8	1232	(4D0)

Intel Storage Module Controller (SMD I/F) (CTYPE=4)

000	300MB SMD		823	19	18	281466	(44B7A)

Xylogics File Server Controller (SMD I/F) (CTYPE=4)

001	CDC (pn 3863)		711	24	26	443664	(6C510)
002	NEC (pn 5100)		760	19	31	447640	(6D498)

(1) Software uses only 525 cylinders.
(2) Software uses only 12 sectors.

		SEEK TIMES (mSecs)				AVG	TRANSFER	AVG
DTYPE	MODEL	T-to-T	AVG	MAX	RPM	LATENCY	RATE(MBS)	READ(*)
001	PRIAM 3350	8	45	85	3100	9.7	1.04	55.7
006	PRIAM 6650	8	45	85	3100	9.7	1.04	55.7
007	PRIAM 15450	8	45	75	3100	9.7	1.04	55.7
103	Micropolis 1203	12	42	85	3600	8.3	0.92	51.6
104	PRIAM 3450	8	42	75	3600	8.3	0.8	51.4
105	PRIAM 7050	8	42	75	3600	8.3	0.8	51.4
201	NEC D2257	5	20	40	3510	8.55	1.2	29.5
202	NEC D2246	7	25	50	3510	8.55	1.2	34.5
301	Micropolis 50MB	?	?	?	3600	8.3	.625	?
302	Micro. 86MB	6	28	62	3600	8.3	.625	38.0
303	Fujitsu 86MB	5	30	65	3600	8.3	.625	40.0
304	Maxtor 140MB	5	30	52	3600	8.3	.625	40.0
305	Maxtor 190MB	5	30	52	3600	8.3	.625	40.0
306	Vertex 86MB	?	?	?	3600	8.3	.625	?
603	Max 170 (info not available)							
604	Max 380	4	27	52	3600	8.33	1.25	35.3
607	Mic 170	5	28	62	3600	8.33	1.25	36.3**
607	Mic 170	6	23	50	3600	8.33	1.25	31.3***
001	Floppy	3	77	231	360	83.3	.0625	176.0
000	300MB SMD	6	29	55	3600	8.3	1.2	38.18
001	CDC (pn 3863)	5	20	45	3600	8.3	1.8	28.9
002	NEC (pn 5100)	?	15	?	3070	9.8	1.8	25.4

(*) Average read = Avg. Seek time + Avg. Latency + Sector Time

(**) Soft Sectored

(***) Hard Sectored

(?) Exact specifications not available at time of printing.

	DADDR(hex)/CYL(dec)		CYLS	BLKS
	BAD-SPOT	DIAGNOSTIC	USED	USED(*)
001 PRIAM 3350	7620/560	75AE/559	556	30024 (7548)
006 PRIAM 6650	EC40/1120	EC0A/1119	1116	60264 (EB68)
007 PRIAM 15450	22740/1120	226C2/1119	1116	140616 (22548)
103 Micropolis 1203	9303/579	92C2/578	520	31200 (79E0)
104 PRIAM 3450	7AD0/524	7A94/523	520	31200 (79E0)
105 PRIAM 7050	F5A0/1048	F564/1047	1044	62640 (F4B0)
201 NEC D2257	23F70/1023	23EE0/1022	1019	146736 (23D30)
202 NEC D2246	12168/686	120FC/685	682	73656 (11FB8)
301 Micropolis 50MB	9B70/829	9B10/827	827	39696 (9B10)
302 Micro. 86MB	FFC0/1023	FF40/1021	1021	65344 (FF40)
303 Fujitsu 86MB	102D8/753	10228/751	751	66088 (FF10)
304 Maxtor 140MB	1ADD8/917	1ACE8/915	915	109800 (1ACE8)
305 Maxtor 190MB	23D48/1223	23C58/1221	1221	146520 (23C58)
306 Vertex 86MB	FED8/1165	FE68/1163	1163	65128 (FE68)
603 Max 170	25974/1222	25878/1220	1220	153720 (25878)
604 Max 380	508D4/1222	506b8/1220	1220	329400 (506B8)
607 Mic 170	23EE0/1022	23DC0/1020	1020	146880 (23DC0)
001 Floppy	---	---	77	1232 (4D0)
000 300MB SMD	448CE/821	44A24/822	821	280782 (448CE)
001 CDC (pn 3863)	6C030/709	6C2A0/710	709	442416 (6C020)
002 NEC (pn 5100)	6CFFE/758	6D24B/759	751	442339 (6BFE3)

(*) Cyls, blocks used reflect loss of badspot and diagnostic cylinders and cylinders excluded for matching of secondary sources. For the 6XX dtypes four cylinders are reserved - the last for manufacturer encoded badspots, two diagnostic cylinders, and Apollo's badspot cylinder.

I/O MAP ALLOCATION

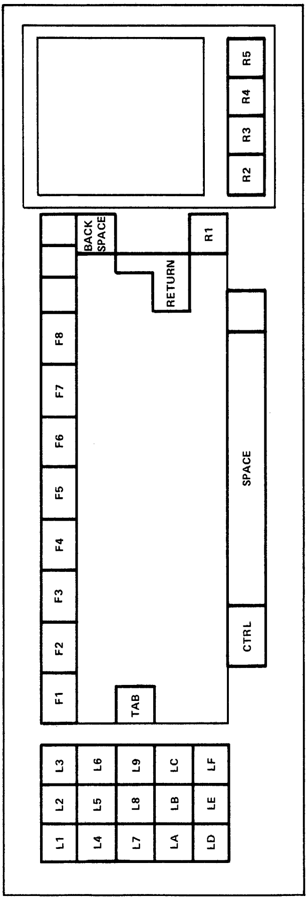
DEVICE	I/O PAGES	I/O MAP ENTRY ADDRESSES	
Floppy	0-1	FFF800-FFF802	(2 pages)
Unused	2-3F	FFF804-FFF87E	(62 pages)
Winchester	40-41	FFF880-FFF882	(2 pages)
Ring Transmit	42-45	FFF884-FFF88A	(4 pages)
Ring Receive	46-49	FFF88C-FFF892	(4 pages)
Ring 2nd Rcv Chan	4A-4D	FFF894-FFF89A	(4 pages)
Unused	4E-5E	FFF89C-FFF8BC	(17 pages)
Color DMA	5F-7F	FFF8BE-FFF8FE	(33 pages)
Bit blit	80-9F	FFF900-FFF93E	(32 pages)
Unused	A0-BF	FFF940-FFF97E	(32 pages)
Multibus	C0-FF	FFF980-FFF9FE	(64 pages)

I/O MAP

```
            DN4xx/6xx/DSP160
                                         CONTROLLER
                                              |
                                              |
                                      15      V      0
         16-bit address              +----------------+
            from controller:         |                |
                                     +----------------+
                                              |
         0-2 bits "strapped"     +-+-+        |
            per controller:      | | |        |
                                 +-+-+        |
                                   |          |
                                   |          |
                                17 V   10 9   V      0
         18-bit virtual         +--------+------------+
            DMA address:        |                     |
                                +--------+------------+
                                     |          |
                                     |          |
                     8-bit VPN       |          |
             +-----------------------+          |
             |                                  |
             |        IOMAP                     |
             | +00 +---------+                  |
             |     |         |                  |
             |     |         |                  | 10-bit page
             |     +--+------+                  |     displacement
             +---> |//|      |---+              |
                   +--+------+   |              |
                   |         |   |              |
                   |         |   | 12-bit PPN   |
                   |         |   |              |
                   |         |   |              |
               +FF +---------+   |              |
                                 |              |
                                 |              |
                                 |              |
                          21     V     10 9     V    0
         22-bit physical +---------------+------------+
            address:     |                            |
                         +---------------+------------+
                                       |             |
                                       |             +--> Byte
                                       V                  Select

                                     MEMORY
```

IOMAP has 256 one-word entries from FFF800 - FFF9FE.

880 Keyboard - Map



880 Keyboard Chart - Physical

High Order Nibble

Low Order Nibble

ASCII Codes

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	^`	^P	BLNK	0	@	P	`	p	R1	N0	R1U	N0U	F1	F1S	F1U	F1C	0
1	^A	^Q	!	1	A	Q	a	q	L1	N1	L1U	N1U	F2	F2S	F2U	F2C	1
2	^B	^R	"	2	B	R	b	r	L2	N2	L2U	N2U	F3	F3S	F3U	F3C	2
3	^C	^S	#	3	C	S	c	s	L3	N3	L3U	N3U	F4	F4S	F4U	F4C	3
4	^D	^T	$	4	D	T	d	t	L4	N4	L4U	N4U	F5	F5S	F5U	F5C	4
5	^E	^U	%	5	E	U	e	u	L5	N5	L5U	N5U	F6	F6S	F6U	F6C	5
6	^F	^V	&	6	F	V	f	v	L6	N6	L6U	N6U	F7	F7S	F7U	F7C	6
7	^G	^W	'	7	G	W	g	w	L7	N7	L7U	N7U	F8	F8S	F8U	F8C	7
8	^H	^X	(	8	H	X	h	x	L8	N8	L8U	N8U	\	\S	tpad	^\	8
9	^I	^Y	)	9	I	Y	i	y	L9	N9	L9U	N9U	|	|S		|C	9
A	^J	^Z	*	:	J	Z	j	z	LA	N.	LAU	N.U	TAB	TABS		TABC	A
B	^K	^[	+	;	K	{	k	[	LB	N=	LBU	N=U	CR	CRS		CRC	B
C	^L		,	<	L		l		LC	N+	LCU	N+U	/	?		^/	C
D	^M	^]	-	=	M	}	m	]	LD	N-	LDU	N-U	R2	R5	R2U	R5U	D
E	^N	^~	.	>	N	^	n	~	LE	N*	LEU	N*U	R3	BS	R3U		E
F	^O				O	—	o		LF	N/	LFU	N/U	R4		R4U		F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

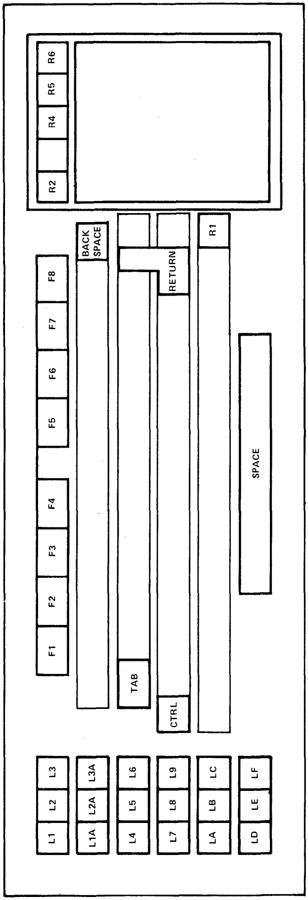
880 Keyboard Chart - Translated (User Mode)

High Order Nibble

Low Order Nibble ASCII Codes

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	^`	^P	SP	0	@	P	`	p		R1		R1U	F1	F1S	F1U	F1C	0
1	^A	^Q	!	1	A	Q	a	q	L1	R2	L1U	R2U	F2	F2S	F2U	F2C	1
2	^B	^R	"	2	B	R	b	r	L2	R3	L2U	R3U	F3	F3S	F3U	F3C	2
3	^C	^S	#	3	C	S	c	s	L3	R4	L3U	R4U	F4	F4S	F4U	F4C	3
4	^D	^T	$	4	D	T	d	t	L4	R5	L4U	R5U	F5	F5S	F5U	F5C	4
5	^E	^U	%	5	E	U	e	u	L5	BS	L5U		F6	F6S	F6U	F6C	5
6	^F	^V	&	6	F	V	f	v	L6	CR	L6U		F7	F7S	F7U	F7C	6
7	^G	^W	'	7	G	W	g	w	L7	TAB	L7U		F8	F8S	F8U	F8C	7
8	^H	^X	(	8	H	X	h	x	L8	STAB	L8U		N0	N8	N0U	N8U	8
9	^I	^Y	)	9	I	Y	i	y	L9	CTAB	L9U		N1	N9	N1U	N9U	9
A	^J	^Z	*	:	J	Z	j	z	LA		LAU		N2	N.	N2U	N.U	A
B	^K	^[	+	;	K	[	k	{	LB		LBU		N3	N=	N3U	N=U	B
C	^L	^\	,	<	L	\	l	|	LC		LCU		N4	N+	N4U	N+U	C
D	^M	^]	-	=	M	]	m	}	LD		LDU		N5	N-	N5U	N-U	D
E	^N	^~	.	>	N	^	n	~	LE		LEU		N6	N*	N6U	N*U	E
F	^O	^/	/	?	O	_	o	^|	LF		LFU		N7	N/	N7U	N/U	F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

Low-Profile Model I Keyboard - Map



Low-Profile Model I Keyboard Chart - Physical

Hign Order Nibble

Low Order Nibble ASCII Codes

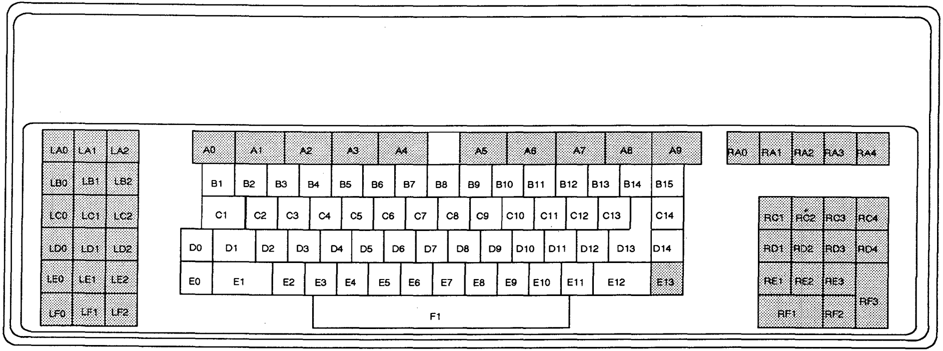
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	^SP	^P	SP	0	@	P	`	p	R1	R1S	R1U	L1A	F1	F1S	F1U	F1C	0
1	^A	^Q	!	1	A	Q	a	q	L1	L1S	L1U	L2A	F2	F2S	F2U	F2C	1
2	^B	^R	"	2	B	R	b	r	L2	L2S	L2U	L3A	F3	F3S	F3U	F3C	2
3	^C	^S	#	3	C	S	c	s	L3	L3S	L3U	R6	F4	F4S	F4U	F4C	3
4	^D	^T	$	4	D	T	d	t	L4	L4S	L4U	L1AS	F5	F5S	F5U	F5C	4
5	^E	^U	%	5	E	U	e	u	L5	L5S	L5U	L2AS	F6	F6S	F6U	F6C	5
6	^F	^V	&	6	F	V	f	v	L6	L6S	L6U	L3AS	F7	F7S	F7U	F7C	6
7	^G	^W	'	7	G	W	g	w	L7	L7S	L7U	R6S	F8	F8S	F8U	F8C	7
8	^H	^X	(	8	H	X	h	x	L8	L8S	L8U	L1AU	\		tpad	^\	8
9	^I	^Y	)	9	I	Y	i	y	L9	L9S	L9U	L2AU	|		R2S		9
A	^J	^Z	*	:	J	Z	j	z	LA	LAS	LAU	L3AU	TAB	TABS	R3S	TABC	A
B	^K	ESC	+	;	K	{	k	[	LB	LBS	LBU	R6U	CR	CRS	R4S	CRC	B
C	^L		,	<	L		l		LC	LCS	LCU		/	?	R5S	^/	C
D	^M	^]	-	=	M	}	m	]	LD	LDS	LDU		R2	R5	R2U	R5U	D
E	^N	^~	.	>	N	^	n	~	LE	LES	LEU		R3	BS	R3U		E
F	^O				O	___	o	DEL	LF	LFS	LFU		R4	mous	R4U		F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

Low-Profile Model I Keyboard Chart - Translated (User Mode)

High Order Nibble

Low Order Nibble ASCII Codes

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	^SP	^P	SP	0	@	P	`	p		R1		R1U	F1	F1S	F1U	F1C	0
1	^A	^Q	!	1	A	Q	a	q	L1	R2	L1U	R2U	F2	F2S	F2U	F2C	1
2	^B	^R	"	2	B	R	b	r	L2	R3	L2U	R3U	F3	F3S	F3U	F3C	2
3	^C	^S	#	3	C	S	c	s	L3	R4	L3U	R4U	F4	F4S	F4U	F4C	3
4	^D	^T	$	4	D	T	d	t	L4	R5	L4U	R5U	F5	F5S	F5U	F5C	4
5	^E	^U	%	5	E	U	e	u	L5	BS	L5U	R2S	F6	F6S	F6U	F6C	5
6	^F	^V	&	6	F	V	f	v	L6	CR	L6U	R3S	F7	F7S	F7U	F7C	6
7	^G	^W	'	7	G	W	g	w	L7	TAB	L7U	R4S	F8	F8S	F8U	F8C	7
8	^H	^X	(	8	H	X	h	x	L8	STAB	L8U	R5S	R1S	L8S	L1A	L1AU	8
9	^I	^Y	)	9	I	Y	i	y	L9	CTAB	L9U		L1S	L9S	L2A	L2AU	9
A	^J	^Z	*	:	J	Z	j	z	LA		LAU		L2S	LAS	L3A	L3AU	A
B	^K	ESC	+	;	K	[	k	{	LB		LBU		L3S	LBS	R6	R6U	B
C	^L	^\	,	<	L	\	l	|	LC		LCU		L4S	LCS	L1AS		C
D	^M	^]	-	=	M	]	m	}	LD		LDU		L5S	LDS	L2AS		D
E	^N	^~	.	>	N	^	n	~	LE		LEU		L6S	LES	L3AS		E
F	^O	^?	/	?	O	_	o	DEL	LF		LFU		L7S	LFS	R6S		F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	


LA0 LA1 LA2
LB0 LB1 LB2
LC0 LC1 LC2
LD0 LD1 LD2
LE0 LE1 LE2
LF0 LF1 LF2
A0 A1 A2 A3 A4 A5 A6 A7 A8 A9
B1 B2 B3 B4 B5 B6 B7 B8 B9 B10 B11 B12 B13 B14 B15
C1 C2 C3 C4 C5 C6 C7 C8 C9 C10 C11 C12 C13 C14
D0 D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14
E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 E10 E11 E12 E13
F1
RA0 RA1 RA2 RA3 RA4
RC1 RC2 RC3 RC4
RD1 RD2 RD3 RD4
RE1 RE2 RE3
RF1 RF2 RF3

KEY NUMBERS

Low-Profile Model II Keyboard Chart - ASCII Mode

High Order Nibble

Low Order Nibble

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0		^C11	F1	B11	+B3	+C11	B14	C11	E13	+E13	:E13	LB0	A1	+A1	:A1	^A1	0
1	^D2	^C2	+B2	B2	+D2	+C2	D2	C2	LA0	+LA0	:LA0	LB1	A2	+A2	:A2	^A2	1
2	^E6	^C5		B3	+E6	+C5	E6	C5	LA1	+LA1	:LA1	LB2	A3	+A3	:A3	^A3	2
3	^E4	^D3	+B4	B4	+E4	+D3	E4	D3	LA2	+LA2	:LA2	RA4	A4	+A4	:A4	^A4	3
4	^D4	^C6	+B5	B5	+D4	+C6	D4	C6	LC0	+LC0	:LC0	+LB0	A5	+A5	:A5	^A5	4
5	^C4	^C8	+B6	B6	+C4	+C8	C4	C8	LC1	+LC1	:LC1	+LB1	A6	+A6	:A6	^A6	5
6	^D5	^E5	+B8	B7	+D5	+E5	D5	E5	LC2	+LC2	:LC2	+LB2	A7	+A7	:A7	^A7	6
7	^D6	^C3	D12	B8	+D6	+C3	D6	C3	LD0	+LD0	:LD0	+RA4	AB	+A8	:A8	^A8	7
8	^D7	^E3	+B10	B9	+D7	+E3	D7	E3	LD1	+LD1	:LD1	:LB0	D14			^D14	8
9	^C9	^C7	+B11	B10	+C9	+C7	C9	C7	LD2	+LD2	:LD2	:LB1			+RA0		9
A	^D8	^E2	+B9		+D8	+E2	D8	E2	LE0	+LE0	:LE0	:LB2	C1	+C1	+RA1	^C1	A
B	^D9	B1	+B13	D11	+D9	+C12	D9	C12	LE1	+LE1	:LE1	:RA4	D13		+RA2	^D13	B
C	^D10	A0	E9	+E9	+D10	+A0	D10	^A0	LE2	+LE2	:LE2	:A0	E11	+E11	+RA3	^E11	C
D	^E8	^C13	B12	B13	+E8	+C13	E8	C13	LF0	+LF0	:LF0	:A9	RA0		:RA0	:RA3	D
E	^E7	^B14	E10	+E10	+E7	+B7	E7	+B14	LF1	+LF1	:LF1		RA1	B15	:RA1		E
F	^C10	A9	+A9	^A9	+C10	+B12	C10	C14	LF2	+LF2	:LF2		RA2		:RA2		F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

+ = Shift   ^ = Control   : = Up transition

Low-Profile Model II Keyboard Chart - Keystate Mode

High Order Nibble

Low Order Nibble

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0		RA2	B10	C5	LD0	D12	E2	RE3		: RA2	: B10	: C5	: LD0	: D12	: E2	: RE3	0
1	LA0	RA3	B11	C6	LD1		E3		: LA0	: RA3	: B11	: C6	: LD1		: E3		1
2	LA1	RA4	B12	C7	LD2	D13	E4	LF0	: LA1	: RA4	: B12	: C7	: LD2	: D13	: E4	: LF0	2
3	LA2	LB0	B13	C8	LD3	D14	E5	LF1	: LA2	: LB0	: B13	: C8	: LD3	: D14	: E5	: LF1	3
4	A0	LB1	B14	C9			E6	LF2	: A0	: LB1	: B14	: C9			: E6	: LF2	4
5	A1	LB2	B15	C10		RD1	E7		: A1	: LB2	: B15	: C10		: RD1	: E7		5
6	A2			C11	D2	RD2	E8	F1	: A2			: C11	: D2	: RD2	: E8	: F1	6
7	A3	B1	LC0	C12	D3	RD3	E9		: A3	: B1	: LC0	: C12	: D3	: RD3	: E9		7
8	A4	B2	LC1	C13	D4	RD4	E10		: A4	: B2	: LC1	: C13	: D4	: RD4	: E10		8
9	A5	B3	LC2		D5	LE0	E11	RF1	: A5	: B3	: LC2		: D5	: LE0	: E11	: RF1	9
A	A6	B4		C14	D6	LE1	E12		: A6	: B4		: C14	: D6	: LE1	: E12		A
B	A7	B5			D7	LE2		RF2	: A7	: B5			: D7	: LE2		: RF2	B
C	A8	B6	C1	RC1	D8		E13	RF3	: A8	: B6	: C1	: RC1	: D8		: E13	: RF3	C
D	A9	B7	C2	RC2	D9	E0			: A9	: B7	: C2	: RC2	: D9	: E0			D
E	RA0	B8	C3	RC3	D10	E1	RE1	LED ON	: RA0	: B8	: C3	: RC3	: D10	: E1	: RE1	LED ON	E
F	RA1	B9	C4	RC4	D11		RE2		: RA1	: B9	: C4	: RC4	: D11		: RE2		F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

: = Up transition

MAGTAPE CONTROLLER

```
Controller control page:  FE8000
Interrupt vector number:  B3 ($2CC in page 0)
MULTIBUS interrupt level: 3 (See MIC)
I/OMAP entries: FFF980-FFF9FE (64 pages)

  (3)FE80AA     CHANNEL ATTENTION (do something useful)
  (3)FE80AB     CONTROLLER RESET
   |
   +-- (020 MACHINES)
```

System Configuration Pointer (at xxxFF6)

```
 +00       +01       +02                 +04
+--------+--------+-----------------+-----------------+
|00000001|00000000| -> CONFIG BLOCK |        0        |
+--------+--------+-----------------+-----------------+
```

System Configuration Block

```
 +00       +01       +02                 +04
+--------+--------+-----------------+-----------------+
|00000011|00000000|->CHAN CNTRL BLK |        0        |
+--------+--------+-----------------+-----------------+
```

Channel Control Block

```
     15      8 7      0
    +--------+--------+
+00 |  CCW   |  GATE  |  CCW: Set to $11 for normal
    +--------+--------+     operations. Set to $09 to
+02 |  -> PARM BLOCK  |     clear active interrupt.
    +-----------------+  GATE: Set to $FF before starting
+04 |        0        |     an operation. Set to 00 by
    +-----------------+     controller on completion.
+06 |        0        |
    +-----------------+
```

To initiate an operation

(Set up parameter block)

```
MOVE.W   #$11FF,GATE      CLOSE GATE
MOVE.B   #0,$FE80AA       WAKE UP CONTROLLER
```

To acknowledge tape interrupt:

```
MOVE.W   #$09FF,GATE       CLOSE GATE
MOVE.B   #$20,CMD_BYTE     DO-NOTHING COMMAND
MOVE.W   NO_I_BIT,CONTROL  ENSURE INTERRUPT BIT OFF
MOVE.B   #00,$FE80AA       WAKE UP CONTROLLER
TST.B    GATE+1            WAIT FOR ACK TO FINISH
BNE      *-4
```

Parameter Block

```
           15      8 7       0
          +--------+--------+
      +00 |CMD BYTE|00000000|
          +--------+--------+
      +02 |        0        |
          +--------+--------+
      +04 |WG-SHR--|DLIMUU--|   (CONTROL -- SEE BELOW)
          +--------+--------+
      +06 |  ACTUAL COUNT   |
          +-----------------+
      +08 | COUNT TO R/W (*)|
          +-----------------+
      +0A | RECORDS/OVERRUN |
          +-----------------+
      +0C |  BUFFER POINTER |
          +-----------------+
      +0E |        0        |
          +-----------------+
      +10 |SC-EEEEE|-OLERBP-|   (STATUS -- SEE BELOW)
          +-----------------+
      +12 | INT OR LINK PTR |
          +-----------------+
      +14 |        0        |
          +-----------------+
```

* Manufacturer recommends 4K - 8K.

COMMAND BYTE

00	Initialize	3C	Edit (rewrite prior record)
20	No operation	40	Write filemark
28	Return status	44	Skip to filemark
2C	Read	48	Space 'n' records
30	Write	70	Space 'n' or to filemark
34	Rewind	4C	Erase fixed (3.5" * 'n')
38	Rewind/unload	50	Erase from here to EOT

CONTROL WORD (appropriate value)

W	8000	Bus width (0 => 8 bits, 1 => 16)	(1)
G	4000	Grab bus before tape movement	(0)
S	1000	Operate in streaming mode	(?)
H	0800	Select high speed (100ips)	(0)
R	0400	Reverse direction for operation	(?)
D	0080	Grab bus during DMA transfers	(0)
L	0040	Link (=> ignore I and M bits)	(?)
I	0020	Interrupt when done	(1)
M	0010	1 => use mailbox interrupts	(0)
UU	000C	Unit select (00 through 11)	(00)

STATUS WORD (normal state at completion)

S	8000	Execution (of parm block) started	(1)
C	4000	Execution completed ok	(1)
E..E	1F00	Error code:	(00)
	00	No unrecoverable error	
	01	Timed out waiting for Data Busy false	
	02	Timed out waiting for Data Busy false, Formatter Busy false and Ready true	
	03	Timed out waiting for Ready false	
	04	Timed out waiting for Ready true	
	05	Timed out waiting for Data Busy true	
	06	Memory time-out during system memory reference	
	07	Blank tape encountered	
	08	Error in micro-diagnostic	
	09	Unexpected end of tape	
	0A	Tape read/write error	
	0B	Tape overrun	
	0D	Read parity error	
	0E	Checksum error	
	0F	Tape timeout (read on blank tape or reading larger block than written)	
	10	Tape not ready	
	11	Write attempted on protected tape	
	13	Diagnostic mode jumper not installed	
	14	Illegal attempt to link	
	15	Filemark encountered during read operation	
	16	Parameter error (byte count zero or too large)	
	18	Hardware error	
	19	Read or write terminated by OS or disk	
O	0040	Online	(1)
L	0020	Load point	(?)
E	0010	End of tape	(?)
R	0008	Ready	(1)
B	0004	Formatter busy	(0)
P	0002	Tape is write protected	(?)

MULTIBUS DEVICES

INTERRUPT LEVEL	PRODUCT CODE	PRODUCT DESCRIPTION	PART NO	CSR-ADDR (PAGE)	MEMORY MAPPED ADDRESSES
0	---	Reserved for customers	---	0-3C00	0000-7C00
1	COM-ETH	Interlan Ethernet Ctlr	003613	80-8F	
1	---	FPS array proc.	---	7400	
2	COM-X25	TITN X25 Ctlr	002858		7000-7FFF
2	COM-CTL COM-PC8	DOMAIN 3270,3770, PCI		0800	4000-7FFF
3	MSD1600	CIPRICO TAPE-MASTER Ctlr	001012	7800	OCCUPIES NO MEMORY SPACE.
3	MSD6250	CIPRICO TAPE-MATER "A" CTRL	007141	7800	OCCUPIES NO MEMORY SPACE.
4	MSD-300	Intel SMD Ctlr	001380	6C00	OCCUPIES NO MEMORY SPACE
4	MSD-500	Xylogics FSD Ctlr	003864	6840	OCCUPIES NO MEMORY SPACE
5	SFW-VER	IKON 10071-5, IKON 10085 Mbus Versatec (V80) Ctlr		400	
6	--	Printronix Parallel Ptr.		7C00	

REGISTER SET

```
    31            16 15      8 7      0
   +----------------+--------+--------+
   |                |        |        |  D0
   +----------------+--------+--------+
   |                |        |        |  D1
   +----------------+--------+--------+
   |                |        |        |  D2
   +----------------+--------+--------+
   |                |        |        |  D3
   +----------------+--------+--------+
   |                |        |        |  D4
   +----------------+--------+--------+
   |                |        |        |  D5
   +----------------+--------+--------+
   |                |        |        |  D6
   +----------------+--------+--------+
   |                |        |        |  D7
   +----------------+--------+--------+

   +----------------+-----------------+
   |                |                 |  A0
   +----------------+-----------------+
   |                |                 |  A1
   +----------------+-----------------+
   |                |                 |  A2
   +----------------+-----------------+
   |                |                 |  A3
   +----------------+-----------------+
   |                |                 |  A4
   +----------------+-----------------+
   |                |                 |  A5   (DB)
   +----------------+-----------------+
   |                |                 |  A6   (SB)
   +----------------+-----------------+
      DB --> Start of Data Section
      SB --> Top of current Stack Frame
   +- - - - - - - - - - - - - - - - - +
   |        USER STACK POINTER        |
   +----------------------------------+  A7   (SP)  *
   |     SUPERVISOR STACK POINTER     |
   +- - - - - - - - - - - - - - - - - +
      * Choice made on basis of S bit in SR
   +----------------------------------+
   |                                  |  PC
   +----------------------------------+
                     System    User
                    +--------+--------+
                    |T-S--III|---XNZOC|  SR
                    +--------+--------+
T-Trace Mode       III-Interrupt Level  X-Extend   O-Overflow
S-Supervisor Mode    000-enabled        N-Negative C-Carry
                     111 - disabled     Z-Zero
```

TIMERS

Write CR1 or CR3	[8800	FFAC00]
Rd Status, Wr CR2	[8802	FFAC02]
Rd/Wr Timer 1, Hi	[8804	FFAC04]
Rd/Wr Timer 1, Lo	[8806	FFAC06]
Rd/Wr Timer 2, Hi	[8808	FFAC08]
Rd/Wr Timer 2, Lo	[880A	FFAC0A]
Rd/Wr Timer 3, Hi	[880C	FFAC0C]
Rd/Wr Timer 3, Lo	[880E	FFAC0E]
Calendar Control	[8880	FFAC80]
Calendar Data Wr	[8882	FFAC82]
Calendar Data Rd	[8884	FFAC84]

TOUCHPAD

The touchpad sends approximately 30 data points per second through the same SIO port (zero) as the keyboard, at a speed of 1200 baud. Each data point consists of four bytes, as follows:

```
+---------------------------------------------------------+
| escape code |     lo 8      | lo 4 : hi 4 |     hi 8     |
|             |     bits      | bits : bits |     bits     |
|     E8      |     of X      | of Y : of X |     of Y     |
+---------------------------------------------------------+
                                4-7    0-3
    byte 0         byte 1          byte 2         byte 3
```

The range of X and Y coordinates is approximately 30 to 1100.

XYLOGICS CONTOLLER

MULTIBUS STANDARD I/O ADDRESSES

DESCRIPTION	8-BIT
IOPB Relocation Register Low Byte	40
IOPB Relocation Register High Byte	41
IOPB Address Register Low Byte	42
IOPB Address Register High Byte	43
Controller Status Register (CSR)	44
Controller Reset/Update IOPB Register	45

```
   7      6      5      4      3      2      1      0
+------+------+------+------+---------------------------+
| AUD  | RELO | CHEN | IEN  |       COMMAND CODE        |
+------+------+------+------+------+------+------+------+
|  0   | IEI  | IERR | HDP  | ASR  | EEF  |  ECC CODE   |
+------+------+------+------+------+------+------+------+
| ERRS |      0      |  CONTROLLER TYPE   |  0   | DONE |
+------+-------------+--------------------+------+------+
|               ERROR OR COMPLETION CODE                |
+------+---------------------------+--------------------+
| B/W  |    INTERLEAVE FACTOR      |     THROTTLE       |
+------+------+--------------------+------+-------------+
| DRIVE TYPE  |            0              | UNIT SELECT |
+-------------+---------------------------+-------------+
|                    HEAD ADDRESS                       |
+-------------------------------------------------------+
|                   SECTOR ADDRESS                      |
+-------------------------------------------------------+
|             CYLINDER ADDRESS LOW BYTE                 |
+----------------------------------+--------------------+
|                0                 | CYL ADDRESS HIGH   |
+----------------------------------+--------------------+
|                SECTOR COUNT LOW BYTE                  |
+-------------------------------------------------------+
|               SECTOR COUNT HIGH BYTE                  |
+-------------------------------------------------------+
|         DATA TRANSFER ADDRESS LOW BYTE                |
+-------------------------------------------------------+
|         DATA TRANSFER ADDRESS HIGH BYTE               |
+-------------------------------------------------------+
|         DATA TRANSFER RELOCATION ADDRESS LOW BYTE     |
+-------------------------------------------------------+
|      DATA TRANSFER RELOCATION ADDRESS HIGH BYTE       |
+-------------------------------------------------------+
|                      HEAD OFFSET                      |
+-------------------------------------------------------+
|                           0                           |
+-------------------------------------------------------+
|             NEXT IOPB ADDRESS LOW BYTE                |
+-------------------------------------------------------+
|             NEXT IOPB  ADDRESS HIGH BYTE              |
+--------------------+----------------------------------+
| ECC PATTERN HIGH   |                0                 |
+--------------------+----------------------------------+
|                  ECC PATTERN LOW                      |
+-------------------------------------------------------+
|                  ECC OFFSET BYTE LOW                  |
+-------------------------------------------------------+
|                 ECC OFFSET BYTE HIGH                  |
+-------------------------------------------------------+
CSR:
+------+------+------+------+------+------+------+------+
| GBSY |  ERR | DERR | IPND | ADRM | AREQ | AACK | DRDY |
+------+------+------+------+------+------+------+------+
```

A on DRV STAT:

```
+------+------+------+------+------+------+-------------+
| ONCL | DRDY | WRPT |  DPB | SKER | DFLT |      0      |
+------+------+------+------+------+------+-------------+
```

Bit Number

0 - COMM	D - DATAH
1 - IMODE	E - DATARL
2 - STAT1	F - DATARH
3 - STAT2	10 - HDOFST
4 - THROT	11 - RES
5 - DRIVE	12 - NIOPL
6 - HEAD	13 - NIOPH
7 - SECT	14 - ECCMB
8 - CYLL	15 - ECCML
9 - CYLH	16 - ECCAL
A - SCNTL	17 - ECCAH
B - SCNTH	
C - DATAL	

Commands

Code	Function
0	NOP
1	Write
2	Read
3	Write Track Headers
4	Read Track Headers
5	Seek
6	Drive Reset
7	Write Format
8	Read H-D-E
9	Rd Drive Stat
A	Write H-D-E
B	Set Drive Size
C	Self Test
D	Reserved
E	Buffer Fill
F	Buffer Dump

Abbreviations:

AACK - Attention Acknowledge
ADRM - 24 bit Address mode
AREQ - Attention Request
ASR - Automatic seek retry
AUD - Automatic Update
B/W - Byte / Word mode
CHEN - Chain Enable
DPB - Dual Port Busy
DERR - Double Error
DFLT - Drive Faulted
DONE - Operation Done
DRDY - Drive Ready (L)
EEF - Enable Extended Function
ERR - Error
ERRS - Error Summary
GBSY - Go Busy
HDP - Hold Dual Port
IEI - Interrupt on each IOPB
IEN - Interrupt Enable
IERR - Interrupt on Error
IPND - Interrupt Pending
ONCL - On Cylinder (L)
RELO - Relocate Enable
SKER - Seek Error
WRPT - Write Protect

CHAPTER 7

DN300,DN320,DN330

ADDRESS SPACE (DN300/320)

physical		virtual
100400	traps	0
400	PROM	400->7FFF (one-to-one)
100800	phys mem	100800->FFFFF
700000	ptt	700000->7FFFFF
100000	MD STK,DATA	E00000
20000	displ_mem	FC0000->FDFFFF
B000	FPU ctl	FF7000
B400	FPU cmd	FF7400
B800	FPU cs	FF7800
9400	disp 1	FF9800
9800	ring 2	FF9C00
9000	DMA ctl	FFA000
9C00	FLP,WIN,CAL	FFA800
8800	timers	FFAC00
8400	sios	FFB000
8000	mmu	FFB400
4000	pft	FFB800->FFF7FF

