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

DPS8M X3.1.0: Omnibus Documentation

The DPS8M Development Team and contributors

DPS8M X3.1.0: 2025-05-15 01:17:54 UTC

DPS8M Omnibus Documentation

- **Version** `git_f0d311a99c630e4d7d067d16000b62ddca81fefc`
- **Updated** **2025-05-15 01:17:54 UTC**

Version Information

- This documentation is intended for use with the following software:
  - **DPS8M** **X3.1.0+1**
  - **tap2raw** **1.1.1**
  - **prt2pdf** **3.0.3**
  - **punutil** **0.2**

Generated 2025-05-15 01:51:47 UTC

Contents

DPS8M Omnibus Documentation i

Version Information . . . i

DPS8M Authors and Contributors 1

The DPS8M Development Team . . . 1

Introduction 3

About this manual . . . 3

What is DPS8M? . . . 4

GE/Honeywell/Bull DPS-8/M Processor . . . 4

Processor characteristics . . . 4

Functional organization . . . 5

Appending unit . . . 5

Associative memory assembly . . . 5

Control unit . . . 5

Operation unit . . . 5

Decimal unit . . . 5

Modes of operation . . . 6

Native data sizes . . . 6

Interrupt handling . . . 6

Fault handling . . . 6

Instruction repertoire . . . 7

Registers . . . 7

The DPS8M Simulator . . . 8

Simulator overview . . . 8

Security . . . 8

Supported components . . . 9

Unsupported components . . . 9

Obtaining the simulator 11

Binary distributions . . . 11

Source code distributions . . . 12

Source kits . . . 12

Git repository . . . 12

Git cloning . . . 12

Git mirroring . . . 12

Compiling from source 13

General Information . . . 13

FreeBSD . . . 14

FreeBSD prerequisites . . . 14

Standard FreeBSD compilation . . . 14

Optimized FreeBSD compilation . . . 14

blinkenLights2 on FreeBSD . . . 14

Additional FreeBSD Notes . . . 15

NetBSD . . . 15
NetBSD prerequisites . . . 15
Standard NetBSD compilation . . . 15
Optimized NetBSD compilation . . . 16
Compilation using Clang . . . 16
blinkenLights2 on NetBSD . . . 16
OpenBSD . . . 17
OpenBSD prerequisites . . . 17
Standard OpenBSD compilation . . . 17
Optimized OpenBSD compilation . . . 17
Compilation using Clang . . . 18
Additional OpenBSD Notes . . . 18
DragonFly BSD . . . 18
DragonFly BSD prerequisites . . . 18
Standard DragonFly BSD compilation . . . 19
Optimized DragonFly BSD compilation . . . 19
Compiling using Clang . . . 19
Solaris . . . 19
Solaris prerequisites . . . 20
Solaris compilation . . . 20
GCC . . . 20
Clang . . . 20
Oracle Developer Studio . . . 21
OpenIndiana . . . 21
OpenIndiana prerequisites . . . 21
Standard OpenIndiana compilation . . . 21
Compiling using Clang . . . 21
AIX . . . 22
Recommended compilers . . . 22
Other supported compilers . . . 22
AIX prerequisites . . . 23
Libraries and tools . . . 23
GNU C compilers . . . 23
Clang compilers . . . 23
IBM compiler support . . . 23
AIX compilation . . . 23
IBM Open XL C/C++ for AIX . . . 23
Clang . . . 24
IBM XL C/C++ for AIX . . . 25
GCC . . . 25
Haiku . . . 25
Haiku prerequisites . . . 26
Standard Haiku compilation . . . 26
Compiling using Clang . . . 26
Additional Haiku Notes . . . 26
Linux . . . 26
Linux compilers . . . 27
Linux prerequisites . . . 27
Standard Linux compilation . . . 28
Alternative Linux compilation . . . 28
Clang . . . 28

Intel oneAPI DPC++/C++ . . . 28
AMD Optimizing C/C++ . . . 28
AOCC with AMD Optimized CPU Libraries . . . 29
Oracle Developer Studio . . . 29
IBM Open XL C/C++ for Linux . . . 29
IBM XL C/C++ for Linux . . . 29
NVIDIA HPC SDK C Compiler . . . 30
Arm HPC C/C++ Compiler for Linux . . . 30
ACFL with Arm Performance Libraries . . . 30
Linux cross-compilation . . . 31
IBM Advance Toolchain . . . 31
Arm GNU Toolchain . . . 31
Linux/ARMv7-HF . . . 31
Linux/ARM64 . . . 32
Linux/ARM64BE . . . 32
Linaro GNU Toolchain . . . 32
Linux/ARMv7-HF . . . 32
Linux/ARM64 . . . 33
crosstool-NG . . . 33
Linux/RV64 . . . 33
Linux/i686 . . . 33
Linux/ARMv6-HF . . . 34
Linux/PPC64le . . . 34
Additional Linux Notes . . . 34
macOS . . . 35
macOS prerequisites . . . 35
macOS compilation . . . 35
macOS cross-compilation . . . 35
ARM64 . . . 36
Intel . . . 36
Universal . . . 36
Windows . . . 37
Cygwin . . . 38
Cygwin prerequisites . . . 38
Standard Cygwin compilation . . . 38
Cygwin-hosted cross-compilation to MinGW . . . 39
Windows i686 . . . 39
Windows x86_64 . . . 39
MSYS2 . . . 40
Unix-hosted LLVM-MinGW Clang cross-compilation . . . 40
Windows i686 . . . 40
Windows x86_64 . . . 41
Windows ARM64 . . . 41
Unix-hosted MinGW-w64 GCC cross-compilation . . . 41
Windows i686 . . . 42
Windows x86_64 . . . 42

Simulator Command Reference **43**

! . . . 43
Executing System Commands . . . 43
ASSERT . . . 43

ATTACH (AT) . . . 44
Switches . . . 44
-n . . . 44
-e . . . 44
-r . . . 44
-q . . . 44
-f . . . 44
AUTOINPUT (AI) . . . 45
AUTOINPUT2 (AI2) . . . 45
BOOT (BO) . . . 46
BURST . . . 46
CABLE (C) . . . 46
UNCABLE (U) . . . 47
CABLE_RIPOUT . . . 48
CABLE_SHOW . . . 48
CABLE DUMP . . . 50
CABLE GRAPH . . . 50
Complete Cabling Graph . . . 51
CPU / SCU / IOM Cabling Graph . . . 52
Storage Cabling Graph . . . 53
Controller Cabling Graph . . . 54
CALL . . . 54
CHECKPOLL . . . 55
CLRAUTOINPUT . . . 55
CLRAUTOINPUT2 . . . 55
CONTINUE (CO) . . . 55
DEFAULT_BASE_SYSTEM . . . 55
DETACH (DET) . . . 55
DO . . . 56
Switches . . . 56
-v . . . 56
-e . . . 56
-o . . . 56
-q . . . 56
ECHO . . . 56
EVALUATE . . . 56
EXIT (QUIT, BYE) . . . 57
FNPSERVERADDRESS . . . 57
FNPSERVERPORT . . . 57
FNPSTART . . . 57
GO . . . 58
GOTO . . . 58
IF . . . 58
Conditional Expressions . . . 58
Simulator State Expressions . . . 59
String Comparison Expressions . . . 59
LOAD (UNLOAD) . . . 60
LUF (NOLUF) . . . 60
MOUNT . . . 60
NEXT (N) . . . 60
ON . . . 60

POLL . . . 60
PROCEED (IGNORE) . . . 60
READY . . . 61
RESET . . . 61
RETURN . . . 61
REWIND . . . 62
RUN (RU) . . . 62
SEGLDR . . . 62
SET . . . 62
Logging . . . 62
Switches . . . 63
-N . . . 63
-B . . . 63
Debug Messages . . . 63
Switches . . . 63
-T . . . 63
-A . . . 63
-R . . . 63
-P . . . 63
-N . . . 63
-D . . . 63
-E . . . 63
Environment Variables . . . 64
Command Status Trap Dispatching . . . 64
Command Execution Display . . . 64
Command Error Status Display . . . 64
Command Output Display . . . 64
Local Operator Console . . . 64
Command Prompt . . . 64
Device and Unit Settings . . . 65
CPU Configuration . . . 65
L68 (DPS8M) . . . 65
RESET . . . 65
INITIALIZE . . . 66
INITIALIZEANDCLEAR (IAC) . . . 66
NUNITS . . . 66
KIPS . . . 67
STALL . . . 67
DEBUG (NODEBUG) . . . 68
CONFIG . . . 69
FAULTBASE . . . 69
NUM . . . 69
DATA . . . 69
STOPNUM . . . 70
MODE . . . 70
SPEED . . . 70
PORT . . . 71
ASSIGNMENT . . . 71
INTERLACE . . . 71
ENABLE . . . 72
INIT_ENABLE . . . 72

STORE_SIZE . . . 72
ENABLE_CACHE . . . 72
SDWAM . . . 73
PTWAM . . . 73
DIS_ENABLE . . . 73
STEADY_CLOCK . . . 74
HALT_ON_UNIMPLEMENTED . . . 74
ENABLE_WAM . . . 74
REPORT_FAULTS . . . 74
TRO_ENABLE . . . 75
DRL_FATAL . . . 75
USEMAP . . . 75
ADDRESS . . . 76
PROM_INSTALLED . . . 76
HEX_MODE_INSTALLED . . . 76
CACHE_INSTALLED . . . 77
CLOCK_SLAVE_INSTALLED . . . 77
ENABLE_EMCALL . . . 77
ISOLTS_MODE . . . 77
NODIS . . . 78
L68_MODE . . . 78
IOM Configuration . . . 78
NUNITS . . . 78
DEBUG (NODEBUG) . . . 79
RESET . . . 79
CONFIG . . . 79
PORT . . . 79
ADDR . . . 79
INTERLACE . . . 80
ENABLE . . . 80
INITENABLE . . . 80
HALFSIZE . . . 80
STORE_SIZE . . . 80
MODEL . . . 80
OS . . . 80
BOOT . . . 81
IOM_BASE . . . 81
MULTIPLEX_BASE . . . 81
TAPECHAN . . . 82
CARDCHAN . . . 82
SCUPORT . . . 82
SCU Configuration . . . 82
DEBUG (NODEBUG) . . . 82
NUNITS . . . 82
RESET . . . 83
CONFIG . . . 83
MODE . . . 83
MASKA . . . 83
MASKB . . . 83
PORT0 . . . 83
PORT1 . . . 84

PORT2 . . . 84
PORT3 . . . 84
PORT4 . . . 84
PORT5 . . . 84
PORT6 . . . 84
PORT7 . . . 84
LWRSTORESIZE . . . 84
CYCLIC . . . 84
NEA . . . 85
ONL . . . 85
INT . . . 85
LW . . . 85
ELAPSED_DAYS . . . 85
STEADY_CLOCK . . . 85
BULLET_TIME . . . 85
CLOCK_DELTA . . . 85
TAPE Configuration . . . 86
DEBUG (NODEBUG) . . . 86
DEFAULT_PATH . . . 86
ADD_PATH . . . 87
CAPACITY_ALL . . . 88
CAPACITY . . . 88
REWIND . . . 88
READY . . . 88
NAME . . . 88
NUNITS . . . 89
MTP Configuration . . . 89
DEBUG (NODEBUG) . . . 89
NAME . . . 89
NUNITS . . . 89
BOOT_DRIVE . . . 90
CONFIG . . . 90
IPC Configuration . . . 90
NAME . . . 90
NUNITS . . . 90
MSP Configuration . . . 91
NAME . . . 91
NUNITS . . . 91
Disk Configuration . . . 91
DEBUG (NODEBUG) . . . 91
TYPE . . . 91
READY . . . 91
NAME . . . 91
NUNITS . . . 92
RDR Configuration . . . 92
Submitting punched card decks to the card reader . . . 92
Punched card formats . . . 92
Punched card “`card`” format . . . 92
Punched card “`stream`” format . . . 93
Punched card “`7punch`” format . . . 93
Punched card “`xDeck`” format . . . 93

Punched card format determination . . . 93
Punched card format / simulator interaction . . . 93
Restarting the card reader device . . . 93
Working with card decks . . . 94
Using the punched card format "card" ('cdeck.*') . . . 94
Using the punched card format "stream" ('sdeck.*') . . . 94
Using the punched card format "7punch" ('7deck.*') . . . 95
Using the punched card format "xDeck" ('xdeck.*') . . . 95
DEBUG (NODEBUG) . . . 95
PATH . . . 95
NAME . . . 96
NUNITS . . . 96
PUN Configuration . . . 97
Overview . . . 97
Card format . . . 97
DEBUG (NODEBUG) . . . 98
PATH . . . 98
NAME . . . 98
NUNITS . . . 98
CONFIG . . . 99
LOGCARDS . . . 99
PRT Configuration . . . 99
DEBUG (NODEBUG) . . . 99
PATH . . . 100
MODEL . . . 100
READY . . . 100
NAME . . . 100
NUNITS . . . 100
CONFIG . . . 101
SPLIT . . . 101
FNP Configuration . . . 101
DEBUG (NODEBUG) . . . 101
NUNITS . . . 101
CONFIG . . . 102
MAILBOX . . . 102
IPC_NAME . . . 102
SERVICE . . . 102
OPC Configuration . . . 102
DEBUG (NODEBUG) . . . 102
AUTOINPUT . . . 103
NAME . . . 103
NUNITS . . . 103
CONFIG . . . 103
AUTOACCEPT . . . 103
NOEMPTY . . . 104
ATTN_FLUSH . . . 104
PORT . . . 104
ADDRESS . . . 104
PW . . . 104
URP Configuration . . . 104
DEBUG (NODEBUG) . . . 104

