DPS8M: GE™ / Honeywell™ / Bull™ 6000-series & DPS-8/M simulator

DPS8M R3.0.0: Omnibus Documentation

The DPS8M Development Team and contributors

DPS8M R3.0.0: 2022-11-23 08:06:43 UTC

DPS8M Omnibus Documentation

- **Version** `git_21f490c3b5a07a0f6ee8ac052c9363da7f0e4b85`
- **Updated** **2022-11-23 08:06:43 UTC**

Version Information

- This documentation is intended for use with the following software:
  - **DPS8M** **R3.0.0**
  - **Prt2PDF** **3.0**
  - **PunUtil** **0.1**
  - **Unifdef** **2120.2.16-dps**
  - **mcmb** **2120.4.16-dps (cmb 3.9.5 + libcmb 3.5.6)**
  - **vmpctool** **2101.14.8-dps**
  - **empty** **0.6.24i-dps**

Generated 2022-11-23 12:39:35 UTC

Contents

DPS8M Omnibus Documentation **i**
- Version Information . . . i

DPS8M Authors and Contributors **1**
- The DPS8M Development Team . . . 1

Introduction **3**
- GE/Honeywell/Bull DPS-8/M Processor . . . 3
  - Processor characteristics . . . 3
  - Functional organization . . . 4
    - Appending unit . . . 4
    - Associative memory assembly . . . 4
    - Control unit . . . 4
    - Operation unit . . . 4
    - Decimal unit . . . 4
  - Modes of operation . . . 5
  - Native data sizes . . . 5
  - Interrupt handling . . . 5
  - Fault handling . . . 5
  - Instruction repertoire . . . 6
  - Registers . . . 6
- The DPS8M Simulator . . . 7
  - Simulator overview . . . 7
  - Supported components . . . 8
  - Unsupported components . . . 8

Obtaining the simulator **9**
- Binary distributions . . . 9
- Source code distributions . . . 10
  - Source kits . . . 10
  - Git repository . . . 10
    - Git mirroring . . . 10

Compiling from source **11**
- General Information . . . 11
- FreeBSD . . . 12
  - FreeBSD prerequisites . . . 12
  - Standard FreeBSD compilation . . . 12
  - Optimized FreeBSD compilation . . . 12
  - blinkenLights2 on FreeBSD . . . 12
  - Additional FreeBSD Notes . . . 13
- NetBSD . . . 13
  - NetBSD prerequisites . . . 13
  - Standard NetBSD compilation . . . 13
  - Optimized NetBSD compilation . . . 14

Compilation using Clang . . . 14
blinkenLights2 on NetBSD . . . 14
OpenBSD . . . 15
OpenBSD prerequisites . . . 15
Standard OpenBSD compilation . . . 15
Optimized OpenBSD compilation . . . 15
Compilation using Clang . . . 16
Additional OpenBSD Notes . . . 16
DragonFly BSD . . . 16
DragonFly BSD prerequisites . . . 16
Standard DragonFly BSD compilation . . . 17
Optimized DragonFly BSD compilation . . . 17
Compiling using Clang . . . 17
Solaris . . . 17
Solaris prerequisites . . . 18
Solaris compilation . . . 18
GCC . . . 18
Clang . . . 18
Oracle Developer Studio . . . 18
OpenIndiana . . . 19
OpenIndiana prerequisites . . . 19
Standard OpenIndiana compilation . . . 19
Compiling using Clang . . . 19
AIX . . . 20
AIX prerequisites . . . 20
AIX compilation . . . 20
IBM Open XL C/C++ for AIX . . . 20
IBM XL C/C++ for AIX . . . 21
GCC . . . 21
Haiku . . . 21
Haiku prerequisites . . . 22
Standard Haiku compilation . . . 22
Compiling using Clang . . . 22
Additional Haiku Notes . . . 22
GNU/Hurd . . . 22
Linux . . . 23
Linux compilers . . . 23
Linux prerequisites . . . 23
Standard Linux compilation . . . 24
Alternative Linux compilation . . . 24
Clang . . . 24
Intel oneAPI DPC++/C++ . . . 24
AMD Optimizing C/C++ . . . 24
AOCC with AMD Optimized CPU Libraries . . . 25
Oracle Developer Studio . . . 25
IBM Open XL C/C++ for Linux . . . 25
IBM XL C/C++ for Linux . . . 25
NVIDIA HPC SDK C Compiler . . . 26
Arm HPC C/C++ Compiler for Linux . . . 26
ACFL with Arm Performance Libraries . . . 26

Linux cross-compilation . . . 27
IBM Advance Toolchain . . . 27
Arm GNU Toolchain . . . 27
Linux/ARMv7-HF . . . 27
Linux/ARM64 . . . 28
Linux/ARM64BE . . . 28
Linaro GNU Toolchain . . . 28
Linux/ARMv7-HF . . . 28
Linux/ARM64 . . . 29
crosstool-NG . . . 29
Linux/RV64 . . . 29
Linux/i686 . . . 29
Linux/ARMv6-HF . . . 30
Linux/PPC64le . . . 30
Additional Linux Notes . . . 30
macOS . . . 31
macOS prerequisites . . . 31
macOS compilation . . . 31
Xcode . . . 31
Intel C/C++ Compiler Classic for macOS . . . 31
macOS cross-compilation . . . 31
ARM64 . . . 32
Intel . . . 32
Universal . . . 32
Windows . . . 33
Cygwin . . . 34
Cygwin prerequisites . . . 34
Standard Cygwin compilation . . . 34
Cygwin-hosted cross-compilation to MinGW . . . 35
Windows i686 . . . 35
Windows x86_64 . . . 35
Unix-hosted cross-compilation to Cygwin . . . 36
MSYS2 . . . 37
Unix-hosted LLVM-MinGW Clang cross-compilation . . . 37
Windows i686 . . . 37
Windows x86_64 . . . 38
Windows ARMv7 . . . 38
Windows ARM64 . . . 38
Unix-hosted MinGW-w64 GCC cross-compilation . . . 39
Windows i686 . . . 39
Windows x86_64 . . . 39

Simulator Command Reference **41**

! . . . 41
ASSERT . . . 41
ATTACH (AT) . . . 42
Switches . . . 42
-n . . . 42
-e . . . 42
-r . . . 42
-q . . . 42
-f . . . 42

AUTOINPUT (AI) . . . 43
AUTOINPUT2 (AI2) . . . 43
BOOT (BO) . . . 44
BURST . . . 44
CABLE (C) . . . 44
UNCABLE (U) . . . 45
CABLE_RIPOUT . . . 46
CABLE_SHOW . . . 46
CABLE DUMP . . . 48
CABLE GRAPH . . . 48
Complete Cabling Graph . . . 49
CPU / SCU / IOM Cabling Graph . . . 50
Storage Cabling Graph . . . 51
Controller Cabling Graph . . . 52
CALL . . . 52
CHECKPOLL . . . 53
CLRAUTOINPUT . . . 53
CLRAUTOINPUT2 . . . 53
CONTINUE (CO) . . . 53
DEFAULT_BASE_SYSTEM . . . 53
DETACH (DET) . . . 53
DO . . . 54
Switches . . . 54
-v . . . 54
-e . . . 54
-o . . . 54
-q . . . 54
ECHO . . . 54
EVALUATE . . . 54
EXIT (QUIT, BYE) . . . 55
FNPSERVERADDRESS . . . 55
FNPSERVERPORT . . . 55
FNPSTART . . . 55
GO . . . 56
GOTO . . . 56
IF . . . 56
Conditional Expressions . . . 56
Simulator State Expressions . . . 57
String Comparison Expressions . . . 57
LOAD (UNLOAD) . . . 58
LUF (NOLUF) . . . 58
MOUNT . . . 58
NEXT (N) . . . 58
ON . . . 58
POLL . . . 58
PROCEED (IGNORE) . . . 58
READY . . . 59
RESET . . . 59
RETURN . . . 59
REWIND . . . 60
RUN (RU) . . . 60

SEGLDR . . . 60
SET . . . 60
Logging . . . 60
Switches . . . 61
−N . . . 61
−B . . . 61
Debug Messages . . . 61
Switches . . . 61
-T . . . 61
-A . . . 61
-R . . . 61
-P . . . 61
-N . . . 61
-D . . . 61
-E . . . 61
Environment Variables . . . 62
Command Status Trap Dispatching . . . 62
Command Execution Display . . . 62
Command Error Status Display . . . 62
Command Output Display . . . 62
Command Prompt . . . 62
Device and Unit Settings . . . 62
CPU Configuration . . . 63
L68 (DPS8M) . . . 63
RESET . . . 63
INITIALIZE . . . 64
INITIALIZEANDCLEAR (IAC) . . . 64
NUNITS . . . 64
KIPS . . . 65
STALL . . . 65
DEBUG (NODEBUG) . . . 66
CONFIG . . . 66
FAULTBASE . . . 67
NUM . . . 67
DATA . . . 67
STOPNUM . . . 68
MODE . . . 68
SPEED . . . 68
PORT . . . 69
ASSIGNMENT . . . 69
INTERLACE . . . 69
ENABLE . . . 70
INIT_ENABLE . . . 70
STORE_SIZE . . . 70
ENABLE_CACHE . . . 70
SDWAM . . . 71
PTWAM . . . 71
DIS_ENABLE . . . 71
STEADY_CLOCK . . . 71
HALT_ON_UNIMPLEMENTED . . . 72
ENABLE_WAM . . . 72

REPORT_FAULTS . . . 72
TRO_ENABLE . . . 73
Y2K . . . 73
DRL_FATAL . . . 73
USEMAP . . . 74
ADDRESS . . . 74
PROM_INSTALLED . . . 74
HEX_MODE_INSTALLED . . . 75
CACHE_INSTALLED . . . 75
CLOCK_SLAVE_INSTALLED . . . 75
ENABLE_EMCALL . . . 75
ISOLTS_MODE . . . 76
NODIS . . . 76
L68_MODE . . . 76
IOM Configuration . . . 77
NUNITS . . . 77
DEBUG (NODEBUG) . . . 77
RESET . . . 77
CONFIG . . . 77
PORT . . . 77
ADDR . . . 78
INTERLACE . . . 78
ENABLE . . . 78
INITENABLE . . . 78
HALFSIZE . . . 78
STORE_SIZE . . . 78
MODEL . . . 78
OS . . . 79
BOOT . . . 79
IOM_BASE . . . 79
MULTIPLEX_BASE . . . 80
TAPECHAN . . . 80
CARDCHAN . . . 80
SCUPORT . . . 80
SCU Configuration . . . 81
DEBUG (NODEBUG) . . . 81
NUNITS . . . 81
RESET . . . 81
CONFIG . . . 81
MODE . . . 81
MASKA . . . 82
MASKB . . . 82
PORT0 . . . 82
PORT1 . . . 82
PORT2 . . . 82
PORT3 . . . 82
PORT4 . . . 82
PORT5 . . . 82
PORT6 . . . 82
PORT7 . . . 83
LWRSTORESIZE . . . 83

CYCLIC . . . 83
NEA . . . 83
ONL . . . 83
INT . . . 83
LW . . . 83
ELAPSED_DAYS . . . 83
STEADY_CLOCK . . . 83
BULLET_TIME . . . 84
Y2K . . . 84
TAPE Configuration . . . 84
DEBUG (NODEBUG) . . . 84
DEFAULT_PATH . . . 84
ADD_PATH . . . 84
CAPACITY_ALL . . . 84
CAPACITY . . . 84
REWIND . . . 84
READY . . . 85
NAME . . . 85
NUNITS . . . 85
MTP Configuration . . . 85
DEBUG (NODEBUG) . . . 85
NAME . . . 85
NUNITS . . . 86
BOOT_DRIVE . . . 86
CONFIG . . . 86
IPC Configuration . . . 86
NAME . . . 86
NUNITS . . . 86
MSP Configuration . . . 87
NAME . . . 87
NUNITS . . . 87
Disk Configuration . . . 87
DEBUG (NODEBUG) . . . 87
TYPE . . . 87
READY . . . 88
NAME . . . 88
NUNITS . . . 88
RDR Configuration . . . 88
DEBUG (NODEBUG) . . . 88
PATH . . . 88
NAME . . . 88
NUNITS . . . 89
PUN Configuration . . . 89
DEBUG (NODEBUG) . . . 89
PATH . . . 89
NAME . . . 89
NUNITS . . . 90
CONFIG . . . 90
LOGCARDS . . . 90
PRT Configuration . . . 90
DEBUG (NODEBUG) . . . 90

PATH . . . 90
MODEL . . . 90
READY . . . 91
NAME . . . 91
NUNITS . . . 91
CONFIG . . . 91
SPLIT . . . 91
FNP Configuration . . . 92
DEBUG (NODEBUG) . . . 92
NUNITS . . . 92
CONFIG . . . 92
MAILBOX . . . 92
IPC_NAME . . . 92
SERVICE . . . 92
OPC Configuration . . . 93
DEBUG (NODEBUG) . . . 93
AUTOINPUT . . . 93
NAME . . . 93
NUNITS . . . 93
CONFIG . . . 93
AUTOACCEPT . . . 94
NOEMPTY . . . 94
ATTN_FLUSH . . . 94
PORT . . . 94
ADDRESS . . . 94
PW . . . 94
URP Configuration . . . 94
DEBUG (NODEBUG) . . . 94
NAME . . . 94
NUNITS . . . 95
SHIFT . . . 95
SHOW . . . 95
BUILDINFO (B) . . . 96
CLOCKS (CL) . . . 96
CONFIGURATION (C) . . . 96
DEFAULT_BASE_SYSTEM (D) . . . 97
DEVICES (DEV) . . . 97
MODIFIERS (M) . . . 97
ON (O) . . . 97
PROM (P) . . . 97
QUEUE (Q) . . . 98
SHOW (S) . . . 98
TIME (T) . . . 98
VERSION (V) . . . 98
CPU . . . 99
CONFIG . . . 99
NUNITS . . . 100
KIPS . . . 100
STALL . . . 100
DEBUG . . . 101

DISK . . . 101
NAME . . . 101
NUNITS . . . 101
TYPE . . . 101
DEBUG . . . 102
FNP . . . 102
CONFIG . . . 102
IPC_NAME . . . 102
NAME . . . 102
NUNITS . . . 103
SERVICE . . . 103
STATUS . . . 103
DEBUG . . . 104
IOM . . . 104
CONFIG . . . 104
NUNITS . . . 105
DEBUG . . . 105
IPC . . . 105
NAME . . . 105
NUNITS . . . 106
MSP . . . 106
NAME . . . 106
NUNITS . . . 106
MTP . . . 106
BOOT_DRIVE . . . 106
NAME . . . 107
NUNITS . . . 107
DEBUG . . . 107
OPC . . . 107
CONFIG . . . 107
ADDRESS . . . 108
AUTOINPUT . . . 108
NAME . . . 108
NUNITS . . . 108
PORT . . . 108
PW . . . 109
DEBUG . . . 109
PRT . . . 109
CONFIG . . . 109
MODEL . . . 109
NAME . . . 110
NUNITS . . . 110
PATH . . . 110
DEBUG . . . 110
PUN . . . 110
CONFIG . . . 111
NAME . . . 111
NUNITS . . . 111
PATH . . . 111
DEBUG . . . 111

RDR . . . 112
NAME . . . 112
NUNITS . . . 112
PATH . . . 112
DEBUG . . . 112
SCU . . . 113
CONFIG . . . 113
NUNITS . . . 113
STATE . . . 113
DEBUG . . . 114
TAPE . . . 114
ADD_PATH . . . 114
CAPACITY . . . 115
DEFAULT_PATH . . . 115
NAME . . . 115
NUNITS . . . 115
DEBUG . . . 116
URP . . . 116
NAME . . . 116
NUNITS . . . 116
DEBUG . . . 116
SKIPBOOT . . . 117
SLOWPOLL . . . 117
STEP (S) . . . 117
Switches . . . 117
−T . . . 117

Simulator Defaults **119**
Default Base System Script . . . 119
Default Base System Configuration . . . 131
Default Base System Cable Dump . . . 137

Legal **143**
DPS8M Omnibus Documentation Licensing Terms . . . 143
DPS8M Omnibus Documentation-specific Content Licenses . . . 143
The DPS8M Documentation License . . . 144
General Attribution License . . . 145
Multics License . . . 146
General Attribution License (Multics Rings Logo) . . . 147
BSD License (Eisvogel) . . . 148
ISC License (Pagebreak) . . . 149
SIL™ Open Font License 1.1 (Source™ Code Pro) . . . 150
SIL™ Open Font License 1.1 (Source™ Sans Pro) . . . 151
DPS8M Software Licensing Terms . . . 152
DPS8M Simulator . . . 152
ICU License . . . 152
FSF All Permissive License . . . 152
Multics License . . . 153
Third-party Software . . . 153
libuv . . . 154
Unifdef . . . 155
UDPLib . . . 156

UtHash . . . 157
Open SIMH . . . 158
PunUtil . . . 159
Prt2PDF . . . 160
decNumber . . . 161
LibTELNET . . . 162
Line History . . . 163
BSD Random . . . 164
AMD LibM . . . 165
empty . . . 166
vmpctool . . . 167
musl libc . . . 168
mcmb . . . 169
novdso . . . 170
Fair CPU Dispatch . . . 171
Multics Software Materials and Documentation . . . 172
Scope of Intended Application . . . 173
Disclaimer of Liability and Endorsement . . . 173

DPS8M Authors and Contributors

The DPS8M Development Team

- ***The DPS8M Development Team*** is, in alphabetical order:
  - Dean S. Anderson,
  - Charles Anthony <charles.unix.pro@gmail.com>,
  - Jeffrey H. Johnson <trnsz@pobox.com>,
  - Jean-Michel Merliot,
  - Michael Mondy <michael.mondy@coffeebird.net>,
  - Harry Reed <doon386@reedclan.org>,
  - Craig Ruff,
  - Eric Swenson <eric@swenson.org>,
  - M——— T———,
  - Juergen Weiss <weiss@uni-mainz.de>,
  - M. Williams (OrangeSquid)
  - *and others …*

Introduction

DPS8M is a simulator of the **36-bit** GE Large Systems / Honeywell / Bull 600/6000-series mainframe computers (Honeywell 6180, Honeywell Series-60 / Level-68, and Honeywell / Bull **DPS-8/M**) descended from the **GE-645** and engineered to support the **Multics** operating system.

GE/Honeywell/Bull DPS-8/M Processor

Processor characteristics

- Hardware-based enforcement of access restrictions
- Seven hierarchical protection rings (`0` … 7)
- Segmentation and paging
- 36- & 72-bit fixed-point integer, fixed-point fraction, and floating point arithmetic
- Big-endian word ordering
- Two's complement numeric representation
- Hexadecimal floating point (*HFP*) option (*range of* ±10 *to the* 153*rd power*)
- Hardware rounding and normalization
- Content-addressable associative memory for intermediate storage of address and control information
- Address modification and appending
- Absolute address computation at execution time
- High-resolution asynchronous alarm timer (512 KHz; 1.953125 µs *precision*)
- Direct memory access (*DMA*) I/O
- Multilevel fault and priority interrupt handling
- Deferred handling of low-priority faults
- Instruction and operand caching (with *selective clear* and *bypass*)

Functional organization

Appending unit

- Controls data input/output to main memory
- Interfaces with caches
- Performs main memory selection and interlace control
- Does address appending and virtual address translation
- Controls fault recognition

Associative memory assembly

- Provides register-based access to pointers to most recently used segments and pages
- Reduces the need for multiple memory accesses (before obtaining an absolute address of an operand or instruction)

Control unit

- Performs address modification
- Controls mode of operation
- Performs interrupt recognition
- Decodes instruction words and indirect words
- Performs timer register loading and decrementing

Operation unit

- Performs fixed- and floating-point binary (*base*-2) arithmetic
- Does shifting and boolean operations

Decimal unit

- Performs decimal (*base*-10) arithmetic
- Decimal number formatting (*e.g.* **COBOL** or **PL/I** "`PIC`")
- Specialized character-string and bit-string operations (*e.g.* **PL/I** "`INDEX`")

Modes of operation

- Three memory addressing modes
  - **absolute** mode
  - **append** mode
  - **BAR** mode
- Two instruction execution modes
  - **normal** mode
  - **privileged** mode

Native data sizes

- **36-bit** native word size
- **4-** and **6-bit** “character” sizes
- **9-bit** byte size
- **18-bit** half-word size
- **72-bit** double-word (*word pair*) size
- **15-bit** segment size (**32,768** *segments*)
- **18-bit** address size (**262,144** *words per segment*)

Interrupt handling

- **32** interrupt cells *per* **SCU**
  - interrupt cells are organized in a numbered priority chain

Fault handling

- **27** fault conditions (*expandable to* 32)
  - **32** fault priority levels
  - **7** fault priority groups

Instruction repertoire

- **547** instructions (*expandable to* 1024)
  - **456** basic instructions (*in* **7** *functional classes*)
    - **181** fixed-point binary arithmetic instructions
    - **85** boolean operation instructions
    - **75** pointer register operation instructions
    - **36** control flow instructions
    - **34** floating-point binary arithmetic instructions
    - **28** privileged instructions
    - **17** miscellaneous instructions
  - **91** Extended Instruction Set (**EIS**) instructions
    - **62** single-word and **29** multi-word instructions, *operating on*:
      - **4-**, **6-**, and **9-bit** alphanumeric strings,
      - **4-** and **9-bit** numeric strings, *and*
      - bit strings
    - **21** opcodes for moving, comparison, scanning, conversion, and translation
    - **20** opcodes for loading, storing, and modifying address pointers and lengths
    - **17** "*micro-operations*" for control of string move and edit operations
    - **8** opcodes for decimal arithmetic

Registers

- Two **36-bit** accumulator registers (**A** & **Q**)
- One **72-bit** accumulator-quotient register (**AQ**)
- Eight **18-bit** index registers (**X0** … **X7**)
- One **8-bit** exponent register (**E**)
- One **80-bit** exponent-accumulator-quotient register (**EAQ**)
- One **14-bit** indicator register (**IR**)
- One **18-bit** base address register (**BAR**)
- One **27-bit** timer register (**TR**)
- One **3-bit** ring alarm register (**RALR**)
- Eight **42-bit** pointer registers (**PR0** … **PR7**)
- Eight **24-bit** address registers (**AR0** … **AR7**)
- One **37-bit** procedure pointer register (**PPR**)
- One **42-bit** temporary pointer register (**TPR**)
- One **51-bit** descriptor segment base register (**DSBR**)
- One **35-bit** fault register (**FR**)
- One **33-bit** mode register (**MR**)
- One **28-bit** cache mode register (**CMR**)
- Sixteen **51-bit** page table word associative memory (**PTWAM**) registers
- Sixteen **108-bit** segment descriptor word associative memory (**SDWAM**) registers
- Sixteen **72-bit** control unit (**CU**) history registers
- Sixteen **72-bit** operations unit (**OU**) history registers
- Sixteen **72-bit** decimal unit (**DU**) history registers
- Sixteen **72-bit** appending unit (**APU**) history registers
- Five **36-bit** configuration switch data (**CSD**) registers
- One **288-bit** control unit data (**CUD**) register
- One **288-bit** decimal unit data (**DUD**) register

The DPS8M Simulator

Simulator overview

- **DPS8M** is **open source software** developed by **The DPS8M Development Team** and **many contributors**.
- **DPS8M** supports most operating systems conforming to *IEEE Std 1003.1-2008* (**POSIX.1-2008**), many C compilers conforming to *ISO/IEC 9899:1999* (**C99**), and numerous hardware architectures.
  - Operating systems supported are **AIX**, **FreeBSD**, **NetBSD**, **OpenBSD**, **DragonFly BSD**, **Haiku**, **GNU/Hurd**, **illumos OpenIndiana**, **Linux**, **macOS**, **Solaris**, and **Windows**.
  - C compilers supported are **Clang**, **LLVM-MinGW**, AMD Optimizing C/C++ (**AOCC**), Arm C/C++ Compiler (**ARMClang**), GNU C (**GCC**), IBM Advance Toolchain, IBM XL C/C++ (**XLC**), IBM Open XL C/C++ (**IBMClang**), Intel oneAPI DPC++/C++ (**ICX**), Intel C++ Compiler Classic for macOS (**ICC**), NVIDIA HPC SDK C Compiler (**NVC**), and Oracle Developer Studio (**SunCC**).
  - Hardware architectures supported are **Intel x86** (i686, x86_64), **ARM** (ARMv6, ARMv7, ARM64), **PowerPC** (PPC, PPC64, PPC64le), **RISC-V** (RV64), and **m68k** (68020+).
- Various releases of **DPS8M** have been ported to **embedded systems**, **cell phones**, **tablets**, **handheld gaming consoles**, **wireless routers**, and even modern **mainframes**. A full-featured port should be possible for any 32- or 64-bit platform with appropriate hardware atomic operations and able to support the **libuv** asynchronous I/O library.
- The simulator is distributed as an **easy-to-build** source code **distribution** (buildable simply via **`make`** on most systems) or as ready-to-run **pre-compiled binaries** for several popular platforms.
- **DPS8M** development is hosted courtesy of **GitLab**, providing version control, issue tracking, and CI/CD services.
- Static application security testing by **PVS-Studio**, a static analysis tool for code quality, security, and safety.

Supported components

DPS8M simulates not only the **DPS-8/M** CPU, but an *entire* Honeywell / Bull ***Distributed Processing System*** mainframe.

- *{6×}* Central Processing Units (**CPU**)
- *{8×}* Front-End Network Processors (**FNP**)
- *{4×}* System Control Units (**SCU**)
- *{2×}* Operator Consoles (**OPC**)
- *{2×}* Input/Output Multiplexers (**IOM**)
- *{10×}* Unit Record Processors (**URP**)
- Integrated Peripheral Controllers (**IPC**)
- Magnetic Tape Processors (**MTP**)
- Mass Storage Processors (**MSP**)
- Microprogrammed Peripheral Controllers (**MPC**)
- Socket Controllers (**SKC**)
- ABSI Interfaces
- Tape Drives
- Disk Storage Units
- Printers
- Card Readers
- Card Punches
- DIA cabling

Unsupported components

- Diagnostic Processor Unit (**DPU**)
- Chaosnet Interfaces (**MGP**)
- Information Multiplexer Units (**IMU**)
- Maintenance Channel Adapter (**MCA**)
- Remote Maintenance Interface (**RMI**)
- Multidrop Interfaces (**MDI**)
- New System Architecture Extensions (**VU**)
- DPS-6 (*Level-6*) Satellite Processors
- DIA Port Expanders

Obtaining the simulator

Official releases of the simulator are available from:

The DPS8M Simulator Homepage	https://dps8m.gitlab.io/	*Binary distributions and source kits*
GitLab DPS8M `git` Repository	https://gitlab.com/dps8m/dps8m/	*Developmental source code*

Binary distributions

- **The DPS8M Simulator Homepage** offers **binary releases** for several popular operating systems, including:
  - **AIX**
  - **Linux**
  - **macOS**
  - **FreeBSD**
  - **NetBSD**
  - **OpenBSD**
  - **Windows**
  - **Solaris**
  - **illumos**
  - **Haiku**
  - *and others …*

- Executables are provided for many hardware architectures, including:
  - **POWER** (*e.g.* IBM Power Systems, Raptor Talos™, Freescale PowerPC …)
  - **ARM** (*e.g.* Raspberry Pi, Pine64, Orange Pi …)
  - **ARM64** / **AArch64** (*e.g.* Apple M1, Fujitsu A64FX, Arm Neoverse N1 …)
  - **RISC-V** (*e.g.* SiFive HiFive, AndesCore™ AX25/AX27 …)
  - **OpenRISC** (*e.g.* OpenRISC OR1200, mor1kx, marocchino …)
  - **x86** / **ix86**
  - **x86_64** / **AMD64**
  - *and others …*

NOTE: Other operating systems and hardware architectures are supported when building the from source code.

Source code distributions

Source kits

The **DPS8M Simulator Homepage** offers downloadable **source kit distributions** for released versions of the simulator, bleeding edge snapshots, and historical releases.

- **The DPS8M Development Team** recommends most new users who wish to build from source code download a source kit distribution from the **DPS8M Releases** section of **The DPS8M Simulator Homepage**.
- Advanced users and developers may wish to clone the **git** repository and work with the `master` branch.

Git repository

DPS8M is development is coordinated using the **git** distributed version control system, hosted courtesy of *GitLab*, providing project management tools, wiki and web hosting, issue tracking, and CI/CD (*continuous integration*/*continuous delivery*) services.

Most development takes place on branches in the **git** repository, which are merged into the `master` branch after GitLab CI/CD verification and other manual testing. Simulator releases are cut from the `master` branch. The head of the **git** `master` branch is the version of the simulator used by most of the development team, and *should* be stable enough for daily usage, although regressions and new bugs may be encountered periodically.

- Clone the repository via HTTPS:

```
git clone https://gitlab.com/dps8m/dps8m.git
```

- Users with a *GitLab* account may clone the repository via SSH:

```
git clone git@gitlab.com:/dps8m/dps8m.git
```

  - After cloning the repository, it can be updated by executing the following command from the repository directory:

```
git pull
```

Git mirroring

Users who wish to **mirror** the **git repository** for backup or archival purposes (*i.e.* copying **all** remote-tracking branches, tags, notes, references, etc.) should use the mirroring functionality of `git`:

- Mirror the repository via HTTPS:

```
git clone --mirror https://gitlab.com/dps8m/dps8m.git
```

- Users with a *GitLab* account may mirror the repository via SSH:

```
git clone --mirror git@gitlab.com:/dps8m/dps8m.git
```

  - After cloning the repository, it can be updated, including removing local branches when they are removed upstream, by executing the following command from the repository directory:

```
git remote update --prune
```

Compiling from source

The simulator is distributed in various forms, including an easy-to-build **source code distribution**, which can be built simply via **`make`** on *most* systems.

General Information

- For optimal performance, building the simulator from source is highly recommended.
- Building on a **64-bit** platform is **strongly encouraged for optimal performance**.
- **The DPS8M Development Team** recommends most users download a **source kit distribution**.
  - A source kit requires approximately **8 MiB** of storage space to decompress and **45 MiB** to build.
- Advanced users may prefer to clone the **git repository** which contains additional tools not required for simulator operation, but useful to developers.
  - The **git repository** requires approximately **175 MiB** of storage space to clone and **250 MiB** to build.

The following sections document ***only*** platform-specific variations, and are **not** intended to be a general or exhaustive reference.

FreeBSD

- Ensure you are running a supported release of **FreeBSD** on a supported platform.
  - The current release versions of **FreeBSD**/**amd64** and **FreeBSD**/**arm64** are regularly tested by **The DPS8M Development Team**.

FreeBSD prerequisites

Install the required prerequisites (using FreeBSD Packages or Ports):

- Using FreeBSD Packages (as *root*):

```
pkg install gmake libuv
```

- Using FreeBSD Ports (as *root*):

```
cd /usr/ports/devel/gmake/ && make install clean
cd /usr/ports/devel/libuv/ && make install clean
```

Standard FreeBSD compilation

- Build the simulator (*standard build*) from the top-level source directory (using **GNU Make**):

```
gmake
```

Optimized FreeBSD compilation

- **FreeBSD** provides the **Clang** compiler as part of the base system. While *sufficient* to build the simulator, we recommend that version 10 or later of the **GNU C** (`gcc`) compiler be used for optimal performance.
- At the time of writing, **GCC 12** is available for **FreeBSD** systems and is the version of GCC currently recommended by **The DPS8M Development Team**.

  It can be installed via FreeBSD Packages or Ports:
  - Using FreeBSD Packages (as *root*):

```
pkg install gcc12
```

  - Using FreeBSD Ports (as *root*):

```
cd /usr/ports/lang/gcc12/ && make install clean
```

- Build the simulator from the top-level source directory (using **GNU Make**):

```
env CC="gcc12" LDFLAGS="-Wl,-rpath=/usr/local/lib/gcc12" gmake
```

blinkenLights2 on FreeBSD

- To build the **blinkenLights2** front-panel display, install it's prerequisites via FreeBSD Packages or Ports:
  - Using FreeBSD Packages (as *root*):

```
pkg install pkgconf gtk3
```

  - Using FreeBSD Ports (as *root*):

```
cd /usr/ports/devel/pkgconf/ && make install clean
cd /usr/ports/x11-toolkits/gtk30/ && make install clean
```

- Build **blinkenLights2** from the top-level source directory (using **GNU Make**):

```
gmake blinkenLights2
```

Additional FreeBSD Notes

- When running on **FreeBSD**, **DPS8M** utilizes **FreeBSD-specific** atomic operations.
- The **FreeBSD-specific** `atomic_testandset_64` operation is currently not implemented in all versions of **FreeBSD** or on all platforms **FreeBSD** supports (*e.g.* **powerpc64**, or **arm64** prior to **13.0-RELEASE**).

  If you are unable to build the simulator because this atomic operation is unimplemented on your platform, specify `ATOMICS="GNU"` as an argument to `gmake`, or export this value in the shell environment before compiling.

NetBSD

- Ensure you are running a supported release of **NetBSD** on a supported platform.
  - **NetBSD**/**amd64** is regularly tested by **The DPS8M Development Team**.
- **DPS8M** is fully supported on **NetBSD 9.2** or later.

NetBSD prerequisites

Install the required prerequisites (using NetBSD Packages or pkgsrc):

- Using NetBSD Packages (as *root*):

```
pkgin in gmake libuv
```

- Using pkgsrc (as *root*):

```
cd /usr/pkgsrc/devel/gmake/ && make install clean
cd /usr/pkgsrc/devel/libuv/ && make install clean
```

Standard NetBSD compilation

- Build the simulator (*standard build*) from the top-level source directory (using **GNU Make**):

```
gmake
```

Optimized NetBSD compilation

- **NetBSD** provides an older version of **GCC** (or **Clang**) as part of the base system (depending on the platform). While *sufficient* to build the simulator, we recommend that version 10 or later of the **GNU C** (`gcc`) compiler be used for optimal performance.
- At the time of writing, **GCC 12** is available for **NetBSD** systems and is the version of GCC currently recommended by **The DPS8M Development Team**.

