FirmWorks

Power Firmware(tm) Complete Feature Set

FirmWorks Value-Added Characteristics

- Portable internal structure born of longevity across multiple platforms/environments
- FirmWorks Extensions
  - 64-bit Extensions
  - Additional keyboard chords for regaining control of a misconfigured system
  - Additional tools for working with configuration variables
  - Additional control over the dictionary
  - File system extensions
  - Additional methods for downloading over serial links
  - Breakpoint extensions
  - Source level debugger extensions
  - Debugging hooks and tools created out of experience
- Build environment
  - All files in our source tree are either pre-compiled or can be compiled within the Forth environment without the use of any third party tools. (Goodbye make and gcc.)
- Documentation
  - "Open Firmware Command Reference"
  - "Writing FCode Programs for PCI"
  - "Open Firmware Client Interface Developer's Guide"
  - "Porting Guide" (available only to source code license holders)
- Training Classes
  - Forth Language / Writing FCode Drivers
  - Writing and Debugging Client Interface Programs
  - Using Open Firmware Systems
- Porting Services
- Device Driver Development Services
- Open Firmware Contract Engineering Services
- Telephone support

Internal Structure

- Forth language environment
  - Dialect of ANSI X3.215-1994
    - Dictionary — The list of Forth words
    - Data Space — The memory used by Forth words
    - Data Stack — The stack used for parameter passing
    - Return Stack — The stack used for procedure nesting
    - Input Buffer — The current line of textual input
    - Input Source — The source device for textual input
    - Output Stream — The destination of textual output
    - Text Interpreter — Processes textual commands
- Device tree
  - Device nodes
    - Elided pathname resolution
    - Device aliases
  - Packages
    - Properties
      - Property names
      - Property values
        - byte array
        - 32-bit integer
        - text string
        - composite values
    - Methods
    - Private data

```
         Instance-specific data
         Static data
Configuration memory
Configuration variables
Custom startup script
Standard property names including
   "name"
   "reg"
   "device_type"
   "interrupts"
   "model"
   "address"
   "compatible"
   "status"
Standard system nodes
   /
      "name"
   /aliases
      "name"
   /openprom
      "name"
      "model"
      "relative-addressing"
   /openprom/client-services
      "name"
   /options
      "name"
   /chosen
      "name"
      "stdin"
      "stdout"
      "bootpath"
      "bootargs"
      "memory"
      "mmu"
   /packages
      "name"
Standard Packages
   Parent methods
      open           Prepare this device for subsequent use.
      close          Close this previously-opened device.
      decode-unit    Convert text unit-string to physical address.
      encode-unit    Convert physical address to text unit-string.
   Generic methods
      selftest       Perform selftest for this device.
      reset          Put this device into a quiescent state.
   Package I/O model
   Expansion bus device class template
      map-in         Map the specified region, return a virtual
                     address.
      map-out        Destroy mapping from previous map-in.
      dma-alloc      Allocate a memory region for later use.
      dma-free       Free memory allocated with dma-alloc.
      dma-map-in     Convert virtual address to device bus DMA
                     address.
      dma-map-out    Free DMA mapping set up with dma-map-in.
      dma-sync       Synchronize (flush) DMA memory caches.
      probe-self     Interpret FCode, as a child of this node.

      "ranges"       Standard property-name to define a device's
                     physical address.

      "#address-cells"       Standard property to define the
```

```
                              package's address format.
      "#size-cells"  Standard property-name to define the package's
                     address size format.
   Memory Management Device Class Template
      claim          Allocate (claim) addressable resource
      release        Free (release) addressable resource.
      map            Create address translation
      unmap          Invalidate existing address translation.
      modify         Modify existing address translation.
      translate      Translate virtual address to physical address
      "available"    The regions of virtual address space denote
                     the virtual address space that is currently
                     unallocated by Open Firmware and is available
                     for use by client programs.
      "existing"     The value of this property defines the regions
                     of virtual address space managed by the MMU,
                     in whose package this property is defined,
                     without regard to whether or not these regions
                     are currently in use.
      "translations" The value of this property describes the translations
                     in use by Open Firmware.
      "page-size"    The value of this property describes the number of
                     bytes in the smallest mappable region of virtual
                     address space.
Standard device types
   "display"
      open           Prepare this device for subsequent use.
      close          Close this previously-opened device.
      write          Write memory buffer to device, return actual
                     byte count.
      draw-logo      Calls draw-logo routine for this device.
      restore        Restore device to useable state after
                     unexpected reset.
      "character-set"        Standard property to specify the
                             character set.

   "block"
      open           Prepare this device for subsequent use.
      close          Close this previously-opened device.
      read           Read device into memory buffer, return actual
                     byte count.
      write          Write memory buffer to device, return actual
                     byte count.
      seek           Set device position for next read or write.
      load           Load a client program from device to memory.
      size           Return the size of the device in bytes.
      #blocks        Return the size of the device in blocks.
      offset-low     Returns the less significant cell of the double
                     number denoting the beginning offset of the disk
                     partition that was specified when the "disk-label"
                     support package was opened.
      offset-high    Returns the more significant cell of the double
                     number denoting the beginning offset of the disk
                     partition that was specified when the "disk-label"
                     support package was opened.

   "byte"
      open           Prepare this device for subsequent use.
      close          Close this previously-opened device.
      read           Read device into memory buffer, return actual
                     byte count.
      write          Write memory buffer to device, return actual
                     byte count.
```

```
   seek           Set device position for next read or write.
   load           Load a client program from device to memory.
   size           Return the size of the device in bytes.
   #blocks        Return the size of the device in blocks.

"network"
   open           Prepare this device for subsequent use including
                  parsing of arguments as defined by "Device Support
                  Extensions" recommended practice.
   close          Close this previously-opened device.
   read           Read device into memory buffer, return actual
                  byte count.
   write          Write memory buffer to device, return actual
                  byte count.
   load           Load a client program from device to memory.
   "local-mac-address"   Standard property to specify
                         preassigned network address.
   "mac-address"         Standard property to specify network
                         address last used.
   "address-bits"        Standard property to indicate network
                         address length.
   "max-frame-size"      Standard property to indicate maximum
                         packet size.
   "chosen-network-types"
                         Standard property reporting the network types
                         that can be supported by this device.
   "chosen-network-type" Standard property reporting the network type
                         that is being supported by this device.

"serial"
   open           Prepare this device for subsequent use including
                  parsing of arguments as defined by "Device Support
                  Extensions" recommended practice.
   close          Close this previously-opened device.
   read           Read device into memory buffer, return actual
                  byte count.
   write          Write memory buffer to device, return actual
                  byte count.
   install-abort  Begin polling for a console abort sequence.
   remove-abort   Cease polling for a console abort sequence.
   restore        Restore device to useable state after
                  unexpected reset.
   ring-bell      Ring the bell.
   set-mode       Sets the device mode according to a string of the
                  same format as the arguments to the open method.
   set-modem-control
                  Sets the RTS and DTR signals as specified by a
                  bitmask.

"memory"
   claim          Allocate (claim) addressable resource.
   release        Free (release) addressable resource.
   "reg"          Standard property defining the physical
                  addresses installed in the system, without
                  regard to whether or not that memory is
                  currently in use by Open Firmware or a client
                  program.
   "available"    Standard "reg" format property defining the
                  regions of physical address space that are
                  currently unallocated by Open Firmware.

The following device types defined by the "Device Support Extensions"
Recommended Practice are also provided by FirmWorks's implementations.
```