NAME . . . 104
NUNITS . . . 105
SHIFT . . . 105
SHOW . . . 105
BUILDINFO (B) . . . 105
CLOCKS (CL) . . . 106
CONFIGURATION (C) . . . 106
DEFAULT_BASE_SYSTEM (D) . . . 107
DEVICES (DEV) . . . 107
MODIFIERS (M) . . . 107
ON (O) . . . 107
PROM (P) . . . 107
QUEUE (Q) . . . 108
SHOW (S) . . . 108
TIME (T) . . . 108
VERSION (V) . . . 108
CPU . . . 109
CONFIG . . . 109
NUNITS . . . 110
KIPS . . . 111
STALL . . . 111
DEBUG . . . 111
DISK . . . 111
NAME . . . 111
NUNITS . . . 112
TYPE . . . 112
DEBUG . . . 112
FNP . . . 112
CONFIG . . . 112
IPC_NAME . . . 113
NAME . . . 113
NUNITS . . . 113
SERVICE . . . 113
STATUS . . . 114
DEBUG . . . 114
IOM . . . 115
CONFIG . . . 115
NUNITS . . . 115
DEBUG . . . 115
IPC . . . 116
NAME . . . 116
NUNITS . . . 116
MSP . . . 116
NAME . . . 116
NUNITS . . . 117
MTP . . . 117
BOOT_DRIVE . . . 117
NAME . . . 117
NUNITS . . . 117
DEBUG . . . 118

OPC . . . 118
CONFIG . . . 118
ADDRESS . . . 118
AUTOINPUT . . . 118
NAME . . . 119
NUNITS . . . 119
PORT . . . 119
PW . . . 119
DEBUG . . . 119
PRT . . . 120
CONFIG . . . 120
MODEL . . . 120
NAME . . . 120
NUNITS . . . 120
PATH . . . 121
DEBUG . . . 121
PUN . . . 121
CONFIG . . . 121
NAME . . . 121
NUNITS . . . 122
PATH . . . 122
DEBUG . . . 122
RDR . . . 122
NAME . . . 122
NUNITS . . . 123
PATH . . . 123
DEBUG . . . 123
SCU . . . 123
CONFIG . . . 123
NUNITS . . . 124
STATE . . . 124
DEBUG . . . 125
TAPE . . . 125
ADD_PATH . . . 125
CAPACITY . . . 125
DEFAULT_PATH . . . 125
NAME . . . 126
NUNITS . . . 126
DEBUG . . . 126
URP . . . 126
NAME . . . 126
NUNITS . . . 127
DEBUG . . . 127
SKIPBOOT . . . 127
SLOWPOLL . . . 127
STEP (S) . . . 127
Step Execution . . . 127
Switches . . . 127
−T . . . 127

Simulator Defaults **129**
Default Base System Script . . . 129

Default Base System Configuration . . . 141
Default Base System Cable Dump . . . 147

External Simulator Utilities **153**
Multics Punched Card Utility — `punutil` . . . 153
Multics Print File Utility — `prt2pdf` . . . 154
Multics Tape Conversion Utility — `tap2raw` . . . 155

Tips and Tricks **157**
Protecting DPS8M in low-memory situations . . . 157
Exempting DPS8M from the Linux OOM Killer . . . 157
Automatic Linux `oom_adj` adjustment . . . 158
Exempting DPS8M from FreeBSD OOM termination . . . 158
Automatic FreeBSD OOM protection . . . 159

Legal **161**
DPS8M Omnibus Documentation Licensing Terms . . . 161
DPS8M Omnibus Documentation-specific Content Licensing . . . 161
General Attribution License (Documentation) . . . 162
Multics License (Documentation) . . . 163
General Attribution License (Multics Rings Logo) . . . 164
BSD License (Eisvogel) . . . 165
ISC License (Pagebreak) . . . 166
SIL™ Open Font License 1.1 (Source™ Code Pro) . . . 167
SIL™ Open Font License 1.1 (Source™ Sans Pro) . . . 168
DPS8M Software Licensing Terms . . . 169
DPS8M Simulator . . . 169
ICU License (DPS8M Simulator) . . . 169
Multics License (Simulator Code Comments) . . . 170
Third-party Software . . . 170
libsir . . . 171
libuv . . . 172
UDPLib . . . 173
Open SIMH . . . 174
punutil . . . 175
tap2raw . . . 176
prt2pdf . . . 177
decNumber . . . 178
LibTELNET . . . 179
Line History . . . 180
BSD Random . . . 181
musl libc . . . 182
mcmb . . . 183
Multics Software Materials and Documentation . . . 184
Scope of Intended Application . . . 185
Disclaimer of Liability and Endorsement . . . 185

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,
  - Craig Ruff,
  - Eric Swenson <eric@swenson.org>,
  - M——— T———,
  - Björn Victor <bjorn@victor.se>,
  - Juergen Weiss <weiss@uni-mainz.de>,
  - M. Williams (OrangeSquid)
  - *and others …*

Introduction

About this manual

This manual describes the **DPS8M Simulator**.

It is important to note that this manual does **not** provide documentation on **Multics**. It *only* documents the simulator.

This manual is intended for:

- anyone who wants to better understand the configuration and usage of the **DPS8M Simulator**, *especially*,
- Multics *System* and *Site Administrators* who wish to change the default configuration.

For detailed information about **Multics** itself, please see the following resources:

Multics Wiki	https://swenson.org/multics_wiki/	*Home of the Multics operating system*
Bitsavers	https://bitsavers.org/pdf/honeywell/	*Multics documentation (in PDF format)*
Multicians	https://multicians.org/	*Multics history*

What is DPS8M?

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 9899:2011* (**C11**), and numerous hardware architectures.
  - Operating systems supported are **IBM AIX**, **IBM i** (**OS/400**), **QNX**, **FreeBSD**, **NetBSD**, **OpenBSD**, **DragonFly BSD**, **Haiku**, **Android**, **illumos** (**OpenIndiana**), **Linux**, **macOS**, **Solaris**, **SerenityOS**, and **Windows** (both **Cygwin** and native).
  - C compilers supported are **Clang**, **LLVM-MinGW**, AMD Optimizing C/C++ (**AOCC**), Arm C/C++ Compiler (**ARMClang**), GNU C (**GCC**), IBM Advance Toolchain (**AT**), IBM XL C/C++ (**XLC**), IBM Open XL C/C++ (**IBMClang**), Intel oneAPI DPC++/C++ (**ICX**), NVIDIA HPC SDK C Compiler (**NVC**), Oracle Developer Studio (**SunCC**), and МЦСТ Эльбрус C Compiler (**LCC**).
  - Hardware architectures actively supported include **Intel x86** (i686, x86_64), **ARM** (ARMv6, ARMv7, ARM64), **LoongArch** (LA64), **MIPS** (MIPS, MIPS64), **OpenRISC** (OR1200, MOR1KX), **PowerPC** (PPC, PPC64, PPC64le), **RISC-V** (RV64), **SPARC** (SPARC, UltraSPARC), **E2K** (МЦСТ Эльбрус), **KVX** (Kalray MPPA®, Coolidge™), and **IBM Z** (s390x, z13+).
- 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 (preferably with appropriate hardware atomic operations) and able to support ports of the **libuv** asynchronous I/O library and **libsir** logging 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.

Security

- Static application security testing by:
  - **PVS-Studio** - A static analysis tool for code quality, security, and safety.
  - **Coverity® Scan** - Find and fix defects in Java, C/C++, C#, JavaScript, Ruby, or Python code.
  - **Oracle Developer Studio** - Performance, security, and thread analysis tools to help write higher quality code.
  - **Cppcheck** - A static analysis tool for C/C++ code.
  - **Clang Static Analyzer** - A source code analysis tool that finds bugs in C, C++, and Objective-C programs.
  - **Flawfinder** - Examine C/C++ source code for possible security weaknesses.

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**
  - **Windows**
  - **Solaris**
  - **illumos**
  - **Haiku**
  - *and others …*

- Executables are provided for many hardware architectures, including:
  - **x86** / **ix86**
  - **x86_64** / **AMD64**
  - **ARM** (*e.g.* Raspberry Pi, Pine64, Orange Pi …)
  - **ARM64** / **AArch64** (*e.g.* Apple M, Fujitsu A64FX, Ampere AmpereOne, Arm Neoverse …)
  - **POWER** (*e.g.* IBM Power Systems, Raptor Talos™, Freescale PowerPC …)
  - **RISC-V** (*e.g.* SiFive HiFive, AndesCore™ V5 …)
  - **МЦСТ Эльбрус**
  - *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 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 of 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.

Git cloning

- Anyone may clone the repository via **HTTPS**:

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

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

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

- After cloning the repository, it can be updated by executing the following command from the cloned 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`**.

- Anyone may mirror the repository via HTTPS:

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

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

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

- After mirroring the repository, it can be updated (including removing local branches when they are removed upstream) by executing the following command from the mirrored 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

- Expert users may wish to build the simulator from source code to enable experimental features, or for reasons of performance and compatibility.
- 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 **20 MiB** of storage space to decompress and **40 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 **275 MiB** of storage space to clone and **300 MiB** to build.

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 12 or later of the **GNU C** (`gcc`) compiler be used for optimal performance.
- At the time of writing, **GCC 14.2** 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 gcc14
```

  - Using FreeBSD Ports (as *root*):

```
cd /usr/ports/lang/gcc14/ && make install clean
```

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

```
env CC="gcc14" LDFLAGS="-Wl,-rpath=/usr/local/lib/gcc14" 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** and **NetBSD**/**evbarm** are 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 12 or later of the **GNU C** (gcc) compiler be used for optimal performance.
- At the time of writing, **GCC 14.2** is available for **NetBSD 10** 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 gcc14
    ```

  - Using pkgsrc (as *root*):

    ```
    cd /usr/pkgsrc/lang/gcc14/ && make install clean
    ```

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

```
env CC="/usr/pkg/gcc14/bin/gcc" LDFLAGS="-Wl,-rpath=/usr/pkg/gcc14/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 7.5** 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 11 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 **11.2**) 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** 16.0.6 or later is recommended for linking, even when using **GCC 11.2** 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 the appropriate environment variables before invoking the '`gmake`' program (*i.e.* '`CC="/usr/local/bin/clang"`' and '`LDFLAGS="-fuse-ld=lld"`').

Additional OpenBSD Notes

- **OpenBSD**/**luna88k** is **not supported**.

DragonFly BSD

- At the time of writing, **DragonFly BSD 6.4.0** was current and used to verify the following instructions.

DragonFly BSD prerequisites

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

  ```
  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 12 or later of the **GNU C** (`gcc`) compiler be used for optimal performance.
- At the time of writing, **GCC 13.1** is available for DragonFly BSD and recommended by **The DPS8M Development Team**.
  - **GCC 13.1** may be installed using DragonFly BSD DPorts (as *root*):

```
pkg install gcc13
```

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

```
env CC="gcc13" CFLAGS="-I/usr/local/include"                      \
    LDFLAGS="-L/usr/local/lib -Wl,-rpath=/usr/local/lib/gcc13"    \
    ATOMICS="GNU"                                                 \
  gmake
```

Compiling using Clang

- **GCC** is recommended for optimal performance, but compilation using **Clang** is supported.
- At the time of writing, **Clang 17** is available for DragonFly BSD and recommended by **The DPS8M Development Team**.
- While some optional utilities *may* fail to build using **Clang** on DragonFly, the simulator (`src/dps8/dps8`) is fully tested with each DragonFly release.
  - **Clang 17** may be installed using DragonFly BSD DPorts (as *root*):

```
pkg install llvm17
```

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

```
env CC="clang17" 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, *i.e.* '**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* **December 2024** was used to verify these instructions.
- **GCC 14.2** is currently the recommended compiler for optimal performance.
  - **GCC 14.2** can be installed from the standard IPS repository via '**`pkg install developer/gcc-14`**'.
  - 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** (version 13 or later) is also supported (*but not recommended due to lack of LTO support*).
  - **Clang 19** can be installed from the standard IPS repository via '**`pkg install developer/clang-19`**'.

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 14**):

```
env CC="gcc-14" gmake
```

Compiling using Clang

- Build the simulator from the top-level source directory (using **GNU Make** and **Clang 19**):