  It can be installed via Packages or pkgsrc.

  - Using NetBSD Packages (as *root*):

    ```
    pkgin in gcc12
    ```

  - Using pkgsrc (as *root*):

    ```
    cd /usr/pkgsrc/lang/gcc12/ && make install clean
    ```

- Build the simulator from the top-level source directory (using **GNU Make**):

```
env CC="/usr/pkg/gcc12/bin/gcc" LDFLAGS="Wl,-rpath=/usr/pkg/gcc12/lib" gmake
```

Compilation using Clang

- **GCC** is recommended for optimal performance, but compilation using **Clang** (and linking using **LLD**, the LLVM linker) is supported.

  Both **Clang** and **LLD** can be installed via Packages or pkgsrc.

  - Using NetBSD Packages (as *root*):

    ```
    pkgin in clang lld
    ```

  - Using pkgsrc (as *root*):

    ```
    cd /usr/pkgsrc/lang/clang/ && make install clean
    cd /usr/pkgsrc/devel/lld/ && make install clean
    ```

- Build the simulator from the top-level source directory (using **GNU Make**):

```
env CC="clang" \
  LDFLAGS="-L/usr/lib -L/usr/pkg/lib -fuse-ld="$(command -v ld.lld)" gmake
```

blinkenLights2 on NetBSD

- To build the **blinkenLights2** front-panel display, install it's prerequisites via NetBSD Packages or pkgsrc:

  - Using NetBSD Packages (as *root*):

    ```
    pkg install pkgconf gtk3+
    ```

  - Using pkgsrc (as *root*):

    ```
    cd /usr/pkgsrc/devel/pkgconf/ && make install clean
    cd /usr/pkgsrc/x11/gtk3/ && make install clean
    ```

- Build **blinkenLights2** from the top-level source directory (using **GNU Make**):

```
gmake blinkenLights2
```

OpenBSD

- Ensure you are running an up-to-date and supported release of **OpenBSD** on a supported platform.
  - **OpenBSD**/**amd64** and **OpenBSD**/**arm64** are regularly tested by **The DPS8M Development Team**.
- The following instructions were verified using **OpenBSD 7.1** on **amd64** and **arm64**.

OpenBSD prerequisites

Install the required prerequisites (using OpenBSD Packages or Ports):

- Using OpenBSD Packages (as *root*):

```
pkg_add gmake libuv
```

- Using OpenBSD Ports (as *root*):

```
cd /usr/ports/devel/gmake/ && make install clean
cd /usr/ports/devel/libuv/ && make install clean
```

Standard OpenBSD compilation

- Build the simulator (*standard build*) from the top-level source directory (using **GNU Make**):

```
gmake
```

Optimized OpenBSD compilation

- **OpenBSD** provides an older version of **GCC** (or **Clang**) as part of the base system (depending on the platform). While *sufficient* to build the simulator, we recommend that a recent version of the **GNU assembler** (`gas`) and version 10 or later of the **GNU C** (`gcc`) compiler be used for optimal performance.
- At the time of writing, appropriate versions of the **GNU assembler** and **GNU C** (version **11**) are available for **OpenBSD**. (In addition, **LLD**, the LLVM linker, may be required.) These tools have been tested and are highly recommended by **The DPS8M Development Team**.

  They can be installed via OpenBSD Packages or Ports:

  - Using OpenBSD Packages (as *root*):

```
pkg_add gas gcc
```

  - Using OpenBSD Ports (as *root*):

```
cd /usr/ports/devel/gas/ && make install clean
cd /usr/ports/lang/gcc/11/ && make install clean
```

- LLVM **LLD** 13.0.0 or later is recommended for linking, even when using **GCC 11** for compilation. **LLD** is the default linker on ***most*** (*but not all*) supported OpenBSD platforms. To determine the linker in use, examine the output of '`ld --version`'.

  If the linker identifies itself by a name *other* than **LLD** (*e.g.* "**`GNU ld`**" or similar), **LLD** must be installed via OpenBSD Packages or Ports.

  - Using OpenBSD Packages (as *root*):

    ```
    pkg_add llvm
    ```

  - Using OpenBSD Ports (as *root*):

    ```
    cd /usr/ports/devel/llvm/ && make install clean
    ```

- Configure **GCC** to execute the correct assembler and/or linker:

  ```
  mkdir -p ~/.override
  test -x /usr/local/bin/gas && ln -fs /usr/local/bin/gas ~/.override/as
  test -x /usr/local/bin/lld && ln -fs /usr/local/bin/lld ~/.override/ld
  ```

- Build the simulator from the top-level source directory (using **GNU Make**):

  ```
  env CC="egcc" CFLAGS="-B ~/.override" gmake
  ```

Compilation using Clang

- **GCC** is recommended for optimal performance, but compilation using **Clang** is supported.
- A version of **Clang** newer than the base system version may be available via the '**`llvm`**' package or port.
- Once installed, it can be used for compilation by setting `CC=/usr/local/bin/clang`.

Additional OpenBSD Notes

- At the time of writing, **OpenBSD**/**luna88k** has not been tested.

  It should be possible to build the simulator for this architecture using the `gcc` compiler provided by the base system and specifying the `NO_LTO=1` build option.

DragonFly BSD

- At the time of writing, **DragonFly BSD 6.2.2** was current and used to verify the following instructions.

DragonFly BSD prerequisites

- Install the required prerequisites (using DragonFly BSD DPorts):

  ```
  pkg install gmake libuv
  ```

Standard DragonFly BSD compilation

- Build the simulator (*standard build*) from the top-level source directory (using **GNU Make**):

```
env CFLAGS="-I/usr/local/include"  \
     LDFLAGS="-L/usr/local/lib"    \
     ATOMICS="GNU"                 \
  gmake
```

Optimized DragonFly BSD compilation

- **DragonFly BSD** provides an older version of **GCC** as part of the base system. While this compiler is *sufficient* to build the simulator, we recommend that version 10 or later of the **GNU C** (`gcc`) compiler be used for optimal performance.
- At the time of writing, **GCC 11** is available for DragonFly BSD and recommended by **The DPS8M Development Team**.
  - **GCC 11** may be installed using DragonFly BSD DPorts:

```
pkg install gcc11
```

- Build the simulator from the top-level source directory (using **GNU Make**):

```
env CC="gcc11" CFLAGS="-I/usr/local/include"                      \
    LDFLAGS="-L/usr/local/lib -Wl,-rpath=/usr/local/lib/gcc11"    \
    ATOMICS="GNU"                                                 \
  gmake
```

Compiling using Clang

- **GCC** is recommended for optimal performance, but compilation using **Clang** is supported.
- At the time of writing, **Clang 14** is available for DragonFly BSD and recommended by **The DPS8M Development Team**.
  - **Clang 14** may be installed using DragonFly BSD DPorts:

```
pkg install llvm14
```

- Build the simulator from the top-level source directory (using **GNU Make**):

```
env CC="clang14" CFLAGS="-I/usr/include -I/usr/local/include"  \
    LDFLAGS="-L/usr/lib -L/usr/local/lib -fuse-ld=lld"         \
    ATOMICS="GNU"                                              \
  gmake
```

Solaris

- Ensure your **Solaris** installation is reasonably current. **Oracle Solaris 11.4 SRU42** or later is recommended.
- The simulator can be built on **Solaris** using the **GCC**, **Clang**, and **Oracle Developer Studio** compilers.
  - **GCC 11** is the recommended compiler for optimal performance on all Intel-based **Solaris** systems.
    - **GCC 11** can be installed from the standard IPS repository via '**`pkg install gcc-11`**'.
  - Link-time optimization (*LTO*) is supported ***only*** when building with **GCC** version 10 or later.

    * The NO_LTO=1 build option should be specified when using earlier versions of the **GCC** compiler.
  - Building with **Clang 11** or later is also supported (*but not recommended due to lack of LTO support*).
    * **Clang 11** can be installed from the standard IPS repository via '**pkg install clang@11 llvm@11**'.
  - Building with the **Oracle Developer Studio 12.6** (suncc) compiler is also supported.

- Note that building for **Solaris** using the **Oracle Developer Studio** compiler currently requires a non-trivial amount of CFLAGS to be specified. This will be simplified in a future release of the simulator.

Solaris prerequisites

- Install the required prerequisites from the standard IPS repository (as *root*):

```
pkg install gnu-make gnu-binutils gnu-sed gnu-grep gnu-tar gawk \
  gnu-coreutils pkg-config libtool autoconf automake wget
```

Solaris compilation

The following commands will download and build a local static **libuv** before compiling the simulator.

If a site-provided **libuv** library has been installed (in the "**/usr/local**" prefix), the **libuvrel** stage of the build may be omitted.

Build **libuv** and the simulator from the top-level source directory (using **GNU Make**):

GCC

- Build using **GCC**:
  - Build libuv:

```
env TAR="gtar" TR="gtr" CC="gcc" gmake libuvrel
```

  - Build the simulator:

```
env TR="gtr" CC="gcc" gmake
```

Clang

- Build using **Clang**:
  - Build libuv:

```
env TAR="gtar" NO_LTO=1 TR="gtr" CC="clang" gmake libuvrel
```

  - Build the simulator:

```
env NO_LTO=1 TR="gtr" CC="clang" gmake
```

Oracle Developer Studio

- Build using **Oracle Developer Studio 12.6**:

- Build `libuv`:

```
env TAR="gtar" NO_LTO=1 SUNPRO=1 NEED_128=1 TR="gtr" CSTD="c11"          \
    CFLAGS="-DNO_C_ELLIPSIS -Qy -xO5 -m64 -xlibmil -xCC -mt -xlibmopt  \
    -fno-semantic-interposition -xprefetch=auto -xprefetch_level=3"    \
    CC="/opt/developerstudio12.6/bin/suncc"                            \
  gmake libuvrel
```

- Build the simulator:

```
env NO_LTO=1 SUNPRO=1 NEED_128=1 TR="gtr" CSTD="c11"                   \
    CFLAGS="-DNO_C_ELLIPSIS -Qy -xO5 -m64 -xlibmil -xCC -mt -xlibmopt  \
    -fno-semantic-interposition -xprefetch=auto -xprefetch_level=3"    \
    CC="/opt/developerstudio12.6/bin/suncc"                            \
  gmake
```

OpenIndiana

- Ensure your **OpenIndiana** installation is up-to-date.
  - **OpenIndiana** *Hipster* **2022-08-26** was used to verify these instructions.
- **GCC 11** is currently the recommended compiler for optimal performance.
  - **GCC 11** can be installed from the standard IPS repository via '**`pkg install gcc-11`**'.
  - Link-time optimization (*LTO*) is supported ***only*** when building with **GCC** version 10 or later.
  - The `NO_LTO=1` build option should be specified when using earlier versions of the **GCC** compiler.
- Building with **Clang 13** or later is also supported (*but not recommended due to lack of LTO support*).
  - **Clang 13** can be installed from the standard IPS repository via '**`pkg install clang-13`**'.

OpenIndiana prerequisites

- Install the required prerequisites from the standard IPS repository (as *root*):

```
pkg install gnu-make libuv gnu-binutils gnu-coreutils
```

Standard OpenIndiana compilation

- Build the simulator from the top-level source directory (using **GNU Make** and **GCC 11**):

```
env CC="gcc-11" gmake
```

Compiling using Clang

- Build the simulator from the top-level source directory (using **GNU Make** and **Clang**):

```
env NO_LTO=1 CC="clang-13" gmake
```

AIX

- Ensure you are running a supported release of **IBM AIX®** on a supported platform.
  - **AIX 7.2** and **7.3** on **POWER8®** and **POWER9™** are regularly tested by **The DPS8M Development Team**.
- The simulator can be built for **64-bit AIX** using **IBM XL C/C++ for AIX** (`xlc`), **IBM Open XL C/C++ for AIX** (`ibm-clang`), or **GNU C** (`gcc`). **The DPS8M Development Team** recommends building with **IBM Open XL C/C++ V17.1** (or later) or **GCC 10** (or later) for optimal performance.
- **IBM Open XL C/C++ for AIX V17.1** (*5725-C72, 5765-J18*) is the *minimum* recommended version of the **Open XL C/C++** compiler on **POWER8**, **POWER9**, and **Power10** systems.
- **IBM XL C/C++ for AIX V16.1 Service Pack 10** (*IJ36514*) is the *minimum* recommended version of the **IBM XL C/C++** compiler on **POWER8** and **POWER9** systems.
- Verify via **IBM Preventive Service Planning** that you are using the latest available **XL** compiler PTF for your **IBM AIX** OS level.
- When building the simulator using **GNU C**, it recommended to use **GCC 10** or later for optimal performance.
  - **GCC 10** can be installed from the IBM AIX® Toolbox for Open Source Software repository.
- Note that building for **AIX** currently requires a non-trivial number of options to be specified *after* the `gmake` command, which overrides various build defaults appropriate for **Linux**, **macOS**, and **BSD** systems, but not IBM **AIX**. This will be simplified in a future release of the simulator.

AIX prerequisites

- Install the required prerequisites from the IBM AIX® Toolbox for Open Source Software repository (as *root*):

```
/opt/freeware/bin/dnf install sed gmake libuv libuv-devel popt coreutils \
  gawk compat-getopt compat-getopt-devel
```

- *Optionally* install **GCC 10** from the IBM AIX® Toolbox for Open Source Software repository (as *root*):

```
/opt/freeware/bin/dnf install gcc gcc10
```

AIX compilation

Build the simulator from the top-level source directory (using **GNU Make**):

IBM Open XL C/C++ for AIX

- Using **IBM Open XL C/C++ for AIX V17.1.0**:

```
env PATH="/opt/freeware/bin:${PATH}"                                    \
    CC="/opt/IBM/openxlC/17.1.0/bin/ibm-clang_r"                        \
    ATOMICS="AIX"                                                       \
    AWK="gawk"                                                          \
    OBJECT_MODE=64                                                      \
    NEED_128=1                                                          \
  gmake PULIBS="-lpopt"                                                 \
        LDFLAGS="-L/opt/freeware/lib -L/usr/local/lib -flto=auto -b64"  \
        LIBS="-lpthread -luv -lbsd -lm"                                 \
        CFLAGS="-flto=auto -I/opt/freeware/include -I/usr/local/include \
                -I../simh -I../decNumber -DUSE_FLOCK=1 -DUSE_FCNTL=1    \
                -DHAVE_POPT=1 -DNEED_128=1 -DAIX_ATOMICS=1 -m64         \
                -DLOCKLESS=1 -D_ALL_SOURCE -D_GNU_SOURCE -O3            \
                -U__STRICT_POSIX__ -fno-strict-aliasing -mcpu=power8"
```

- When building on IBM **POWER9** (or **Power10**) systems, '`-mcpu=power9`' (*or* '`-mcpu=power10`') should replace '`-mcpu=power8`' in the above compiler invocation.
- Refer to the **IBM Open XL C/C++ for AIX V17.1.0 documentation** for additional information.

IBM XL C/C++ for AIX

- Using **IBM XL C/C++ for AIX V16.1.0**:

```
env PATH="/opt/freeware/bin:${PATH}"                                    \
    ATOMICS="AIX"                                                       \
    AWK="gawk"                                                          \
    NO_LTO=1                                                            \
    OBJECT_MODE=64                                                      \
  gmake CC="/opt/IBM/xlC/16.1.0/bin/xlc_r"                              \
        NEED_128=1                                                      \
        USE_POPT=1                                                      \
        PULIBS="-lpopt"                                                 \
        LDFLAGS="-L/opt/freeware/lib -L/usr/local/lib -b64"             \
        LIBS="-luv -lbsd -lm"                                           \
        CFLAGS="-O3 -qhot -qarch=pwr8 -qalign=natural -qtls -DUSE_POPT=1  \
                -DUSE_FLOCK=1 -DUSE_FCNTL=1 -DAIX_ATOMICS=1 -DNEED_128=1  \
                -DLOCKLESS=1 -I/opt/freeware/include -I../simh          \
                -I../decNumber -I/usr/local/include -D_GNU_SOURCE       \
                -D_ALL_SOURCE -U__STRICT_POSIX__"
```

- When building on **POWER9** systems, '`-qarch=pwr9`' should replace '`-qarch=pwr8`' in the above compiler invocation.
- Compilation using higher optimization levels (*i.e.* '`-O4`' or '`-O5`' replacing '`-O3 -qhot -qarch=pwr8`') and/or enabling automatic parallelization (*i.e.* '`-qsmp`') is possible, but the resulting binaries have *not* been benchmarked or extensively tested by **The DPS8M Development Team**.
- Refer to the **IBM XL C/C++ for AIX V16.1 Optimization and Tuning Guide** for additional information.

GCC

- Using **GCC 10**:

```
env PATH="/opt/freeware/bin:${PATH}" CC="gcc-10" \
  ATOMICS="AIX" NO_LTO=1 gmake
```

Haiku

- Ensure you are running a recent release of **Haiku** on a supported **64-bit** platform.
  - Use '**SoftwareUpdater**' application to ensure your **Haiku** installation is up-to-date.
- **The DPS8M Development Team** regularly tests the simulator using the nightly **Haiku x86_64** snapshots.
  - **Haiku x86_64** (**`hrev56490`**) was used to verify the following instructions.

Haiku prerequisites

The default **Haiku** installation includes the required header files, the recommended compiler (at the time of writing, **GCC 11**), and most of the necessary development utilities (*i.e.* **GNU Make**) required to build **DPS8M**. The remaining prerequisites are available via the standard package management tools.

- Install the required prerequisites (from *HaikuPorts* using the '**HaikuDepot**' application):
  - `libuv`
  - `libuv_devel`
  - `getconf`

Standard Haiku compilation

- Build the simulator (*standard build*) from the top-level source directory (using **GNU Make**):

```
make
```

Compiling using Clang

- Building with **GCC** is strongly recommended for optimal performance, but compilation using **Clang** is also supported (*although not recommended, due to the lack of support for LTO optimization*).
  - At the time of writing, **Clang 12** is available from *HaikuPorts* (as the '**`llvm12_clang`**' and '**`llvm12_lld`**' packages), installable using the '**HaikuDepot**' application.
- Build the simulator from the top-level source directory (using **GNU Make**):

```
env CSTD="c11" CC="clang" CFLAGS="-fPIC" NO_LTO=1 make
```

Additional Haiku Notes

- **Haiku** on **32-bit** platforms (*i.e.* **x86**, **x86_gcc2**) is **not** officially supported at this time.
  - Support for **Haiku** on **32-bit** platforms is planned for a future release of the simulator.

GNU/Hurd

- **DPS8M** is supported on **GNU/Hurd** when using **Debian GNU/Hurd 2021** (or later).
- **GCC 11** (or later) is the recommended compiler for optimal performance.
  - Compilation is also supported using **Clang 11** or later.
- Building on **GNU/Hurd** should be essentially the same as building on **Debian GNU/Linux**.

Linux

- Most major **Linux** distributions using the **GNU C Library**, **Bionic**, **uClibc-ng**, and **musl-libc** are supported.
  - **Debian GNU/Linux** and derivatives (**Raspberry Pi OS**), **Red Hat** variants (**Fedora**, **CentOS Stream**, **RHEL**) and compatibles (**AlmaLinux**, **Amazon Linux**, **Oracle Linux**), **Alpine**, **SUSE** (**SLES**, **OpenSUSE**), **Void**, and **Ubuntu** are regularly tested on **Intel**, **ARM**, and **POWER** systems.

Linux compilers

- **GCC 12** or later is recommended for optimal performance on most architectures including **Intel** and **ARM**.
  - **The DPS8M Development Team** regularly tests and supports a wide range of Linux compilers, including **Clang**, AMD Optimizing C/C++ (**AOCC**), Arm C/C++ Compiler (**ARMClang**), GNU C (**GCC**) (*version* **8**+), IBM Advance Toolchain for Linux, IBM XL C/C++ for Linux (**XLC**), IBM Open XL C/C++ for Linux (**IBMClang**), Intel oneAPI DPC++/C++ (**ICX**), NVIDIA HPC SDK C Compiler (**NVC**), and Oracle Developer Studio (**SunCC**).
  - **Red Hat** offers the **Red Hat Developer Toolset** for **Red Hat Enterprise Linux** and **CentOS Stream**, which provides up-to-date versions of **GCC** on a rapid release cycle, with *full support*.
    - The *Toolset* packages are also included in various downstream distributions such as **AlmaLinux**. These tools are regularly tested and highly recommended by **The DPS8M Development Team**. Check your distribution packager manager (*i.e.* '`dnf search`') for packages named '`gcc-toolset-12`' (or similar).
  - **Canonical** similarly offers two **Ubuntu Toolchain PPAs**, one providing **GCC** updates for release branches, and the other providing new **GCC** versions for both current and **LTS** releases, maintained by the Ubuntu Toolchain team.
    - For example, at the time of writing, Ubuntu 20.04 LTS ships **GCC 9.3** and **GCC 10**, and the **Toolchain PPAs** ship **GCC 9.4** and **GCC 11**. Although these packages are *not supported* by Canonical, they are regularly and successfully used by **The DPS8M Development Team**.
  - **Intel® C++ Compiler Classic** (**ICC**) for Linux is ***no longer supported*** for building **DPS8M** (*as of* ***R3.0.0***):
    - Users should upgrade to the current version of the **Intel® oneAPI DPC++/C++** (**ICX**) **Compiler**.
    - **ICC** remains a supported compiler option for building **DPS8M** on **Intel**-based **macOS** systems.
- Cross-compilation is supported. Popular targets including various **Linux** platforms, **Microsoft Windows** on **Intel** and **ARM** (using the **MinGW-w64** and **LLVM-MinGW** toolchains) and **Linux on POWER** (using the **IBM Advance Toolchain for Linux**) are regularly built and tested.

Linux prerequisites

Users of some **Red Hat** variants may need to enable the **PowerTools** repository or the **CodeReady Builder** AppStream to install `libuv`:

- RHEL 8:

```
subscription-manager repos --enable \
  "codeready-builder-for-rhel-8-$(arch)-rpms"
```

- CentOS Stream 8:

```
dnf config-manager --set-enabled "powertools"
```

- RHEL 9:

```
subscription-manager repos --enable \
  "codeready-builder-for-rhel-9-$(arch)-rpms"
```

- CentOS Stream 9:

```
dnf config-manager --set-enabled "crb"
```

Install the required prerequisites using a distribution package manager:

- Using **dnf** (for most **rpm**-based distributions) (as *root*):

```
dnf install "@Development Tools" libuv-devel
```

- Using **apt** (for most **deb**-based distributions) (as *root*):

```
apt install build-essential libuv1-dev
```

Standard Linux compilation

- Build the simulator (*standard build*) from the top-level source directory (using **GNU Make**):

```
make
```

Alternative Linux compilation

To use a compiler other than the default it is *usually* sufficient to simply set the **CC** environment variable (*if* the compiler accepts command-line arguments compatible with **GCC** or **Clang**). Other compilers are supported as well, but require additional configuration.

Examples of building the simulator on **Linux** using various popular compilers follows:

Clang

- Build the simulator using **Clang** (**Clang 14** *or later recommended*):

```
env CC="clang" make
```

Intel oneAPI DPC++/C++

- Build the simulator using **Intel oneAPI DPC++/C++** (**ICX**):

```
source /opt/intel/oneapi/setvars.sh && \
env CC="icx" make
```

AMD Optimizing C/C++

- Build the simulator using **AMD Optimizing C/C++** (**AOCC**), version 4.0.0, (with **AOCC**-provided **AMD LibM**):

```
export AOCCVER="4.0.0" &&                                  \
export AOCLPATH="/opt/AMD/aocc-compiler-${AOCCVER}" &&     \
source ${AOCCPATH}/setenv_AOCC.sh &&                       \
env CC="clang" CFLAGS="-mllvm -vector-library=AMDLIBM"     \
    LDFLAGS="-Wno-unused-command-line-argument"            \
    LOCALLIBS="-lalm"                                      \
  make
```

AOCC with AMD Optimized CPU Libraries

- Build the simulator using **AMD Optimizing C/C++** (**AOCC**), version 4.0.0, with **AMD Optimized CPU Libraries** (**AOCL**) (**AMD AOCL-LibM** and **AMD AOCL-LibMem**), version 4.0:

```
export AOCCVER="4.0.0" &&                                          \
export AOCCPATH="/opt/AMD/aocc-compiler-${AOCCVER}" &&             \
export AOCLVER="4.0" &&                                            \
export AOCLPATH="/opt/AMD/aocl/aocl-linux-aocc-${AOCLVER}" &&      \
export LD_LIBRARY_PATH="${AOCLPATH}/lib:${LD_LIBRARY_PATH}" &&     \
source ${AOCCPATH}/setenv_AOCC.sh &&                               \
env CC="clang" CFLAGS="-mllvm -vector-library=AMDLIBM"             \
    LDFLAGS="-Wno-unused-command-line-argument -L${AOCLPATH}/lib"  \
    LOCALLIBS="-lalm -laocl-libmem"                                \
  make
```

Oracle Developer Studio

- Build the simulator using **Oracle Developer Studio** (**SunCC**) for Linux, version 12.6:

```
env CFLAGS="-DNO_C_ELLIPSIS -Qy -xO5 -m64 -xlibmil -xCC -mt  \
            -xlibmopt -fno-semantic-interposition            \
            -xprefetch=auto -xprefetch_level=3"              \
    CC="/opt/oracle/developerstudio12.6/bin/suncc"           \
    NO_LTO=1 SUNPRO=1 NEED_128=1 CSTD="c11"                  \
  make
```

IBM Open XL C/C++ for Linux

- Build the simulator using **IBM Open XL C/C++ for Linux V17.1.1** for Linux on POWER:

```
env CFLAGS="-mcpu=power8"                          \
    CC="/opt/ibm/openxlC/17.1.1/bin/ibm-clang_r"   \
  make
```

- When building on IBM **POWER9** (or **Power10**) systems, '`-mcpu=power9`' (*or* '`-mcpu=power10`') should replace '`-mcpu=power8`' in the above compiler invocation.
- Refer to the **IBM Open XL C/C++ for Linux V17.1.1 documentation** for additional information.

IBM XL C/C++ for Linux

- Build the simulator using **IBM XL C/C++ for Linux V16.1.1** for Linux on POWER:

```
env CFLAGS="-qtls -qarch=pwr8"           \
    CC="/opt/ibm/xlC/16.1.1/bin/c99_r"   \
    CSTD="c11" NO_LTO=1
  make
```

  - When building on **POWER9** or higher systems, '`-qarch=pwr9`' should replace '`-qarch=pwr8`' in the above compiler invocation.
  - Compilation using higher optimization levels (*e.g.* '`-O4`', '`-O5`', '`-qhot`', *etc.*) and/or enabling automatic parallelization (*i.e.* '`-qsmp`') is possible, but the resulting binaries have *not* been benchmarked or extensively tested by **The DPS8M Development Team**.

NVIDIA HPC SDK C Compiler

- Build the simulator using **NVIDIA HPC SDK C Compiler** (**NVC**), version 22.7, for Linux/**x86_64** (also available for Linux/**ARM64** and Linux/**OpenPOWER**):

```
export NVCVER="22.7" &&                                                 \
export NVCPATH="/opt/nvidia/hpc_sdk/Linux_x86_64/${NVCVER}/bin" &&  \
env CFLAGS="-noswitcherror" CC="${NVCPATH}/nvc" NO_LTO=1             \
  make OPTFLAGS="-fast -O4 -Mipa=fast,inline"
```

  - The **NVIDIA HPC SDK C Compiler** is the successor to the **PGI C Compiler** product. If you are using the earlier **PGI C Compiler** (**PGCC**), adjust paths appropriately, and replace '**nvc**' with '**pgcc**' in the above invocation.
  - **DPS8M** is known to trigger bugs in many versions of the **PGCC** and **NVC** compilers, such as:

```
NVC++-F-0000-Internal compiler error. add_cilis(): bad jmp code 1056
```

    If you encounter this (*or similar*) compiler errors, try adding '`-Mnovect`' to '`OPTFLAGS`' as a workaround.

Arm HPC C/C++ Compiler for Linux

The **Arm HPC C/C++ Compiler for Linux** with **Arm Performance Libraries** (*also available as a component of* **Arm Allinea Studio**) provides a packaged **Clang**/**LLVM**-based toolchain with optimized math and string libraries, validated against common ARM HPC platforms.

Note the following examples *do not* make use of **Environment Modules** and/or **Lmod**, commonly used to manage compiler and development tool installations in HPC environments.

If your site uses modules (*i.e.* `module avail`), loading the appropriate module is usually preferred to setting paths manually. Contact your system administrator for site-specific configuration details and recommended local compiler flags.

- Build the simulator using the **Arm HPC C/C++ Compiler for Linux** (**ARMClang**), version 22.1, for Linux/**ARM64**:

```
export ACFLVER="22.1" &&                                        \
export ACFLCMP="arm-linux-compiler-${ACFLVER}" &&           \
export ACFLTYP="Generic-AArch64_RHEL-8_aarch64-linux" &&    \
export ACFLPATH="/opt/arm/${ACFLCMP}_${ACFLTYP}" &&         \
export PATH="${ACFLPATH}/bin:${PATH}" &&                    \
env CFLAGS="-mcpu=native"                                   \
    CC="armclang"                                           \
  make
```

ACFL with Arm Performance Libraries

- Build the simulator using the **Arm HPC C/C++ Compiler for Linux** (**ARMClang**) with the integrated **Arm Performance Libraries** (**ArmPL**), version 22.1, for Linux/**ARM64**:

```
export ACFLVER="22.1" &&                                        \
export ACFLCMP="arm-linux-compiler-${ACFLVER}" &&           \
export ACFLTYP="Generic-AArch64_RHEL-8_aarch64-linux" &&    \
export ACFLPATH="/opt/arm/${ACFLCMP}_${ACFLTYP}" &&         \
export PATH="${ACFLPATH}:${PATH}" &&                        \
env CFLAGS="-mcpu=native -armpl"                            \
    LDFLAGS="-armpl"                                        \
    CC="armclang"                                           \
  make OPTFLAGS="-Ofast"
```

- Build the simulator using the **Arm HPC C/C++ Compiler for Linux** (**ARMClang**) with the integrated **Arm Performance Libraries** (**ArmPL**), version 22.1, for Linux/**ARMv8-A+SVE2** (*Scalable Vector Extensions*):

```
export ACFLVER="22.1" &&                                        \
export ACFLCMP="arm-linux-compiler-${ACFLVER}" &&               \
export ACFLCMP="arm-linux-compiler-${ACFLVER}" &&               \
export ACFLPATH="/opt/arm/${ACFLCMP}_${ACFLTYP}" &&             \
export PATH="${ACFLPATH}:${PATH}" &&                            \
env CFLAGS="-march=armv8-a+sve2 -mcpu=native -armpl=sve"        \
    LDFLAGS="-armpl=sve"                                        \
    CC="armclang"                                               \
  make OPTFLAGS="-Ofast"
```

Linux cross-compilation

The following commands will download and cross-compile a local static **libuv** and then cross-compile the simulator.

IBM Advance Toolchain

- Using the **IBM Advance Toolchain V16** to cross-compile for Linux/**POWER**:
  - Build libuv:

```
env CC="/opt/at16.0/bin/powerpc64le-linux-gnu-gcc"  \
    LOCAL_CONFOPTS="--host=powerpc64le-linux-gnu"   \
    CFLAGS="-mcpu=power8 -mtune=power8"             \
  make libuvrel
```

  - Build the simulator:

```
env CC="/opt/at16.0/bin/powerpc64le-linux-gnu-gcc"  \
    CFLAGS="-mcpu=power8 -mtune=power8"             \
  make
```

  - When targeting **POWER9** or **Power10** processors, 'power9' or 'power10' should replace 'power8' in the above compiler invocation.
  - The **IBM Advance Toolchain** versions **14**, **15**, and **16** have been extensively tested and used for cross-compiling **DPS8M**.

Arm GNU Toolchain

The **GNU Toolchain for the Arm Architecture** (referred to as the "**Arm GNU Toolchain**") provides regularly updated, high-quality, validated Linux/**ARM** cross-compilers running on Microsoft Windows, Linux, and macOS.