ADDRESS SPACE (DN330)

physical		virtual
400	prom	400-3FFF (one-to-one)
4000	pft	3FFB800-3FFF7FF
8000	mmu	3FFB400
8400	sios	3FFB000
8800	timers	3FFAC00
9800	ring	3FF9C00
9C00	disk, tape, cal	3FFA800
A400	pbu ctl	3FF7C00 (DSP90/DN560)
A800	lpr	3FF8000 (DSP90)
BC00	VME control	3FF9400 (DN560)
E000	color_sup	3FF6000 (DN560)
E400	color_user	3FF6400 (DN560)
E800	color_wcs	3FF6800 (DN560)
F000	displ_sup	3FF9800

```
          F400 displ_user       3FFA000
          F800 displ_wcs        3FFA400
         10000                                           iomap
3FF5000-3FF5FFF(DSP90/DN560)
         14000 prom2               -
         20000 displ_mem        3FC0000-3FDFFFF(DN560)
         40000                                       color_mem
3FA0000-3FBFFFF(DSP90/DN560)
         70000 pbu                 i/o                     ref
3FE0000-3FE7FFF(DSP90/DN560)
         80000 pbu 1st half        -
        100000 mem: md data     3D00000
        100400 mem: traps          0
        100800 mem               100800
        380000 pbu       2nd        half                     -
(DSP90/DN560)
        400000 ptt               400000-7FFFFF
```

CACHE CONTROL REGISTER (CACR) [MOVEC to/from 002] (DN330)

```
 31                             8  7  6  5  4  3  2  1  0
+-----------------------------+--+--+--+--+--+--+--+--+
|0                            0|00|00|00|00|CC|CE|FC|EC|
+-----------------------------+--+--+--+--+--+--+--+--+
```

CC - Clear entire cache (W/0)
CE - Clear entry addresses by CAAR (W/O)
FC - Freeze cache (enabled, but no replacing data) (R/W)
EC - Enable cache (R/W)

CACHE ADDRESS REGISTER (CAAR) [MOVEC to/from 802] (DN330)

```
 31                                                 0
+------------------------------------------+-----------+
|          CACHE FUNCTION ADDRESS          |   INDEX   |
+------------------------------------------+-----------+
```

CONFIGURATION

```
+------+ +------+
|      | |      |                        SBUS
| CPU  |==| MMU  |===============================================
|      | |      |   |         |        |         |        |      "
+------+ +------+   |         |        |         |        |      "
                +-------+ +-------+ +------+ +-------+ +------+  "
                | MAIN  | | BOOT  | | SIO  | | REAL  | | MMU  |  "
                | MEM   | | PROM  | |      | | TIME  | | CTL  |  "
                |       | |       | |      | | CLK   | |      |  "
                +-------+ +-------+ +------+ +-------+ +------+  "
                                     | | |                       "
                                     | | +--keyboard             "
                                     | +---SIO 1                 "
                                     +-----SIO 2                 "
                                                                 "
                                                                 "
         <====================================================== "
               |         |        |         |        |
               |         |        |         |        |
           +-------+ +-------+ +------+ +-------+ +-------+
           | DISP  | | DISP  | | DMA  | | RING  | | DSK/  |
           | MEM   | | CTL   | | CTL  | | CTL   | | FLPY  |
           |       | |       | |      | |       | | CTL   |
           +-------+ +-------+ +------+ +-------+ +-------+
                         |                  |        | |
                         +                  +        | +- disk
                     monitor             network     +--- floppy
```

DISK (FLOPPY/WINCHESTER) CONTROLLER

```
        Address: [ 9C00 | 0FFA800 ]

                      WRITE                      READ

        +00 |    ANSI COMMAND    |      | ATTENTION STATUS  |
            |                    |      |                   |
        +02 |   ANSI PARM OUT    |      |    ANSI PARM IN   |
            |                    |      |                   |
        +04 |                    |      | DRIVE # OF STATUS |
            |                    |      |                   |
        +06 |       SECTOR       |      |CONTROLLER  STAT-HI|
        +07 |                    |      |CONTROLLER  STAT-LO|
        +08 |    CYLINDER-HI     |      |                   |
        +09 |    CYLINDER-LO     |      |                   |
        +0A |       HEAD         |      |                   |
            |                    |      |                   |
        +0C | INTERRUPT CONTROL  |      |                   |
            |                    |      |                   |
        +0E |  CONTROLLER CMND   |      |                   |
            +===============================================+
        +10 |                    |      |   FLOPPY STATUS   |
            |                    |      |                   |
        +12 | FLOPPY WRITE DATA  |      | FLOPPY READ DATA  |
            |                    |      |                   |
        +14 |  FLOPPY CONTROL    |      |                   |
            +===============================================+
        +20 | CALENDAR CONTROL   |      |                   |
            |                    |      |                   |
        +22 |CALENDAR WRITE DATA |      |                   |
            |                    |      |                   |
        +24 |                    |      |CALENDAR READ DATA |
```

CONTROLLER STATUS [9C06 | 0FFA806]

			Reset by
8000	15	Controller busy	Self clearing
4000	14	Drive busy (from bus)	Self clearing
2000	13	Drive attention (from bus)	Cntr, if status avail enab
1000	12	Status available interrupt	Read attn status reg
0800	11	End of operation interrupt	Write to ctlr cmd reg
0400	10	Floppy interrupting	Read floppy status reg
0080	07	Timeout	Write to ctlr cmd reg
0040	06	Overrun	Write to ctlr cmd reg
0020	05	CRC error	Write to ctlr cmd reg
0010	04	Controller bus parity error	Write to ctlr cmd reg
0008	03	Illegal configuration	Write to ctlr cmd reg
0004	02	Status timeout	Read attention status register
0002	01	Parity error during DMA	Write to controller command register

ANSI COMMANDS [9000 | 0FFA800]

PARAMETER OUT COMMANDS

		Parameter	
40	Attention control	Bit 7 = 0 => enable attention 1 => disable attention	
41	Write control	Bit 7 = 0 => write protect 1 => write enable	
42	Load Cyl. Addr. high	MSB of cylinder address	
43	Load Cyl. Addr. low	LSB of cylinder address	
44	Select head	Head number	Mandatory
50	Load attribute number	Attribute number	Optional
51	Load attribute	Attribute	
53	Read control	Bits 7,6 = 0x - nominal strobe 10 - strobe early 11 - strobe late	
54	Offset control	Bits 7,6 = 0x - no offset 10 - offset forward 11 - offset reverse	
55	Spin control	Bit 7 = 0 - spin down 1 - spin up	
56	Load sect/trk high	MSB of sectors/track	
57	Load sect/trk medium	MedSB of sectors/track	
58	Load sect/trk low	LSB of sectors/track	
59	Load bytes/sect high	MSB of bytes/sector	
5A	Load bytes/sect medium	MedSB of bytes/sector	
5B	Load bytes/sect low	LSB of bytes/sector	
6B	Load read permit high	MSB of read enable on cyl >=	
6C	Load read permit low	LSB of read enable cyl	
6D	Load write permit high	MSB of write enable on cyl >=	
6E	Load write permit low	LSB of write enable cyl	
6F	Load test byte	Test byte	

PARAMETER IN COMMANDS

00	Report illegal command		General status	
01	Clear fault		General status	
02	Clear attention		General status	
03	Seek	*	General status	
04	Rezero	*	General status	
0D	Report sense byte 2		Sense byte 2	
0E	Report sense byte 1		Sense byte 1	
0F	Report general status		General status	Mandatory

10	Report drive attribute		Drive attribute	Optional
11	Set attention	*	General status	
14	Selective reset	*	General status	
15	Seek to landing zone	*	General status	
16	Reformat track	*	General status	
29	Report cyl. addr. high		MSB of cylinder address	
2A	Report cyl. addr. low		LSB of cylinder address	
2B	Report read permit high		MSB of cylinder address	
2C	Report read permit high		LSB of cylinder address	
2D	Report wrt permit high		MSB of cylinder address	
2E	Report wrt permit high		LSB of cylinder address	
2F	Report test byte		Test byte	

* Time dependent command, attention set on completion.

<u>ATTENTION STATUS</u> [9000 | 0FFA800]

				Cleared by
7	80	Normal completion	*	Clear attention command
6	40	Busy		Self clearing
5	20	Read sense byte 2		See sense byte 2
4	10	Read sense byte 1		See sense byte 1
3	08	Illegal parameter	*	Clear fault command
2	04	Illegal command	*	Clear fault command
1	02	Control bus error	*	Clear fault command
0	01	Not ready		Self clearing

* Zero to one transition sets attention.

<u>SENSE BYTE 1</u>

				mand/opt
7	80	Vendor unique errors	*	O
6	40	Other errors	*	O
5	20	Command reject	*	M
4	10	Speed error	*	O
3	08	R/W permit violation	*	O
2	04	Power fault	*	O
1	02	Read/write fault	*	M
0	01	Seek error	*	M

SENSE BYTE 1

				mand/opt
7	80	Vendor unique attns	*	O
6	40	In write-protected area		M
5	20	Attr. table modified	*	O
4	10	Dev rsrved to alt port		O
3	08	Forced release	*	O
2	04	Dev rsrved to this port		O
1	02	Ready transition	*	M
0	01	Initial state	*	M

* Zero to one transition sets attention.

INTERRUPT CONTROL [9C0C | 0FFA80C]

```
+---+---+---+---+---+---+---+---+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---+---+---+---+---+---+---+---+
                  |   |   |   |
                  |   |   |   Enable End of Op int
                  |   |   Enable Status Avail int
                  |   Enable Attention  int
                  Overall interrupt enable
```

CONTROLLER COMMANDS [9C0E | 0FFA80E }

00 - No-op
01 - Read record
02 - Write record
03 - Format track
04 - Seek
05 - Execute ANSI command sequence
06 - Execute drive select sequence
07 - Execute attention in sequence
08 - Select head

Any command clears the controller status register.

FLOPPY CONTROL [9C14 | 0FFA814]

```
+--------------------+
|           | 1 | 0 |
+--------------------+
              |   |
              |   0 - Read
              |   1 - Write
              |
              1 => Enable floppy interrupt
```

DISPLAY CONTROL AND STATUS REGISTER (DCSR)

DISPLAY REGISTERS [9400 | 0FF9800]

	WRITE	READ
+00	DISPLAY CONTROL	DISPLAY STATUS
+02	DEB	
+04	WSSY	
+06	WSSX	
+08	DCY	
+0A	DCX	
+0C	WSDY	
+0E	WSDX	

DISPLAY CONTROL REGISTER [9400 | 0FF9800]

8000	- GO (Start BLT operation)	15
0020	- Interrupt at end of frame	5
0010	- Interrupt at end of BLT operation	4
0008	- Increment Y coordinate	3
0004	- Increment X coordinate	2
0002	- Fill mode BLT operation	1
0001	- Enable display (blank if reset)	0

DISPLAY STATUS REGISTER [9400 | 0FF9800]

8000	- BLT operation in progress	15
0080	- End of frame interrupt	7
0002	- Reserved	1
0001	- Reserved	0

BLT REGISTERS

(Each has an address used for reading and a separate address used for writing.)

DESTINATION COUNT Y REGISTER [9414 | 0FF9814]

```
 15    9        0
+----------------+
|------CCCCCCCCCC|
+----------------+
```

CCCCCC = -1 - ABS(WDSY - WDEY)

= two's complement for number of lines in height of destination block.

DESTINATION COUNT X REGISTER [9416 | 0FF9816]

```
 15        5    0
+----------------+
|----------CCCCCC|
+----------------+
```

CCCCCC = -1 - ABS(WDSX/16 - WDEX/16)

= two's complement for number of 16-bit aligned words involved in x coordinate.

DESTINATION END BIT REGISTER [941C | 0FF981C]

```
 15      7   3  0
+----------------+
|--------0000EEEE|
+----------------+
```

EEEE = WDEX mod 16

= bit number in word of last bit of X.

DMA CONTROLLER

DMAC page at [9000 | 0FFA000]

DMA controller is a Motorola M68450.

9000-903F - ring receive header
9040-907F - ring receive data
9080-90BF - ring transmit
90C0-90FF - winchester/floppy

Register summary (for each channel):

+00	Channel status register	(CSR)	R/W
+01	Channel error register	(CER)	R
+04	Device control register	(DCR)	R/W
+05	Operation control register	(OCR)	R/W
+06	Sequence control register	(SCR)	R/W
+07	Channel control register	(CCR)	R/W
+0A	Memory transfer counter	(MTC)	R/W
+0C	Memory address register	(MAR)	R/W
+14	Device address register (not used)		
+1A	Base transfer counter	(BTC)	R/W
+1C	Base address register	(BAR)	R/W
+25	Normal interrupt vector (not used)		
+27	Error interrupt vector (not used)		
+29	Memory function code register	(MFCR)	R/W
+2D	Channel priority register	(CPR)	R/W
+31	Device function code register	(not used)	
+39	Base function code register	(BFCR)	R/W
+FF	General Control Register (not used)		

CHANNEL STATUS REGISTER (CSR) [9000 | 0FFA000]

```
  7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+
|COC|BTC|NDT|ERR|ACT| 0 |PTC|PCS|
+---+---+---+---+---+---+---+---+
  |   |   |   |   |       |   |
  |   |   |   |   |       |   State of input PCL line
  |   |   |   |   |       1 => PCL transition occurred (*)
  |   |   |   |   1 => Channel active
  |   |   |   1 => Error as coded in CER (**)
  |   |   1 => Normal Device termination (*)
  |   1 => Block transfer complete and continue (*)
  1 => Channel operation complete (*)
```

(*) Bit cleared by writing a 1 bit to CSR.
(**) Ditto, and clearing also clears CER.

CHANNEL ERROR REGISTER (CER) [9001 | 0FFA001]

```
  7   6   5   4   3   2   1   0
+---+---+---+--------------------+
| 0 | 0 | 0 | E R R O R   C O D E|
+---+---+---+--------------------+

00 - No error
01 - Configuration error
02 - Operation timing error
03 - (undefined, reserved)
05 - Address error: memory address or memory counter
06 - Address error: device address
07 - Address error: base address or base counter
09 - Bus error: memory address or memory counter
0A - Bus error: device address
0B - Bus error: base address or base counter
0D - Count error: memory address or memory counter
0E - Count error: device address
0F - Count error: base address or base counter
10 - External abort
11 - Software abort
```

DEVICE CONTROL REGISTER (DCR) [9004 | 0FFA004]

```
  7   6   5   4   3   2   1   0
+-------+-------+---+---+-------+
|  XRM  |  DTYP |DPS| 0 |  PCL  |        (=28)
+-------+-------+---+---+-------+
  |   |   |   |   |       |   |
  |   |   |   |   |       0   0 - PCL = Status input
  |   |   |   |   |
  |   |   |   |   1 - 16-bit port
  |   |   |   |
  |   |   1   0 - Device with ACK, implicitly addressed
  |   |
  0   0 - Burst mode transfers
```

OPERATION CONTROL REGISTER (OCR) [9005 | 0FFA005]

```
  7   6   5   4   3   2   1   0
+---+---+-------+-------+-------+
|DIR| 0 |  SIZE | CHAIN |  REQG |        (=92)
+---+---+-------+-------+-------+
  |       |   |   |   |   |   |
  |       |   |   |   |   1   0 - REQ line initiates xfer
  |       |   |   |   |
  |       |   |   0   0 - Chain operation disabled
  |       |   |
  |       0   1 - Word transfers
  |
  0 - Transfer from memory to device
  1 - Transfer from device to memory
```

SEQUENCE CONTROL REGISTER (SCR) [9006 | 0FFA006]

```
  7   6   5   4   3   2   1   0
+---+---+---+---+-------+-------+
| 0 | 0 | 0 | 0 |  MAC  |  DAC  |        (=04)
+---+---+---+---+-------+-------+
                  |   |   |   |
                  |   |   0   0 - N/A (Device address reg)
                  |   |
                  0   1 - Memory address reg counts up
```

CHANNEL CONTROL REGISTER (CCR) [9007 | 0FFA007]

```
  7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+
|STR|CNT|HLT|SAB|INT| 0 | 0 | 0 |
+---+---+---+---+---+---+---+---+
  |   |   |   |   |
  |   |   |   |   1 => Enable interrupts
  |   |   |   |
  |   |   |   1 => Software abort
  |   |   |
  |   |   1 => Halt operation
  |   |
  |   1 => Continue opration
  |
  1 => Start operation
```

MEMORY TRANSFER COUNTER (MTC) [900A-900B|0FFA00A-0FFA00B]

```
 15       8         0
+---------+---------+
| W O R D C O U N T |
+---------+---------+
```

(E.g., 512 to transfer a page.)

MEMORY ADDRESS REGISTER (MAR) [900C-900F|0FFA00C-0FFA00F]

```
 31      24       16        8        0
+--------+--------+--------+--------+
|  HIGH  | UP-MID | LO-MID |   LO   |
+--------+--------+--------+--------+
 9018     901A     901C     901E
```

Load with MOVEP.L A0,9018.

DEVICE ADDRESS REGISTER (DAR) [9014-9017|0FFA014-0FFA017]

Not used.

BASE TRANSFER COUNTER (BTC) [901A-901B|0FFA01A-0FFA01B]

Same as Memory Transfer Counter.

BASE ADDRESS REGISTER (BAR) [901C-901F | 0FFA01C-0FFA01F]

Same as Memory Address Register.

NORMAL INTERRUPT VECTOR REGISTER [9025 | 0FFA025]

Not used.

ERROR INTERRUPT VECTOR REGISTER [9027 | 0FFA027]

Not used.

MEMORY FUNCTION CODE REGISTER (MFCR) [9029 | 0FFA029]

```
  7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+
| 0 | 0 | 0 | 0 | 0 | F | F | F |
+---+---+---+---+---+---+---+---+
                      |   |   |
                      0   0   0 - ring transmit data
                      0   0   1 - ring transmit header
```

Function code not used on other channels.

CHANNEL PRIORITY REGISTER (CPR) [902D | 0FFA02D]

```
  7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+
| 0 | 0 | 0 | 0 | 0 | 0 | P | P |
+---+---+---+---+---+---+---+---+
                          |   |
              Channel 0:  0   0 - ring receive header
                                  (highest)
              Channel 1:  0   1 - ring receive data
              Channel 2:  1   0 - ring transmit
              Channel 3:  1   1 - Winchester/floppy
                                  (lowest)
```

DEVICE FUNCTION CODE REGISTER (DFCR) [9031 | 0FFA031]

Not used.

BASE FUNCTION CODE REGISTER (BFCR) [9039 | 0FFA039]

Not used.

FAULT FRAME

BUS/ADDRESS ERROR STACK FRAME FORMAT (DN300/320)

```
         15                     0
         +----------------------+ ---------  ---------
   +00   |   STATUS REGISTER    |     ^          ^
         +----------------------+     |          |
   +02   |       PROGRAM        |   SHORT        |
         +- - - - - - - - - - - +   FRAME        |
         |       COUNTER        |     |          |
         +----------------------+     |          |
   +06   |  FRAME FORMAT/VOR    |     V          |
         +----------------------+ ---------      |
   +08   |  SPEC. STATUS WORD   |                |
         +----------------------+                |
   +0A   |        FAULT         |                |
         +- - - - - - - - - - - +                |
         |       ADDRESS        |                |
         +----------------------+                |
   +0E   |  INTERNAL REGISTER   |                |
         +----------------------+               BUS
   +10   | DATA OUTPUT BUFFER   |              ERROR
         +----------------------+              FRAME
   +12   |  INTERNAL REGISTER   |                |
         +----------------------+                |
   +14   |  DATA INPUT BUFFER   |                |
         +----------------------+                |
   +16   |  INTERNAL REGISTER   |                |
         +----------------------+                |
   +18   | INSTRUCTION REGISTER |                |
         +----------------------+                |
   +1A   |                      |                |
         /       INTERNAL       /                |
         /       REGISTERS      /                |
         |                      |                V
   +3A   +----------------------+ ------------------
```

BUS/ADDRESS ERROR STACK FRAME FORMAT (DN330)

```
         15      (medium)        0          15      (long)         0
        +----------------------+          +----------------------+
    +00 |   STATUS REGISTER    |      +00 |   STATUS REGISTER    |
        +----------------------+          +----------------------+
    +02 |       PROGRAM        |      +02 |       PROGRAM        |
        +- - - - - - - - - - - -+          +- - - - - - - - - - - -+
        |       COUNTER        |          |       COUNTER        |
        +----------------------+          +----------------------+
    +06 | 1010 0000 0000 1000  |      +06 | 1011 0000 0000 1000  |
        +----------------------+          +----------------------+
    +08 |  internal register   |      +08 |  internal register   |
        +----------------------+          +----------------------+
    +0A |  SPEC. STATUS WORD   |      +0A |  SPEC. STATUS WORD   |
        +----------------------+          +----------------------+
    +0C |  inst pipe stage c   |      +0C |  inst pipe stage c   |
        +----------------------+          +----------------------+
    +0E |  inst pipe stage b   |      +0E |  inst pipe stage b   |
        +----------------------+          +----------------------+
    +10 |     DATA  FAULT      |      +10 |     DATA  FAULT      |
        +- - - - - - - - - - - -+          +- - - - - - - - - - - -+
        |       ADDRESS        |          |       ADDRESS        |
        +----------------------+          +----------------------+
    +14 |  internal registers  |      +14 |  internal registers  |
        +----------------------+          +----------------------+
    +18 |     DATA OUTPUT      |      +18 |     DATA OUTPUT      |
        +- - - - - - - - - - - -+          +- - - - - - - - - - - -+
        |       BUFFER         |          |       BUFFER         |
        +----------------------+          +----------------------+
    +1C |  internal registers  |      +1C |  internal registers  |
    +20 +----------------------+          +----------------------+
                                      +24 |       STAGE B        |
                                          +- - - - - - - - - - - -+
    Real fault address:                   |       ADDRESS        |
                                          +----------------------+
    DF => data fault address          +28 |  internal registers  |
                                          +----------------------+
    FB => if medium then PC + 4       +2C |     DATA  INPUT      |
          if long then stage b            +- - - - - - - - - - - -+
          address                         |       BUFFER         |
    FC => if medium then PC + 2           +----------------------+
          if long then stage b 2      +30 |                      |
          addr -2                         /       internal       /
                                          /       registers      /
                                          |                      |
                                      +5C +----------------------+
```

Coprocessor Mid-instruction frame

```
 0      2            6      8          C                      14
+------+------------+------+-----------+-----------------------+
|  SR  |     PC     | 9xxx | Instr Addr |  Internal Registers   |
+------+------------+------+-----------+-----------------------+
```

FRAME FORMAT/VECTOR OFFSET WORD (DN300/320)

```
 15 12     9          0
+-----+-+-+------------+
| FMT |0|0| VEC OFFSET |
+-----+-+-+------------+

 0000 - four-word format: SR, PC, VOR
 1000 - 29-word 68010 format.
```

FRAME FORMAT/VECTOR OFFSET WORD (DN330)

```
 15 12     9          0
+-----+-+-+------------+
| FMT |0|0| VEC OFFSET |
+-----+-+-+------------+
 0000 - 4-word format: SR, PC, VOR
 0001 - Throwaway (4 words)
 0010 - Instruction (trapv, chkv) (6 words)
 1000 - 29-word 68010 bus error
 1001 - Coprocessor mid-instruction (10 words)
 1010 - 68020 medium bus errur (16 words)
 1011 - 68020 long bus errur (46 words)
```

SPECIAL STATUS WORD (DN300/320)

```
15 14 13 12 11 10  9  8             3   0
+-----------+-----------+-----------+-----+
|RR| 0|IF|DF|RM|HB|BY|RW| 0| 0| 0| 0| FCN |
+-----------+-----------+-----------+-----+
 |     |  |  |  |  |  |                |
 |     |  |  |  |  |  |               001 - User data
 |     |  |  |  |  |  |               010 - User program
 |     |  |  |  |  |  |               101 - Supervisor data
 |     |  |  |  |  |  |               110 - Supervisor
 |     |  |  |  |  |  |                       program
 |     |  |  |  |  |  0 - Write       111 - Interrupt
 |     |  |  |  |  |  1 - Read                acknowledge
 |     |  |  |  |  |
 |     |  |  |  |  1 => Byte transfer
 |     |  |  |  |
 |     |  |  |  1 => High byte (valid iff BY on)
 |     |  |  |
 |     |  |  1 => Read-modify-write cycle
 |     |  |
 |     |  0 - Data store from DOB
 |     |  1 - Data fetch to DIB
 |     |
 |     1 => Instruction fetch
 |
 0 - Processor will rerun bus cycle on RTE
 1 - Software has completed the bus cycle prior to RTE
```

SPECIAL STATUS WORD (DN330)

```
15 14 13 12           8  7          3   0
+-----------+-----------+--+--+-----+-----+
|FC|FB|RC|RB| 0| 0| 0|DF|RM|RW| SIZ | FCN |
+-----------+-----------+--+--+-----+-----+
 |  |  |  |           |  |   |   |     |
 |  |  |  |           |  |   |   |    001 - User data
 |  |  |  |           |  |   |   |    010 - User program
 |  |  |  |           |  |   |   |    101 - Supervisor data
 |  |  |  |           |  |   |   |    110 - Supervisor prog
 |  |  |  |           |  |   |   |    111 - CPU space ref
 |  |  |  |           |  |   |   |
 |  |  |  |           |  |   |   |
 |  |  |  |           |  |   |  Size ----- 00 longword
 |  |  |  |           |  |   |             01 byte
 |  |  |  |           |  |  0 - Write       10 word
 |  |  |  |           |  |  1 - Read        11 3 byte
 |  |  |  |           |  |
 |  |  |  |           | Read-modify-write cycle
 |  |  |  |           |
 |  |  |  |          Fault/rerun flag for data cycle
 |  |  |  |
 |  |  | Rerun flag for Stage B
 |  |  |
 |  | Rerun flag for Stage C
 |  |
 | Fault on Stage B of instruction pipe
 |
Fault on Stage C of instruction pipe
```

FAULT TYPES

Group	Exception	Processing
0	Reset Bus Error Address Error	Current instruction is aborted.
1	Trace Interrupt Illegal Ins. Privilege Ins.	Exception occurs before next instruction.
2	TRAP, TRAPV CHK Zero Divide	Processed by normal instruction execution.

Group 0 exceptions have the highest priority.

FAULT VECTORS

Exception vector at [100400 | 0]
(+ => new on 68020, &=> new on 68010, - => unused)

	Vector	Address	Assignment		frame length		FF
					was	is(020s)	
	00	000	Reset: Initial SSP		-	-	
		004	Reset: Initial PC		-	-	
	02	008	Bus Error		3A	20,5C	8,A,B
	03	00C	Address Error		3A	20,5C	8,A,B
	04	010	Illegal Instruction		8	8	0
	05	014	Zero Divide		8	C	0,2
	06	018	CHK Instruction		8	C	0,2
	07	01C	TRAPV Instruction		8	C	0,2
	08	020	Privilege Violation		8	8	0
	09	024	Trace		8	C	0,2
	0A	028	Unimp. instruction (A-line)		8	8	0
	0B	02C	Unimp. instruction (F-line)		8	8	0
	0C	030	(Unassigned, reserved)		-	-	
+	0D	034	Coprocessor protocol violation		-	14	9
&	0E	038	Invalid Stack Format		8	8	0
	0F	03C	Uninitialized vector interrupt		-	-	
	10-17	040	(Unassigned, reserved)		-	-	
	18	060	Spurious Interrupt		8	8	0
&	19-1F	064	Level 1-7 Auto-Vector				
&	19	064	SIO (rcv and xmit)	1			
&	1A	068	Keyboard input	2			
&	1B	06C	Ring	3			
&	1C	070	Display	4			
&	1D	074	Disk/floppy	5			
&	1E	078	Timers 1,2,3	6			
&	1F	07C	Parity error	7			
	20-2F	080	TRAP Instruction Vectors		8	8	
+	30	0C0	FP Branch or Set on Unordered cond (FPBSUN)				
+	31	0C4	FP Inexact Result (FPINEX)				
+	32	0C8	FP Divide by Zero (FPDIVZ)				
+	33	0CC	FP Underflow (FPUNFL)				
+	34	0D0	FP Operand Error (FPOPER)				
+	35	0D4	FP Overflow (FLOVFL)				
+	36	0D8	FP Signalling NAN (FPSNAN)				
	37-3F	0DC	(Unassigned, reserved)		-	-	

FLOATING-POINT FORMAT

Single-Precision Floating-Point Format

```
         Implied normalization bit
         \/
 31       22                    0
+--------------------------------+
|SEEEEEEEEMMMMMMMMMMMMMMMMMMMMMMM|
+--------------------------------+

S    - Sign; S = 1 => Negative
E..E - Exponent plus 127
M..M - Mantissa
```

Double-Precision Floating-Point Format

```
            Implied normalization bit
            \/
 31          19                 0
+--------------------------------+
|SEEEEEEEEEEEMMMMMMMMMMMMMMMMMMMM|
+--------------------------------+
|MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM|
+--------------------------------+

S    - Sign; S = 1 => Negative
E..E - Exponent plus 1023
M..M - Mantissa
```

FLOATING-POINT REGISTERS (DN330)

Floating Point Control Register

```
   31                                            16
  +-----------------------------------------------+- - -
  |                      0                        |
  +-----------------------------------------------+- - -

       15 14 13 12 11 10  9  8  7  6  5  4  3  2  1  0
  - - -+--+--+--+--+--+--+--+--+--+--+--+--+-----------+
       |  |  |  |  |  |  |  |  |  |  |  |  |   unused  |
  - - -+--+--+--+--+--+--+--+--+--+--+--+--+-----------+
         |  |  |  |  |  |  |  | \___/ \___/
         |  |  |  |  |  |  |  |   |     +> rounding mode (rnd)
         |  |  |  |  |  |  |  |   |         00 to nearest
         |  |  |  |  |  |  |  |   |         01 toward zero
         |  |  |  |  |  |  |  |   |         10 toward minus infinity
         |  |  |  |  |  |  |  |   |         11 toward plus infinity
         |  |  |  |  |  |  |  |   > rounding precision (prec)
         |  |  |  |  |  |  |  |      00 extended
         |  |  |  |  |  |  |  |      01 single
         |  |  |  |  |  |  |  |      10 double
         |  |  |  |  |  |  |  |      11 (undefined, reserved) this is
         |  |  |  |  |  |  |  |          reactive precision on FPX
         |  |  |  |  |  |  |  > inexact decimal input (inex1)
         |  |  |  |  |  |  > inexact operation (inex2)
         |  |  |  |  |  > divide by zero (dz)
         |  |  |  |  > underflow (unfl)
         |  |  |  > overflow (ovfl)
         |  |  > operand error (operr)
         |  > signalling not a number (snan)
         > branch/set on unordered condition (bsun)
```

Floating Point Status Register (DN330)

```
|   Condition Code Byte    |   Quotient Byte    |

 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16
+-----------+--+--+--+--+--+--------------------+- - -
|  unused   |  |  |  |  |  |      quotient      |
+-----------+--+--+--+--+--+--------------------+- - -
              |  |  |  |  | \------------------/
              |  |  |  |  |         +> seven least significant
              |  |  |  |  |              bits of quotient
              |  |  |  |  > sign of quotient
              |  |  |  > not a number or unordered
              |  |  > infinity
              |  > zero
              > negative


     | Exception Status Byte | Accrued Exception Byte|

      15 14 13 12 11 10  9  8  7  6  5  4  3  2  1  0
- - -+--+--+--+--+--+--+--+--+--+--+--+--+--+--------+
     |  |  |  |  |  |  |  |  |  |  |  |  |  | unused |
- - -+--+--+--+--+--+--+--+--+--+--+--+--+--+--------+
       |  |  |  |  |  |  |  |  |  |  |  |  |
       |  |  |  |  |  |  |  |  |  |  |  |  > inexact (inex)
       |  |  |  |  |  |  |  |  |  |  |  > divide by zero (dz)
       |  |  |  |  |  |  |  |  |  |  > underflow (unfl)
       |  |  |  |  |  |  |  |  |  > overflow (ovfl)
       |  |  |  |  |  |  |  |  > invalid operation (iop)
       |  |  |  |  |  |  |  > inexact decimal input (inex1)
       |  |  |  |  |  |  > inexact operation (inex2)
       |  |  |  |  |  > divide by zero (dz)
       |  |  |  |  > underflow (unfl)
       |  |  |  > overflow (ovfl)
       |  |  > operand error (operr)
       |  > signalling not a number (snan)
       > branch/set on unordered condition (bsun)
```

FPU (DN320)

The FPU is essentially the same as PEB without the cache (and different microcode). Refer to the PEB section of Chapter 8, DN400,DN420,DN600.

MEMORY CONTROL/STATUS REGISTERS (MCSR)

Memory Control Register (DN300/320)

```
Address: [ 8005 | 0FFB405 ]
+---+---+---+---+---+---+---+---+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---+---+---+---+---+---+---+---+
 \ ____________/       |   |   |
       LEDS            |   |   1 => Enable parity err traps
 TOP         BOT       |   |
 (RIGHT)  (LEFT)       |   1 => Force right byte parity
                       |
                       1 => Force left byte parity
```

Memory Status Register (DN300/320)

```
Address:  [ 8006 | 0FFB406 ]
15          4                0
+-------------+---+---+---+---+
| FAILING PPN | 3 | 2 | 1 | 0 |
+-------------+---+---+---+---+
                |   |   |   |
                |   |   |   1 => Parity error traps enabled
                |   |   |
                |   |   1 => Right byte parity error
                |   |
                |   1 => Left byte parity error
                |
                1 = > Parity during DMA cycle
```

Writing MSR clears parity error condition.

Memory Control/Status [8004-8007|3FFB404-3FFB407] (Byte writeable) (DN330)

```
|------R/O------|------CLR------|--R/W--|--R/O--|
 31           12 11            8               4               0
+---------------+---+---+---+---+---+---+---+---+---+---+---+---+
|FAILING A(21:2)|   |   |   |   |   |   |   |   |   |   |   |   |
+---------------+---+---+---+---+---+---+---+---+---+---+---+---+
     20 bit       3   2   1   0   |   |   |   |   |
                \_______________/ |   |   |   |   |
                  Byte Parity     |   |   |   |   |
                  Error Flags     |   |   |   |   Failing addr bit 1
                  (0 => error)    |   |   |   |
                                  |   |   |   1 => B Port Access
                                  |   |   |
                                  |   |   1 => DMA Access
                                  |   |
                                  |   1 => Parity Interrupt Enable
                                  |
                                  1 => Write Bad Parity
```

LEDS/Hardware Rev Register [8008-8009|3FFB408-3FFB409] (DN330)

```
 15               8 7              0
+-----------------+----------------+
| L E D S x x x x | H D W R  R E V |
+-----------------+----------------+
```

MEMORY MANAGEMENT UNIT (MMU) (DN300/320)

```
PID/PRIV/POWER       [ 8000 | FFB400 ]
MMU Status           [ 8002 | FFB402 ]
```

PID/PRIV Register

```
Address: [8000 | 0FFB400 ]
     15         8                 0
    +---+--------+-----+---+---+---+
    | 0 |  ASID  |-----| D | P | M |
    +---+--------+-----+---+---+---+
                     |   |   |
                     |   |   1 - Enable MMU
                     |   |
                     |   1 - Enable PTT access
                     |
                     0 - DOMAIN 0
                     1 - DOMAIN 1
```

MMU Status Register

```
Address: [ 8002 | 0FFB402 ]
    +---+---+---+---+---+---+---+---+
    | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
    +---+---+---+---+---+---+---+---+
      |   |   |   |   |   |   |   |
      |   |   |   |   |   |   |   1 => MMU enabled
      |   |   |   |   |   |   |
      |   |   |   |   |   |   1 => PTT access enabled
      |   |   |   |   |   |
      |   |   |   |   |   1 => UNUSED
      |   |   |   |   |
      |   |   |   |   1 => Interrupt pending
      |   |   |   |
      |   |   |   1 => Normal mode
      |   |   |
      |   |   1 => Bus timeout
      |   |
      |   1 => Page fault
      |
      1 => Access violation
```

MEMORY MANAGEMENT UNIT (MMU) (DN330)

PID/PRIV/POWER	[8000 | 3FFB400]
MMU Status	[8002 | 3FFB403]

PID/PRIV Register

```
    Address: [8000 | 03FFB400 ]

 15         8                     0
|----W/O------|     |R/O|----R/W----|
+---+---------+-----+---+---+---+---+
| 0 |  ASID   |-----| F | D | P | M |
+---+---------+-----+---+---+---+---+
                      |   |   |   |
                      |   |   |   1 - Enable MMU (R/W - was W/O)
                      |   |   |
                      |   |   1 - Enable PTT access
                      |   |
                      |   0 - Domain 0
                      |   1 - Domain 1
                      |
                     FP Trap = 1 if trap occurred
                             Cleared by write to FPU Owner Reg.
```

FPU OWNER REGISTER [8002 | 3FFB402]

```
            7   6                  0
               |------W/O-----------|
           +---+--------------------+
           | X |  ASID of FPU OWNER |
           +---+--------------------+
             |
             Unused
```

MMU Status Register [8003 | 03FFB403]

```
+---+---+---+---+---+---+---+---+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---+---+---+---+---+---+---+---+
  |   |   |   |   |   |   |   |
  |   |   |   |   |   |   |   0 => Stingray 020 board (R/O)
  |   |   |   |   |   |   |  (was MMU enabled - changed for O/S)
  |   |   |   |   |   |   1 => PTT access enabled
  |   |   |   |   |   |
  |   |   |   |   |   orderly shutdown (toggle)
  |   |   |   |   |
  |   |   |   |   1 => MMU Error (Timeout or Parity Error)
  |   |   |   |        valid only if bit 5 set (was Int pending)
  |   |   |   1 => Normal mode
  |   |   |
  |   |   1 => Bus/MMU timeout or MMU Parity Error
  |   |        (was Bus Timeout only)
  |   1 => Page fault
  |
  1 => Access violation

   [Any write to register clears bits 5-7]
```

MMU Parity Register [800A-800B]

```
|--R/W--|--CLR--|----------------R/O---------------------------|
 15           12 11                                            0
+---+---+---+---+----------------------------------------------+
|   |   |   |   |       PFTX (PFT Parity Error Index)          |
+---+---+---+---+----------------------------------------------+
  |   |   |   |
  |   |   |   |
  |   |   |   1 => PFT Parity Error
  |   |   |
  |   |   1 => PTT Parity Error
  |   |
  |   1 => MMU Parity Fault Enable (MMU PFE)
  |
  1 => Write Wrong MMU Parity (Both PFT and PTT)
      [Bus error occurs on parity error in normal MMU
    operation if MMU PFE is set,
       and bits 12 and 13 WERE clear]
```

Diagnostic Loopback Register [800C-800F] - Loopback of SBUS signals - TBD

PAGE FRAME TABLE ENTRY (PFTE)

(type "pfte" in mmpft.pvt.pas)

```
 31     24 23     16 15      8 7       0
+--------+--------+--------+--------+
|AAAAAAAS|DWRXPPPP|EMUGLLLL|LLLLLLLL|
+--------+--------+--------+--------+
```

```
A..A - Address space ID (0-127) (.elsid)
S    - Supervisor domain (.elccess)
D    - DOMAIN (0 or 1)
W    - Write access
R    - Read access
X    - Execute access
P..P - Excess virtual page number (.xsvpn)
E    - End of chain (.eoc)
M    - Page modified (.bbmod)
U    - Page referenced (.used)
G    - Page is global (.global)
L..L - PFT hash thread (.link)
```

PFT at [4000 | FFB800] through [8000 | FFF800].