"keyboard"
   open   Prepare this device for subsequent use including
                  parsing of arguments as defined by "Device Support
                  Extensions" recommended practice.
   close   Close this previously-opened device.
   read   Read device into memory buffer, return actual
                  byte count.
   "language"   Standard property that indicates the current scan-code
                  to character conversion to which the keyboard driver
                  is currently set.
"mouse"
   open   Prepare this device for subsequent use.
   close   Close this previously-opened device.
   get-event   Obtain the next event from the mouse.
   "#buttons"   Standard property that indicates the number of physical
                  buttons supported by the device.
   "absolute-position"
                  Standard property that indicates that the device
                  supplies absolute X,Y coordinates. Absence of this
                  property indicates that the device supplies relative
                  X,Y position (e.g. a mouse).
"rtc"
   open   Prepare this device for subsequent use.
   close   Close this previously-opened device.
   get-time   Return the current time as year, month, day, hour,
                  minute and second.
   set-time   Set the current time as year, month, day, hour,
                  minute and second.
"sound"
   open   Prepare this device for subsequent use.
   close   Close this previously-opened device.
   read   Acquire sound data, storing the samples into memory
                  buffer.
   write   Output sound samples stored in memory buffer.
   "#input-channels"
                  Standard property that defines the possible numbers of
                  input channels supported.
   "#output-channels"
                  Standard property that defines the possible numbers of
                  output channels supported.
   "sample-precisions"
                  Standard property that defines the possible numbers of
                  bits required to store one audio sample from one
                  channel.
   "sample-frame-size"
                  Standard property that defines the possible numbers of
                  bits required to store one sample frame - one sample
                  from each channel.
   "input-frame-rates"
                  Standard property that defines the possible input
                  sampling rates, in sample frames per second.
   "output-frame-rates"
                  Standard property that defines the possible output
                  sampling rates, in sample frames per second.
   "input-encoding-types"
                  Standard property that defines the possible input
                  encoding types.
   "output-encoding-types"
                  Standard property that defines the possible output
                  encoding types.
"nvram"
   open   Prepare this device for subsequent use.

```
      close          Close this previously-opened device.
      read           Read device into memory buffer, return actual
                     byte count.
      write          Write memory buffer to device, return actual
                     byte count.
      seek           Set device position for next read or write.
      size           Return the number of NVRAM bytes available to the
                     client interface.
      "#bytes"       Standard property that describes the number of bytes
                     the device is capable of storing.
   "parallel"
      open           Prepare this device for subsequent use.
      close          Close this previously-opened device.
      write          Write memory buffer to device, return actual
                     byte count.
```

Standard Support Packages

These standard "libraries" provide support within the system ROM for accessing file systems, booting/loading files from network devices using TFTP and either RARP or BOOTP for name resolution, deblocking "block" devices, and terminal emulation and character rendering for the Open Firmware console.

The existance of these packages:

* Minimizes the size of FCode programs by providing services that are guaranteed to be on any 1275-compliant system.
* Speeds FCode driver development time by reducing the amount of new code that must be written and debugged.
* Enables FCode drivers written today to work with file systems and protocols developed tomorrow since support for such new technology is the responsibility of the system ROM.

```
   "disk-label"
      open           Prepare this device for subsequent use.
      close          Close this previously-opened device.
      load           Load a client program from device to memory.
      offset         Convert partition-relative disk position to
                     absolute position.
      offset-low     Returns the less significant cell of the double
                     number denoting the beginning offset of the disk
                     partition that was specified when the "disk-label"
                     support package was opened.
      offset-high    Returns the more significant cell of the double
                     number denoting the beginning offset of the disk
                     partition that was specified when the "disk-label"
                     support package was opened.
      size           Return the size of the device in bytes.

   "obp-tftp"
      open           Prepare this device for subsequent use.
      close          Close this previously-opened device.
      load           Load a client program from device to memory.

   "deblocker"
      open           Prepare this device for subsequent use.
      close          Close this previously-opened device.
      read           Read device into memory buffer, return actual
                     byte count.
      write          Write memory buffer to device, return actual
                     byte count.
      seek           Set device position for next read or write.

   "terminal emulator"
```