Linux/ARMv7-HF

- Using the **Arm GNU Toolchain** on Linux/x86_64, version **11.3.Rel1**, to cross-compile for Linux/**ARMv7-HF** (*hardware floating point*):
  - Build libuv:

```
export AGTREL="arm-gnu-toolchain-11.3.rel1-x86_64" &&               \
export AGTPATH="/opt/${AGTREL}-arm-none-linux-gnueabihf/bin/" &&    \
env CC="${AGTPATH}/arm-none-linux-gnueabihf-gcc"                    \
    CFLAGS="-march=armv7-a+fp"                                      \
    LOCAL_CONFOPTS="--host=arm-none-linux-gnueabihf"                \
  make libuvrel
```

- Build the simulator:

```
env CC="${AGTPATH}/arm-none-linux-gnueabihf-gcc"                    \
    CFLAGS="-march=armv7-a+fp"                                      \
     NEED_128=1                                                     \
  make
```

Linux/ARM64

- Using the **Arm GNU Toolchain** on Linux/x86_64, version **11.3.Rel1**, to cross-compile for Linux/**ARM64**:
  - Build `libuv`:

```
export AGTREL="arm-gnu-toolchain-11.3.rel1-x86_64" &&               \
export AGTPATH="/opt/${AGTREL}-aarch64-none-linux-gnu/bin/" &&      \
env CC="${AGTPATH}/aarch64-none-linux-gnu-gcc"                      \
    LOCAL_CONFOPTS="--host=aarch64-none-linux-gnu"                  \
  make libuvrel
```

  - Build the simulator:

```
env CC="${AGTPATH}/aarch64-none-linux-gnu-gcc" \
  make
```

Linux/ARM64BE

- Using the **Arm GNU Toolchain** on Linux/x86_64, version **11.3.Rel1**, to cross-compile for Linux/**ARM64BE** (*big endian*):
  - Build `libuv`:

```
export AGTREL="arm-gnu-toolchain-11.3.rel1-x86_64" &&                  \
export AGTPATH="/opt/${AGTREL}-aarch64_be-none-linux-gnu/bin/" &&      \
env CC="${AGTPATH}/aarch64_be-none-linux-gnu-gcc"                      \
    LOCAL_CONFOPTS="--host=aarch64_be-none-linux-gnu"                  \
  make libuvrel
```

  - Build the simulator:

```
env CC="${AGTPATH}/aarch64_be-none-linux-gnu-gcc" \
  make
```

Linaro GNU Toolchain

The **Linaro GNU Toolchain Integration Builds**, updated monthly, provides Linux/**ARM** and Linux/**ARM64** reference toolchains, closely tracking upstream repositories, allowing developers to easily test new compiler and processor features before the next **Arm GNU Toolchain** release.

Linux/ARMv7-HF

- Using the **Linaro GNU Toolchain Integration Build** on Linux/x86_64, version **13.0.0-2022.09**, to cross-compile for Linux/**ARMv7-HF** (*hardware floating point*):
  - Build `libuv`:

```
export LINREL="gcc-linaro-13.0.0-2022.09-x86_64" &&                \
export LINPATH="/opt/${LINREL}_arm-linux-gnueabihf/bin/" &&         \
env CC="${LINPATH}/arm-linux-gnueabihf-gcc"                        \
    CFLAGS="-march=armv7-a+fp"                                     \
    LOCAL_CONFOPTS="--host=arm-linux-gnueabihf"                    \
  make libuvrel
```

- Build the simulator:

```
env CC="${LINPATH}/arm-linux-gnueabihf-gcc"                    \
    CFLAGS="-march=armv7-a+fp"                                 \
    NEED_128=1                                                 \
  make
```

Linux/ARM64

- Using the **Linaro GNU Toolchain Integration Build** on Linux/x86_64, version **13.0.0-2022.09**, to cross-compile for Linux/**ARM64**:
  - Build `libuv`:

```
export LINREL="gcc-linaro-13.0.0-2022.09-x86_64" &&            \
export LINPATH="/opt/${LINREL}_aarch64-linux-gnu/bin/" &&       \
env CC="${LINPATH}/aarch64-linux-gnu-gcc"                      \
    LOCAL_CONFOPTS="--host=aarch64-linux-gnu"                  \
  make libuvrel
```

  - Build the simulator:

```
env CC="${LINPATH}/aarch64-linux-gnu-gcc" \
  make
```

crosstool-NG

crosstool-NG is a versatile cross-toolchain generator, which can be used to generate **GCC**-based toolchains for a huge variety of architectures and operating systems (*mainly* **Linux**).

DPS8M is regularly built by **The DPS8M Development Team** for many **Linux** architectures using **crosstool-NG** generated toolchains, utilizing both the **glibc** and **musl** C libraries. The following **CT-NG** examples are intended to be instructive, but are by no means exhaustive.

Linux/RV64

- Using a **crosstool-NG** generated **GCC**+**musl** toolchain to cross-compile for Linux/**RV64** (**64-bit RISC-V** *static binary*):
  - Build `libuv`:

```
export CTNG="riscv64-local-linux-musl" &&                      \
env CC="/home/jhj/x-tools/${CTNG}/bin/${CTNG}-gcc"             \
    LOCAL_CONFOPTS="--host=${CTNG}"                            \
  make libuvrel
```

  - Build the simulator:

```
env CC="/home/jhj/x-tools/${CTNG}/bin/${CTNG}-gcc"             \
    LDFLAGS="-static"                                          \
    LOCALLIBS="-latomic"                                       \
  make
```

Linux/i686

- Using a **crosstool-NG** generated **GCC**+**musl** toolchain to cross-compile for Linux/**i686** (**32-bit** *static binary*):
  - Build `libuv`:

```
export CTNG="i686-local-linux-musl" &&                  \
env CC="/home/jhj/x-tools/${CTNG}/bin/${CTNG}-gcc"      \
    LOCAL_CONFOPTS="--host=${CTNG}"                     \
  make libuvrel
```

- Build the simulator:

```
env CC="/home/jhj/x-tools/${CTNG}/bin/${CTNG}-gcc"      \
    LDFLAGS="-static"                                   \
    LOCALLIBS="-latomic"                                \
    NEED_128=1                                          \
  make
```

Linux/ARMv6-HF

- Using a **crosstool-NG** generated **GCC**+**glibc** toolchain to cross-compile for Linux/**ARMv6-HF** (*hardware floating point*):
  - Build `libuv`:

```
export CTNG="armv6-local-linux-gnueabihf" &&            \
env CC="/home/jhj/x-tools/${CTNG}/bin/${CTNG}-gcc"      \
    LOCAL_CONFOPTS="--host=${CTNG}"                     \
    CFLAGS="-march=armv6+fp"                            \
  make libuvrel
```

  - Build the simulator:

```
env CC="/home/jhj/x-tools/${CTNG}/bin/${CTNG}-gcc"      \
    CFLAGS="-march=armv6+fp"                            \
    LOCALLIBS="-latomic"                                \
    NEED_128=1                                          \
  make
```

Linux/PPC64le

- Using a **crosstool-NG** generated **GCC**+**musl** toolchain to cross-compile for Linux/**PPC64le** (**64-bit POWER9** *little endian static binary*):
  - Build `libuv`:

```
export CTNG="powerpc64le-local-linux-musl" &&           \
env CC="/home/jhj/x-tools/${CTNG}/bin/${CTNG}-gcc"      \
    LOCAL_CONFOPTS="--host=${CTNG}"                     \
    CFLAGS="-mcpu=power9"                               \
  make libuvrel
```

  - Build the simulator:

```
env CC="/home/jhj/x-tools/${CTNG}/bin/${CTNG}-gcc"      \
    CFLAGS="-mcpu=power9"                               \
    LDFLAGS="-static"                                   \
    LOCALLIBS="-latomic"                                \
  make
```

Additional Linux Notes

- Although normally handled automatically, when building for (or cross-compiling to) many 32-bit targets (or when using a compiler lacking support for 128-bit integers) it may be necessary to set the **`NEED_128=1`** build option (via the environment or as an argument to **`make`**).

macOS

- Ensure you are running a supported release of **macOS** with current updates applied.
  - Both **Intel** and **ARM64** systems are regularly tested by **The DPS8M Development Team**.
- **Xcode** is required; it is **strongly recommended** to use the most recent release for optimal performance.
  - Building with **Intel® C++ Compiler Classic for macOS** (**`icc`**) **2022.6.0** or later is also supported.
  - At the time of writing, building the simulator on **macOS** using **GCC** is ***not recommended***.
- The following instructions were verified using **macOS 12.6** with **Xcode 14.0** (Apple Clang 14.0.0).

macOS prerequisites

- **Homebrew** is the recommended package manager for installing build prerequisites:

```
brew install libuv pkg-config
```

- Users of other package managers (*e.g.* pkgsrc, MacPorts) must set the **`CFLAGS`** (*e.g.* '`-I/opt/include`'), **`LDFLAGS`** (*e.g.* '`-L/opt/lib`'), and **`LIBUV`** (*e.g.* '`-luv`') environment variables appropriately.

macOS compilation

Build the simulator from the top-level source directory (using **GNU Make**):

Xcode

- Build using **Xcode**:

```
make
```

Intel C/C++ Compiler Classic for macOS

- Build using **Intel® C/C++ Compiler Classic for macOS** (**`icc`**):

```
env CC="icc" CFLAGS="-xHost" make
```

macOS cross-compilation

The following commands will download and cross-compile a local static **`libuv`** and then cross-compile the simulator using **Xcode**.

You **must** perform a '**`make distclean`**' before building for a different target.

- Install required prerequisites using **Homebrew**:

```
brew install wget pkg-config autoconf automake libtool coreutils
```

Build the simulator from the top-level source directory (using **GNU Make**):

ARM64

- Cross-compilation targeting **ARM64 macOS** 11:
  - Build `libuv`:

```
make distclean &&                                   \
env CFLAGS="-target arm64-apple-macos11             \
            -mmacosx-version-min=11.0"              \
    LOCAL_CONFOPTS="--host=arm64-apple-darwin"      \
  make libuvrel HOMEBREW_INC= HOMEBREW_LIB=
```

  - Build the simulator:

```
env CFLAGS="-target arm64-apple-macos11             \
            -mmacosx-version-min=11.0"              \
    LDFLAGS="-target arm64-apple-macos11            \
             -mmacosx-version-min=11.0"             \
  make HOMEBREW_INC= HOMEBREW_LIB=
```

Intel

- Cross-compilation targeting **Intel macOS** 10.15:
  - Build `libuv`:

```
make distclean &&                                   \
env CFLAGS="-target x86_64-apple-macos10.15         \
            -mmacosx-version-min=10.15"             \
    LOCAL_CONFOPTS="--host=x86_64-apple-darwin"     \
  make libuvrel HOMEBREW_INC= HOMEBREW_LIB=
```

  - Build the simulator:

```
env CFLAGS="-target x86_64-apple-macos10.15         \
            -mmacosx-version-min=10.15"             \
    LDFLAGS="-target x86_64-apple-macos10.15        \
             -mmacosx-version-min=10.15"            \
  make HOMEBREW_INC= HOMEBREW_LIB=
```

Universal

- The following more complex example builds a **macOS *Universal Binary***.
  - The ***Universal Binary*** will support *three* architectures: **ARM64**, **Intel**, and **Intel Haswell**.
  - The simulator (and **`libuv`**) will be cross-compiled three times each, once for each architecture.
  - The **`lipo`** utility will be used to create the universal **`dps8`** binary (in the top-level build directory).
- Cross-compilation targeting **ARM64**, **Intel**, **Intel Haswell** (**AVX2**):
  - Build **ARM64** `libuv`:

```
make distclean &&                                   \
env CFLAGS="-target arm64-apple-macos11             \
            -mmacosx-version-min=11.0"              \
    LOCAL_CONFOPTS="--host=arm64-apple-darwin"      \
  make libuvrel HOMEBREW_INC= HOMEBREW_LIB=
```

- Build **ARM64** `dps8`:

```
env CFLAGS="-target arm64-apple-macos11          \
            -mmacosx-version-min=11.0"           \
    LDFLAGS="-target arm64-apple-macos11          \
             -mmacosx-version-min=11.0"          \
  make HOMEBREW_INC= HOMEBREW_LIB= &&             \
cp -f "src/dps8/dps8" "dps8.arm64"
```

- Build **Intel** `libuv`:

```
make distclean &&                                 \
env CFLAGS="-target x86_64-apple-macos10.15       \
            -mmacosx-version-min=10.15"           \
    LOCAL_CONFOPTS="--host=x86_64-apple-darwin"   \
  make libuvrel HOMEBREW_INC= HOMEBREW_LIB=
```

- Build **Intel** `dps8`:

```
env CFLAGS="-target x86_64-apple-macos10.15       \
            -mmacosx-version-min=10.15"           \
    LDFLAGS="-target x86_64-apple-macos10.15      \
             -mmacosx-version-min=10.15"          \
  make HOMEBREW_INC= HOMEBREW_LIB= &&             \
cp -f "src/dps8/dps8" "dps8.x86_64"
```

- Build **Intel Haswell** `libuv`:

```
make distclean &&                                 \
env CFLAGS="-target x86_64h-apple-macos10.15      \
            -mmacosx-version-min=10.15            \
            -march=haswell"                       \
    LOCAL_CONFOPTS="--host=x86_64-apple-darwin"   \
  make libuvrel HOMEBREW_INC= HOMEBREW_LIB=
```

- Build **Intel Haswell** `dps8`:

```
env CFLAGS="-target x86_64h-apple-macos10.15      \
            -mmacosx-version-min=10.15            \
            -march=haswell"                       \
    LDFLAGS="-target x86_64h-apple-macos10.15     \
            -mmacosx-version-min=10.15"           \
  make HOMEBREW_INC= HOMEBREW_LIB= &&             \
cp -f "src/dps8/dps8" "dps8.x86_64h"
```

- Create the **Universal Binary** using `lipo`:

```
lipo -create -output "dps8"                       \
  "dps8.x86_64" "dps8.x86_64h" "dps8.arm64" &&    \
make distclean && rm -f                           \
  "dps8.x86_64" "dps8.x86_64h" "dps8.arm64" &&    \
lipo -detailed_info "dps8"
```

Windows

- Ensure you are running a supported release of Microsoft **Windows** on a supported platform.

  - Microsoft **Windows 10** and **11** on **x86_64** and **i686** are regularly tested by **The DPS8M Development Team**.

- Microsoft **Windows** supports various development and runtime environments, including **MSVCRT**/**UCRT**/**MinGW**, **Cygwin**, **Midipix**, **MSYS2**, **UWIN**, **UWP**, and others.
  - Care must be taken to avoid mixing incompatible libraries and tools.
- Cross-compilation is supported. Builds targeting Microsoft **Windows** (**MinGW** and **Cygwin**) running on **x86_64**, **i686**, **ARMv7**, and **ARM64** platforms are regularly cross-compiled from a variety of UNIX-like systems (using **LLVM-MinGW** and **MinGW-GCC**), and from Microsoft **Windows** using **Cygwin**.
- Microsoft **Windows** also provides **Linux** compatibility via the **Windows Subsystem for Linux** (**WSL**).
  - **Windows Subsystem for Linux** users should refer to the **Linux** sections of the documentation.

Cygwin

- Ensure you are running a current and up-to-date version of **Cygwin**.
- Only the **64-bit** version of **Cygwin** is regularly tested by **The DPS8M Development Team**.
  - Although the **32-bit** version of **Cygwin** is not regularly tested (*and not recommended due to suboptimal performance*), it *should* work for building **DPS8M** (with the `NEED_128=1` build option).

Cygwin prerequisites

- Compilation problems in the **Cygwin** environment are often caused by incomplete or interrupted package installations, or by the installation of packages using non-standard tools (*e.g.* `apt-cyg`), resulting in missing files and dangling or missing symbolic links.
  - Before attempting to build **DPS8M** using **Cygwin**:
    1. First, update the **Cygwin** **`setup.exe`** application to the latest available version.
    2. Update ***all*** installed packages using the new **Cygwin** **`setup.exe`** application.
    3. Install the required prerequisite packages using **Cygwin** **`setup.exe`**:
       * `autoconf`
       * `cmake`
       * `cygcheck`
       * `gcc`
       * `libtool`
       * `libuv1`
       * `libuv1-devel`
       * `make`
       * `pkg-config`
       * `unzip`
       * `wget`
    4. **Most importantly**, invoke the **`cygcheck`** utility (*i.e.* `cygcheck -cv | grep -v "OK$"`) to verify the integrity of all currently installed packages and correct any problems before continuing.

Standard Cygwin compilation

- Build the simulator from the top-level source directory (using **GNU Make**):

```
env CFLAGS="-DUSE_FLOCK=1 -DUSE_FCNTL=1" make
```

Cygwin-hosted cross-compilation to MinGW

The following commands will download and cross-compile a local native **libuv** library and then cross-compile the simulator.

You **must** perform a 'make distclean' followed by an 'rm -rf ${HOME}/libuv-build' before building for a different target (or changing build flags).

In the following cross-compilation examples, the *latest* **libuv** sources (from the v1.x *git* branch) are used, but the current official release (available from https://libuv.org/) can also be used.

Windows i686

- Using **GCC** (*the* **Cygwin** **mingw64-i686-gcc-core** *package*) to cross-compile a native **32-bit** Windows executable (*not depending on Cygwin*):
  - Build libuv:

```
mkdir -p "${HOME}/libuv-build" &&                                  \
mkdir -p "${HOME}/libuv-win32-i686" &&                             \
( cd "${HOME}/libuv-build" &&                                      \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&     \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" &&                        \
  mkdir -p "build" && cd "build" &&                                \
  cmake .. -DCMAKE_SYSTEM_NAME="Windows"                           \
           -DCMAKE_SYSTEM_VERSION="6.1"                            \
           -DCMAKE_C_COMPILER="i686-w64-mingw32-gcc"               \
           -DCMAKE_INSTALL_PREFIX="${HOME}/libuv-win32-i686" &&    \
  cmake --build . && cmake --install . )
```

  - Build the simulator:

```
env CFLAGS="-I${HOME}/libuv-win32-i686/include -D__MINGW64__       \
            -DUSE_FLOCK=1 -DUSE_FCNTL=1"                           \
    CC="i686-w64-mingw32-gcc"                                      \
    LDFLAGS="-L${HOME}/libuv-win32-i686/lib" NEED_128=1            \
  make CROSS="MINGW64"
```

  - The compiled native binary will require libwinpthread-1.dll (located at /usr/i686-w64-mingw32/sys-root/mingw/bin/libwinpthread-1.dll) and libuv.dll (located at ${HOME}/libuv-win32-i686/bin/libuv.dll) at runtime.
    - It is sufficient to copy these files into the directory containing the dps8.exe binary.

Windows x86_64

- Using **GCC** (*the* **Cygwin** **mingw64-x86_64-gcc-core** *package*) to cross-compile a native **64-bit** Windows executable (*not depending on Cygwin*):
  - Build libuv:

```
mkdir -p "${HOME}/libuv-build" &&                                    \
mkdir -p "${HOME}/libuv-win32-x86_64" &&                             \
( cd "${HOME}/libuv-build" &&                                        \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&       \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" &&                          \
  mkdir -p "build" && cd "build" &&                                  \
  cmake .. -DCMAKE_SYSTEM_NAME="Windows"                             \
           -DCMAKE_SYSTEM_VERSION="6.1"                              \
           -DCMAKE_C_COMPILER="x86_64-w64-mingw32-gcc"               \
           -DCMAKE_INSTALL_PREFIX="${HOME}/libuv-win32-x86_64" &&    \
  cmake --build . && cmake --install . )
```

- Build the simulator:

```
env CFLAGS="-I${HOME}/libuv-win32-x86_64/include -D__MINGW64__       \
            -DUSE_FLOCK=1 -DUSE_FCNTL=1"                             \
    CC="x86_64-w64-mingw32-gcc"                                      \
    LDFLAGS="-L${HOME}/libuv-win32-x86_64/lib"                       \
  make CROSS="MINGW64"
```

- The compiled native binary will require `libwinpthread-1.dll` (located at `/usr/x86_64-w64-mingw32/sys-root/mingw/bin/libwinpthread-1.dll`) and `libuv.dll` (located at `${HOME}/libuv-win32-x86_64/bin/libuv.dll`) at runtime.
  - It is sufficient to copy these files into the directory containing the `dps8.exe` binary.

Unix-hosted cross-compilation to Cygwin

This section documents the procedure for building Windows Cygwin binaries using a Unix-based host system to download and cross-compile **`libuv`** library and then cross-compile the simulator.

In the following cross-compilation examples, the *latest* **`libuv`** sources (from the `v1.x` *git* branch) are used, but the current official release (available from https://libuv.org/) can also be used.

- Using **GCC** from the **Fedora Cygwin cross-compiler toolchain** (maintained by Yaakov Selkowitz) to cross-compile a **64-bit** Windows Cygwin executable (*depending on* `cygwin1.dll`):
  - Build `libuv`:

```
mkdir -p "${HOME}/libuv-build" &&                                    \
mkdir -p "${HOME}/libuv-cygwin-x64" &&                               \
( cd "${HOME}/libuv-build" &&                                        \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&       \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" && sh ./autogen.sh &&       \
  cygwin64-configure --prefix="${HOME}/libuv-cygwin-x64"             \
    --enable-static --disable-shared &&                              \
  make && make install )
```

  - Build the simulator:

```
cygwin64-make
```

- Using **GCC** from the **Fedora Cygwin cross-compiler toolchain** to cross-compile a **32-bit** Windows Cygwin executable (*depending on* `cygwin1.dll`):
  - Build `libuv`:

```
mkdir -p "${HOME}/libuv-build" &&                                  \
mkdir -p "${HOME}/libuv-cygwin-x86" &&                             \
( cd "${HOME}/libuv-build" &&                                      \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&     \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" && sh ./autogen.sh &&     \
  cygwin32-configure --prefix="${HOME}/libuv-cygwin-x86"           \
    --enable-static --disable-shared &&                            \
  make && make install )
```

- Build the simulator:

```
env NEED_128=1 cygwin32-make
```

MSYS2

- **DPS8M** can be built as a native **MSYS2** application without special configuration, using the "**MSYS2 Environment**".
- Cross-compilation using the **MSYS2**-provided **MINGW32**, **MINGW64**, **UCRT64**, **CLANG32**, **CLANG64**, and **CLANGARM64** environments is *currently untested*.

Unix-hosted LLVM-MinGW Clang cross-compilation

The **LLVM-MinGW Clang** toolchain supports building native Windows binaries (**i686**, **x86_64**, **ARMv7**, and **ARM64** systems) on *non*-**Windows** host systems (or using the **Windows Subsystem for Linux**).

The **LLVM-MinGW Docker Container** provides pre-built and fully configured **LLVM-MinGW** toolchains (including appropriate compiler symlinks) which are regularly used by **The DPS8M Development Team**.

In the following cross-compilation examples, the *latest* **libuv** sources (from the `v1.x` *git* branch) are used, but the current official release (available from https://libuv.org/) can also be used.

Windows i686

- Using **Clang** (*the* **LLVM-MinGW** *compiler*) to cross-compile a local static `libuv` library and a native **32-bit** Windows/**i686** executable:
  - Build `libuv`:

```
mkdir -p "${HOME}/libuv-build" &&                                  \
mkdir -p "${HOME}/libuv-win32-i686" &&                             \
( cd "${HOME}/libuv-build" &&                                      \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&     \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" && sh ./autogen.sh &&     \
  ./configure --prefix="${HOME}/libuv-win32-i686"                  \
    --enable-static --disable-shared --host="i686-w64-mingw32" &&  \
  make && make install )
```

  - Build the simulator:

```
env CC="i686-w64-mingw32-clang"                                    \
    CFLAGS="-I${HOME}/libuv-win32-i686/include -D__MINGW64__"      \
    LDFLAGS="-L${HOME}/libuv-win32-i686/lib" NEED_128=1            \
  make CROSS="MINGW64"
```

Windows x86_64

- Using **Clang** (*the* **LLVM-MinGW** *compiler*) to cross-compile a local static `libuv` library and a native **64-bit** Windows/**x86_64** executable:
  - Build `libuv`:

```
mkdir -p "${HOME}/libuv-build" &&                                    \
mkdir -p "${HOME}/libuv-win32-x86_64" &&                             \
( cd "${HOME}/libuv-build" &&                                        \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&       \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" && sh ./autogen.sh &&       \
  ./configure --prefix="${HOME}/libuv-win32-x86_64"                  \
    --enable-static --disable-shared --host="x86_64-w64-mingw32" &&  \
  make && make install )
```

  - Build the simulator:

```
env CC="x86_64-w64-mingw32-clang"                                    \
    CFLAGS="-I${HOME}/libuv-win32-x86_64/include -D__MINGW64__"      \
    LDFLAGS="-L${HOME}/libuv-win32-x86_64/lib"                       \
  make CROSS="MINGW64"
```

Windows ARMv7

- Using **Clang** (*the* **LLVM-MinGW** *compiler*) to cross-compile a local static `libuv` library and a native **32-bit** Windows/**ARMv7** executable:
  - Build `libuv`:

```
mkdir -p "${HOME}/libuv-build" &&                                    \
mkdir -p "${HOME}/libuv-win32-armv7" &&                              \
( cd "${HOME}/libuv-build" &&                                        \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&       \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" && sh ./autogen.sh &&       \
  ./configure --prefix="${HOME}/libuv-win32-armv7"                   \
    --enable-static --disable-shared --host="armv7-w64-mingw32" &&   \
  make && make install )
```

  - Build the simulator:

```
env CC="armv7-w64-mingw32-clang"                                     \
    CFLAGS="-I${HOME}/libuv-win32-armv7/include -D__MINGW64__"       \
    LDFLAGS="-L${HOME}/libuv-win32-armv7/lib"                        \
    NEED_128=1                                                       \
  make CROSS="MINGW64"
```

Windows ARM64

- Using **Clang** (*the* **LLVM-MinGW** *compiler*) to cross-compile a local static `libuv` library and a native **64-bit** Windows/**ARM64** executable:
  - Build `libuv`:

```
mkdir -p "${HOME}/libuv-build" &&                                    \
mkdir -p "${HOME}/libuv-win32-arm64" &&                              \
( cd "${HOME}/libuv-build" &&                                        \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&       \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" && sh ./autogen.sh &&       \
  ./configure --prefix="${HOME}/libuv-win32-arm64"                   \
    --enable-static --disable-shared --host="aarch64-w64-mingw32" &&  \
  make && make install )
```

- Build the simulator:

```
env CC="aarch64-w64-mingw32-clang"                                   \
    CFLAGS="-I${HOME}/libuv-win32-arm64/include -D__MINGW64__"       \
    LDFLAGS="-L${HOME}/libuv-win32-arm64/lib"                        \
  make CROSS="MINGW64"
```

Unix-hosted MinGW-w64 GCC cross-compilation

The **MinGW-w64 GCC** toolchain supports building native Windows (**i686** and **x86_64**) executables on *non*-**Windows** host systems (or **Windows** using the **Windows Subsystem for Linux**).

- Many **MinGW-w64 toolchains** are available for a wide variety of host platforms and operating systems.
- Version **9.0** is the *minimum* version of **MinGW-w64** tested with **DPS8M**; version **10.0** is *recommended*.
- **The DPS8M Development Team** regularly cross-compiles **Windows** executables using **GCC**-based **MinGW-w64** toolchains on **Alpine Linux** and **Fedora Linux** host systems.

In the following cross-compilation examples, the *latest* **libuv** sources (from the `v1.x` *git* branch) are used, but the current official release (available from https://libuv.org/) can also be used.

Windows i686

- Using **GCC** (*the* **MinGW-w64** *compiler*) to cross-compile a local static `libuv` library and a native **32-bit** Windows/**i686** executable:
  - Build `libuv`:

```
mkdir -p "${HOME}/libuv-build" &&                                    \
mkdir -p "${HOME}/libuv-win32-i686" &&                               \
( cd "${HOME}/libuv-build" &&                                        \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&       \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" && sh ./autogen.sh &&       \
  ./configure --prefix="${HOME}/libuv-win32-i686"                    \
    --enable-static --disable-shared --host="i686-w64-mingw32" &&    \
  make && make install )
```

  - Build the simulator:

```
env CC="i686-w64-mingw32-gcc"                                        \
    CFLAGS="-I${HOME}/libuv-win32-i686/include                       \
            -D__MINGW64__ -pthread"                                  \
    LDFLAGS="-L${HOME}/libuv-win32-i686/lib -lpthread"               \
    NEED_128=1                                                       \
  make CROSS="MINGW64"
```

Windows x86_64

- Using **GCC** (*the* **MinGW-w64** *compiler*) to cross-compile a local static `libuv` library and a native **64-bit** Windows/**x86_64** executable:

- Build `libuv`:

```
mkdir -p "${HOME}/libuv-build" &&                                    \
mkdir -p "${HOME}/libuv-win32-x86_64" &&                             \
( cd "${HOME}/libuv-build" &&                                        \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&       \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" && sh ./autogen.sh &&       \
  ./configure --prefix="${HOME}/libuv-win32-x86_64"                  \
    --enable-static --disable-shared --host="x86_64-w64-mingw32" &&  \
  make && make install )
```

- Build the simulator:

```
env CC="x86_64-w64-mingw32-gcc"                                      \
    CFLAGS="-I${HOME}/libuv-win32-x86_64/include                     \
            -D__MINGW64__ -pthread"                                  \
    LDFLAGS="-L${HOME}/libuv-win32-x86_64/lib -lpthread"             \
  make CROSS="MINGW64"
```

Simulator Command Reference

This chapter provides reference documentation for the **DPS8M** simulator command set.

- This information is also available from within the simulator; it is accessible by using the interactive HELP command.

!

- The simulator can execute host operating system commands with the "!" (*spawn*) command.

"**!**"	Spawn the hosts default command interpreter
"**!** <command>"	Execute the host operating system command

- **NOTE**: The *exit status* from the command which was executed is set as the *command completion status* for the "!" command. This may influence any enabled ON condition traps.

ASSERT

The ASSERT command tests a simulator state condition and halts command file execution if the condition is false:

```
        ASSERT <Simulator State Expressions>
```

- If the indicated expression evaluates to false, the command completes with an AFAIL condition. By default, when a command file encounters a command which returns the AFAIL condition, it will exit the running command file with the AFAIL status to the calling command file. This behavior can be changed with the ON command as well as switches to the invoking DO command.

Examples

The command file below might be used to bootstrap a hypothetical system that halts after the initial load from disk. The ASSERT command can then be used to confirm that the load completed successfully by examining the CPU's "A" register for the expected value:

```
        ; Example INI file
        BOOT
        ; A register contains error code; 0 = good boot
        ASSERT A=0
        RUN
```

- In the above example, if the "A" register is *not* 0, the "ASSERT A=0" command will be displayed to the user, and the command file will be aborted with an "Assertion failed" message. Otherwise, the command file will continue to bring up the system.
- See the **IF** command documentation for more information and details regarding simulator state expressions.

ATTACH (AT)

The ATTACH (*abbreviation* AT) command associates a unit and a file:

```
        ATTACH <unit> <filename>
```

Some devices have more detailed or specific help available with:

```
        HELP <device> ATTACH
```

Switches

-n If the "-n" switch is specified when ATTACH is executed, a new file will be created when the filename specified does not exist, or an existing file will have it's size truncated to zero, and an appropriate message is printed.

-e If the file does not exist, and the "-e" switch *was not* specified, a new file is created, and an appropriate message is printed. If the "-e" switch *was* specified, a new file is *not* created, and an error message is printed.

-r If the "-r" switch is specified, or the file is write protected by host operating system, ATTACH tries to open the file in read only mode. If the file does not exist, or the unit does not support read only operation, an error occurs. Input-only devices, such as card readers, or storage devices with write locking switches, such as disks or tapes, support read only operation - other devices do not. If a file is attached read only, its contents can be examined but not modified.

-q If the "-q" switch is specified when creating a new file ("-n") or opening one read only ("-r"), the message announcing this fact is suppressed.

-f For simulated magnetic tapes, the ATTACH command can specify the format of the attached tape image file:

```
        ATTACH -f <tape_unit> <format> <filename>
```

- The currently supported magnetic tape image file formats are:

"**SIMH**"	The **SIMH** / **DPS8M** native portable tape format
"**E11**"	The *D Bit* **Ersatz-11** simulator format
"**TPC**"	The **TPC** format (*used by* SIMH *prior to V2.3*)
"**P7B**"	The **Paul Pierce** 7-track tape archive format

- The default tape format can also be specified with the SET command prior to using the ATTACH command:

```
SET <tape_unit> FORMAT=<format>
ATTACH <tape_unit> <filename>
```

- The format of a currently attached tape image can be displayed with the `SHOW FORMAT` command:

```
SHOW <unit> FORMAT
```

Examples

The following example illustrates common `ATTACH` usage:

```
; Associate the tape image file "12.7MULTICS.tap" with the tape0 unit
; in read-only mode, where tape0 corresponds to the first tape device.
ATTACH -r tape0 12.7MULTICS.tap

; Associate the disk image file "root.dsk" with the disk0 unit.
; The disk0 unit corresponds to the first disk device.
ATTACH disk0 root.dsk
```

AUTOINPUT (AI)

The `AUTOINPUT` command (abbreviated `AI`) provides the specified input to the primary operator console (OPC0):

```
AUTOINPUT <string>
```

- To send a **<CR><LF>** to the console, include "**\n**" in the *string*.
- Specifying "**\z**" as the content of the *string* will end all autoinput from the invoking script.
- The `AUTOINPUT` command can open the console for input only when a specific *matching string* is found.
  - To specify a *matching string*, use the form of "**\yString\y**" for a substring match, or "**\xString\x**" for an exact match.

Examples

```
; Opens the console when AUTOINPUT sees the "M->" string.
; Any line of text containing this string will match.
AUTOINPUT \yM->\y

; Open the console when AUTOINPUT sees the "Ready" string.
; A line of text containing only this exact string will match.
AUTOINPUT \xReady\x
```

AUTOINPUT2 (AI2)

The `AUTOINPUT2` command (abbreviated `AI2`) provides the specified input to the secondary operator console (OPC1).

- Refer to the documentation for the `AUTOINPUT` command for `AUTOINPUT2` usage.

BOOT (BO)

The BOOT command (*abbreviated* BO) resets all devices and bootstraps the device and unit given by its argument. If no unit is supplied, unit 0 is bootstrapped. The specified unit must be ATTACH'ed.

When booting Multics, the boot device should always be iom0. Assuming a tape is attached to the tape0 device, it will be bootstrapped into memory and the system will transfer control to the boot record.

Example

```
          ; Boot Multics using iom0
          boot iom0
```

BURST

Burst process output from printer.