```
env NO_LTO=1 CC="clang-19" gmake
```

AIX

- Ensure you are running a supported release of **IBM AIX®** on a supported platform.
  - **IBM AIX® 7.2** (*TL5 SP9*) and **7.3** (*TL3*) on **POWER8®**, **POWER9™**, and **Power10** systems are regularly tested by **The DPS8M Development Team**.
- The simulator can be built for **64-bit AIX®** systems using **IBM XL C/C++ for AIX** (`xlc`), **IBM Open XL C/C++ for AIX** (`ibm-clang`), mainline **Clang** (`clang`), and **GNU C** (`gcc`). When using **XL compilers**, ensure that you are using the latest available XL compiler and associated PTF updates for your **IBM AIX** OS level. Review the current fix list for XL compilers on AIX for complete details.
- **The DPS8M Development Team** recommends building with **IBM Open XL C/C++ V17.1.2** (or later) or mainline **Clang 18** (or later) for optimal performance. Building with **GNU C** is supported (*but not recommended*) when using **GCC 10** (or later).

Recommended compilers

- **IBM Open XL C/C++ for AIX V17.1.3 Fix Pack 2** (April 2025) is the recommended compiler for **Power11** systems.
  - LTO (*link-time optimization*) and native 128-bit integer operations are both fully supported.
- **IBM Open XL C/C++ for AIX V17.1.2 Fix Pack 15** (February 2025) is the *minimum* recommended version of the **Open XL C/C++ V17.1.2** compiler for **POWER8**, **POWER9**, and **Power10** systems.
  - LTO (*link-time optimization*) and native 128-bit integer operations are both fully supported.
- **Clang 20.1.2** is the *minimum* recommended version of the mainline **Clang** compiler.
  - LTO *(link-time optimization)* and native 128-bit integer operations are both fully supported.

Other supported compilers

- **IBM Open XL C/C++ for AIX V17.1.1 Fix Pack 6** (February 2024) is the *minimum* recommended version of the **Open XL C/C++ V17.1.1** compiler for **POWER8**, **POWER9**, and **Power10** systems.
  - LTO (*link-time optimization*) is supported, but 128-bit integer operations are **not supported**.
    - Use of the `NEED_128=1` option is required when building with **Open XL C/C++ V17.1.1**.
- **IBM XL C/C++ for AIX V16.1.0 Fix Pack 20** (December 2024) is the *minimum* recommended version of the **IBM XL C/C++ V16.1.0** compiler for **POWER8** and **POWER9** systems.
  - LTO (*link-time optimization*) and 128-bit integer operations are both **not supported**.
    - Use of the `NEED_128=1` and `NO_LTO=1` options are required when building with **IBM XL C/C++ V16.1.0**.
  - **IBM XL C/C++ V16.1.0** is **not** recommended for **Power10** systems.
- **GNU C 11.3.0** is the *minimum* recommended version of the **GNU C** compiler for **POWER8**, **POWER9**, and **Power10** systems.
  - LTO (*link-time optimization*) and 128-bit integer operations are both **not supported**.
    - Use of the `NEED_128=1` and `NO_LTO=1` options are required when building with **GNU C**.
  - **GCC 11** (and **GCC 12**) can be installed from the IBM AIX Toolbox for Open Source Software repository.
- Note that building fully-featured binaries for **IBM 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

Libraries and tools

- 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
```

GNU C compilers

- *Optionally* install **GCC 11.3** (or **GCC 12.3**) from the IBM AIX Toolbox for Open Source Software repository (as *root*).
  - GCC availability from the IBM AIX Toolbox for Open Source Software site varies depending on **IBM AIX** OS version.
- Install **GCC 11.3**:

```
/opt/freeware/bin/dnf install gcc gcc11
```

- Install **GCC 12.3**:

```
/opt/freeware/bin/dnf install gcc gcc12
```

Clang compilers

- *Optionally* install mainline **Clang**. At the time of writing, mainline **Clang** for **AIX** was not yet available from the IBM AIX Toolbox for Open Source Software repository, but can be obtained from:
  - LLVM Downloads (*source code*), or,
  - IBM GitHub Releases (*source code and binary releases*)
  - LLVM GitHub Releases (*source code and binary releases*)

IBM compiler support

- IBM does not provide support for any version of **GNU C** through **IBM AIX** support cases; we recommend IBM customers with support contracts use **Open XL C/C++** if possible.

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 V17.1.2** (*and later*):

```
env PATH="/opt/freeware/bin:${PATH}"                                    \
    CC="/opt/IBM/openxlC/17.1.2/bin/ibm-clang_r"                        \
    ATOMICS="AIX"                                                       \
    AWK="gawk"                                                          \
    OBJECT_MODE=64                                                      \
  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    \
                -I../libsir/include -DHAVE_POPT=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 V17.1.2 documentation** for additional information.

- Using **IBM Open XL C/C++ for AIX V17.1.1**:

```
env PATH="/opt/freeware/bin:${PATH}"                                    \
    CC="/opt/IBM/openxlC/17.1.1/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    \
                -I../libsir/include -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 V17.1.1 documentation** for additional information.

Clang

- Using mainline **Clang 18.1.8**:

```
env PATH="/opt/llvm/bin:/opt/freeware/bin:${PATH}"                      \
    CC="/opt/llvm/bin/clang"                                            \
    ATOMICS="AIX"                                                       \
    AWK="gawk"                                                          \
    OBJECT_MODE=64                                                      \
  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    \
                -I../libsir/include -DHAVE_POPT=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 **Clang 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       \
                -I../libsir/include -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`') *may* be 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 11.3**:

```
env PATH="/opt/freeware/bin:${PATH}" CC="gcc-11" \
  ATOMICS="AIX" NO_LTO=1 gmake
```

  - Refer to the **GCC 11.3 online documentation** for additional information.

- Using **GCC 12.3**:

```
env PATH="/opt/freeware/bin:${PATH}" CC="gcc-12" \
  ATOMICS="AIX" NO_LTO=1 gmake
```

  - Refer to the **GCC 12.3 online documentation** for additional information.

Haiku

- Ensure you are running a recent release of **Haiku** on a supported **64-bit** platform.
  - Use the '**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** (**hrev58181**) was used to verify the following instructions.

Haiku prerequisites

The default **Haiku** installation includes the required header files, an acceptable compiler (at the time of writing, **GCC 13.3**), 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 pkgman or 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

- Compiling with Clang on Haiku is **highly recommended**; the resulting binary runs approximately **35% faster**.
- At the time of writing, **Clang 19** is available from *HaikuPorts* (as the '**llvm19_clang**' and '**llvm19_lld**' packages), installable using pkgman or the '**HaikuDepot**' application.
- Build the simulator from the top-level source directory (using **GNU Make**):

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

Additional Haiku Notes

- **DPS8M** on **32-bit Haiku** platforms (*i.e.* **x86**, **x86_gcc2**) is **not** supported at this time.

Linux

- Most major **Linux** distributions using the **GNU C Library**, **musl-libc**, and **uClibc-ng** (with NO_LOCALE=1) 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**, **RISC-V**, **POWER**, and **МЦСТ Эльбрус** 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* **9**+), 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**), Oracle Developer Studio (**SunCC**), and МЦСТ Эльбрус C Compiler (**LCC**)..
- **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 "clone" 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-13`', '`gcc-toolset-14`', 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 22.04 LTS ships **GCC 11.3** and **GCC 12.1**, and the **Toolchain PPAs** ship **GCC 12.3** and **GCC 13.1**. Although these packages are *not formally supported* by Canonical, they are regularly and successfully used by **The DPS8M Development Team**.
  - **NOTE**: Ubuntu has shipped multiple versions of the **LLVM** and **Clang** packages that are ***broken*** in various ways (*incorrect library paths, missing libraries, broken LTO support, etc.*). If you encounter problems building from source code using **Clang** on Ubuntu, try installing different package versions, wait for an update, or use **GCC** to build instead.
- **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**.
  - Intel® has *retired* support for **ICC**.
- 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 15** *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 5.0.0, (with **AOCC**-provided **AMD LibM**):

```
export AOCCVER="5.0.0" &&                                  \
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 5.0.0, with **AMD Optimized CPU Libraries** (**AOCL**) (**AMD AOCL-LibM** and **AMD AOCL-LibMem**), version 5.0.0:

```
export AOCCVER="5.0.0" &&                                                \
export AOCCPATH="/opt/AMD/aocc-compiler-${AOCCVER}" &&                   \
export AOCLVER="5.0.0" &&                                                \
export AOCLPATH="/opt/AMD/aocl/aocl-linux-aocc-${AOCLVER}" &&            \
export LD_LIBRARY_PATH="${AOCLPATH}/aocc/lib:${LD_LIBRARY_PATH}" &&      \
source ${AOCCPATH}/setenv_AOCC.sh &&                                     \
env CC="clang" CFLAGS="-mllvm -vector-library=AMDLIBM"                   \
    LDFLAGS="-Wno-unused-command-line-argument -L${AOCLPATH}/aocc/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 Fix Pack 2** (*February 2024*) 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 Fix Pack 15** (*September 2023*) 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`') may be possible, but the resulting binaries have *not* been benchmarked or tested by **The DPS8M Development Team**.

NVIDIA HPC SDK C Compiler

- Build the simulator using **NVIDIA HPC SDK C Compiler** (**NVC**), version 25.3, for Linux/**x86_64** (with versions also available for Linux/**ARM64** and Linux/**OpenPOWER**):

```
export NVCVER="25.3" &&                                                  \
export NVPLAT="Linux_x86_64" &&                                          \
export NVCPATH="/opt/nvidia/hpc_sdk/${NVPLAT}/${NVCVER}/compilers/bin" &&  \
env CFLAGS="-noswitcherror --diag_suppress=mixed_enum_type               \
                           --diag_suppress=branch_past_initialization    \
                           --diag_suppress=set_but_not_used"             \
    CC="${NVCPATH}/nvc" NO_LTO=1                                         \
  make OPTFLAGS="-fast -O4 -Mnofprelaxed"
```

  - **DPS8M** is known to trigger bugs in some versions of the **NVIDIA HPC SDK C Compiler**, 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 24.10.1, for Linux/**ARM64**:

```
export ACFLVER="24.10.1" &&                                  \
export ACFLCMP="arm-linux-compiler-${ACFLVER}" &&            \
export ACFLTYP="Generic-AArch64_RHEL-9_aarch64-linux" &&     \
export ACFLPATH="/opt/arm/${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 24.10.1, for Linux/**ARM64**:

```
export ACFLVER="24.10.1" &&                                  \
export ACFLCMP="arm-linux-compiler-${ACFLVER}" &&            \
export ACFLTYP="Generic-AArch64_RHEL-9_aarch64-linux" &&     \
export ACFLPATH="/opt/arm/${ACFLCMP}_${ACFLTYP}" &&          \
export PATH="${ACFLPATH}:${PATH}" &&                         \
env CFLAGS="-mcpu=native -armpl"                             \
    LDFLAGS="-armpl"                                         \
    CC="armclang"                                            \
  make
```

- Build the simulator using the **Arm HPC C/C++ Compiler for Linux** (**ARMClang**) with the integrated **Arm Performance Libraries** (**ArmPL**), version 24.10.1, for Linux/**ARMv8-A+SVE2** (*Scalable Vector Extensions*):

```
export ACFLVER="24.10.1" &&                                    \
export ACFLCMP="arm-linux-compiler-${ACFLVER}" &&              \
export ACFLTYP="Generic-AArch64_RHEL-9_aarch64-linux" &&       \
export ACFLPATH="/opt/arm/${ACFLCMP}_${ACFLTYP}" &&            \
export PATH="${ACFLPATH}:${PATH}" &&                           \
env CFLAGS="-march=armv8-a+sve2 -mcpu=native -armpl=sve"       \
    LDFLAGS="-armpl=sve"                                       \
    CC="armclang"                                              \
  make
```

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 V17** to cross-compile for Linux/**POWER**:
  - Build libuv:

```
env CC="/opt/at17.0/bin/powerpc64le-linux-gnu-gcc"  \
    LOCAL_CONFOPTS="--host=powerpc64le-linux-gnu"   \
    CFLAGS="-mcpu=power8 -mtune=power8"             \
  make libuvrel
```

  - Build the simulator:

```
env CC="/opt/at17.0/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**, **16**, and **17** 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 **13.2.Rel1**, to cross-compile for Linux/**ARMv7-HF** (*hardware floating point*):
  - Build libuv:

```
export AGTREL="arm-gnu-toolchain-13.2.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 **13.2.Rel1**, to cross-compile for Linux/**ARM64**:
  - Build `libuv`:

```
export AGTREL="arm-gnu-toolchain-13.2.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 **13.2.Rel1**, to cross-compile for Linux/**ARM64BE** (*big endian*):
  - Build `libuv`:

```
export AGTREL="arm-gnu-toolchain-13.2.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** 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 **14.0.0-2023.06**, to cross-compile for Linux/**ARMv7-HF** (*hardware floating point*):
  - Build `libuv`:

```
export LINREL="gcc-linaro-14.0.0-2023.06-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 **14.0.0-2023.06**, to cross-compile for Linux/**ARM64**:
  - Build `libuv`:

```
export LINREL="gcc-linaro-14.0.0-2023.06-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/user/x-tools/${CTNG}/bin/${CTNG}-gcc"            \
    LOCAL_CONFOPTS="--host=${CTNG}"                            \
  make libuvrel
```

  - Build the simulator:

```
env CC="/home/user/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/user/x-tools/${CTNG}/bin/${CTNG}-gcc"     \
    LOCAL_CONFOPTS="--host=${CTNG}"                     \
  make libuvrel
```

- Build the simulator:

```
env CC="/home/user/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/user/x-tools/${CTNG}/bin/${CTNG}-gcc"     \
    LOCAL_CONFOPTS="--host=${CTNG}"                     \
    CFLAGS="-march=armv6+fp"                            \
  make libuvrel
```

  - Build the simulator:

```
env CC="/home/user/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/user/x-tools/${CTNG}/bin/${CTNG}-gcc"     \
    LOCAL_CONFOPTS="--host=${CTNG}"                     \
    CFLAGS="-mcpu=power9"                               \
  make libuvrel
```

  - Build the simulator:

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

Additional Linux Notes

- Although normally handled automatically, when building on (*or cross-compiling to*) some 32-bit targets using a compiler lacking support for 128-bit integer types, it may be necessary to explicitly 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.
  - **Intel® C++ Compiler Classic for macOS** (`icc`) is ***no longer supported*** for building **DPS8M** (*as of* ***R3.0.2***).
    - Intel® has *retired* support for **ICC**.
  - Building the simulator on **macOS** using **GCC** is ***not recommended***.
- The following instructions were verified using **macOS 15.4.1** with **Xcode 16.4** (Apple Clang 17.0.0).

macOS prerequisites

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

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

- Installation and configuration of **Homebrew** and its prerequisites (*i.e.* Xcode CLT) is outside the scope of the **DPS8M** documentation. Refer to the **Homebrew Installation Documentation** if support is required.
- 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**):

- Build using **Xcode**:

```
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** 12:
  - Build `libuv`:

```
make distclean &&                                   \
env CFLAGS="-target arm64-apple-macos12             \
            -mmacosx-version-min=12.0"              \
    LOCAL_CONFOPTS="--host=arm64-apple-darwin"      \
  make libuvrel HOMEBREW_INC= HOMEBREW_LIB=
```

  - Build the simulator:

```
env CFLAGS="-target arm64-apple-macos12             \
            -mmacosx-version-min=12.0"              \
    LDFLAGS="-target arm64-apple-macos12            \
             -mmacosx-version-min=12.0"             \
  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 "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 **Universal CRT**, **MinGW-w64**, **Cygwin**, **Midipix**, **MSYS2**, and many 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**, 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 current **64-bit** versions of **Cygwin** are regularly tested by **The DPS8M Development Team**.
  - The **32-bit** version of **Cygwin** was **discontinued** in 2022 and is ***no longer supported*** for building **DPS8M** (*as of **R3.0.2***).

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`) 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**):

```
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 "${HOME}/libuv-build" &&                                      \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&     \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" &&                        \
  mkdir -p "build" && cd "build" &&                                \
  cmake .. -DCMAKE_SYSTEM_NAME="Windows"                           \
           -DCMAKE_SYSTEM_VERSION="10"                             \
           -DCMAKE_C_COMPILER="i686-w64-mingw32-gcc"               \
           -DCMAKE_INSTALL_PREFIX="${HOME}/libuv-win32-i686" &&    \
  cmake --build . && cmake --install . )
```

  - Build the simulator:

```
env CFLAGS="-I${HOME}/libuv-win32-i686/include -D__MINGW64__"      \
    CC="i686-w64-mingw32-gcc"                                      \
    LDFLAGS="-L${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 "${HOME}/libuv-build" &&                                      \
  wget -v "https://github.com/libuv/libuv/archive/v1.x.zip" &&     \
  unzip -xa "v1.x.zip" && cd "libuv-1.x" &&                        \
  mkdir -p "build" && cd "build" &&                                \
  cmake .. -DCMAKE_SYSTEM_NAME="Windows"                           \
           -DCMAKE_SYSTEM_VERSION="10"                             \
           -DCMAKE_C_COMPILER="x86_64-w64-mingw32-gcc"             \
           -DCMAKE_INSTALL_PREFIX="${HOME}/libuv-win32-x86_64" &&  \
  cmake --build . && cmake --install . )
```

- Build the simulator:

```
env CFLAGS="-I${HOME}/libuv-win32-x86_64/include -D__MINGW64__"     \
    CC="x86_64-w64-mingw32-gcc"                                     \
    LDFLAGS="-L${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.

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 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 **11.0** (or later) is *strongly* 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.

!

Executing System Commands

- 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 "12.8MULTICS.tap" with the tape0 unit
; in read-only mode, where tape0 corresponds to the first tape device.
ATTACH -r tape0 12.8MULTICS.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**" 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      0x4e5480   0x5aaa60   0x459660
    0   11        0     0 IPC   PSI      0x4e5540   0x5abb60   0x433f10
    0   12        0     0 MSP   PSI      0x4e5600   0x5ab2e0   0x433f10
    0   13        0     0 URP   PSI      0x4e56c0   0x4e4b40   0x460cd0
    0   14        1     0 URP   PSI      0x4e56c0   0x4e4bc8   0x460cd0
    0   15        2     0 URP   PSI      0x4e56c0   0x4e4c50   0x460cd0
    0   16        3     0 FNP   Direct   0x4e3d00   0x4e3f58   0x44c490
    0   17        0     0 FNP   Direct   0x4e3d00   0x4e3dc0   0x44c490
    0   18        1     0 FNP   Direct   0x4e3d00   0x4e3e48   0x44c490
    0   19        2     0 FNP   Direct   0x4e3d00   0x4e3ed0   0x44c490
    0   20        4     0 FNP   Direct   0x4e3d00   0x4e3fe0   0x44c490
    0   21        5     0 FNP   Direct   0x4e3d00   0x4e4068   0x44c490
    0   22        6     0 FNP   Direct   0x4e3d00   0x4e40f0   0x44c490
    0   23        7     0 FNP   Direct   0x4e3d00   0x4e4178   0x44c490
    0   30        0     0 OPC   CPI      0x4e4640   0x4e4700   0x41e010
    0   40        3     0 URP   PSI      0x4e56c0   0x4e4cd8   0x460cd0
    0   41        4     0 URP   PSI      0x4e56c0   0x4e4d60   0x460cd0
    0   42        5     0 URP   PSI      0x4e56c0   0x4e4de8   0x460cd0
    0   43        1     0 OPC   CPI      0x4e4640   0x4e4788   0x41e010
    0   45        6     0 URP   PSI      0x4e56c0   0x4e4e70   0x460cd0
    0   46        7     0 URP   PSI      0x4e56c0   0x4e4ef8   0x460cd0
    0   47        8     0 URP   PSI      0x4e56c0   0x4e4f80   0x460cd0
    0   48        9     0 URP   PSI      0x4e56c0   0x4e5008   0x460cd0
    1   10        0     1 MTP   PSI      0x4e5480   0x5aaa60   0x459660
    1   11        0     1 IPC   PSI      0x4e5540   0x5abb60   0x433f10
    1   12        0     1 MSP   PSI      0x4e5600   0x5ab2e0   0x433f10

controller <--> device
  MTP  dev_code --> TAPE   command
    0     1           1   0x459660
    0     2           2   0x459660
    0     3           3   0x459660
    0     4           4   0x459660
    0     5           5   0x459660
    0     6           6   0x459660
    0     7           7   0x459660
    0     8           8   0x459660
    0     9           9   0x459660
    0    10          10   0x459660
    0    11          11   0x459660
    0    12          12   0x459660
    0    13          13   0x459660
    0    14          14   0x459660
    0    15          15   0x459660
    0    16          16   0x459660
  IPC  dev_code --> DISK   command
    0     0           0   0x433f10
    0     1           1   0x433f10
    0     2           2   0x433f10
    0     3           3   0x433f10
```