There is one entry per physical page of memory.

PAGE TRANSLATION TABLE ENTRY (PTTE)

(type "ppn_t" in base.ins.pas)

```
 15              0
+----------------+
|XXXXPPPPPPPPPPPP|
+----------------+
```

P..P - Physical page number (PPN)

XXX - Junk - ignore

Page Translation Table at [n/a | 700000]
 through [n/a | 800000]

One PPTE every 1024 bytes in table.

PEB

Refer to the FPU section.

PROM ENTRY POINTS

```
100:    dc.w  2,0            2 => swallow, no aux info
104:    ac    getc           returns char in D1
108     ac    putc           prints char in D1
10C     ac    init_dsk       initialize disk
110     ac    read_dsk       read a record from disk
114     ac    reload_font    reload font
118     ac    pollc          returns char in D1, else -1 in d1.w
11C     ac    quiet_ret      quiet return to prom
```

RING REGISTERS

RING page at [9800 | 0FF9C00]

```
          WRITE FUNCTION                        READ FUNCTION

       15                      0           15                      0
      +------------------------+           +------------------------+
  +00 |      XMIT COMMAND      |       +00 |      XMIT STATUS       |
      +------------------------+           +------------------------+
  +02 |       RCV COMMAND      |       +02 |       RCV STATUS       |
      +-----------+------------+           +-----------+------------+
  +04 |   TMASK   |   UNUSED   |       +04 |   TMASK   |   UNUSED   |
      +-----------+------------+           +-----------+------------+
  +06 |      DIAG COMMAND      |       +06 |      DIAG STATUS       |
      +------------------------+           +------------------------+
  +08 |          RING          |       +08 |          RING          |
      +- - - - - - - - - - - --+           +- - - - - - - - - - - --+
      |           ID           |           |           ID           |
      +------------------------+           +------------------------+
                                       +0C |         UNUSED         |
                                           +------------------------+
                                       +0E |         UNUSED         |
                                           +-----------+------------+
                                       +10 |    ID3    |   UNUSED   |
                                           +-----------+------------+
                                       +12 |    ID2    |   UNUSED   |
                                           +-----------+------------+
                                       +14 |    ID1    |   UNUSED   |
                                           +-----------+------------+
                                       +16 |    ID0    |   UNUSED   |
                                           +-----------+------------+
```

TRANSMIT COMMAND [9800 | 0FF9C00]

4000	transmit interrupt enable	
2000	transmit enable	(start the transmit)
1000	force transmit	

NOTES:

1. To start a tramsmit normally, use 6000.
2. To force transmit, use 7000.
3. To stop a transmit that has already started, clear the transmit enable bit.
4. Writing anything to this register clears the transmit interrupt.

RECEIVE COMMAND [9802 | 0FF9C02]

4000	enable interrupt	
2000	enable receive	(start the receive)

NOTES:

1. To start a normal receive, use 6000.
2. To stop a receive that has already started, clear the receive enable bit.

TRANSMIT STATUS [9800 | 0FF9C00]

8000	interrupt pending	
4000	interrupt enabled	
2000	busy	
1000	disconnected	
0800	bi-phase error	
0400	elastic store buffer error	
0200	no return	(a complete pkt frame never arrived)
0100	crc error	
0080	ack parity error	(0=no error, 1=error detected)
0040	external error	(err during DMA, e.g. parity, bus-error)
0020	protocol error	(the pkt hdr with FROM ID never came back)
0010	icopy	(somebody Intended to COPY -- was willing to rcv)
0008	ack byte errbit	(somebody (anybody!) set the "error detected" bit)
0004	copy	(somebody did COPY the pkt)
0002	wack	
0001	underrun	(DMA didn't keep up with xmit data rate)

NOTES:

1. A successful transmit will have a transmit status of 0014
2. A WACK will have a transmit status of 0012

RECEIVE STATUS [9802 | 0FF9C02]

8000	interrupt pending	
4000	interrupt enabled	
2000	busy	
1000	disconnected	
0800	bi-phase error	
0400	elastic store buffer error	
0200	timeout (The hdr of a msg was seen, but it never ended)	
0100	crc error	
0080	ack parity error (0=no error, 1=error detected)	
0040	external error (err during DMA, e.g. parity, bus-error)	
0020	DMA end of range	
0010	icopy	(somebody before me Intended to COPY)
0008	ack byte errbit	(somebody before me set the "error detected" bit)
0004	copy	(somebody before me did COPY the pkt)
0002	wack	(somebody before me WACKed the ptk)
0001	overrun	(DMA didn't keep up with rcv data rate)

DIAGNOSTIC STATUS [9806 | 0FF9C06]

8000	interrupt pending	(bad_pkt_cnt_overflow interrupt)
4000	interrupt enabled	(bad_pkt_cnt_overflow interrupt)
2000	connected to the network	
1000	sticky bi-phase error	(error seen since bit was cleared)
0800	delay on	(the delay is enabled)
0400	sticky good_seen	(good pkt seen since bit was cleared)
0200	sticky elastic store bfr err	(error seen since bit was cleared)
01FF	bad packet count	(9-bit counter for 1st detecting errs)

NOTES:

1. Counter is number of times this node found an error in a packet going by (regardless of packet target node ID), found the error bit in the ackbyte clear, and so was the first to set the error bit to a one.

2. The bad_pkt_cnt interrupt occurs when the counter counts from 255 to 256 (i.e. first uses its highest order bit).

3. The counter sticks at 511 if more than 511 errors are seen.

4. Writing anything to the diagnostic command register (word) (see below) clears interrupt and all sticky bits.

DIAGNOSTIC COMMAND [9806 | 0FF9C06]

8000	dma test	(loop xmit DMA to rcv DMA)
4000	enable interrupt	(bad_pkt_cnt overflow intterupt)
2000	connect	(to the network)
1000	disconnect	(from the network)
0800	delay off	(disable the delay)
0400	delay on	(enable the delay)
0200	snoop	(accept all pkts but only set ackbyte for packets actually addressed to me)

NOTE:
Writing anything to the register (word) clears interrupt and all sticky bits in diagnostic status register.

TMASK [9804 | 0FF9C04]

80	broadcast
40	hardware diagnostic
20	thank you
10	please
08	paging
04	user
02	software diagnostic
01	xtype3

NOTE:
Except for BROADCAST, these bits are software defined.

SERIAL I/O INTERFACE

SIO page at [8400 | 0FFB000]

The SIO lines are implemented with a Signetics SC2681 DUART. The display keyboard interface implemented with a Motorola MC6850.

When both SIO lines are being used, it is possible to have incompatible baud rates due to limitations of the SC2681 chip. One SIO line can't have a baud rate from Group A while the other SIO line is set from Group B:

Group A	Group B
50	75
7200	150
	2000
	19.2K

Register summary:

		READ	WRITE
8400	0FFB000	Mode Reg. A (MRA)	Mode Reg. A (MRA)
8402	0FFB002	Status Reg. A (SRA	Clock Select Reg. A (CSRA)
8404	0FFB004	---	Command Reg A (CRA)
8406	0FFB006	Rcv Hld Reg. A (RHRA)	Transmit Hld Reg. A (THRA)
8408	0FFB008	Input Port Change Reg. (IPCR)	Aux. Control Reg. (ACR)
840A	0FFB00A	Interrupt Status Reg. (ISR)	Interrupt Mask Reg. (IMR)
8410	0FFB010	Mode Reg. B (MRB)	Mode Register B (MRB)
8412	0FFB012	Status Reg. B (SRB)	Clock Select Reg. B (CSRB)
8414	0FFB014	---	Command Reg B (CRB)
8416	0FFB016	Rcv Hld Reg. B (RHRB)	Transmit Hld Reg. B (THRB)
841A	0FFB01A	Input Port Register (IPR)	Output Port Config. Reg. (OPCR)
841C	0FFB01C	---	Set Output Port Reg. (OPR)
841E	0FFB01E	---	Reset Output Port Reg. (OPR)
8420	0FFB020	Display Keyboard Status/Command Register	
8422	0FFB022	Display Keyboard Data I/O Register	

MODE REGISTER A, first access [8400 | 0FFB000]

```
+---------------+---------------+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---------------+---------------+
  |   |   |   |   |   |   |   |
  |   |   |   |   |   |   0   0 = 5 bits/char
  |   |   |   |   |   |   0   1 = 6     ''
  |   |   |   |   |   |   1   0 = 7     ''
  |   |   |   |   |   |   1   1 = 8     ''
  |   |   |   |   |   |
  |   |   |   |   |   0 = even parity
  |   |   |   |   |   1 = odd     ''
  |   |   |   |   |
  |   |   |   0   0 = check parity
  |   |   |   0   1 = force parity
  |   |   |   1   0 = no parity
  |   |   |   1   1 = special multidrop mode
  |   |   |
  |   |   0 = report error on each char
  |   |   1 = accumulate error info since last reset err
  |   |                                            commmand
  |   0 = interrupt on receiver ready
  |   1 = interrupt on input FIFO full
  |
  1 = drop RTS (OP0) when input FIFO is full
```

MODE REGISTER A, second & subsequent accesses

(until mode register pointer reset)

```
+---------------+---------------+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---------------+---------------+
  |   |   |   |   |   |   |   |
  |   |   |   |   0   1   1   1 = 1   stop bit
  |   |   |   |   1   0   0   0 = 1.5 stop bits
  |   |   |   |   1   1   1   1 = 2   stop bits
  |   |   |   |
  |   |   |   0 = transmit regardless of CTS (IP0)
  |   |   |   1 = wait for CTS to transmit
  |   |   |
  |   |   0 = leave RTS as is
  |   |   1 = drop RTS (OP0) after transmitter disabled
  |   |
  0   0 = normal mode
  0   1 = auto echo mode
  1   0 = local loop mode
  1   1 = remote loop mode
```

STATUS REGISTER A [8402 | 0FFB002] read-only

```
+---------------+---------------+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---------------+---------------+
  |   |   |   |   |   |   |   |
  |   |   |   |   |   |   |   1 = input data ready
  |   |   |   |   |   |   |       (reset by reading RHR)
  |   |   |   |   |   |   |
  |   |   |   |   |   |   1 = input FIFO full (reset when
  |   |   |   |   |   |       RHR and no data in shift reg)
  |   |   |   |   |   |
  |   |   |   |   |   1 = transmitter ready (reset when
  |   |   |   |   |       THR loaded)
  |   |   |   |   1 = transmitter underrun (reset when
  |   |   |   |       THR loaded)
  |   |   |   1 = rcver overrun (reset by reset error
  |   |   |       status cmd)
  |   |   1 = rcv parity error (reset by reset error
  |   |       status cmd)
  |   1 = receive framing error (reset by reset err status
  |       cmd)
  1 = break received  (reset by ???)
```

CLOCK SELECT REGISTER A [8402 | 0FFB002] write-only

```
 7            4 3              0
+---------------+----------------+
| receive clock | transmit clock |
+---------------+----------------+
```

	ACR[7]=0	ACR[7]=1		ACR[7]=0	ACR[7]=1
0 -	50	75	8 -	2400	2400
1 -	110	110	9 -	4800	4800
2 -	134.5	134.5	A -	7200	1800(??)
3 -	200	150	B -	9600	9600
4 -	300	300	C -	38.4K	19.2K
5 -	600	600	D -	Timer	Timer
6 -	1200	1200	E -	IP4-16X	IP4-16X
7 -	1050	2000	F -	IP4-1X	IP4-1X

COMMAND REGISTER A [8404 | 0FFB004] write-only

```
+---------------+---------------+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---------------+---------------+
  |   |   |   |   |   |   |   |
  |   |   |   |   |   |   |   1 = enable receiver
  |   |   |   |   |   |   |
  |   |   |   |   |   |   1 = disable receiver
  |   |   |   |   |   |
  |   |   |   |   |   1 = enable transmitter
  |   |   |   |   |
  |   |   |   |   1 = disable transmitter
  |   |   |   |
  |   0   0   0 = no-op
  |   0   0   1 = reset mode register pointer
  |   0   1   0 = reset & disable receiver
  |   0   1   1 = reset transmitter
  |   1   0   0 = reset error status
  |   1   0   1 = reset break-change interrupt
  |   1   1   0 = start transmitting break
  |   1   1   1 = stop  tranmsmitting break
  |
 must be zero
```

RECEIVE/TRANSMIT HOLDING REGISTER A [8406 | 0FFB006]

```
+-------------------------------+
|         D   A   T   A         |
+-------------------------------+

     READ  = top byte in input FIFO
     WRITE = byte of data to transmit
```

INPUT PORT CHANGE REGISTER [8408 | 0FFB008] read-only

```
 change in IPx:    state of IPx:
+---------------+---------------+
| 3 | 2 | 1 | 0 | 3 | 2 | 1 | 0 |
+---------------+---------------+

    IP0 - Clear To Send (CTS) - A
    IP1 - Clear To Send (CTS) - B
    IP2 - Data Carrier Detect (DCD) - A
    IP3 - Data Carrier Detect (DCD) - B
```

AUXILIARY CONTROL REGISTER   [8408 | 0FFB008] write-only

```
+---------------+---------------+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---------------+---------------+
  |   |   |   |   |   |   |   |
  |   |   |   |   |   |   |   enab int on change in IP0
  |   |   |   |   |   |   |
  |   |   |   |   |   |   enab int on change in IP1
  |   |   |   |   |   |
  |   |   |   |   |   enab int on change in IP2
  |   |   |   |   |
  |   |   |   |   enab in on change in IP3
  |   |   |   |
  |   counter/timer stuff
  |
  baud rate generator set select (see clock select reg)
```

INTERRUPT STATUS REGISTER   [840A | 0FFB00A] read-only

```
+---------------+---------------+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---------------+---------------+
  |   |   |   |   |   |   |   |
  |   |   |   |   |   |   |   transmitter ready A
  |   |   |   |   |   |   +-rcver rdy or inpt FIFO full A
  |   |   |   |   |   +-----beginning or end of break A
  |   |   |   |   |
  |   |   |   |   +---------counter ready (not used)
  |   |   |   |
  |   |   |   transmitter ready B
  |   |   +---receiver ready or input FIFO full B
  |   +-------beginning or end of break B
  |
  +-----------input port change status
```

INTERRUPT MASK REGISTER   [840A | 0FFB00A] write-only

```
+---------------+---------------+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---------------+---------------+
```

Each bit enables the corresponding bit
in the Interrupt Status Register.

MODE REGISTER B   [8410 | 0FFB010]

See MODE REGISTER A.

STATUS/CLOCK SELECT REGISTER B   [8412 | 0FFB012]

See STATUS/CLOCK SELECT REGISTER A.

COMMAND REGISTER B [8414 | 0FFB014]

See COMMAND REGISTER A.

RECEIVE/TRANSMIT HOLDING REGISTER B [8416 | 0FFB016]

See RECEIVE/TRANSMIT HOLDING REGISTER A.

INPUT PORT REGISTER [841A | 0FFB01A] read-only

```
+-----------------------+-----------------------+
| IP7 | IP6 | IP5 | IP4 | IP3 | IP2 | IP1 | IP0 |
+-----------------------+-----------------------+
```

```
    IP0 - Clear To Send (CTS) - A
    IP1 - Clear To Send (CTS) - B
    IP2 - Data Carrier Detect (DCD) - A
    IP3 - Data Carrier Detect (DCD) - B
    IP4-IP7 - Undefined
```

OUTPUT PORT CONFIGURATION REGISTER [841A | 0FFB01A]

write-only

```
+---------------+---------------+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---------------+---------------+
```

Load with 0. This selects:

```
        OP0 = Ready To Send (RTS) A
        OP1 = Ready To Send (RTS) B
        OP2 = Data Terminal Ready (DTR) A
        OP3 = Data Terminal Ready (DTR) B
        OP4 - OP6 - Unused
        OP7 = Speaker control
```

SET OUTPUT PORT REGISTER (OPR) [841C | 0FFB01C]

write-only

```
+-----------------------+-----------------------+
| OP7 | OP6 | OP5 | OP4 | OP3 | OP2 | OP1 | OP0 |
+-----------------------+-----------------------+
```

```
    OP0 - Ready To Send (RTS) A
    OP1 - Ready To Send (RTS) B
    OP2 - Data Terminal Ready (DTR) A
    OP3 - Data Terminal Ready (DTR) B
    OP4-OP6 - Unused
    OP7 - Turn off speaker
```

RESET OUTPUT PORT REGISTER (OPR) [841E | 0FFB01E]

write-only

```
+-----------------------+-----------------------+
| OP7 | OP6 | OP5 | OP4 | OP3 | OP2 | OP1 | OP0 |
+-----------------------+-----------------------+

    OP0 - Ready To Send (RTS) A
    OP1 - Ready To Send (RTS) B
    OP2 - Data Terminal Ready (DTR) A
    OP3 - Data Terminal Ready (DTR) B
    OP4-OP6 - Unused
    OP7 - Turn on speaker
```

DISPLAY KEYBOARD STATUS REGISTER [8420 | 0FFB020]

read-only

```
+---------------+---------------+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---------------+---------------+
  |   |   |   |   |   |   |   |
  |   |   |   |   |   |   |   receive data register full
  |   |   |   |   |   |   |
  |   |   |   |   |   |   transmit data register empty
  |   |   |   |   |   |
  |   |   |   |   |   no DCD (always 0)
  |   |   |   |   |
  |   |   |   |   no CTS (always 0)
  |   |   |   |
  |   |   |   receive framing error
  |   |   |
  |   |   receive overrun (reset by reading data)
  |   |
  |   receive parity error
  |
  interrupt request (cleared by data read or write)
```

DISPLAY KEYBOARD COMMAND REGISTER [8420 | 0FFB020]

```
write-only
+---------------+---------------+
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
+---------------+---------------+
  |   |   |   |   |   |   |   |
  |   |   |   |   |   |   0   0 = clock/1
  |   |   |   |   |   |   0   1 = clock/16
  |   |   |   |   |   |   1   0 = clock/64
  |   |   |   |   |   |   1   1 = master reset
  |   |   |   |   |   |
  |   |   |   0   0   0 =7 bits, even parity, 2 stop bits
  |   |   |   0   0   1 =7 bits,  odd parity, 2 stop bits
  |   |   |   0   1   0 =7 bits, even parity, 1 stop bit
  |   |   |   0   1   1 =7 bits,  odd parity, 1 stop bit
  |   |   |   1   0   0 =8 bits, 2 stop bits
  |   |   |   1   0   1 =8 bits, 1 stop bits
  |   |   |   1   1   0 =8 bits, even parity, 1 stop bit
  |   |   |   1   1   1 =8 bits,  odd parity, 1 stop bit
  |   |   |
  |   0   0 = Set RTS, disable transmitter interrupt
  |   0   1 = Set RTS, enable        ''        ''
  |   1   0 = Reset RTS, disable     ''        ''
  |   1   1 = Set RTS, transmit break, disable transmitter
  |           interrupt
  1 = Enable receiver interrupts (receive data register
      full, overrun, loss of DCD).
```

DISPLAY KEYBOARD DATA REGISTER [8422 | 0FFB022]

```
+-------------------------------+
|         D   A   T   A         |
+-------------------------------+

     READ  = empties receive data register
     WRITE = loads  transmit data register
```

CHAPTER 8

DN400,DN420,DN600

ADDRESS SPACE

physical		virtual
100400	traps	0
400	prom	400 (one-to-one)
80000	phys mem	80000->FFFFF
100800	phys mem	100800
100000	debug data	E00000
40000	disp2_mem	FA0000->FBFFFF
20000	disp1_mem	FC0000->FDFFFF
10000	multibus	FE0000->FEFFFF
E000	Color	FF6000
E400	Color	FF6400
E800	Color	FF6800
B000	PEB Ctl	FF7000
B400	FPU Cmd	FF7400
C000	FPU CS	FF7800
C400	Cache W 0	FF7C00
C800	Cache W 1	FF8000
FC00	Memory Ctl	FF9000
F400	disp 2	FF9400
F000	disp 1	FF9800
BC00	ring 2	FF9C00
B800	ring 1	FFA000
8C00	floppy	FFA800
8800	timers	FFAC00
8400	sios	FFB000
8000	mmu	FFB400
4000	pft	FFB800->FFF7FF
	iomap	FFF800->FFF9FF

CONFIGURATION

```
        +------------+   +-------------+   +----------+
        |   MEMORY   |===|   MEMORY    |---|   CPU    |
        +------------+   | MANAGEMENT  |   +----+-----+
                         | UNIT (MMU)  |        |
                         +----++-------+        |
                              ||           +----+-----+
               PHOEBUS        ||===========| FLOPPY   |
        +---------------------+|           +----------+
        |+---------------------+            (Shugart)
        ||
        ||   +-------------+
        |+---|    BLOCK    |---------------------+
        |+---| MULTIPLEXOR |------++------------+|
        ||   +-------------+      ||            ||
        ||                     +------+      +------+
        ||                     | RING |      | DISK |
        ||                     +------+      +------+
        ||                                    (Priam)
        ||   +--------------+
        |+---|  NODE PERI-  |-----+
        |+---|PHERAL ADAPTER|----+|
        ||   +--------------+    || MULTIBUS
        ||                       ||
        ||   +--------------+    ||
        |+===|    TIMER     |    ||  +--------------+
        ||   +--------------+    |+--| LINE PRINTER |
        ||                       |+--|  CONTROLLER  |
        ||   +--------------+    ||  +------++------+
        |+===|  BOOT PROM   |    ||         ||
        ||   +--------------+    ||  +------++------+
        ||                       ||  | LINE PRINTER |
        ||   +--------------+    ||  +--------------+
        |+===|  CAL CLOCK   |    ||   (Printronix)
        ||   +--------------+    ||
        ||                       ||   (Tapemaster)
        ||   +--------------+    ||  +------------+
        |+===|  SIO LINES   |    |+--|  MAG TAPE  |
        ||   +--+--+--+--+--+    |+--| CONTROLLER |
        ||      |  |  |  |       ||  +-----++-----+
        ||     0|  1  2  3       ||        ||
        ||    +---+  Spinwriter  ||   +----++----+
        ||    |KBD|  Tablet      ||   | MAG TAPE |
        ||    +---+  HASP, 3270  ||   +----------+
        ||                       ||   (Cipher-F880)
        ||   +---------+ +----+  ||                +------------+
        |+---+ DISPLAY |=|TUBE|  |+----------------|    SMD     |
        |+---+ MEMORY  | +----+  |+----------------| CONTROLLER |
        ||   +---++----+         ||                +-----++-----+
        ||       ||              |+--- TWO FREE          ||
        ||   +---++----+         |+----   SLOTS    +-----++-----+
        |+===|   BLT   |         ||    |           |   AMPEX    |
        ||   +---------+         ||    VERSATEC    | 300 MB SMD |
        ||                       ||    ETHERNET    +------------+
```

DISPLAY BOARD JUMPERS

DN4xx

15 GREEN CRT:

W01-DOWN	W02-DOWN	W03-UP	W04-UP	W05-UP
W06-UP	W07-TOP-LEFT*	W08-UP	W09-UP	W10-UP
W11-UP	W12-UP	W13-LEFT	W14-RIGHT	W15-LEFT
W16-TOP-LEFT	W17-UP	W18-UP	W19-UP	W20-UP
W21-UP	W22-UP	W23-UP	W24-UP	W31-UP

15 BLACK & WHITE CRT:

WO1-DOWN	W02-DOWN	W03-UP	W04-UP	WO5-UP
W06-DOWN	W07-*	W08-UP	W09-UP	10-DOWN
W11-DOWN	W12-DOWN	W13-LEFT	W14-LEFT	W15-LEFT
W16-*	W17-DOWN	W18-UP	W19-DOWN	W20-DOWN
W21-UP	W22-OUT	W23-DOWN	W24-DOWN	W31-UP

19 BLACK & WHITE CRT

W01-DOWN	W02-UP	W03-UP	W04-DOWN	W05-DOWN
W06-DOWN	WO7- *	W08-DOWN	W09-DOWN	W10-DOWN
W11-DOWN	W12-DOWN	W13-RIGHT	W14-LEFT	W15-RIGHT
W16-*	W17-DOWN	W18-UP	W19-DOWN	W20-DOWN
W21-DOWN	W22-DOWN	W23-DOWN	W24-DOWN	W31-DOWN

* W07 AND W16 SHOULD BE SET BY SPECIFICATIONS ON CLOTH TAG, MOUNTED ON PCB.

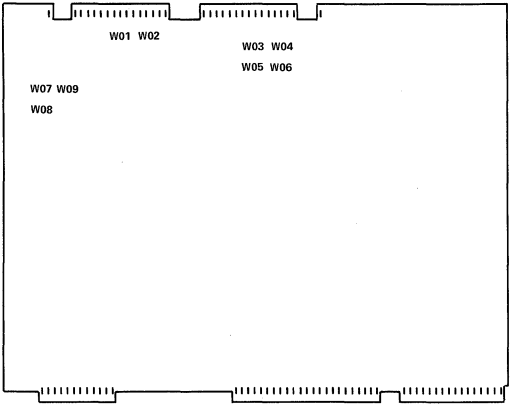
DISPLAY BOARD 2		DISPLAY BOARD 1	
W25-DOWN	W26-DOWN	W25-RIGHT	W26-UP
W27-DOWN	W28-DOWN	W27-RIGHT	W28-UP
W29-DOWN	W30-DOWN	W29-UP	W30-UP

ARRAY BOARD JUMPER PLACEMENT:

ARRAY BOARD JUMPER PLACEMENT:

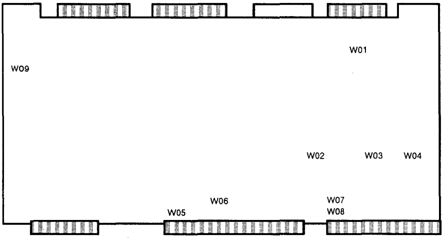
BOARD ORIENTATION: EJECTORS FACING UPWARDS

JUMP	W01	W02	W03	W04	W05	W06	W07	W08	W09
LOCT	~A08	~A09	~A11	~A12	~A11	~A12	C02-02	C02-08	C02-18
APR 1	LEFT	LEFT	UP	UP	UP	UP	UP	UP	OUT
APR 2	RIGHT	RIGHT	DOWN	DOWN	DOWN	DOWN	DOWN	UP	IN



CONTROL BOARD JUMPER PLACEMENT:

Control Board Jumper Placement:
Board orientation: ejectors facing upwards



Order of Boards:
Facing Front of Node

Control Array 1 Array 2

Jumper Orientation	W01	W02	W03	W04	W05	W06	W07	W08	W09
< Rev 12	Up	Down	Down	Down	Right	Up	Up	Down	None
>= Rev 12									
800 Line	Down	Down	Down	Down	Right	Up	Up	Down	Down
1024 Line	Down	Down	Down	Down	Right	Up	Up	Down	Up

BLT REGISTERS

DN4xx

Source and Destination Control Register:

```
 31              16                0
+----------------+----------------+
| Y addr. control| X addr. control|
+----------------+----------------+
```

	Increment	Decrement
Y address control:	0202	0606
X address control:	0020	0060

		OS Virtual Address
(RCS)	Read source control reg.	$FF988C
(WCS)	Write source control reg.	$FF9884
(RCD)	Read destination control reg.	$FF9888
(WCD)	Write destination control reg.	$FF9880

Source/Destination Start/End Registers:

```
 31   26 25       16        9         0
+------+----------+------+----------+
|-----V|     Y    |------|    X     |
+------+----------+------+----------+
          Y-coordinate      X-coordinate (Low order four bits
                              are write only)
```

V - Used in BLTs to or from main memory

		OS Virtual Address
(RSS)	Read source start reg.	$FF989C
(WSS)	Write source start reg.	$FF98AC
(RES)	Read source end reg.	$FF9894
(WES)	Write source end reg.	$FF98B4
(RSD)	Read destination start reg.	$FF9898
(WSD)	Write destination start reg.	$FF98A8
(RED)	Read destination end reg.	$FF9890
(WED)	Write destination end reg.	$FF98B0

When start or end registers are used to specify locations in main memory, the correspondence is as follows:

```
 31    26     19  16        9    4    0
+-----+--------+---+------+------+----+    Start or End
|-----| IOVPN  |BN |------|  BN  |----|    Register
+-----+--------+---+------+------+----+
          |       9:7         6:1
          |        |           |
          |        |     +-----+
          |        |     |
          v        v     v
      +--------+---+------+-+       I/O Virtual Address
      |                   |0|
      +--------+---+------+-+
       <---8--> <---10--->

    I/O virtual   Byte in page
    page number
```

DN6xx

```
  15                   8  7         4  3        0
+-----------------------+------------+-----------+
| X  X  X  X  X  X  X  X| HALT STATUS| X IN SY SA|
+-----------------------+------------+-----------+
```

IN = 1 Color display is executing an instruction queue starting at the address held in the instruction queue start location.

SY = 1 Color display is in system mode and will execute system functions.

SA Most significant address bit of all i/o addresses to be used by the color display for dma.

HALT STATUS

0 0 0 0	Normal completion
0 0 0 1	Illegal opcode
0 0 1 0	Unimplemented instruction
0 0 1 1	BLT to main memory out of image memory region
1 1 1 1	Not halted

DISPLAY CONTROL AND STATUS REGISTER (DCSR)

DN4xx

```
[ F000 | FF9800 ]

   15     14     13     12     11     10      9      8
+-----------------------------------------------------------+
| GO   |   x   |  x   |  x   |  x   |  x   |  x   |  NIL |
+-----------------------------------------------------------+

    7      6      5      4      3      2      1      0
+-----------------------------------------------------------+
|FROMMM| TO MM | I EOF| IDONE|NONCONF| DECR| FILL|  ON   |
+-----------------------------------------------------------+
```

R/W BIT NAME

R/W 15 GO - When set, this bit initiates a BLT operation. When cleared, this bit will abort any BLT in progress. When read, this bit will be set so long at a BLT operation is in progress.

W 8 NIL - When set, enables non-interlaced mode. In this mode, only the odd lines are displayed at 60 frames per second.

W 7 FROM MM - When set, enables BLTs from main memory as specified by the source box. This bit must be set in advance of the write operation that initiates a BLT. It should be cleared by a separate write operation after the BLT completes.

W 6 TO MM - When set, enables BLTs to main memory as specified by the destination box

W 5 IEOF - When set, enables a 60 hz interrupt request after the last line in the current field. This request must be acknowledged by reading from address DCSR + 2.

W 4 IDONE - When set, enables interrupt request when BLT is done.

W 3 NON CONF- When set, enables non-conforming BLT mode.

W	2	DECR	- Must be set when doing a BLT that decrements the x coordinate.
W	1	FILL	- When set, BLT fill mode is enabled.
W	0	ON	- When set, display is on; when reset, display is blanked.

DN6xx

CONTROL REGISTER [E000 | FF6000] (System)
[E400 | FF6400] (User)

BIT	NAME	FUNCTION
15	Reserved	Set to zero
14	Reserved	Set to zero
13	Unused	
12	Unused	
11	Unused	
10	Unused	
9 - 4	WCSADH<5:0>+	During reads and writes to the control store through the control store window these bits are used as the upper part of the control store address.
3	VIDENB+	This bit is used to enable or disable the video to the monitor.
2	RESET-	When this bit is asserted the micro machine is reset,the control store may be accessed through its window and no micro code is executed. The display will continue to be refreshed, no changes will be made to the display memory, and no software visible state will change.
1	CLKRUN+	When this bit is set the micro machine clocks will run.
0	CLKSTEP+	When the CLKRUN+ bit is not set the micro-machine clocks can be made to go through one clock cycle by toggling this bit low to high.

STATUS REGISTER

BIT	NAME	FUNCTION
15	DONE-	This bit is asserted by micro code to indicate that the micro machine is ready to handle reads or writes to its command pages
14	VBLANK-	When this bit is asserted the display is currently in the vertical blanking period.
13	INST_DONE-	This bit is asserted by micro code to indicate that is done executing an instruction queue.
12	CLKON+	This bit indicates that the micro machine clocks are running.
11 - 0	NXTAD<11:0>-	These bits are the state of the next address bus at the end of the last micro cycle. This is needed in order to know the micro address when micro code is being clock stepped.

FAULT FRAME

(type "fault_$bus_info_t" in fault.ins.pas)

```
                 15             0
                +----------------+                -----
   SP--> +00    |-----------RNFFF| }                  ^
                +----------------+ }                  |
         +02    |     ACCESS     | }  .access_address
                +                + } fault_$bus_info_t
                |     ADDRESS    | }                  |
                +----------------+ }        Bus/addr excep.
         +06    |INSTRUCTION REG.| }  .inst_reg       |
                +----------------+      -----         |
         +08    |   STATUS REG.  |        ^           |
                +----------------+        |           |
         +0A    |     PROGRAM    |    Group 1 & 2     |
                +                +     Exceptions     |
                |     COUNTER    |        |           |
         +0E    +----------------+     ___V___________V_
```

R   - Read operation (.write_op)
N   - Not instruction reference (.not_inst)
FFF - Instruction function code (.function_code):

001	User data
010	User program
101	Supervisor data
110	Supervisor program
111	Interrupt acknowledge

FAULT TYPES

GROUP	EXCEPTION	PROCESSING
0	Reset Bus Error Address Error	Current instruction is aborted.
1	Trace Interrupt Illegal Ins. Privilege Ins.	Exception occurs before next instruction.
2	TRAP, TRAPV CHK Zero Divide	Processed by normal instruction execution.

Group 0 exceptions are highest priority.

FAULT VECTORS

Exception vector at [100400 | 0]

Vector	Address	Assignment
00	000	Reset: Initial SSP
	004	Reset: Initial PC
02	008	Bus Error
03	00C	Address Error
04	010	Illegal Instruction
05	014	Zero Divide
06	018	CHK Instruction
07	01C	TRAPV Instruction
08	020	Privilege Violation
09	024	Trace
0A	028	Unimplemented instruction
0B	02C	Unimplemented instruction
0C-0D	030	(Unassigned, reserved)
0F-17	03C	(Unassigned, reserved)
18	060	Spurious Interrupt
20-2F	080	TRAP Instruction Vectors
30-3F	0C0	(Unassigned, reserved)
40-8F	100	User Int Vectors - unused
90-9F	240	Ring/disk board
A0-AF	280	User Int Vectors - unused
B0	2C0	INT0/ -
B1	2C4	INT1/ -
B2	2C8	INT2/ -
B3	2CC	INT3/ - Tape Controller
B4	2D0	INT4/ - Storage Module
B5	2D4	INT5/ -
B6	2D8	INT6/ - Line Printer
B7	2DC	INT7/ - Parallel Input
B8-C3	2E0	User Int Vectors -- unused
C4-CD	310	Unused
CE-CF	338	Color
D0-EF	340	SIO lines (even vectors only odd vectors unused)
F0	3C0	P - ECCC (Automatic vectors)
F1	3C4	O -
F2	3C8	N - Display #2
F3	3CC	M - Floppy
F4	3D0	L - Display #1 (BLT)
F5	3D4	K -
F6	3D8	J -
F7	3DC	I -
F8-FA	3E0	Unused
FB	3EC	Timers 1,2,3
FC-FD	3F0	Unused
FE	3F8	CPU B-to-A
FF	3FC	ECCU

FLOATING-POINT FORMAT

Refer to the FLOATING-POINT FORMAT section of CHAPTER 7, DN300,DN320, for this information.

FLOPPY CONTROLLER

```
Floppy Status     [ 8C00 | FFA800 ]
Floppy I/O        [ 8C00 | FFA800 ]
Floppy DMA        [ 8C80 | FFA880 ]
```

Floppy Status Registers

```
          Main Status Register
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
```

D7 - {RQM} Data register ready to send/receive data from Processor.

D6 - {DIO} I/O Direction, 1 = Data Reg to Processor, 0 = Processor to Data Reg.

D5 - {EXM} Execution mode for non-DMA transfers

D4 - {CB} BUSY, a Read or Write Command is in process

D3 - {D3B} FDD 3 is in Seek Mode

D2 - {D2B} FDD 2 is in Seek Mode

D1 - {D1B} FDD 1 is in Seek Mode

D0 - {D0B} FDD 0 is in Seek Mode

```
STATUS REGISTER 0
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
```

D7 -	D6	
0	0	Normal Termination of Command
0	1	Abnormal Term {command started, not completed}
1	0	Invalid Command{ command issued never started }
1	1	Abnormal Termination{ FDD went not ready during command execution }

D5 - Set when Seek Completed

D4 - Set if FDD issues Fault Signal or Track 0 Signal fails to occur after 77 Step Pulses.