Interprets ANSI X3.64 control sequences

Sequence	ANSI X3.64 Mnemonic	Affected Words
ESC[#A	Cursor Up (CUU)	Affects: line#
ESC[#B	Cursor Down (CUD)	Affects: line#
ESC[#C	Cursor Forward (CUF)	Affects: column#
ESC[#D	Cursor Backward (CUB)	Affects: column#
ESC[#E	Cursor Next Line (CNL)	Affects: line#
ESC[#1;#2f	Cursor Position (CUP)	Affects: line# and column#
ESC[#1;#2H	Cursor Position (CUP)	Affects: line# and column#
ESC[J	Erase in Display (ED)	Uses: delete-characters and delete-lines
ESC[K	Erase in Line (EL)	Uses: delete-characters
ESC[#L	Insert Line (IL)	Uses: insert-lines
ESC[#M	Delete Line (DL)	Uses: delete-lines
ESC[#@	Insert Character (ICH)	Uses: insert-characters
ESC[#P	Delete Character (DCH)	Uses: delete-characters
ESC[#m	Select Graphic Rendition (SGR)	Affects: inverse?

Character	Description
CTRL-G (0x7)	An audible indicator sounds or a visible indication is given.
CTRL-H (0x8)	The cursor moves one position to the left on the current line. If it is already at the left edge of the screen, nothing happens.
CTRL-I (0x9)	The cursor moves right on the current line to the next tab stop. The tab stops are fixed at every multiple of eight columns. If the cursor is already at the right edge of the screen, nothing happens; otherwise the cursor moves right a minimum of one and a maximum of eight character positions.
CTRL-J (0xA)	The cursor moves down one line, remaining at the same character position on the line. If the cursor is already at the bottom line, the screen scrolls up before the cursor is moved down.
CTRL-K (0xB)	The cursor moves up one line, remaining at the same character position on the line. If the cursor is already at the top line, nothing happens.
CTRL-L (0xC)	The cursor is positioned to the Home position (upper-left corner) and the entire screen is cleared.
CTRL-M (0xD)	The cursor moves to the leftmost character position on the current line.

Sequence	Description
ESC[p	If inverse-screen? is false, does nothing. If inverse-screen? is true, sets it to false, changes the value of inverse? to its opposite value (i.e. from true to false or vice versa), and executes invert-screen. The effect of this is to establish the default foreground and background colors for the entire screen.
ESC[q	If inverse-screen? is true, does nothing. If inverse-screen? is false, sets it to true,

changes the value of inverse? to its opposite value (i.e. from true to false or vice versa), and executes invert-screen. The effect of this is to establish inverted foreground and background colors for the entire screen (i.e the screen background uses the default foreground color, and vice versa).

ESC[s — Resets the display device associated with the terminal emulator.

is-install	Create open and other methods for this display device.
is-remove	Create close method for this display device.
is-selftest	Create selftest method for this display device.
line#	Return the current cursor line number.
column#	Return the current cursor column number.
inverse?	Indicates how to paint characters.
inverse-screen?	Indicates how to paint the background.
#lines	Return number of lines of text in text window.
#columns	Return number of columns of text in text window.

Display device low-level interfaces

draw-character	Draw a character at the current cursor position.
reset-screen	Perform frame buffer device initialization.
toggle-cursor	Toggle the state of the text cursor.
erase-screen	Clear the screen.
blink-screen	Flash the screen.
invert-screen	Exchange the foreground and background colors.
insert-characters	Insert n spaces to the right of the cursor.
delete-characters	Delete n characters to the right of the cursor.
insert-lines	Insert n blank lines at and below the cursor line.
delete-lines	Delete n lines at and below the cursor line.
draw-logo	Draw (at line#) the logo stored at location addr.

Frame buffer support routines

default-font	Return the font parameters for the default system font.
set-font	Set the current font as specified.
>font	Return beginning address for char in the current font.
frame-buffer-adr	Return current frame buffer virtual address.
screen-height	Return total height of the display in pixels.
screen-width	Return total width of the display in pixels.
window-top	Return window top border in pixels.
window-left	Return window left border in pixels.
char-height	Return the height of a font character in pixels.
char-width	Return the width of a font character in pixels.
fontbytes	Return interval between entries in the font table.