```
    0     4           14   0x433f10
    0     5           15   0x433f10
    0     6           16   0x433f10
    0     7           17   0x433f10
    0     8           18   0x433f10
    0     9           19   0x433f10
    0    10           20   0x433f10
    0    11           21   0x433f10
    0    12           22   0x433f10
    0    13           23   0x433f10
    0    14           24   0x433f10
    0    15           25   0x433f10
  MSP  dev_code --> DISK   command
    0     1            4   0x433f10
    0     2            5   0x433f10
    0     3            6   0x433f10
    0     4            7   0x433f10
    0     5            8   0x433f10
    0     6            9   0x433f10
    0     7           10   0x433f10
    0     8           11   0x433f10
    0     9           12   0x433f10
    0    10           13   0x433f10
  URP  dev_code --> URP    command
    0     1            0   0x42a960
    1     1            0   0x429800
    2     1            0   0x45c610
    3     1            1   0x45c610
    4     1            2   0x45c610
    5     1            3   0x45c610
    6     1            1   0x42a960
    7     1            2   0x42a960
    8     1            1   0x429800
    9     1            2   0x429800
```

CABLE DUMP

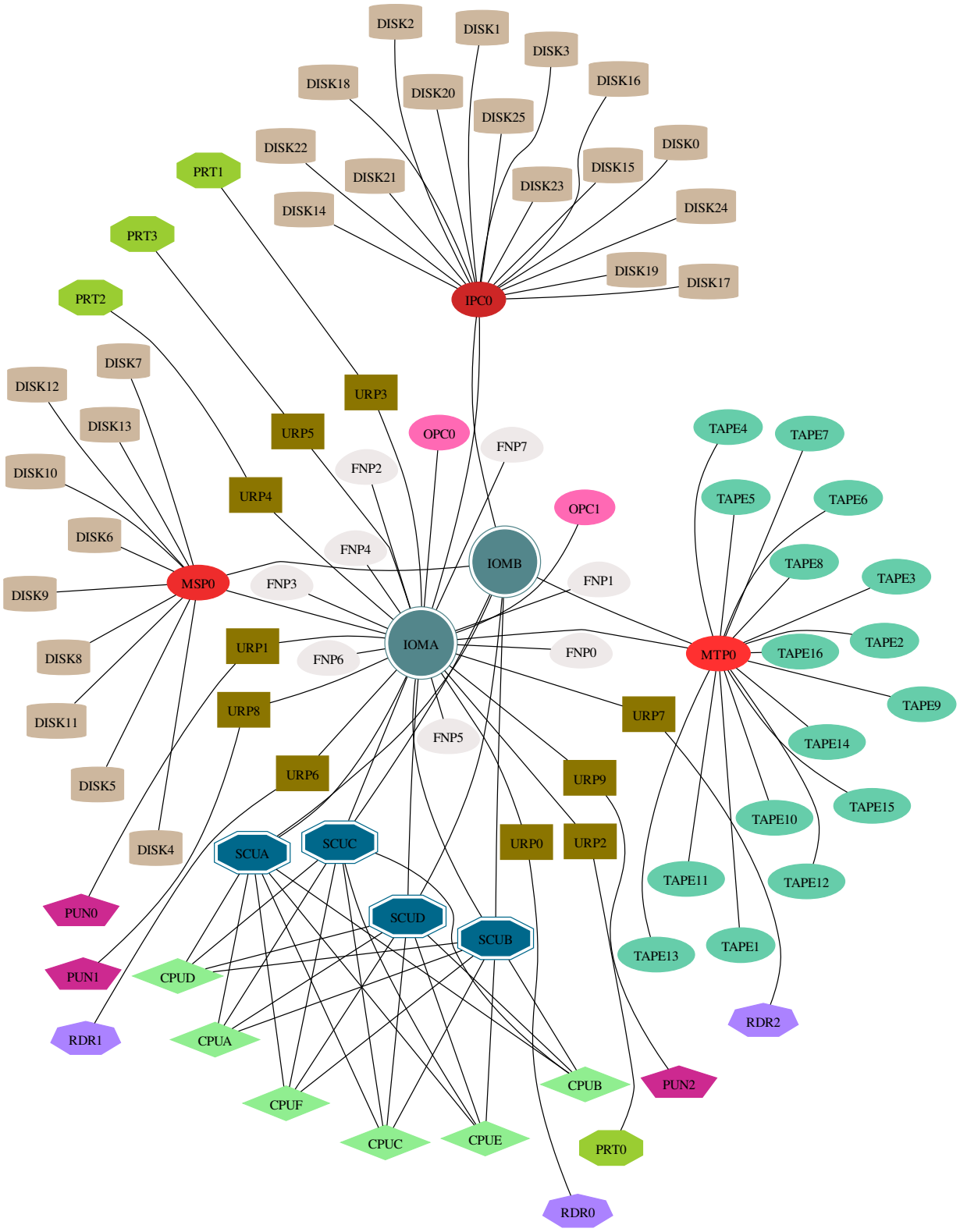
The "**CABLE DUMP**" command prints the current cabling configuration in great detail.

CABLE GRAPH

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

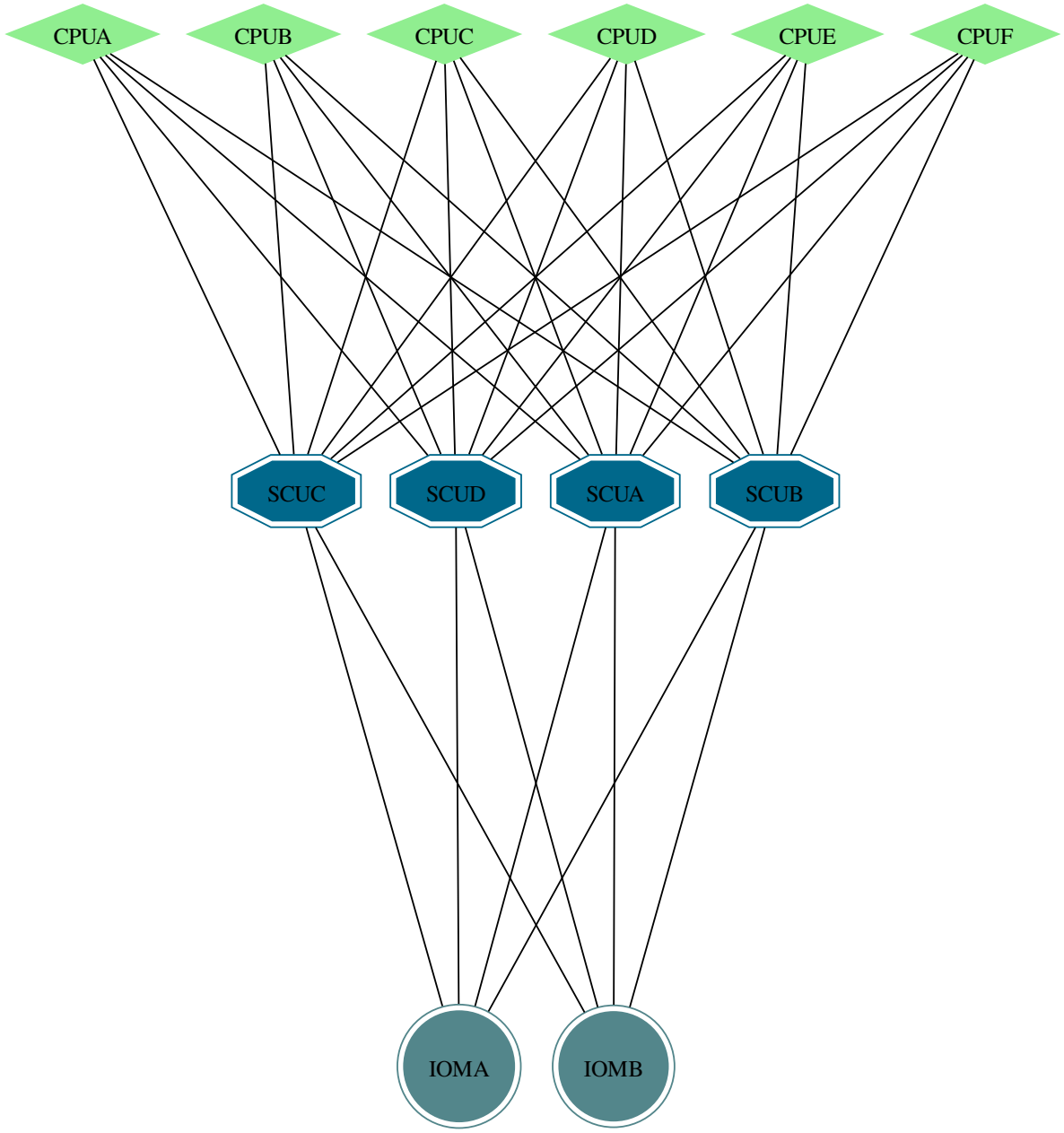
Complete Cabling Graph

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



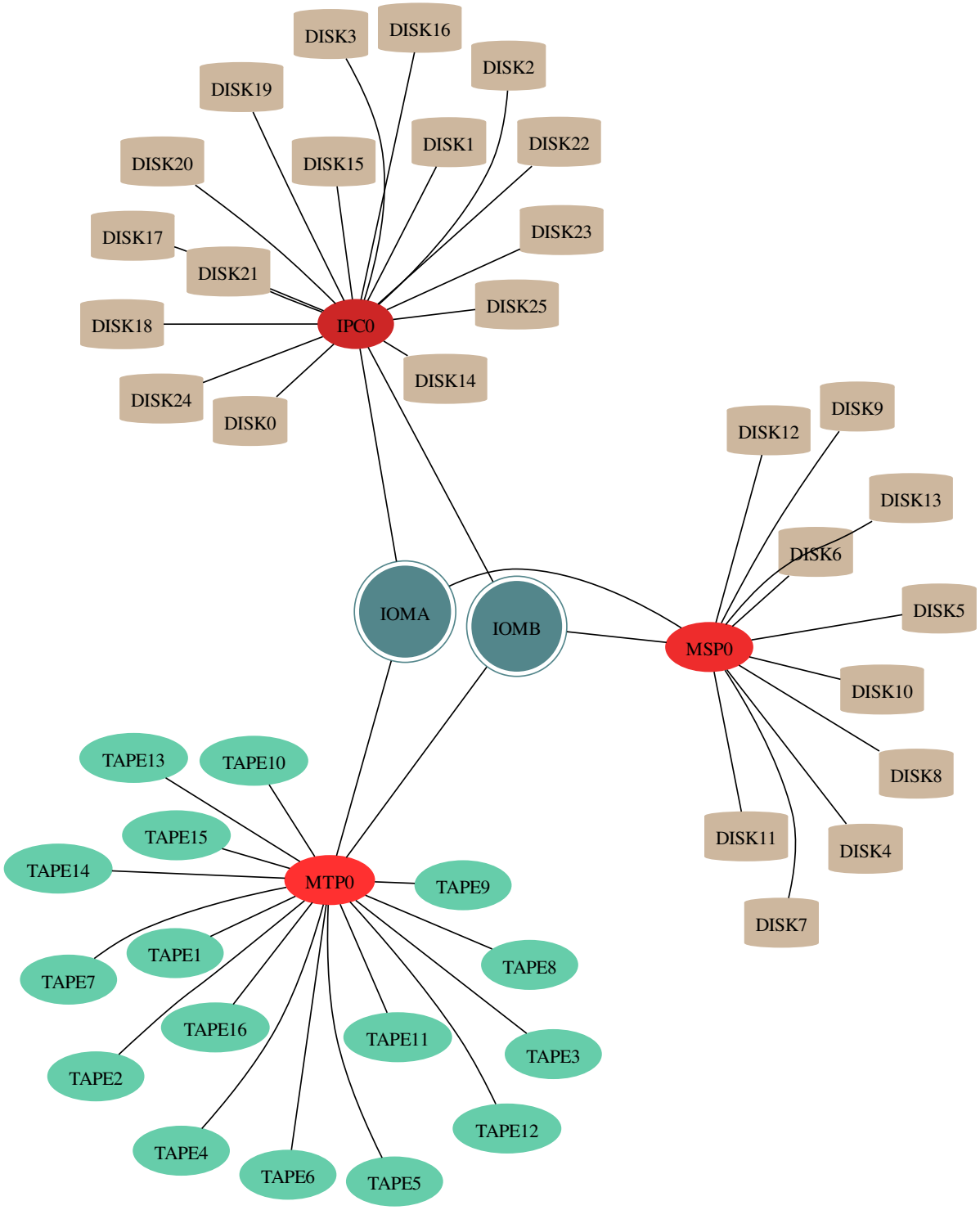
CPU / SCU / IOM Cabling Graph

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



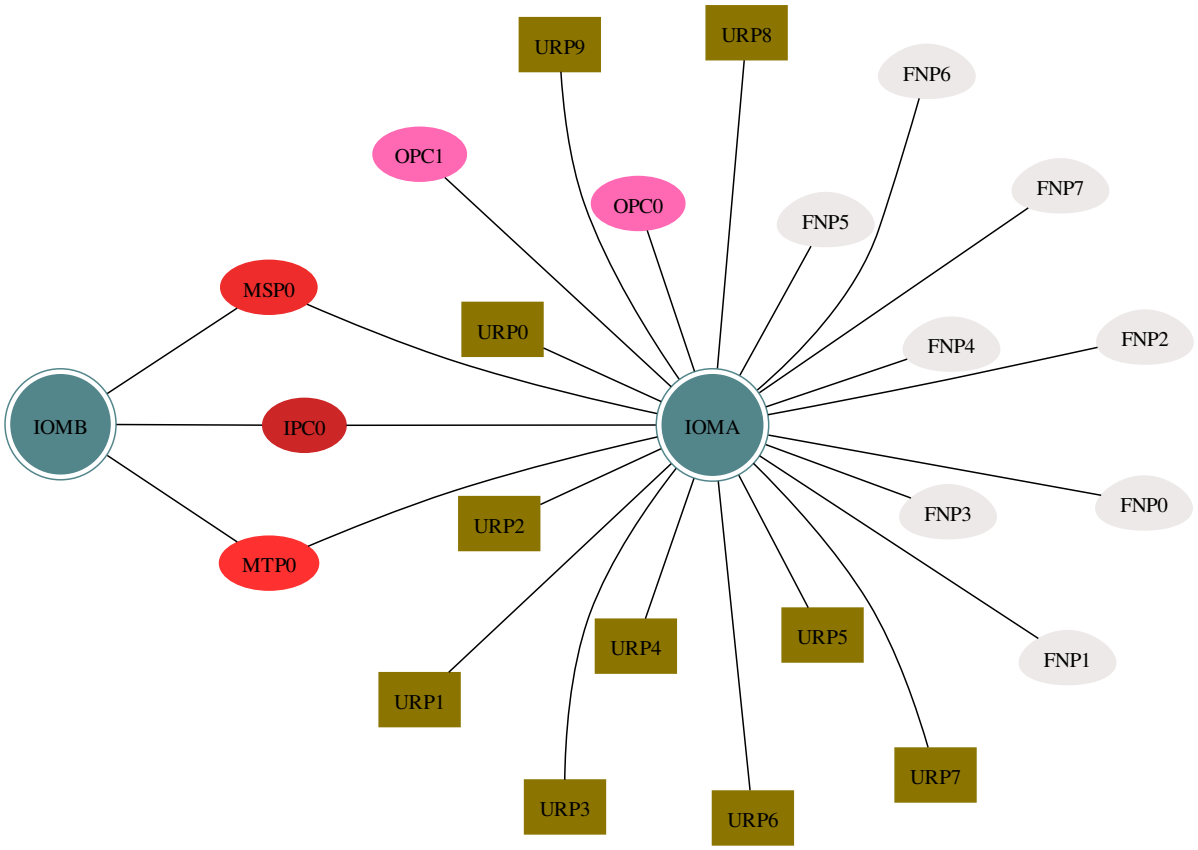
Storage Cabling Graph

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 explicit 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

Name	Meaning	Name	Meaning
EXIT	Goodbye	TTIERR	Console input I/O error
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
```

Command Output Display

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

Local Operator Console

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

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 CPU1 L68=disable
      SET CPU1 DPS8M=enable
  ```

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 of the original hardware prevent this condition from occurring when the hardware is properly functioning. Unfortunately, without introducing strict hard-realtime requirements for simulated operations such as requiring the simulator execute under the control of a hard-realtime RTOS, 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 runout (*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
```

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

"**USEMAP**" 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 IOM**0** 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=OS=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)

CLOCK_DELTA

"**CLOCK_DELTA**" modifies the SCU clock to run a user-specified number of seconds ahead (or behind) the host clock, where "0", the default, is no modification.

```
    CLOCK_DELTA=<-2147483648 to 2147483647>
```

- Setting "CLOCK_DELTA" to a negative value is useful to enable booting older Multics releases (*i.e.* MR12.3 and MR12.5) that are *not* Y2K-compliant without modification.

- Setting "CLOCK_DELTA" to a positive value is useful for Multics development or working with disks that were used "in the future".
- The *GNU* "date" tool (with the help of the Unix shell) can be used to calculate an appropriate value. The following example calculates a value representing a modification of *5 years*:

```
printf '%s\n' "$(( $(date --date='5 years' +%s) - $(date +%s) ))"
```

  - macOS and BSD users usually have this tool available as "gdate" (part of the *GNU* *coreutils* package, which may require separate installation).

- It is the responsibility of the user to set a reasonable "CLOCK_DELTA" value.
  - Multics may not handle times in the year 2043 and beyond.
  - Undefined behavior may result if the modified time is beyond *January 19 2038 03:14:07 UTC* on a host system that measures time as seconds elapsed since *January 1 1970 00:00:00 UTC* (*the Unix epoch*) and stores the result in a signed 32-bit integer (*i.e.* Unix systems using 32-bit time_t).
- It is not expected that the "CLOCK_DELTA" value will require modification by most users under normal operating circumstances.

Examples

- Configure the SCU**0** clock to run *5 years* ahead of the host clock:

```
SET SCU0 CONFIG=CLOCK_DELTA=157766400
```

- Configure the SCU**0** clock to run *33 years, 6 months* behind of the host clock:

```
SET SCU0 CONFIG=CLOCK_DELTA=-1057017600
```

TAPE Configuration

DEBUG (NODEBUG)

- TBD

DEFAULT_PATH

"DEFAULT_PATH" configures the default path to use when searching for tape files.

```
DEFAULT_PATH=<path>
```

- Setting this option will clear any previous "ADD_PATH" parameters.
- If "DEFAULT_PATH" is *not* set, the simulator will use the current working directory of the host operating system.
- When using relative paths (beginning with "**.**" or "**..**") the starting point is the current working directory of the host operating system.

Example

- Set the default path to search for tape files to "`./tapes`":

```
    SET TAPE DEFAULT_PATH=./tapes
```

ADD_PATH

"`ADD_PATH`" adds a new tape search path at the end of the current search path list.

```
    ADD_PATH=<prefix=directory>
```

- When using relative paths (beginning with "`.`" or "`..`"), the starting point is the current working directory of the host operating system.
- If "`SET DEFAULT_PATH`" is used, all previous "`ADD_PATH`" entries will be removed.
- New tape search paths cannot be added unless a "`DEFAULT_PATH`" has been explicitly set.