D3 - Set if FDD Not Ready

D2 - Head Address

D1 - Unit Select 1 Status

D0 - Unit Select 0 Status

STATUS REGISTER 1

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
D7 - End of Cylinder
D6 - 0
D5 - Data Error
D4 - Overrun
D3 - 0
D2 - No Data
D1 - Not Writable
D0 - Missing Address Mark
```

STATUS REGISTER 2

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
D7 - 0
D6 - Control Mark
D5 - Data Error in Data Field
D4 - Wrong Cylinder
D3 - Scan Equal Hit
D2 - Scan NOT Satisfied
D1 - Bad Cylinder
D0 - Missing Address Mark in Address Field
```

STATUS REGISTER 3

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
D7 - Fault from FDD
D6 - Write Protected
D5 - Ready from FDD
D4 - Track 0
D3 - Two Sided
D2 - Head Address
D1 - Unit Select 1 Status
D0 - Unit Select 0 Status
```

I/O MAP

```
 15              0
+----------------+
|----PPPPPPPPPPPP|
+----------------+
```

P..P - Physical page number (PPN)
(type "ppn_t" in base.ins.pas)

I/O MAP at [-- | FFF800].
I/O MAP for DN460/660 at [9000 | FFFF800].

I/O map supplies physical addresses for non-CPU memory references (ring/disk, PNA, display, floppy).

See also I/O MAP in Peripheral I/O section, Chapter 6.

MEMORY CONTROL/STATUS REGISTERS (MCSR)

```
Board 1:  [ FC02 | FF9002 ]
Board 2:  [ FD02 | FF9102 ]
Board 3:  [ FE02 | FF9202 ]
Board 4:  [ FF02 | FF9302 ]
```

MCSR Control (Write-Only)

```
 15              0
+----------------+
|-------------DCU|
+----------------+
```

D - Lock check bits on write (diag mode only)
C - Enable ECCC interrupts (through 3C0)
U - Enable ECCU interrupts (through 3FC)

Any write to this register acknowledges the interrupt.

MCSR Status Register (Read-Only)

```
 31     24 23                     0
+--------+------------------------+
|SSSSSSNU|PPPPPPPPPPPPPPPPPPPPPPPW|
+--------+------------------------+
```

S..S - Syndrome (valid only for ECCC error):

D0	data bit	00	70	data bit	08	2C	data bit	15
C8	"	01	68	"	09	F8	check bit	0
C4	"	02	58	"	10	F4	"	1
B0	"	03	54	"	11	EC	"	2
A8	"	04	4C	"	12	DC	"	3
A4	"	05	38	"	13	BC	"	4
94	"	06	34	"	14	7C	"	5
8C	"	07						

N - No error (0 => error detected, status valid)

U - Uncorrectable error (ECCU)

P..P - High order 23 bits of physical address
(low order bit is always 0)

W - Error during read-modify-write operation

MEMORY BOARD JUMPERS

Board #	JP3	JP4	Abs Address	Mapped Address
1	in	in	fc02	ff9002
2	out	in	fd02	ff9102
3	in	out	fe02	ff9202
4	out	out	ff02	ff9302

Each board type has an address range that it can span specified by three jumpers, JP13, JP14, and JP15.

Memory Board Address Ranges

256Kb Board #	JP13	JP14	JP15	Address Range
1	in	out	out	100000-13ffff
2	out	out	out	140000-17ffff
3	in	out	in	180000-1bffff
4	out	out	in	1c0000-1fffff
3*	in	in	in	80000-bffff
4*	out	in	in	c0000-fffff

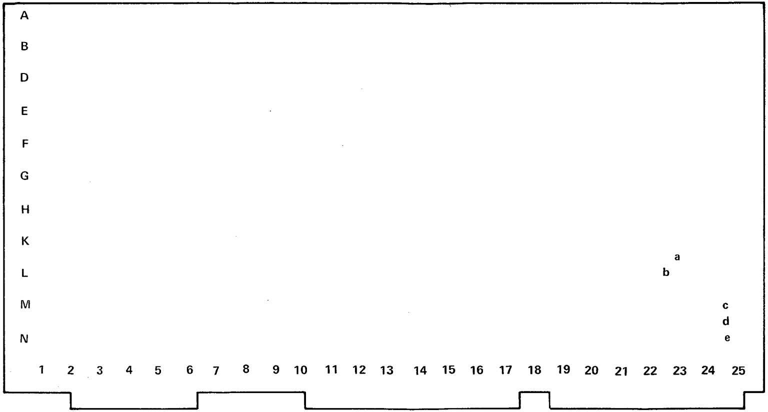
* Use these values if there are 2 512Kb boards or 1 1mb board in the system

512Kb Board #	JP13	JP14	JP15	Address Range
1	in	out	out	100000-17ffff
2	in	out	in	180000-1fffff
3	out	in	out	200000-27ffff
4	in	in	in	080000-0fffff

1Mb Board #	JP13	JP14	JP15	Address Range
1	in	out	out	100000-1fffff
2	out	in	out	200000-2fffff
3	out	out	out	300000-3fffff
4	*** not allowed ***			

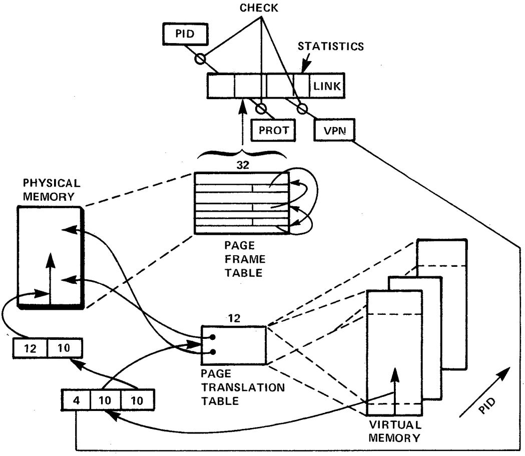
The memory test program checks that the tables above are satisfied.

MEMORY BOARD



LABEL			COORDINATE LOCATION (APPROXIMATE)
a	=	JP3	L-23
b	=	JP4	L-23
c	=	JP13	M-25
d	=	JP14	M-25
e	=	JP15	N-25

MEMORY MANAGEMENT UNIT (MMU)



PID/PRIV/POWER	[8000	FFB400]
CPU A Control	[8002	FFB402]
MMU Status, High	[8004	FFB404]
MMU Status, Low	[8006	FFB406]
Clear MMU Status	[8008	FFB408]
Bus Status	[800A	FFB40A]
Enable CPU B	[800E	FFB40E]

PID/PRIV Register

```
Address: [ 8000 | FFB400]
 15      8 7      0
+-+-------+--------+
|E| ASID  |----DDPM|
+-+-------+--------+
```

```
E  - Enable power-off switch
DD - DOMAIN:
     00  User domain 0 (least protected)
     01  User domain 1
     10  Supervisor domain 0
     11  Supervisor domain 1 (most protected)
P  - PTT access enable
M  - Enable MMU (virtual memory operations)
```

CPU A Control Register

```
Address: [ 8002 | FFB402 ]
 15     8 7      0
+--------+--------+
|--------|------IA|
+--------+--------+
```

```
IA - Interrupt and/or abort:
     00  CPU A will resume operation
     01  CPU A will abort the attempted bus cycle
     10  CPU A will receive level 6 interrupt
```

MMU Status Register

```
Address: [8004 | FFB404 ]
 31    24 23                        0
+--------+-------------------------+
|MAFFFRVB| FAULTING VIRTUAL ADDRESS |
+--------+-------------------------+
```

```
M   - PFT miss (ASID and/or VPN not found)
A   - Access fault (protection violation)
FFF - CPU function code:
      001  User data reference
      010  User procedure reference
      101  Supervisor data reference
      110  Supervisor procedure reference
      111  Interrupt acknowledge
R   - Read operation
V   - MMU status register is valid
B   - Miss occurred while CPU B running
```

Clear MMU Status

Address: [8008 | FFB408]

When written, clears MMU status conditions.

Bus Status Register

```
Address: [ 800A | FFB40A ]
 16      8 7      0
+--------+--------+
|--------|IMPGTBWS|
+--------+--------+
```

I = 0 => unacknowledged interrupt pending
M - MMU enabled
P - PTT enabled
G - Big PFT (2K entries if not set, 4K if set)
T - Bus time-out (cleared by a read)
B - CPU B active
W - State of power-off switch (1 => ON position)
S - State of service switch (1 => Normal position)

Enable CPU B Register

Address: [800E | FFB40E]

Any write will explicitly enable CPU B. Return to CPU A, using Reset only.

MONITOR TIMING (DN6xx)

Axis	Item		Duration 800 Lines	Duration 1024 Lines
HORIZONTAL	1H	HORIZONTAL FREQUENCY	31.2 uSec @32.051 KHz	31.2 uSec @32.052 KHz
	H-FP	HORIZONTAL FRONT PORCH	0.7 uSec	0.7 uSec
	H-SYNC	HORIZONTAL SYNC	3.0 uSec	3.0 uSec
	H-BP	HORIZONTAL BACK PORCH	3.5 uSec	3.5 uSec
	H-BL	HORIZONTAL BLANKING	7.2 uSec	7.2 uSec
	H-DISP	HORIZONTAL DISPLAY AREA	24.0 uSec	24.0 uSec
VERTICAL	1V	VERTICAL FREQUENCY	13.2 mSec @ 75.7 Hz	16.7 mSec @ 60.0 Hz
	V-FP	VERTICAL FRONT PORCH	0.0	0.5 uSec
	V-SYNC	VERTICAL SYNC	93.6 uSec	93.6 uSec
	V-BP	VERTICAL BACK PORCH	600.0 uSec	616.0 uSec
	V-BL	VERTICAL BLANKING	693.6 uSec	710.0 uSec
	V-DISP	VERTICAL DISPLAY AREA	12.516 mSec	15.956 mSec

```
      |<-x-BL-|<-------- x-DISP ---------->|
 -----+       +----------------------------+
 /////|       |////////////////////////////|
 /////|       |////////////////////////////|
 -----+-+  +--+----------------------------+-+  +--
      | |  |  |                              |  |
      | +--+  |                              +--+
      | |  |  |                              |
x-FP->|-|--|--|<-x-BP                        |
        | ^                                  |
        | +--x-SYNC                          |
        |<--------------1x------------------>|

  x = V or H
```

PAGE FRAME TABLE ENTRY (PFTE)

Refer to the PAGE FRAME TABLE ENTRY (PFTE) section of CHAPTER 7, DN300,DN320, for this information.

PAGE TRANSLATION TABLE ENTRY (PTTE)

Refer to the PAGE TRANSLATION TABLE ENTRY (PTTE) section of CHAPTER 7, DN300,DN320, for this information.

PEB

[B000 | FF7000]

CACHE Pages: [C400 | FF7C00]
[C800 | FF8000]

Definitions

peb_ctrl	FF7000	PEB control register
peb_status	FF7000	PEB status register (sys space)
fpu_status	FF77FC	FPU status register (user space)
fpu_cs_sa	FF7800	FPU control store sa
fpu_cs_ea	FF7C00	FPU control store ea
wcs_1k_size	1024*8	FPU control store bytes
wcs_4k_size	4096*8	FPU control store in bytes
fpu_defalth	00B2000E	FPU default microword, high 32
fpu_defaltl	500079E3	FPU default microword, low 32
wcserr_vec	F1<<2	WCS error interrupt vector
xcp_int_vec	2e0	FPU Exception Interrupt Vector
BUS_FA_SA	1*4+PEB_BASE	SP ADD
LFT_FA_SA_FT	2*4+PEB_BASE	
FA_BUS_SA	3*4+PEB_BASE	
*		
BUS_FA_SS	4*4+PEB_BASE	SP SUBTRACT
LFT_FA_SS_FT	5*4+PEB_BASE	
FA_BUS_SS	6*4+PEB_BASE	
*		
BUS_FA_RS	7*4+PEB_BASE	SP REVERSE SUBTRACT
LFT_FT_SS_FA	8*4+PEB_BASE	
FA_BUS_RS	9*4+PEB_BASE	
*		
BUS_FA_SM	10*4+PEB_BASE	SP MULTIPLY
LFT_FA_SM_FT	11*4+PEB_BASE	
FA_BUS_SM	12*4+PEB_BASE	
*		
BUS_FA_SD	13*4+PEB_BASE	SP DIVIDE
LFT_FA_SD_FT	14*4+PEB_BASE	
FA_BUS_SD	15*4+PEB_BASE	
*		
BUS_FA_RDS	16*4+PEB_BASE	SP REVERSE DIVIDE
LFT_FT_SD_FA	17*4+PEB_BASE	
FA_BUS_RDS	18*4+PEB_BASE	
*		
BUS_FAH_DA	19*4+PEB_BASE	DP ADD
BUS_FAL_DA	20*4+PEB_BASE	
BUS_FTH_DA	21*4+PEB_BASE	
LFTL_FA_DA_FT	22*4+PEB_BASE	
FAH_BUS_DA	23*4+PEB_BASE	
FAL_BUS_DA	24*4+PEB_BASE	
*		
BUS_FAH_DS	25*4+PEB_BASE	DP SUBTRACT
BUS_FAL_DS	26*4+PEB_BASE	
BUS_FTH_DS	27*4+PEB_BASE	
LFTL_FA_DS_FT	28*4+PEB_BASE	
FAH_BUS_DS	29*4+PEB_BASE	

```
FAL_BUS_DS       30*4+PEB_BASE
*
BUS_FAH_RD       31*4+PEB_BASE    DP REVERSE SUBTRACT
BUS_FAL_RD       32*4+PEB_BASE
BUS_FTH_RD       33*4+PEB_BASE
LFTL_FT_DS_FA    34*4+PEB_BASE
FAH_BUS_RD       35*4+PEB_BASE
FAL_BUS_RD       36*4+PEB_BASE
*
BUS_FAH_DM       37*4+PEB_BASE    DP MULTIPLY
BUS_FAL_DM       38*4+PEB_BASE
BUS_FTH_DM       39*4+PEB_BASE
LFTL_FA_DM_FT    40*4+PEB_BASE
FAH_BUS_DM       41*4+PEB_BASE
FAL_BUS_DM       42*4+PEB_BASE
*
BUS_FAH_DD       43*4+PEB_BASE    DP DIVIDE
BUS_FAL_DD       44*4+PEB_BASE
BUS_FTH_DD       45*4+PEB_BASE
LFTL_FA_DD_FT    46*4+PEB_BASE
FAH_BUS_DD       47*4+PEB_BASE
FAL_BUS_DD       48*4+PEB_BASE
*
BUS_FAH_RDD      49*4+PEB_BASE    DP REVERSE DIVIDE
BUS_FAL_RDD      50*4+PEB_BASE
BUS_FTH_RDD      51*4+PEB_BASE
LFTL_FT_DD_FA    52*4+PEB_BASE
FAH_BUS_RDD      53*4+PEB_BASE
FAL_BUS_RDD      54*4+PEB_BASE
*
R_W_UR0          55*4+PEB_BASE    READ/WRITE INTERNAL REGISTER
R_W_UR1          56*4+PEB_BASE
R_W_UR2          57*4+PEB_BASE
R_W_UR3          58*4+PEB_BASE
R_W_UR4          59*4+PEB_BASE
R_W_UR5          60*4+PEB_BASE
R_W_UR6          61*4+PEB_BASE
R_W_XCP          61*4+PEB_BASE    READ/WRITE EXCEPTION REGISTER
R_W_UR7          62*4+PEB_BASE
R_W_UR8          63*4+PEB_BASE
R_W_UR9          64*4+PEB_BASE
R_W_URA          65*4+PEB_BASE
R_W_IAC          65*4+PEB_BASE    READ/WRITE INTEGER ACCUMULATOR
R_W_URB          66*4+PEB_BASE
R_W_URC          67*4+PEB_BASE
R_W_URD          68*4+PEB_BASE
R_W_URE          69*4+PEB_BASE
R_W_URF          70*4+PEB_BASE
*
HIGH_BUS         71*4+PEB_BASE    READ UPPER 24 BITS OF DP MANTISSA
*
* 5 spares here (72 thru 76)
*
lft_smin         77*4+PEB_BASE    Min(FA,FT) => FA           (e.f.)
fa_bus_smin      78*4+PEB_BASE    Condition codes => IAC high word
*
```

```
bus_fth_dmin     79*4+PEB_BASE    Min(DFA,DFT) => DFA         (e.f.)
lftl_dmin        80*4+PEB_BASE    Condition codes => IAC high word
fah_bus_dmin     81*4+PEB_BASE
fal_bus_dmin     82*4+PEB_BASE
*
lft_smax         83*4+PEB_BASE    Max(FA,FT) => FA            (e.f.)
fa_bus_smax      84*4+PEB_BASE    Condition codes => IAC high word
*
bus_fth_dmax     85*4+PEB_BASE    Max(DFA,DFT) => DFA         (e.f.)
lftl_dmax        86*4+PEB_BASE    Condition codes => IAC high word
fah_bus_dmax     87*4+PEB_BASE
fal_bus_dmax     88*4+PEB_BASE
*
lfa_sax          89*4+peb_base    FA + FX => FA (single prec)(e.f.)
fa_bus_sax       90*4+peb_base
*
bus_fah_dax      91*4+peb_base    FA + FX => FA (double prec)(e.f.)
lfal_dax         92*4+peb_base
fah_bus_dax      93*4+peb_base
fal_bus_dax      94*4+peb_base
*
lfa_smx          95*4+peb_base    FA * FX => FA (single prec)(e.f.)
fa_bus_smx       96*4+peb_base
*
bus_fah_dmx      97*4+peb_base    FA * FX => FA (double prec)(e.f.)
lfal_dmx         98*4+peb_base
fah_bus_dmx      99*4+peb_base
fal_bus_dmx     100*4+peb_base
*
fx_to_fa        101*4+peb_base    FX => FA                    (e.f.)
fa_to_fx        102*4+peb_base    FA => FX
*
REV_BUS         103*4+PEB_BASE    READ MICRO-CODE REVISION LEVEL
*
BUS_FX          104*4+PEB_BASE    WRITE SP X REGISTER FOR SP POLYNOMIAL
BUS_FA_SP       105*4+PEB_BASE    SP POLYNOMIAL
LFT_SP          106*4+PEB_BASE    (FA * FX) + FT => FA
FA_BUS_SP       107*4+PEB_BASE
*
BUS_FXH         108*4+PEB_BASE    WRITE DP X REGISTER FOR DP POLYNOMIAL
BUS_FXL         109*4+PEB_BASE
BUS_FAH_DP      110*4+PEB_BASE    DP POLYNOMIAL
BUS_FAL_DP      111*4+PEB_BASE    (DFA * DFX) + DFT => DFA
BUS_FTH_DP      112*4+PEB_BASE
LFTL_DP         113*4+PEB_BASE
FAH_BUS_DP      114*4+PEB_BASE
FAL_BUS_DP      115*4+PEB_BASE
*
FXH_BUS         116*4+PEB_BASE    READ DP X REGISTER
FXL_BUS         117*4+PEB_BASE
*
XCHNG           118*4+PEB_BASE   ; This swaps the  FA with the FX
*
FTH_BUS         119*4+PEB_BASE    READ DP TEMP REGISTER HIGH PART ONLY
*
F_NEG           120*4+PEB_BASE    NEGATE SP/DP ACCUMULATOR
```

```
*
F_ABS           121*4+PEB_BASE    ABSOLUTE VALUE SP/DP OF ACCUMULATOR
*
SP_DP           122*4+PEB_BASE    CONVERT SP IN ACCUM TO DP NUMBER
DP_SP           123*4+PEB_BASE    CONVERT DP IN ACCUM TO SP NUMBER
*
L_SP            124*4+PEB_BASE    FLOAT  INTEGER ACCUMULATOR INTO SP
L_DP            125*4+PEB_BASE    FLOAT  INTEGER ACCUMULATOR INTO DP
*
sp_l            126*4+peb_base    FIX SP TO INTEGER ACCUMULATOR (e.f.)
dp_l            127*4+peb_base    FIX DP TO INTEGER ACCUMULATOR (e.f.)
*
LIT_INTMUL      128*4+PEB_BASE    LOAD INT TEMP REG. (32 BIT) MULTIPLY
LIT_INTDIV      129*4+PEB_BASE    LOAD INT TEMP REG. (32 BIT) DIVIDE
LIT_RINTDIV     130*4+PEB_BASE    LOAD INT TEMP REG. (32 BIT) REV DIVIDE
*
sp_nint         131*4+peb_base    NEAREST INTEGER OF SP => SP (e.f.)
dp_nint         132*4+peb_base    NEAREST INTEGER OF DP => DP (e.f.)
*
W_IAC_SP        133*4+PEB_BASE    LOAD INT ACCUMULATOR, THEN FLOAT TO SP
W_IAC_DP        134*4+PEB_BASE    LOAD INT ACCUMULATOR, THEN FLOAT TO DP
*
BUS_FAH_DMA     135*4+PEB_BASE    DP MULTIPLY AND ACCUMULATE
BUS_FAL_DMA     136*4+PEB_BASE    (DFA * DFT) + DFX => DFA,DFX
BUS_FTH_DMA     137*4+PEB_BASE
LFTL_DMA        138*4+PEB_BASE
FAH_BUS_DMA     139*4+PEB_BASE
FAL_BUS_DMA     140*4+PEB_BASE
*
bus_fa_sma      141*4+peb_base    SP MULTIPLY AND ACCUMULATE  (e.f.)
lft_sma         142*4+peb_base    (FA * FT) + FX => FA,FX     (e.f.)
fa_bus_sma      143*4+peb_base                                (e.f.)
*
*
*           THE FOLLOWING COMMANDS ARE ONLY AVAILABLE ON THE 4K
*           CONTROL STORE 'FPU' USED ON THE DN3XX-DN5XX.
*
sp_trunc        144*4+peb_base    TRUNCATE SP => SP (e.f.)
dp_trunc        145*4+peb_base    TRUNCATE DP => DP (e.f.)
*
sp_log          146*4+peb_base    LOG(SP)  => SP (e.f.)
dp_log          147*4+peb_base    LOG(DP)  => DP (e.f.)
sp_exp          148*4+peb_base    EXP(SP)  => SP (e.f.)
dp_exp          149*4+peb_base    EXP(DP)  => DP (e.f.)
sp_sqrt         150*4+peb_base    SQRT(SP) => SP (e.f.)
dp_sqrt         151*4+peb_base    SQRT(DP) => DP (e.f.)
*
lft_pwr         152*4+peb_base    SP FA**FT => SP  (e.f.)
bus_fth_pwr     153*4+peb_base    DP FA**FT => DP  (e.f.)
lftl_pwr        154*4+peb_base                     (e.f.)
iac_sp_pwr      155*4+peb_base    SP FA**I  => SP  (e.f.)
iac_dp_pwr      156*4+peb_base    DP FA**I  => DP  (e.f.)
*
sp_sin          157*4+peb_base    SIN(SP) => SP  (e.f.)
dp_sin          158*4+peb_base    SIN(DP) => DP  (e.f.)
sp_cos          159*4+peb_base    COS(SP) => SP  (e.f.)
```

```
dp_cos          160*4+peb_base    COS(DP) => DP   (e.f.)
sp_tan          161*4+peb_base    TAN(SP) => SP   (e.f.)
dp_tan          162*4+peb_base    TAN(DP) => DP   (e.f.)
sp_atan         163*4+peb_base    ATAN(SP) => SP   (e.f.)
*
*       DP-ATAN currently has a bug and is not used by the library
*
dp_atan         164*4+peb_base    DATAN(DP) => DP (e.f.)
lft_atan2       165*4+peb_base    ATAN2(FA,FT) => FA   (e.f.)
*
*       DP-ATAN2 currently has a bug and is not used by the library
*
bus_fth_datan2 166*4+peb_base    DATAN2(DFA,DFT) => DFA (e.f.)
lftl_datan2     167*4+peb_base         . . . .
*
* 87 spares here (168 thru 254)
*
FPU_STATUS_BUS 255*4+PEB_BASE    FPU STATUS REGISTER
*
cache_sa          FF7C00          cache window start
cache_ea          FF8400          cache window end
memory_sa         120000          memory start
memory_ea         120800          memory end
tag_inval         800             invalid tag word
tag_valid_bit     11
```

<u>PEB Control Register Bits</u>

```
peb_fpu_en           0001       FPU enable
peb_fpu_step         0002       FPU step
peb_fpu_reset        0004       FPU reset
peb_fpu_xie          0008       FPU interrupt enable
peb_fpu_csad         001F       upper control store address bits
peb_cache_a          0200       cache A (0050)
peb_cache_b          0400       cache B (0051)
peb_test_t           0800       test tag RAM (0052)
peb_test_d           1000       test data RAM (0053)
peb_cache_tpe        2000       cache tag parity enable (0054)
peb_cache_dpe        4000       cache data parity enable (0056)
```

<u>Useful Combinations</u>

```
peb_cache_tta     peb_cache_a+peb_test_t  (0060)    test tag A
peb_cache_ttb     peb_cache_b+peb_test_t  (0061)    test tag B
peb_cache_tda     peb_cache_a+peb_test_d  (0062)    test data A
peb_cache_tdb     peb_cache_b+peb_test_d  (0063)    test data B
enab_tag_a        peb_cache_tta+peb_cache_tpe+peb_cache_dpe (0065)
enab_tag_b        peb_cache_ttb+peb_cache_tpe+peb_cache_dpe (0066)
enab_data_a       peb_cache_tda+peb_cache_dpe+peb_cache_tpe (0067)
enab_data_b       peb_cache_tdb+peb_cache_dpe+peb_cache_tpe (0068)
enab_cache_a      peb_cache_a+peb_cache_dpe+peb_cache_tpe   (0069)
enab_cache_b      peb_cache_b+peb_cache_dpe+peb_cache_tpe   (0070)
enab_cache        enab_cache_a+peb_cache_b (0071)
```

PEB Status Register Bits

peb_fpu_cspe	0001	control store parity error
peb_cache_pe	0002	cache parity error
peb_fpu_xip	0004	FPU exception interrupt pending
peb_fpu_upc	07FF	FPU program counter bits
peb_fpu_busy	8000	FPU busy

FPP Commands

(from fpp.ins.pas)

FPP_$TSTX	Test S.P. or D.P. FAC for -,0,+; set cc’s & D0.L then exit
FPP_$SSTX	Same as FPP_$SSTA but exits when done (sets cc’s)
FPP_$EXIT	Return to caller beginning with instruction in next word
FPP_$SLV	FAC := (SP)+ (Single precision Load Value)
FPP_$SLA	FAC := ((SP)+) (Single precision Load using Address)
FPP_$SLC	FAC := Next four bytes in instruction stream (Constant)
FPP_$SSTA	((SP)+) := FAC (Single precision STore using Addr)
FPP_$SAV	FAC := FAC + (SP)+ (Single precision Add Value)
FPP_$SAA	FAC := FAC + ((SP)+)
FPP_$SAC	FAC := FAC + Next four bytes (Constant)
FPP_$SSV	FAC := FAC - (SP)+ (Single precision Subtract Value)
FPP_$SSA	FAC := FAC - ((SP)+)
FPP_$SSC	FAC := FAC - Next four bytes (Constant)
FPP_$SISV	FAC := (SP)+ - FAC (Single precision Inverse Subtract Value)
FPP_$SISA	FAC := ((SP)+) - FAC
FPP_$SISC	FAC := Next four bytes (Constant) - FAC
FPP_$SMV	FAC := FAC * (SP)+ (Single precision Multiply Value)
FPP_$SMA	FAC := FAC * ((SP)+)
FPP_$SMC	FAC := FAC * Next four bytes (Constant)
FPP_$SDV	FAC := FAC / (SP)+ (Single precision Divide Value)
FPP_$SDA	FAC := FAC / ((SP)+)
FPP_$SDC	FAC := FAC / Next four bytes (Constant)
FPP_$SIDV	FAC := (SP)+ / FAC (Single precision Inverse Divide Value)
FPP_$SIDA	FAC := ((SP)+) / FAC
FPP_$SIDC	FAC := Next four bytes (Constant) / FAC
FPP_$SCVX	Compare FAC : (SP)+; set cc’s & D0.L then exit
FPP_$SCAX	Compare FAC : ((SP)+); set cc’s & D0.L then exit
FPP_$SCCX	Compare FAC : Next 4 bytes; set cc’s & D0.L then exit
FPP_$SLWV	FAC := Float[(SP)+] (Float 16 bit integer)
FPP_$SLWA	FAC := Float[((SP)+)] (Float 16 bit integer)
FPP_$SLLV	FAC := Float[(SP)+] (Float 32 bit integer)
FPP_$SLLA	FAC := Float[((SP)+)] (Float 32 bit integer)
FPP_$SSTWX	D0.W := Fix[FAC]; then Exit with cc’s set

FPP_$SSTLX	D0.L := Fix[FAC]; then Exit with cc's set
FPP_$NEG	FAC := -FAC (Single or Double Precision)
FPP_$ABS	FAC := Abs[FAC] (Absolute value Single or Double Precision)
FPP_$DLA	D.P. FAC := ((SP)+) (Double precision Load using Address)
FPP_$DLC	D.P. FAC := Next 8 bytes (Constant)
FPP_$DSTA	D.P. ((SP)+) := FAC (Double precision Store using Address)
FPP_$DSTX	D.P. Same as FPP_$DSTA but exits when done (sets cc's)
FPP_$DAA	D.P. FAC := FAC + ((SP)+)
FPP_$DAC	D.P. FAC := FAC + Next 8 bytes (Constant)
FPP_$DSA	D.P. FAC := FAC - ((SP)+)
FPP_$DSC	D.P. FAC := FAC - Next 8 bytes (Constant)
FPP_$DISA	D.P. FAC := ((SP)+) - FAC (Inverse Subtract)
FPP_$DISC	D.P. FAC := Next 8 bytes (Constant) - FAC
FPP_$DMA	D.P. FAC := FAC * ((SP)+)
FPP_$DMC	D.P. FAC := FAC * Next 8 bytes (Constant)
FPP_$DDA	D.P. FAC := FAC / ((SP)+)
FPP_$DDC	D.P. FAC := FAC / Next 8 bytes (Constant)
FPP_$DIDA	D.P. FAC := ((SP)+) / FAC (Inverse Divide)
FPP_$DIDC	D.P. FAC := Next 8 bytes (Constant) / FAC
FPP_$DCAX	D.P. Compare FAC : ((SP)+); set cc's & D0.L then exit
FPP_$DCCX	D.P. Compare FAC : Next 8 bytes; set cc's & D0.L then exit
FPP_$DLWV	D.P. FAC := Float[(SP)+] (Float 16 bit integer)
FPP_$DLWA	D.P. FAC := Float[((SP)+)] (Float 16 bit integer)
FPP_$DLLV	D.P. FAC := Float[(SP)+] (Float 32 bit integer)
FPP_$DLLA	D.P. FAC := Float[((SP)+)] (Float 32 bit integer)
FPP_$DSTWX	D.P. D0.W := Fix[FAC]; then Exit
FPP_$DSTLX	D.P. D0.L := Fix[FAC]; then Exit
FPP_$SCNV	Convert D.P. FAC to Single Precision
FPP_$DCNV	Convert Single Precision FAC to D.P.
FPP_$SSQR	Take square root of FAC
FPP_$DSQR	Take square root of DFAC
FPP_$SEXP	EXP(<FAC>)
FPP_$DEXP	DEXP(<DFAC>)
FPP_$SLOG	ALOG(<FAC>)
FPP_$DLOG	DLOG(<DFAC>)
FPP_$SSIN	SIN(<FAC>)
FPP_$DSIN	DSIN(<DFAC>)
FPP_$SCOS	COS(<FAC>)
FPP_$DCOS	DCOS(<DFAC>)
FPP_$STAN	TAN(<FAC>)
FPP_$DTAN	DTAN(<DFAC>)
FPP_$SATAN	ATAN(<FAC>)
FPP_$DATEN	DATAN(<DFAC>)}
FPP_$SATAN2A	ATAN2(<FAC>, ((sp+))
FPP_$SATAN2V	ATAN2(<FAC>, (sp+)
FPP_$SATAN2C	ATAN2(<FAC>,<CONST>)
FPP_$DATAN2A	DATAN2(<DFAC>, ((sp+))
FPP_$DATAN2C	DATAN2(<DFAC>,<CONST>)
FPP_$E$21V	E$21(<FAC>, (sp)+)
FPP_$E$22A	E$22(<FAC>, ((sp)+))

FPP_$E$22V	E$22(<FAC>,(sp)+)
FPP_$E$22C	E$22(<FAC>,<CONST>)
FPP_$E$61V	E$61(<DFAC>,(sp)+)
FPP_$E$62A	E$62(<DFAC>,((sp)+))
FPP_$E$62V	E$62(<DFAC>,(sp)+)
FPP_$E$62C	E$62(<DFAC>,<CONST>)
FPP_$E$66A	E$66(<DFAC>,((sp)+))
FPP_$E$66C	E$66(<DFAC>,<CONST>)
FPP_$STRUNC	FAC := Int[FAC]
FPP_$SNINT	FAC := Nint[FAC] (Nearest integer)
FPP_$DTRUNC	D.P. FAC := Int[FAC]
FPP_$DNINT	D.P. FAC := Nint[FAC] (Nearest integer)
FPP_$SMINV	FAC := Min[FAC, (SP)+]
FPP_$SMINA	FAC := Min[FAC, ((SP)+)]
FPP_$SMINC	FAC := Min[FAC, Next 4 bytes]
FPP_$SMAXV	FAC := Max[FAC, (SP)+]
FPP_$SMAXA	FAC := Max[FAC, ((SP)+)]
FPP_$SMAXC	FAC := Max[FAC, Next 4 bytes]
FPP_$DMINA	D.P. FAC := Min[FAC, ((SP)+)]
FPP_$DMINC	D.P. FAC := Min[FAC, Next 8 bytes]
FPP_$DMAXA	D.P. FAC := Max[FAC, ((SP)+)]
FPP_$DMAXC	D.P. FAC := Max[FAC, Next 8 bytes]
FPP_$CLA	Complex FAC := ((SP)+)
FPP_$CLC	Complex FAC := Next 8 Bytes
FPP_$CAA	Complex FAC := FAC + ((SP)+)
FPP_$CAC	Complex FAC := FAC + Next 8 bytes
FPP_$CSA	Complex FAC := FAC - ((SP)+)
FPP_$CSC	Complex FAC := FAC - Next 8 bytes
FPP_$CISA	Complex FAC := ((SP)+) - FAC
FPP_$CISC	Complex FAC := Next 8 bytes - FAC
FPP_$CMA	Complex FAC := FAC * ((SP)+)
FPP_$CMC	Complex FAC := FAC * Next 8 bytes
FPP_$CDA	Complex FAC := FAC / ((SP)+)
FPP_$CDC	Complex FAC := FAC / Next 8 bytes
FPP_$CIDA	Complex FAC := ((SP)+) / FAC
FPP_$CIDC	Complex FAC := Next 8 bytes / FAC
FPP_$CSTA	Complex Store FAC thru (SP)+
FPP_$CSTX	Complex Store FAC thru (SP)+ and exit
FPP_$CSWAP	Complex Exchange Real and Imaginary parts of FAC
FPP_$CCNV	Convert Single Prec to Complex (set imagFAC := 0)
FPP_$CCONJ	Complex Conjugate (imagFAC := -imagFAC)
FPP_$SLUV	FAC := Float[(SP)+] (Float unsigned 32-bit integer)
FPP_$DLUV	D.P. FAC := Float[(SP)+] (Dbl Flt unsigned 32-bit int)
fpp_$dcla	DP Complex FAC := ((SP)+)
fpp_$dclc	DP Complex FAC := Next 16 Bytes
fpp_$dcaa	DP Complex FAC := FAC + ((SP)+)
fpp_$dcac	DP Complex FAC := FAC + Next 16 bytes
fpp_$dcsa	DP Complex FAC := FAC - ((SP)+)
fpp_$dcsc	DP Complex FAC := FAC - Next 16 bytes
fpp_$dcisa	DP Complex FAC := ((SP)+) - FAC
fpp_$dcisc	DP Complex FAC := Next 16 bytes - FAC
fpp_$dcma	DP Complex FAC := FAC * ((SP)+)
fpp_$dcmc	DP Complex FAC := FAC * Next 16 bytes
fpp_$dcda	DP Complex FAC := FAC / ((SP)+)
fpp_$dcdc	DP Complex FAC := FAC / Next 16 bytes
fpp_$dcida	DP Complex FAC := ((SP)+) / FAC

fpp_$dcidc	DP Complex FAC := Next 8 bytes / FAC
fpp_$dcsta	DP Complex Store FAC thru (SP)+
fpp_$dcstx	DP Complex Store FAC thru (SP)+ and exit
fpp_$dcswap	DP Complex Exchange Real and Imaginary parts of FAC
fpp_$dccnv	DP Convert Double Prec to Complex (set imagFAC := 0)
fpp_$dcconj	DP Complex Conjugate (imagFAC := -imagFAC)
FPP_$CCVX	Complex Compare FAC:(SP)+; set cc's & D0.L, exit
FPP_$CCAX	Complex Compare FAC:((SP)+); set cc's & D0.L, exit
FPP_$CCCX	Complex Compare FAC:Next 8 bytes; set cc's & D0.L, exit
FPP_$dav	FAC := FAC - (SP)+ (Double precision Add Value)
FPP_$dmv	FAC := FAC - (SP)+ (Double precision Multiply Value)
FPP_$didv	FAC := FAC - (SP)+ (Double precision Divide Value)
FPP_$dsv	FAC := FAC - (SP)+ (Double precision Subtract Value)

RING/DISK

```
address: Controller #2    [ BC00 | FF9C00 ]
         Controller #1    [ B800 | FFA000 ]
```

(today's Controllers are #2's)

Address	Write function	Read function
00BC00	Cylinder MSByte	Cylinder MSByte
00BC01		-
00BC02	Cylinder LSByte	Cylinder LSByte
00BC03		-
00BC04	Disk command register	Disk status register
00BC05		-
00BC06	(reserved hole)	-
00BC07		-
00BC08	Head number	-
00BC09		-
00BC0A	Sector	Node ID 1
00BC0B		-
00BC0C	Controller command	Node ID 2
00BC0D		-
00BC0E	(reserved hole)	Node ID 3
00BC0F		-
00BC10	Network type mask	Disk status MSB
00BC11		Disk status LSB
00BC12	Network receive command	Network rcv stat MSB
00BC13		Network rcv stat LSB
00BC14	Network transmit command	Network tx status MSB
00BC15		Network tx status LSB
00BC16		-
00BC17		-
00BC18	Disk Interrupt ACK	-
00BC19		-
00BC1A	Network Trans Interrupt ACK	-
00BC1B		-
00BC1C	Network Rec Interrupt ACK	-
00BC1D		-
00BC1E		-

00BC1F		-

Address	Write function	Read function
00BC40	Disk DMA address 0	Disk DMA address 0
00BC41		-
00BC42	Disk DMA count 0	Disk DMA count 0
00BC43		-
00BC44	Disk DMA Address 1	Disk DMA Address 1
00BC45		-
00BC46	Disk DMA count 1	Disk DMA count 1
00BC47		-
00BC48	Transmit DMA Address	Transmit DMA Address 0
00BC49		-
00BC4A	Transmit DMA count 0	Transmit DMA count 0
00BC4B		-
00BC4C	Transmit DMA address	Transmit DMA address 1
00BC4D		-
00BC4E	Transmit DMA count 1	Transmit DMA count 1
00BC4F		-
00BC50	DMA command	DMA status
00BC51		-
00BC52	DMA request	-
00BC53		-
00BC54	DMA single mask	-
00BC55		-
00BC56	DMA mode	-
00BC57		-
00BC58	DMA clear byte pointer	DMA temporary
00BC59		-
00BC5A	DMA master clear	-
00BC5B		-
00BC5C	-	-
00BC5D		-
00BC5E	DMA all masks	-
00BC5F		-

Address	Write function	Read function
00BC60	Receive 0 DMA address	Receive 0 DMA address 0
00BC61		-
00BC62	Receive 0 DMA count 0	Receive 0 DMA count 0
00BC63		-
00BC64	Receive 0 DMA address	Receive 0 DMA address 1
00BC65		-
00BC66	Receive 0 DMA count 1	Receive 0 DMA count 1
00BC67		-
00BC68	Receive 1 DMA address	Receive 1 DMA address 0
00BC69		-
00BC6A	Receive 1 DMA count 0	Receive 1 DMA count 0
00BC6B		-
00BC6C	Receive 1 DMA address	Receive 1 DMA address 1
00BC6D		-
00BC6E	Receive 1 DMA count 1	Receive 1 DMA count 1
00BC6F		-
00BC70	DMA command	DMA status
00BC71		-
00BC72	DMA request	-
00BC73		-
00BC74	DMA single mask	-
00BC75		-
00BC76	DMA mode	-
00BC77		-
00BC78	DMA clear byte pointer	DMA temporary
00BC79		-
00BC7A	DMA master clear	-
00BC7B		-
00BC7C	-	-
00BC7D		-
00BC7E	DMA all masks	-
00BC7F		-

Cylinder Address Register

MSB address: [BC00 | FF9C00]

15	14	13	12	11	10	9	8
0	0	0	0	0	C10	C09	C08

LSB address: [BC02 | FF9C02]

15	14	13	12	11	10	9	8
C07	C06	C05	C04	C03	C02	C01	C00

PRIAM Command Register

address: [BC04 | FF9C04]