CABLE (C)

The CABLE command (*abbreviated* C) connects (or "*strings*") a simulated cable between devices.

```
     CABLE <device> <port/channel> <device> {port/channel}
```

Examples

- Connect a cable from (*system controller unit*) "SCU**i**" port "**j**" to (*central processing unit*) "CPU**k**" port "**l**".

  ```
       CABLE SCUi j CPUk l
  ```

- Connect a cable from "SCU**i**" port "**j**" to (*input/output multiplexer*) "IOM**k**" port "**l**".

  ```
       CABLE SCUi j IOMk l
  ```

- Connect a cable from "IOM**i**" channel "**j**" to (*magnetic tape processor*) "MTP**k**" port "**l**", where "**l**" defaults to "**0**" when not specified.

  ```
       CABLE IOMi j MTPk
       CABLE IOMi j MTPk l
  ```

- Connect a cable from "IOM**i**"'s channel "**j**" to (*mass storage processor*) "MSP**k**" port "**l**", where "**l**" defaults to "**0**" when not specified.

  ```
       CABLE IOMi j MSPk
       CABLE IOMi j MSPk l
  ```

- Connect a cable from "IOM**i**" channel "**j**" to (*integrated peripheral controller*) "IPC**k**" port "**l**", where "**l**" defaults to "**0**" when not specified.

```
CABLE IOMi j IPCk
CABLE IOMi j IPCk l
```

- Connect a cable from "IOM**i**" channel "**j**" to (*operator console*) "OPC**k**".

```
CABLE IOMi j OPCk
```

- Connect a cable from "IOM**i**" channel "**j**" to (*front-end network processor*) "FNP**k**".

```
CABLE IOMi j FNPk
```

- Connect a cable from "MTP**i**" device code "**j**" to (*tape drive*) "TAPE**k**".

```
CABLE MTPi j TAPEk
```

- Connect a cable from "IPC**i**" device code "**j**" to (*disk drive*) "DISK**k**".

```
CABLE IPCi j DISKk
```

- Connect a cable from "MSP**i**" device code "**j**" to "DISK**k**".

```
CABLE MSPi j DISKk
```

- Connect a cable from (*unit record processor*) "URP**i**" device code "**j**" to (*card reader*) "RDR**k**".

```
CABLE URPi j RDRk
```

- Connect a cable from "URP**i**" device code "**j**" to (*card punch*) "PUN**k**".

```
CABLE URPi j PUNk
```

- Connect a cable from "URP**i**" device code "**j**" to (*printer*) "PRT**k**".

```
CABLE URPi j PRTk
```

UNCABLE (U)

The "**UNCABLE**" command (*abbreviated* U) removes (or "*unstrings*") a simulated cable.

```
UNCABLE <device> <port/channel> <device>
```

Example

- Unstring the cable connecting "IOM**0**" channel "**12**" to "MSP**0**".

```
UNCABLE IOM0 12 MSP0
```

CABLE_RIPOUT

The "**CABLE_RIPOUT**" command (*alias* "**CABLE RIPOUT**") removes (or *unstrings*) ***all*** cables from the configuration.

Example

```
    CABLE RIPOUT
    CABLE_RIPOUT
```

CABLE_SHOW

The "**CABLE_SHOW**" command (*alias* "**CABLE SHOW**") prints the current cabling configuration in human readable form.

Example

```
sim> CABLE SHOW
SCU <--> IOM
   SCU port --> IOM port
    0    0       0    0
    0    1       1    0
    1    0       0    1
    1    1       1    1
    2    0       0    2
    2    1       1    2
    3    0       0    3
    3    1       1    3

SCU <--> CPU
   SCU port --> CPU port
    0    2       5    0
    0    3       4    0
    0    4       3    0
    0    5       2    0
    0    6       1    0
    0    7       0    0
    1    2       5    1
    1    3       4    1
    1    4       3    1
    1    5       2    1
    1    6       1    1
    1    7       0    1
    2    2       5    2
    2    3       4    2
    2    4       3    2
    2    5       2    2
    2    6       1    2
    2    7       0    2
    3    2       5    3
    3    3       4    3
    3    4       3    3
    3    5       2    3
    3    6       1    3
```

```
    3    7       0    3

IOM <--> controller
                ctlr       ctlr  chan
   IOM chan -->  idx  port  type  type      device      board    command
    0   10        0     0 MTP     PSI      0x4cf4c0   0x86b1c0   0x44a400
    0   11        0     0 IPC     PSI      0x4cf580   0x86c2c0   0x41a040
    0   12        0     0 MSP     PSI      0x4cf640   0x86ba40   0x41a040
    0   13        0     0 URP     PSI      0x4cf700   0x4ceb80   0x451ef0
    0   14        1     0 URP     PSI      0x4cf700   0x4cec08   0x451ef0
    0   15        2     0 URP     PSI      0x4cf700   0x4cec90   0x451ef0
    0   16        3     0 FNP     Direct   0x4cdd40   0x4cdf98   0x431000
    0   17        0     0 FNP     Direct   0x4cdd40   0x4cde00   0x431000
    0   18        1     0 FNP     Direct   0x4cdd40   0x4cde88   0x431000
    0   19        2     0 FNP     Direct   0x4cdd40   0x4cdf10   0x431000
    0   20        4     0 FNP     Direct   0x4cdd40   0x4ce020   0x431000
    0   21        5     0 FNP     Direct   0x4cdd40   0x4ce0a8   0x431000
    0   22        6     0 FNP     Direct   0x4cdd40   0x4ce130   0x431000
    0   23        7     0 FNP     Direct   0x4cdd40   0x4ce1b8   0x431000
    0   30        0     0 OPC     CPI      0x4ce680   0x4ce740   0x40e940
    0   40        3     0 URP     PSI      0x4cf700   0x4ced18   0x451ef0
    0   41        4     0 URP     PSI      0x4cf700   0x4ceda0   0x451ef0
    0   42        5     0 URP     PSI      0x4cf700   0x4cee28   0x451ef0
    0   43        1     0 OPC     CPI      0x4ce680   0x4ce7c8   0x40e940
    0   45        6     0 URP     PSI      0x4cf700   0x4ceeb0   0x451ef0
    0   46        7     0 URP     PSI      0x4cf700   0x4cef38   0x451ef0
    0   47        8     0 URP     PSI      0x4cf700   0x4cefc0   0x451ef0
    0   48        9     0 URP     PSI      0x4cf700   0x4cf048   0x451ef0
    1   10        0     1 MTP     PSI      0x4cf4c0   0x86b1c0   0x44a400
    1   11        0     1 IPC     PSI      0x4cf580   0x86c2c0   0x41a040
    1   12        0     1 MSP     PSI      0x4cf640   0x86ba40   0x41a040

controller <--> device
  MTP  dev_code --> TAPE   command
    0     1           1   0x44a400
    0     2           2   0x44a400
    0     3           3   0x44a400
    0     4           4   0x44a400
    0     5           5   0x44a400
    0     6           6   0x44a400
    0     7           7   0x44a400
    0     8           8   0x44a400
    0     9           9   0x44a400
    0    10          10   0x44a400
    0    11          11   0x44a400
    0    12          12   0x44a400
    0    13          13   0x44a400
    0    14          14   0x44a400
    0    15          15   0x44a400
    0    16          16   0x44a400
  IPC  dev_code --> DISK   command
    0     0           0   0x41a040
    0     1           1   0x41a040
    0     2           2   0x41a040
    0     3           3   0x41a040
```

```
    0     4           14   0x41a040
    0     5           15   0x41a040
    0     6           16   0x41a040
    0     7           17   0x41a040
    0     8           18   0x41a040
    0     9           19   0x41a040
    0    10           20   0x41a040
    0    11           21   0x41a040
    0    12           22   0x41a040
    0    13           23   0x41a040
    0    14           24   0x41a040
    0    15           25   0x41a040
  MSP  dev_code --> DISK   command
    0     1            4   0x41a040
    0     2            5   0x41a040
    0     3            6   0x41a040
    0     4            7   0x41a040
    0     5            8   0x41a040
    0     6            9   0x41a040
    0     7           10   0x41a040
    0     8           11   0x41a040
    0     9           12   0x41a040
    0    10           13   0x41a040
  URP  dev_code --> URP    command
    0     1            0   0x419900
    1     1            0   0x41b800
    2     1            0   0x44d440
    3     1            1   0x44d440
    4     1            2   0x44d440
    5     1            3   0x44d440
    6     1            1   0x419900
    7     1            2   0x419900
    8     1            1   0x41b800
    9     1            2   0x41b800
```

CABLE DUMP

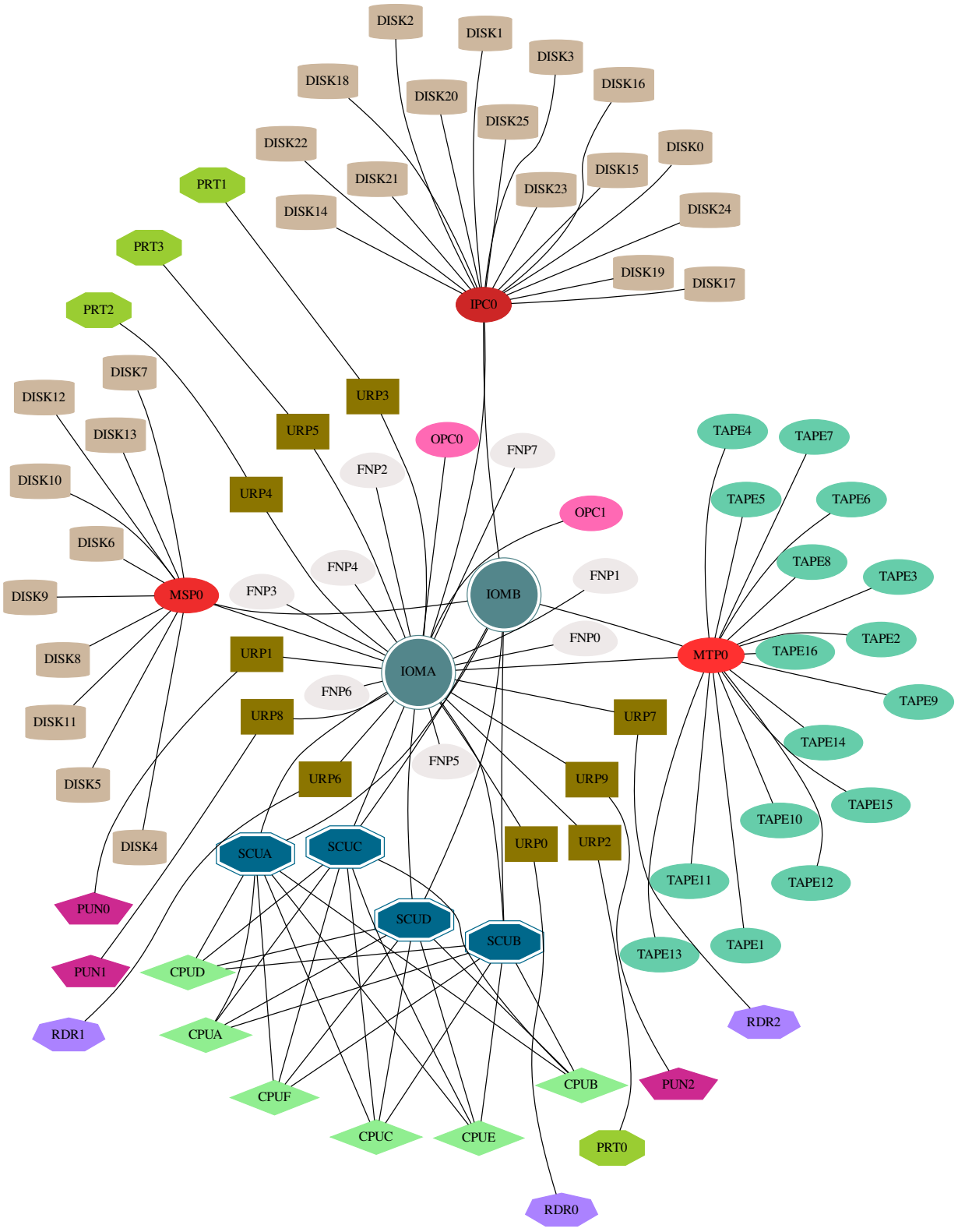
The "**CABLE DUMP**" command prints the current cabling configuration in great detail.

CABLE GRAPH

The "**CABLE GRAPH**" command prints the current cabling configuration in the "**DOT**" graph description language (suitable for rendering with *GraphViz*, etc).

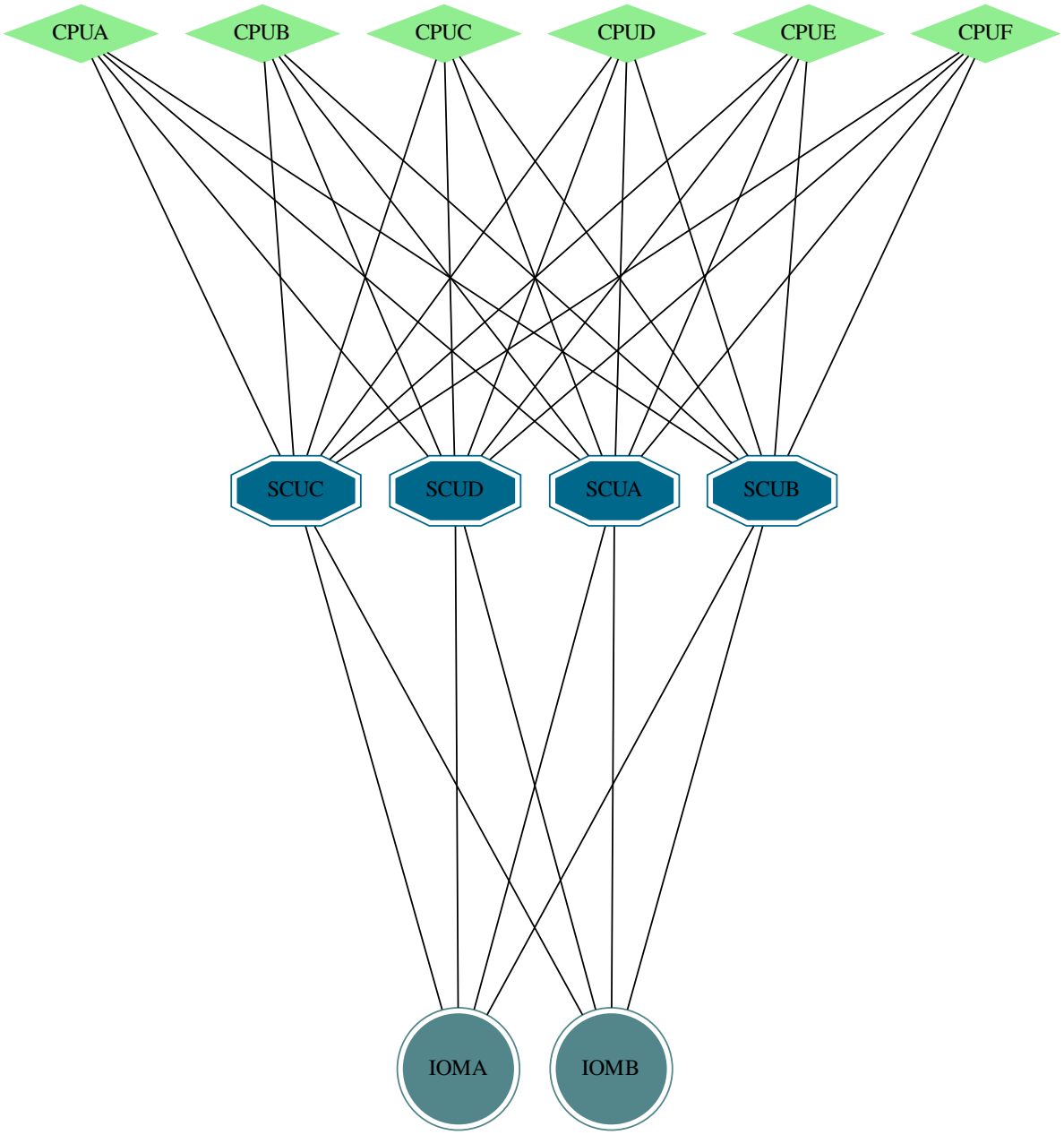
Complete Cabling Graph

The following is a **complete** cabling graph of the *default base system*:



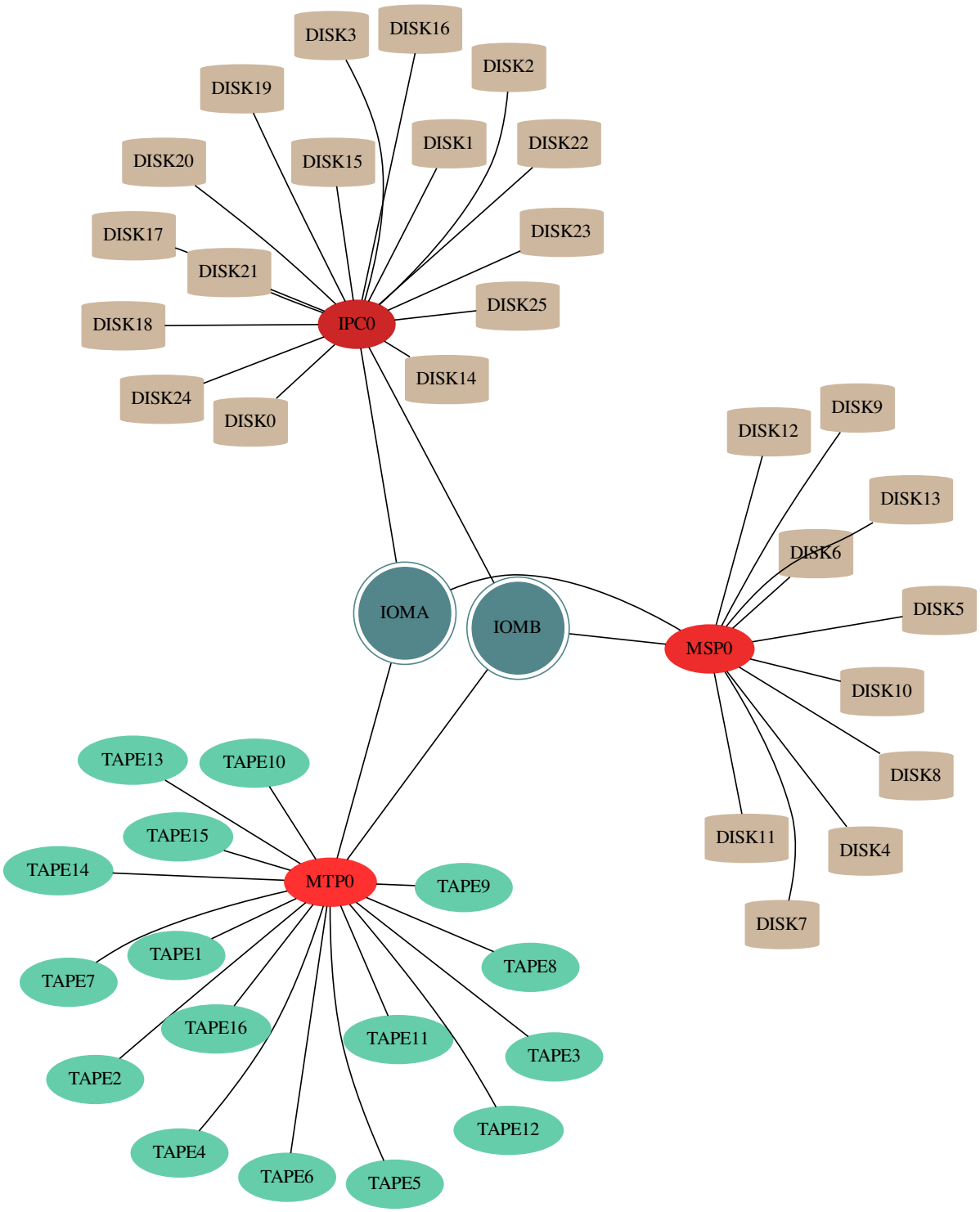
CPU / SCU / IOM Cabling Graph

The following graph shows the cabling configuration of the *default base system*'s "**CPU**", "**SCU**", and "**IOM**" devices:



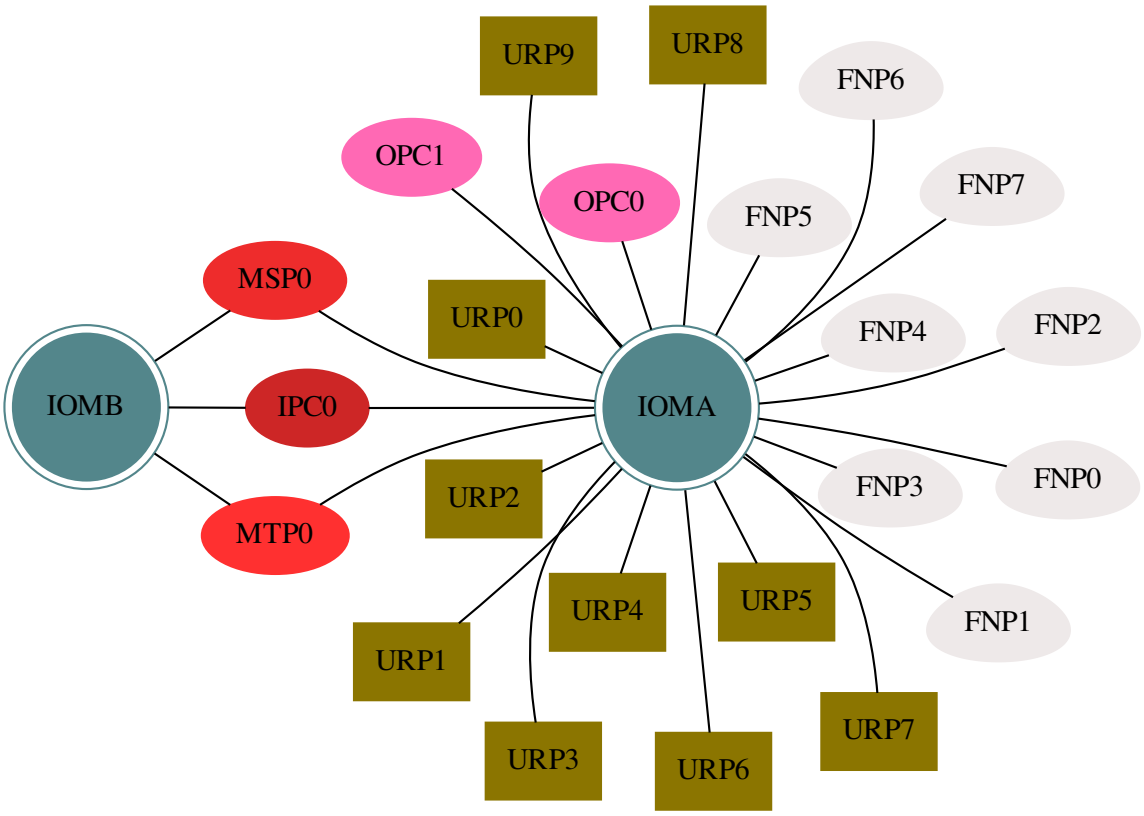
Storage Cabling Graph

The following graph shows the cabling configuration of the *default base system*'s **storage** devices:



Controller Cabling Graph

The following graph shows the cabling configuration of the *default base system*'s **controller** devices:



See the "**Simulator Defaults**" chapter for more details (including the full output of the "`CABLE DUMP`" command).

CALL

Control can be transferred to a labeled subroutine using `CALL`.

Example

```
        CALL routine
        BYE

        :routine
        ECHO routine called
        RETURN
```

CHECKPOLL

Set polling check rate (in polling intervals).

CLRAUTOINPUT

The `CLRAUTOINPUT` command clears the auto-input buffer for the primary operator operator console (OPC0).

- This is normally not required, but may be useful for developers or when writing complex scripts.

CLRAUTOINPUT2

The `CLRAUTOINPUT2` command clears the auto-input buffer for the secondary operator operator console (OPC1).

- Refer to the documentation for the `CLRAUTOINPUT` command for `CLRAUTOINPUT2` usage.

CONTINUE (CO)

The `CONTINUE` command (*abbreviated* `CONT` or `CO`) resumes execution (if execution was stopped, possibly due to hitting a breakpoint) at the current program counter without resetting any devices.

DEFAULT_BASE_SYSTEM

The `DEFAULT_BASE_SYSTEM` command restores the configuration of the simulator to startup defaults by executing the *default base system script*.

- For complete details including all commands executed by the *default base system script*, see the "**Default Base System Script**" section of the "**Simulator Defaults**" chapter of this manual.

DETACH (DET)

The `DETACH` (*abbreviation* `DET`) command breaks the association between a unit and its backing file or device:

```
        DETACH ALL              Detach all units
        DETACH <unit>           Detach specified unit
```

- **NOTE:** The `EXIT` command performs an automatic `DETACH ALL`.

DO

The simulator can invoke another script file with the "DO" command:

```
DO <filename> {arguments...}      execute commands in specified file
```

The "DO" command allows command files to contain substitutable arguments. The string "%n", where "n" is a number between "1" and "9", is replaced with argument "n" from the "DO" command line. (*i.e.* "%0", "%1", "%2", etc.). The string "%0" is replaced with "<filename>". The sequences "\%" and "\\" are replaced with the literal characters "%" and "\", respectively. Arguments with spaces must be enclosed in matching single or double quotation marks.

- **NOTE**: Nested "DO" commands are supported, up to ten invocations deep.

Switches

-v If the switch "-v" is specified, commands in the command file are echoed *before* they are executed.

-e If the switch "-e" is specified, command processing (including nested command invocations) will be aborted if any command error is encountered. (A simulation stop **never** aborts processing; use ASSERT to catch unexpected stops.) Without this switch, all errors except ASSERT failures will be ignored, and command processing will continue.

-o If the switch "-o" is specified, the ON conditions and actions from the calling command file will be inherited by the command file being invoked.

-q If the switch "-q" is specified, *quiet mode* will be explicitly enabled for the called command file, otherwise the *quiet mode* setting is inherited from the calling context.

ECHO

The ECHO command is a useful way of annotating command files. ECHO prints out its arguments to the console (and to any applicable log file):

```
ECHO <string>      Output string to console
```

NOTE: If no arguments are specified, ECHO prints a blank line. This may be used to provide spacing for console messages or log file output.

EVALUATE

The EVAL command evaluates a symbolic expression and returns the equivalent numeric value.

EXIT (QUIT, BYE)

The EXIT command (*synonyms* QUIT *and* BYE) exits the simulator, returning control to the host operating system.

FNPSERVERADDRESS

The FNPSERVERADDRESS command directs the simulator to bind the simulated FNP (*front-end network processor*) **TELNET** server to the specified "**<address>**" of the host system. The **TELNET** server answers incoming connections and presents a list of open communication channels to the user. If the FNPSERVERADDRESS command is not issued prior to FNP bootload, a default "**<address>**" of "`0.0.0.0`" is used (*i.e.* listening to all addresses). Only IPv4 addresses may be specified, using quad-dotted decimal notation.

```
    FNPSERVERADDRESS <address>
```

Example

- Listen on "`127.0.0.1`":

  ```
      FNPSERVERADDRESS 127.0.0.1
  ```

See also: FNPSERVERPORT.

FNPSERVERPORT

The FNPSERVERPORT command directs the simulator to listen for FNP (*front-end network processor*) **TELNET** server connections on the specified "**<port>**" of the host system. If the FNPSERVERPORT command is not issued prior to FNP bootload, a default "**<port>**" of "`6180`" is used.

```
    FNPSERVERPORT <port>
```

Example

- Listen on port "`6180`" (*the default port*):

  ```
      FNPSERVERPORT 6180
  ```

See also: FNPSERVERADDRESS.

FNPSTART

Directs the simulator to immediately start listening for FNP connections.

GO

The GO command does *not* reset devices, deposits its argument (if given) in the PC, and starts execution. If no argument is given, execution starts at the current PC (program counter).

GOTO

Commands in a command file execute in sequence until either an error trap occurs (when a command completes with an error status), or when an explict request is made to start command execution elsewhere with the GOTO command:

```
GOTO <label>
```

- Labels are lines in a command file which the first non-whitespace character is a "`:`".
- The target of a GOTO is the first matching label in the current DO command file which is encountered.

Example

The following example illustrates usage of the GOTO command (by creating an infinite loop):

```
:Label
:: This is a loop.
GOTO Label
```

IF

The IF command tests a simulator state condition and executes additional commands if the condition is true:

```
IF <Simulator State Expressions> commandtoprocess{; additionalcommand}...
```

Examples

The command file below might be used to bootstrap a hypothetical system that halts after the initial load from disk. The IF command can then be used to confirm that the load completed successfully by examining the CPU's "A" register for an expected value:

```
; Example INI file
BOOT
; A register contains error code; 0 = good boot
IF NOT A=0 echo Boot failed - Failure Code ; EX A; exit AFAIL
RUN
```

- In the above example, if the "A" register is *not* 0, the message "Boot failed - Failure Code" will be displayed, the contents of the "A" register will be displayed, and the command file will be aborted with an "Assertion failed" message. Otherwise, the command file will continue to bring up the system.

Conditional Expressions

The IF and ASSERT commands evaluate the following two different forms of conditional expressions.

Simulator State Expressions

The values of simulator registers can be evaluated with:

```
{NOT} {<dev>} <reg>|<addr>{<logical-op><value>}<conditional-op><value>
```

- If "<dev>" is not specified, CPU is assumed. "<reg>" is a register belonging to the indicated device.
- The "<addr>" is an address in the address space of the indicated device.
- The "<conditional-op>" and optional "<logical-op>" are the same as those used for "search specifiers" by the EXAMINE and DEPOSIT commands. The "<value>" is expressed in the radix specified for "<reg>", not in the radix for the device when referencing a register; when an address is referenced the device radix is used as the default.
- If "<logical-op>" and "<value>" are specified, the target register value is first altered as indicated. The result is then compared to the "<value>" via the "<conditional-op>".
  - If the result is *true*, the command(s) are executed before proceeding to the next line in the command file.
  - If the result is *false*, the next command in the command file is processed.

String Comparison Expressions

String Values can be compared with:

```
{-i} {NOT} "<string1>" <compare-op> "<string2>"
```

- The "-i" switch, if present, causes a comparison to be case insensitive.
- The "<string1>" and "<string2>" arguments are quoted string values which may have environment variables substituted as desired.
- The "<compare-op>" may be one of:

"**==**"	equal
"**EQU**"	equal
"**!=**"	not equal
"**NEQ**"	not equal
"**<**"	less than
"**LSS**"	less than
"**<=**"	less than or equal
"**LEQ**"	less than or equal
"**>**"	greater than
"**GTR**"	greater than
"**>=**"	greater than or equal
"**GEQ**"	greater than or equal

- **NOTE**: Comparisons are *generic*. This means that if both "<string1>" and "<string2>" are comprised of all numeric digits, then the strings are converted to numbers and a numeric comparison is performed. For example, the comparison '"+1" EQU "1"' evaluates to *true*.

LOAD (UNLOAD)

The LOAD and UNLOAD commands mount or unmount a disk or tape image and signals Multics.

LUF (NOLUF)

The LUF and NOLUF commands enable or disable normal LUF (*lockup fault*) handling.

MOUNT

Mount tape image and signal Multics

NEXT (N)

The NEXT command (*abbreviated* N) resumes execution at the current PC for one instruction, attempting to execute *through* subroutine calls. If the next instruction to be executed is *not* a subroutine call, then one instruction is executed.

ON

The ON command performs actions after a condition, or clears a condition.

```
          ON <condition> <action>  Perform action after condition
          ON <condition>           Clears action of specified condition
```

POLL

Set polling interval (in milliseconds)

PROCEED (IGNORE)

The PROCEED (or IGNORE) command does nothing. It is potentially useful as a placeholder for any ON action condition that should be explicitly ignored, allowing command file execution to continue without taking any specific action.

READY

Signal Multics that media is ready

RESET

The RESET command (*abbreviated* RE) resets a device or the entire simulator to a predefined condition. If the switch "-p" is specified, the device is reset to its initial power-on state:

```
          RESET                 resets all devices
          RESET -p              power-cycle all devices
          RESET ALL             resets all devices
          RESET <device>        resets the specified <device>
```

- Typically, RESET *aborts* in-progress I/O operations, *clears* any interrupt requests, and returns the device to a quiescent state.
- It does **NOT** clear the main memory or affect associated I/O connections.

RETURN

The RETURN command causes the current procedure call to be restored to the calling context, possibly returning a specific return status. If no return status is specified, the return status from the last command executed will be returned. The calling context may have ON traps defined which may redirect command flow in that context.

```
          RETURN                return from command file with last command status
          RETURN {-Q} <status>  return from command file with specific status
```

- The status return can be any numeric value or one of the standard SCPE_ condition names.
- The "-Q" switch on the RETURN command will cause the specified status to be returned, but normal error status message printing to be suppressed.

Condition Names

The available standard SCPE_ condition names and their meanings are:

Name	Meaning	Name	Meaning
NXM	Address space exceeded	UNATT	Unit not attached
IOERR	I/O error	CSUM	Checksum error
FMT	Format error	NOATT	Unit not attachable
OPENERR	File open error	MEM	Memory exhausted
ARG	Invalid argument	STEP	Step expired
UNK	Unknown command	RO	Read only argument
INCOMP	Command not completed	STOP	Simulation stopped
EXIT	Goodbye	TTIERR	Console input I/O error

Name	Meaning	Name	Meaning
TTOERR	Console output I/O error	EOF	End of file
REL	Relocation error	NOPARAM	No settable parameters
ALATT	Unit already attached	TIMER	Hardware timer error
SIGERR	Signal handler setup error	TTYERR	Console terminal setup error
NOFNC	Command not allowed	UDIS	Unit disabled
NORO	Read only operation not allowed	INVSW	Invalid switch
MISVAL	Missing value	2FARG	Too few arguments
2MARG	Too many arguments	NXDEV	Non-existent device
NXUN	Non-existent unit	NXREG	Non-existent register
NXPAR	Non-existent parameter	NEST	Nested DO command limit exceeded
IERR	Internal error	MTRLNT	Invalid magtape record length
LOST	Console Telnet connection lost	TTMO	Console Telnet connection timed out
STALL	Console Telnet output stall	AFAIL	Assertion failed
INVREM	Invalid remote console command		

REWIND

Rewind tape

RUN (RU)

The RUN command (*abbreviated* RU) resets all devices, deposits its argument, if given, in the PC (program counter), and starts execution. If no argument is given execution starts at the current PC.