Eight-bit frame buffer support routines

fb8-install	Install all built-in generic 8-bit frame buffer routines.
fb8-draw-character	Implement the "fb8" draw-character

```
                              function.
         fb8-reset-screen     Implement the "fb8" reset-screen
                              function.
         fb8-toggle-cursor    Implement the "fb8" toggle-cursor
                              function.
         fb8-erase-screen     Implement the "fb8" erase-screen
                              function.
         fb8-blink-screen     Implement the "fb8" blink-screen
                              function.
         fb8-invert-screen    Implement the "fb8" invert-screen
                              function.
         fb8-insert-characters Implement the "fb8" insert-characters
                              function.
         fb8-delete-characters Implement the "fb8" delete-characters
                              function.
         fb8-insert-lines     Implement the "fb8" insert-lines
                              function.
         fb8-delete-lines     Implement the "fb8" delete-lines
                              function.
         fb8-draw-logo        Implement the "fb8" draw-logo function.

Internal Procedures
   Startup sequence
      Initial selftest            Not required / not provided
      Firmware initialization
         Determining the memory configuration.
         Creating Open Firmware environment.
         Initializing various devices required for the basic
          functioning of Open firmware.
         Initializing a "fallback" diagnostic output device.
         Testing configuration memory and resetting if required.
      Startup script evaluation
      Plug-in device probing
      Console selection
      Secondary selftest
      Booting
   Path resolution
      Path resolution including resolution of elided components
Device Interface
   Contents
      FCode evaluator
      /packages standard system node
      Complete set of support packages as defined by IEEE 1275-1994
      Additional support packages to support FAT file systems, etc.
   FCodes
      Forth FCode Functions
         Standard Forth
            dup         Duplicate the top item on the stack.
            2dup        Duplicate the top two items on the stack.
            ?dup        Duplicate top stack item if it is non-zero.
            over        Copy second stack item to top of stack.
            2over       Copy second pair of stack items to top of
                        stack.
            pick        Copy u-th stack item to top of stack.
            tuck        Copy top stack item underneath the second
                        stack item.
            drop        Remove top item from the stack.
            2drop       Remove top two items from the stack.
            nip         Remove the second stack item.
            roll        Rotate u+1 stack items as shown.
            rot         Rotate top three stack items as shown.
            -rot        Rotate top three stack items as shown.
            2rot        Rotate three pairs of stack items as shown.
```

swap	Exchange top two stack items.
2swap	Exchange top two pairs of stack items.
>r	Move top stack item to the return stack.
r>	Move top return stack item to the stack.
r@	Copy top return stack item to the stack.
depth	Return count of items on the stack.
+	Add nu1 to nu2.
-	Subtract nu2 from nu1.
*	Multiply nu1 by nu2.
/	Divide n1 by n2, return quotient.
mod	Divide n1 by n2, return remainder.
/mod	Divide n1 by n2, return remainder and quotient.
u/mod	Divide n1 by n2, all unsigned.
abs	Return absolute value of n.
negate	Return negation of n1.
max	Return greater of n1 and n2.
min	Return lesser of n1 and n2.
bounds	Prepare arguments for do or ?do loop.
lshift	Shift x1 left by u bit-places. Zero-fill low bits.
rshift	Shift x1 right by u bit-places. Zero-fill high bits.
2*	Shift x1 left by one bit-place. Zero-fill low bit.
2/	Shift x1 right by one bit-place. High bit unchanged.
and	Return bitwise logical ’and’ of x1 and x2.
or	Return bitwise logical ’inclusive-or’ of x1 and x2.
xor	Return bitwise logical ’exclusive-or’ of x1 and x2.
invert	Invert all bits of x1.
d+	Add d1 to d2 giving double-number d.sum.
d-	Subtract d2 from d1 giving double-number difference d.diff.
um*	Unsigned multiply with double number product.
um/mod	Divide unsigned double number ud by u.
char+	Increment addr1 by the value of /c.
cell+	Increment addr1 by the value of /n.
chars	Multiply nu1 by the value of /c.
cells	Multiply nu1 by the value of /n.
aligned	Increase n1 as necessary to give valid address boundary.
@	Fetch item x from cell at a-addr.
2@	Fetch cell pair from a-addr.
c@	Fetch byte from addr.
!	Store item x to cell at a-addr.
2!	Store cell pair at a-addr.
+!	Add nu to cell at a-addr.
c!	Store byte to addr.
move	Copy len bytes from src-addr to dest-addr.
fill	Set len bytes beginning at addr to the value byte.
key?	Return true if an input character available.
key	Read a character from the console input device.
expect	Get edited input line, storing it at addr.
span	Variable holding number of characters received by expect.
bl	ASCII code for "space" (blank) character.
emit	Display the given ASCII character.
type	Display text-len characters beginning at address text-str.
cr	Subsequent output goes to the next line.

count	Unpack a counted string to a text string.
base	Variable containing the number-conversion radix.
.	Display number (and trailing space).
u.	Display an unsigned number, with a trailing space.
.r	Display a signed number, right-justified.
u.r	Display an unsigned number, right-justified.
.s	Display entire stack contents, unchanged.
<#	Initialize pictured numeric output conversion.
#	Convert a digit in pictured numeric output conversion.
#s	Convert remaining digits in pictured numeric output.
#>	End pictured numeric output conversion.
hold	Add char in pictured numeric output conversion.
sign	If n < 0 , insert "-" in pictured numeric output.
<	Return true if n1 is less than n2.
<>	Return true if x1 is not equal to x2.
=	Return true if x1 is equal to x2.
>	Return true if n1 is greater than n2.
within	Return true if n is between min and max-1, inclusive.
0<	Return true if n is less than zero.
0<>	Return true if n is not equal to zero.
0=	Return true if nu|flag is equal to zero.
0>	Return true if n is greater than zero.
u<	Return true if u1 is less than u2, unsigned.
u>	Return true if u1 is greater than u2, unsigned.
i	Return current loop index value.
j	Return next outer loop index value.
unloop	Discard loop control parameters.
evaluate	Evaluate Forth text from the given string.
execute	Execute the command whose execution token is xt.
exit	Exit from the currently-executing command.
abort	Abort program execution, clear stacks.
catch	Execute command indicated by xt. Return throw result.
throw	Transfer back to catch routine.
here	Return current dictionary pointer.
c,	Compile a byte into the dictionary.
,	Append x to data space.
compile,	Compile the behavior of the word given by xt.
state	Variable containing true if in compilation state.
>body	Convert execution token to data field address.