Example

- Look for tapes with a volume name starting with "`BK`" in the directory "`./tapes/backups`":

```
    SET TAPE ADD_PATH=BK=./tapes/backups
```

Notes

- When the tape search path table is evaluated during a tape file lookup, it is processed in order, with the first matching entry being accepted.
- For example, if the following commands are given, a tape with a volume name of "`BK001`" will be placed in the "`./tapes/billing`" directory and not the "`./tapes/backups`" directory, since it begins with the substring "`B`", matching the first entry in the search table:

```
    SET TAPE DEFAULT_PATH=./tapes/general
    SET TAPE ADD_PATH=B=./tapes/billing
    SET TAPE ADD_PATH=BK=./tapes/backups
```

- To get the intended effect, the commands may be reordered to ensure names starting with "`BK`" are checked before names starting with "`B`":

```
    SET TAPE DEFAULT_PATH=./tapes/general
    SET TAPE ADD_PATH=BK=./tapes/backups
    SET TAPE ADD_PATH=B=./tapes/billing
```

- The simulator "`ATTACH`" command used in script files (usually for mounting the boot tape) is *not* aware of the alternative tape search paths. If a directory is needed for an "`ATTACH`" command, the full path should be specified, for example:

```
    ATTACH -r tape0 ./tapes/12.8/12.8MULTICS.tap
```

CAPACITY_ALL

"CAPACITY_ALL" configures the default maximum size (in megabytes) that can be written to a tape file for all tape devices.

```
    CAPACITY_ALL=<n>
```

- If "CAPACITY_ALL" is *not* set, the simulator will default to using a maximum size of **40MB**.

Example

- Set the default maximum size of tape files for all tape devices to **40MB**:

```
    SET TAPE CAPACITY_ALL=40
```

CAPACITY

"CAPACITY" configures the maximum size (in megabytes) that can be written to a tape file for a specific tape devices.

```
    CAPACITY=<n>
```

- If "CAPACITY" is *not* set, the simulator will default to using a maximum size of **40MB**.

Example

- Set the maximum size of tape files for the "TAPE**1**" device to **40MB**:

```
    SET TAPE1 CAPACITY_ALL=40
```

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

The simulated card reader devices (**RDR**) operate by reading simulated punched card deck files from a queue directory.

The following sections document the commands for configuring the simulated card reader devices and provide an overview of their usage. The simulator includes a utility, "**punutil**", used to manipulate punched card deck files, which is referred to in the following documentation.

Submitting punched card decks to the card reader

Decks are submitted by copying the deck files into the queue directory.

The location of the queue directory on the host system is controlled with the "SET RDR PATH" command (and can be viewed with the "SHOW RDR PATH" command).

Punched card formats

- There are three card formats recognized by the simulator:
  1. 'card'
  2. 'stream'
  3. '7punch'
  4. 'xDeck'

Punched card "card" format

- In 'card' format, each line of text is treated as an 80-column card image.
- Newlines in the file are treated as card separators.
- Lines longer than 80 characters are silently extended to the next card image.

Punched card "stream" format

- In 'stream' format, each byte of input is read and translated to *MCC* punch codes, grouped into collections of 80 columns.
- No newline processing is done by the simulator.

Punched card "7punch" format

- In '7punch' format, after the '++INPUT' card, all input is treated as a *bit stream* intended to pass binary files (*in '7punch' format*) to '++DATA' decks for copying the file to a Multics segment.

Punched card "xDeck" format

- Newline (<NL>) delimited text.

Punched card format determination

- The deck format is determined by the prefix of the filename being submitted:

  1. 'card' format: "cdeck.*"
  2. 'stream' format: "sdeck.*"
  3. '7punch' format: "7deck.*"
  4. 'xDeck' format: "xdeck.*"

Punched card format / simulator interaction

Any file in the card reader queue directory that does *not* start with one of these prefixes will be ignored (other than a file named "discard", see "*Restarting the card reader device*" for more details).

Note that the simulator "peeks" at the card data, because it needs to know when to start using a specific encoding. The simulator always starts by translating the cards into *MCC* punch codes for the job control cards (*i.e.* those that start with a "++"). When it detects an "++input" card, it will switch to processing further input based on the deck format as specified by the filename prefixes given above.

Restarting the card reader device

If the card reader device gets jammed due to an error in a deck, use the following procedure to clear and reset it:

1. On the host system, create an empty file in the card reader queue directory named "discard".
   - On Unix systems, this can be accomplished with "touch discard".
2. On the Multics console, issue the following commands:

```
r rdra reinit
r rdra read_cards
```

   - You should replace "rdra" above with the appropriate card reader device name if you have more than one card reader device configured.

This will close out the card reader file in the simulator and restart the card reader in Multics.

Working with card decks

When working with card decks, it is important to understand how Multics is going to process them. Refer to *AG91-04A* (*Multics Programmer's Reference Manual*), Pages 5-50 through 5-60 ("*Bulk Input and Output*"), and Appendix C ("*Punched-Card Input Output and Returned Output Control*") for complete details. Some simple examples follow, but *AG91-04A* is the definitive reference.

Note that other than the punched card deck format prefixes above, the rest of this information does not involve the simulator, only Multics. It is also important to note that, when dealing with punched card decks, you are using a simulated media that is limited to a maximum of 80 characters per line. Trying to use lines longer than 80 characters will cause unexpected behavior and will likely not result in what you expect.

Using the punched card format "card" ('cdeck.*')

The '++FORMAT' card should read '++format mcc addnl trim'. This tells the card reader processor to trim trailing blanks and add newlines to each card image.

The following example loads a data file into Multics:

```
    ++data bootload_1.alm \Anthony \Sys\Eng
    ++password XXX
    ++format mcc addnl trim
    ++control overwrite
    ++input
```

- Since card input gets translated into lower case, there needs to be a method of designating upper case letters (because Multics is case-sensitive, and Multics *Person IDs* and *Project IDs* typically use mixed-case names). Therefore, prefixing a letter with a backslash ('\') will cause that character to be translated as upper case.
- The '++format' card above will tell Multics to trim trailing blanks and add newlines to each card image. The "mcc" tells it that the cards are punched with *MCC* codes (this is done by the simulator when using 'cdeck' or 'sdeck' formats). Even though it is optional, it is best practice to *always* include a '++format' card so you know how Multics will process your deck.

The following example executes a Multics absentee job:

```
    ++rje test.absin \Anthony \Sys\Eng
    ++password XXX
    ++format mcc addnl trim
    ++input
    ccd bootload_1.alm
    alm -list bootload_1
```

- This deck will log in as Anthony.SysEng and execute the two commands after the '++input' card. When completed, a print job containing the results of executing these commands will be queued.

Using the punched card format "stream" ('sdeck.*')

This example is the same as the first 'cdeck' example above, except formatted to use 'stream' format:

```
    ++data bootload_1.alm \Anthony \Sys\Eng
    ++password XXX
    ++format mcc noaddnl notrim
    ++control overwrite
```

```
    ++input
```

- The '++format' card specifies *not* to add newlines for each card and *not* to trim trailing spaces.
- The '++control' card specifies "overwrite", which will truncate an existing file with the same name if it already exists.
- For this example, compared to the equivalent example in 'cdeck' format above, there is no significant advantage to choose the 'sdeck' format over 'cdeck'. Just pick one and use the correct ++format card.

Using the punched card format "7punch" ('7deck.*')

The '7punch' format is used when you need to transfer a binary file via punched card. The simulator will just read raw bytes after the '++input' card and transfer them to Multics. Using Multics, you can create a '7punch' output deck on the card punch device and use the "**punutil**" utility to extract just the binary cards from the deck. The resulting file can be loaded by prefixing it with the following:

```
    ++data bound_segment_ \Anthony \Sys\Eng
    ++password XXX
    ++format viipunch noaddnl notrim
    ++control overwrite
    ++input
```

- The '++format' card ***must*** have "viipunch" and specify *not* to add newlines and *not* to trim.

Using the punched card format "xDeck" ('xdeck.*')

The 'xDeck' format is used when you need to transfer <NL> delimited records, converting to Hollerith encoding. Does not add any additional card images. This is useful for submitting decks to the Multics **GCOS Daemon**. See the **GCOS** page on the Multics Wiki (https://multics-wiki.swenson.org/index.php/GCOS) for additional details and usage examples.

DEBUG (NODEBUG)

- TBD

PATH

"**PATH**" configures the path that will be used as the queue directory for reading simulated punched card deck files (for all 'RDR' devices).

```
    PATH=<path>
```

Example

- Set the punched card reader queue directory to "/home/tom/card_decks":

```
    SET RDR PATH=/home/tom/card_decks
```

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

Overview

The following commands configure simulated card punch devices.

The simulated card punch devices operate by writing simulated punched card decks to *spool files*. The location of the spool files on the host system is controlled with the "`SET PUN PATH`" command.

The *spool file* will contain all '***banner***', '***lace***', and '***trailer***' cards that Multics punches. The '***lace***' cards are parsed to extract job information that is used to name the spool files on the host system.

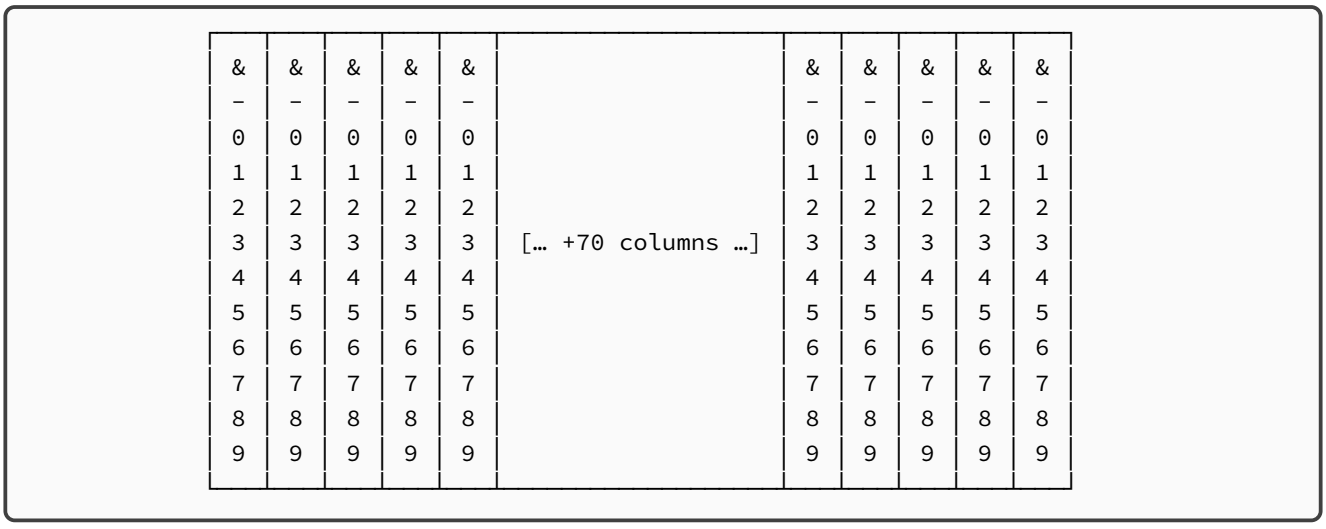
- The simulator includes a utility, "**`punutil`**", that can be used to extract the various cards from the deck in easily usable formats (such as `ASCII` text).