SEGLDR

Segment Loader

SET

Logging

Interactions with the simulator session can be recorded to a log file.

```
SET LOG log_file            Specify the log destination
                                 (STDOUT, DEBUG, or filename)
SET NOLOG                   Disables any currently active logging
```

Switches

-N By default, log output is written at the *end* of the specified log file. A new log file can created if the "-N" switch is used on the command line.

-B By default, log output is written in *text* mode. The log file can be opened for *binary* mode writing if the "-B" switch is used on the command line.

Debug Messages

```
SET DEBUG debug_file        Specify the debug destination
                                 (STDOUT, STDERR, LOG, or filename)
SET NODEBUG                 Disables any currently active debug output
```

Switches

Debug message output contains a timestamp which indicates the number of simulated instructions which have been executed prior to the debug event.

Debug message output can be enhanced to contain additional, potentially useful information.

NOTE: If neither "-T" or "-A" is specified, "-T" is implied.

-T The "-T" switch causes debug output to contain a time of day displayed as `hh:mm:ss.msec`.

-A The "-A" switch causes debug output to contain a time of day displayed as `seconds.msec`.

-R The "-R" switch causes timing to be relative to the start of debugging.

-P The "-P" switch adds the output of the PC (program counter) to each debug message.

-N The "-N" switch causes a new (empty) file to be written to. (The default is to append to an existing debug log file).

-D The "-D" switch causes data blob output to also display the data as **RADIX-50** characters.

-E The "-E" switch causes data blob output to also display the data as "**EBCDIC**" characters.

Environment Variables

```
SET ENVIRONMENT NAME=val    Set environment variable
SET ENVIRONMENT NAME        Clear environment variable
```

Command Status Trap Dispatching

```
SET ON                      Enables error checking command execution
SET NOON                    Disables error checking command execution
SET ON INHERIT              Enables inheritance of ON state and actions
SET ON NOINHERIT            Disables inheritance of ON state and actions
```

Command Execution Display

```
SET VERIFY                  Enables display of processed script commands
SET VERBOSE                 Enables display of processed script commands
SET NOVERIFY                Disables display of processed script commands
SET NOVERBOSE               Disables display of processed script commands
```

Command Error Status Display

```
SET MESSAGE                 Re-enables display of script error messages
SET NOMESSAGE               Disables display of script error messages

SET LOCALOPC                Enables local operator console
SET NOLOCALOPC              Disables local operator console
```

Command Output Display

```
SET QUIET                   Disables suppression of some messages
SET NOQUIET                 Re-enables suppression of some messages
```

Command Prompt

```
SET PROMPT "string"         Sets an alternate simulator prompt string
```

Device and Unit Settings

```
SET <dev> OCT|DEC|HEX       Set device display radix
SET <dev> ENABLED           Enable device
SET <dev> DISABLED          Disable device
SET <dev> DEBUG{=arg}       Set device debug flags
SET <dev> NODEBUG={arg}     Clear device debug flags
SET <dev> arg{,arg...}      Set device parameters
```

```
SET <unit> ENABLED          Enable unit
SET <unit> DISABLED         Disable unit
SET <unit> arg{,arg...}     Set unit parameters
HELP <dev> SET              Displays any device specific SET commands
```

CPU Configuration

L68 (DPS8M)

"**L68**" configures the "CPU" unit specified to simulate a **Level 68** processor, where "1" or "enable" configures the CPU as a **Level 68** and "0" or "disable" configures the CPU as a **DPS-8/M**.

"**DPS8M**" configures the "CPU" unit specified to simulate a **DPS-8/M** processor, where "1" or "enable" configures the CPU as a **DPS-8/M** and "0" or "disable" configures the CPU as a **Level 68**:

```
    L68=<0 or 1>
    L68=<disable or enable>
    DPS8M=<0 or 1>
    DPS8M=<disable or enable>
```

Examples

- Configure CPU**0** to simulate a **Level 68** processor. (The default is to simulate a **DPS-8/M**):

```
    SET CPU0 L68=1
    SET CPU0 DPS8M=0
```

- Configure CPU**1** to simulate a **DPS-8/M** processor:

```
    SET CPU0 L68=0
    SET CPU0 DPS8M=1
```

RESET

"**RESET**" will reset the specified "CPU" unit (or *all* "CPU" units when no unit is explicitly specified) when passed an argument of "1".

```
    RESET=<1>
```

Examples

- Reset all CPUs:

```
    SET CPU RESET=1
```

- Reset CPU**0**:

```
    SET CPU0 RESET=1
```

INITIALIZE

"**INITIALIZE**" will initialize the specified "CPU" unit (or *all* "CPU" units when no unit is explicitly specified) when passed an argument of "1".

```
    INITIALIZE=<1>
```

Examples

- Initialize all CPUs:

```
    SET CPU INITIALIZE=1
```

- Initialize CPU**0**:

```
    SET CPU0 INITIALIZE=1
```

INITIALIZEANDCLEAR (IAC)

"**INITIALIZEANDCLEAR**" (*abbreviated* "**IAC**") will initialize the specified "CPU" unit and clear it's state (or initialize *all* "CPU" units and clear their states when no unit is explicitly specified) when passed an argument of "1".

```
    INITIALIZEANDCLEAR=<1>
    IAC=<1>
```

Examples

- Initialize and clear CPUs:

```
    SET CPU INITIALIZEANDCLEAR=1
```

- Initialize and clear CPU**0**:

```
    SET CPU0 IAC=1
```

NUNITS

"**NUNITS**" configures the number of "CPU" units.

```
    NUNITS=<n>
```

Example

- Set the number of "CPU" units to "6" (*the default number*):

```
    SET CPU NUNITS=6
```

KIPS

"**KIPS**" configures the global CPU lockup fault timer scaling factor.

```
KIPS=<n>
```

When simulating multiple processors, occasional spurious "*Fault in idle process*" messages may be seen when additional CPU's are brought online. This is caused by a lockup fault occurring while the Multics hardcore is waiting on a flag to be set by the CPU being brought online.

Latency guarantees on the original hardware prevent this from ever occurring when the hardware is properly functioning. Unfortunately, without introducing a strict hard-realtime requirements for simulated operations, such as requiring users run the simulator under control of a capable RTOS, or perhaps on a completely idle system without other processes active, there is no practical way to provide Multics these precision timing guarantees.

When running on original hardware, the lockup fault would be triggered by an independent watchdog timer that monitors the CPU's polling of interrupts. If interrupts would be inhibited for too long, this watchdog timer would run out and trigger the lockup fault. Multics rightfully considers a lockup fault in hardcore as indicative of a serious (*possibly fatal*) problem with the faulting CPU.

Unfortunately, what is a clear sign of hardware trouble on real hardware (*i.e* a CPU that is struggling to get up to speed, or running at fluctuating speeds) is the normal, expected, and unavoidable reality when the simulator is running on a general purpose system.

Configuring the "KIPS" scaling factor loosens this hard realtime requirement and avoids spurious lockup faults.

Note that if "KIPS" is set to a large value, a genuine hardcore error or broken userspace process will still be detected, although it will take slightly longer for the lockup fault to occur.

Example

- Set the global CPU lockup fault scaling factor for 8 MIPS:

```
SET CPU KIPS=8000
```

It is not expected that the "KIPS" value will require modification by most users under normal operating circumstances. See also: "**STALL**".

STALL

"**STALL**" configures a stall (*i.e.* a delay) of "**<iterations>**" when the IC reaches the address specified by "**<segno>**" and "**<offset>**", identified by "**<num>**".

```
STALL=<num>=<segno>:<offset>=<iterations>
```

- "**<num>**" is a number allowing for the identification (and modification) of multiple stall points.
- "**<segno>**" is the segment number of the Multics segment at which to stall.
- "**<offset>**" is the offset within the segment specified by "**<segno>**" at which to stall.
- "**<iterations>**" is a positive integer representing the number of iterations to loop.

Example

- Configure two stalls, #1 for 250 iterations at "0132:011227", and #2 for 1250 iterations at "0122:052137":

```
SET CPU STALL=1=0132:011227=250
SET CPU STALL=2=0122:052137=1250
```

DEBUG (NODEBUG)

"**DEBUG**" enables CPU debugging and/or enables specified debugging options.

"**NODEBUG**" disables CPU debugging and/or disables specified debugging options.

```
DEBUG                                ; Enables debugging
NODEBUG                              ; Disables debugging
DEBUG=<list-of-options>              ; Enables specified debugging options
NODEBUG=<list-of-options>            ; Disables specified debugging options
```

- The "**<list-of-options>**" is a semicolon ("**;**") delimited list of one or more of the following options:

TRACE	**TRACEEXT**	**MESSAGES**	**REGDUMPAQI**	**REGDUMPIDX**
REGDUMPPR	**REGDUMPPPR**	**REGDUMPDSBR**	**REGDUMPFLT**	**REGDUMP**
ADDRMOD	**APPENDING**	**NOTIFY**	**INFO**	**ERR**
WARN	**DEBUG**	**ALL**	**FAULT**	**INTR**
CORE	**CYCLE**	**CAC**	**FINAL**	**AVC**

Examples

- Enable the debugging options "TRACE" and "FAULT" for all "CPU" units:

```
SET CPU DEBUG=TRACE;FAULT
```

- Disable all debugging options for all "CPU" units:

```
SET CPU NODEBUG
```

- Enable the debugging option "CYCLE" for CPU**0** and disable all debugging options for all other "CPU" units:

```
SET CPU NODEBUG
SET CPU0 DEBUG=CYCLE
```

CONFIG

The following "**CPU**" configuration options are associated with a specified "CPU" unit (*i.e.* CPU**n**) and are configured using the "SET" command (*i.e.* "**SET CPUn CONFIG**"):

```
SET CPUn CONFIG=<set-option>
```

- The following "**<set-option>**"'s are available, in the form of "**<option>=<value>**", for example:

```
SET CPUn CONFIG=<option>=<value>
```

FAULTBASE

"**FAULTBASE**" configures the fault base of the specified "CPU" unit:

```
FAULTBASE=<value>
```

Example

- For Multics operation, configure the "**<value>**" to "**Multics**" (for CPU**0**):

```
SET CPU0 CONFIG=FAULTBASE=Multics
```

NUM

"**NUM**" configures the CPU number of the specified "CPU" unit:

```
NUM=<n>
```

Example

- Set the CPU number of CPU**0** to **1**.

```
SET CPU0 CONFIG=NUM=1
```

DATA

"**DATA**" configures the CPU switches of the specified "CPU" unit:

```
DATA=<word>
```

Example

- Set CPU**0** switches to "024000717200" (*the default value*).

```
SET CPU0 CONFIG=DATA=024000717200
```

See Also

Refer to *GB61-01* **Operators Guide, Appendix A** for more details.

STOPNUM

"**STOPNUM**" configures the CPU switches of the specified "CPU" unit so that Multics will stop during boot at the check stop specified by "**<n>**".

```
    STOPNUM=<n>
```

Example

- Set CPU**0** switches so Multics will stop during boot at check stop #2030.

```
    SET CPU0 CONFIG=STOPNUM=2030
```

MODE

"**MODE**" configures the operating mode of specified "CPU" unit as indicated by "**<value>**":

```
    MODE=<value>
```

- The supported "**<value>**"'s are:

<value>	Operating mode
"**0**" or "**GCOS**"	**GCOS** operating mode
"**1**" or "**Multics**"	**Multics** operating mode

Examples

- Set the operating mode of CPU**1** to **GCOS** operating mode:

```
    SET CPU1 CONFIG=MODE=0
```

- Set the operating mode of CPU**0** to **Multics** operating mode:

```
    SET CPU0 CONFIG=MODE=Multics
```

SPEED

"**SPEED**" configures the CPU speed setting of the specified CPU unit, where an "**<identifier>**" of "**0**" indicates a **DPS-8/M Model 70**.

```
    SPEED=<identifier>
```

Example

- Configure CPU**0** with a speed setting of "**0**", indicating a **DPS-8/M Model 70** (*the default setting*):

```
    SET CPU0 CONFIG SPEED=0
```

PORT

"**PORT**" selects the CPU port "**<p>**" for which subsequent "SET CPUn CONFIG=" configuration commands apply:

```
    PORT=<p>
```

Example

- Configure ASSIGNMENT, INTERLACE, ENABLE, INIT_ENABLE, and STORE_SIZE for port "**A**" on CPU**0**:

```
    SET CPU0 CONFIG=PORT=A
    SET CPU0   CONFIG=ASSIGNMENT=0
    SET CPU0   CONFIG=INTERLACE=0
    SET CPU0   CONFIG=ENABLE=1
    SET CPU0   CONFIG=INIT_ENABLE=1
    SET CPU0   CONFIG=STORE_SIZE=4M
```

ASSIGNMENT

"**ASSIGNMENT**" configures the CPU port assignment to "**<a>**" of "CPU" unit "**n**"'s CPU port "**x**", as specified by a previous "**SET CPUn CONFIG=PORT=x**" command:

```
    ASSIGNMENT=<a>
```

Example

- Configure the CPU port assignment of CPU**1**'s port "**B**" to "**2**":

```
    SET CPU1 CONFIG=PORT=B
    SET CPU1   CONFIG=ASSIGNMENT=2
```

INTERLACE

"**INTERLACE**" configures the position of the *interlace* switch for the currently selected CPU port:

```
    INTERLACE=<0, 1, 2>
```

Example

- Configure the position of the *interlace* switch for the currently selected CPU port of CPU**0** to "**1**":

```
    SET CPU0 CONFIG=INTERLACE=1
```

ENABLE

"**ENABLE**" configures whether the currently selected CPU port is enabled, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
    ENABLE=<0 or 1>
    ENABLE=<disable or enable>
```

INIT_ENABLE

"**INIT_ENABLE**" configures whether *init* is enabled for the currently selected CPU port, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
    INIT_ENABLE=<0 or 1>
    INIT_ENABLE=<disable or enable>
```

STORE_SIZE

"**STORE_SIZE**" configures the size of memory (in *megawords*) for the currently selected CPU port:

```
    STORE_SIZE=<num>M
```

Example

- Configure the memory size of the currently selected CPU port on CPU**0** to "**4**" megawords:

```
    SET CPU0 CONFIG=STORE_SIZE=4M
```

ENABLE_CACHE

"**ENABLE_CACHE**" configures whether the CPU cache is enabled for a CPU with CPU cache, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
    ENABLE_CACHE=<0 or 1>
    ENABLE_CACHE=<disable or enable>
```

Example

- Configure CPU**0** to enable the CPU cache, if installed (*the default setting*):

```
    SET CPU0 CONFIG=ENABLE_CACHE=1
```

SDWAM

"**SDWAM**" configures whether SDW associative memory is enabled, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
SDWAM=<0 or 1>
SDWAM=<disable or enable>
```

Example

- Configure CPU**0** to enable SDW associative memory (*the default setting*):

```
SET CPU0 CONFIG=SDWAM=1
```

PTWAM

"**PTWAM**" configures whether PTW associative memory is enabled, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
SDWAM=<0 or 1>
SDWAM=<enable or disable>
```

Example

- Configure CPU**0** to enable PTW associative memory (*the default setting*):

```
SET CPU0 CONFIG=PTWAM=1
```

DIS_ENABLE

"**DIS_ENABLE**" configures whether DIS (*delay until interrupt serviced*) handling is enabled, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
DIS_ENABLE=<0 or 1>
DIS_ENABLE=<disable or enable>
```

Example

- Configure CPU**0** to enable DIS handling (*the default setting*):

```
SET CPU0 CONFIG=DIS_ENABLE=1
```

STEADY_CLOCK

"**STEADY_CLOCK**" configures whether the *steady CPU clock* should be enabled, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
STEADY_CLOCK=<0 or 1>
STEADY_CLOCK=<disable or enable>
```

Example

- Configure CPU**0** with the *steady CPU clock*:

```
SET CPU0 CONFIG=STEADY_CLOCK=1
```

HALT_ON_UNIMPLEMENTED

"**HALT_ON_UNIMPLEMENTED**" configures whether the simulator should halt when the specified "CPU" unit encounters an unimplemented instruction, where "0" or "disable" is disabled and "1" or "enable" is enabled.

```
HALT_ON_UNIMPLEMENTED=<0 or 1>
HALT_ON_UNIMPLEMENTED=<disable or enable>
```

Example

- Configure CPU**0** to **not** halt when encountering an unimplemented instruction (*the default setting*):

```
SET CPU0 CONFIG=HALT_ON_UNIMPLEMENTED=0
```

ENABLE_WAM

"**ENABLE_WAM**" configures whether WAM (*word associative memory*) is enabled, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
ENABLE_WAM=<0 or 1>
ENABLE_WAM=<disable or enable>
```

Example

- Configure CPU**0** to disable word associative memory (*the default setting*):

```
SET CPU0 CONFIG=ENABLE_WAM=0
```

REPORT_FAULTS

"**REPORT_FAULTS**" configures whether fault reporting is enabled, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
REPORT_FAULTS=<0 or 1>
REPORT_FAULTS=<disable or enable>
```

Example

- Configure CPU**0** to disable fault reporting (*the default setting*):

```
    SET CPU0 CONFIG=REPORT_FAULTS=0
```

TRO_ENABLE

"**TRO_ENABLE**" configures whether timer run-off (*TRO*) is enabled, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
    TRO_ENABLE=<0 or 1>
    TRO_ENABLE=<disable or enable>
```

Example

- Configure CPU**0** to enable TRO (*the default setting*):

```
    SET CPU0 CONFIG=TRO_ENABLE=1
```

Y2K

"**Y2K**" configures whether the Year 2000 compatibility workaround is enabled, where "0" or "disable" is disabled and "1" or "enable" is enabled.

- The Year 2000 compatibility workaround modifies the system clock to enable older Multics releases (*i.e.* MR12.3 and MR12.5) that are *not* Y2K-compliant to be booted and used without modification.

```
    Y2K=<0 or 1>
    Y2K=<disable or enable>
```

Example

- Configure the system to enable the *Year 2000 compatibility workaround*:

```
    SET CPU CONFIG=Y2K=1
```

DRL_FATAL

"**DRL_FATAL**" configures whether a DRL fault is fatal (stopping the simulator), where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
    DRL_FATAL=<0 or 1>
    DRL_FATAL=<disable or enable>
```

Example

- Configure CPU**0** to not treat DRL faults as fatal (*the default setting*):

```
SET CPU0 CONFIG=DRL_FATAL=0
```

USEMAP

"**USE_MAP**" configures whether to enable mapping, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
USEMAP=<0 or 1>
USEMAP=<disable or enable>
```

Example

- Configure CPU**0** to disable mapping (*the default setting*):

```
SET CPU0 CONFIG=USEMAP=0
```

ADDRESS

"**ADDRESS**" configures the CPU address to the specified "**<word>**":

```
ADDRESS=<word>
```

Example

- Configure CPU**0** address to "000000000000" (*the default setting*):

```
SET CPU0 CONFIG=ADDRESS=000000000000
```

See also

Refer to *GB61-01* **Operators Guide, Appendix A** for more details.

PROM_INSTALLED

"**PROM_INSTALLED**" configures whether a CPU identification PROM is installed, where "0" or "disable" is *not installed* and "1" or "enable" is *installed*:

```
PROM_INSTALLED=<0 or 1>
PROM_INSTALLED=<disable or enable>
```

Example

- Configure CPU**0** to install a CPU identification PROM (*the default setting for DPS-8/M processors*):

```
SET CPU0 CONFIG=PROM_INSTALLED=1
```

HEX_MODE_INSTALLED

"**HEX_MODE_INSTALLED**" configures whether the hexadecimal floating point (**HFP**) CPU option is installed, where "0" or "disable" is *not installed* and "1" or "enable" is *installed*:

```
HEX_MODE_INSTALLED=<0 or 1>
HEX_MODE_INSTALLED=<disable or enable>
```

Example

- Configure CPU**0** to install the hexadecimal floating point (**HFP**) CPU option:

```
SET CPU0 CONFIG=HEX_MODE_INSTALLED=1
```

CACHE_INSTALLED

"**CACHE_INSTALLED**" configures whether the CPU cache option is installed, where "0" or "disable" is *not installed* and "1" or "enable" is *installed*:

```
CACHE_INSTALLED=<0 or 1>
CACHE_INSTALLED=<disable or enable>
```

Example

- Configure CPU**0** to install the CPU cache option (*the default setting*):

```
SET CPU0 CONFIG=CACHE_INSTALLED=1
```

CLOCK_SLAVE_INSTALLED

"**CLOCK_SLAVE_INSTALLED**" configures whether the CPU clock slave is installed, where "0" or "disable" is *not installed* and "1" or "enable" is *installed*:

```
CLOCK_SLAVE_INSTALLED=<0 or 1>
CLOCK_SLAVE_INSTALLED=<disable or enable>
```

Example

- Configure CPU**0** to install the CPU clock slave (*the default setting*):

```
SET CPU0 CONFIG=CLOCK_SLAVE_INSTALLED=1
```

ENABLE_EMCALL

"**ENABLE_EMCALL**" configures whether *emcall* is enabled, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
ENABLE_EMCALL=<0 or 1>
ENABLE_EMCALL=<disable or enable>
```

Example

- Configure CPU**0** to disable *emcall*:

```
SET CPU0 CONFIG=ENABLE_EMCALL=0
```

ISOLTS_MODE

"**ISOLTS_MODE**" configures the CPU in such a way that facilitates running *ISOLTS* diagnostics, where "0" or "disable" is disabled and "1" or "enable" is enabled:

```
ISOLTS_MODE=<0 or 1>
ISOLTS_MODE=<disable or enable>
```

Example

- Configure CPU**0** for *ISOLTS* testing:

```
SET CPU0 CONFIG=ISOLTS_MODE=1
```

NODIS

"**NODIS**" configures whether normal CPU initial DIS state is disabled, where "1" or "enable" enables the disabling of the normal state and "0" or "disable" disables the disabling the normal state if such disabling was previously enabled:

```
NODIS=<0 or 1>
NODIS=<disable or enable>
```

Example

- Configure CPU**0** to disable the normal initial DIS state:

```
SET CPU0 CONFIG=NODIS=1
```

L68_MODE

"**L68_MODE**" configures the "CPU" unit specified to simulate a **Level 68** processor, where "1" or "enable" configures the CPU as a **Level 68** and "0" or "disable" configures the CPU as a **DPS-8/M**.

```
L68_MODE=<0 or 1>
L68_MODE=<disable or enable>
```

Examples

- Configure CPU**0** to simulate a **Level 68** processor. (The default is to simulate a **DPS-8/M**):

```
SET CPU0 CONFIG=L68_MODE=1
```

IOM Configuration

NUNITS

"**NUNITS**" configures the number of "IOM" units.

```
NUNITS=<n>
```

Example

- Set the number of "IOM" units to "2" (*the default number*):

```
SET IOM NUNITS=2
```

DEBUG (NODEBUG)

- TBD

RESET

- TBD

CONFIG

The following "**IOM**" configuration options are associated with a specified "IOM" unit (*i.e.* "IOM**n**") and are configured using the "SET" command (*i.e.* "**SET IOMn CONFIG**"):

```
SET IOMn CONFIG=<set-option>
```

- The following "**<set-option>**"'s are available, in the form of "**<option>=<value>**", for example:

```
SET IOMn CONFIG=<option>=<value>
```

PORT

"**PORT**" selects the IOM port "**<p>**" for which subsequent "SET IOMn CONFIG=" configuration commands apply:

```
    PORT=<p>
```

Example

- Select port "**0**" of `IOM0` for subsequent "`SET IOM0 CONFIG=`" configuration commands:

```
    SET IOM0 CONFIG=PORT=0
```

ADDR

* TBD (Address of the port)

INTERLACE

* TBD (Interlace switch setting)

ENABLE

* TBD (Port enabled or disabled)

INITENABLE

* TBD (Init enabled or disabled for port)

HALFSIZE

* TBD (Halfsize setting)

STORE_SIZE

* TBD (Size of memory - one of 32 64 128 256 512 1024 2048 4096 32K 64K 128K 256K 512K 1024K 2048K 4096K 1M 2M 4M, 4M is default.)

MODEL

"**`MODEL`**" configures the IOM model, where "**`<model>`**" is **`iom`** or **`imu`**.

```
MODEL=<model>
```

Example

- Set the model of **IOM0** to "`iom`" (*the default model*):

```
SET IOM0 CONFIG=MODEL=iom
```

OS

"**`OS`**" configures the allowed operating system for the `IOM` unit specified, where "**`<os>`**" is **`gcos`**, **`gcosext`**, or **`multics`**.

```
OS=<os>
```

Example

- Set the allowed operating system mode for **IOM0** to **`multics`** (*the default allowed mode*):

```
SET IOM0 CONFIG=MODE=multics
```

BOOT

"**`BOOT`**" configures the device to boot from for the `IOM` unit specified, where "**`<value>`**" is **`disk`** or **`tape`**.

```
BOOT=<value>
```

Example

- Set **IOM0** to boot from "`tape`" (*the default boot device setting*):

```
SET IOM0 CONFIG=BOOT=tape
```

IOM_BASE

"**`IOM_BASE`**" configures the base address for the `IOM` unit specified, where "**`<base_value>`**" is **`Multics`** (`014`), **`Multics1`** (`014`), **`Multics2`** (`020`), **`Multics3`** (`024`), or **`Multics4`** (`030`).

```
IOM_BASE=<base_value>
```

Example

- Set the base address of **IOM0** to **`Multics`** (`014`) (*the default base address*):

```
SET IOM0 CONFIG=IOM_BASE=Multics
```

MULTIPLEX_BASE

"**MULTIPLEX_BASE**" configures the multiplex base address for the IOM unit specified, where "**<n>**" is "0120" (for **IOM0**) or "0121" (for **IOM1**).

```
    MULTIPLEX_BASE=<n>
```

Example

- Set the multiplex base address of **IOM0** to 0120 (*the default multiplex base address*):

```
    SET IOM0 CONFIG=MULTIPLEX_BASE=0120
```

TAPECHAN

"**TAPECHAN**" configures the default tape channel for the IOM unit specified:

```
    TAPECHAN=<n>
```

Example

- Set the default tape channel of **IOM0** to 012 (*the default tape channel setting*):

```
    SET IOM0 CONFIG=TAPECHAN=012
```

CARDCHAN

"**CARDCHAN**" configures the default card channel for the IOM unit specified:

```
    CARDCHAN=<n>
```

Example

- Set the default card channel of **IOM0** to 011 (*the default card channel setting*):

```
    SET IOM0 CONFIG=CARDCHAN=011
```

SCUPORT

"**SCUPORT**" configures which port on the SCU the IOM unit specified will be connected to.

```
    SCUPORT=<n>
```

Example

- Set the SCU port of **IOM0** to 0 (*the default port*):

```
SET IOM0 CONFIG=SCUPORT=0
```

SCU Configuration

DEBUG (NODEBUG)

- TBD

NUNITS

"**NUNITS**" configures the number of "SCU" units.

```
NUNITS=<n>
```

Example

- Set the number of SCU units to "4" (*the default number*):

```
SET SCU NUNITS=6
```

RESET

- TBD

CONFIG

The following "**SCU**" configuration options are associated with a specified "SCU" unit (*i.e.* "SCU**n**") and are configured using the "SET" command (*i.e.* "**SET SCUn CONFIG**"):

```
SET SCUn CONFIG=<set-option>
```

- The following "**<set-option>**"'s are available, in the form of "**<option>=<value>**", for example:

```
SET SCUn CONFIG=<option>=<value>
```

MODE

* TBD (0 or 1)

MASKA

* TBD (7 for SCU0 by default, and “off” for other SCUs.)

MASKB

* TBD (“off” for all SCUs by default)

PORT0

* TBD (0 or 1)

PORT1

* TBD (0 or 1)

PORT2

* TBD (0 or 1)

PORT3

* TBD (0 or 1)

PORT4

* TBD (0 or 1)

PORT5

* TBD (0 or 1)

PORT6

* TBD (0 or 1)

PORT7

* TBD (0 or 1)

LWRSTORESIZE

* TBD (32 64 128 256 512 1024 2048 4096 32K 64K 128K 256K 512K 1024K 2048K 4096K 1M 2M 4M, 4M is default)

CYCLIC

* TBD (0040 by default)

NEA

* TBD (0200 by default)

ONL

* TBD (014 by default)

INT

* TBD (0 or 1)

LW

* TBD (0 or 1)

ELAPSED_DAYS

* TBD (0 or 1)

STEADY_CLOCK

* TBD (0 or 1, disable or enable)

BULLET_TIME

* TBD (0 or 1, disable or enable)

Y2K

* TBD (0 or 1, disable or enable)

TAPE Configuration

DEBUG (NODEBUG)

- TBD

DEFAULT_PATH

- TBD

ADD_PATH

- TBD

CAPACITY_ALL

- TBD

CAPACITY

- TBD

REWIND

- TBD

READY

- TBD

NAME

"**NAME**" configures the device name of the specified "TAPE" unit.

```
    NAME=<name>
```

Example

- Set the device name of "TAPE**1**" to "tapa_01" (*the default device name*).

```
    SET TAPE1 NAME=tapa_01
```

NUNITS

"**NUNITS**" configures the number of "TAPE" units.

```
    NUNITS=<n>
```

Example

- Set the number of "TAPE" units to "16" (*the default number*):

```
    SET TAPE NUNITS=16
```

MTP Configuration

DEBUG (NODEBUG)

- TBD

NAME

"**NAME**" configures the device name of the specified "MTP" unit.

```
    NAME=<name>
```

Example

- Set the device name of **MTP0** to "MTP0" (*the default name*).

```
    SET MTP0 NAME=MTP0
```

NUNITS

"**NUNITS**" configures the number of "MTP" units.

```
NUNITS=<n>
```

Example

- Set the number of "MTP" units to "1" (*the default number*):

```
SET MTP NUNITS=6
```

BOOT_DRIVE

- TBD

CONFIG

The following "**MTP**" configuration options are associated with a specified "MTP" unit (*i.e.* "MTP**n**") and are configured using the "SET" command (*i.e.* "**SET MTPn CONFIG**"):

```
SET MTPn CONFIG=<set-option>
```

- The following "**<set-option>**"'s are available, in the form of "**<option>=<value>**", for example:

```
SET MTPn CONFIG=<option>=<value>
```

IPC Configuration

NAME

"**NAME**" configures the device name of the specified "IPC" unit.

```
NAME=<name>
```

Example

- Set the device name of **IPC0** to "IPC0" (*the default name*).

```
SET IPC0 NAME=IPC0
```

NUNITS

"**NUNITS**" configures the number of "IPC" units.

```
    NUNITS=<n>
```

Example

- Set the number of "IPC" units to "2" (*the default number*):

```
    SET IPC NUNITS=6
```

MSP Configuration

NAME

"**NAME**" configures the device name of the specified "MSP" unit.

```
    NAME=<name>
```

Example

- Set the device name of **MSP0** to "MSP0" (*the default name*).

```
    SET MSP0 NAME=MSP0
```

NUNITS

"**NUNITS**" configures the number of "MSP" units.

```
    NUNITS=<n>
```

Example

- Set the number of "MSP" units to "2" (*the default number*):

```
    SET MSP NUNITS=2
```

Disk Configuration

DEBUG (NODEBUG)

- TBD

TYPE

- TBD

READY

- TBD

NAME

"**NAME**" configures the device name of the specified "DISK" unit.

```
    NAME=<name>
```

Example

- Set the device name of **DISK0** to "disk_00" (*the default name*).

```
    SET DISK0 NAME=disk_00
```

NUNITS

"**NUNITS**" configures the number of "DISK" units.

```
    NUNITS=<n>
```

Example

- Set the number of "DISK" units to "26" (*the default number*):

```
    SET DISK NUNITS=26
```

RDR Configuration

DEBUG (NODEBUG)

- TBD

PATH

- TBD

NAME

"**NAME**" configures the device name of the specified "RDR" unit.

```
    NAME=<name>
```

Example

- Set the device name of **RDR0** to "rdra" (*the default name*).

```
    SET RDR0 NAME=rdra
```

NUNITS

"**NUNITS**" configures the number of "RDR" units.

```
    NUNITS=<n>
```

Example

- Set the number of "RDR" units to "3" (*the default number*):

```
    SET RDR NUNITS=3
```

PUN Configuration

DEBUG (NODEBUG)

- TBD

PATH

- TBD

NAME

"**NAME**" configures the device name of the specified "PUN" unit.

```
    NAME=<name>
```

Example

- Set the device name of **PUN0** to "puna" (*the default name*).

```
    SET PUN0 NAME=puna
```

NUNITS

"**NUNITS**" configures the number of "PUN" units.

```
NUNITS=<n>
```

Example

- Set the number of "PUN" units to "3" (*the default number*):

  ```
  SET PUN NUNITS=2
  ```

CONFIG

The following "**PUN**" configuration options are associated with a specified "PUN" unit (*i.e.* "PUN**n**") and are configured using the "SET" command (*i.e.* "**SET PUNn CONFIG**"):

```
SET PUNn CONFIG=<set-option>
```

- The following "**<set-option>**"'s are available, in the form of "**<option>=<value>**", for example:

  ```
  SET PUNn CONFIG=<option>=<value>
  ```

LOGCARDS

* TBD

PRT Configuration

DEBUG (NODEBUG)

- TBD

PATH

- TBD

MODEL

- TBD

READY

- TBD

NAME

"**NAME**" configures the device name of the specified "PRT" unit.