Basic Forth extensions

/c	The number of address units to a byte, one.
/w	The number of address units to a doublet, typically two.
/l	The number of address units to a quadlet, typically four.
/n	The number of address units in a cell.
ca+	Increment addr1 by index times the value of /c.
wa+	Increment addr1 by index times the value of /w.
la+	Increment addr1 by index times the value of /l.
na+	Increment addr1 by index times the value of /n.
wa1+	Increment addr1 by the value of /w.
la1+	Increment addr1 by the value of /l.
/w*	Multiply nu1 by the value of /w.

/l*	Multiply nu1 by the value of /l.
w@	Fetch doublet w from waddr.
<w@	Fetch doublet from waddr, sign-extended.
l@	Fetch quadlet from qaddr.
w!	Store doublet w to waddr.
l!	Store quadlet to qaddr.
w,	Compile a doublet w into the dictionary (doublet-aligned).
l,	Compile a quadlet into the dictionary (doublet-aligned).
off	Store false to cell at a-addr.
on	Store true to cell at a-addr.
u#	Convert a digit in pictured numeric output conversion.
u#s	Convert remaining digits in pictured numeric output.
u#>	End pictured numeric output conversion.
comp	Compare two arrays of length len.
lbsplit	Split a quadlet into four bytes.
lwsplit	Split a quadlet into two doublets.
wbsplit	Split a doublet w into two bytes.
bljoin	Join four bytes to form a quadlet.
bwjoin	Join two bytes to form a doublet w.
wljoin	Join two doublets to form a quadlet.
wbflip	Swap the bytes within a doublet.
wbflips	Swap the bytes within each doublet in the given region.
lbflip	Reverse the bytes within a quadlet.
lbflips	Reverse the bytes within each quadlet in the given region.
lwflip	Swap the doublets within a quadlet.
lwflips	Swap the doublets within each quadlet in the given region.
u2/	Shift x1 right by one bit-place. Zero-fill high bit.
between	Return true if n is between min and max, inclusive.
>=	Return true if n1 is greater than or equal to n2.
<=	Return true if n1 is less than or equal to n2.
0<=	Return true if n is less than or equal to zero.
0>=	Return true if n is greater than or equal to zero.
u<=	Return true if u1 less or equal to u2, unsigned.
u>=	Return true if u1 greater or equal to u2, unsigned.
>>a	Arithmetic shift x1 right by u bit-places.
body>	Convert data field address to execution token.
noop	Do nothing.
bell	ASCII code for "bell" character.
bs	ASCII code for "backspace" character.
#line	Variable holding the output line number.
#out	Variable holding the output column number.
pack	Pack a text string into a counted string.
lcc	Convert ASCII char1 to lower-case.
upc	Convert ASCII char1 to upper-case.
-1	Constant -1.
0	Constant 0.
1	Constant 1.
2	Constant 2.
3	Constant 3.
(cr	Output the carriage-return character, (0x0D).
$number	Convert a string to a number.
digit	Convert a character to a digit in the given base.
$find	Find the command named name-string in the

dictionary.
alloc-mem Allocate len bytes of memory.
free-mem Free memory allocated by alloc-mem.

FCode implementation functions

Defining new FCode functions

instance	Mark next defining word as instance-specific.
new-token	Create a new unnamed FCode function.
named-token	Create a new possibly-named FCode function.
external-token	Create a new named FCode function.
b(;)	End an FCode colon definition.
b(:)	Defines type of new FCode function as "colon definition".
b(buffer:)	Defines type of new FCode function as buffer:.
b(constant)	Defines type of new FCode function as constant.
b(create)	Defines type of new FCode function as create word.
b(defer)	Defines type of new FCode function as defer word.
b(field)	Defines type of new FCode function as field.
b(value)	Defines type of new FCode function as value.
b(variable)	Defines type of new FCode function as variable.
(is-user-word)	Create a new named user interface command.
get-token	Convert FCode Number to function execution token.
set-token	Assign FCode Number to existing function.

Literals

b(lit)	Numeric literal FCode. Followed by FCode-num32.
b(')	Function literal FCode. Followed by FCode#.
b(")	String literal FCode. Followed by FCode-string.

Controlling values and defers

behavior	Retrieve execution behavior of a defer word.
b(to)	FCode for setting values and defers. Followed by FCode#.

Control flow

offset16	Makes subsequent FCode-offsets use 16-bit (not 8-bit) form.
bbranch	Unconditional branch FCode. Followed by FCode-offset.
b?branch	Conditional branch FCode. Followed by FCode-offset.
b(<mark)	Target of backward branches.
b(>resolve)	Target of forward branches.
b(loop)	End FCode do ... loop. Followed by FCode-offset.
b(+loop)	End FCode do ... +loop. Followed by FCode-offset.
b(do)	Begin FCode do ... loop. Followed by FCode-offset.
b(?do)	Begin FCode ?do ... loop. Followed by FCode-offset.
b(leave)	Exit from a do ... loop.
b(case)	Begin a case (multiple selection) statement.

```
      b(endcase)  End a case (multiple selection) statement.
      b(of)       FCode for of in case statement. Followed by
                  FCode-offset.
      b(endof)    FCode for endof in case statement. Followed by
                  FCode-offset.

Package access
   Open/close packages
      find-package   Locate the support package named by
                     name-string.
      open-package   Open the package indicated by phandle.
      $open-package  Open the package named by name-string.
      close-package  Close the specified package instance.
      my-self        Return the ihandle of the current instance.
      my-parent      Return the ihandle of the parent of the
                     current instance.
      ihandle>phandle   Return the phandle for the indicated
                        ihandle.
      next-property  Return the name of the property following
                     previous of phandle.
      peer           Return the phandle of the next sibling node.
      child          Return the phandle of the first child node
                     of parent.

   Call methods from other packages
      find-method    Find the method named method-string in the
                     package phandle.
      call-package   Execute the method xt within the instance
                     ihandle.
      $call-method   Execute the method named method-string in
                     the instance ihandle.
      $call-parent   Execute the method named method-string in
                     the parent instance.

   Get local arguments
      my-address  Return low component(s) of device's physical
                  address.
      my-space    Return high component of device's physical
                  address.
      my-unit     Return the unit-address of the current
                  instance.
      my-args     Return the instance-argument string for this
                  instance.
      left-parse-string   Split the string at first occurrence
                          of delimiter char.
      parse-2int  Convert a "hi,lo" string into a pair of values.

   Mapping tools
      map-low        Map the specified region, return a virtual
                     address.
      free-virtual   Destroy mapping and "address" property.

Property management
   Property array encoding
      encode-int      Encode a number into a prop-encoded-array.
      encode-string   Encode a string into a prop-encoded-array.
      encode-bytes    Encode a byte array into a
                      prop-encoded-array.
      encode-phys     Encode a unit-address into a
                      prop-encoded-array.
      encode+         Concatenate two prop-encoded-arrays into
                      a single array.
      sbus-intr>cpu   Converts SBus interrupt level to CPU
```