```
   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
|  0   |  0   |  0   |  0   |  0   |  0   |  0   |  0   |
+------+------+------+------+------+------+------+------+
Sequence Up =                          0      0      1
Sequence Down =                        0      1      0
Restore =                              0      1      1
Seek =                                 1      0      0
Fault Reset =                          1      0      1
```

PRIAM Status Register

address: [BC04 | FF9C04]

```
   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
|CMDRJT|WRITPT|DRVFLT| BUSY | CYL 0|SEEKFT|SEEKOK| READY|
+------+------+------+------+------+------+------+------+
```

READY - The drive is up to speed, servo is locked on track, and the unit is in a state to read or write.

SEEKOK- SEEK COMPLETE - This bit indicates the seek has completed successfully.

SEEKFT- SEEK FAULT - A fault was detected during a seek operation.

CYL 0 - Head on cylinder 0.

BUSY - The drive is in the process of executing a command and will not accept any other commands.

DRVFLT- DRIVE FAULT - A fault was detected during a write operation or a drive unsafe condition was detected.

WRITPT- WRITE PROTECT - The head selected is write protected. Write protection is set by switches in the drive or when the drive isn't sequenced up.

CMDRJT- Control or register load command received while drive is not ready, or improper command received.

Head/Drive Select

```
address:   [ BC08 | FF9C08 ]

   15      14      13      12      11      10       9       8
+------+------+------+------+------+-------+-------+----+
| SEL3\| SEL2\| SEL1\| SEL0\|   x  |  H2\  |  H1\  |H0\ |
+------+------+------+------+------+-------+-------+----+
SEL3-SEL0 select the DRIVE:

         Drive #0 =  1   1   1   0
                1 =  1   1   0   1
                2 =  1   0   1   1
                3 =  0   1   1   1

H2-H0 select the HEAD:

         Head  #0 =  1   1   1
                1 =  1   1   0
                2 =  1   0   1
```

Sector Select

```
address:   [ BC0A | FF9C0A ]

   15      14      13      12      11      10       9       8
+------+------+------+------+------+------+------+------+
|   x  |   x  |   x  |  S4  |  S3  |  S2  |  S1  |  S0  |
+------+------+------+------+------+------+------+------+
S4-S0 select the SECTOR(0-17):

      Sector #0 =  0   0   0   0   0
              1 =  0   0   0   0   1
              2 =  0   0   0   1   0
              3 =  0   0   0   1   1
      etc.
```

Disk Controller Command

```
address:   [ BC0C | FF9C0C ]
   15      14      13      12      11      10       9       8

+------+------+------+------+------+------+------+------+
| READ | WRITE|FORMAT|   x  |   x  |   x  |   x  | INTE |
+------+------+------+------+------+------+------+------+
```

Disk Controller Status

address: [BC10 | FF9C10]

```
   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
| BUSY |READY\|CRCERR|TIMOUT|  0   |  0   |BUSERR| OVRUN|
+------+------+------+------+------+------+------+------+
    7      6      5      4      3      2      1      0
+------+------+------+------+------+------+------+------+
|  0   |  0   |  x   |  x   |  x   |  x   |  x   |  x   |
+------+------+------+------+------+------+------+------+
```

BUSY - Controller has a disk READ, WRITE or FORMAT
        request pending or in progress.

READY - PRIAM ready bit, indicates that the PRIAM is
        sequenced up, on cylinder and ready to accept
        READS, WRITES or FORMATS.

CRCERR- CRC on the last disk operation was in error.
        This bit is undefined except for disk READs.

TIMOUT- The last disk operation didn't finish before 3
        revolutions of the disk. During a READ or WRITE
        operation it is an indication that the drive has
        has been positioned to the wrong cylinder,
        sector number is too large, surface is unformatted, or there was a CRC error on the disk header.

BUSERR- A BUS error occurred during last DMA transfer.
        DMA address register points past the erroring
        address.

OVRUN - DMA overrun occurred during last disk operation.

Status	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
Controller busy	1	x	x	x	0	0	x	x	0	0	x	x	x	x	x	x
PRIAM not ready	x	1	x	x	0	0	x	x	0	0	x	x	x	x	x	x
Time out-sector error	0	0	x	1	0	0	0	0	0	0	x	x	x	x	x	x
DMA bus error	0	0	x	0	0	0	1	x	0	0	x	x	x	x	x	x
DMA over run	0	0	x	0	0	0	0	1	0	0	x	x	x	x	x	x
CRC error	0	0	1	0	0	0	0	0	0	0	x	x	x	x	x	x
Disk operation ok	0	0	0	0	0	0	0	0	0	0	x	x	x	x	x	x

Packet Type

address: [BC10 | FF9C10]

```
   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
|BRDCST|  T6  |  T5  |  T4  |  T3  |  T2  |  T1  |  T0  |
+------+------+------+------+------+------+------+------+
```

BRDCST - Accept broadcast messages

T6-T0 - Type mask, accept message if any bits in type
         field of message are set in corresponding bit of
         type mask.

Receive Command Register

address: [BC12 | FF9C12]

```
   15      14      13      12      11      10       9       8
+-------+-------+-------+-------+-------+-------+-------+-------+
|  REC  | STOP  |  CON  |  DIS  |   x   |   x   |   x   | INTE  |
+-------+-------+-------+-------+-------+-------+-------+-------+
```

REC - Enable receive. This informs the controller that the registers in the controller are set up to receive a message.

STOP - Abort a previously posted REC. This will be successful if the controller has not already seen a message begin.

CON - Connect to the ring.

DIS - Disconnect from the ring. This also happens on reset.

INTE - Enable interrupts to happen after a receive.

Receive Status Register

address: [BC12 | FF9C12]

```
   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
| BUSY |  CON |CRCERR|TIMOUT|   0  |EORERR|BUSERR| OVRUN|
+------+------+------+------+------+------+------+------+
    7      6      5      4      3      2      1      0
+------+------+------+------+------+------+------+------+
|MSGER\|ACKPE\|PKTERR|   0  |   0  |   0  |ESBERR|BPHERR|
+------+------+------+------+------+------+------+------+
```

BUSY - The controller is currently observing a message or a request is still pending.

CON - The controller is currently connected to the ring.

CRCERR - The last message received had a CRC error.

TIMOUT - The last message received started but didn't finish in 2**12 byte times.

EORERR - End of range error. One or both of the message fields was bigger than the DMA channel was set up for.

BUSERR - A BUS error occurred during the DMA transfer. The DMA address register is pointing one location past the point of the error.

OVRUN - A DMA overrun occurred during the last receive.

MSGER - No message error occured during the last receive. A message error is any error that can be detected by the microcode of the controller. Generally it checks for the packet protocol.

ACKPE - Ack parity OK. No parity error was discovered in the ack bytes in the last receive.

PKTERR - Either the transmitter or another receiver had an error in the packet. If the transmitter had an error in the transmission of the packet, one of the following errors occurred in the transmitter:

ESBERR - Elastic Store Buffer Error
BPHERR - Bi-Phase Error
OVRUN - DMA Overrun
BUSERR - DMA bus error
ACKPAR - Ack byte parity error
SFTABT - Software abort
NCOPY - No receiver enabled to copy this msg
MSGERR - Message error

If the receiver had an error in the reception of the packet, one of the following errors occurred in the receiver:

ESBERR - Elastic Store Buffer Error
BPHERR - Bi-Phase Error
OVRUN - DMA Overrun
BUSERR - DMA bus error
EORERR - End of Range error in DMA channel
CRCERR - CRC error in packet
ACKPAR - Ack byte parity error

ESBERR - An error occurred in the modems elastic store buffer.

BPHERR - An error occurred in the modems decode of the Bi-phase encoded data.

Status	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
Controller Busy	1	x	x	x	0	x	x	x	x	x	x	0	0	0	x	x
Connected to Ring	x	1	x	x	0	x	x	x	x	x	x	0	0	0	x	x
DMA Bus Error	0	x	x	x	0	x	1	x	x	x	x	0	0	0	x	x
DMA Overrun	0	x	x	x	0	x	0	1	x	x	x	0	0	0	x	x
Bi-phase Error	0	1	x	x	0	x	0	0	x	x	x	0	0	0	x	1
Elastic Buffer Error	0	1	x	x	0	x	0	0	x	x	x	0	0	0	1	0
Time-out Error	0	1	x	1	0	x	0	0	x	x	x	0	0	0	0	0
Sync Protocol Error	0	1	x	0	0	x	0	0	0	x	x	0	0	0	0	0
Ack Byte Parity Error	0	1	x	0	0	x	0	0	1	0	x	0	0	0	0	0
CRC Error	0	1	1	0	0	x	0	0	1	1	x	0	0	0	0	0
End of Range Error	0	1	0	0	0	1	0	0	1	1	x	0	0	0	0	0
Packet Error	0	1	0	0	0	0	0	0	1	1	1	0	0	0	0	0
Received OK	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Transmit Command Register

address: [BC14 | FF9C14]

```
   15     14     13     12     11     10      9      8
+------+------+-------------+------+------+------+------+
| TMT  | STOP |FTRANS|CH1DIS|   x  |DELAY |NDELAY| INTE |
+------+------+------+------+------+------+------+------+
```

TMT - Enable transmit. This bit informs the controller that all the registers are set up for a message to be transmitted when the next token is received on the network.

STOP - Stop a previously posted TMT. This will abort a request for a transmit. It will abort a packet if currently being transmitted.

FTRANS - Force transmit. Allow the TMT to start even though no token has been seen. This bit must be accompanied by the TMT bit.

CH1DIS - Disable the second transmit DMA channel from putting out any data. This will make for a 0 length data field in the message.

DELAY - Enable an additional 7 bit delay into the length of the network. This may be required to support the recirculation of the token, which is 9 bits.

NDELAY - Disable the 7 bit delay.

INTE - Enable an interrupt to be generated at the completion of the transmit.

Transmit Status Register

address: [BC14 | FF9C14]

```
   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
| BUSY |   0  |   0  |TIMOUT|   0  |   0  |BUSERR| OVRUN|
+------+------+------+------+------+------+------+------+
    7      6      5      4      3      2      1      0
+------+------+------+------+------+------+------+------+
|MSGER\|ACKPE\|PKTERR| NCOPY| COPY | WACK |ESBERR|BPHERR|
+------+------+------+------+------+------+------+------+
```

BUSY - The controller is currently transmitting a message or the request is still pending.

TIMOUT - The last message transmitted started but didn't finish in 2**12 byte times.

BUSERR - A BUS error occurred during the DMA transfer. The DMA address register is pointing 1 location past the point of the error.

OVRUN - A DMA overrun occurred during the last transmit. transmit. A message error is any error that can be detected by the microcode of the controller. It is generally the packet protocol that is checked for.

MSGER\ - No message error occured during the last receive. A message error is any error that can be detected by the microcode of the controller. Generally it checks for the packet protocol.

ACKPE\ - Ack parity OK. No parity error was discovered in the ack bytes in the last transmit.

PKTERR - Either the transmitter or the receiver had an error in the packet.
If the transmitter had an error in the transmission of the packet, one of the following errors occurred in the transmitter:

ESBERR - Elastic Store Buffer Error
BPHERR - Bi-Phase Error
OVRUN - DMA Overrun
BUSERR - DMA bus error
ACKPAR - Ack byte parity error
SFTABT - Software abort
NCOPY - No receiver was enabled to copy this message
MSGERR - Message error

If the receiver had an error in the reception of the packet, one of the following errors occurred in the receiver:

ESBERR - Elastic Store Buffer Error
BPHERR - Bi-Phase Error
OVRUN - DMA Overrun
BUSERR - DMA bus error
EORERR - End of Range error in DMA channel
CRCERR - CRC error in packet
ACKPAR - Ack byte parity error

NCOPY - No receiver observed his node address in this packet and/or was enabled to copy this message.
MSGCPY - The receiver successfully copied the message.
WACK - A receiver observed his node id, but wasn't enabled to copy this message.
ESBERR - An error occurred in the modem's elastic store buffer.
BPHERR - An error occurred in the modem's decode of the Bi-phase encoded data.

Status	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0
Controller busy	1	0	0	x	0	0	x	x	x	x	x	x	x	x	x	x
DMA Bus Error	0	0	0	x	0	0	1	x	x	x	x	x	x	x	x	x
DMA Overrun	0	0	0	x	0	0	0	1	x	x	x	x	x	x	x	x
Bi-Phase Error	0	0	0	x	0	0	0	0	x	x	x	x	x	x	x	1
Elastic Buffer Error	0	0	0	x	0	0	0	0	x	x	x	x	x	x	1	0
Time-out Error	0	0	0	1	0	0	0	0	x	x	x	x	x	x	0	0
Sync Protocol Error	0	0	0	0	0	0	0	0	0	x	x	x	x	x	0	0
Ack Byte Parity Error	0	0	0	0	0	0	0	0	1	0	x	x	x	x	0	0
Negative ACK	0	0	0	0	0	0	0	0	1	1	x	1	0	0	0	0
Wack	0	0	0	0	0	0	0	0	1	1	x	x	x	1	0	0
Packet Error	0	0	0	0	0	0	0	0	1	1	1	0	x	x	0	0
Message Copied	0	0	0	0	0	0	0	0	1	1	0	0	1	x	0	0

Node ID Register

address: [BC0A | FF9C0A]
[BC0C | FF9C0C]
[BC0E | FF9C0E]

15	14	13	12	11	10	9	8
ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0

DMA Control/Status Registers

DMA Address address: [base + 0]|
['' + 4]
['' + 8]
['' + C]

15	14	13	12	11	10	9	8
A08	A07	A06	A05	A04	A03	A02	A01
IVA16	IVA15	IVA14	IVA13	IVA12	IVA11	IVA10	A09

Notice that the address is shifted right by 1.

DMA Count address: [base + 2]
['' + A]
['' + E]

15	14	13	12	11	10	9	8
C07	C06	C05	C04	C03	C02	C01	C00
C15	C14	C13	C12	C11	C10	C09	C08

C(15-00) is the desired count in words minus 1.

DMA Command Register

address: [base + 10]

15	14	13	12	11	10	9	8
0	1	1	0	0	x	0	0

BIT 7 - DACK Sense Active High
6 - DREQ Sense Active Low
5 - Extended Write
4 - Rotating Priority
3 - Compressed Timing
2 - Controller Disable
1 - Channel 0 Address Hold Enable
0 - Memory to Memory Enable

DMA Mode Register

```
address:    [ base + 16 ]

   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
|  0   |  0   |  0   |  x   |  x   |  x   |  x   |  x   |
+------+------+------+------+------+------+------+------+
BIT 7,6 - 00 - Demand Mode
          01 - Single Mode
          10 - Block Mode
          11 - Cascade Mode
    5 - Address Decrement
    4 - Autoinitialize
    3,2 - 00 - Verify Transfer
          01 - Write Transfer
          10 - Read Transfer
          11 - undefefined
    1,0 - Channel Select
```

DMA Request Register

```
address:    [ base + 12 ]

   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
|  x   |  x   |  x   |  x   |  x   |  x   |  x   |  x   |
+------+------+------+------+------+------+------+------+
BIT 2 - Request Bit
    1,0 - Channel Select
```

DMA Mask Register

```
address:    [ base + 14 ]

   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
|  x   |  x   |  x   |  x   |  x   |  x   |  x   |  x   |
+------+------+------+------+------+------+------+------+
BIT 2 - Set Mask Bit
    1,0 - Channel Select
```

DMA ALL Mask Register

```
address:   [ base + 1E ]

   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
|  x   |  x   |  x   |  x   |  x   |  x   |  x   |  x   |
+------+------+------+------+------+------+------+------+
BIT 3 - Set Channel 3 Mask Bit
    2 - Set Channel 2 Mask Bit
    1 - Set Channel 1 Mask Bit
    0 - Set Channel 0 Mask Bit
```

DMA Status Register

```
address:   [ base + 10 ]

   15     14     13     12     11     10      9      8
+------+------+------+------+------+------+------+------+
|      |      |      |      |      |      |      |      |
+------+------+------+------+------+------+------+------+
BIT 7 - Channel 3 Request
    6 - Channel 2 Request
    5 - Channel 1 Request
    4 - Channel 0 Request
    3 - Channel 3 has Reached TC
    2 - Channel 2 has Reached TC
    1 - Channel 1 has Reached TC
    0 - Channel 0 has Reached TC
```

SERIAL I/O INTERFACE

Data Input/Output,	Line 0	[8400	FFB000]
Control/Status,	Line 0	[8402	FFB002]
Data Input/Output,	Line 1	[8404	FFB004]
Control/Status,	Line 1	[8406	FFB006]
Data Input/Output,	Line 2	[8408	FFB008]
Control/Status,	Line 2	[840A	FFB00A]
Data Input/Output,	Line 3	[840C	FFB00C]
Control/Status,	Line 3	[840E	FFB00E]
Bit-Rate Generator,	Line 0	[8482	FFB082]
Bit-Rate Generator,	Line 1	[8486	FFB086]
Bit-Rate Generator,	Line 2	[848A	FFB08A]
Bit-Rate Generator,	Line 3	[848E	FFB08E]

Data				
03	02	01	00	Bit-Rate
0	0	0	0	50
0	0	0	1	75
0	0	1	0	110
0	0	1	1	134.5
0	1	0	0	150
0	1	0	1	300
0	1	1	0	600
0	1	1	1	1200
1	0	0	0	2400
1	0	0	1	2000
1	0	1	0	2400
1	0	1	1	3600
1	1	0	0	4800
1	1	0	1	7200
1	1	1	0	9600
1	1	1	1	19200

Display Speaker

SIO line 0 is wired to the Display Speaker.

- asserting RTS will emit a constant sound
- toggling DTR with RTS set emits different tones

SIO Write Control/Status Registers

```
WRITE REGISTER 0
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
                              0    0    0  Reg 0 Select
                              0    0    1   '' 1   ''
                              0    1    0   '' 2   ''
                              0    1    1   '' 3   ''
                              1    0    0   '' 4   ''
                              1    0    1   '' 5   ''
                              1    1    0   '' 6   ''
                              1    1    1   '' 7   ''
               0    0    0   NULL code
               0    0    1   Send Abort
               0    1    0   Reset External Status Int
               0    1    1   Channel Reset
               1    0    0   Enable Interrrupton next Rx Char
               1    0    1   Reset Tx Interrupt pending
               1    1    0   Error Reset
               1    1    1   Return from Interrrupt(CH-A only)
   0     0   NULL Code
   0     1   Reset Rx CRC Checker
   1     0   Reset Tx CRC Generator
   1     1   Reset Tx Underrun/EOM Latch
```

```
WRITE REGISTER 1
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
D7 - Wait/Ready Enable
D6 - Wait/Ready Function
D5 - Wait/Ready on R/T
D4 - D3
 0   0   Rx Int. Disable
 0   1   Rx Int. on 1st Character
 1   0   Int. on all Rx Chars (Parity affects vector)
 1   1    "    "  "  "     "    "  does not   "   ")
D2 - Status affects vector(CH-B only)
D1 - Tx Interrupt only
D0 - EXT Interrupt Enable
```

```
WRITE REGISTER 2 (CHANNEL B ONLY)
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
D7 - D0 = Interrupt Vectors V7 - V0
```

WRITE REGISTER 3

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
D7 - D6
 0    0    Rx 5 Bits/Character
 0    1    Rx 7 Bits/Character
 1    0    Rx 6 Bits/Character
 1    1    Rx 8 Bits/Character
D5 - Auto Enables
D4 - Enter Hunt Phase
D3 - Rx CRC Enable
D2 - Address Search Mode(SDLC)
D1 - SYNC Character Load Inhibit
D0 - Rx Enable
```

WRITE REGISTER 4

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
D7 - D6
 0    0    X1 Clock Mode
 0    1    X16 Clock Mode
 1    0    X32 Clock Mode
 1    1    X64 Clock Mode
D5 - D4
 0    0    8 Bit Programmed SYNC
 0    1    16 Bit Programmed SYNC
 1    0    SDLC mode(01111110 flag pattern)
 1    1    External SYNC Mode
D3 - D2
 0    0    SYNC Modes Enable
 0    1    1 Stop Bit/Character
 1    0    1.5 Stop Bits/Character
 1    1    2 Stop Bits/Character
D1 - 0 = Odd Parity, 1 = Even Parity
D0 - Enable Parity
```

WRITE REGISTER 5

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
D7 - DTR
D6 - D5
 0    0    Tx 5 Bits(or less) Character
 0    1    Tx 7 Bits/Character
 1    0    Tx 6 Bits/Character
 1    1    Tx 8 Bits/Character
D4 - Send BREAK
D3 - Tx Enable
D2 - SDLC/CRC-16
D1 - RTS
D0 - Tx CRC Enable
```

WRITE REGISTER 6

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
```

D7 - D0 = SYNC Bits 7 - 0

WRITE REGISTER 7

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
```

D7 - D0 = SYNC Bits 15 - 8

SIO Read Control/Status Registers

READ REGISTER 0

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
```

D7 - BREAK/ABORT *
D6 - Tx Underrun/EOM *
D5 - CTS *
D4 - SYNC/Hunt *
D3 - DCD *
D2 - Tx Buffer Empty
D1 - Interrupt Pending(CH-A only)
D0 - Rx Character Available
  * Used with External/Status Interrupt Mode

READ REGISTER 1

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
```

D7 - End of Frame(SDLC)
D6 - CRC/Framing Error
D5 - Rx Overrun Error
D4 - Parity Error

D3	D2	D1	I Field Prev Byte	I Field 2nd Prev Byte *
1	0	0	0	3
0	1	0	0	4
1	1	0	0	5
0	0	1	0	6
1	0	1	0	7
0	1	1	0	8
1	1	1	1	8
0	0	0	2	8

D0 - All Sent
  * Residue data for 8 Rx bits/character programmed

READ REGISTER 2

```
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
```

D7 - D0 = Interrupt Vectors V7 - V0

CHAPTER 9

DN460,DN660,DSP160

ADDRESS SPACE

physical		virtual
200800	traps	0
400	prom	0400-> 7FFF
200800	phys mem	0000400->FF7FFFF
200000	debug data	F800000->F8007FF
40000	disp2_mem	FFA0000->FFBFFFF
20000	disp1_mem	FFC0000->FFDFFFF
10000	multibus	FFE0000->FFEFFFF
E000	color	FFF6000
E800	color	FFF6800
B000	sequencer	(not mapped)
C000	wcs	(not mapped)
B400	decoder i/o	(not mapped)
D000	i_cache0 i/o	(not mapped)
D400	i_cache1 i/o	(not mapped)
D800	i_cache2 i/o	(not mapped)
DC00	i_cache3 i/o	(not mapped)
sftw	chksum buffer	FFF8800
sftw	zeroing buffer	FFF8C00
FC00	memory ctl	FFF9000
F400	disp 2	FFF9400
F000	disp 1	FFF9800
BC00	ring 2	FFF9C00
B800	ring 1	FFFA000
8C00	floppy	FFFA800
8800	timers	FFFAC00
8400	sios	FFFB000
8000	control panel	FFFB400
9000	io map	FFFF800

BLT REGISTERS

Refer to the BLT REGISTERS section of CHAPTER 8, DN400,DN420,DN600, for this infornation.

CACHE

MD commands (DC, IC) are used to turn data and instruction caches on and off. Machine must be in physical mode to toggle the instruction cache. Data cache is not controllable from CPIO. Sysboot (when running in CPU) turns both caches on.

CPU CONTROL REGISTERS

DNx60 is a micro-coded machine with a forward-mapped address translation mechanism (using in-memory page tables). There is no PTT or PFT. Some of the MMU Control Register functions described above are in the Control Panel Registers, while much of the control of the CPU is achieved via the MOVEC instruction. The following control register definitions are of interest:

*	120-13F	-	Current Region Registers (0-31)
*	11F	-	Current ASID
	11E	-	Current Floating Point ASID
	11D	-	Purge Translation Buffer. Values defined: 1 => purge VA in Control Register 11C 2 => purge entire TB 4 => purge half of TB containing VA in 11C
	11C	-	VA used in TB purge
*	119	-	Address Translation and Data Cache Bit 0 : Enable/Disable Address Translation Bit 1 : Enable/Disable Data Cache Bit 7 : Domain Bit
	113	-	Read Hardware and Microcode Revision Levels (Requires Micro Exec)

* These registers/functions are directly supported by the PROM.

CONFIGURATION

```
+------------+   +-------------+   +----------+
|   CPU 1    |   |   MEMORY    |   |   CPIO   |
+------------+   +-------------+   +----+-----+
|   CPU 2    |--------||                |
+------------+        ++----------------+
                      ||
       PHOEBUS        ||           +----+-----+
+---------------------+|===========| FLOPPY   |
|+---------------------+           +----------+
||                                  (Shugart)
||   +-------------+
|+---|    BLOCK    |---------------------+
|+---| MULTIPLEXOR |------++------------+|
||   +-------------+      ||            ||
||                     +-------+     +-------+
||                     | RING  |     | DISK  |
||                     +-------+     +-------+
||   +--------------+
|+---|  NODE PERI-  |-----+
|+---|PHERAL ADAPTER|----+|
||   +--------------+    || MULTIBUS
||                       ||
||   +--------------+    ||
|+===|    TIMER     |    ||  +--------------+
||   +--------------+    |+--| LINE PRINTER |
||                       |+--|  CONTROLLER  |
||   +--------------+    ||  +------++------+
|+===|  BOOT PROM   |    ||        ||
||   +--------------+    ||  +------++------+
||                       ||  | LINE PRINTER |
||   +--------------+    ||  +--------------+
|+===|  CAL CLOCK   |    ||   (Printronix)
||   +--------------+    ||
||                       ||   (Tapemaster)
||   +--------------+    ||  +------------+
|+===|  SIO LINES   |    |+--|  MAG TAPE  |
||   +--+--+--+--+--+    |+--| CONTROLLER |
||      |  |  |  |       ||  +-----++-----+
||     0|  1  2  3       ||      ||
||    +---+  Spinwriter  ||   +----++----+
||    |KBD|  Tablet      ||   | MAG TAPE |
||    +---+  HASP, 3270  ||   +----------+
||                       ||   (Cipher-F880)
||   +---------+ +----+  ||                +------------+
|+---+ DISPLAY |=|TUBE|  |+----------------|    SMD     |
|+---+ MEMORY  | +----+  |+----------------| CONTROLLER |
||   +---++----+         ||                +-----++-----+
||       ||              |+--- TWO FREE          ||
||   +---++----+         |+----   SLOTS    +-----++-----+
|+===|   BLT   |         ||    |           |    AMPEX   |
||   +---------+         ||    VERSATEC    | 300 MB SMD |
||                       ||    ETHERNET    +------------+
```

CONTROL PANEL (CPIO) REGISTERS

CPIO Control Register

```
Address: [ 8000 | FFFB400]

 15      8 7       0
+--------+--------+
|K-------|---K-HSM|
+--------+--------+
```

```
K  - Enable power-down switch (in two places)
H  - Halt CPIO
S  - CPIO becomes bus slave
M  - CPIO becomes NOT bus master
```

CPIO Status Register

```
Address:  [ 8002 | FFFB402]

 15      8 7       0
+--------+--------+
|K------C|BINK-HSM|
+--------+--------+
```

```
K  - Power-down switch enabled (in two places)
C  - 0 => This is CPIO running
     1 => This is CPU running
B  - Bus timeout
I  - Interrupt pending
N  - Normal mode, 0 => Service Mode
H  - CPIO is halted
S  - CPIO is slave
M  - CPIO is NOT master
```

LED Register

```
Address:  [ 8004 | FFFB404]

 15       8 7       0
+---------+--------+
|---------|LED DATA|
+---------+--------+
```

Micro Machine Control Registers

None of these registers have mapped addresses.

These are used for loading micro code:

WCS Control Register	C003	WCS Data/Address	C005
Decode Rams Control	B400	Decode Rams Address	B402
Decode Rams Data	B404	Decode Rams Shift	B406
Microsequencer	B000	CPIO/Microexec Port	B003

Micro Machine Control and Status [B004]:

```
 15      8 7       0
+--------+--------+
|abcde-fg|TIPMSCRE|
+--------+--------+
```

Left byte is read-only status:

a - Micro machine is frozen
b - Micro machine is in trap routine
c - Micro machine traps are enabled
d - Micro machine is fetching
e - 0 => micro machine is running
    1 => hit freeze on last instruction
f - CPIO data available in port for Micro Exec
g - Micro Exec data available in port for CPIO

Right byte may be written for control or read for status:
T - Cause Trap to Micro Exec
I - Enable Instruction Cache
P - Enable Pipeline Register
M - Macro step the micro machine
S - Micro step the micro machine
C - Clock step the micro machine
R - Start clocks (runs at address in pipeline)
E - Enable micro machine (0 => reset)

DISPLAY BOARD JUMPERS

Refer to the DISPLAY BOARD JUMPERS section of CHAPTER 8, DN400,DN420,DN600, for this information.

DISPLAY CONTROL AND STATUS REGISTER (DCSR)

Refer to the DISPLAY CONTROL AND STATUS REGISTER (DCSR) section of CHAPTER 8, DN400,DN420,DN600, for this information.

FAULT FRAME

LONG FAULT FRAME FORMAT

CPIO fault frames are the same as DN300 (68010). CPU short fault frames are also the same as DN300. Long fault frame used for bus error, address error, access violation, and region, segment and page faults is:

```
          15                      0
         +-----------------------+
     +00 |    STATUS REGISTER    |
         +-----------------------+
     +02 |        PROGRAM        |
         +- - - - - - - - - - - -+
         |        COUNTER        |
         +-----------------------+
     +06 |0|1|0|0| VECTOR OFFSET |
         +-----------------------+
     +08 |       NOT USED        |
         +-----------------------+
     +0A |       FAULTING        |
         +- - - - - - - - - - - -+
         |        ADDRESS        |
     +0E +-----------------------+
         |    '0BAD' SIGNAL      |
     +10 +-----------------------+
```

FAULT TYPES

Group	Exception	Processing
0	Reset Bus Error Address Error	Current instruction is aborted.
1	Trace Interrupt Illegal Ins. Privilege Ins.	Exception occurs before next instruction.
2	TRAP, TRAPV CHK Zero Divide	Processed by normal instruction execution.

Group 0 exceptions have the highest priority.

FAULT VECTORS

Exception vectors at [200800 | 0]

Vector	Address	Assignment
00	000	Reset: Initial SSP
	004	Reset: Initial PC
02	008	Bus Error
03	00C	Address Error
04	010	Illegal Instruction
05	014	Zero Divide
06	018	CHK Instruction
07	01C	TRAPV Instruction
08	020	Privilege Violation
09	024	Trace
0A	028	Unimplemented instruction
0B	02C	Unimplemented instruction
0C-0D	030	(Unassigned, reserved)
0E	038	Invalid Stack Format
0F-17	03C	(Unassigned, reserved)
18	060	Spurious Interrupt
19-1F	064	(Unassigned, reserved)
20-2F	080	TRAP Instruction Vectors
30	0C0	(Unassigned, reserved)
31	0C4	Floating Point Inexact Result
32	0C8	Floating Point Divide by Zero
33	0CC	Floating Point Underflow
34	0D0	Floating Point Operand Error
35	0D4	Floating Point Overflow
36-3F	0D8	(Unassigned, reserved)
40-7F	100	User Int Vectors - unused
80	200	Region Fault
81	204	Segment Fault
82	208	Page Fault
83	20C	Access Violation
84	210	Floating Point ASID Trap
85-8F	214	(Unassigned, reserved)
90-9F	240	Ring/disk board
A0-AF	280	User Int Vectors - unused
B0-B2	2C0	INT0/ INT2 -
B3	2CC	INT3/ - Tape Controller
B4	2D0	INT4/ - Storage Module
B5	2D4	INT5/ -
B6	2D8	INT6/ - Line Printer
B7	2DC	INT7/ - Parallel Input
B8-C3	2E0	User Int Vectors -- unused
C4-CD	310	Unused
CE-CF	338	Color
D0-EF	340	Unused
F0	3C0	P - ECCC (Automatic vectors)
F1	3C4	O -
F2	3C8	N - Display #2
F3	3CC	M - Floppy

F4	3D0	L - Display #1 (BLT)
F5-F7	3D4	K, J, I -
F8	3E0	Unused
F9-FA	3E4	Sio Lines (2 lines/Vector)
FB	3EC	Timers 1,2,3
FC-FE	3F0	Unused
FF	3FC	ECCU

FLOATING-POINT FORMAT

Refer to the FLOATING-POINT FORMAT section of Chapter 7, DN330/DN320/DN330 for this information.

FLOATING-POINT REGISTERS

FLoating-Point Control Register

```
 31                                            16
+-----------------------------------------------+- - -
|                     0                         |
+-----------------------------------------------+- - -

     15 14 13 12 11 10  9  8  7  6  5  4  3  2  1  0
- - -+-----+--+--+--+--+--+--+-----------------------+
     |     |  |  |  |  |  |  |         unused        |
- - -+-----+--+--+--+--+--+--+-----------------------+
        |    |  |  |  |  |  |
        |    |  |  |  |  |  > inexact decimal input (inex1)
     unused  |  |  |  |  > inexact operation (inex2)
             |  |  |  > divide by zero (dz)
             |  |  > underflow (unfl)
             |  > overflow (ovfl)
             > operand error (operr)
```

Floating Point Status Register

```
        |   Condition Code Byte |

         31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16
        +-----------+--+--+--+--+-----------------------+- - -
        |  unused   |  |  |  |  |        unused         |
        +-----------+--+--+--+--+-----------------------+- - -
                     |  |  |  |
                     |  |  |  > not a number or unordered
                     |  |  > infinity
                     |  > zero
                     > negative

             |Exception Status Byte|

             15 14 13 12 11 10  9  8                       0
        - - -+-----+--+--+--+--+--+------------------------+
             |     |  |  |  |  |  |        unused          |
        - - -+-----+--+--+--+--+--+------------------------+
                |     |  |  |  |  |
                |     |  |  |  |  > inexact operation (inex2)
             unused   |  |  |  > divide by zero (dz)
                      |  |  > underflow (unfl)
                      |  > overflow (ovfl)
                      > operand error (operr)
```

FLOPPY CONTROLLER

Refer to the FLOPPY CONTROLLER section of CHAPTER 8, DN400,DN420,DN600, for this information.

MEMORY CONTROL/STATUS REGISTERS (MCSR)

There are 64 possible memory control register addresses, beginning at [FC02 | FFF9002]. Within each range of $100 addresses, only one board may respond, but owing to the variety of boards supported ('old', 'new', interleaved, non-interleaved), a board may respond at any one of 8 addresses within that range (at intervals of $20). Also boards do not have to respond at contiguous ranges. There can be at most 4 memory boards.

Refer to Chapter 8, DN400,DN420,DN600, for control and status format.