```
    NAME=<name>
```

Example

- Set the device name of **PRT0** to "prta" (*the default name*).

```
    SET PRT0 NAME=prta
```

NUNITS

"**NUNITS**" configures the number of "PRT" units.

```
    NUNITS=<n>
```

Example

- Set the number of "PRT" units to "4" (*the default number*):

```
    SET PRT NUNITS=4
```

CONFIG

The following "**PRT**" configuration options are associated with a specified "PRT" unit (*i.e.* "PRT**n**") and are configured using the "SET" command (*i.e.* "**SET PRTn CONFIG**"):

```
    SET PRTn CONFIG=<set-option>
```

- The following "**<set-option>**"'s are available, in the form of "**<option>=<value>**", for example:

```
    SET PRTn CONFIG=<option>=<value>
```

SPLIT

* TBD

FNP Configuration

DEBUG (NODEBUG)

- TBD

NUNITS

"**NUNITS**" configures the number of "FNP" units.

```
    NUNITS=<n>
```

Example

- Set the number of "FNP" units to "8" (*the default number*):

  ```
      SET FNP NUNITS=8
  ```

CONFIG

The following "**FNP**" configuration options are associated with a specified "FNP" unit (*i.e.* "FNP**n**") and are configured using the "SET" command (*i.e.* "**SET FNPn CONFIG**"):

```
    SET FNPn CONFIG=<set-option>
```

- The following "**<set-option>**"'s are available, in the form of "**<option>=<value>**", for example:

  ```
      SET FNPn CONFIG=<option>=<value>
  ```

MAILBOX

* TBD

IPC_NAME

* TBD

SERVICE

* TBD

OPC Configuration

DEBUG (NODEBUG)

- TBD

AUTOINPUT

- TBD

NAME

"**NAME**" configures the device name of the specified "OPC" unit.

```
    NAME=<name>
```

Example

- Set the device name of **OPC0** to "OPC0" (*the default name*).

```
    SET OPC0 NAME=OPC0
```

NUNITS

"**NUNITS**" configures the number of "OPC" units.

```
    NUNITS=<n>
```

Example

- Set the number of "OPC" units to "2" (*the default number*):

```
    SET OPC NUNITS=2
```

CONFIG

The following "**OPC**" configuration options are associated with a specified "OPC" unit (*i.e.* "OPC**n**") and are configured using the "SET" command (*i.e.* "**SET OPCn CONFIG**"):

```
    SET OPCn CONFIG=<set-option>
```

- The following "**<set-option>**"'s are available, in the form of "**<option>=<value>**", for example:

```
    SET OPCn CONFIG=<option>=<value>
```

AUTOACCEPT

* TBD

NOEMPTY

* TBD

ATTN_FLUSH

* TBD

PORT

* TBD

ADDRESS

* TBD

PW

* TBD

URP Configuration

DEBUG (NODEBUG)

- TBD

NAME

"**NAME**" configures the device name of the specified "URP" unit.

```
NAME=<name>
```

Example

- Set the device name of **URP0** to "urpa" (*the default name*).

```
SET URP0 NAME=urpa
```

NUNITS

"**NUNITS**" configures the number of "URP" units.

```
NUNITS=<n>
```

Example

- Set the number of "URP" units to "10" (*the default number*):

```
SET URP NUNITS=10
```

SHIFT

Shift the command files positional parameters

SHOW

```
SH{OW} B{UILDINFO}               Show build-time compilation information
SH{OW} CL{OCKS}                  Show wall clock and timer information
SH{OW} C{ONFIGURATION}           Show simulator configuration
SH{OW} D{EFAULT_BASE_SYSTEM}     Show default base system script
SH{OW} DEV{ICES}                 Show devices
SH{OW} M{ODIFIERS}               Show SET commands for all devices
SH{OW} O{N}                      Show ON condition actions
SH{OW} P{ROM}                    Show CPU ID PROM initialization data
SH{OW} Q{UEUE}                   Show event queue
SH{OW} S{HOW}                    Show SHOW commands for all devices
SH{OW} T{IME}                    Show simulated timer
SH{OW} VE{RSION}                 Show simulator version
H{ELP} <dev> SHOW                Show device-specific SHOW commands
SH{OW} <dev> {arg,...}           Show device parameters
SH{OW} <dev> DEBUG               Show device debug flags
SH{OW} <dev> MODIFIERS           Show device modifiers
SH{OW} <dev> RADIX               Show device display radix
SH{OW} <dev> SHOW                Show device SHOW commands
SH{OW} <unit> {arg,...}          Show unit parameters
```

BUILDINFO (B)

"**BUILDINFO**" (*abbreviated* "B") shows build-time compilation information including the complete compiler flags (*i.e.* "CFLAGS", "LDFLAGS", etc.) used when building the simulator, the libraries the simulator is linked against and their versions, relevant definitions set by the C preprocessor at build-time, the types of file locking mechanisms available, and the style of atomic operation primitives in use.

Example

```
sim> SHOW BUILDINFO
 Build Information:
      Compilation info: cc -march=native -Wall -O3 -g3 -fno-strict-aliasing
      -DUSE_FLOCK=1 -DUSE_FCNTL=1 -std=c99 -U__STRICT_ANSI__ -D_GNU_SOURCE
      -flto=auto -pthread -DLOCKLESS -DVER_CURRENT_TIME=2022-11-23 12:39
      UTC -DBUILD_PROM_OSV_TEXT=Linux -DBUILD_PROM_OSA_TEXT=x86_64 -lpthread
      -luv -lrt -ldl

  Relevant definitions: -DCLOCK_MONOTONIC=1 -DCLOCK_MONOTONIC_COARSE=6
  -DCLOCK_MONOTONIC_RAW=4 -DHOST_NAME_MAX=64 -DPATH_MAX=4096
  -DUINT_MAX=(INT_MAX * 2U + 1U) -D_POSIX_C_SOURCE=200809L
  -D_POSIX_HOST_NAME_MAX=255 -D_POSIX_PATH_MAX=256 -D__GNUC__=12
  -D__GNU_LIBRARY__=6 -D__SIZEOF_LONG__=8 -D__STDC__=1 -D__amd64__=1
  -D__linux=1 -D__linux__=1 -D__x86_64__=1

    Event loop library: Built with libuv 1.44.2 (release); 1.44.2 in use
          Math library: decNumber 3.68-20210520p3
     Atomic operations: GNU-style
          File locking: POSIX-style fcntl() and BSD-style flock() locking
```

CLOCKS (CL)

"**CLOCKS**" (*abbreviated* "CL") shows the current wall clock time and internal timer details.

Example

```
sim> SHOW CLOCKS
DPS8/M clock device is Internal Calibrated Timer()
Uncalibrated Timer 8:
  Seconds Running:           0 ()
  Current Insts Per Tick:    50000
  Initializations:           1
  Total Ticks:               0
  Wall Clock Time Now:       12:39:50.282
```

CONFIGURATION (C)

"**CONFIGURATION**" (*abbreviated* "C") shows a detailed overview of the current simulator configuration.

- See the **Default Base System Configuration** section of the **Simulator Defaults** chapter for example output of the "SHOW CONFIGURATION" command.

DEFAULT_BASE_SYSTEM (D)

"**DEFAULT_BASE_SYSTEM**" (*abbreviated* "D") shows the *default base system configuration script* which is executed automatically at simulator startup or via the "**DEFAULT_BASE_SYSTEM**" command.

- See the documentation for the "**DEFAULT_BASE_SYSTEM**" command earlier in this chapter and the **Default Base System Script** section of the **Simulator Defaults** chapter for additional details.

DEVICES (DEV)

"**DEVICES**" (*abbreviated* "DEV") shows devices by name and number of units (*i.e.* "**NUNITS**") configured.

Example

```
sim> SHOW DEVICES
CPU      6 units
IOM      2 units
TAPE    17 units
MTP      1 unit
FNP      8 units
DISK    26 units
IPC      2 units
MSP      2 units
SCU      4 units
OPC      2 units
URP     10 units
RDR      3 units
PUN      3 units
PRT      4 units
```

MODIFIERS (M)

"**MODIFIERS**" (*abbreviated* "M") shows a summary of available "SET" commands for all devices.

ON (O)

"**ON**" (*abbreviated* "O") shows information about "ON" condition actions.

PROM (P)

"**PROM**" (*abbreviated* "P") shows the CPU ID PROM initialization data.

Example

```
sim> SHOW PROM
  CPU Model:            'DPS 8/SIM M'
  CPU Serial:           '0          '
  Ship Date:            '221123'
```

QUEUE (Q)

"**QUEUE**" (*abbreviated* "Q") shows information about the simulator's event queue.

SHOW (S)

"**SHOW**" (*abbreviated* "S") shows a summary of available "SHOW" commands for all devices.

TIME (T)

"**TIME**" (*abbreviated* "T") shows information about the simulated timer.

VERSION (V)

"**VERSION**" (*abbreviated* "V") shows the version of the simulator.

- If available, additional related information will be shown, which may include:
  - a *git commit hash* to uniquely identify the simulator build,
  - the date and time the simulator was last modified,
  - the date and time the *source kit distribution* the simulator was built from was prepared,
  - the date and time the simulator was compiled,
  - a statement regarding support availability for the build (or lack thereof),
  - the name and architecture of the operating system that was used to build the simulator,
  - the name, version, and vendor of the compiler used to build the simulator,
  - information about the person (*or automated process*) that performed the build,
  - the name, version, and architecture of the host operating system executing the simulator,
  - and information regarding licensing terms and conditions.

Example

```
sim> SHOW VERSION
 DPS8/M Simulator:
   Version: R3.0.0 (64-bit)
    Commit: 21f490c3b5a07a0f6ee8ac052c9363da7f0e4b85
  Released: 2022-11-23 08:06 UTC - Kit Prepared: 2022-11-23 12:39 UTC
  Compiled: 2022-11-23 12:39 UTC

 Build Information:
  Build OS: Linux x86_64
```

```
 Compiler: GCC 12.2.1 20220819 (Red Hat 12.2.1-2) x86_64
 Built by: root@runner-punwssdb-project-10664906-concurrent-0

Host System Information:
  Host OS: Linux 6.0.9-300.fc37.x86_64 x86_64

This software is made available under the terms of the ICU License,
version 1.8.1 or later.  For complete details, see the "LICENSE.md"
included or https://gitlab.com/dps8m/dps8m/-/blob/master/LICENSE.md
```

CPU

The following device-specific "SHOW" commands are available for the "CPU" device. If the "SHOW CPU" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* CPU**0**) is assumed.

Example

```
    SHOW CPU COMMAND
    SHOW CPUn COMMAND
```

CONFIG

"**CONFIG**" shows configuration details (set via "SET CPU CONFIG") for the "CPU" unit specified.

Example

```
sim> SHOW CPU0 CONFIG
CPU unit number 0
Fault base:                 002(8)
CPU number:                 0(8)
Data switches:              002684591744(8)
Address switches:           000000(8)
PortA enable:               1(8)
PortA init enable:          1(8)
PortA assignment:           0(8)
PortA interlace:            0(8)
PortA store size:           7(8)
PortB enable:               1(8)
PortB init enable:          1(8)
PortB assignment:           1(8)
PortB interlace:            0(8)
PortB store size:           7(8)
PortC enable:               1(8)
PortC init enable:          1(8)
PortC assignment:           2(8)
PortC interlace:            0(8)
PortC store size:           7(8)
PortD enable:               1(8)
PortD init enable:          1(8)
PortD assignment:           3(8)
```

```
PortD interlace:              0(8)
PortD store size:             7(8)
Processor mode:               Multics [0]
8K Cache:                     Enabled
SDWAM:                        Enabled
PTWAM:                        Enabled
Processor speed:              00(8)
DIS enable:                   1(8)
Steady clock:                 0(8)
Halt on unimplemented:        0(8)
Enable simulated SDWAM/PTWAM: 0(8)
Report faults:                0(8)
TRO faults enabled:           1(8)
Y2K enabled:                  0(8)
drl fatal enabled:            0(8)
useMap:                       0
PROM installed:               1(8)
Hex mode installed:           0(8)
8K cache installed:           1(8)
Clock slave installed:        1(8)
ISOLTS mode:                  0(8)
NODIS mode:                   0(8)
6180 mode:                    0(8) [DPS8/M]
```

NUNITS

"**NUNITS**" shows the number of "CPU" units configured (set via "SET CPU NUNITS").

Example

```
sim> SHOW CPU NUNITS
Number of CPUs in system is 6
```

KIPS

"**KIPS**" shows the global CPU lockup fault scaling factor (set via "SET CPU KIPS").

Example

```
sim> SHOW CPU KIPS
CPU KIPS 1000
```

STALL

"**STALL**" shows the currently configured stall points (set via "SET CPU STALL").

Example

```
sim> SET CPU STALL=0=132:3737=12500
sim> SHOW CPU STALL
Stall points
 0 00132:003737  12500
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET CPU DEBUG").

Example

```
sim> SHOW CPU DEBUG
Debugging disabled
```

DISK

The following device-specific "SHOW" commands are available for the "DISK" device. If the "SHOW DISK" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* DISK**0**) is assumed.

NAME

"**NAME**" shows the currently configured device name (set via "SET DISKn NAME").

Example

```
sim> SHOW DISK0 NAME
name     : dska_00
```

NUNITS

"**NUNITS**" shows the number of "DISK" units configured (set via "SET DISK NUNITS").

Example

```
sim> SHOW DISK NUNITS
Number of DISK units in system is 26
```

TYPE

"**TYPE**" shows the currently configured disk type (set via "SET DISK TYPE").

Example

```
sim> SHOW DISK TYPE
type     : 3381
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET DISK DEBUG").

Example

```
sim> SHOW DISK DEBUG
Debugging disabled
```

FNP

The following device-specific "SHOW" commands are available for the "FNP" device. If the "SHOW FNP" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* FNP**0**) is assumed.

CONFIG

"**CONFIG**" shows configuration details (set via "SET FNPn CONFIG") for the "FNP" unit specified.

Example

```
sim> SHOW FNP0 CONFIG
FNP unit number 0
FNP mailbox address:         3400(8)
```

IPC_NAME

"**IPC_NAME**" shows the IPC name for the "FNP" unit specified (set via "SET FNPn IPC_NAME").

Example

```
sim> SHOW FNP0 IPC_NAME
 FNP IPC name: fnp-a
```

NAME

"**NAME**" shows the currently configured device name (set via "SET FNPn NAME").

Example

```
sim> SHOW FNP0 NAME
         name: default
```

NUNITS

"**NUNITS**" shows the number of "FNP" units configured (set via "SET FNP NUNITS").

Example

```
sim> SHOW FNP NUNITS
Number of FNP units in system is 8
```

SERVICE

"**SERVICE**" shows the configured service for each line for the "FNP" unit specified.

- **NOTE**: Only the first ten lines of command output have been reproduced here; the remaining output has been excluded for brevity.

Example

```
sim> SHOW FNP0 SERVICE
        a.000: undefined
               a.001: undefined
               a.002: undefined
               a.003: undefined
               a.004: undefined
               a.005: undefined
               a.006: undefined
               a.007: undefined
               a.008: undefined
               a.009: undefined
               a.010: undefined
```

STATUS

"**STATUS**" shows detailed line status information for the "FNP" unit specified.

- **NOTE**: Only the status of the first line has been reproduced here; the remaining output has been excluded for brevity.

Example

```
sim> SHOW FNP0 STATUS
FNP unit number 0:
        mailboxAddress:              3400
        fnpIsRunning:                0
        fnpMBXinUse:                 0 0 0 0
        lineWaiting:                 0 0 0 0
```

```
        fnpMBXlineno:               0 0 0 0
        accept_calls:               0
  line 0:
        service:                    0
        line_client:                (nil)
        was_CR:                     0
        listen:                     0
        inputBufferSize:            0
        line_break:                 0
        send_output:                0
        accept_new_terminal:        0
        line_disconnected:          0
        acu_dial_failure:           0
        accept_input:               0
        waitForMbxDone:             0
        input_reply_pending:        0
        input_break:                0
        nPos:                       0
        inBuffer:                   (nil)
        inSize:                     0
        inUsed:                     0
        port:                       0
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET FNP DEBUG").

Example

```
sim> SHOW FNP DEBUG
Debugging disabled
```

IOM

The following device-specific "SHOW" commands are available for the "IOM" device. If the "SHOW IOM" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* IOM**0**) is assumed.

CONFIG

Example

"**CONFIG**" shows configuration details (set via "SET IOMn CONFIG") for the "IOM" unit specified.

```
sim> SHOW IOM0 CONFIG
IOM unit number 0
Allowed Operating System: Multics
IOM Base Address:         014(8)
Multiplex Base Address:   0120(8)
```

```
Bootload Card/Tape:       TAPE
Bootload Tape Channel:    12(8)
Bootload Card Channel:    11(8)
Bootload Port:            00(8)
Port Address:             000 001 002 003 000 000 000 000
Port Interlace:             0   0   0   0   0   0   0   0
Port Enable:                1   1   1   1   0   0   0   0
Port Sysinit Enable:        0   0   0   0   0   0   0   0
Port Halfsize:              0   0   0   0   0   0   0   0
Port Storesize:             7   7   7   7   0   0   0   0
```

NUNITS

"**NUNITS**" shows the number of "IOM" units configured (set via "SET IOM NUNITS").

Example

```
sim> SHOW IOM NUNITS
Number of IOM units in system is 2
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET IOM DEBUG").

```
sim> SHOW IOM DEBUG
Debugging disabled
```

IPC

The following device-specific "SHOW" commands are available for the "IPC" device. If the "SHOW IPC" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* IPC**0**) is assumed.

NAME

"**NAME**" shows the currently configured device name (set via "SET IPCn NAME").

Example

```
sim> SHOW IPC0 NAME
name     : IPC0
```

NUNITS

"**NUNITS**" shows the number of "IPC" units configured (set via "SET IPC NUNITS").

Example

```
sim> SHOW IPC NUNITS
Number of IPC units in system is 2
```

MSP

The following device-specific "SHOW" commands are available for the "MSP" device. If the "SHOW MSP" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* MSP**0**) is assumed.

NAME

"**NAME**" shows the currently configured device name (set via "SET MSPn NAME").

Example

```
sim> SHOW MSP0 NAME
name     : MSP0
```

NUNITS

"**NUNITS**" shows the number of "MSP" units configured (set via "SET MSP NUNITS").

Example

```
sim> SHOW MSP NUNITS
Number of MSP units in system is 2
```

MTP

The following device-specific "SHOW" commands are available for the "MTP" device. If the "SHOW MTP" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* MTP**0**) is assumed.

BOOT_DRIVE

"**BOOT_DRIVE**" shows the currently configured boot drive for the unit specified (set via "SET MTPn BOOT_DRIVE").

Example

```
sim> SHOW MTP0 BOOT_DRIVE
boot     : 0
```

NAME

"**NAME**" shows the currently configured device name (set via "SET MTPn NAME").

Example

```
sim> SHOW MTP0 NAME
name     : MTP0
```

NUNITS

"**NUNITS**" shows the number of "MTP" units configured (set via "SET MTP NUNITS").

Example

```
sim> SHOW MTP NUNITS
Number of MTP controllers in the system is 1
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET MTP DEBUG").

Example

```
sim> SHOW MTP DEBUG
Debugging disabled
```

OPC

The following device-specific "SHOW" commands are available for the "OPC" device. If the "SHOW OPC" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* OPC**0**) is assumed.

CONFIG

"**CONFIG**" shows configuration details (set via "SET OPCn CONFIG") for the "OPC" unit specified.

Example

```
sim> SHOW OPC0 CONFIG
flags    : autoaccept=0, noempty=0, attn_flush=1
```

ADDRESS

"**ADDRESS**" shows the currently configured address for the specified unit (set via "SET OPCn ADDRESS").

Example

```
sim> SHOW OPC0 ADDRESS
address  : any
```

AUTOINPUT

"**AUTOINPUT**" shows the autoinput buffer of the specified OPC unit.

Example

```
sim> SHOW OPC0 AUTOINPUT
autoinput: empty
```

NAME

"**NAME**" shows the currently configured device name (set via "SET OPCn NAME").

Example

```
sim> SHOW OPC0 NAME
name     : OPC0
```

NUNITS

"**NUNITS**" shows the number of "OPC" units configured (set via "SET OPC NUNITS").

Example

```
sim> SHOW OPC NUNITS
2 units
```

PORT

"**PORT**" shows the currently configured port for the specified unit (set via "SET OPCn PORT").

Example

```
sim> SHOW OPC0 PORT
port     : disabled
```

PW

"**PW**" shows the currently configured password for the specified unit (set via "SET OPCn PW").

Example

```
sim> SHOW OPC0 PW
password : MulticsRulez
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET OPC DEBUG").

Example

```
sim> SHOW OPC DEBUG
Debugging disabled
```

PRT

The following device-specific "SHOW" commands are available for the "PRT" device. If the "SHOW PRT" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* PRT**0**) is assumed.

CONFIG

"**CONFIG**" shows configuration details (set via "SET PRTn CONFIG") for the "PRT" unit specified.

Example

```
sim> SHOW PRT0 CONFIG
split    : 0
```

MODEL

"**MODEL**" shows configured model (set via "SET PRTn MODEL") for the "PRT" unit specified.

Example

```
sim> SHOW PRT0 MODEL
model    : 1600
```

NAME

"**NAME**" shows the currently configured device name (set via "SET PRTn NAME").

Example

```
sim> SHOW PRT0 NAME
name     : prta
```

NUNITS

"**NUNITS**" shows the number of "PRT" units configured (set via "SET PRT NUNITS").

Example

```
sim> SHOW PRT NUNITS
Number of PRT units in system is 4
```

PATH

"**PATH**" shows the currently configured output path (set via "SET PRT PATH").

- An empty path is equivalent to the host operating system current working directory.

Example

```
sim> SHOW PRT PATH
Path to PRT files is
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET PRT DEBUG").

Example

```
sim> SHOW PRT DEBUG
Debugging disabled
```

PUN

The following device-specific "SHOW" commands are available for the "PUN" device. If the "SHOW PUN" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* PUN**0**) is assumed.

CONFIG

"**CONFIG**" shows configuration details (set via "SET PUNn CONFIG") for the "PUN" unit specified.

Example

```
sim> SHOW PUN0 CONFIG
logcards : 0
```

NAME

"**NAME**" shows the currently configured device name (set via "SET PUNn NAME").

Example

```
sim> SHOW PUN0 NAME
name     : puna
```

NUNITS

"**NUNITS**" shows the number of "PUN" units configured (set via "SET PUN NUNITS").

Example

```
sim> SHOW PUN NUNITS
Number of PUN units in system is 3
```

PATH

"**PATH**" shows the currently configured output path. (set via "SET PUN PATH").

- An empty path is equivalent to the host operating system current working directory.

Example

```
sim> SHOW PUN PATH
Path to card punch directories is
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET PUN DEBUG").

Example

```
sim> SHOW PUN DEBUG
Debugging disabled
```

RDR

The following device-specific "SHOW" commands are available for the "RDR" device. If the "SHOW RDR" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* RDR**0**) is assumed.

NAME

"**NAME**" shows the currently configured device name (set via "SET RDRn NAME").

Example

```
sim> SHOW RDR0 NAME
name     : rdra
```

NUNITS

"**NUNITS**" shows the number of "RDR" units configured (set via "SET RDR NUNITS").

Example

```
sim> SHOW RDR NUNITS
Number of RDR units in system is 3
```

PATH

"**PATH**" shows the currently configured card reader directory path. (set via "SET RDR PATH").

- An empty path is equivalent to the host operating system current working directory.

Example

```
sim> SHOW RDR PATH
Path to card reader directories is
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET RDR DEBUG").

Example

```
sim> SHOW RDR DEBUG
Debugging disabled
```

SCU

The following device-specific "SHOW" commands are available for the "SCU" device. If the "SHOW SCU" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* SCU**0**) is assumed.

CONFIG

"**CONFIG**" shows configuration details (set via "SET SCUn CONFIG") for the "SCU" unit specified.

Example

```
sim> SHOW SCU0 CONFIG
SCU unit number 0
Mode:                       Program
Port Enable:                1   1   1   1   1   1   1   1
Mask A:                     7
Mask B:                     Off
Lower Store Size:           7
Cyclic:                     040
Non-existent address:       200
```

NUNITS

"**NUNITS**" shows the number of "SCU" units configured (set via "SET SCU NUNITS").

Example

```
sim> SHOW SCU NUNITS
Number of SCU units in system is 4
```

STATE

"**STATE**" shows detailed information about the state of the "SCU" unit specified.

Example

```
sim> SHOW SCU0 STATE
SCU unit number 0
    Mode PROGRAM
    Port 0 ENABLE  dev_idx 0 dev_port 0 type IOM
    Port 1 ENABLE  dev_idx 1 dev_port 0 type IOM
    Port 2 ENABLE  dev_idx 5 dev_port 0 type CPU
    Port 3 ENABLE  dev_idx 4 dev_port 0 type CPU
    Port 4 ENABLE  dev_idx 3 dev_port 0 type CPU
```

```
    Port 5 ENABLE  dev_idx 2 dev_port 0 type CPU
    Port 6 ENABLE  dev_idx 1 dev_port 0 type CPU
    Port 7 ENABLE  dev_idx 0 dev_port 0 type CPU
    Cell A
        exec_intr_mask 037777777777
        mask_enable ENABLE
        mask_assignment 7
        cells 00000000000000000000000000000000
    Cell B
        exec_intr_mask 037777777777
        mask_enable DISABLE
        mask_assignment 0
        cells 00000000000000000000000000000000
Lower store size: 7
Cyclic: 040
NEA: 200
Online: 14
Interlace: 0
Lower: 0
ID: 2
mode_reg: 000000
Elapsed days: 0
Steady clock: 0
Bullet time: 0
Y2K enabled: 0
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET SCU DEBUG").

Example

```
sim> SHOW SCU DEBUG
Debugging disabled
```

TAPE

The following device-specific "SHOW" commands are available for the "TAPE" device. If the "SHOW TAPE" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* TAPE**0**) is assumed.

ADD_PATH

"**ADD_PATH**" shows the currently configured tape directory search paths (set via "SET TAPE ADD_PATH").

Example

```
sim> SHOW TAPE ADD_PATH
Tape directory search paths:
Prefix                          Directory
```

```
------                              ---------
[default]
```

CAPACITY

"**CAPACITY**" shows the storage capacity of the specified "TAPE" unit.

Example

```
sim> SHOW TAPE0 CAPACITY
capacity : 40MB
```

DEFAULT_PATH

"**DEFAULT_PATH**" shows the currently configured default tape path (set via "SET TAPE DEFAULT_PATH").

- An empty path is equivalent to the host operating system current working directory.

Example

```
sim> SHOW TAPE DEFAULT_PATH
TAPE DEFAULT_PATH:
```

NAME

"**NAME**" shows the currently configured device name (set via "SET TAPEn NAME").

Example

```
sim> SHOW TAPE0 NAME
name     : default
```

NUNITS

"**NUNITS**" shows the number of "TAPE" units configured (set via "SET TAPE NUNITS").

Example

```
sim> SHOW TAPE NUNITS
Number of TAPE units in system is 17
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET TAPE DEBUG").

Example

```
sim> SHOW TAPE DEBUG
Debugging disabled
```

URP

The following device-specific "SHOW" commands are available for the "URP" device. If the "SHOW URP" command can operate on a specific unit, but no unit is specified, the first unit (*i.e.* URP**0**) is assumed.

NAME

"**NAME**" shows the currently configured device name (set via "SET URPn NAME").

Example

```
sim> SHOW URP0 NAME
name     : urpa
```

NUNITS

"**NUNITS**" shows the number of "URP" units configured (set via "SET URP NUNITS").

Example

```
sim> SHOW URP NUNITS
Number of URP units in system is 10
```

DEBUG

"**DEBUG**" shows the currently configured debug options (set via "SET URP DEBUG").

Example

```
sim> SHOW URP DEBUG
Debugging disabled
```

SKIPBOOT

Skip forward on boot tape

SLOWPOLL

Set slow polling interval (in polling intervals).

STEP (S)

The STEP command (*abbreviated* S) resumes execution at the current PC for the number of instructions given by its argument. If no argument is supplied, one instruction is executed.

Switches

-T If the STEP command is invoked with the "-T" switch, the step command will cause execution to run for *microseconds* rather than instructions.

Simulator Defaults

Default Base System Script

The following script defines the *default base system* configuration, and is executed automatically at simulator startup. It may also be explicitly re-executed via the "**DEFAULT_BASE_SYSTEM**" command.