interrupt level.

Property array decoding

decode-int	Decode a number from a prop-encoded-array.
decode-phys	Decode a unit-address from a prop-encoded-array.
decode-string	Decode a string from a prop-encoded-array.

Property declaration

property	Create a new property with the given name and value.
delete-property	Delete the named property in the active package.
device-name	Create the "name" property, value is indicated string.
device-type	Create "device_type" property, value is indicated string.
reg	Create the "reg" property with the given values.
model	Create the "model" property, value is indicated string.

Property value access

get-package-property	Return value for name-string property in package phandle.
get-inherited-property	Return value for given property in the current instance or its parents.
get-my-property	Return value for given property in this package.

Display device management

Terminal emulator routines

line#	Return the current cursor line number.
column#	Return the current cursor column number.
inverse?	Indicates how to paint characters.
inverse-screen?	Indicates how to paint the background.
#lines	Return number of lines of text in text window.
#columns	Return number of columns of text in text window.
draw-character	Draw a character at the current cursor position.
reset-screen	Perform frame buffer device initialization.
toggle-cursor	Toggle the state of the text cursor.
erase-screen	Clear the screen.
blink-screen	Flash the screen.
invert-screen	Exchange the foreground and background colors.
insert-characters	Insert n spaces to the right of the cursor.
delete-characters	Delete n characters to the right of the cursor.
insert-lines	Insert n blank lines at and below the cursor line.
delete-lines	Delete n lines at and below the cursor line.
draw-logo	Draw (at line#) the logo stored at location addr.

```
   Frame buffer support routines
      default-font   Return the font parameters for the default
                     system font.
      set-font       Set the current font as specified.
      >font          Return beginning address for char in the
                     current font.

      frame-buffer-adr    Return current frame buffer virtual
                          address.
      screen-height       Return total height of the display in
                          pixels.
      screen-width        Return total width of the display in
                          pixels.
      window-top          Return window top border in pixels.
      window-left         Return window left border in pixels.
      char-height         Return the height of a font character
                          in pixels.
      char-width          Return the width of a font character
                          in pixels.
      fontbytes           Return interval between entries in the
                          font table.

   Display device support
      Frame-buffer package interface
         is-install    Create open and other methods for this
                       display device.
         is-remove     Create close method for this display
                       device.
         is-selftest   Create selftest method for this display
                       device.

      Generic eight-bit frame buffer support
         fb8-install             Install all built-in generic
                                 8-bit frame buffer routines.
         fb8-draw-character      Implement the "fb8"
                                 draw-character function.
         fb8-reset-screen        Implement the "fb8"
                                 reset-screen function.
         fb8-toggle-cursor       Implement the "fb8"
                                 toggle-cursor function.
         fb8-erase-screen        Implement the "fb8"
                                 erase-screen function.
         fb8-blink-screen        Implement the "fb8"
                                 blink-screen function.
         fb8-invert-screen       Implement the "fb8"
                                 invert-screen function.
         fb8-insert-characters   Implement the "fb8"
                                 insert-characters function.
         fb8-delete-characters   Implement the "fb8"
                                 delete-characters function.
         fb8-insert-lines        Implement the "fb8"
                                 insert-lines function.
         fb8-delete-lines        Implement the "fb8"
                                 delete-lines function.
         fb8-draw-logo           Implement the "fb8" draw-logo
                                 function.

Other FCode functions
   Peek/poke
      cpeek       Attempt to fetch the byte at addr.
      wpeek       Attempt to fetch the doublet w at waddr.
      lpeek       Attempt to fetch the quadlet at qaddr.
      cpoke       Attempt to store the byte to addr.
```

wpoke — Attempt to store the doublet w to waddr.
lpoke — Attempt to store the quadlet to qaddr.

Device-register access
- rb@ — Fetch a byte from device register at addr.
- rw@ — Fetch a doublet w from device register at waddr.
- rl@ — Fetch a quadlet from device register at qaddr.
- rb! — Store a byte to device register at addr.
- rw! — Store a doublet w to device register at waddr.
- rl! — Store a quadlet to device register at qaddr.

Time
- get-msecs — Return elapsed time, in milliseconds.
- ms — Delay for at least n milliseconds.
- alarm — Execute xt repeatedly, at intervals of n milliseconds.
- user-abort — After alarm routine is finished, abort program execution.

System information
- fcode-revision — Return revision level of FCode interface.
- mac-address — Return a sequence of bytes containing network address.