Card format

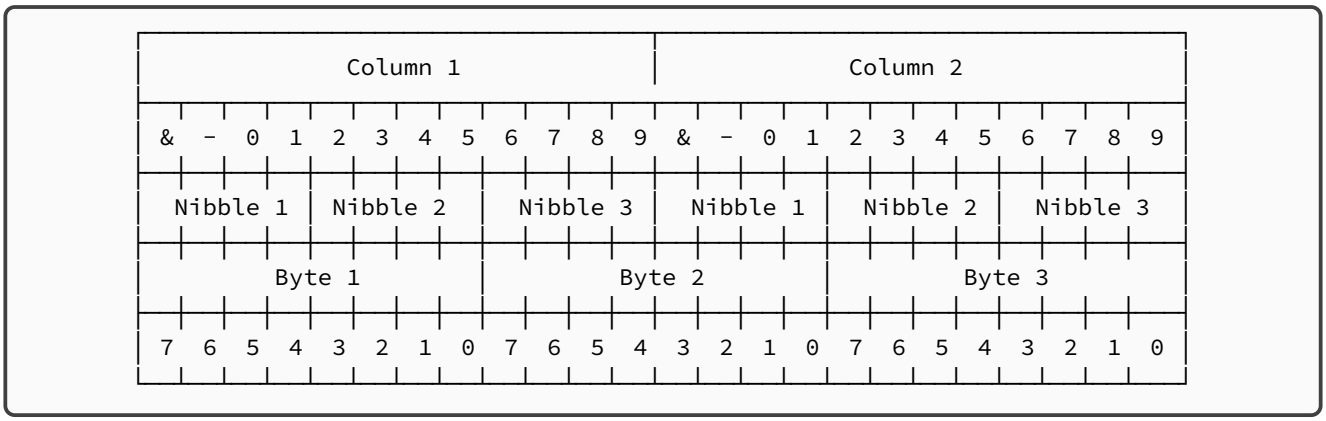
Each column in a punched card has **twelve** rows, designated from top to bottom:

'&'	'–'	'0'	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'	'9'

Given the above, an **`80`-column** punched card would then look like:



The most obvious way to represent a punched card column in a *spool file* is to assign a single bit per punch. This means that there are **12 bits** for each column. When looking at a hexadecimal dump of a *spool file*, each column occupies **three 4-bit** nibbles. The following representation is *big-endian*:



DEBUG (NODEBUG)

- TBD

PATH

"PATH" configures the path that will be used as the location to write punch *spool files*.

```
    PATH=<path>
```

Example

- Set the punch spool file path to "/home/tom/punches":

```
    SET PUN PATH=/home/tom/punches
```

Notes

- When unset, all punch spool files will be written to the current working directory of the host operating system.
- When set, the simulator expects a subdirectory to exist for each named punch device — the simulator will **not** automatically create these subdirectories. For example, if "PATH" is set to "/home/tom/punches" and three punch devices are configured (puna, punb, punc), then the following directory structure must exist for punch jobs to be output properly:

```
/home
└── tom
    └── punches
        ├── puna
        ├── punb
        └── punc
```

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

"LOGCARDS" configures the simulator to output a significant amount of diagnostic details regarding a punch device to the simulator console. This includes an "ASCII art" representation of the punched cards, where asterisks (*) are used to represent the punched holes. It is recommended to enable this option only if you are attempting to diagnose a card punch issue.

```
    SET PUNn CONFIG=LOGCARDS=<0 or 1>
    SET PUNn CONFIG=LOGCARDS=<disable or enable>
```

Example

- Turn on logging of card punch diagnostic data for punch device "PUN**0**":

```
    SET PUN0 CONFIG=LOGCARDS=1
```

- Turn off logging of card punch diagnostic data for punch device "PUN**1**":

```
    SET PUN1 CONFIG=LOGCARDS=disable
```

PRT Configuration

DEBUG (NODEBUG)

- TBD

PATH

"**PATH**" configures the output path for printer devices.

```
    PATH=<path>
```

- If "PATH" is *not* set, the simulator will use the current working directory of the host operating system for printer output.
- When using relative paths (beginning with "`.`" or "`..`") the starting point is the current working directory of the host operating system.

Example

- Set the output path for printer devices to "/home/tom/printers":

```
    SET PRT PATH=/home/tom/printers
```

NOTE: Normally, no matter how many printers are configured, all print jobs will be output to the same directory (with guaranteed unique names). If it is desirable to have each printer output to a separate directory, refer to the documentation for the "CONFIG=SPLIT" option.

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

"**SPLIT**" configures the specified "PRT" unit to output jobs into a subdirectory of the directory specified by the "PATH" setting. Print devices that do not enable "SPLIT" will output their files into the directory specified by "PATH".

```
    SPLIT=<0 or 1>
    SPLIT=<off or on>
```

Example

- Enable output splitting for the "PRT**1**" device:

```
    SET PRT1 CONFIG=SPLIT=on
```

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} H{INTS}                   Show configuration hints
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 statically and dynamically 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 -Wall -O3 -g -U_FORTIFY_SOURCE -fno-stack-protector
      -fno-math-errno -fno-trapping-math -fno-signed-zeros
      -fomit-frame-pointer -ffp-contract=fast -fno-strict-aliasing
      -D_LARGE_FILES -D_FILE_OFFSET_BITS=64 -DUSE_FLOCK=1 -DUSE_FCNTL=1
      -I../libsir/include -std=c11 -U__STRICT_ANSI__ -D_GNU_SOURCE
      -flto=auto -pthread -DLOCKLESS -DVER_CURRENT_TIME=2025-05-15 01:51
      UTC -DBUILD_PROM_OSV_TEXT=Linux -DBUILD_PROM_OSA_TEXT=x86_64 -lpthread
      -luv -lrt -ldl

 Loaded shared objects: linux-vdso.so.1 /usr/libexec/coreutils/libstdbuf.so
 /lib64/libm.so.6 /lib64/libuv.so.1 /lib64/libc.so.6
 /lib64/ld-linux-x86-64.so.2

    Event loop library: Built with libuv 1.51.0 (release); 1.51.0 in use
   Log support library: Built with libsir 2.2.6-dev; 2.2.6-dev in use
          Math library: decNumber 3.68-20210520p4
     Atomic operations: C11 and 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:       01:52:43.748
```

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
 PROM size: 1024 bytes
 PROM initialization data:
     Field Description      Length   Offset              Contents
 ========================= ======== ======== ==================================
 CPU Model                    11       0(8)     'DPS 8/SIM M'
 CPU Serial                   11      13(8)     '0          '
 Ship Date                     6      26(8)     '250514'
 PROM Layout Version           1      74(8)     '2'
 Release Git Commit Date      10     106(8)     '2025-05-14'
 Release Major                 3     120(8)     '003'
 Release Minor                 3     123(8)     '001'
 Release Patch                 3     126(8)     '000'
 Release Iteration             3     131(8)     '000'
 Release Build Number          8     134(8)     '        '
 Release Type                  1     144(8)     'X'
 Release Version Text         29     145(8)     '3.1.0                        '
 Build Architecture           20     202(8)     'x86_64              '
 Build Operating System       20     226(8)     'Linux               '
 Target Architecture          20     252(8)     'Intel x86_64 (AMD64)'
 Target Operating System      20     276(8)     'GNU/Linux           '
```

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: X3.1.0+1 (64-bit)
    Commit: f0d311a99c630e4d7d067d16000b62ddca81fefc
  Modified: 2025-05-14 21:17 UTC - Kit Prepared: 2025-05-15 01:51 UTC
  Compiled: 2025-05-15 01:51 UTC

 ****** THIS BUILD IS NOT SUPPORTED BY THE DPS8M DEVELOPMENT TEAM ******

 Build Information:
    Target: GNU/Linux on Intel x86_64 (AMD64)
  Build OS: Linux x86_64
  Compiler: GCC 15.1.1 20250425 (Red Hat 15.1.1-1) x86_64
  Built by: root@runner-punwssdb-project-10664906-concurrent-0

 Host System Information:
   Host OS: Linux 6.14.0-0.rc7.56.fc42.x86_64 x86_64

 This software is made available under the terms of the ICU License.
 For complete license details, see the LICENSE file included with the
 software 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:                024000717200(8)
                              00001010000000000011100111010000000(2)
Address switches:             000000(8)
                              000000000000000000(2)
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)
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").

- The default path is the current working directory of the host operating system.

Example

```
sim> SHOW PRT PATH
Path to PRT files is /builds/dps8m/dps8m/docs
```

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.

For details on usage and configuration of "PUN" devices, see the "SET PUN" section of this documentation.

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 unset.
```

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.

For details on usage and configuration of "RDR" devices, see the "SET RDR" section of this documentation.

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 unset.
```

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
Clock delta: 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)

Step Execution

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 X3.1.0+1
; DEFAULT_BASE_SYSTEM_SCRIPT (658 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=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=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=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=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=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=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 SCU0 CONFIG=CLOCK_DELTA=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 SCU1 CONFIG=CLOCK_DELTA=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 SCU2 CONFIG=CLOCK_DELTA=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 SCU3 CONFIG=CLOCK_DELTA=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      0x4e5480   0x5aaa60   0x459660
    0   11        0     0 IPC     PSI      0x4e5540   0x5abb60   0x433f10
    0   12        0     0 MSP     PSI      0x4e5600   0x5ab2e0   0x433f10
    0   13        0     0 URP     PSI      0x4e56c0   0x4e4b40   0x460cd0
    0   14        1     0 URP     PSI      0x4e56c0   0x4e4bc8   0x460cd0
    0   15        2     0 URP     PSI      0x4e56c0   0x4e4c50   0x460cd0
    0   16        3     0 FNP     Direct   0x4e3d00   0x4e3f58   0x44c490
    0   17        0     0 FNP     Direct   0x4e3d00   0x4e3dc0   0x44c490
    0   18        1     0 FNP     Direct   0x4e3d00   0x4e3e48   0x44c490
    0   19        2     0 FNP     Direct   0x4e3d00   0x4e3ed0   0x44c490
    0   20        4     0 FNP     Direct   0x4e3d00   0x4e3fe0   0x44c490
    0   21        5     0 FNP     Direct   0x4e3d00   0x4e4068   0x44c490
    0   22        6     0 FNP     Direct   0x4e3d00   0x4e40f0   0x44c490
    0   23        7     0 FNP     Direct   0x4e3d00   0x4e4178   0x44c490
    0   30        0     0 OPC     CPI      0x4e4640   0x4e4700   0x41e010
    0   40        3     0 URP     PSI      0x4e56c0   0x4e4cd8   0x460cd0
    0   41        4     0 URP     PSI      0x4e56c0   0x4e4d60   0x460cd0
    0   42        5     0 URP     PSI      0x4e56c0   0x4e4de8   0x460cd0
    0   43        1     0 OPC     CPI      0x4e4640   0x4e4788   0x41e010
    0   45        6     0 URP     PSI      0x4e56c0   0x4e4e70   0x460cd0
    0   46        7     0 URP     PSI      0x4e56c0   0x4e4ef8   0x460cd0
    0   47        8     0 URP     PSI      0x4e56c0   0x4e4f80   0x460cd0
    0   48        9     0 URP     PSI      0x4e56c0   0x4e5008   0x460cd0
    1   10        0     1 MTP     PSI      0x4e5480   0x5aaa60   0x459660
    1   11        0     1 IPC     PSI      0x4e5540   0x5abb60   0x433f10
    1   12        0     1 MSP     PSI      0x4e5600   0x5ab2e0   0x433f10
  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   0x459660
    0     2           2   0x459660
    0     3           3   0x459660
    0     4           4   0x459660
    0     5           5   0x459660
    0     6           6   0x459660
    0     7           7   0x459660
    0     8           8   0x459660
    0     9           9   0x459660
    0    10          10   0x459660
    0    11          11   0x459660
    0    12          12   0x459660
    0    13          13   0x459660
    0    14          14   0x459660
    0    15          15   0x459660
    0    16          16   0x459660
  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   0x433f10
    0     1           1   0x433f10
    0     2           2   0x433f10
    0     3           3   0x433f10
```

```
    0     4           14   0x433f10
    0     5           15   0x433f10
    0     6           16   0x433f10
    0     7           17   0x433f10
    0     8           18   0x433f10
    0     9           19   0x433f10
    0    10           20   0x433f10
    0    11           21   0x433f10
    0    12           22   0x433f10
    0    13           23   0x433f10
    0    14           24   0x433f10
    0    15           25   0x433f10
  MSP  dev_code --> DISK   command
    0     1            4   0x433f10
    0     2            5   0x433f10
    0     3            6   0x433f10
    0     4            7   0x433f10
    0     5            8   0x433f10
    0     6            9   0x433f10
    0     7           10   0x433f10
    0     8           11   0x433f10
    0     9           12   0x433f10
    0    10           13   0x433f10
  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   0x42a960
    1     1            0   0x429800
    2     1            0   0x45c610
    3     1            1   0x45c610
```

```
    4     1           2   0x45c610
    5     1           3   0x45c610
    6     1           1   0x42a960
    7     1           2   0x42a960
    8     1           1   0x429800
    9     1           2   0x429800
  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
```

External Simulator Utilities

The following sections document the various utilities included with the simulator distribution.

Multics Punched Card Utility — `punutil`

The "**`punutil`**" utility (version `0.2`) manipulates punched card deck files for use with simulated punch devices.

```
Multics Punch Utility Program

Invoking:
    punutil [options]

Where options are:
  -h, --help      = Output this message
  -a, --auto      = Attempt to automatically determine the type of card deck
                      (Default; disables any previously enabled output options)
  -n, --no-auto   = Disable auto selection of the card format
                      (You must specify output control options)
  -v, --version   = Add version information to stderr output

Output control options:
    By default 'auto' mode is active, where a scan attempts to determine the
    type of deck.  The scan order is: 'MCC', '7punch', and 'raw' (if neither
    of  the  first  two seem  to apply).  Note  that  the options  below are
    mutually exclusive.

  -7, --7punch    = Interpret the cards as 7punch data and output the data
                      (currently not supported!)
  -c, --cards     = Output an ASCII art form of the punched cards with an '*'
                      representing the punches
  -f, --flip      = If --cards is specified, the banner cards will be 'flipped'
                      so they can be read normally
  -g, --glyphs    = Output the glyphs parsed from the banner cards as ASCII
  -m, --mcc       = Interpret the cards as MCC Punch Codes
                      (Invalid punch codes will be converted to spaces)
  -r, --raw       = Dump the raw card data as 12-bit words in column order

This program will read a card spool file produced by the DPS8M Simulator,
parse it, and produce the requested output on standard output.

Note that only one output mode may be selected per execution.