- See the "**DEFAULT_BASE_SYSTEM**" command documentation for more information.

```
;; DPS8/M simulator R3.0.0
; DEFAULT_BASE_SYSTEM_SCRIPT (660 lines follow)
CABLE_RIPOUT
SET CPU NUNITS=6
SET IOM NUNITS=2
SET TAPE NUNITS=17
SET MTP NUNITS=1
SET IPC NUNITS=2
SET MSP NUNITS=2
SET DISK NUNITS=26
SET SCU NUNITS=4
SET OPC NUNITS=2
SET FNP NUNITS=8
SET URP NUNITS=10
SET RDR NUNITS=3
SET PUN NUNITS=3
SET PRT NUNITS=4
SET CPU0 CONFIG=FAULTBASE=Multics
SET CPU0 CONFIG=NUM=0
SET CPU0 CONFIG=DATA=024000717200
SET CPU0 CONFIG=ADDRESS=000000000000
SET CPU0 CONFIG=PORT=A
SET CPU0   CONFIG=ASSIGNMENT=0
SET CPU0   CONFIG=INTERLACE=0
SET CPU0   CONFIG=ENABLE=1
SET CPU0   CONFIG=INIT_ENABLE=1
SET CPU0   CONFIG=STORE_SIZE=4M
SET CPU0 CONFIG=PORT=B
SET CPU0   CONFIG=ASSIGNMENT=1
SET CPU0   CONFIG=INTERLACE=0
SET CPU0   CONFIG=ENABLE=1
SET CPU0   CONFIG=INIT_ENABLE=1
SET CPU0   CONFIG=STORE_SIZE=4M
SET CPU0 CONFIG=PORT=C
SET CPU0   CONFIG=ASSIGNMENT=2
SET CPU0   CONFIG=INTERLACE=0
SET CPU0   CONFIG=ENABLE=1
SET CPU0   CONFIG=INIT_ENABLE=1
SET CPU0   CONFIG=STORE_SIZE=4M
```

```
SET CPU0 CONFIG=PORT=D
SET CPU0   CONFIG=ASSIGNMENT=3
SET CPU0   CONFIG=INTERLACE=0
SET CPU0   CONFIG=ENABLE=1
SET CPU0   CONFIG=INIT_ENABLE=1
SET CPU0   CONFIG=STORE_SIZE=4M
SET CPU0 CONFIG=MODE=Multics
SET CPU0 CONFIG=ENABLE_CACHE=enable
SET CPU0 CONFIG=SDWAM=enable
SET CPU0 CONFIG=PTWAM=enable
SET CPU0 CONFIG=SPEED=0
SET CPU0 CONFIG=DIS_ENABLE=enable
SET CPU0 CONFIG=STEADY_CLOCK=disable
SET CPU0 CONFIG=HALT_ON_UNIMPLEMENTED=disable
SET CPU0 CONFIG=ENABLE_WAM=disable
SET CPU0 CONFIG=REPORT_FAULTS=disable
SET CPU0 CONFIG=TRO_ENABLE=enable
SET CPU0 CONFIG=Y2K=disable
SET CPU0 CONFIG=DRL_FATAL=disable
SET CPU0 CONFIG=USEMAP=disable
SET CPU0 CONFIG=PROM_INSTALLED=enable
SET CPU0 CONFIG=HEX_MODE_INSTALLED=disable
SET CPU0 CONFIG=CACHE_INSTALLED=enable
SET CPU0 CONFIG=CLOCK_SLAVE_INSTALLED=enable
SET CPU1 CONFIG=FAULTBASE=Multics
SET CPU1 CONFIG=NUM=1
SET CPU1 CONFIG=DATA=024000717200
SET CPU1 CONFIG=ADDRESS=000000000000
SET CPU1 CONFIG=PORT=A
SET CPU1   CONFIG=ASSIGNMENT=0
SET CPU1   CONFIG=INTERLACE=0
SET CPU1   CONFIG=ENABLE=1
SET CPU1   CONFIG=INIT_ENABLE=1
SET CPU1   CONFIG=STORE_SIZE=4M
SET CPU1 CONFIG=PORT=B
SET CPU1   CONFIG=ASSIGNMENT=1
SET CPU1   CONFIG=INTERLACE=0
SET CPU1   CONFIG=ENABLE=1
SET CPU1   CONFIG=INIT_ENABLE=1
SET CPU1   CONFIG=STORE_SIZE=4M
SET CPU1 CONFIG=PORT=C
SET CPU1   CONFIG=ASSIGNMENT=2
SET CPU1   CONFIG=INTERLACE=0
SET CPU1   CONFIG=ENABLE=1
SET CPU1   CONFIG=INIT_ENABLE=1
SET CPU1   CONFIG=STORE_SIZE=4M
SET CPU1 CONFIG=PORT=D
SET CPU1   CONFIG=ASSIGNMENT=3
SET CPU1   CONFIG=INTERLACE=0
SET CPU1   CONFIG=ENABLE=1
SET CPU1   CONFIG=INIT_ENABLE=1
SET CPU1   CONFIG=STORE_SIZE=4M
SET CPU1 CONFIG=MODE=Multics
SET CPU1 CONFIG=ENABLE_CACHE=enable
SET CPU1 CONFIG=SDWAM=enable
```

```
SET CPU1 CONFIG=PTWAM=enable
SET CPU1 CONFIG=SPEED=0
SET CPU1 CONFIG=DIS_ENABLE=enable
SET CPU1 CONFIG=STEADY_CLOCK=disable
SET CPU1 CONFIG=HALT_ON_UNIMPLEMENTED=disable
SET CPU1 CONFIG=ENABLE_WAM=disable
SET CPU1 CONFIG=REPORT_FAULTS=disable
SET CPU1 CONFIG=TRO_ENABLE=enable
SET CPU1 CONFIG=Y2K=disable
SET CPU1 CONFIG=DRL_FATAL=disable
SET CPU1 CONFIG=USEMAP=disable
SET CPU1 CONFIG=PROM_INSTALLED=enable
SET CPU1 CONFIG=HEX_MODE_INSTALLED=disable
SET CPU1 CONFIG=CACHE_INSTALLED=enable
SET CPU1 CONFIG=CLOCK_SLAVE_INSTALLED=enable
SET CPU2 CONFIG=FAULTBASE=Multics
SET CPU2 CONFIG=NUM=2
SET CPU2 CONFIG=DATA=024000717200
SET CPU2 CONFIG=ADDRESS=000000000000
SET CPU2 CONFIG=PORT=A
SET CPU2   CONFIG=ASSIGNMENT=0
SET CPU2   CONFIG=INTERLACE=0
SET CPU2   CONFIG=ENABLE=1
SET CPU2   CONFIG=INIT_ENABLE=1
SET CPU2   CONFIG=STORE_SIZE=4M
SET CPU2 CONFIG=PORT=B
SET CPU2   CONFIG=ASSIGNMENT=1
SET CPU2   CONFIG=INTERLACE=0
SET CPU2   CONFIG=ENABLE=1
SET CPU2   CONFIG=INIT_ENABLE=1
SET CPU2   CONFIG=STORE_SIZE=4M
SET CPU2 CONFIG=PORT=C
SET CPU2   CONFIG=ASSIGNMENT=2
SET CPU2   CONFIG=INTERLACE=0
SET CPU2   CONFIG=ENABLE=1
SET CPU2   CONFIG=INIT_ENABLE=1
SET CPU2   CONFIG=STORE_SIZE=4M
SET CPU2 CONFIG=PORT=D
SET CPU2   CONFIG=ASSIGNMENT=3
SET CPU2   CONFIG=INTERLACE=0
SET CPU2   CONFIG=ENABLE=1
SET CPU2   CONFIG=INIT_ENABLE=1
SET CPU2   CONFIG=STORE_SIZE=4M
SET CPU2 CONFIG=MODE=Multics
SET CPU2 CONFIG=ENABLE_CACHE=enable
SET CPU2 CONFIG=SDWAM=enable
SET CPU2 CONFIG=PTWAM=enable
SET CPU2 CONFIG=SPEED=0
SET CPU2 CONFIG=DIS_ENABLE=enable
SET CPU2 CONFIG=STEADY_CLOCK=disable
SET CPU2 CONFIG=HALT_ON_UNIMPLEMENTED=disable
SET CPU2 CONFIG=ENABLE_WAM=disable
SET CPU2 CONFIG=REPORT_FAULTS=disable
SET CPU2 CONFIG=TRO_ENABLE=enable
SET CPU2 CONFIG=Y2K=disable
```

```
SET CPU2 CONFIG=DRL_FATAL=disable
SET CPU2 CONFIG=USEMAP=disAble
SET CPU2 CONFIG=PROM_INSTALLED=enable
SET CPU2 CONFIG=HEX_MODE_INSTALLED=disable
SET CPU2 CONFIG=CACHE_INSTALLED=enable
SET CPU2 CONFIG=CLOCK_SLAVE_INSTALLED=enable
SET CPU3 CONFIG=FAULTBASE=Multics
SET CPU3 CONFIG=NUM=3
SET CPU3 CONFIG=DATA=024000717200
SET CPU3 CONFIG=ADDRESS=000000000000
SET CPU3 CONFIG=PORT=A
SET CPU3   CONFIG=ASSIGNMENT=0
SET CPU3   CONFIG=INTERLACE=0
SET CPU3   CONFIG=ENABLE=1
SET CPU3   CONFIG=INIT_ENABLE=1
SET CPU3   CONFIG=STORE_SIZE=4M
SET CPU3 CONFIG=PORT=B
SET CPU3   CONFIG=ASSIGNMENT=1
SET CPU3   CONFIG=INTERLACE=0
SET CPU3   CONFIG=ENABLE=1
SET CPU3   CONFIG=INIT_ENABLE=1
SET CPU3   CONFIG=STORE_SIZE=4M
SET CPU3 CONFIG=PORT=C
SET CPU3   CONFIG=ASSIGNMENT=2
SET CPU3   CONFIG=INTERLACE=0
SET CPU3   CONFIG=ENABLE=1
SET CPU3   CONFIG=INIT_ENABLE=1
SET CPU3   CONFIG=STORE_SIZE=4M
SET CPU3 CONFIG=PORT=D
SET CPU3   CONFIG=ASSIGNMENT=3
SET CPU3   CONFIG=INTERLACE=0
SET CPU3   CONFIG=ENABLE=1
SET CPU3   CONFIG=INIT_ENABLE=1
SET CPU3   CONFIG=STORE_SIZE=4M
SET CPU3 CONFIG=MODE=Multics
SET CPU3 CONFIG=ENABLE_CACHE=enable
SET CPU3 CONFIG=SDWAM=enable
SET CPU3 CONFIG=PTWAM=enable
SET CPU3 CONFIG=SPEED=0
SET CPU3 CONFIG=DIS_ENABLE=enable
SET CPU3 CONFIG=STEADY_CLOCK=disable
SET CPU3 CONFIG=HALT_ON_UNIMPLEMENTED=disable
SET CPU3 CONFIG=ENABLE_WAM=disable
SET CPU3 CONFIG=REPORT_FAULTS=disable
SET CPU3 CONFIG=TRO_ENABLE=enable
SET CPU3 CONFIG=Y2K=disable
SET CPU3 CONFIG=DRL_FATAL=disable
SET CPU3 CONFIG=USEMAP=disable
SET CPU3 CONFIG=PROM_INSTALLED=enable
SET CPU3 CONFIG=HEX_MODE_INSTALLED=disable
SET CPU3 CONFIG=CACHE_INSTALLED=enable
SET CPU3 CONFIG=CLOCK_SLAVE_INSTALLED=enable
SET CPU4 CONFIG=FAULTBASE=Multics
SET CPU4 CONFIG=NUM=4
SET CPU4 CONFIG=DATA=024000717200
```

```
SET CPU4 CONFIG=ADDRESS=000000000000
SET CPU4 CONFIG=PORT=A
SET CPU4   CONFIG=ASSIGNMENT=0
SET CPU4   CONFIG=INTERLACE=0
SET CPU4   CONFIG=ENABLE=1
SET CPU4   CONFIG=INIT_ENABLE=1
SET CPU4   CONFIG=STORE_SIZE=4M
SET CPU4 CONFIG=PORT=B
SET CPU4   CONFIG=ASSIGNMENT=1
SET CPU4   CONFIG=INTERLACE=0
SET CPU4   CONFIG=ENABLE=1
SET CPU4   CONFIG=INIT_ENABLE=1
SET CPU4   CONFIG=STORE_SIZE=4M
SET CPU4 CONFIG=PORT=C
SET CPU4   CONFIG=ASSIGNMENT=2
SET CPU4   CONFIG=INTERLACE=0
SET CPU4   CONFIG=ENABLE=1
SET CPU4   CONFIG=INIT_ENABLE=1
SET CPU4   CONFIG=STORE_SIZE=4M
SET CPU4 CONFIG=PORT=D
SET CPU4   CONFIG=ASSIGNMENT=3
SET CPU4   CONFIG=INTERLACE=0
SET CPU4   CONFIG=ENABLE=1
SET CPU4   CONFIG=INIT_ENABLE=1
SET CPU4   CONFIG=STORE_SIZE=4M
SET CPU4 CONFIG=MODE=Multics
SET CPU4 CONFIG=ENABLE_CACHE=enable
SET CPU4 CONFIG=SDWAM=enable
SET CPU4 CONFIG=PTWAM=enable
SET CPU4 CONFIG=SPEED=0
SET CPU4 CONFIG=DIS_ENABLE=enable
SET CPU4 CONFIG=STEADY_CLOCK=disable
SET CPU4 CONFIG=HALT_ON_UNIMPLEMENTED=disable
SET CPU4 CONFIG=ENABLE_WAM=disable
SET CPU4 CONFIG=REPORT_FAULTS=disable
SET CPU4 CONFIG=TRO_ENABLE=enable
SET CPU4 CONFIG=Y2K=disable
SET CPU4 CONFIG=DRL_FATAL=disable
SET CPU4 CONFIG=USEMAP=disable
SET CPU4 CONFIG=PROM_INSTALLED=enable
SET CPU4 CONFIG=HEX_MODE_INSTALLED=disable
SET CPU4 CONFIG=CACHE_INSTALLED=enable
SET CPU4 CONFIG=CLOCK_SLAVE_INSTALLED=enable
SET CPU5 CONFIG=FAULTBASE=Multics
SET CPU5 CONFIG=NUM=5
SET CPU5 CONFIG=DATA=024000717200
SET CPU5 CONFIG=ADDRESS=000000000000
SET CPU5 CONFIG=PORT=A
SET CPU5   CONFIG=ASSIGNMENT=0
SET CPU5   CONFIG=INTERLACE=0
SET CPU5   CONFIG=ENABLE=1
SET CPU5   CONFIG=INIT_ENABLE=1
SET CPU5   CONFIG=STORE_SIZE=4M
SET CPU5 CONFIG=PORT=B
SET CPU5   CONFIG=ASSIGNMENT=1
```

```
SET CPU5   CONFIG=INTERLACE=0
SET CPU5   CONFIG=ENABLE=1
SET CPU5   CONFIG=INIT_ENABLE=1
SET CPU5   CONFIG=STORE_SIZE=4M
SET CPU5 CONFIG=PORT=C
SET CPU5   CONFIG=ASSIGNMENT=2
SET CPU5   CONFIG=INTERLACE=0
SET CPU5   CONFIG=ENABLE=1
SET CPU5   CONFIG=INIT_ENABLE=1
SET CPU5   CONFIG=STORE_SIZE=4M
SET CPU5 CONFIG=PORT=D
SET CPU5   CONFIG=ASSIGNMENT=3
SET CPU5   CONFIG=INTERLACE=0
SET CPU5   CONFIG=ENABLE=1
SET CPU5   CONFIG=INIT_ENABLE=1
SET CPU5   CONFIG=STORE_SIZE=4M
SET CPU5 CONFIG=MODE=Multics
SET CPU5 CONFIG=ENABLE_CACHE=enable
SET CPU5 CONFIG=SDWAM=enable
SET CPU5 CONFIG=PTWAM=enable
SET CPU5 CONFIG=SPEED=0
SET CPU5 CONFIG=DIS_ENABLE=enable
SET CPU5 CONFIG=STEADY_CLOCK=disable
SET CPU5 CONFIG=HALT_ON_UNIMPLEMENTED=disable
SET CPU5 CONFIG=ENABLE_WAM=disable
SET CPU5 CONFIG=REPORT_FAULTS=disable
SET CPU5 CONFIG=TRO_ENABLE=enable
SET CPU5 CONFIG=Y2K=disable
SET CPU5 CONFIG=DRL_FATAL=disable
SET CPU5 CONFIG=USEMAP=disable
SET CPU5 CONFIG=PROM_INSTALLED=enable
SET CPU5 CONFIG=HEX_MODE_INSTALLED=disable
SET CPU5 CONFIG=CACHE_INSTALLED=enable
SET CPU5 CONFIG=CLOCK_SLAVE_INSTALLED=enable
SET IOM0 CONFIG=IOM_BASE=Multics
SET IOM0 CONFIG=MULTIPLEX_BASE=0120
SET IOM0 CONFIG=OS=Multics
SET IOM0 CONFIG=BOOT=tape
SET IOM0 CONFIG=TAPECHAN=012
SET IOM0 CONFIG=CARDCHAN=011
SET IOM0 CONFIG=SCUPORT=0
SET IOM0 CONFIG=PORT=0
SET IOM0   CONFIG=ADDR=0
SET IOM0   CONFIG=INTERLACE=0
SET IOM0   CONFIG=ENABLE=1
SET IOM0   CONFIG=INITENABLE=0
SET IOM0   CONFIG=HALFSIZE=0
SET IOM0   CONFIG=STORE_SIZE=4M
SET IOM0 CONFIG=PORT=1
SET IOM0   CONFIG=ADDR=1
SET IOM0   CONFIG=INTERLACE=0
SET IOM0   CONFIG=ENABLE=1
SET IOM0   CONFIG=INITENABLE=0
SET IOM0   CONFIG=HALFSIZE=0
SET IOM0   CONFIG=STORE_SIZE=4M
```

```
SET IOM0 CONFIG=PORT=2
SET IOM0   CONFIG=ADDR=2
SET IOM0   CONFIG=INTERLACE=0
SET IOM0   CONFIG=ENABLE=1
SET IOM0   CONFIG=INITENABLE=0
SET IOM0   CONFIG=HALFSIZE=0
SET IOM0   CONFIG=STORE_SIZE=4M
SET IOM0 CONFIG=PORT=3
SET IOM0   CONFIG=ADDR=3
SET IOM0   CONFIG=INTERLACE=0
SET IOM0   CONFIG=ENABLE=1
SET IOM0   CONFIG=INITENABLE=0
SET IOM0   CONFIG=HALFSIZE=0
SET IOM0   CONFIG=STORE_SIZE=4M
SET IOM0 CONFIG=PORT=4
SET IOM0   CONFIG=ENABLE=0
SET IOM0 CONFIG=PORT=5
SET IOM0   CONFIG=ENABLE=0
SET IOM0 CONFIG=PORT=6
SET IOM0   CONFIG=ENABLE=0
SET IOM0 CONFIG=PORT=7
SET IOM0   CONFIG=ENABLE=0
SET IOM1 CONFIG=IOM_BASE=Multics2
SET IOM1 CONFIG=MULTIPLEX_BASE=0121
SET IOM1 CONFIG=OS=Multics
SET IOM1 CONFIG=BOOT=tape
SET IOM1 CONFIG=TAPECHAN=012
SET IOM1 CONFIG=CARDCHAN=011
SET IOM1 CONFIG=SCUPORT=0
SET IOM1 CONFIG=PORT=0
SET IOM1   CONFIG=ADDR=0
SET IOM1   CONFIG=INTERLACE=0
SET IOM1   CONFIG=ENABLE=1
SET IOM1   CONFIG=INITENABLE=0
SET IOM1   CONFIG=HALFSIZE=0
SET IOM1 CONFIG=PORT=1
SET IOM1   CONFIG=ADDR=1
SET IOM1   CONFIG=INTERLACE=0
SET IOM1   CONFIG=ENABLE=1
SET IOM1   CONFIG=INITENABLE=0
SET IOM1   CONFIG=HALFSIZE=0
SET IOM1 CONFIG=PORT=2
SET IOM1   CONFIG=ENABLE=0
SET IOM1 CONFIG=PORT=3
SET IOM1   CONFIG=ENABLE=0
SET IOM1 CONFIG=PORT=4
SET IOM1   CONFIG=ENABLE=0
SET IOM1 CONFIG=PORT=5
SET IOM1   CONFIG=ENABLE=0
SET IOM1 CONFIG=PORT=6
SET IOM1   CONFIG=ENABLE=0
SET IOM1 CONFIG=PORT=7
SET IOM1   CONFIG=ENABLE=0
SET SCU0 CONFIG=MODE=program
SET SCU0 CONFIG=PORT0=enable
```

```
SET SCU0 CONFIG=PORT1=enable
SET SCU0 CONFIG=PORT2=enable
SET SCU0 CONFIG=PORT3=enable
SET SCU0 CONFIG=PORT4=enable
SET SCU0 CONFIG=PORT5=enable
SET SCU0 CONFIG=PORT6=enable
SET SCU0 CONFIG=PORT7=enable
SET SCU0 CONFIG=MASKA=7
SET SCU0 CONFIG=MASKB=off
SET SCU0 CONFIG=LWRSTORESIZE=7
SET SCU0 CONFIG=CYCLIC=0040
SET SCU0 CONFIG=NEA=0200
SET SCU0 CONFIG=ONL=014
SET SCU0 CONFIG=INT=0
SET SCU0 CONFIG=LWR=0
SET SCU1 CONFIG=MODE=program
SET SCU1 CONFIG=PORT0=enable
SET SCU1 CONFIG=PORT1=enable
SET SCU1 CONFIG=PORT2=enable
SET SCU1 CONFIG=PORT3=enable
SET SCU1 CONFIG=PORT4=enable
SET SCU1 CONFIG=PORT5=enable
SET SCU1 CONFIG=PORT6=enable
SET SCU1 CONFIG=PORT7=enable
SET SCU1 CONFIG=MASKA=off
SET SCU1 CONFIG=MASKB=off
SET SCU1 CONFIG=LWRSTORESIZE=7
SET SCU1 CONFIG=CYCLIC=0040
SET SCU1 CONFIG=NEA=0200
SET SCU1 CONFIG=ONL=014
SET SCU1 CONFIG=INT=0
SET SCU1 CONFIG=LWR=0
SET SCU2 CONFIG=MODE=program
SET SCU2 CONFIG=PORT0=enable
SET SCU2 CONFIG=PORT1=enable
SET SCU2 CONFIG=PORT2=enable
SET SCU2 CONFIG=PORT3=enable
SET SCU2 CONFIG=PORT4=enable
SET SCU2 CONFIG=PORT5=enable
SET SCU2 CONFIG=PORT6=enable
SET SCU2 CONFIG=PORT7=enable
SET SCU2 CONFIG=MASKA=off
SET SCU2 CONFIG=MASKB=off
SET SCU2 CONFIG=LWRSTORESIZE=7
SET SCU2 CONFIG=CYCLIC=0040
SET SCU2 CONFIG=NEA=0200
SET SCU2 CONFIG=ONL=014
SET SCU2 CONFIG=INT=0
SET SCU2 CONFIG=LWR=0
SET SCU3 CONFIG=MODE=program
SET SCU3 CONFIG=PORT0=enable
SET SCU3 CONFIG=PORT1=enable
SET SCU3 CONFIG=PORT2=enable
SET SCU3 CONFIG=PORT3=enable
SET SCU3 CONFIG=PORT4=enable
```

```
SET SCU3 CONFIG=PORT5=enable
SET SCU3 CONFIG=PORT6=enable
SET SCU3 CONFIG=PORT7=enable
SET SCU3 CONFIG=MASKA=off
SET SCU3 CONFIG=MASKB=off
SET SCU3 CONFIG=LWRSTORESIZE=7
SET SCU3 CONFIG=CYCLIC=0040
SET SCU3 CONFIG=NEA=0200
SET SCU3 CONFIG=ONL=014
SET SCU3 CONFIG=INT=0
SET SCU3 CONFIG=LWR=0
SET FNP0 CONFIG=MAILBOX=03400
SET FNP0 IPC_NAME=fnp-a
SET FNP1 CONFIG=MAILBOX=03700
SET FNP1 IPC_NAME=fnp-b
SET FNP2 CONFIG=MAILBOX=04200
SET FNP2 IPC_NAME=fnp-c
SET FNP3 CONFIG=MAILBOX=04500
SET FNP3 IPC_NAME=fnp-d
SET FNP4 CONFIG=MAILBOX=05000
SET FNP4 IPC_NAME=fnp-e
SET FNP5 CONFIG=MAILBOX=05300
SET FNP5 IPC_NAME=fnp-f
SET FNP6 CONFIG=MAILBOX=05600
SET FNP6 IPC_NAME=fnp-g
SET FNP7 CONFIG=MAILBOX=06100
SET FNP7 IPC_NAME=fnp-h
SET MTP0 BOOT_DRIVE=0
SET MTP0 NAME=MTP0
CABLE IOM0 012 MTP0 0
CABLE IOM1 012 MTP0 1
CABLE MTP0 1 TAPE1
SET TAPE1 NAME=tapa_01
CABLE MTP0 2 TAPE2
SET TAPE2 NAME=tapa_02
CABLE MTP0 3 TAPE3
SET TAPE3 NAME=tapa_03
CABLE MTP0 4 TAPE4
SET TAPE4 NAME=tapa_04
CABLE MTP0 5 TAPE5
SET TAPE5 NAME=tapa_05
CABLE MTP0 6 TAPE6
SET TAPE6 NAME=tapa_06
CABLE MTP0 7 TAPE7
SET TAPE7 NAME=tapa_07
CABLE MTP0 8 TAPE8
SET TAPE8 NAME=tapa_08
CABLE MTP0 9 TAPE9
SET TAPE9 NAME=tapa_09
CABLE MTP0 10 TAPE10
SET TAPE10 NAME=tapa_10
CABLE MTP0 11 TAPE11
SET TAPE11 NAME=tapa_11
CABLE MTP0 12 TAPE12
SET TAPE12 NAME=tapa_12
```

```
CABLE MTP0 13 TAPE13
SET TAPE13 NAME=tapa_13
CABLE MTP0 14 TAPE14
SET TAPE14 NAME=tapa_14
CABLE MTP0 15 TAPE15
SET TAPE15 NAME=tapa_15
CABLE MTP0 16 TAPE16
SET TAPE16 NAME=tapa_16
SET IPC0 NAME=IPC0
CABLE IOM0 013 IPC0 0
CABLE IOM1 013 IPC0 1
CABLE IPC0 0 DISK0
SET DISK0 TYPE=3381
SET DISK0 NAME=dska_00
cable IPC0 1 DISK1
set disk1 type=3381
set disk1 name=dska_01
CABLE IPC0 2 DISK2
SET DISK2 TYPE=3381
SET DISK2 NAME=dska_02
CABLE IPC0 3 DISK3
SET DISK3 TYPE=3381
SET DISK3 NAME=dska_03
SET MSP0 NAME=MSP0
CABLE IOM0 014 MSP0 0
CABLE IOM1 014 MSP0 1
CABLE MSP0 1 DISK4
SET disk4 TYPE=d501
SET disk4 NAME=dskb_01
CABLE MSP0 2 DISK5
SET DISK5 TYPE=d501
SET DISK5 NAME=dskb_02
CABLE MSP0 3 DISK6
SET DISK6 TYPE=d501
SET DISK6 NAME=dskb_03
CABLE MSP0 4 DISK7
SET DISK7 TYPE=d501
SET DISK7 NAME=dskb_04
CABLE MSP0 5 DISK8
SET DISK8 TYPE=d451
SET DISK8 NAME=dskb_05
CABLE MSP0 6 DISK9
SET DISK9 TYPE=d451
SET DISK9 NAME=dskb_06
CABLE MSP0 7 DISK10
SET DISK10 TYPE=d451
SET DISK10 NAME=dskb_07
CABLE MSP0 8 DISK11
SET DISK11 TYPE=d451
SET DISK11 NAME=dskb_08
CABLE MSP0 9 DISK12
SET DISK12 TYPE=d500
SET DISK12 NAME=dskb_09
CABLE MSP0 10 DISK13
SET DISK13 TYPE=d500
```

```
SET DISK13 NAME=dskb_10
CABLE IPC0 4 DISK14
SET DISK14 TYPE=3381
SET DISK14 NAME=dska_04
CABLE IPC0 5 DISK15
SET DISK15 TYPE=3381
SET DISK15 NAME=dska_05
CABLE IPC0 6 DISK16
SET DISK16 TYPE=3381
SET DISK16 NAME=dska_06
CABLE IPC0 7 DISK17
SET DISK17 TYPE=3381
SET DISK17 NAME=dska_07
CABLE IPC0 8 DISK18
SET DISK18 TYPE=3381
SET DISK18 NAME=dska_08
CABLE IPC0 9 DISK19
SET DISK19 TYPE=3381
SET DISK19 NAME=dska_09
CABLE IPC0 10 DISK20
SET DISK20 TYPE=3381
SET DISK20 NAME=dska_10
CABLE IPC0 11 DISK21
SET DISK21 TYPE=3381
SET DISK21 NAME=dska_11
CABLE IPC0 12 DISK22
SET DISK22 TYPE=3381
SET DISK22 NAME=dska_12
CABLE IPC0 13 DISK23
SET DISK23 TYPE=3381
SET DISK23 NAME=dska_13
CABLE IPC0 14 DISK24
SET DISK24 TYPE=3381
SET DISK24 NAME=dska_14
CABLE IPC0 15 DISK25
SET DISK25 TYPE=3381
SET DISK25 NAME=dska_15
CABLE IOMA 036 OPC0
CABLE IOMA 053 OPC1
SET OPC1 CONFIG=MODEL=m6601
CABLE IOMA 020 FNPD
CABLE IOMA 021 FNPA
CABLE IOMA 022 FNPB
CABLE IOMA 023 FNPC
CABLE IOMA 024 FNPE
CABLE IOMA 025 FNPF
CABLE IOMA 026 FNPG
CABLE IOMA 027 FNPH
CABLE IOM0 015 URP0
SET URP0 NAME=urpa
CABLE URP0 1 RDR0
SET RDR0 NAME=rdra
CABLE IOM0 016 URP1
SET URP1 NAME=urpb
CABLE URP1 1 PUN0
```

```
SET PUN0 NAME=puna
CABLE IOM0 017 URP2
SET URP2 NAME=urpc
CABLE URP2 1 PRT0
SET PRT0 NAME=prta
CABLE IOMA 050 URP3
SET URP3 NAME=urpd
CABLE URP3 1 PRT1
SET PRT1 NAME=prtb
CABLE IOMA 051 URP4
SET URP4 NAME=urpe
CABLE URP4 1 PRT2
SET PRT2 NAME=prtc
CABLE IOMA 052 URP5
SET URP5 NAME=urpf
CABLE URP5 1 PRT3
SET PRT3 NAME=prtd
CABLE IOMA 055 URP6
SET URP6 NAME=urpg
CABLE URP6 1 RDRB
SET RDR1 NAME=rdrb
CABLE IOMA 056 URP7
SET URP7 NAME=urph
CABLE URP7 1 RDRC
SET RDR2 NAME=rdrc
CABLE IOMA 057 URP8
SET URP8 NAME=urpi
CABLE URP8 1 PUNB
SET PUN1 NAME=punb
CABLE IOMA 060 URP9
SET URP9 NAME=urpj
CABLE URP9 1 PUNC
SET PUN2 NAME=punc
CABLE SCU0 0 IOM0 0
CABLE SCU1 0 IOM0 1
CABLE SCU2 0 IOM0 2
CABLE SCU3 0 IOM0 3
CABLE SCU0 1 IOM1 0
CABLE SCU1 1 IOM1 1
CABLE SCU2 1 IOM1 2
CABLE SCU3 1 IOM1 3
CABLE SCU0 7 CPU0 0
CABLE SCU0 6 CPU1 0
CABLE SCU0 5 CPU2 0
CABLE SCU0 4 CPU3 0
CABLE SCU0 3 CPU4 0
CABLE SCU0 2 CPU5 0
CABLE SCU1 7 CPU0 1
CABLE SCU1 6 CPU1 1
CABLE SCU1 5 CPU2 1
CABLE SCU1 4 CPU3 1
CABLE SCU1 3 CPU4 1
CABLE SCU1 2 CPU5 1
CABLE SCU2 7 CPU0 2
CABLE SCU2 6 CPU1 2
```

```
CABLE SCU2 5 CPU2 2
CABLE SCU2 4 CPU3 2
CABLE SCU2 3 CPU4 2
CABLE SCU2 2 CPU5 2
CABLE SCU3 7 CPU0 3
CABLE SCU3 6 CPU1 3
CABLE SCU3 5 CPU2 3
CABLE SCU3 4 CPU3 3
CABLE SCU3 3 CPU4 3
CABLE SCU3 2 CPU5 3
SET CPU0 RESET
SET SCU0 RESET
SET SCU1 RESET
SET SCU2 RESET
SET SCU3 RESET
SET IOM0 RESET
SET CPU NUNITS=6
SET SYS CONFIG=CONNECT_TIME=-1
```

Default Base System Configuration

The following listing of configuration details, generated by the "**SHOW CONFIGURATION**" command, shows the configuration state of the simulator immediately after the execution of the *default base system script* (documented in the previous section of this chapter).

This configuration state is the default state upon simulator startup. The simulator may also be explicitly reconfigured to this state after startup by using the "**DEFAULT_BASE_SYSTEM**" command.

- **NOTE**: The configuration details of individual FNP lines have been excluded for brevity.
- See the documentation for the "**DEFAULT_BASE_SYSTEM**" command (in the **Simulator Command Reference** chapter) and the "**Default Base System Script**" section of this chapter for additional details.

```
DPS8/M simulator configuration
CPU  6 units
   CPU0
     16MW
   CPU1
     16MW
   CPU2
     16MW
   CPU3
     16MW
   CPU4
     16MW
   CPU5
     16MW
IOM  2 units
   IOM0
   IOM1
TAPE    17 units
   TAPE0
    status   : not attached
    name     : default
```

```
 capacity : 40MB
TAPE1
 status   : not attached
 name     : tapa_01
 capacity : 40MB
TAPE2
 status   : not attached
 name     : tapa_02
 capacity : 40MB
TAPE3
 status   : not attached
 name     : tapa_03
 capacity : 40MB
TAPE4
 status   : not attached
 name     : tapa_04
 capacity : 40MB
TAPE5
 status   : not attached
 name     : tapa_05
 capacity : 40MB
TAPE6
 status   : not attached
 name     : tapa_06
 capacity : 40MB
TAPE7
 status   : not attached
 name     : tapa_07
 capacity : 40MB
TAPE8
 status   : not attached
 name     : tapa_08
 capacity : 40MB
TAPE9
 status   : not attached
 name     : tapa_09
 capacity : 40MB
TAPE10
 status   : not attached
 name     : tapa_10
 capacity : 40MB
TAPE11
 status   : not attached
 name     : tapa_11
 capacity : 40MB
TAPE12
 status   : not attached
 name     : tapa_12
 capacity : 40MB
TAPE13
 status   : not attached
 name     : tapa_13
 capacity : 40MB
TAPE14
 status   : not attached
```

```
     name     : tapa_14
     capacity : 40MB
   TAPE15
     status   : not attached
     name     : tapa_15
     capacity : 40MB
   TAPE16
     status   : not attached
     name     : tapa_16
     capacity : 40MB
MTP  1 unit
     boot     : 0
     name     : MTP0
FNP  8 units
   FNP0
     FNP IPC name: fnp-a
             name: default
   FNP1
     FNP IPC name: fnp-b
             name: default
   FNP2
     FNP IPC name: fnp-c
             name: default
   FNP3
     FNP IPC name: fnp-d
             name: default
   FNP4
     FNP IPC name: fnp-e
             name: default
   FNP5
     FNP IPC name: fnp-f
             name: default
   FNP6
     FNP IPC name: fnp-g
             name: default
   FNP7
     FNP IPC name: fnp-h
             name: default
DISK    26 units
   DISK0
     status   : not attached 451KW
     type     : 3381
     name     : dska_00
   DISK1
     status   : not attached 451KW
     type     : 3381
     name     : dska_01
   DISK2
     status   : not attached 451KW
     type     : 3381
     name     : dska_02
   DISK3
     status   : not attached 451KW
     type     : 3381
     name     : dska_03
```

```
DISK4
 status   : not attached 1075KW
 type     : d501
 name     : dskb_01
DISK5
 status   : not attached 1075KW
 type     : d501
 name     : dskb_02
DISK6
 status   : not attached 1075KW
 type     : d501
 name     : dskb_03
DISK7
 status   : not attached 1075KW
 type     : d501
 name     : dskb_04
DISK8
 status   : not attached 616KW
 type     : d451
 name     : dskb_05
DISK9
 status   : not attached 616KW
 type     : d451
 name     : dskb_06
DISK10
 status   : not attached 616KW
 type     : d451
 name     : dskb_07
DISK11
 status   : not attached 616KW
 type     : d451
 name     : dskb_08
DISK12
 status   : not attached 618KW
 type     : d500
 name     : dskb_09
DISK13
 status   : not attached 618KW
 type     : d500
 name     : dskb_10
DISK14
 status   : not attached 451KW
 type     : 3381
 name     : dska_04
DISK15
 status   : not attached 451KW
 type     : 3381
 name     : dska_05
DISK16
 status   : not attached 451KW
 type     : 3381
 name     : dska_06
DISK17
 status   : not attached 451KW
 type     : 3381
```

```
     name     : dska_07
   DISK18
     status   : not attached 451KW
     type     : 3381
     name     : dska_08
   DISK19
     status   : not attached 451KW
     type     : 3381
     name     : dska_09
   DISK20
     status   : not attached 451KW
     type     : 3381
     name     : dska_10
   DISK21
     status   : not attached 451KW
     type     : 3381
     name     : dska_11
   DISK22
     status   : not attached 451KW
     type     : 3381
     name     : dska_12
   DISK23
     status   : not attached 451KW
     type     : 3381
     name     : dska_13
   DISK24
     status   : not attached 451KW
     type     : 3381
     name     : dska_14
   DISK25
     status   : not attached 451KW
     type     : 3381
     name     : dska_15
IPC  2 units
   IPC0
     name     : IPC0
   IPC1
     name     : default
MSP  2 units
   MSP0
     name     : MSP0
   MSP1
     name     : default
SCU  4 units
   SCU0
   SCU1
   SCU2
   SCU3
OPC  2 units
   OPC0
     autoinput: empty
     flags    : autoaccept=0, noempty=0, attn_flush=1
     name     : OPC0
     port     : disabled
     address  : any
```

```
     password : MulticsRulez
    OPC1
     autoinput: empty
     flags    : autoaccept=0, noempty=0, attn_flush=1
     name     : OPC1
     port     : disabled
     address  : any
     password : MulticsRulez
  URP 10 units
    URP0
     status   : not attached
     name     : urpa
    URP1
     status   : not attached
     name     : urpb
    URP2
     status   : not attached
     name     : urpc
    URP3
     status   : not attached
     name     : urpd
    URP4
     status   : not attached
     name     : urpe
    URP5
     status   : not attached
     name     : urpf
    URP6
     status   : not attached
     name     : urpg
    URP7
     status   : not attached
     name     : urph
    URP8
     status   : not attached
     name     : urpi
    URP9
     status   : not attached
     name     : urpj
  RDR  3 units
    RDR0
     status   : not attached
     name     : rdra
    RDR1
     status   : not attached
     name     : rdrb
    RDR2
     status   : not attached
     name     : rdrc
  PUN  3 units
    PUN0
     status   : not attached
     name     : puna
     logcards : 0
    PUN1
```

```
     status   : not attached
     name     : punb
     logcards : 0
    PUN2
     status   : not attached
     name     : punc
     logcards : 0
PRT  4 units
   PRT0
     status   : not attached
     name     : prta
     model    : 1600
     split    : 0
   PRT1
     status   : not attached
     name     : prtb
     model    : 1600
     split    : 0
   PRT2
     status   : not attached
     name     : prtc
     model    : 1600
     split    : 0
   PRT3
     status   : not attached
     name     : prtd
     model    : 1600
     split    : 0
```

Default Base System Cable Dump

The following cabling configuration listing, generated by the "**CABLE DUMP**" command, shows the cabling configuration of the simulator immediately after the execution of the *default base system script* (documented in an earlier section of this chapter).