FCode selftest
- display-status — Display the results of a device selftest.
- memory-test-suite — Perform tests of memory, starting at addr for len bytes.
- mask — Variable to control bits tested with memory-test-suite.
- diagnostic-mode? — If true, boot from diag sources, perform longer selftests.

Start and end
- start0 — Begin program with spread 0. Followed by FCode-header.
- start1 — Begin program with spread 1. Followed by FCode-header.
- start2 — Begin program with spread 2. Followed by FCode-header.
- start4 — Begin program with spread 4. Followed by FCode-header.
- version1 — Begin program with spread 1. Followed by FCode-header.
- end0 — Cease evaluating this FCode Program.
- end1 — Cease evaluating this FCode Program.
- ferror — Standard FCode Number for undefined FCode Functions.
- suspend-fcode — Pause FCode Evaluation if desired, can resume later.
- new-device — Start new package, as child of active package.
- finish-device — Finish this package, set active package to parent.
- byte-load — Evaluate FCode beginning at location addr.
- set-args — Set address and arguments of new device node.

Client Program Interface
Specification
Implements the standard set of client interface services.

```
   Provides the client execution environment specified by the
    PowerPC Binding.
Client interface services
   Client interface
      test   Determines whether the specified service is available
             in this implementation.
      test-method
             Determines whether the specified device method is
             available in the specified device node.

   Device tree
      peer       Obtains the identifier of the device node that is
                 the next sibling of the specified device node, or
                 reports that there are no more siblings.
      child      Obtains the identifier of the device node that is
                 the first child of the specified device node, or
                 reports that there are no children.
      parent     Obtains the identifier of the device node that is
                 the parent of the specified device node, or
                 reports that this node is the root node.
      instance-to-package
                 Translates an ihandle to a phandle.
      instance-to-packages
                 Translates an ihandle to a fully-qualified pathname,
                 including any interposed packages.
      getproplen Obtains information about the specified property
                 in the specified node.
      getprop    Obtains the value of the specified property from
                 the specified node.
      nextprop   Obtains the value of the property which follows
                 the specified property in the property list of the
                 specified node.
      setprop    Sets the property value of the property name in
                 the specified device node, creating the property
                 if necessary.
      canon      Converts the possibly-ambiguous device-specifier
                 to a fully-qualified pathname.
      finddevice Obtains phandle of the device node specified by
                 device-specifier.
      instance-to-path
                 Returns the fully-qualified pathname corresponding to the
                 identifier ihandle.
      package-to-path
                 Returns the fully-qualified pathname corresponding
                 to the specified node.
      call-method
                 Executes the specified package method in the
                 specified instance as with $call-method, guarded
                 by catch and returns the result(s).
   Device I/O
      open       Opens the specifed package.
      close      Closes the specified instance.
      read       Executes the read method in the specified instance.
      write      Executes the write method in the specified
                 instance.
      seek       Executes the seek method in the specified instance.
   Memory
      claim      Allocates memory and returns a virtual address.
      release    Frees memory starting at specified virtual address.
   Control transfer
      boot       Exits the client program, resets the system, and
                 re-boots the system with the specified device and
                 arguments.
```

enter — Enters the Open Firmware command interpreter. The client program may be resumed if the user continues execution with the go command.

exit — Exits from the client program. The execution of the client program may not be resumed.

chain — Frees memory starting at specified virtual address, then executes another client program beginning at specified address. The argument buffer is copied into the Open Firmware memory and passed to the other program.

User interface

interpret — Executes the specified Forth command line, guarded by catch. Returns the result(s).

set-callback — This service sets the callback handler to the specified address.

set-symbol-lookup — Sets the symbol table resolution defer words sym>value and value>sym so that they execute the client program callbacks whose addresses are given by the specified arguments. If either argument is zero, the corresponding defer word is set to the action of false.

sym-to-value — Searches for the specified symbol. Returns the symbol's value or an error indication.

value-to-sym — Locates the symbol whose value is closest to but not greater than the specified value, returning the non-negative offset from the value of that symbol to the specified value, and that symbol's name.

Time

milliseconds — Returns a number which increases periodically, representing the passage of time in units of one millisecond.