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

Multics Print File Utility — `prt2pdf`

The “**prt2pdf**” utility (version `3.0.3`) converts the output of simulated line printer devices to ISO `32000` Portable Document Format (*PDF*).

```
prt2pdf: A filter to convert text files with ASA carriage control to a PDF.

   prt2pdf(1) reads input from standard input. The first character of each line
   is interpreted as a control character. Lines beginning with any character
   other than those listed in the ASA carriage-control characters table are
   interpreted as if they began with a blank, and an appropriate diagnostic
   appears on standard error. The first character of each line is not printed.

     +------------+-------------------------------------------------+
     | +          | Do not advance; overstrike previous line.       |
     | blank      | Advance one line.                               |
     | null lines | Treated as if they started with a blank         |
     | 0          | Advance two lines.                              |
     | -          | Advance three lines (IBM extension).            |
     | 1          | Advance to top of next page.                    |
     | all others | Discarded (except for extensions listed below)|
     +------------+-------------------------------------------------+

 Extensions:
    H  Advance one-half line.
    R  Do not advance; overstrike previous line. Use red text color
    G  Do not advance; overstrike previous line. Use green text color
    B  Do not advance; overstrike previous line. Use blue text color
    r  Advance one line. Use red text color
    g  Advance one line. Use green text color
    b  Advance one line. Use blue text color
    ^  Overprint but add 127 to the ADE value of the character
       (i.e. use ASCII extended character set)

PRINTABLE PAGE AREA:
  The page size may be specified using -H for height, -W for width, and -u
  to indicate the points per unit (72 makes H and W in inches,
  1 is used when units are in font points). For example:
    -u 72 -H 8.5 -W 11                # page Height and Width in inches
    -u 72 -B 0.5 -L 0.5 -R 0.5 -T 0.5 # margins (Top, Bottom, Left, Right)

    Common media sizes with -u 1:
    +-------------------+-----+------+
    | Name              |  W  |  H   |
    +-------------------+-----+------+
    | Letterdj (11x8.5) | 792 |  612 | (LandScape)
    | A4dj              | 842 |  595 |
    | Letter (8.5x11)   | 612 |  792 | (Portrait)
    | Legal             | 612 | 1008 |
    | A5                | 420 |  595 |
    | A4                | 595 |  842 |
    | A3                | 842 | 1190 |
    +-------------------+-----+------+

SHADING:
```

```
    -g 0.800781     # gray-scale value  for shaded bars ( 0 < g 1 )
                    # 0 is black, 1 is white.
    -i 2            # repeat shade pattern every N lines
    -d ' '          # dashcode pattern (seems buggy!)
    -G              # green bar

MARGIN LABELS:
   -s ''            # top middle page label.
   -t ''            # top left page label.
   -P               # add page numbers to right corners

TEXT OPTIONS:
   -l 60            # lines per page
   -f Courier       # font names: Courier, Courier-Bold,Courier-Oblique
                      Helvetica, Symbol, Times-Bold, Helvetica-Bold,
                      ZapfDingbats, Times-Italic, Helvetica-Oblique,
                      Times-BoldItalic, Helvetica-BoldOblique,
                      Times-Roman, Courier-BoldOblique
OTHER OPTIONS:
   -S 0             # right shift 1 for non-ASA files
   -N               # add line numbers

VERSION AND HELP:
   -v               # version number
   -h               # display this help

ENVIRONMENT VARIABLES:
   $IMPACT_TOP      # will be printed in large red letters across the page top
   $IMPACT_GRAY     # sets the default gray-scale value, same as the -g switch
```

Multics Tape Conversion Utility — `tap2raw`

The "**`tap2raw`**" utility (version `1.1.1`) converts `tap` files produced by the simulated tape devices to `raw` format.

```
Usage:
  tap2raw < infile.tap > outfile.raw
  tap2raw [ -v | --version ]
  tap2raw [ -h | --help ]
```

Tips and Tricks

Protecting DPS8M in low-memory situations

If the simulator is terminated unexpectedly, data loss or corruption could occur. Users may wish to exempt the **DPS8M** process from being terminated in low memory situations.

- Exempting the simulator from the operating system out-of-memory handler does *not* necessarily ensure stability in out-of-memory situations. The system could decide to kill other important system processes leaving the machine in a hung or otherwise unusable state. It is always best to ensure that enough memory is free to avoid triggering the out-of-memory condition in the first place.

Exempting DPS8M from the Linux OOM Killer

Linux users may wish to exempt the **DPS8M** process from the Linux "**OOM** (***out-of-memory***) **Killer**" to avoid unexpected terminations in low memory situations.

Before initiating a kernel panic due to total memory exhaustion, the Linux *OOM Killer* is activated to inspect all running processes and assigns a score to each one. It then terminates the process with the highest score, repeating the procedure until enough memory has been freed to avoid a kernel panic.

The actual scoring algorithm is complex and varies greatly depending on the Linux kernel version. The algorithm may be further tuned by the vendor or distributor. The score is generally based on a large number of factors including the amount of memory allocated by a process, the length of time a process has been running, the status of overall system memory overcommitment, the user running the process, any cgroups-based configuration constraints, and most importantly, an "`oom_adj`" value.

- There may be additional considerations involved if you are running the simulator in a namespace or container (*e.g.* Docker, Kubernetes, LXC, Virtuozzo/OpenVZ, etc.) or using a Linux distribution with vendor kernel modifications or additional system daemons (*e.g.* Android's '`lmkd`', systemd's '`systemd-oomd`', Facebook's '`oomd`', the '`earlyoom`' daemon, etc.) that are outside the scope of this documentation. Consult your Linux distribution documentation for further details.

Users who are running the simulator in production environments should adjust the "`oom_adj`" value so other processes are considered for termination first (or to exempt the simulator completely). The "`oom_adj`" is one of 32 values ranging from '14' through '−17', with '14' indicating the process should most likely be selected by the *OOM Killer* and '−16' indicating the process should most likely *not* be selected by the *OOM Killer*. The special value of '−17' exempts the process from consideration entirely.

> At the time of writing, this functionality is *not* integrated in the simulator itself, because allowing a non-`root` process to *securely* adjust it's own "`oom_adj`" value requires adding a compile-time dependency on the "`libcap`" headers and a run-time dependency on the "`libcap`" library.

- To adjust the "`oom_adj`" value to '−17' for all running "`dps8`" processes, use the following shell commands:

```
pgrep "dps8" | while read pid; do
  printf '%s\n' "-17" | sudo tee "/proc/${pid:?}/oom_adj"
done
```

- If running with appropriate privileges (*i.e.* from `root`'s crontab) you can use the following shell commands:

```
pgrep "dps8" | while read pid; do
  printf '%s\n' "-17" > "/proc/${pid:?}/oom_adj"
done
```

Automatic Linux `oom_adj` adjustment

If you want to automate the process of adjusting the "`oom_adj`" value, or completely exempt the simulator from consideration by the Linux *OOM Killer* each time the simulator starts, you can do so using "`sudo`" and a "`bash`" script.

1. First, create the file "`/usr/local/sbin/dps8-oom`" containing the "`dps8-oom`" script. This script is available from "`https://gitlab.com/dps8m/misc-scripts/-/raw/master/dps8-oom`":
   - Download using '`wget`':

```
wget -O - https://gitlab.com/dps8m/misc-scripts/-/raw/master/dps8-oom |\
  sudo tee /usr/local/sbin/dps8-oom > /dev/null
```

   - Download using '`curl`':

```
curl -fSsL https://gitlab.com/dps8m/misc-scripts/-/raw/master/dps8-oom |\
  sudo tee /usr/local/sbin/dps8-oom > /dev/null
```

2. Ensure the script has secure ownership and proper permissions:

```
sudo chown root:root /usr/local/sbin/dps8-oom
sudo chmod 755 /usr/local/sbin/dps8-oom
```

3. Next, use the "`sudo visudo`" command to edit the "`sudoers`" file and add the following line:

```
username ALL=(ALL) NOPASSWD: /usr/local/sbin/dps8-oom
```

   - You should replace "`username`" in the text above with the actual name of the user that runs the simulator. If you want to allow all users of a particular group to run the script, prefix the group name with the "`%`" symbol (*i.e.* "`%wheel`").

4. Finally, at the top of your simulator script (`INI` file), add the following line:

```
! sudo /usr/local/sbin/dps8-oom
```

- You will see a message similar to the following when the "`dps8-oom`" script is called from the simulator:

```
PID 375995 oom_adj value adjusted successfully.
```

Exempting DPS8M from FreeBSD OOM termination

The **FreeBSD** operating system will automatically terminate processes that are not explicitly protected if all memory and swap space is exhausted. FreeBSD users may wish to protect the **DPS8M** process to avoid unexpected termination of the simulator in low memory situations.

The system decides which processes to terminate based on a large number of factors and heuristics which vary from release to release, but include process priority, memory usage, resource consumption, interactive vs. non-interactive status, and most importantly, the setting of the process '`protect`' flag.

Users running the simulator on FreeBSD in production environments should set the '`protect`' flag for the simulator process using the "`protect(1)`" utility.

- To set the 'protect' flag for all running "dps8" processes, use the following shell command (*as root*):

```
pgrep dps8 | xargs protect -p
```

  - For more information on the "protect(1)" utility, see the FreeBSD man pages (*i.e.* 'man protect').

Automatic FreeBSD OOM protection

If you want to automate the process of setting the 'protect' flag each time the simulator starts, you can do so using "sudo" and a shell script. These instructions assume your user is member of the "wheel" group.

1. First, install and configure the "sudo" package (*as root*):

```
pkg install sudo
```

   - Run "visudo -f /usr/local/etc/sudoers.d/0000-wheel" (*as root*) and add the following line to enable "sudo" for the "wheel" group:

```
%wheel ALL=(ALL) ALL
```

2. Create the file "/usr/local/sbin/dps8-oom" containing the "dps8-oom-fbsd" script. This script is available from "https://gitlab.com/dps8m/misc-scripts/-/raw/master/dps8-oom-fbsd":

```
sudo fetch https://gitlab.com/dps8m/misc-scripts/-/raw/master/dps8-oom-fbsd \
  -o /usr/local/sbin/dps8-oom
```

3. Ensure the script has secure ownership and proper permissions:

```
sudo chown root:wheel /usr/local/sbin/dps8-oom
sudo chmod 755 /usr/local/sbin/dps8-oom
```

4. Run "sudo visudo -f /usr/local/etc/sudoers.d/9999-dps8-oom" and add the following line:

```
username ALL=(ALL) NOPASSWD: /usr/local/sbin/dps8-oom
```

   - You should replace "username" in the text above with the actual name of the user that runs the simulator. If you want to allow all users of a particular group to run the script, prefix the group name with the "%" symbol (*i.e.* "%wheel").

5. Finally, at the top of your simulator script (INI file), add the following line:

```
! sudo /usr/local/sbin/dps8-oom
```

- You will see a message similar to the following when the "dps8-oom" script is called from the simulator:

```
PID 55290 protected successfully.
```

NOTE: If "sudo" asks for a password to run the script (executed via the full absolute path) even after the above configuration you should examine the output of "sudo -l" to review your rule ordering, and ensure that the entry for "dps8-oom" appears *after* any generic "(ALL)ALL" or similar entries. See the "sudo" documentation for further details.

Legal

DPS8M Omnibus Documentation Licensing Terms

DPS8M Omnibus Documentation-specific Content Licensing

- The *complete* **DPS8M Omnibus Documentation** ("*this documentation*") is licensed under the terms of the **General Attribution License**.
- **Third-party content** *within* this documentation is copyrighted by their respective owners and licensed as indicated in the following disclosures.

General Attribution License (Documentation)

General Attribution License

Multics License (Documentation)

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 (DPS8M Simulator)

- The **DPS8M Simulator** ("**DPS8M**") is distributed under the **ICU License**.

COPYRIGHT AND PERMISSION NOTICE

Multics License (Simulator Code Comments)

- 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

- Copyright © 1972 **The Massachusetts Institute of Technology**
- Copyright © 1972 **Honeywell Information Systems, Inc**.
- Copyright © 2006 **Bull HN Information Systems, Inc**.
- Copyright © 2006 **Bull SAS**

All rights reserved.

- 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.

libsir

- **libsir** is a cross-platform, thread-safe logging and utility library written in C (ISO/IEC 9899:2011 C11) that is designed to simplify and streamline the generation and distribution of human-readable information in software. It is developed by **Ryan M. Lederman** and **Jeffrey H. Johnson** and is distributed under the terms of the **MIT License**.

libsir License

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

UDPLib

- **UDPLib** is a library that implements the BBN ARPAnet IMP/TIP Modem/Host Interface over UDP. It is derived from code written by **Robert Armstrong**, and is distributed under the terms of the following **MIT license**.

UDPLib 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 following **MIT-style license**.

Open SIMH License

Original code published in 1993-2023, 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**.

tap2raw

- **tap2raw** is a utility to convert the `tap` output of the simulated **DPS8M** tape devices to `raw` format. It was written by **Jeffrey H. Johnson** and distributed under the terms of the **MIT No Attribution License** (**MIT-0**).

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**.

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**.

COPYRIGHT AND PERMISSION NOTICE

- Copyright © 1995-2010 **International Business Machines Corporation** and others

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.

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

- Copyright © 1983-1991 **The Regents of the University of California**
- Copyright © 2021-2025 **The DPS8M Development Team**

All rights reserved.

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.
3. Neither the name of the University 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 REGENTS 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 REGENTS 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.

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

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**.