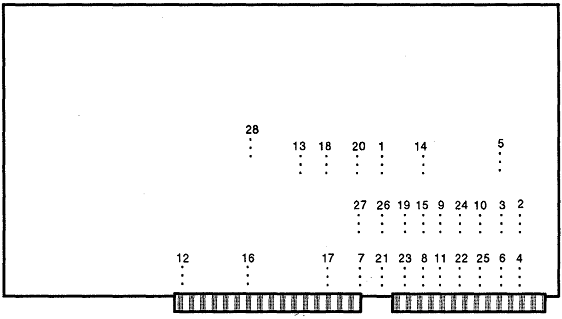
MEMORY BOARD JUMPERS

	NON-INTERLEAVED ADDRESS SPACE						INTERLEAVED ADDRESS SPACE							
		1-Mb			2-Mb		1-Mb				2-Mb			
	1	2	3	4	2	4	2	2	4	4	0	0	2	2
	0	0	0	0	0	0	0	0	0	0	8	8	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	2	0	2	0	2	0	2
	|	|	|	|	|	|	|	|	|	|	|	|	|	|
	1	2	3	4	3	5	3	3	5	5	3	3	5	5
	f	f	f	f	f	f	f	f	f	f	f	f	f	f
	f	f	f	f	f	f	f	f	f	f	f	f	f	f
	f	f	f	f	f	f	f	f	f	f	f	f	f	f
	f	f	f	f	f	f	f	f	f	f	f	f	f	f
	f	f	f	f	f	f	d	f	d	f	d	f	d	f
wp 1	u	u	u	u	u	u	u	u	u	u	u	u	u	u
wp 2	u	u	u	u	u	u	u	u	u	u	u	u	u	u
wp 3	u	u	u	d	u	u	u	u	d	d	u	u	u	u
wp 4	u	d	d	u	u	d	d	d	u	u	u	u	d	d
wp 5	d	u	d	u	d	u	u	d	u	d	u	d	u	d
wp 6	u	u	u	u	u	u	u	u	u	u	u	u	u	u
wp 7	u	u	u	u	u	u	d	d	d	d	d	d	d	d
wp 8	u	u	d	d	u	d	u	d	d	u	u	u	u	d
wp 9	u	d	u	d	d	d	d	u	d	u	d	u	d	u
wp 10	u	u	u	u	d	d	u	u	u	u	d	d	d	d
wp 11	u	u	u	u	u	u	u	u	u	u	u	u	d	d
wp 12	d	d	d	d	d	d	d	d	d	d	d	d	d	d
wp 13	d	d	d	d	u	u	d	d	d	d	u	u	u	u
wp 14	d	d	d	d	d	d	u	d	u	d	u	d	u	d
wp 15	d	d	d	d	d	d	d	u	d	u	d	u	d	u
wp 16	u	u	u	u	u	u	d	d	d	d	d	d	d	d
wp 17	u	u	u	u	u	u	d	d	d	d	d	d	d	d
wp 18	u	u	u	u	u	u	u	u	u	u	u	u	u	u
wp 19	u	u	u	u	d	d	u	u	u	u	d	d	d	d
wp 20	u	u	u	u	d	d	u	u	u	u	d	d	d	d
wp 21	u	u	u	u	d	d	u	u	u	u	d	d	d	d
wp 22	u	u	u	u	d	d	u	u	u	u	d	d	u	u
wp 23					Rev.01							Rev.01		
	u	u	u	u	o	o	u	u	u	u	o	o	o	o
					Rev.02							Rev.02		
					d	d					d	d	d	d
wp 24	u	u	u	u	d	d	u	u	u	u	d	d	d	d
wp 25	u	u	u	u	u	u	u	u	u	u	u	u	u	u
wp 26	u	u	u	u	u	u	u	u	u	u	u	u	u	u
wp 27	d	d	d	d	d	d	d	d	d	d	d	d	d	d
wp 28	u	u	u	u	u	u	d	d	d	d	d	d	d	d

u=UP d=DOWN o=OUT

MEMORY BOARD

ADM60



MEMORY MANAGEMENT UNIT (MMU)

REGION REGISTER ARRAYS

RARS: ARRAY[0..26,0..31] OF RAR_T

REGION REGISTER (RAR_T):
(type "rar_t" in vm.ins.pas)

```
     31             16 15             0
    +---------------+----------------+
+00 |VG------|PHYS ADDR OF SEGMENT MAP| .smap_phadd
    +---------------+----------------+
```

V - Region is valid (.valid)
G - Region is global (.global)

RARS is a dynamically-allocated per-asid table whose entries contain the region register values for that process. There are 32 hardware region registers, each covers 8mb of VA. The region registers are used by the address translation hardware.

SEGMENT MAPS

SMAPS: ARRAY[0..26,0..7680] OF SMAPE_T

SEGMENT MAP ENTRY (SMAPE):
(type "smape_t" in vm.ins.pas)

```
     31             16 15             0
    +---------------+----------------+
+00 |VAAAAA--| PHYS ADDR OF PAGE MAP  | .pmap_phadd
    +---------------+----------------+
```

V - Entry is valid (.valid)
A - Access rights as given in 16mb MSTE

The SMAP is a dynamically-allocated per-ASID table whose entries match one-for-one with a process's mstes. It is used by the address translation hardware and is organized in 256 segment units, so that one page of smapes represents one REGION (8mb of VA) of one address space.

See also the PAGE MAPS and MST sections of Chapter 1, AEGIS.

MONITOR TIMING (DN660)

Refer to the MONITOR TIMING section of CHAPTER 8, DN400,DN420,DN600, for this information.

RING/DISK

Refer to the RING/DISK section of CHAPTER 8, DN400,DN420,DN600, for this information.

SERIAL I/O INTERFACE

SIO page at [8400 | FFFB00]

DNx60 uses two SC681 chips providing four SIO lines, with control at FFFB010. The first line (make B at FFFB000) is used for the display keyboard.

Due to limitations of the SC2681 chip, when both SIO lines of a chip are being used, it is possible to have incompatible baud rates. One SIO line can't have a baud rate from Group A while the other SIO line is set from Group B:

Group A	Group B
50	75
7200	150
	2000
	19.2K

Lines 0 (the keyboard, which runs at 1200 baud) and 1 are paired on one chip, and lines 2 and 3 are on the other chip.

Refer to the SERIAL I/O INTERFACE section of CHAPTER 7, DN300,DN320, for more information.

CHAPTER 10

DN5xx

FOR INFORMATION SPECIFIC TO THE DN5xx-T NODES, REFER TO THE DN570-T/DN580-T/DSP500-T HARDWARE ARCHITECTURE HANDBOOK (009490)

ADDRESS SPACE (DN550)

physical		virtual
400	prom	400-3FFF (one-to-one)
4000	pft	FFB800-FFF7FF
8000	mmu	FFB400
8400	sios	FFB000
8800	timers	FFAC00
9800	ring	FF9C00
9C00	disk, tape, cal	FFA800
A400	pbu ctl	FF7C00
B000	fpu ctl	FF7000
B400	fpu cmd	FF7400
B800	fpu cs	FF7800
BC00	VME control	FF9400
E000	color_sup	FF6000
E400	color_user	FF6400
E800	color_wcs	FF6800
F000	displ_sup	FF9800
F400	displ_user	FFA000
F800	displ_wcs	FFA400
10000	iomap	FF5000-FF5FFF
14000	prom2	-
20000	displ_mem	FC0000-FDFFFF
40000	color_mem	FA0000-FBFFFF
70000	pbu i/o ref	FE0000-FE7FFF
80000	pbu 1st half	-
100000	mem: md data	E00000
100400	mem: traps	0
100800	mem	100800
380000	pbu 2nd half	-
700000	ptt	700000-7FFFFF

ADDRESS SPACE (DN560/570/580)

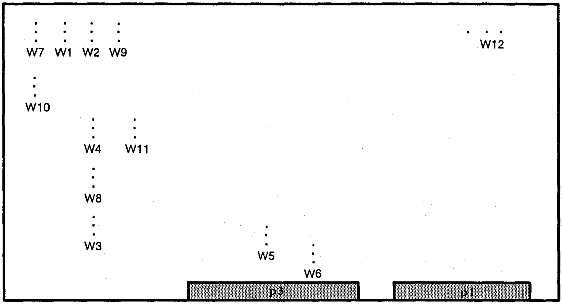
Refer to the ADDRESS SPACE (DN330) in Chapter 7.

CONFIGURATION

```
               +------+
               |      |
               | FPU  |
               |DN550 |
               +------+
+------+            ||
|      |  SBUS      ||
| CPU  |=============|
|      |----------+ ||
+------+    B-PORT| ||
   |  |           | ||
   |  |           | ||
   |  |           | ||
   |  | +------+ +------+
   |  | | EXP  | | VME  |     VME BUS
   |  +-| MEM  | |INT-  |=======|
   |    |      | | FACE |      ||
   |    +------+ +------+      ||
   |                           ||
   |                           ||
   |                           ||
   |     +------+  +------+    ||   +------+   +------+
   |     |      |  |COLOR |    ||   | VME  |   | NET  |
   +-----| SIO  |--| CONT |====||===|  TO  |---| XCVR |
         |      |  |      |    ||   | RING |   |      |
         +------+  +------+    ||   +------+   +------+
          ||||  DISPLAY||      ||              |
          ||||      BUS||      ||              |
 SIO 1  --+|||         ||      ||              +--- TO RING
 SIO 2  ---+||     +------+    ||   +------+
            ||     |COLOR |    ||   | VME  |
 VIDEO  ----+|     |ARRAY |    ||===|  TO  |----- TO DISK
             |     | (S)  |    ||   |D/T/C |----- TO TAPE
 KEYBOARD  --+     +------+    ||   +------+
                               ||
                               ||   +------+
                               ||   | VME  |
                               ||===|  TO  |
                               ||   |MULTI |
                               ||   +------+
                        (TERMINATOR)   ||
                                       ||   MULTIBUS
                                     <======================>
                                        |     |     |     |
                                  (SLOTS FOR APOLLO OR USER
                                        SUPPLIED DEVICES)
```

CPU BOARD JUMPERS (DN550)

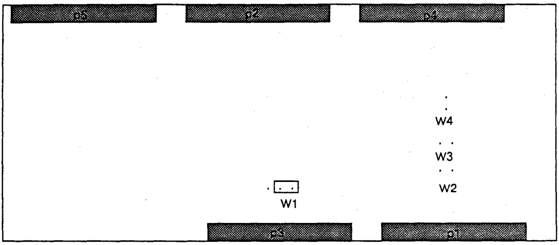
DN550 CPU Board (APNs 4141, 4145)



W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12
Up	Up	Up	Up	Up	Up	Down	Down	Down	Down	Up	Left

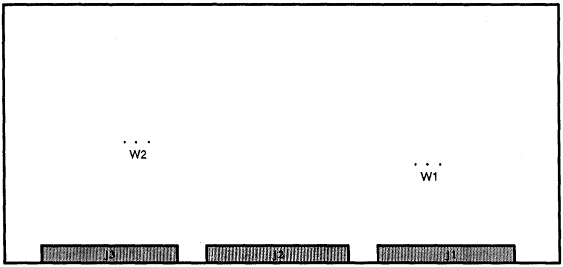
CPU BOARD JUMPERS (DN560/570/580)

DN560/570/580 CPU (APN 005373)



DISPLAY BOARD JUMPERS (DN550/560)

ARRAY BOARD

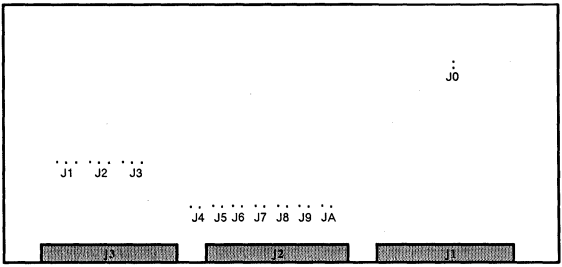


W1 (Timing Delay)
Right = Normal, Left = 3 nSec delay

W2 (Board Select)
Right = Board 1, Left = Board 2

CONTROL BOARD

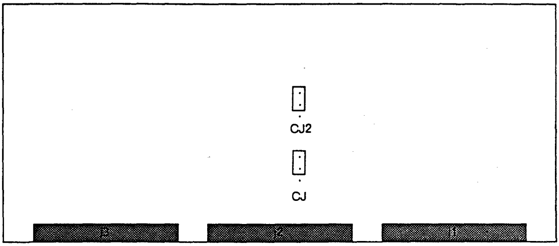
DN550 Color Controller (APN 3954)



J0	J1	J2	J3	J4	J5	J6	J7	J8	J9	JA
In	Left	Right	Right	Out	Out	In	Out	Out	Out	Out

DISPLAY BOARD JUMPERS (DN570)

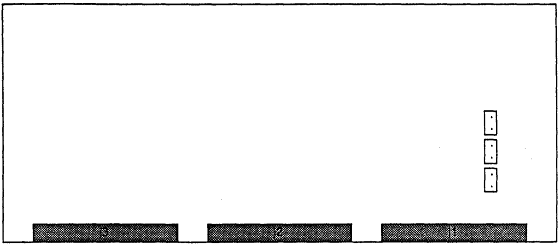
DN570 DISPLAY CONTROLLER (APN 005710)



DISPLAY BOARD JUMPERS (DN580)

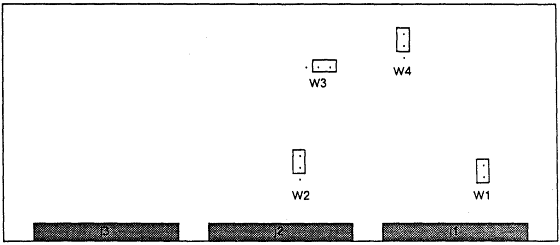
ARRAY BOARD

DN580 ARRAY BOARD (APN 005209)



CONTROL BOARD

DN580 COLOR CONTROLLER (APN 005213)



DN550/560 DISPLAY REGISTERS [F000 | 0FF9800]

```
disp1_sup  [ F000 | 0FF9800 ]   (also second color display)
disp1_user [ F400 | 0FFA000 ]
disp1_wcs  [ F800 | 0FFA400 ]

color_sup  [ E000 | 0FF6000 ]   (also second
color_user [ E400 | 0FF6400 ]      black & white display)
color_wcs  [ E800 | 0FF6800 ]
```

Display controller is a 600 with microcode changes.

DN570 DISPLAY CONTROL/STATUS REGISTERS [F000 | 3FF9800]

DN570 USER CONTROL REGISTER [F400 | 3FFA000]

```
 15                    8        5  4  3  2  1  0
+-----------------------+-----------------------+
|EH|----RESERVED-----|Z2|--|--|MR|TE|DS|DC|ST|DR|
+-----------------------+-----------------------+
  |  Zoom by 2  ------+        |  |  |  |  |  |
  |  0 => zoom factor = 1      |  |  |  |  |  |
  |  1 => zoom factor = 2      |  |  |  |  |  |
  +-- Error handshake          |  |  |  |  |  |
       Memory Reset ----------+  |  |  |  |  |
        0 => Reset                |  |  |  |  |
        1 => no effect            |  |  |  |  |
       Trap Enable  -------------+  |  |  |  |
        0 => Trap disabled           |  |  |  |
        1 => Trap enabled            |  |  |  |
       DP Start -------------------+  |  |  |
        0 => no effect                  |  |  |
        1 => Start the draw processor   |  |  |
       DP Continue -------------------+  |  |
        0 => no effect                     |  |
        1 => continue from last PC count   |  |
       DP Stop --------------------------+  |
        0 => Stop the draw processor          |
        1 => no effect                        |
       DP Reset ----------------------------+
        0 => Reset the draw processor
        1 => no effect
```

DN570 SYSTEM CONTROL REGISTER

```
 15       12 11        8  7  6     4  3  2  1  0
+-----------------------+-----------------------+
|   UWP     |CONTEXT ID |IE|--------|VE|RE|RU|CI|
+-----------------------+-----------------------+
  |  Zoom by 2  ------+        |  |  |  |  |  |
  |  0 => zoom factor = 1      |  |  |  |  |  |
  |  1 => zoom factor = 2      |  |  |  |  |  |
  +-- Error handshake          |  |  |  |  |  |
        Memory Reset ----------+  |  |  |  |  |
         0 => Reset               |  |  |  |  |
         1 => no effect           |  |  |  |  |
        Trap Enable  -------------+  |  |  |  |
         0 => Trap disabled          |  |  |  |
         1 => Trap enabled           |  |  |  |
        DP Start --------------------+  |  |  |
         0 => no effect                 |  |  |
         1 => Start the draw processor  |  |  |
        DP Continue --------------------+  |  |
         0 => no effect                    |  |
         1 => continue from last PC count  |  |
        DP Stop ---------------------------+  |
         0 => Stop the draw processor         |
         1 => no effect                       |
        DP Reset -----------------------------+
         0 => Reset the draw processor
         1 => no effect
```

DN580 DISPLAY CONTROL/STATUS REGISTERS [F000 | 3FF9800]

DN580 CONTROL REGISTER

BIT	NAME	FUNCTION
15-8	RESERVED	
7	INTERRUPT ENABLE	When set to a 1, interrupts are enabled. When set to a 0, interrupts are disabled.
6-4	RESERVED	
3	VIDEO ENABLE	When set to a 1, video is enabled. When set to a 0, video is disabled.
2	RESET	When set to a 0, the draw processor is reset. Setting this bit to a 1 has no effect.
1	RESERVED	
0	CONTEXT SWITCH	When set to a 1, context interrupts enabled. Setting it to a 0 enables normal interrupts.

DN580 STATUS REGISTER

BIT	NAME	FUNCTION
15	FIFO READY	When set to a 0, OK to write to FIFO. When set to a 1, FIFO not ready for input.

14	DISPLAY MEMORY ACCESS OK	When set to a 1, OK to do a display mem access. When set to a 0, not OK.
13	NEW INSTRUCTION PENDING	When set to a 1, instruction not in IR. When set to a 0, instruction is in IR.
12	PARAMETER PASSING HANDSHAKE	When set to a 1, OK to pass parameters. When set to a 0, not OK.
11	CONTEXT SWITCH HANDSHAKE	When set to a 0, request has been recognized. When set to a 1, request not recognized.
10	RESERVED	
9	HORIZONTAL BLANK	When set to a 0, horizontal blank occuring. When set to a 1, horizontal blank not occuring.
8	VERTICAL BLANK	When set to a 0, vertical blank occuring. When set to a 1, vertical blank not occuring.
7-0	RESERVED	

DISK/TAPE/CALENDAR CONTROLLER [9C00 | 0FFA800]

```
          15       8 7       0
         +----------+---------+
     +00 |---------IE| CMD REG |
         +----------+---------+
```

00-0F - Disk commands:

00 Nop (ignored by controller)
01 Seek
02 Read/write sector(s)
03 Format track
04 Read sector ID(s) (into IOPBs)
05 Restore (to cylinder 0)
06 Force ECC error(s)
07-0F Reserved (illegal disk commands)

10-1F - Tape commands:

10 Select tape drive 0
11 Select tape drive 1
12 Select tape drive 2
13 Select tape drive 3
14 Read status
15 Rewind to BOT
16 Erase entire tape
17 Initialize (retension) tape
18 Write block(s)
19 Write <count> filemarks
1A Read block(s)
1B Space forward <count> blocks
1C Space reverse <count> blocks (not implemented)
1D Space forward <count> filemarks
1E Space reverse <count> filemarks (not implemented)
1F Reset drive

20-2F - Controller commands:

20 Reset controller

```
  21  Execute controller diagnostics
  22  Calibrate memory timing loop
  23  Calibrate BX timing loop
  24  Enable force underrun/overrun (on next command)
  25  Disable  "             "
  26-2F Reserved (illegal commands)

30-FF - Reserved (illegal commands)

I = 1 => controller interrupting (reset to get next interrupt)
E = 1 => enable controller interrupt

      15                    0
     +-----------------------+
+02  |   CONTROLLER STATUS    |
     +-----------------------+
     15  8000 - controller busy
     14  4000 - disk busy
     13  2000 - tape busy
     12  1000 -
     11  0800 - disk op complete
     10  0400 - disk status valid
      9  0200 - tape op complete
      8  0100 - tape status valid
      7  0080 - DMA not at end of range
      6  0040 - DMA overrun/underrun
      5  0020 - memory parity during dma
      4  0010 -
      3  0008 - illegal controller command
      2  0004 - controller timeout
      1  0002 - controller diagnostic failed
      0  0001 -

      15       8 7       0
     +---------+---------+
+04  | DSK CNT |DSK STAT |   # sectors to transfer, disk status
     +---------+---------+
     01 Seek did not complete
     02 Write fault
     03 Unit not present
     04 Sector not found
     05 No index pulse
     06 Drive not ready
     07 No track 0
     08 Addr mark not found
     09 ECC error in ID field
     0A Correctable ECC error in data field
     0B Uncorrectable ECC error in data field
     0C Recovered ECC error (see dsk_ecc_cnt)
     0D Recovered overrun/underrun (see dsk_unr_cnt)
```

```
        15       8 7        0
       +---------+---------+
 +06   |TAPE CNT |TAPE DRV | # blocks to transfer, tape drive #
       +---------+---------+
  08   | HEADER WORD CNT   | (For tape, multiple blocks
       +-------------------+  must all have the same length.)
  0A   |  DATA WORD CNT    |
       +-------------------+
  0C   |  HEADER END CNT   | (disk or tape, will contain
       +-------------------+  ending counts for last two
  0E   |   DATA END CNT    |  blocks transferred)
       +---------+---------+
  10   |ID DRV 0 |---------| Drive 0 parameters
       +---------+---------+  (for IDs, see DISK PARAMETERS)
  12   | NUM CYLINDERS   0 |
       +---------+---------+
  14   | HEADS 0 |SECTORS 0|
       +---------+---------+
  16   |  PRE-COMP CYL 0   |
       +---------+---------+
  18   |ID DRV 1 |---------| Drive 1 parameters
       +---------+---------+
  1A   | NUM CYLINDERS   1 |
       +---------+---------+
  1C   | HEADS 1 |SECTORS 1|
       +---------+---------+
  1E   |  PRE-COMP CYL 1   |
       +-------------------+

        15       8 7        0
       +---------+---------+
 +20   | CTLR ID |---------|    controller ID
       +---------+---------+
 +22   | DISK IOPB OFFSET  |
       +---------+---------+
 +24   | ECC CNT | UNR CNT |    disk retry counters
       +---------+---------+
 +26   |  DISK ECC OFFSET  |    offset to correction
       +-------------------+
 +28   |   DISK ECC MASK   |    mask
       +-------------------+
         15      8 7       0
        +---------+---------+
+2A     |TAPE ST 0|TAPE ST 1|    status from drive
        +---------+---------+
           7  80  Status byte 0 bits
           6  40  Cartridge not in place
           5  20  Unselected drive
           4  10  Write protected cartridge
           3  08  End of media (EOM)
           2  04  Unrecoverable data error
           1  02  Bad block not located
           0  01  Filemark detected

                     7  80  Status byte 1 bits
                     6  40  Illegal command
                     5  20  No data detected
                     4  10  Marginal block detected
                     3  08  Beginning of media (BOM)
                     2  04  Reserved for bus parity error
                     1  02  Reserved for end of recorded media
                     0  01  Power on/reset occurred
```

TAPE ST 0	TAPE ST 1	STATUS SUMMARY
11110001	00000000	Drive not ready (controller generated)
110X0000	00000000	No cartridge
11110000	00000000	No drive
10010000	X000X000	Write protected
10001000	00000000	End of media
100X0100	10001000	Read or write abort
100X0100	00000000	Read error, bad blk transfer
100X0110	00000000	Read error, filler blk transfer
100X0110	10100000	Read error, no data
100X1110	10100000	Read error, no data and EOM
100X0110	101X1XX0	Read error, no data and BOM
100X0001	00000000	Filemark read
XXXX0000	1100X000	Illegal command
XXXX0000	1000X001	Power on/reset
100X0001	00010000	Marginal block detected

```
         15                0
        +-------------------+
+2C     |  TAPE ERROR CNTR  |    tape error counter
        +-------------------+
+2E     | TAPE UNDERRUN CTR |    tape underrun counter
        +-------------------+
+30     |CTLR TIMEOUT STATUS|    timeout status
        +---------+---------+
+32     |LAST CMD |        P|    most recent command executed
        +---------+---------+    P => precomp enabled
        .                   .
        .                   .
        .                   .
```

```
     +-------------------+
+34  | DIAG ERROR LOC    |    location of diagnostic failure
     +-------------------+
+36  | DIAG ERROR SB     |    what data should have been
     +-------------------+
+38  | DIAG ERROR IS     |    what data was
     +-------------------+

      31                  16 15      8 7         0
     +---------------------+---------+-----------+
+80  |      CYLINDER0      | HEAD0   | SECTOR0   |
     +---------------------+---------+-----------+
     |              HEADER ADDRESS 0             |
     +---------------------+---------------------+
     |      DATA PPN 0     |----------------CIWDD|
     +---------------------+---------------------+
                         - - -
     +---------------------+---------+-----------+
1F4  |      CYLINDER31     | HEAD31  | SECTOR31  |
     +---------------------+---------+-----------+
     |             HEADER ADDRESS 31             |
     +---------------------+---------------------+
     |      DATA PPN 31    |----------------CIWDD|
     +---------------------+---------------------+
200                         C - continue on error
                            I - interrupt when done
                            W - write (else read)
                            DD - unit number (00 or 01)
```

CALENDAR CONTROLS [9E00 | 0FFAA00]

```
        15              0
       +--------+---------+
9E00   |CAL_CTL |CAL_RD/WR|    Calender control and
       +--------+---------+           read/write registers
```

DISKTAPE VME ADDRESS MODIFIERS [9F00 | 0FFAB00]

```
        15                     8                       0
       +-----------------------+-----------------------+
9F00   |LW|--|M5|M4|M3|M2|M1|M0|--|--|--|--|--|M2|M1|M0|
       +-----------------------+-----------------------+
              \       /\       /                \      /
                 ||        111  Supervisor sequential (not imp.)
                 ||        110  Supervisor program
                 ||        101  Supervisor data
                 ||        010  User program
                 ||        001  User data
                 ||        000  State after reset
                 ||
                 111 Standard (24 bits) addressing
                 101 Short    (16 bits)
                 001 Extended (32 bits)

           LW = 1 => use 32-bit transfers
```

FAULT FRAME

Refer to the FAULT FRAME section of CHAPTER 7, DN300,DN320,DN330 for this information. For DN550 information, refer to the DN300/320 section; for DN560/570/580 information, refer to the DN330 sections.

FAULT TYPES

Refer to the FAULT TYPES section of CHAPTER 7, DN300,DN320,DN330 for this information.

FAULT VECTORS

Exception vector at [100400 | 0]

Vector	Address	Assignment		
00	000	Reset: Initial SSP		
	004	Reset: Initial PC		
02	008	Bus Error		
03	00C	Address Error		
04	010	Illegal Instruction		
05	014	Zero Divide		
06	018	CHK Instruction		
07	01C	TRAPV Instruction		
08	020	Privilege Violation		
09	024	Trace		
0A	028	Unimplemented instruction		
0B	02C	Unimplemented instruction		
0C	030	(Unassigned, reserved)		
0D	034	Coprocessor protocol violation		
0E	038	Invalid Stack Format		
0F-17	03C	(Unassigned, reserved)		
18	060	Spurious Interrupt		
19-1F	064	Level 1-7 Auto-Vector	interrupt level	
19	064	SIO (rcv and xmit)	1	
1A	068	Display keyboard	2	
1B	06C	PEB	3	
1C	070	VME	4	
1D	074	VME bus error	5	
1E	078	Timers 1,2,3	6	
1F	07C	Parity error	7	
20-2F	080	TRAP Instruction Vectors		
30	0C0	FP Branch or Set on Unordered cond (FPBSUN)		
31	0C4	FP Inexact Result (FPINEX)		
32	0C8	FP Divide by Zero (FPDIVZ)		
33	0CC	FP Underflow (FPUNFL)		
34	0D0	FP Operand Error (FPOPER)		
35	0D4	FP Overflow (FLOVFL)		
36	0D8	FP Signalling NAN (FPSNAN)		
37-3F	0DC	(Unassigned, reserved)	-	-
41	104	getc		
42	108	putc		
43	10C	init_dsk		
44	110	read_dsk		
45	114	reload_font		
46	118	pollc		
47	11C	quiet_ret		
48	120	write_dsk		
49	124	log_error		
4A	128	crash		
4B	12C	led_update		
4C-5F	130	-		
60	180	VME pseudo-vectors	0	(unused)
61	184	ring	1	

62	188	disktape	2
63	18C	display	3
64	190	-	4
65	194	pbu	5
66	198	-	6
67	19C	-	7
68-6F	1A0	-	
70-FF	1C0	-	

FLOATING-POINT FORMAT

Refer to the FLOATING-POINT FORMAT section of CHAPTER 7, DN300,DN320,DN330 for this information.

FLOATING-POINT REGISTERS (DN560/570/580)

Refer to the FLOATING-POINT REGISTERS section of CHAPTER 7, DN300,DN320,DN330 for this information.

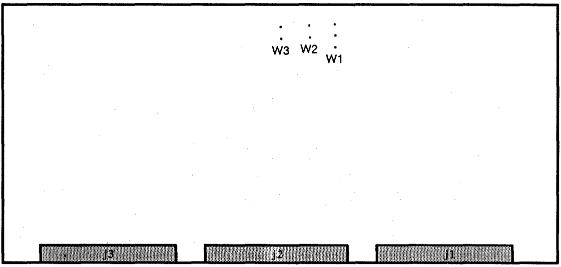
FPU (DN550)

Refer to the FPU section of CHAPTER 7, DN300,DN320,DN330 for this information.

MEMORY CONTROL/STATUS REGISTERS (MCSR)

Refer to the MEMORY CONTROL/STATUS REGISTERS (MCSR) section of CHAPTER 7, DN300,DN320,DN330 for this information. For DN550 information, refer to the DN300/320 section; for DN560/570/580 information, refer to the DN330 sections.

MEMORY BOARD JUMPERS (DN550)



W1

Up = CPU/Mem=2MB,
Down = CPU/Mem=0.5MB

W2	W3
Out	Out = No memory expansion
Out	In = Fully populated (APN 3354)
In	Out = 1/2 populated (APN 4151)
In	In = Not allowed

MEMORY MANAGEMENT UNIT (MMU)

Refer to the MEMORY MANAGEMENT UNIT section of CHAPTER 7, DN300,DN320,DN330 for this information. For DN550 information, refer to the DN300/320 section; for DN560/570/580 information, refer to the DN330 sections.

MONITOR TIMING (DN550/560)

19-inch Hitcahi and Ikegami

Axis	Item		Duration	Frequency
HORIZONTAL	1H	HORIZONTAL FREQUENCY	29.53 uSec	33.855 KHz
	H-FP	HORIZONTAL FRONT PORCH	0.75 uSec	n/a
	H-SYNC	HORIZONTAL SYNC	2.75 uSec	n/a
	H-BP	HORIZONTAL BACK PORCH	3.5 uSec	n/a
	H-BL	HORIZONTAL BLANKING	7.0 uSec	n/a
	H-DISP	HORIZONTAL DISPLAY AREA	22.53 uSec	n/a
VERTICAL	1V	VERTICAL FREQUENCY	2.75 mSec	78.4 Hz
	V-FP	VERTICAL FRONT PORCH	8.6 uSec	n/a
	V-SYNC	VERTICAL SYNC	88.6 uSec	n/a
	V-BP	VERTICAL BACK PORCH	708.0 uSec	n/a
	V-BL	VERTICAL BLANKING	885.0 uSec	n/a
	V-DISP	VERTICAL DISPLAY AREA	11.865 mSec	n/a

MONITOR TIMING (DN570)

15-inch and 19-inch Panasonic

Axis	Item		Duration	Frequency
HORIZONTAL	1H	HORIZONTAL FREQUENCY	19.79 uSec	50.519 KHz
	H-FP	HORIZONTAL FRONT PORCH	0.942 uSec	n/a
	H-SYNC	HORIZONTAL SYNC	1.88 uSec	n/a
	H-BP	HORIZONTAL BACK PORCH	1.88 uSec	n/a
	H-BL	HORIZONTAL BLANKING	4.71 uSec	n/a
	H-DISP	HORIZONTAL DISPLAY AREA	15.08 uSec	n/a
VERTICAL	1V	VERTICAL FREQUENCY	16.67 mSec	60.0 Hz
	V-FP	VERTICAL FRONT PORCH	79.18 uSec	n/a
	V-SYNC	VERTICAL SYNC	79.18 uSec	n/a
	V-BP	VERTICAL BACK PORCH	673.0 uSec	n/a
	V-BL	VERTICAL BLANKING	831.0 uSec	n/a
	V-DISP	VERTICAL DISPLAY AREA	15.839 mSec	n/a

MONITOR TIMING (DN580)

19-inch Sony

Axis	Item		Duration	Frequency
HORIZONTAL	1H	HORIZONTAL FREQUENCY	15.78 uSec	63.357 KHz
	H-FP	HORIZONTAL FRONT PORCH	0.37 uSec	n/a
	H-SYNC	HORIZONTAL SYNC	1.71 uSec	n/a
	H-BP	HORIZONTAL BACK PORCH	1.79 uSec	n/a
	H-BL	HORIZONTAL BLANKING	3.87 uSec	n/a
	H-DISP	HORIZONTAL DISPLAY AREA	11.91 uSec	n/a
VERTICAL	1V	VERTICAL FREQUENCY	16.67 mSec	59.99 Hz
	V-FP	VERTICAL FRONT PORCH	47.35 uSec	n/a
	V-SYNC	VERTICAL SYNC	47.35 uSec	n/a
	V-BP	VERTICAL BACK PORCH	410.37uSec	n/a
	V-BL	VERTICAL BLANKING	505.1 uSec	n/a
	V-DISP	VERTICAL DISPLAY AREA	16.162 mSec	n/a

```
      |<-x-BL-|<--------- x-DISP ---------->|
 -----+       +-----------------------------+
 /////|       |/////////////////////////////|
 /////|       |/////////////////////////////|
 -----+-+  +--+-----------------------------+-+  +--
      | |  |  |                               |  |
      | +--+  |                               +--+
      | |  |  |                               |
x-FP->|-|--|--|<-x-BP                         |
        | ^                                   |
        | +--x-SYNC                           |
        |<---------------1x------------------>|

  x = V or H
```

MULTIBUS REGISTERS

MULTIBUS CONTROL STATUS REGISTER [A400 | FF7C00]

```
 15  14  13  12  11  10   9   8
+---+---+---+---+---+---+---+---+- - -
|   |   |   |   | r | r |   |   |
+---+---+---+---+---+---+---+---+- - -
  |   |   |   |  \-----/  |   |
  |   |   |   | reserved  |   +- Loopback data (see below)
  |   |   |   |           +----- XACK on time-out
  |   |   |   +----------------- MB_BPRO (diag use only)
  |   |   +--------------------- Enable early release (should be 0)
  |   +--------------- Time-out (when MULTIBUS master talking
  |                    to another MULTIBUS device; write 0
  |                    to clear)
  +------------------- Watchdog Timer

                            7   6   5   4   3   2   1   0
                     - - -+---+---+---+---+---+---+---+---+
                          |   |   |   |   |   |   |   |   |
                     - - -+---+---+---+---+---+---+---+---+
                            |   |   |   |   |   |   |   |
Lookback enable ------------+   |   |   |   |   |   |   |
Xchange sim (diag only)---------+   |   |   |   |   |   |
Upper MULTIBUS Enable --------------+   |   |   |   |   |
Lower MULTIBUS Enable ------------------+   |   |   |   |
Force int enable (diag only) ---------------+   |   |   |
MULTIBUS Multiport Memory Lock Enable-----------+   |   |
Lock MULTIBUS Arbiter (asserted when 0) ------------+   |
MULTIBUS INIT (asserted when 0) ------------------------+
              (set to one to drop reset)
```

MULTIBUS INTERRUPT CONTROL STATUS REGISTER [A402 | FF7C02]

```
 15                 8 7                    0
+-------------------+--------------------+
|0000000000000000000| Mask for lines 7-0 |
+-------------------+--------------------+
```

(Interrupt level now in VME interrupt/ID register 5 following interr
acknowlege cycle.)

MULTIBUS ADDRESS MODIFIER REGISTER [A404 | FF7C04]

```
 15     TRANSMITTED     8        RECEIVED       0
+------------------------+-----------------------+
|--|--|M5|M4|M3|M2|M1|M0|--|--|--|--|--|M2|M1|M0|
+------------------------+-----------------------+
       \      /\       /                \      /
          ||        111  Supervisor sequential (not imp.)
          ||        110  Supervisor program
          ||        101  Supervisor data
          ||        010  User program
          ||        001  User data
          ||        000  State after reset
          ||
          111 Standard (24 bits) addressing
          101 Short    (16 bits)
          001 Extended (32 bits)
```

MULTIBUS MAP [10000 | FF5000]

```
 31                  16 15                    0
+---------------------+--+--+----------------+
|------------------210|15|--|   PPN (13:0)   |
+---------------------+--+--+----------------+
                   |||  |
                   |||  1 => Page is mapped
                   |||
                   |\/
                   | Swap Mode
                   |  (See DSP80 spec for more info)
                   |
                   |
                   0 => Page is read only (was 1)
```

LOOPBACK DATA

(No longer exists.)

PAGE FRAME TABLE ENTRY (PFTE)

Refer to the PAGE FRAME TABLE ENTRY (PFTE) section of CHAPTER 7, DN300,DN320, for this information.

PAGE TRANSLATION TABLE ENTRY (PFTE)

Refer to the PAGE TRANSLATION TABLE ENTRY (PFTE) section of CHAPTER 7, DN300,DN320, for this information.

PROM ENTRY POINTS

```
100:    dc.w   5,0            5 => stingray
104:    ac     getc           returns character in D1
108     ac     putc           prints character in D1
10C     ac     init_dsk       initialize disk
110     ac     read_dsk       read a record from disk
114     ac     reload_font    reload font
118     ac     pollc          returns character in D1,
                              else -1 in d1.w
11C     ac     quiet_ret      quiet return to PROM
120     ac     write_disk
124     ac     log_error      on/off
128     ac     crash
12C     ac     led_update
```

RING REGISTERS

```
        RING page at [ 9800 | 0FF9C00 ]
             WRITE FUNCTION                        READ FUNCTION

          15                     0          15                     0
         +-----------------------+         +-----------------------+
     +00 |    TRANSMIT COMMAND   |     +00 |   TRANSMIT STATUS     |
         +-----------------------+         +-----------------------+
     +02 |    RECEIVE COMMAND    |     +02 |    RECEIVE STATUS     |
         +-----------+-----------+         +-----------+-----------+
     +04 |    TMASK  |  unused   |     +04 |   TMASK   |  UNUSED   |
         +-----------+-----------+         +-----------+-----------+
     +06 |     DIAG COMMAND      |     +06 |     DIAG STATUS       |
         +-----------------------+         +-----------------------+
     +08 |         RING          |     +08 |         RING          |
         +- - - - - - - - - - - -+         +- - - - - - - - - - - -+
         |          ID           |         |        ID (*)         |
         +-----------------------+         +-----------------------+
                                       +0C |        UNUSED         |
                                           +-----------------------+
    (*) gate_array only. must be       +0E |        UNUSED         |
        written after a reset.             +-----------+-----------+
                                       +10 |    ID3    |  UNUSED   |
                                           +-----------+-----------+
                                       +12 |    ID2    |  UNUSED   |
                                           +-----------+-----------+
                                       +14 |    ID1    |  UNUSED   |
                                           +-----------+-----------+
                                       +16 |    ID0    |  UNUSED   |
                                           +-----------+-----------+
```

RING MAP/WORD COUNT REGISTERS [9820 | 0FF9C20]

```
       31                        15                       0
      +---------------------------------------------------+
9820  |             RECEIVE HEADER ADDRESS      XXXXXXXXXX|  (*)
      +---------------------------------------------------+
9824  |              RECEIVE DATA ADDRESS       XXXXXXXXXX|
      +---------------------------------------------------+
9828  |             TRANSMIT HEADER ADDRESS     XXXXXXXXXX|
      +---------------------------------------------------+
982C  |              TRANSMIT DATA ADDRESS      XXXXXXXXXX|
      +-------------------------+-------------------------+
9830  |  RECEIVE HDR WORD COUNT |  RECEIVE HDR END COUNT  |  (**)
      +-------------------------+-------------------------+
9834  |  RECEIVE DAT WORD COUNT |  RECEIVE DAT END COUNT  |
      +-------------------------+-------------------------+
9838  |(***)XMIT HDR WORD COUNT | TRANSMIT HDR END COUNT  |
      +-------------------------+-------------------------+
983C  | TRANSMIT DAT WORD COUNT | TRANSMIT DAT END COUNT  |
      +-------------------------+-------------------------+

     (*)  low order 10 bits are ignored
    (**)  bits 15:10 of counts are ignored; read as junk
   (***)  bit 11 of transmit header count = 1 => data length = 0
```

RING ADDRESS MODIFIER REGISTERS [9840 | 0FF9C40]

```
 15                    8                     0
+--+--+--+--+--+--+--+----------------------+
|--|--|M5|M4|M3|M2|M1|M0|----------------------|
+--+--+--+--+--+--+--+----------------------+
      \       /\      /
         ||        111  Supervisor sequential (not imp.)
         ||        110  Supervisor program
         ||        101  Supervisor data
         ||        010  User program
         ||        001  User data
         ||        000  State after reset
         ||
         111 Standard (24 bits) addressing
         101 Short    (16 bits)
         001 Extended (32 bits)

9840   Receive modifier      (R/W)
9848   Tranmit modifier      (R/W)
9850   Slave (PIO) modifier  (W/O)
           (bit 3 = 1 => longword transfers)
9860   LSB address counter   (R/O)     (bits 9:0 of word)
```

Refer to the RING REGISTERS section of CHAPTER 7, DN300,DN320 for command and status definitions.