- See the documentation for the "**CABLE**" commands ("CABLE", "UNCABLE", "CABLE_RIPOUT", "CABLE_SHOW", "CABLE DUMP", "CABLE GRAPH") in the **Simulator Command Reference** chapter for additional details.

```
SCU <--> IOM
   SCU port --> IOM port
     0    0       0    0
     0    1       1    0
     1    0       0    1
     1    1       1    1
     2    0       0    2
     2    1       1    2
     3    0       0    3
     3    1       1    3
   IOM port --> SCU port
     0    0       0    0
     0    1       1    0
     0    2       2    0
     0    3       3    0
```

```
     1    0       0    1
     1    1       1    1
     1    2       2    1
     1    3       3    1

SCU <--> CPU
   SCU port --> CPU port
     0    2       5    0
     0    3       4    0
     0    4       3    0
     0    5       2    0
     0    6       1    0
     0    7       0    0
     1    2       5    1
     1    3       4    1
     1    4       3    1
     1    5       2    1
     1    6       1    1
     1    7       0    1
     2    2       5    2
     2    3       4    2
     2    4       3    2
     2    5       2    2
     2    6       1    2
     2    7       0    2
     3    2       5    3
     3    3       4    3
     3    4       3    3
     3    5       2    3
     3    6       1    3
     3    7       0    3
   CPU port --> SCU port subport
     0    0       0    7     0
     0    1       1    7     0
     0    2       2    7     0
     0    3       3    7     0
     1    0       0    6     0
     1    1       1    6     0
     1    2       2    6     0
     1    3       3    6     0
     2    0       0    5     0
     2    1       1    5     0
     2    2       2    5     0
     2    3       3    5     0
     3    0       0    4     0
     3    1       1    4     0
     3    2       2    4     0
     3    3       3    4     0
     4    0       0    3     0
     4    1       1    3     0
     4    2       2    3     0
     4    3       3    3     0
     5    0       0    2     0
     5    1       1    2     0
     5    2       2    2     0
```

```
    5    3       3    2     0

IOM <--> controller
                ctlr       ctlr  chan
   IOM chan -->  idx  port  type  type      device      board    command
    0   10        0     0 MTP     PSI      0x4cf4c0   0x86b1c0   0x44a400
    0   11        0     0 IPC     PSI      0x4cf580   0x86c2c0   0x41a040
    0   12        0     0 MSP     PSI      0x4cf640   0x86ba40   0x41a040
    0   13        0     0 URP     PSI      0x4cf700   0x4ceb80   0x451ef0
    0   14        1     0 URP     PSI      0x4cf700   0x4cec08   0x451ef0
    0   15        2     0 URP     PSI      0x4cf700   0x4cec90   0x451ef0
    0   16        3     0 FNP     Direct   0x4cdd40   0x4cdf98   0x431000
    0   17        0     0 FNP     Direct   0x4cdd40   0x4cde00   0x431000
    0   18        1     0 FNP     Direct   0x4cdd40   0x4cde88   0x431000
    0   19        2     0 FNP     Direct   0x4cdd40   0x4cdf10   0x431000
    0   20        4     0 FNP     Direct   0x4cdd40   0x4ce020   0x431000
    0   21        5     0 FNP     Direct   0x4cdd40   0x4ce0a8   0x431000
    0   22        6     0 FNP     Direct   0x4cdd40   0x4ce130   0x431000
    0   23        7     0 FNP     Direct   0x4cdd40   0x4ce1b8   0x431000
    0   30        0     0 OPC     CPI      0x4ce680   0x4ce740   0x40e940
    0   40        3     0 URP     PSI      0x4cf700   0x4ced18   0x451ef0
    0   41        4     0 URP     PSI      0x4cf700   0x4ceda0   0x451ef0
    0   42        5     0 URP     PSI      0x4cf700   0x4cee28   0x451ef0
    0   43        1     0 OPC     CPI      0x4ce680   0x4ce7c8   0x40e940
    0   45        6     0 URP     PSI      0x4cf700   0x4ceeb0   0x451ef0
    0   46        7     0 URP     PSI      0x4cf700   0x4cef38   0x451ef0
    0   47        8     0 URP     PSI      0x4cf700   0x4cefc0   0x451ef0
    0   48        9     0 URP     PSI      0x4cf700   0x4cf048   0x451ef0
    1   10        0     1 MTP     PSI      0x4cf4c0   0x86b1c0   0x44a400
    1   11        0     1 IPC     PSI      0x4cf580   0x86c2c0   0x41a040
    1   12        0     1 MSP     PSI      0x4cf640   0x86ba40   0x41a040
  MTP  port --> IOM channel
    0    0       0   10
    0    1       1   10
  MSP  port --> IOM channel
    0    0       0   12
    0    1       1   12
  IPC  port --> IOM channel
    0    0       0   11
    0    1       1   11
  URP  port --> IOM channel
    0    0       0   13
    1    0       0   14
    2    0       0   15
    3    0       0   40
    4    0       0   41
    5    0       0   42
    6    0       0   45
    7    0       0   46
    8    0       0   47
    9    0       0   48
  FNP  port --> IOM channel
    0    0       0   17
    1    0       0   18
    2    0       0   19
```

```
    3    0       0   16
    4    0       0   20
    5    0       0   21
    6    0       0   22
    7    0       0   23
  DIA  port --> IOM channel
  OPC  port --> IOM channel
    0    0       0   30
    1    0       0   43

controller <--> device
  MTP  dev_code --> TAPE   command
    0     1           1   0x44a400
    0     2           2   0x44a400
    0     3           3   0x44a400
    0     4           4   0x44a400
    0     5           5   0x44a400
    0     6           6   0x44a400
    0     7           7   0x44a400
    0     8           8   0x44a400
    0     9           9   0x44a400
    0    10          10   0x44a400
    0    11          11   0x44a400
    0    12          12   0x44a400
    0    13          13   0x44a400
    0    14          14   0x44a400
    0    15          15   0x44a400
    0    16          16   0x44a400
  TAPE --> MTP  dev_code type
    1       0      1      MTP
    2       0      2      MTP
    3       0      3      MTP
    4       0      4      MTP
    5       0      5      MTP
    6       0      6      MTP
    7       0      7      MTP
    8       0      8      MTP
    9       0      9      MTP
   10       0     10      MTP
   11       0     11      MTP
   12       0     12      MTP
   13       0     13      MTP
   14       0     14      MTP
   15       0     15      MTP
   16       0     16      MTP
  IPC  dev_code --> DISK   command
    0     0           0   0x41a040
    0     1           1   0x41a040
    0     2           2   0x41a040
    0     3           3   0x41a040
    0     4          14   0x41a040
    0     5          15   0x41a040
    0     6          16   0x41a040
    0     7          17   0x41a040
    0     8          18   0x41a040
```

```
    0     9           19   0x41a040
    0    10           20   0x41a040
    0    11           21   0x41a040
    0    12           22   0x41a040
    0    13           23   0x41a040
    0    14           24   0x41a040
    0    15           25   0x41a040
  MSP  dev_code --> DISK   command
    0     1            4   0x41a040
    0     2            5   0x41a040
    0     3            6   0x41a040
    0     4            7   0x41a040
    0     5            8   0x41a040
    0     6            9   0x41a040
    0     7           10   0x41a040
    0     8           11   0x41a040
    0     9           12   0x41a040
    0    10           13   0x41a040
  DISK --> CTLR dev_code type
    0       0      0      IPC
    1       0      1      IPC
    2       0      2      IPC
    3       0      3      IPC
    4       0      1      MSP
    5       0      2      MSP
    6       0      3      MSP
    7       0      4      MSP
    8       0      5      MSP
    9       0      6      MSP
   10       0      7      MSP
   11       0      8      MSP
   12       0      9      MSP
   13       0     10      MSP
   14       0      4      IPC
   15       0      5      IPC
   16       0      6      IPC
   17       0      7      IPC
   18       0      8      IPC
   19       0      9      IPC
   20       0     10      IPC
   21       0     11      IPC
   22       0     12      IPC
   23       0     13      IPC
   24       0     14      IPC
   25       0     15      IPC
  URP  dev_code --> URP    command
    0     1            0   0x419900
    1     1            0   0x41b800
    2     1            0   0x44d440
    3     1            1   0x44d440
    4     1            2   0x44d440
    5     1            3   0x44d440
    6     1            1   0x419900
    7     1            2   0x419900
    8     1            1   0x41b800
```

```
    9     1           2   0x41b800
  RDR  --> URP  dev_code type
    0       0      1      URP
    1       6      1      URP
    2       7      1      URP
  PUN  --> URP  dev_code type
    0       1      1      URP
    1       8      1      URP
    2       9      1      URP
  PRT  --> URP  dev_code type
    0       2      1      URP
    1       3      1      URP
    2       4      1      URP
    3       5      1      URP
```

Legal

DPS8M Omnibus Documentation Licensing Terms

DPS8M Omnibus Documentation-specific Content Licenses

The complete **DPS8M Omnibus Documentation** ("*this documentation*") is licensed under **The DPS8M Documentation License** (version `0.954`, or at your option, any later version), with the following exceptions.

- **Original text and graphics** (including examples, "recipes", code, diagrams, drawings, outlines, charts, and illustrations) within this documentation, unless otherwise noted, are ***dual licensed*** under:
    1. **The DPS8M Documentation License** (version `0.954` of 3 July 2022, or at your option, any later version)
    2. The **General Attribution License**
- **Third-party content** within this documentation is copyrighted by their respective owners and licensed as indicated in the following disclosures.

The DPS8M Documentation License

The DPS8M Documentation License

Version 0.954 of 3 July 2022

General Attribution License

General Attribution License

Multics License

This documentation may adapt, include, and/or incorporate **Multics** program code and/or documentation, which is distributed under the **Multics License**.

Multics License

- This edition of the **Multics** software materials and documentation is provided and donated to **The Massachusetts Institute of Technology** by **Group BULL** including **BULL HN Information Systems, Inc**. as a contribution to computer science knowledge.
- This donation is made also to give evidence of the common contributions of **The Massachusetts Institute of Technology**, **Bell Laboratories**, **General Electric**, **Honeywell Information Systems, Inc**., **Honeywell BULL, Inc**., **Groupe BULL** and **BULL HN Information Systems, Inc**. to the development of this operating system.
- **Multics** development was initiated by **The Massachusetts Institute of Technology Project MAC** (1963-1970), renamed the **MIT Laboratory for Computer Science and Artificial Intelligence** in the mid 1970s, under the leadership of *Professor Fernando José Corbató*. Users consider that **Multics** provided the best software architecture for managing computer hardware properly and for executing programs. Many subsequent operating systems incorporated **Multics** principles.
- **Multics** was distributed in 1975 to 2000 by **Group Bull** in Europe, and in the U.S. by **Bull HN Information Systems, Inc**., as successor in interest by change in name only to **Honeywell Bull, Inc**. and **Honeywell Information Systems, Inc**.

General Attribution License (Multics Rings Logo)

This documentation may adapt, include, and/or incorporate the **Multics Rings Logo**, a stylized rendering of concentric rings, representing the logical structure of the protection domains designed specifically for the Multics operating system.

The stylized rendering and the Adobe® PostScript™ code used to produce it were created by **Stanley R. Zanarotti** and distributed under the **General Attribution License**.

- The **Multics Rings Logo** used herein has been altered for use in this documentation by **The DPS8M Development Team**.

Multics Rings Logo License

BSD License (Eisvogel)

This documentation may adapt, include, and/or incorporate **Eisvogel**, a LaTeX template for *pandoc*, which is distributed under a three-clause BSD license.

ISC License (Pagebreak)

This documentation may adapt, include, and/or incorporate **Pagebreak**, a Lua filter for *pandoc*, which is distributed under the **ISC License**.

SIL™ Open Font License 1.1 (Source™ Code Pro)

This documentation may embed **Source™ Code Pro**, a font, which is distributed under the **SIL™ Open Font License 1.1**.

SIL™ Open Font License Version 1.1

- Copyright © 2010-2019 **Adobe®** (*adobe.com*), with Reserved Font Name ***Source™***. ***Source™*** is a trademark of Adobe® in the United States and/or other countries. All rights reserved.

PREAMBLE: The goals of the Open Font License (*OFL*) are to stimulate worldwide development of collaborative font projects, to support the font creation efforts of academic and linguistic communities, and to provide a free and open framework in which fonts may be shared and improved in partnership with others. The *OFL* allows the licensed fonts to be used, studied, modified and redistributed freely as long as they are not sold by themselves. The fonts, including any derivative works, can be bundled, embedded, redistributed and/or sold with any software provided that any reserved names are not used by derivative works. The fonts and derivatives, however, cannot be released under any other type of license. The requirement for fonts to remain under this license does not apply to any document created using the fonts or their derivatives.

DEFINITIONS: "Font Software" refers to the set of files released by the Copyright Holder(s) under this license and clearly marked as such. This may include source files, build scripts and documentation. "Reserved Font Name" refers to any names specified as such after the copyright statement(s). "Original Version" refers to the collection of Font Software components as distributed by the Copyright Holder(s). "Modified Version" refers to any derivative made by adding to, deleting, or substituting – in part or in whole – any of the components of the Original Version, by changing formats or by porting the Font Software to a new environment. "Author" refers to any designer, engineer, programmer, technical writer or other person who contributed to the Font Software.

PERMISSION & CONDITIONS: Permission is hereby granted, free of charge, to any person obtaining a copy of the Font Software, to use, study, copy, merge, embed, modify, redistribute, and sell modified and unmodified copies of the Font Software, subject to the following conditions:

1. Neither the Font Software nor any of its individual components, in Original or Modified Versions, may be sold by itself.
2. Original or Modified Versions of the Font Software may be bundled, redistributed and/or sold with any software, provided that each copy contains the above copyright notice and this license. These can be included either as stand-alone text files, human-readable headers or in the appropriate machine-readable metadata fields within text or binary files as long as those fields can be easily viewed by the user.
3. No Modified Version of the Font Software may use the Reserved Font Name(s) unless explicit written permission is granted by the corresponding Copyright Holder. This restriction only applies to the primary font name as presented to the users.
4. The name(s) of the Copyright Holder(s) or the Author(s) of the Font Software shall not be used to promote, endorse or advertise any Modified Version, except to acknowledge the contribution(s) of the Copyright Holder(s) and the Author(s) or with their explicit written permission.
5. The Font Software, modified or unmodified, in part or in whole, must be distributed entirely under this license, and must not be distributed under any other license. The requirement for fonts to remain under this license does not apply to any document created using the Font Software.

TERMINATION: This license becomes null and void if any of the above conditions are not met.

DISCLAIMER: **THE FONT SOFTWARE IS PROVIDED "*AS IS*", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF COPYRIGHT, PATENT, TRADEMARK, OR OTHER RIGHT. IN NO EVENT SHALL THE COPYRIGHT HOLDER BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, INCLUDING ANY GENERAL, SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF THE USE OR INABILITY TO USE THE FONT SOFTWARE OR FROM OTHER DEALINGS IN THE FONT SOFTWARE.**

SIL™ Open Font License 1.1 (Source™ Sans Pro)

This documentation may embed **Source™ Sans Pro**, a font, which is distributed under the **SIL™ Open Font License 1.1**.

SIL™ Open Font License Version 1.1

- Copyright © 2010-2020 **Adobe®** (*adobe.com*), with Reserved Font Name ***Source™***. ***Source™*** is a trademark of Adobe® in the United States and/or other countries. All rights reserved.

PREAMBLE: The goals of the Open Font License (*OFL*) are to stimulate worldwide development of collaborative font projects, to support the font creation efforts of academic and linguistic communities, and to provide a free and open framework in which fonts may be shared and improved in partnership with others. The *OFL* allows the licensed fonts to be used, studied, modified and redistributed freely as long as they are not sold by themselves. The fonts, including any derivative works, can be bundled, embedded, redistributed and/or sold with any software provided that any reserved names are not used by derivative works. The fonts and derivatives, however, cannot be released under any other type of license. The requirement for fonts to remain under this license does not apply to any document created using the fonts or their derivatives.

DEFINITIONS: "Font Software" refers to the set of files released by the Copyright Holder(s) under this license and clearly marked as such. This may include source files, build scripts and documentation. "Reserved Font Name" refers to any names specified as such after the copyright statement(s). "Original Version" refers to the collection of Font Software components as distributed by the Copyright Holder(s). "Modified Version" refers to any derivative made by adding to, deleting, or substituting – in part or in whole – any of the components of the Original Version, by changing formats or by porting the Font Software to a new environment. "Author" refers to any designer, engineer, programmer, technical writer or other person who contributed to the Font Software.

PERMISSION & CONDITIONS: Permission is hereby granted, free of charge, to any person obtaining a copy of the Font Software, to use, study, copy, merge, embed, modify, redistribute, and sell modified and unmodified copies of the Font Software, subject to the following conditions:

1. Neither the Font Software nor any of its individual components, in Original or Modified Versions, may be sold by itself.
2. Original or Modified Versions of the Font Software may be bundled, redistributed and/or sold with any software, provided that each copy contains the above copyright notice and this license. These can be included either as stand-alone text files, human-readable headers or in the appropriate machine-readable metadata fields within text or binary files as long as those fields can be easily viewed by the user.
3. No Modified Version of the Font Software may use the Reserved Font Name(s) unless explicit written permission is granted by the corresponding Copyright Holder. This restriction only applies to the primary font name as presented to the users.
4. The name(s) of the Copyright Holder(s) or the Author(s) of the Font Software shall not be used to promote, endorse or advertise any Modified Version, except to acknowledge the contribution(s) of the Copyright Holder(s) and the Author(s) or with their explicit written permission.
5. The Font Software, modified or unmodified, in part or in whole, must be distributed entirely under this license, and must not be distributed under any other license. The requirement for fonts to remain under this license does not apply to any document created using the Font Software.

TERMINATION: This license becomes null and void if any of the above conditions are not met.

DISCLAIMER: **THE FONT SOFTWARE IS PROVIDED "*AS IS*", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF COPYRIGHT, PATENT, TRADEMARK, OR OTHER RIGHT. IN NO EVENT SHALL THE COPYRIGHT HOLDER BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, INCLUDING ANY GENERAL, SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF THE USE OR INABILITY TO USE THE FONT SOFTWARE OR FROM OTHER DEALINGS IN THE FONT SOFTWARE.**

DPS8M Software Licensing Terms

DPS8M Simulator

ICU License

- The **DPS8M Simulator** ("**DPS8M**") is distributed under the **ICU License**, version 1.8.1 or later.

ICU License — ICU 1.8.1 to ICU 57.1

COPYRIGHT AND PERMISSION NOTICE

- Copyright © 2006-2022 **Michael Mondy**, **Harry Reed**, **Charles Anthony**, and others
- Copyright © 2012 **Dave Jordan**
- Copyright © 2015-2016 **Craig Ruff**
- Copyright © 2015-2022 **Eric Swenson**
- Copyright © 2016 **Jean-Michel Merliot**
- Copyright © 2017-2022 **Jeffrey H. Johnson**
- Copyright © 2018-2021 **Juergen Weiss**
- Copyright © 2021 **Dean S. Anderson**
- Copyright © 2021-2022 **The DPS8M Development Team**

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

THE SOFTWARE IS PROVIDED "*AS IS*", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

All trademarks and registered trademarks mentioned herein are the property of their respective owners.

FSF All Permissive License

- The **DPS8M Simulator** ("**DPS8M**") distribution includes various build, configuration, and miscellaneous files.
- These supporting files are distributed under the terms of the **FSF All Permissive License**.

Copying and distribution of this file, with or without modification, are permitted in any medium without royalty provided the copyright notice and this notice are preserved. This file is offered "*AS-IS*", without any warranty.

Multics License

- The **DPS8M Simulator** ("**DPS8M**") **source code** may contain **code comments** that adapt, include, and/or incorporate Multics program code and/or documentation distributed under the **Multics License**.
- In the event of any discrepancy between the source code comments and the original Multics materials, the original Multics materials should be considered authoritative unless otherwise noted.
- The **Multics License** covers **only** the simulator's source code comments, and **is not** applicable to compiled object code or binary distributions of the simulator.

Multics License

- This edition of the **Multics** software materials and documentation is provided and donated to **The Massachusetts Institute of Technology** by **Group BULL** including **BULL HN Information Systems, Inc**. as a contribution to computer science knowledge.
- This donation is made also to give evidence of the common contributions of **The Massachusetts Institute of Technology**, **Bell Laboratories**, **General Electric**, **Honeywell Information Systems, Inc**., **Honeywell BULL, Inc**., **Groupe BULL** and **BULL HN Information Systems, Inc**. to the development of this operating system.
- **Multics** development was initiated by **The Massachusetts Institute of Technology Project MAC** (1963-1970), renamed the **MIT Laboratory for Computer Science and Artificial Intelligence** in the mid 1970s, under the leadership of *Professor Fernando José Corbató*. Users consider that **Multics** provided the best software architecture for managing computer hardware properly and for executing programs. Many subsequent operating systems incorporated **Multics** principles.
- **Multics** was distributed in 1975 to 2000 by **Group Bull** in Europe, and in the U.S. by **Bull HN Information Systems, Inc**., as successor in interest by change in name only to **Honeywell Bull, Inc**. and **Honeywell Information Systems, Inc**.

Permission to use, copy, modify, and distribute these programs and their documentation for any purpose and without fee is hereby granted, provided that this copyright notice and the above historical background appear in all copies and that both the copyright notice and historical background and this permission notice appear in supporting documentation, and that the names of *MIT*, *HIS*, *BULL*, or *BULL HN* not be used in advertising or publicity pertaining to distribution of the programs without specific prior written permission.

Third-party Software

- **DPS8M** may incorporate, adapt, or utilize software from third-parties.
- Third-party software is copyrighted by their respective owners and licensed as indicated in the following disclosures.

libuv

- **libuv** is a portable high-performance platform support library, with a focus on asynchronous-I/O based on event loops, originally developed by Ben Noordhuis to support the *Node.js*® runtime. It is distributed under the terms of the **MIT License**.

libuv License

Unifdef

- **Unifdef** is a utility that selectively processes conditional C preprocessor directives derived from software contributed to the *Computer Systems Research Group* at the *University of California, Berkeley* by **Dave Yost**. It was rewritten to support *ANSI C* by **Tony Finch**, with additional modifications by **The DPS8M Development Team**. It is distributed under a **two-clause BSD license**.

Unifdef License

- Copyright © 2002-2020 **Tony Finch** <dot@dotat.at>
- Copyright © 2021-2022 **The DPS8M Development Team**

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS "*AS IS*" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

UDPLib

- **UDPLib** is a library that implements the BBN ARPAnet IMP/TIP Modem/Host Interface over UDP. It was written by Robert Armstrong, and is distributed under the terms of the **X11 license**.

UDPLib License

UtHash

- **UtHash** (*UtList*, *UtArray*, and *UtHash*), a hash table for C structures, was written by Troy D. Hanson and customized by **The DPS8M Development Team**. It is distributed under the terms of a **one-clause BSD license**.

UtHash License

Open SIMH

- **Open SIMH** is a portable systems simulation framework, written by Robert M. Supnik and others, and distributed under the terms of the **X11 license**.

Open SIMH License

Original code published in 1993-2022, written by Robert M. Supnik.

PunUtil

- **PunUtil** is a utility to process the output of the **DPS8M** punch device. It was written by **Dean S. Anderson** of **The DPS8M Development Team** and is distributed under the **ICU License**.

ICU License — ICU 1.8.1 to ICU 57.1

COPYRIGHT AND PERMISSION NOTICE

- Copyright © 2021-2022 **The DPS8M Development Team**

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

THE SOFTWARE IS PROVIDED “*AS IS*”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

All trademarks and registered trademarks mentioned herein are the property of their respective owners.

Prt2PDF

- **Prt2PDF** is a utility to convert the output of the simulated **DPS8M** line printer device to ISO 32000 Portable Document Format (*PDF*), and is distributed under the **ICU License**. It is derived from **Txt2PDF**, a utility to transform ASCII text files to Adobe Acrobat PDF documents written by **P. G. Womack**.

ICU License — ICU 1.8.1 to ICU 57.1

COPYRIGHT AND PERMISSION NOTICE

- Copyright © 1998 **P. G. Womack, Diss, Norfolk, UK**.
- Copyright © 2013-2016 **Charles Anthony** <charles.unix.pro@gmail.com>
- Copyright © 2006 **John S. Urban, USA**. <urbanjost@comcast.net>
- Copyright © 2021-2022 **The DPS8M Development Team**

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

THE SOFTWARE IS PROVIDED “*AS IS*”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

All trademarks and registered trademarks mentioned herein are the property of their respective owners.

decNumber

- **decNumber** is an *ANSI* C reference implementation of the *IBM General Decimal Arithmetic* standard, implementing the decimal floating-point arithmetic and encoding formats described by the (*revised*) IEEE 754 specification. It was written by *IBM* and *IEEE* Fellow **Mike Cowlishaw**, with contributions by **Matthew Hagerty**, **John Matzka**, **Klaus Kretzschmar**, **Stefan Krah**, and **The DPS8M Development Team**. It is distributed under the terms of the **ICU License**.

ICU License — ICU 1.8.1 to ICU 57.1

COPYRIGHT AND PERMISSION NOTICE

- Copyright © 1995-2010 **International Business Machines Corporation** and others

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

THE SOFTWARE IS PROVIDED “*AS IS*”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

All trademarks and registered trademarks mentioned herein are the property of their respective owners.

LibTELNET

- **LibTELNET** is an implementation of the protocol specified by the Network Working Group of the Internet Engineering Task Force, as described in RFC-854, RFC-855, RFC-1091, RFC-1143, RFC-1408, and RFC-1572 — the TELNET protocol and the Q method for TELNET protocol option negotiation — written by **Sean Middleditch** and other contributors.

- **LibTELNET Redistribution Terms**
  - **LibTELNET** has been identified by **The DPS8M Development Team** as being free of known restrictions under copyright law, including all related and neighboring rights.
  - **The DPS8M Development Team** makes no warranties about **LibTELNET**, and disclaims liability for all uses of **LibTELNET** to the fullest extent permitted by applicable law.
  - **The author or authors of this code dedicate any and all copyright interest in this code to the public domain. We make this dedication for the benefit of the public at large and to the detriment of our heirs and successors. We intend this dedication to be an overt act of relinquishment in perpetuity of all present and future rights to this code under copyright law.**

Line History

- **Line History** is a small and self-contained line editor, implementing *Emacs*-style line editing functionality similar to GNU `readline` or BSD `libedit`. It is derived from the *linenoise* library written by **Salvatore** "***antirez***" **Sanfilippo** and **Pieter Noordhuis** and distributed under a **two-clause BSD license**.

Line History License

BSD Random

- **BSD Random** is a collection of random number generation functions derived from software originally developed for the *Berkeley Software Distribution* by the *Computer Systems Research Group* at the *University of California, Berkeley*, and copyrighted by *The Regents of the University of California*. It is distributed under a **three-clause BSD license**.

BSD Random License

AMD LibM

- **AMD LibM**, a component of the *AMD Optimizing CPU Libraries*, is a library containing a collection of basic math functions optimized for AMD x86-64 processor-based machines, and provides high-performance scalar and vector implementations of many core C99 math functions. It was developed by *Advanced Micro Devices, Inc.*, and is distributed under the terms of a **three-clause BSD license**.

AMD LibM License

- Copyright © 2017-2021 **Advanced Micro Devices, Inc**.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "*AS IS*" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

empty

- **empty** is a utility that provides an interface to execute and/or interact with processes in UNIX® pseudo-terminal sessions (PTYs). It was written by **Mikhail E. Zakharov** and contributors, and is distributed under the terms of a **three-clause BSD license**.

empty License

vmpctool

- **vmpctool** is a utility to query and influence the filesystem cache of most UNIX® and POSIX™ conforming systems. It is derived from the `vmtouch` utility written by **Doug Hoyte** and other contributors, and is distributed under the terms of a **three-clause BSD license**.

vmpctool License

musl libc

- **musl libc** is a lightweight and standards conforming implementation of the C standard library, which includes the interfaces defined in the base language standard, POSIX™, and widely agreed-upon extensions. It was written by **Rich "*dalias*" Felker** and other contributors, and is distributed under the terms of the **MIT License**.

musl libc License

- Copyright © 2005-2020 **Rich Felker**, et al.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

- The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "*AS IS*", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

mcmb

- **mcmb** (*miniature cmb*) is a general-purpose tool for complex combinatorics, derived from the libcmb combinatorics library and the cmb combinatorics utility developed **Devin Teske**. It is distributed under the terms of a **two-clause BSD license**.

mcmb License

novdso

- **novdso** is a wrapper, originally authored by **Daniel Teunis** that, forces the wrapped process to not use the Linux vDSO system call acceleration library (by overwriting the `AT_SYSINFO_EHDR` tag in its auxiliary vector) and allows for wrapped processes to be paused at startup for easier system call tracing. **novdso** is distributed under the terms of the **MIT License**.

novdso License

Fair CPU Dispatch

- **Fair CPU Dispatch** is a collection of routines which modify the behavior of the Intel CPU feature dispatcher to the improve performance and compatibility of the Intel Math Kernel Library (MKL) and code generated by the Intel C/C++ and DPC++/C++ compilers on non-Intel processors. These routines are derived from code provided by **Agner Fog** of the Technical University of Denmark, and distributed under the terms of a **two-clause BSD license**.

Fair CPU Dispatch License

Multics Software Materials and Documentation

- **DPS8M** includes some **Multics** software materials and documentation. These materials and documentation are distributed under the terms of the **Multics License**.

Multics License

- This edition of the **Multics** software materials and documentation is provided and donated to **The Massachusetts Institute of Technology** by **Group BULL** including **BULL HN Information Systems, Inc**. as a contribution to computer science knowledge.
- This donation is made also to give evidence of the common contributions of **The Massachusetts Institute of Technology**, **Bell Laboratories**, **General Electric**, **Honeywell Information Systems, Inc**., **Honeywell BULL, Inc**., **Groupe BULL** and **BULL HN Information Systems, Inc**. to the development of this operating system.
- **Multics** development was initiated by **The Massachusetts Institute of Technology Project MAC** (1963-1970), renamed the **MIT Laboratory for Computer Science and Artificial Intelligence** in the mid 1970s, under the leadership of *Professor Fernando José Corbató*. Users consider that **Multics** provided the best software architecture for managing computer hardware properly and for executing programs. Many subsequent operating systems incorporated **Multics** principles.
- **Multics** was distributed in 1975 to 2000 by **Group Bull** in Europe, and in the U.S. by **Bull HN Information Systems, Inc**., as successor in interest by change in name only to **Honeywell Bull, Inc**. and **Honeywell Information Systems, Inc**.

Scope of Intended Application

- The **DPS8M Simulator** is **not designed or intended** for use in any safety-critical, life-critical, or life-sustaining applications or activities.
- These applications and activities include, but are not limited to, military, nuclear reactor control, power generation and transmission, factory automation, industrial process control, robotics, avionics, aerospace, air traffic control, emergency communications, railway, marine, motor vehicle, fire suppression, pharmaceutical, medical or safety devices, any conventional, nuclear, biological or chemical weaponry, or any other applications or activities that could reasonably be expected to potentially impact human health and safety or lead to environmental damage.

Disclaimer of Liability and Endorsement

- While **The DPS8M Development Team** ("*the Team*") has expended reasonable efforts to make the information available in this document as accurate as possible, the Team makes no claims, promises, or guarantees regarding accuracy, completeness, or adequacy of any information contained in this document.
- **The DPS8M Development Team** expressly disclaims liability for any errors and omissions in the contents of this document.
- **NO WARRANTY OF ANY KIND**, either express or implied, including but not limited to the warranties of non-infringement, merchantability, or fitness for a particular purpose, is given with respect to the contents of this document or the contents of linked external resources. Linked external resources are not under the control of **The DPS8M Development Team**.
- Any reference in this document to any specific entity, process, or service, or the use of any trade, firm, or corporation name, or any links to external resources are provided for information and convenience purposes only, and in no way constitute endorsement, sponsorship, affiliation, or recommendation by **The DPS8M Development Team**.