SERIAL I/O INTERFACE

Refer to the SERIAL I/O INTERFACE section of CHAPTER 7, DN300,DN320, for this information.

VME REGISTERS [BC00 | FF9400]

VME INTERRUPT STATUS/ID REGISTERS [BC02 | FF9402]

```
      15                0     (read to clear vme int)
     +---------+---------+
BC00 |---------|LEV0 ID |   unavailable
     +---------+---------+
BC02 |---------|LEV1 ID |   ring
     +---------+---------+
BC04 |---------|LEV2 ID |   disktape
     +---------+---------+
BC06 |---------|LEV3 ID |   display (instruction queue done)
     +---------+---------+
BC08 |---------|LEV4 ID |   display (vertical retrace)
     +---------+---------+
BC0A |---------|1011nnnn|   pbu  nnnn=level, 1111=>device timeout
     +---------+---------+
BC0C |---------|LEV6 ID |   customer vme (?)
     +---------+---------+
BC0E |---------|LEV7 ID |   unused
     +---------+---------+
```

VME INTERRUPT LEVEL REGISTER [BC10 | FF9410]

```
      15              8 7               0
     +----------------+----------------+
BC10 |----------------|L7L6L5L4L3L2L1--|
     +----------------+----------------+

      0-bit => that level is interrupting
```

VME ADDRESS MODIFIER REGISTER [BC20 | FF9420]

```
      15 TRANSMIT ADM  7 RECEIVE ADM  0
     +----------------+-----------------+
BC20 |----A5A4A3A2A1A0|-----------A2A1A0|
     +----------------+-----------------+
          \    /\    /           \    /
            ||    ||               111  Supervisor
            ||    ||                    sequential (not imp)
            ||    ||               110  Supervisor program
            ||    ++----------->   101  Supervisor data
            ||                     010  User program
           111 Standard (24 bits)  001  User data
           101 Short    (16 bits)  000  State after reset
           001 Extended (32 bits)
```

VME MEMORY SIZE REGISTER [BC30 | FF9430]

```
      15              8 7                0
     +----------------+-----------------+
BC30 |----------------|---------PWS2S1S0|
     +----------------+-----------------+
                               / \    /
                              /    010  0.5 Mb
                             /     011  1.0 Mb
                            /      100  1.5 Mb
                           /       101  2.0 Mb
                          /        110  2.5 Mb
                         /         111  3.0 Mb
    PW = 1 => enable power off     000  State after reset
```

VME DIAGNOSTIC CONTROL REGISTERS [BC40 | FF9440]

```
      15                                     8
     +---------------------------------------+ - -
BC40 | -- | -- | -- | -- | -- | -- | -- | -- |
     +---------------------------------------+ - -

           7                                     0
      - - +----------------------------------------+
          |LOOP| ELA| --  |FA23|FA22|FA21|FA20|FA19|
      - - +----------------------------------------+

      15                                     8
     +---------------------------------------+ - -
BC50 | -- | -- | -- |EFBR|FBR3|FBR2|FBR1|FBR0|
     +---------------------------------------+ - -

           7                                     0
      - - +---------------------------------------+
          |FIR7|FIR6|FIR5|FIR4|FIR3|FIR2|FIR1|EFIR|
      - - +---------------------------------------+
```

VME ERROR REGISTERS [BC60 | FF9460]

```
        15                                          8
       +--------------------------------------------+ - -
BC60   |BTO*|PAR*| AM5| AM4| AM3| AM2| AM1| AM0|      add. mod
       +--------------------------------------------+ - -

              7                                          0
        - - +--------------------------------------------+
             |A23------------------------------- A16|
        - - +--------------------------------------------+

        15                                          0
       +--------------------------------------------+
BC70   |A15 ----------------------------A01 AS*|
       +--------------------------------------------+

        15                                                   8
       +--------------------------------------------------+ - -
BC80   |DS1|DS0|WRT*|LWD*|DTACK*|BBSY*|SYSRS*|ACFAL*|
       +--------------------------------------------------+ - -

              7                                          0
        - - +--------------------------------------------+
             | -- | -- | -- | -- | -- | -- | -- | -- |
        - - +--------------------------------------------+

        15                                          8
       +--------------------+-----------------------+ - -
BC90   |BR3*|BR2*|BR1*|BR0*|BG3*|BG2*|BG1*|BG0*|
       +--------------------+-----------------------+ - -

              7                                            0
        - - +----------------------------------------------+
             |IACK*|IR7*|IR6*|IR5*|IR4*|IR3*|IR2*|IR1*|
        - - +----------------------------------------------+
```

CHAPTER 11

DSP80,DSP80A,DSP90

ADDRESS SPACE (DSP80/80A)

physical		virtual
100400	traps	0
400	PROM	400->3FFF (one-to-one)
14000	PROM2	-
100800	phys mem	100800->FFFFF
700000	ptt	700000->7FFFFF
380000	pbu 2nd half	-
100000	md stk,data	E00000
80000	pbu 1st half	-
70000	pbu i/o ref	FE0000->FE7FFF
10000	iomap	FF5000->FF5FFF
A400	pbu ctl	FF7C00
A800	lpr	FF8000
9800	ring 2	FF9C00
9000	DMA ctl	FFA000
A000	calendar	FFA400
8800	timers	FFAC00
8400	sios	FFB000
8000	mmu	FFB400
4000	pft	FFB800->FFC7FF

ADDRESS SPACE (DSP90)

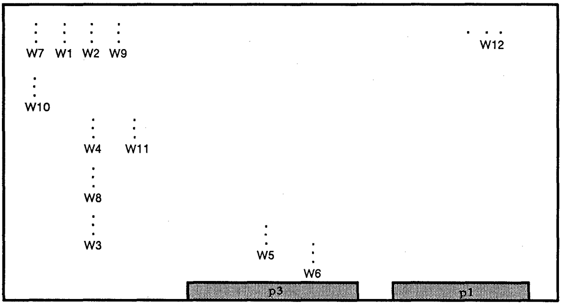
Refer to the ADDRESS SPACE (DN330) section of Chapter 7 for this information.

CONFIGURATION

```
+------+ +------+
|     |  |      |                        SBUS
| CPU |==| MMU  |================================================|
|     |  |      |   |         |         |         |        |    ||
+------+ +------+   |         |         |         |        |    ||
                +-------+ +-------+ +------+ +-------+ +------+ ||
                | MAIN  | | BOOT  | | SIO  | | REAL  | | MMU  | ||
                | MEM   | | PROM  | |      | | TIME  | | CTL  | ||
                |       | |       | |      | | CLK   | |      | ||
                +-------+ +-------+ +------+ +-------+ +------+ ||
                    |                 | |                       ||
                    |                 | +---SIO 1               ||
                    |                 +-----SIO 2               ||
                    |                                           ||
                    |                                           ||
                    |       =====================================|
                    |           |        |         |
                    |           |        |         |
                    |       +-------+ +------+ +-------+
                    |       | PAR   | | DMA  | | RING  |
                +-------+   | LPT   | | CTL  | | CTL   |
                | M.B.  |   | CTL   | |      | |       |
                | CTL/  |   +-------+ +------+ +-------+
                | MAP   |       |                  |
                +-------+    line              network
                    |       printer
                    |                 MULTIBUS
         <=======================================================>
                |        |         |         |        |
                |        |         |         |        |
             ( Slots for Apollo and user-supplied devices)
```

CPU BOARD JUMPERS (DSP80/80A)

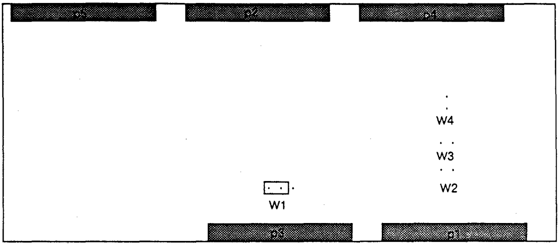
DSP80/A CPU Board (APNs 2422, 4834)



W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12
Up	Up	Up	Up	Up	Up	Down	Down	Down	Down	Up	Right

CPU BOARD JUMPERS (DSP90)

DN330, DSP90 CPU (APN 005373)



DMA CONTROLLER

DMAC page at [9000 | 0FFFA00]

The DMA controller is Motorola M68450.

9000-907F	- ring receive header
9080-90FF	- ring receive data
9100-917F	- ring transmit
9180-91FF	- Unused

Refer to the DMA CONTROLLER section of CHAPTER 7, DN300,DN320,DN330 for more information.

FAULT FRAME

Refer to the FAULT FRAME section of CHAPTER 7, DN300,DN320,DN330 for this information. Use the DN300/320 sections for the DSP80/80A, and the DN330 sections for the DSP90.

FAULT TYPES

Refer to the FAULT TYPES section of CHAPTER 7, DN300,DN320,DN330 for this information. Use the DN300/320 sections for the DSP80/80A, and the DN330 sections for the DSP90.

FAULT VECTORS

Exception vector at [100400 | 0]

Vector	Address	Assignment	
00	000	Reset: Initial SSP	
	004	Reset: Initial PC	
02	008	Bus Error	
03	00C	Address Error	
04	010	Illegal Instruction	
05	014	Zero Divide	
06	018	CHK Instruction	
07	01C	TRAPV Instruction	
08	020	Privilege Violation	
09	024	Trace	
0A	028	Unimplemented instruction	
0B	02C	Unimplemented instruction	
0C	030	(Unassigned, reserved)	
0D	034	Coprocessor	
0E	038	Invalid Stack Format	
0F-17	03C	(Unassigned, reserved)	
18	060	Spurious Interrupt	
19-1F	064	Level 1-7 Auto-Vector	interrupt level
19	064	SIO (rcv and xmit)	1
1A	068	(keyboard input)	2
1B	06C	Ring	3
1C	070	PBU	4
1D	074	Line printer	5
1E	078	Timers 1,2,3	6
1F	07C	Parity error	7
20-2F	080	TRAP Instruction Vectors	
30	0C0	FP Branch or Set on Unordered cond (FPBSUN)	
31	0C4	FP Inexact Result (FPINEX)	
32	0C8	FP Divide by Zero (FPDIVZ)	
33	0CC	FP Underflow (FPUNFL)	
34	0D0	FP Operand Error (FPOPER)	
35	0D4	FP Overflow (FLOVFL)	
36	0D8	FP Signalling NAN (FPSNAN)	
37-3F	0DC	(Unassigned, reserved)	- -

FLOATING-POINT REGISTERS (DSP90)

Refer to the FLOATING-POINT REGISTERS section of CHAPTER 7, DN300,DN320,DN330 for this information.

MEMORY CONTROL/STATUS REGISTERS (MCSR)

Refer to the MEMORY CONTROL/STATUS REGISTERS (MCSR) section of CHAPTER 7, DN300,DN320,DN330 for this information. Use the DN300/320 sections for the DSP80/80A, and the DN330 sections for the DSP90.

MEMORY MANAGEMENT UNIT (MMU)

Refer to the MEMORY MANAGEMENT UNIT section of CHAPTER 7, DN300,DN320,DN330 for this information. Use the DN300/320 sections for the DSP80/80A, and the DN330 sections for the DSP90.

MULTIBUS REGISTERS

MULTIBUS CONTROL STATUS REGISTER [A400 | FF7C00]

```
 15  14  13  12  11  10   9   8
+---+-----------+---+---+---+---+- - -
|   |           | r | r |   |   |
+---+-----------+---+---+---+---+- - -
  |  \----v----/ \-----/  |   |
  |       |     reserved  |   +- Loopback data (see below)
  |       |               +----- XACK on time-out
  |       +--------------------- Current bus master:
  |                                0    - DSP80
  |                                1-5  - contoller in slot 1-5
  |                                         (bottom slot is 1)
  +------------------- Watchdog Timer
 |______read only____|
                            7   6   5   4   3   2   1   0
                     - - -+---+---+---+---+---+---+---+---+
                          |   |   |   |   |   |   |   |   |
                     - - -+---+---+---+---+---+---+---+---+
                            |   |   |   |   |   |   |   |
Lookback enable ------------+   |   |   |   |   |   |   |
Xchange sim (diag only)---------+   |   |   |   |   |   |
Upper MULTIBUS Enable --------------+   |   |   |   |   |
Lower MULTIBUS Enable ------------------+   |   |   |   |
Lock enable for memory ---------------------+   |   |   |
MULTIBUS Multiport Memory Lock Enable-----------+   |   |
Lock MULTIBUS Arbiter (asserted when 0) ------------+   |
MULTIBUS INIT (asserted when 0) ------------------------+
              (set to one to drop reset)
```

MULTIBUS INTERRUPT CONTROL STATUS REGISTER [A402 | FF7C02]

```
 15                8 7                   0
+-------------------+--------------------+
|  Int lines 7-0    | Mask for lines 7-0 |
+-------------------+--------------------+
```

MULTIBUS MAP [10000 | FF5000]

```
 15         11            0 15              0
+--+--+--+--+--------------+-----------------+
|15|14|13|12| MULTIBUS PPN |      unused     |
+--+--+--+--+--------------+-----------------+
  |  |  \--/
  |  |    |
  |  |    Swap Mode:
  |  |     See DSP80 spec for more info
  |  |
  |  |
  |  1 => Page is read-only
  |
  1 => Page mapped
```

LOOPBACK DATA

```
 47  46  45  44  43  42  41  40  39  38  37  36  35  34  33  32
+-------------------------------+-------------------------------+
|   |   |   |   |   |   |   |   | r | r | r | r |   |   |   |   |
+-------------------------------+-------------------------------+
  \---v---/   |   |   |   |   |   \-----v-----/   \-----v-----/
   Master     |   |   |   |   |         |               |
              |   |   |   |   |         |               |
INIT (Reset) -+   |   |   |   |         |               |
BHEN -------------+   |   |   |         |               |
Memory Cycle ---------+   |   |         |               |
Read Cycle ---------------+   |         |               |
XACK -------------------------+         |               |
Reserved for MULTIBUS Address(23:20) ---+               |
MULTIBUS Address (20:16) -------------------------------+

 31  30  29  28  27  26  25  24  23  22  21  20  19  18  17  16
+-------------------------------+-------------------------------+
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
+-------------------------------+-------------------------------+
                      MULTIBUS Address (15:0)

 15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+-------------------------------+-------------------------------+
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
+-------------------------------+-------------------------------+
                      MULTIBUS Data (15:0)
```

PAGE FRAME TABLE ENTRY (PFTE)

Refer to the PAGE FRAME TABLE ENTRY (PFTE) section of CHAPTER 7, DN300,DN320,DN330 for this information.

PAGE TRANSLATION TABLE ENTRY (PFTE)

Refer to the PAGE TRANSLATION TABLE ENTRY (PFTE) section of CHAPTER 7, DN300,DN320,330 for this information.

PROM ENTRY POINTS

```
100:    dc.w   3,0            3 => DSP80, no aux info
104:    ac     getc           returns character in D1
108     ac     putc           prints character in D1
10C     ac     init_dsk       initialize disk
110     ac     read_dsk       read a record from disk
114     ac     reload_font    reload font
118     ac     pollc          returns character in D1,
                              else -1 in d1.w
11C     ac     quiet_ret      quiet return to PROM
120     ac     write_disk
124     ac     log_error      on/off
128     ac     crash
12C     ac     led_update
```

RING REGISTERS

Refer to the RING REGISTERS section of CHAPTER 7, DN300,DN320,DN330 for this information.

SERIAL I/O INTERFACE

The DSP80,DSP80A uses only the Signetics SC2681 DUART (there is no keyboard interface).

Refer to the SERIAL I/O INTERFACE section of CHAPTER 7, DN300,DN320,DN330 for SIO information.

CHAPTER 12

DN3000

ADDRESS SPACE BY PHYSICAL ADDRESS

physical		virtual	I/O Bus
100400	traps	0	
400	prom	400 (1-to-1)	
100000	phys mem	100000->4FFFFF	
100800	phys mem	100800	
100000	debug data	3C00000	
8000	mmu/cpu	3FFB400	
8400	sios	3FFB000	
8800	timers	3FFAC00	
8900	calendar	3FFAD00	
9000	DMA1	3FFA000	
9100	DMA2	3FFA100	
9200	DMA page reg	3FFA200	
9300	Parity ppn	3FFA300	
9400	PIC1 (ints)	3FF9000	
9500	PIC2 (ints)	3FF9100	
0	DMMU RP	[cpu space 3]	
4000000	DMMU TB	[cpu space 3]	
8000000	DMMU TB flush	[cpu space 3]	
C000000	DMMU status	[cpu space 3]	
20000	PMMU	[cpu space 3]	
22000	68881 FP	[cpu space 3]	
40000	i/o bus (i/o addrs 000-3ff => 40000-6FFFF)		
4D000	win	3FFA800	1A0
50000	tape	3FF9C00	200
51000	ring page 1	3FF6800	220
51400	ring page 2	3FF6C00	228
51800	ring page 3	3FF7000	230
51C00	ring page 4	3FF7400	238
57C00	sio3		
58000	ethernet brd1	3FF5C00	300
58400	ethernet brd2	3FF6000	308
59000	ring page 5	3FF7800	320
59400	ring page 6	3FF7C00	328
59800	ring page 7	3FF8000	330
59C00	ring page 8	3FF8400	338
5BC00	parellel		
5D800	Mono	3FF9400	3B0
5DC00	Mono page 2	3FF9800	3B8
5E800	Color	3FF9400	3D0
5EC00	Color page 2	3FF9800	3D8
5F800	flp	3FFA400	3F0

```
 5FC00 sio2
 A0000 dsp_mem color 3FA0000->3FBFFFF
FA0000 dsp_mem mono  3FA0000->3FDFFFF
100000-8FFFFF main mem
900000->FFFFFF        AT-compatible memory space
```

Every 8 bytes of the bus i/o space (0-3FF) is mapped to one page of DN3000 space. Formula for translating AT-compatible bus i/o address to DN3000 bus is: (AT_addr/8) * $400 + $40000

BEEPER

The beeper is in the DN3000 keyboard and is accessed by writing to SIO line 0. Transmit following sequence to turn tone ON:

```
          $FF   $21   $81  $00
```

It will go off automatically after 300 milliseconds.

Transmit following sequence to turn tone OFF:

```
          $FF   $21   $82  $00
```

CALENDAR [8900 | 3FFAD00]

```
                                            RANGE
         +-----------------+
      00 +     SECONDS     +                 0-59
         +-----------------+
      01 +  SECONDS ALARM  +                 0-59
         +-----------------+
      02 +     MINUTES     +                 0-59
         +-----------------+
      03 +  MINUTES ALARM  +                 0-59
         +-----------------+
      04 +      HOURS      +                 0-23
         +-----------------+
      05 +   HOURS ALARM   +                 0-23
         +-----------------+
      06 +   DAY of WEEK   +                 1-7
         +-----------------+
      07 +  DATE of MONTH  +                 1-31
         +-----------------+
      08 +      MONTH      +                 1-12
         +-----------------+
      09 +      YEAR       +                 0-99
         +-----------------+
      0A + COMMAND REGISTER+
         +-----------------+
      0B + STATUS REGISTER +
         +-----------------+
      0C + UNUSED REGISTER +
         +-----------------+
      0D + UNUSED REGISTER +
         +-----------------+
   0E-11 +    CHECKSUM     +      50 bytes of battery backed up RAM
         +-----------------+      used by diagnostics for config info
   12-15 +  VALID PATTERN  +
         +-----------------+
   16-1D + MEM BOARD ARRAY +
         +-----------------+
   1E-21 +     NODEID      +
         +-----------------+     bit
   22-25 + DEV BIT ARRAY   +   <= 0 = flp
         +-----------------+      1 = ctape
      26 +  RING TYPE      +      2 = win
         +-----------------+      3 = fpu
      27 +  DISP TYPE      +      4 = ring
         +-----------------+      5 = user device
      28 +  DISK TYPE      +      6 = ethernet
         +-----------------+      7 = serial/parallel board
   29-3F +     UNUSED      +
         +-----------------+
```

Command Register   [890A | 3FFAD0A]

```
       7   6   5   4   3   2   1   0
     +---+---+---+---+---+---+---+---+
     |UIP| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
     +---+---+---+---+---+---+---+---+
       |
       1=> update in progress
```

Status Register   [890B | 3FFAD0B]

```
       7   6   5   4   3   2   1   0
     +---+---+---+---+---+---+---+---+
     |SET| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
     +---+---+---+---+---+---+---+---+
       |
       1=> abort current update, allow initialization.
```

{ defined in /os/ker/os_cal_asm.asm }

CARTRIDGE TAPE   [50000 | 3FF9C00]

Data Register   (read) [50000 | 3FF9C00]

Drive status information is returned as a series of 6 bytes through the data register after a read status command is sent.

Tape Status Byte 0

```
     7   6   5   4   3   2   1   0
   +---+---+---+---+---+---+---+---+
   | 0 |noc|usd|wp |eom|ude|bnl|fm |
   +---+---+---+---+---+---+---+---+
     |   |   |   |   |   |   |   1=> Filemark detected
     |   |   |   |   |   |   1=> Bad block not located
     |   |   |   |   |   1=> Unrecoverable data error
     |   |   |   |   1=> End of media (EOM)
     |   |   |   1=> Write protected cartridge
     |   |   1=> Unselected drive
     |   1=> Cartridge not in place
     0=> Status byte 0
```

Tape Status Byte 1

```
  7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+
| 1 |ill|nod|mgn|bom|bpe|erm|por|
+---+---+---+---+---+---+---+---+
  |   |   |   |   |   |   |   1=> Power on/reset occurred
  |   |   |   |   |   |   1=> Reserved for end of recorded media
  |   |   |   |   |   1=> Reserved for bus parity error
  |   |   |   |   1=> Beginning of media (BOM)
  |   |   |   1=> Marginal block detected
  |   |   1=> No data detected
  |   1=> Illegal command
  1=> Status byte 1
```

Status 0	Status 1	STATUS SUMMARY
11110001	00000000	Drive not ready (ctlr generated)
110X0000	00000000	No cartridge
11110000	00000000	No drive
10010000	X000X000	Write protected
10001000	00000000	End of media
100X0100	10010000	Read abort
100X0100	10001000	Write abort
100X0100	00000000	Read error, bad blk xfer
100X0110	00000000	Read error, filler blk xfer
100X0110	10100000	Read error, no data
100X1110	10100000	Read error, no data and EOM
100X0110	101X1XX0	Read error, no data and BOM
100X0001	00000000	Filemark read
XXXX0000	1100X000	Illegal command
XXXX0000	1000X001	Power on/reset
100X0001	00010000	Marginal block detected

Tape Status Byte 2 = high byte of data error counter
Tape Status Byte 3 = low byte of data error counter
Tape Status Byte 4 = high byte of underrun counter
Tape Status Byte 5 = low byte of underrun counter

Command Register (write) [50000 | 3ff9C00]

```
  7   6   5   4   3   2   1   0
+-----------+-------------------+
| cmd type  |     cmd data      |
+-----------+-------------------+
```

Command types:

000	MMMMM	Select
001	00MMM	Position
010	00000	Write Data
011	00000	Write File Mark
100	00000	Read Data
101	00000	Read File Mark
110	00000	Read Status
111		Reserved

Control Register (write only) [50001 | 3FF9C01]

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  |rst|req|ien|dni|    not used   |
  +---+---+---+---+---+---+---+---+
    |   |   |   1=> Enables DONE int, dni = 0 masks DONE int
    |   |   1=> Enables interrupts.
    |   1=> Request to LSI chip.
    1=> Reset controller microprocessor.
```

Status Register (read only) [50001 | 3FF9C01]

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  |irq|rdy|exc|don|dir| not used  |
  +---+---+---+---+---+---+---+---+
    |   |   |   |   1=> Direction, indicates direction of bus
    |   |   |   |       is from controller to DN3000.
    |   |   |   1=> Done, from DMA logic.
    |   |   0=> Exception, from LSI chip.
    |   0=> Ready from LSI chip.
    0=> Interrupt request flag, 'or' of rdy and exc, and done
        if dni is set.
```

DMA Go Register [50002 | 3FF9C02]

Start DMA (DMAGO). Any write to this register will cause DMAGO to be active.

DMA Reset Register [50003 | 3FF9C03]

Reset DMA (RSTDMA). Any write to this register will cause RSTDMA to be active.

{ defined in /os/ker/ct.pas }

CONFIGURATION

The block diagram below shows the major components of the DN3000 cpu and memory system, AT-compatible bus, and peripherals.

```
                              --------   --------  --------
                              ;      :   :      :  :       :
                              : SIO  :   : PROM :  :TIMER  :
                              :      :   :      :  :       :
 ---------      ---------     --------   -------   --------
 ;       ;      ;       ;       v          v          v
 ;  CPU  ;======;  MMU  ;==============================
 ;       ;  v   ;       ;             ^               v
 ---------  v   ---------       -----^--------        v
            v                   :            :        v
 ---------  v                   :  CALENDAR  :        v
 :       :  v                   :            :        v
 :  FP   :===                   --------------        v
 :       :                                            v
 ---------             -------         -------        v
                       :     :         :     :        v
                       : DMA :         : RAM :        v
                       :     :         :     :        v
                       -------         ------         v
    DN3000 bus            v               v           v
    =====================================================
                          v
                 ------------------
                 :                :
                 : Bus Interface  :
                 :                :
                 ------------------
                          v             AT-compatible Bus
            ---------     v     ---------
            :       :     v     :       :
            : Winch :====>v<====:Floppy :
            :       :     v     :       :
            ---------     v     ---------
                          v
           ----------     v     ---------
           :        :     v     :       :
           |Graphics|====>v<====| Network|
           |        |     v     |        |
           +--------+     v     +---------+
                          v
           +--------+     v
           |        |     v
           | Ct Tape|====>v
           |        |     v
           +--------+     v
```

CPU (MMU) CONTROL/STATUS REGISTER [8000 | 03FFB400]

Status Register (read)   [8000 | 03FFB400]

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| 0 | 0 |mto|uto|dto|pdm|pio|cto|pe3|pe2|pe1|pe0| ip| fp|iot| nm|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
          |   |   |   |   |   |  \_____________/  |   |   |   |
i/o bus <=1   |   |   |   |   |         |         |   |   |   |
mem space timeout |   |   |   |      1=> parity   |   |   |   |
              |   |   |   |   |          error    |   |   |   |
Coproc bus  <=1   |   |   |   |     (failing byte)|   |   |   |
  timeout         |   |   |   |                   |   |   |   |
                  |   |   |   |                   |   |   |   |
  i/o bus DMA   <=1   |   |   |                   |   |   |   |
      timeout         |   |   |                   |   |   |   |
                      |   |   |    int pending  <=1   |   |   |
 Parity during DMA  <=1   |   |                       |   |   |
                          |   |      fp owner trap  <=1   |   |
      IO parity error   <=1   |                           |   |
      (on i/o bus ref)        |       i/o bus i/o cycle <=1   |
                              |         timeout               |
       on-board CPU timeout <=1                normal mode  <=1
          (ref to non-existent mem)
```

To clear bus timeouts or parity conditions from status register, write to the status register. This register is readonly. It will respond to byte mode access.

Control Register (write)   [8100 | 03FFB500]

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|ld7|ld6|ld5|ld4|ld3|ld2|ld1|ld0|pe3|pe2|pe1|pe0| dg| fp|rsa|nme|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
\______________________________/ \_____________/  |   |   |   |
           |                           |          |   |   |   |
           1=>  led off                |          |   |   |   |
                   force parity err  <=1          |   |   |   |
                    (bytes 3-0)                   |   |   |   |
                             enable diag mode   <=1   |   |   |
                              enable fp owner trap  <=1   |   |
                               reset on-board devices   <=1   |
                              non-maskable interrupt enable <=1
```

Neither RSA nor the reset instruction reset the SIO lines. Non-maskable interrupts must be enabled to receive parity errors. This register is write-only (the BSET instruction may not be used to turn bits on and off). It will respond to byte mode access.

DISK (FLOPPY/WINCHESTER) CONTROLLER [4D000 | 3FFA800]

Status Register [4D000 | 3FFA800]

```
      7   6   5   4   3   2   1   0
    +---+---+---+---+---+---+---+---+
    | 0 | 0 |int|dma|slc| ph|dir|avl|
    +---+---+---+---+---+---+---+---+
              |   |   |   |   |   |
              |   |   |   |   |   1=> available for transfer
              |   |   |   |   1=> transfer to host
              |   |   |   |   0=> transfer from host
              |   |   |   1=> command phase
              |   |   |   0=> data phase
              |   |   1=> controller selected
              |   1=> dma request
              1=> interrupt request
```

Write anything to this register to RESET controller.

Jumper Setting Register [4D002 | 3FFA802]

```
      7   6   5   4   3   2   1   0
    +---+---+---+---+---+---+---+---+
    | 0 | 0 | 0 | 0 | j1| j2| j3| j4|
    +---+---+---+---+---+---+---+---+
                      |   |   |   |
                      |   |   |   1=> jumper4 in place
                      |   |   1=> jumper3 in place
                      |   1=> jumper2 in place
                      1=> jumper1 in place
```

Write anything to this register to generate controller select.

Interrupt Mask Register (write only) [4D003 | 3FFA803]

```
      7   6   5   4   3   2   1   0
    +---+---+---+---+---+---+---+---+
    | 0 | 0 | 0 | 0 | 0 | 0 |dma|int|
    +---+---+---+---+---+---+---+---+
                              |   |
                              |   1=> enable interrupt
                              1=> enable dma
```

Command/Data/Status Register [4D000 | 3FFA800]

All commands are given by moving 6 bytes to the command/data register. The copy command is the only exception which requires 10 bytes. Every command sequence has three phases:

    Command phase
    Data in/out phase
    Status phase (pertains to command op only, not hardware status)

Command Phase

```
                 7    6    5    4    3     2     1     0
     ________________________________________________
     | BYTE 0|Command Class|      Op code               |
     |-------|-----|-------|----------------------------|
     | BYTE 1|MSB  | LUN   |      Head Number           |
     |       |CYL  |       |                            |
     |-------|-----|-------|----|-----------------------|
     | BYTE 2| Cyl.High |      Sector Number            |
     |-------|----------|-------------------------------|
     | BYTE 3|           Cylinder Low                   |
     |-------|------------------------------------------|
     | BYTE 4|      Interleave or Block Count           |
     |-------|------------------------------------------|
     | BYTE 5|           Control Byte                   |
     --------------------------------------------------
```

Control Byte Format

```
        7    6    5    4    3    2    1    0
     +----+----+-------------+-------------+
     | r  | a  | 0    0    0 | s    s    s |
     +----+----+-------------+-------------+
        |    |                \___________/
        |    |               drive type and step option:
        |    |                0   0   0    unknown drive. 3 ms per step
        |    |                0   0   1    N/A
        |    |                0   1   0    25 usec per step
        |    |                0   1   1    50 usec per step
        |    |                1   0   0    200 usec per step
        |    |                1   0   1    70 usec per step
        |    |                1   1   0    3  ms per step
        |    |                1   1   1    3  ms per step
        |    1=> disable ECC retries
        1=> disable Retries
```

Status Phase Byte

```
        7   6   5   4   3   2   1   0
     +---+---+---+---+---+---+---+---+
     | 0 | t | d | 0 | 0 | 0 | e | 0 |
     +---+---+---+---+---+---+---+---+
           |   |                 1=> error condition, check sense.
           |   = winchester drive number
           1=> winchester status
           0=> tape status
```

Error and Operation Codes

Error codes:

Drive errors	
No Index signal from Winc	$01
No seek complete	$02
Write Fault	$03
Drive not ready	$04
No track zero found	$06
Seek in progress	$08
Data errors	
ID Read error	$10
Uncorrectable Data error	$11
ID address Mark not found	$12
Sector not found	$14
Seek error	$15
Sequencer/DMA failure	$16
Write protected	$17
Correctable Data error	$18
Bad track	$19
Command errors	
Invalid Command	$20
Illegal disk address	$21
Illegal Func for Drv Type	$22
Volume Overflow	$23
Diagnostics errors	
RAM error	$30
Program Memory chksm err	$31
Processor Test error	$32
Winc control test error	$33

Op Codes:

Test Drive Ready	$00
Recalibrate	$01
Request Sense	$03
Format Drive	$04
Read Verify	$05
Format Track	$06
Format Bad Track	$07
Read	$08
Write	$0A
Seek	$0B
Initialize	$0C
Read from Sector Buffer	$0E
Write Data to Sector Buffer	$0F
Assign Alternate Track	$10
Copy	$20
Read to Sector Buffer	$30
RAM Diagnostic	$E0
Read ID Device Control Block	$E2
Drive Diagnostic	$E3
Controller Internal Diag	$E4
Read Long	$E5
Write Long	$E6

Error codes 'xx' are returned by sense_status and reported by MD driver as:

```
DISK operation ERROR: xx recordnum unit type
```

Floppy [5F800 | 3FFA400]

Digital Output Register [5F802 | 3FFA402] write only

```
    7   6   5   4   3   2   1   0
  +-------+---+---+---+---+---+---+
  |reserve|BEN|AEN|ENB|RST|XXX|SLC|
  +-------+---+---+---+---+---+---+
            |   |   |   |   |   |
            |   |   |   |   |   1=> Drive select
            |   |   |   1=> Reset
            |   |   1=> Enable interrupts and DMA
            |   1=> Drive motor B enable
            1=> Drive motor B enable
```

Main Status Register (read only) [5F804 | 3FFA404]

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  |RQM|DIO|EXM|CB |D3B|D2B|D1B|D0B|
  +---+---+---+---+---+---+---+---+
    |   |   |   |   |   |   |   1=> FDD 0 in Seek Mode
    |   |   |   |   |   |   1=> FDD 1 in Seek Mode
    |   |   |   |   |   1=>FDD 2 in Seek Mode
    |   |   |   |   1=> FDD 3 in Seek Mode
    |   |   |   1=> BUSY, Read or Write Command in process
    |   |   1=> Execution mode for non-DMA transfers
    |   1=> I/O Direction, 1 = Data Reg to Processor, 0 =
    |       Processor to Data Reg.
    1=> Data register ready to send/receive data from
        Processor.
```

Status information is returned as a series of bytes from the data register after completion of a command. The result phase of most commands produce status register 0,1,2. Recalibrate, Specify, and Seek return no status information. Sense drive status returns status register 3 and Sense interrupt status returns status register 0 only.

Status Register 0 [5F805 | 3FFA405]

Returned at the end of a command through the data address.

```
    7   6   5   4   3   2   1   0
  +-------+---+---+---+---+---+---+
  |  I C  |SE |EC |NR |HD |US1|US0|
  +-------+---+---+---+---+---+---+
  \_______/ |   |   |   |   |   |
      |     |   |   |   |   |   1=> Unit Select 0 Status
      |     |   |   |   |   1=> Unit Select 1 Status
      |     |   |   |   |       fails to occur after 77 Step Pulses
      |     |   |   |   1=> Head 1 0=> HEAD 0
      |     |   |   1=> Not Ready
      |     |   1=> Equipment check or Track 0 Signal
      |     1=> Seek Completed
      |
    0   0 =>  Normal Termination of Command
    0   1 =>  Abnormal Term {command started, not completed}
    1   0 =>  Invalid Command{ command issued never started }
    1   1 =>  Abnormal Termination{ FDD went not ready during
              command execution }
```

Status Register 1 [5F805 | 3FFA405]

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  |ECY| 0 |DE |OR | 0 |ND |NW |MA |
  +---+---+---+---+---+---+---+---+
    |       |   |       |   |   1=> missing address mark
    |       |   |       |   1=> write protected
    |       |   |       1=> no data found
    |       |   1=> overrun
    |       1=> data error
    1=> end of cylinder error
```

Status Register 2 [5F805 | 3FFA405]

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  | 0 |CM |DD |WC |SH |SN |BC |MD |
  +---+---+---+---+---+---+---+---+
        |   |   |   |   |   |   1=> missing addr mark in data field
        |   |   |   |   |   1=> bad cylinder
        |   |   |   |   1=> scan not satisfied
        |   |   |   1=> scan equal hit
        |   |   1=> wrong cylinder
        |   1=> data error in data field
        1=> deleted data-address mark encountered
```

Status Register 3 [5F805 | 3FFA405]

```
   7   6   5   4   3   2   1   0
 +---+---+---+---+---+---+---+---+
 |FT |WP |RDY|TR0|TS |HD |UN1|UN0|
 +---+---+---+---+---+---+---+---+
   |   |   |   |   |   |   |   1=> unit 0 select
   |   |   |   |   |   |   1=> unit 1 select
   |   |   |   |   |   1=> head address
   |   |   |   |  two sided from drive
   |   |   |  track 0 from drive
   |   |  ready from drive
   |  write protect from drive
  fault status from drive
```

Additional Control Register (write only) [5F806 | 3FFA406]

```
   7   6   5   4   3   2   1   0
 +-------+---+---+---+---+---+---+
 |reserve|PN6|PN4|PN2|WP2|WP1|WP0|
 +-------+---+---+---+---+---+---+
           |   |   |   |   |   1=> Write precomp 0
           |   |   |   |   1=> Write precomp 1
           |   |   |   1=> Write precomp 2
           |   |   1=> Interface pin 2 (density & speed control)
           |   1=> Interface pin 4
           1=> Interface pin 6
```

Digital Input Register (read only) [5F807 | 3FFA407]

```
   7   6   5   4   3   2   1   0
 +---+---+---+---+---+---+---+---+
 |DCH|WG |HD3|HD2|HD1|HD0|UN1|UN0|
 +---+---+---+---+---+---+---+---+
   |   |   |   |   |   |   |   1=> unit 0 select
   |   |   |   |   |   |   1=> unit 1 select
   |   |   |   |   |   1=> Head select 0
   |   |   |   |   1=> Head select 1
   |   |   |   1=> Head select 2
   |   |   1=> Head select 3/ Reduced write current
   |   1=> Write gate
   1=> Diskette changed
```

Diskette Control Register (write only) [5F807 | 3FFA407]

```
     7    6    5    4    3    2    1    0
  +------------------------+--------+
  |          UNUSED        |TRN RTE|   transfer rate
  +------------------------+--------+
                            \________/
                                 |
                         Bit 1  Bit 0
                          -------------
                           0     0     500 KBPS
                           0     1     300 KBPS
                           1     0     250 KBPS
                           1     1     N/A
```

(defined in /os/ker/flp.pas)

DISPLAY (COLOR) [5E800 | 3FF9400]

Display memory [A0000 | 3FFA000]

4 planes, each 1024x1024 (1024x800 visible)

Status Register (read only) [5E800 | 3FF9400]

```
  7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+
|blk| vb|syn|rmc|alt| ad|hck|lok|
+---+---+---+---+---+---+---+---+
  |   |   |   |   |   |   |   0=> okay to load LUT (100us)
  |   |   |   |   |   |   0=> horiz sync clock (diagnostics)
  |   |   |   |   |   0=> a-d conversion done (diagnostics)
  |   |   |   |   x=> alternating BLT state
  |   |   |   1=> read-mod-write cycle req'd
  |   |   0=> composite sync
  |   0=> vertical blank
  0=> composite horiz and vert blank
```

Display-Type Register (read only) [5E801 | 3FF9401]

```
  7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
+---+---+---+---+---+---+---+---+
```

Raster Op Register (write only) [5E802 | 3FF9402]

Used to specify the raster operation for each of the 4 planes.

```
 15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|rp3|rp3|rp3|rp3|rp2|rp2|rp2|rp2|rp1|rp1|rp1|rp1|rp0|rp0|rp0|rp0|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

$0		Zero	
$1	Source	AND	Destination
$2	Source	AND	(-Destination)
$3	Source		
$4	(-Source)	AND	Destination
$5			Destination
$6	Source	XOR	Destination
$7	Source	OR	Destination
$8	Source	NOR	Destination
$9	Source	XNOR	Destination
$A			-Destination
$B	Source	OR	(-Destination)
$C	-Source		
$D	(-Source)	OR	Destination
$E	Source	NAND	Destination
$F		One	

BLT Control Register [5EC00 | 3FF9800]

```
  7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+
|md2|md1|md0| s4| s3| s2| s1| s0|
+---+---+---+---+---+---+---+---+
 \_________/ \_________________/
      |              |
      |         Signed shift count (positive is right-shift)
      |
      => Selects blt and memory access mode
         000 => CPU destination BLT
                (bus write to provide disp mem addr,
                 bus read to get data)
         001 => Alternating BLT
                (alternating bus writes provide src/dest addrs,
                 second write provides Write-enables)
         010 => Vector or fill mode
                (bus write to provide Write-enables and addr)
         011 => CPU source BLT
                (bus write to provide src data,
                 bus write to provide Write-enables and addr)
         100 => Double access BLT    {fastest BLT mode}
                (bus write to provide src addr on address lines,
                 dest addr on data lines (16-bit WORD offset)
         111 => Normal access to display memory
```

Misc Control Register   [5EC02 | 3FF9802]

```
  7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+
|adb|dvc|dhc|ren|rst|dpc|syg|ven|
+---+---+---+---+---+---+---+---+
  |   |   |   |   |   |   |   1=> video enable
  |   |   |   |   |   |   1=> enable sync generator (normal = 1)
  |   |   |   |   |   0=> diagnostic pixel clock (normal = 1)
  |   |   |   |   |       watched by hardware only when syg = 0
  |   |   |   |   |       software use: 1 => inuse
  |   |   |   |   0=> reset sync generator (normal = 1)
  |   |   |   1=> raster op enable, raster op spec in ROP reg
  |   |   0=> disable horiz clock (normal = 1)
  |   |       watched by hardware only when syg = 0
  |   0=> disable vert mode clock (normal =1)
  |       watched by hardware only when syg = 0
  1=> additive blt or normal memory access
```

Plane Select Register   [5EC04 | 3FF9804]

```
  7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+
| 0 | 0 |spa|spb|dp3|dp2|dp1|dp0|
+---+---+---+---+---+---+---+---+
 \_____/ \_____/  |   |   |   |
    |       |     |   |   |   0=> select plane 0
    |       |     |   |   0=> select plane 1
    |       |     |   0=> select plane 2
    |       |     0=> select plane 3
    |       |     Any combination of destination planes may
    |       |     selected.
    |      => Source plane selection:
    |       00 => Plane 0
    |       01 => Plane 1
    |       10 => Plane 2
    |       11 => Plane 3
   => BLT source control:
      00 => set source to all ones (used for vectors)
      01 => replicate 4 lsb of data bus
      10 => replicate lsb of shifter
      11 => use source data unchanged (normal use)
```

Chip Init Register [5EC06 | 3FF9806]

Initializes the 8255A chip. Set just once per system reset.

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
  +---+---+---+---+---+---+---+---+
    1 => set all ports to basic output mode
```

To initialize and set normal access to display memory (no blts):

```
Chip_reg    = $80     set all ports to basic output mode
Misc_cntl   = $8B     enable display, normal memory access
Plane_sel   = $CE     select plane 0 for normal memory access
Blt_cntl    = $E0     set for normal memory access, no shift
```

Write-enable Register (write only) [5E800 | 3FF9400]

```
   15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x | x |
  +---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  0=> writing enabled.
  1=> writing disabled.
```

Used to select the pixels within a word to be modified for all display memory-write operations. For normal writes, this register should be cleared (all pixels enabled).

Lookup Table Registers

```
 Red LUT Register   :  [ 5EC01 | 3FF9801 ]
 Green LUT Register :  [ 5EC03 | 3FF9803 ]
 Blue LUT  Register :  [ 5EC05 | 3FF9805 ]

 Layout of each register:
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  | a3| a2| a1| a0| d3| d2| d1| d0|
  +---+---+---+---+---+---+---+---+
   \_____________/ \_____________/
          |               |
   Address in LUT   Data to be loaded
```

Both registers and LUTs are write-only. LUTS can be loaded only during the vertical blanking period. When LOK=0 in status register, there is sufficient time to load all 16 locations of the three LUTs.

Diagnostic Register (write only) [5EC07 | 3FF9807]

```
 WRITE:
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  | 0 | 0 | 0 | 0 | 0 | 1 | x | x |
  +---+---+---+---+---+---+---+---+
                           \_____/
                              |
                   Input Channel Select:
                     00 => Red Video Output
                     01 => Green Video Output
                     10 => Blue Video Output
 READ:
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  | x | x | x | x | x | x | x | x |
  +---+---+---+---+---+---+---+---+
   \_____________________________/
                  |
           Results encoding:
             00 => 0.00 Volts
             01 => 0.01 Volts
             80 => 1.28 Volts
             FF => 2.55 Volts
```

DISPLAY (MONOCHROME) [5D800 | 3FF9400]

Display memory [FA0000 | 3FFA000]

2048x1024 (1280x1024 visible)

Status Register (read only) [5D800 | 3FF9400]

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  |blk| vb| hs|rmc|alt| vs|hck| vd|
  +---+---+---+---+---+---+---+---+
    |   |   |   |   |   |   |   0=> video data (diagnostics)
    |   |   |   |   |   |   0=> horiz sync clock (diagnostics)
    |   |   |   |   |   0=> vertical sync (diagnostics)
    |   |   |   |   x=> alternating BLT state
    |   |   |   1=> read-mod-write cycle req'd
    |   |   0=> horizontal syn (diagnostics)
    |   0=> vertical blank
    0=> composite horiz and vert blank
```

Display-Type Register (read only) [5D801 | 3FF9401]

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
  +---+---+---+---+---+---+---+---+
```

Raster Op Register (write only) [5E802 | 3FF9402]

Used to specify the raster operations.

```
  15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
+-----------------------------------------------+---+---+---+---+
|          not  used                            |rpv|rpv|rpv|rpv|
+-----------------------------------------------+---+---+---+---+
```

Definitions for rp values same as color controller.

BLT Control Register [5EC00 | 3FF9800]

Same as color.

Misc Control Register [5EC02 | 3FF9802]

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  |inv|b17|dhc|ren|rst|dpc|syg|ven|
  +---+---+---+---+---+---+---+---+
    |   |   |   |   |   |   |   1=> video enable
    |   |   |   |   |   |   1=> enable sync generator (normal = 1)
    |   |   |   |   |   0=> diagnostic pixel clock (normal = 1)
    |   |   |   |   |       watched by hardware only when syg = 0
    |   |   |   |   |       software use: 1 => inuse
    |   |   |   |   0=> reset sync generator (normal = 1)
    |   |   |   1=> raster op enable, raster op spec in ROP reg
    |   |   0=> disable horiz clock (normal = 1)
    |   |       watched by hardware only when syg = 0
    |   0=> BLT Mode 4 destination address bit 17 (syg =1)
    |       when syg=0, disable vert mode clock (diagnostics)
   Inverse Video Control (0=> 1 is black, 0 is white)
                         (1=> 0 is black, 1 is white)
```

Plane Select Register [5EC04 | 3FF9804]

```
    7   6   5   4   3   2   1   0
  +---+---+----------------------+
  | 0 | 0 |   unused             |
  +---+---+----------------------+
   \_____/
      |
      => BLT source control:
         00 => set source to all ones (used for vectors)
         01 => replicate 4 lsb of data bus
         10 => replicate lsb of shifter
         11 => use source data unchanged (normal use)
```

Chip Init Register [5EC06 | 3FF9806]

Same as color.

Write-enable Register (write only) [5E800 | 3FF9400]

Same as color.
{ color4_$disp_regs in /us/ins/color4.ins.pas }

DMA CONTROL [9000 | 3FFA000] [9100 | 3FFA100]

DMA control uses two Intel 8237 DMA Controller Chips. The first supports only 8 bit transfers (channels 0-3) and the second supports 16 bittransfers (channels 5-7). Channel 4 is used to cascade channels 0-3 to the cpu and so is not available for dma. (defined in /os/ker/dma.pas)

Channel Usage

Chip 1 [9000 | 3FFA000]		Chip 2 [9100 | 3FFA100]	
CH0	avail	CH4	cascade for chip 1
CH1	SDLC option	CH5	avail
CH2	floppy	CH6	avail
CH3	avail	CH7	avail

Command Register (write) [9008/9108]

```
   7   6   5   4   3   2   1   0
 +---+---+---+---+---+---+---+---+
 |ack|drq|slt|pri|tim|ctl|hld|mmc|
 +---+---+---+---+---+---+---+---+
   |   |   |   |   |   |   |   |
   |   |   |   |   |   |   |   1=> memory to memory enable
   |   |   |   |   |   |   1=> hold enable
   |   |   |   |   |   1=> controller disable
   |   |   |   |   1=> compressed timing
   |   |   |   1=> rotating priority
   |   |   1=> extended write selection
   |   1=> DREQ sense active low
   1=> DACK sense active high
```

Status Register (read) [9008/9108]

```
   7   6   5   4   3   2   1   0
 +---+---+---+---+---+---+---+---+
 |rq3|rq2|rq1|rq0|tc3|tc2|tc1|tc0|
 +---+---+---+---+---+---+---+---+
   |   |   |   |   |   |   |   |
   |   |   |   |   |   |   |   1=> chan 0 has reached TC
   |   |   |   |   |   |   1=> chan 1 reached TC
   |   |   |   |   |   1=> chan 2 has reached TC
   |   |   |   |   1=> chan 3 has reached TC
   |   |   |   1=> chan 0 requesting service
   |   |   1=> chan 1 requesting service
   |   1=> chan 2 requesting service
   1=> chan 3 requesting service
```

Single Channel Mask Register [900A/910A]

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+--------+
  | 0 | 0 | 0 | 0 | 0 |msk|channel |
  +---+---+---+---+---+---+--------+
                      | \_______/
                      |     |=> select channel 0-3
                      0 => enable channel
                      1 => disable channel
```

All Channels Mask Register [900F/910F]

```
    7   6   5   4   3   2   1   0
  +---+---+---+---+---+---+---+---+
  | 0 | 0 | 0 | 0 |c3m|c2m|c1m|c0m|
  +---+---+---+---+---+---+---+---+
                  |   |   |   1=> disable chan 0
                  |   |   1=> disable chan 1
                  |   1=> disable chan 2
                  1=> disable chan 3
```

Mode register (write) [900B/910B]

```
    7   6   5   4   3   2   1   0
  +-------+---+---+-------+--------+
  | mode  |ads|aut|trnsfr |channel |
  +-------+---+---+-------+--------+
  \_______/ |   | \_______/\______/
     |      |   |     |       |=> channel select
     |      |   |     |    | 00 verify transfer
     |      |   |     |===>| 01 write transfer
     |      |   |          | 10 read transfer
     |      |   |          | 11 illegal
     |      |   1=> auto-initialize enable
     |      1=> address decrement select
     |  | 00 demand mode
     |=>| 01 single mode
        | 10 block mode
        | 11 cascade mode
```

Clear Byte Pointer [900D/910D]

This register must be written to (any value) before loading address and count registers.

DMA Page Registers [9200 | 3FFA200]

The DMA chips support only 16-bit addressing, but the system supports 24 bits of physical address. To supply the extra bits of addressing required, 8 one-byte page registers are provided. Each byte is loaded with the high 8 physical address bits for its corresponding DMA channel. Note that DMA can operate on a maximum of 1024 bytes (each channel has only ONE page register).

Addresses of the DMA page registers (note the non-order):

9207	page register for CH0
9203	page register for CH1
9201	page register for CH2
9202	page register for CH3
920B	page register for CH5
9209	page register for CH6
920A	page register for CH7

FAULT FRAME AND FAULT TYPES

Refer to the FAULT FRAME section of CHAPTER 7, DN300,320 for this information.

FAULT VECTORS

Exception page at [100400 | 0]

Vector	Address	Assignment
00	000	Reset: Initial SSP
	004	Reset: Initial PC
02	008	Bus Error
03	00C	Address Error
04	010	Illegal Instruction
05	014	Zero Divide
06	018	CHK Instruction
07	01C	TRAPV Instruction
08	020	Privilege Violation
09	024	Trace
0A	028	Unimplemented instruction
0B	02C	F-line trap (co-processor)
0C	030	(Unassigned, reserved)
0D	034	Co-processor protocol violation
0E	038	Invalid Stack Format
0F-17	03C	(Unassigned, reserved)
18	060	Spurious Interrupt
19-1E	064	Auto vectors 1-6 (unused)
1F	07C	Parity Error non-maskable level 7
20-2F	080	TRAP Instruction Vectors
30	0C0	Floating Point Branch Set on Unord Cond.
31	0C4	Floating Point Inexact Result
32	0C8	Floating Point Divide by Zero
33	0CC	Floating Point Underflow
34	0D0	Floating Point Operand Error
35	0D4	Floating Point Overflow
36	0D8	Floating Point Not a Number
37-3F	0D8	(Unassigned, reserved)
40-9F	100	User Int Vectors - used for prom entry pts
A0	280	IRQ0 - Timers 1,2,3
A1	284	IRQ1 - Sio lines (2 ports)
A2	288	IRQ2 - Ring
A3	28C	IRQ3 - PIC 2 to PIC 1 (unseen by software)
A4	290	IRQ4
A5	294	IRQ5 - Tape
A6	298	IRQ6 - Floppy
A7	29C	IRQ7
A8	2A0	IRQ8 - Calendar
A9	2A4	IRQ9 - Ethernet board 1
AA	2A8	IRQ10 - Ethernet board 2
AB	2AC	IRQ11 - PC-board primary
AC	2B0	IRQ12
AD	2B4	IRQ13 - Diagnostic Interrupt
AE	2B8	IRQ14 - Winchester
AF	2BC	IRQ15 - PC-board alternate
B8-FF	2C0	User Int Vectors -- unused

FLOATING-POINT REGISTERS

Refer to the FLOATING-POINT REGISTERS section of CHAPTER 7, DN300,DN320,DN330 for this information.

PARITY ERROR REGISTER (read only) [9300 | 03FFA300]

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+-------------------------------------------------------+
| x | x |              FAILING    PPN                           |
+---+---+-------------------------------------------------------+
```

The upper two bits must be masked off.

{ type mmu_$parity_ppn in /os/kins/term.pvt.pas }

PROGRAMMABLE INTERRUPT CONTROLLER [9400/9500 | 3FF9000/3FF9100]

Master: [9400 | 3FF9000]   Slave: [9500 | 3FF9100]

There are two Intel 8259A interrupt control chips, the master takes interrupt request lines 0-7; the slave responds to lines 8-15 and is cascaded through request line 3 of the master. This is a complicated chip. Only features used are described.

{ defined in /os/ker/pbu.pas }

Port 0 (+000)

```
   Write '0A' then read Interrupt Request Register at same loc:
        7   6   5   4   3   2   1   0
      +---+---+---+---+---+---+---+---+
      | I7| I6| I5| I4| I3| I2| I1| I0|
      +---+---+---+---+---+---+---+---+
       1=>  channel is requesting interrupt.

   Write '0B' then read In-service Register at same location:
        7   6   5   4   3   2   1   0
      +---+---+---+---+---+---+---+---+
      | S7| S6| S5| S4| S3| S2| S1| S0|
      +---+---+---+---+---+---+---+---+
       1=>  channel interrupt has been accepted.
```

Write '20' (non-specific EOI) to this register to acknowledge and clear highest level In-service Request.

Port 1 (+001)

```
   Interrupt mask register:
        7   6   5   4   3   2   1   0
      +---+---+---+---+---+---+---+---+
      | M7| M6| M5| M4| M3| M2| M1| M0|
      +---+---+---+---+---+---+---+---+
        1=>  channel is masked (ints inhibited)
```

Programmable Interrupt Controller Initialization Sequence

Interrupt vectors are programmed to start at $280.

```
   Master   [ 9400 | 03FF9000 ]
        Port 0   ICW1 = 19    {level triggered, need ICW4}
        Port 1   ICW2 = A0    {vector byte value for ints at $280+}
        Port 1   ICW3 = 08    {have slave at IRQ3}
        Port 1   ICW4 = 01    {normal EOI, 8086 mode}
        Port 1   OCW1 = FF    {mask all ints till further notice}

   Slave    [ 9500 | 03FF9100 ]
        Port 0   ICW1 = 19    {level triggered, need ICW4}
        Port 1   ICW2 = A8    {vector byte value for ints at $2A0+}
        Port 1   ICW3 = 03    {am slave with ID 3}
        Port 1   ICW4 = 01    {normal EOI, 8086 mode}
        Port 1   OCW1 = FF    {mask all ints till further notice}
```

Interrupt Request Line Assignments

Master	Slave	Device
IRQ 0		timers
IRQ 1		sio
IRQ 2		ring
IRQ 3----->		slave pic to master
	IRQ8	calendar
	IRQ9	ethernet board 1
	IRQ10	ethernet board 2
	IRQ11	pc option primary
	IRQ12	
	IRQ13	
	IRQ14	winchester
	IRQ15	pc option alternate
IRQ 4		
IRQ 5		tape
IRQ 6		floppy
IRQ 7		

On slave interrupt, EOI slave first, then master. The priority of IRQ8-15 is higher than IRQ4-7 since they use the second PIC which is slaved to the master at IRQ3.

RING [51000-51C00 59C00-59C00 | 3FF6800-3FF8400]

The DN3000 ring board has an 8K WORD buffer for xmitting/rcving msgs. There is room for 7 rcv msg buffers (each 1k bytes of header and 1k bytes of data) and 1 xmit msg buffer of the same size. In the OS, msgs to the network protocol gate array for initialization use the xmit buffer. There is only one read/write port to the buffer (RAM_DATA) and one address register (RAM_ADDR). The buffer is read or written by setting the ram_addr and doing sequential word reads/writes to the ram_data port. After setting ram_addr, the first word read should be discarded.

RING REGISTERS

Bus Addr	Phy	AEGIS		When Read	When Written
220	51000	3FF6800		Node_ID3 (msb)	SOFT_RCV_REQ
222	51002	3FF6802		Node_ID2	XMIT_ACK
224	51004	3FF6804		Node_ID1	RCV_ACK
226	51006	3FF6806		Node_ID0 (lsb)	TIMO_ACK
228	51400	3FF6C00		(unused - PROM)	ERR_BITS_CLR
22A	51402	3FF6C02		(unused - PROM)	GPS_CLR
22C	51404	3FF6C04		(unused - PROM)	SOFT_XMIT_REQ
22E	51406	3FF6C06		(unused - PROM)	LERR_CLR
230	51800	3FF7000		(unused - PROM)	-
232	51802	3FF7002		(unused - PROM)	-
234	51804	3FF7004		(unused - PROM)	-
236	51806	3FF7006		(unused - PROM)	-
238	51C00	3FF7400		(unused - PROM)	-
23A	51C02	3FF7402		(unused - PROM)	-
23C	51C04	3FF7404		(unused - PROM)	-
23E	51C06	3FF7406		Node_ID_CHECKSUM	-
320	59000	3FF7800		BOARD_TYPE	BOARD_RESET
322	59002	3FF7802	*1	XMIT_ADDR	XMIT_ADDR
324	59004	3FF7804	*2	XMIT_ABORT/RCV_ADD	RCV_ADDR
326	59006	3FF7806	*1	RAM_ADDR	RAM_ADDR
328	59400	3FF7C00		MISC_STAT	MISC_CMD
32A	59402	3FF7C02		XMIT_STAT	XMIT_CMD
32C	59404	3FF7C04		RCV_STAT	RCV_CMD
32E	59406	3FF7C06		RAM_DATA	RAM_DATA
330	59800	3FF8000		RCV_HDR_CNT	RCV_HDR_CNT
332	59802	3FF8002		RCV_PKT_CNT	RCV_PKT_CNT
334	59804	3FF8004		RCV_MAX_CNT	RCV_MAX_CNT
336	59806	3FF8006		CNT_C0_CNTR	CNT_C0_CNTR
338	59C00	3FF8400		XMIT_HDR_CNT	XMIT_HDR_CNT
33A	59C02	3FF8402		XMIT_PKT_CNT	XMIT_PKT_CNT
33C	59C04	3FF8404		BAD_PKT_CNT	BAD_PKT_CNT
33E	59C06	3FF8406		CNT_C1_CNTR	CNT_C1_CNTR

*1 Registers are write only for the two-board version.

*2 Two-board version: XMIT_ABORT.
Single-board version: RCV_ADDR.
(Single-board abort is clear xmit_enable in xmit_cmd.)

RING COMMAND AND STATUS REGISTERS

MISC_STS (read) [59400 | 3FF7C00]

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|nct|tmo|xby|rby|iov|rlk|esb|bpe| x | x | x | x |gps| xi| ri|tmi|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
 |   |   |   |   |   |   |   |  sticky good pkt <=1   |   |   |
 |   |   |   |   |   |   |   |    XMIT intr pending <=0   |   |
 |   |   |   |   |   |   |   |         RCV intr pending <=0   |
 |   |   |   |   |   |   |   |      GA timeout intr pending <=0
 |   |   |   |   |   |   |   1=> sticky bi-phase error
 |   |   |   |   |   |   1=>sticky elastic store buffer error
 |   |   |   |   |   1=> rcv lock error
 |   |   |   |   1=> initialize overrun
 |   |   |   0=> rcv busy
 |   |   0=> xmt busy
 |   1=> timeout
 0=>network connect
```

RCV_STS (read) [59404 | 3FF7C04]

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| pe|cpd|wak|icp| de|pke|akp|ife|nct|ren|rby|bpe|esb|rc2|rc1|rc0|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  |   |   |   |   |   |   |   |   |   |   |   1=> bi-phase error
  |   |   |   |   |   |   |   |   |   |   |       (phs_rs)|   |
  |   |   |   |   |   |   |   |   |   |   0=> rcv busy    |   |
  |   |   |   |   |   |   |   |   |   1=> rcv enable  |   |   |
  |   |   |   |   |   |   |   |   0=> network connect     |   |
  |   |   |   |   |   |   |   1=> ife             |   |   |   |
  |   |   |   |   |   |   |       (overrun or controller error)
  |   |   |   |   |   |   1=> ack parity error    |   |   |   |
  |   |   |   |   |   |       (ackparerr_rs)      |   |   |   |
  |   |   |   |   |   1=> packet error            |   |   |   |
  |   |   |   |   |       (pkt_err/ackbyte_errbit_rs) |   |   |
  |   |   |   |   1=> data error  (crc_rs)        |   |   |   |
  |   |   |   1=> icopy                           |   |   |   |
  |   |   1=> wak                                 |   |   |   |
  |   1=> copied                                  |   |   |   |
  1=> protocol error  (timeout_rs)                |   |   |   |
            elastic store buffer error (esb_rs) <=1   |   |   |
                           rcv counter output bit 2 <=1   |   |
                           (pkt exceeded max_rcv_cnt)     |   |
                               rcv counter output bit 1 <=1   |
                               (data rcv in progress)         |
                                   rcv counter output bit 0 <=1
                                   (hdr rcv in progress)
```

```
XMIT_STS (read) [ 59402 | 3FF7C02 ]

 15  14  13  12  11  10  9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| pe| x | x | x | x | x |abt| x |nct|xen|iby|xby| x | x |xt1|xt0|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  |  \_________________________/  |   |   |   |           |   |
  |              |                |   |   |   |           |   |
  | XMIT status bits 6-0 (see below)  |   |   |           |   |
  |                               |   |   |   |           |   |
  1=> protocol error              |   |   |   |           |   |
               network connect <= 0   |   |   |           |   |
                         xmt enable <=1   |   |           |   |
                        initialize busy <=0   |           |   |
                                   xmt busy <=0           |   |
                                          xmt tag bit 1 <=0   |
                                              xmt tag bit 0 <=0
  xmt tags indicate transmitter state:
       00=> msg complete
       01=> data being transmitted
       11=> hdr being transmitted


If protocol error bit is 0, bits 14-8 have this definition:
   15  14  13  12  11  10   9   8
 +---+---+---+---+---+---+---+---+
 | 0 |cpd|wak|icp|pke| de|abt|ife|
 +---+---+---+---+---+---+---+---+
       |   |   |   |   |   |   |
       |   |   |   |   |   |   x=> interface error IFF abt = 1
       |   |   |   |   |   |  (1=>xmit_underrun, 0=>controller_err)
       |   |   |   |   |   1=> packet aborted. check ife
       |   |   |   |   1=> data error (crc_error)
       |   |   |   1=> packet error (pkterr/ackbyte_errbit)
       |   |   1=> icopy
       |   1=> wak
       1=> copied


If protocol error bit is 1, bits 14-8 have this definition:
   15  14  13  12  11  10   9   8
 +---+---+---+---+---+---+---+---+
 | 1 |tmo|syn|ern|frm|akp|abt|ife|
 +---+---+---+---+---+---+---+---+
       |   |   |   |   |   |   x=> interface error IFF abt = 1
       |   |   |   |   |   |   (1=>rcv_overrun, 0=>controller_err)
       |   |   |   |   |   1=> packet aborted. check ife
       |   |   |   |   1=> ackbyte parity error  (ackpe)
       |   |   |   1=> from error (ferr)
       |   |   1=> error rtn (no_return)
       |   1=> sync error
       1=> timeout in Gate Array
```

MISC_CMD (write only) [59400 | 3FF7C00]

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| 0 | 0 | 0 |bpm|nct|td1|td2|lpb| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
              |   |   |   |   1=> digital loopback enable
              |   |   |   1=> txdiag2
              |   |   1=> txdiag1
              |   1=> network connect
              1=> bad packet marking enable
```

Except for diagnostics, txdiag1, txdiag2, and digital loopback enable should be zero. Bad packet marking enable is only operational on the single board version.

RCV_CMD (write only) [59404 | 3FF7C04]

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| 0 | 0 | 0 | 0 |rcv| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
                  |
                  1=> receive enable
```

Set 'rcv' to 0 to abort an enabled receive.

XMIT_CMD (write only) [59402 | 3FF7C02]

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|fen|ten|ine| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
  |   |   1=> initialize enable
  |   1=> transmit enable
  1=> force transmit
```

Force Transmit flag is a modifier to Transmit Enable, not a separate command. Set 'ten' to 0 to abort an enabled transmit in the single board.

XMIT_ADDR / RCV_ADDR / RAM_ADDR

```
  15  14  13  12  11  10   9   8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| a7| a6| a5| a4| a3| a2| a1| a0|a15|a14|a13|a12|a11|a10| a9| a8|
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

These registers are write-only in the 2-board version.

COUNTER REGISTERS

The packet counters:
RCV_HDR_CNT [+800]   RCV_PKT_CNT [+802]   RCV_MAX_CNT [+804]
XMIT_HDR_CNT [+C00]   XMIT_PKT_CNT [+C02]   BAD_PKT_CNT [+C04]
are implemented with Intel 8254 Chips.

To load values into counters, first load CNT_Cx_CNTR with:
$3000 before loading rcv_hdr_cnt
$7000 before loading rcv_pkt_cnt
$B000 before loading rcv_max_cnt
The counters are then written with successive word accesses:
lsb in high byte of word
msb in high byte of word

To read values from counters, first load CNT_Cx_CNTR with:
$C200 before reading rcv_hdr_cnt
$C400 before reading rcv_pkt_cnt
$C800 before reading rcv_max_cnt
or $CE00 to prepare to read all three
The counters are then read with successive word accesses:
status in high byte of word
lsb in high byte of word
msb in high byte of word

Hardware Packet Types:

80	broadcast
40	hw_diag
20	thank you
10	please
8	paging
4	user
2	sw_diag
1	-

RING SOFTWARE STATUS INFO

RING_$SEND_STAT_T and its relationship to software counters

RING_$SEND_STAT_T:		Counted in:	Reported by NETSTAT in:
8000	copied	rd.xmitcnt	Xmit count
4000	wacked	rd.xmit_wack	WACKs
2000	nacked	rd.xmit_nack rd.xmitcnt	NACKs
1000	forced		
800	delay_added	rd.delay_in	Delay swtchd IN/OUT
400	didn't return	rd.xmit_nortn	Xmit Timout
200	ring down	rd.broken = true	
100	overrun	rd.xmit_overrun	Xmit overrun
* 80	bus error	rd.xmit_bus	Xmit Bus error
40	error	rd.xmit_error or rd.xmit_ackpar ring_$unexpected_xmit_stat (sometimes)	Xmit Pkt error Xmit Ack par
* 20	biphase error	rd.xmit_modem and ring_$xmit_biphase	Xmit modem err
10	not connected		
8	looped back		
* 4	esb error	rd.xmit_modem and ring_$xmit_esb	Xmit modem err

rd. = ring_$data

RING_$SEND_STAT_T and its relationship to DN3000 HW status

RING_$SEND_STAT_T:		Generated by Status in XMIT_STS:
8000	copied	NO errors AND cpd AND icp ([tmo_copy_xs,rerr_icopy_xs]) Some error but cpd AND NOT pe (copied by someone) ([tmo_copy_xs] AND NOT [protocol_err_xs])
4000	wacked	NOT pe AND wak (NOT [protocol_err_xs]) AND [serr_wak_xs])
2000	nacked	NOT pe AND NOT icp (NOT [protocol_err_xs]) AND NOT [rerr_icopy_xs])
1000	forced	
800	delay_added	
400	didn't return	pe and (tmo or syn or ern or frm) ([protocol_err_xs] AND THEN [tmo_copy_xs] or [serr_wak_xs] or [rerr_icopy_xs] or [ferr_pkterr_xs])
200	ring down	
100	overrun	abt AND ife ([abort_xs,iferr_xs])
* 80	bus error	
40	error	pe AND akp ([protocol_err_xs,ackpe_derr_xs]) NOT pe AND pke AND icp ([ferr_pkterr_xs,rerr_icopy_xs])
* 20	biphase error	
10	not connected	nct ([discon_xs])
8	looped back	
* 4	esb error	

* Status not defined for DN3000 transmits

SIO [8400 | 3FFB000]

Same as first chip of DN460 (CHAPTER 9), except for the definition of the output port register in which the dtr_b bit has moved. Channel A is the
keyboard, Channel B is the single sio line. The chip is also used to provide timing for memory refresh.

TIMERS [8800 | 3FFAC00]

Same as all previous systems (defined in /os/kins/time/pvt.pas).

APPENDIX A – ASCII CHARACTER SET

oct	hex		dec	oct	hex		dec	oct	hex		dec
00	00	NUL	00	60	30	0	48	140	60		96
01	01	SOH	01	61	31	1	49	141	61	a	97
02	02	STX	02	62	32	2	50	142	62	b	98
03	03	ETX	03	63	33	3	51	143	63	c	99
04	04	EOT	04	64	34	4	52	144	64	d	100
05	05	ENQ	05	65	35	5	53	145	65	e	101
06	06	ACK	06	66	36	6	54	146	66	f	102
07	07	BEL	07	67	37	7	55	147	67	g	103
10	08	BS	08	70	38	8	56	150	68	h	104
11	09	HT	09	71	39	9	57	151	69	i	105
12	0A	NL(LF)	10	72	3A	:	58	152	6A	j	106
13	0B	VT	11	73	3B	;	59	153	6B	k	107
14	0C	FF	12	74	3C	<	60	154	6C	l	108
15	0D	CR	13	75	3D	=	61	155	6D	m	109
16	0E	RRS	14	76	3E	>	62	156	6E	n	110
17	0F	BRS	15	77	3F	?	63	157	6F	o	111
20	10	RCP	16	100	40		64	160	70	p	112
21	11	XON	17	101	41	A	65	161	71	q	113
22	12	HLF	18	102	42	B	66	162	72	r	114
23	13	XOFF	19	103	43	C	67	163	73	s	115
24	14	HLR	20	104	44	D	68	164	74	t	116
25	15	NAK	21	105	45	E	69	165	75	u	117
26	16	SYN	22	106	46	F	70	166	76	v	118
27	17	ETB	23	107	47	G	71	167	77	w	119
30	18	CAN	24	110	48	H	72	170	78	x	120
31	19	EM	25	111	49	I	73	171	79	y	121
32	1A	SUB	26	112	4A	J	74	172	7A	z	122
33	1B	ESC	27	113	4B	K	75	173	7B	{	123
34	1C	FS	28	114	4C	L	76	174	7C	|	124
35	1D	GS	29	115	4D	M	77	175	7D	}	125
36	1E	RS	30	116	4E	N	78	176	7E	~	126
37	1F	US	31	117	4F	O	79	177	7F	DEL	127
40	20	SP	32	120	50	P	80				
41	21	!	33	121	51	Q	81				
42	22	"	34	122	52	R	82				
43	23	#	45	123	53	S	83				
44	24	$	36	124	54	T	84				
45	25	%	37	125	55	U	85				
46	26	&	38	126	56	V	86				
47	27	'	39	127	57	W	87				
50	28	(	40	130	58	X	88				
51	29	)	41	131	59	Y	89				
52	2A	*	42	132	5A	Z	90				
53	2B	+	43	133	5B	[	91				
54	2C	,	44	134	5C	\	92				
55	2D	–	45	135	5D	]	93				
56	2E	.	46	136	5E	^	94				
57	2F	/	47	137	5F	_	95				

APPENDIX B - POWERS OF TWO

2**n	n	hex
1	0	1
2	1	2
4	2	4
8	3	8
16	4	10
32	5	20
64	6	40
128	7	80
256	8	100
512	9	200
1 024	10	400
2 048	11	800
4 096	12	1000
8 192	13	2000
16 384	14	4000
32 768	15	8000
65 536	16	1 0000
131 072	17	2 0000
262 144	18	4 0000
524 288	19	8 0000
1 048 576	20	10 0000
2 097 152	21	20 0000
4 194 304	22	40 0000
8 388 608	23	80 0000
16 777 216	24	100 0000
33 554 432	25	200 0000
67 108 864	26	400 0000
134 217 728	27	800 0000
268 435 456	28	1000 0000
536 870 912	29	2000 0000
1 073 741 824	30	4000 0000
2 147 483 648	31	8000 0000
4 294 967 296	32	1 0000 0000
8 589 934 592	33	2 0000 0000
17 179 869 184	34	4 0000 0000
34 359 738 368	35	8 0000 0000
68 719 476 736	36	10 0000 0000
137 438 953 472	37	20 0000 0000
274 877 906 944	38	40 0000 0000
549 755 813 888	39	80 0000 0000
1 099 511 627 776	40	100 0000 0000
2 199 023 255 552	41	200 0000 0000
4 398 046 511 104	42	400 0000 0000
8 796 093 022 208	43	800 0000 0000
17 592 186 044 416	44	1000 0000 0000
35 184 372 088 832	45	2000 0000 0000
70 368 744 177 664	46	4000 0000 0000
140 737 488 355 328	47	8000 0000 0000
281 474 976 710 656	48	1 0000 0000 0000

READER'S RESPONSE

Apollo Computer uses readers' comments in revising and improving our documents.

Document Title: Apollo DOMAIN Engineering Handbook
Document Revision: 04 Date of Publication: January, 1987
Order Number: 002398

What is the best feature of this manual? ______________________

____

____

Please list any errors, omissions, or problem areas in the manual. (Identify errors by page, section, figure, or table number wherever possible.)

____

____

____

____

What type of user are you?

___ Systems programmer; language ______________________
___ Applications programmer; language __________________
___ Apollo SSR

How often do you use the Apollo system? ______________________

Nature of your work on the Apollo system: ______________________

____
Your name Date

____
Organization

____
Street Address

____
City State Zip/Country
No postage necessary if mailed in the U.S.

cut or fold along dotted line

fold

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 100 CHELMSFORD, MA 01824

POSTAGE WILL BE PAID BY ADDRESSEE

APOLLO COMPUTER, INC.
16 Elizabeth Drive
Chelmsford, MA 01824

Attn.: Software Documentation
Research and Development

fold