User Interface

Specification

A standard command interpreter

Forth Language command group

FCode Debugging command group

Administration command group

Firmware Debugging command group

Client Program Debugging command group

FirmWorks Extensions

Standard command intepreter

Command-line editing including extensions

Return (Enter)	Finish editing the line; making it available to the program.
^b	Moves backward one character.
esc-b	Moves backward one word.
^f	Moves forward one character.
esc-f	Moves forward one word.
^a	Moves backward to beginning of line.
^e	Moves forward to end of line.
Delete	Erases previous character.
Backspace	Erases previous character.
^h	Erases previous character.
esc-h	Erases from beginning of word to just before the cursor, storing erased characters in a save buffer.

```
      ^w              Erases from beginning of word to just before
                      the cursor, storing erased characters in a
                      save buffer.
      ^d              Erases next character.
      esc-d           Erases from cursor to end of the word,
                      storing erased characters in a save buffer.
      ^k              Erases from cursor to end of line, storing
                      erased characters in a save buffer.
      ^u              Erases entire line, storing erased
                      characters in a save buffer.
      ^r              Retypes the line.
      ^q              Quotes next character (allows you to insert
                      control characters).
      ^y              Inserts the contents of the save buffer
                      before the cursor.
   Command-line history
      ^p        Selects and displays the previous line for
                subsequent editing.
      ^n        Selects and displays the next line for subsequent
                editing.
      ^l        Displays the entire command history list.
   Command-completion
      ^<space>       Complete this word.
      ^? or ^/       Show all possible matches.

Forth language command group
   Stack
      Stack duplication
         dup         Duplicate the top item on the stack.
         2dup        Duplicate the top two items on the stack.
         3dup        Duplicate three stack items.
         ?dup        Duplicate top stack item if it is non-zero.
         over        Copy second stack item to top of stack.
         2over       Copy second pair of stack items to top of
                     stack.
         pick        Copy u-th stack item to top of stack.
         tuck        Copy top stack item underneath the second
                     stack item.
      Stack removal
         clear       Empty the stack.
         drop        Remove top item from the stack.
         2drop       Remove top two items from the stack.
         3drop       Remove top three items from the stack.
         nip         Remove the second stack item.
      Stack rearrangement
         roll        Rotate u+1 stack items.
         rot         Rotate top three stack items.
         -rot        Rotate top three stack items.
         2rot        Rotate three pairs of stack items.
         swap        Exchange top two stack items.
         2swap       Exchange top two pairs of stack items.
      Return stack
         >r          Move top stack item to the return stack.
         r>          Move top return stack item to the stack.
         r@          Copy top return stack item to the stack.
      Stack depth
         depth       Return count of items on the stack.
   Arithmetic
      Single-precision integer arithmetic
         +           Add nu1 to nu2.
         -           Subtract nu2 from nu1.
         *           Multiply nu1 by nu2.
         u*          Multiply u1 by u2 yielding uprod, all unsigned.
```

```
   /           Divide n1 by n2, return quotient.
   */          Calculate n1 times n2 divided by n3.
   mod         Divide n1 by n2, return remainder.
   /mod        Divide n1 by n2, return remainder and quotient.
   */mod       Calculate n1 times n2 divided by n3.
   u/mod       Divide n1 by n2, all unsigned.
   1+          Add 1 to nu1.
   1-          Subtract 1 from nu1.
   2+          Add 2 to nu1.
   2-          Subtract 2 from nu1.
   abs         Return absolute value of n.
   negate      Return negation of n1.
   max         Return greater of n1 and n2.
   min         Return lesser of n1 and n2.
   bounds      Prepare arguments for do or ?do loop.
   even        Round to nearest even integer >= n.
Bitwise logical operators
   lshift      Shift x1 left by u bit-places. Zero-fill low
               bits.
   rshift      Shift x1 right by u bit-places. Zero-fill high
               bits.
   >>a         Arithmetic shift x1 right by u bit-places.
   <<          Synonym for lshift.
   >>          Synonym for rshift.
   2*          Shift x1 left by one bit-place. Zero-fill low
               bit.
   u2/         Shift x1 right by one bit-place. Zero-fill
               high bit.
   2/          Shift x1 right by one bit-place. High bit
               unchanged.
   and         Return bitwise logical 'and' of x1 and x2.
   or          Return bitwise logical 'inclusive-or' of x1
               and x2.
   xor         Return bitwise logical 'exclusive-or' of x1
               and x2.
   invert      Invert all bits of x1.
   not         Synonym for invert.
Double number arithmetic
   s>d         Convert a number to a double number.
   d+          Add d1 to d2 giving double number d.sum.
   d-          Subtract d2 from d1 giving double number
               d.diff.
   um*         Unsigned multiply with unsigned double number
               product.
   m*          Signed multiply with double-number product.
   um/mod      Divide ud by u.
   fm/mod      Divide d by n.
   sm/rem      Divide d by n, symmetric division.
Data type conversion
   lbsplit     Split a quadlet into four bytes.
   lwsplit     Split a quadlet into two doublets.
   wbsplit     Split a doublet into two bytes.
   bljoin      Join four bytes to form a quadlet.
   bwjoin      Join two bytes to form a doublet.
   wljoin      Join two doublets to form a quadlet.
   wbflip      Swap the bytes within a doublet.
   lbflip      Reverse the bytes within a quadlet.
   lwflip      Swap the doublets within a quadlet.
Address arithmetic
   /c          The number of address units to a byte, one.
   /w          The number of address units to a doublet,
               typically two.
   /l          The number of address units to a quadlet,
```

typically four.
/n The number of address units in a cell.
ca+ Increment addr1 by index times the value of /c.
wa+ Increment addr1 by index times the value of /w.
la+ Increment addr1 by index times the value of /l.
na+ Increment addr1 by index times the value of /n.
ca1+ Synonym for char+.
wa1+ Increment addr1 by the value of /w.
la1+ Increment addr1 by the value of /l.
na1+ Synonym for cell+.
/c* Synonym for chars.
/w* Multiply nu1 by the value of /w.
/l* Multiply nu1 by the value of /l.
/n* Synonym for /n*.
aligned Increase n1 as necessary to give a var-aligned address.
char+ Increment addr1 by the value of /c.
cell+ Increment addr1 by index times the value of /n.
chars Multiply nu1 by the value of /c.
cells Multiply nu1 by the value of /n.

Memory control

Memory access

@ Fetch item x from address a-addr.
! Store item x to address a-addr.
2@ Fetch two items from a-addr; item x2 from lower address.
2! Store items x1 and x2 to a-addr; x2 at lower address.
c@ Fetch byte from addr.
c! Store byte to addr.
w@ Fetch doublet w from waddr.
<w@ Fetch doublet w from waddr, sign-extended.
w! Store doublet w to waddr.
l@ Fetch quadlet from qaddr.
l! Store quadlet to qaddr.
unaligned-w@ Fetch doublet w from addr, any alignment is allowed.
unaligned-w! Store doublet w to addr, any alignment is allowed.
unaligned-l@ Fetch quadlet from addr, any alignment is allowed.
unaligned-l! Store quadlet to addr, any alignment is allowed.
comp Compare two arrays of length len.
dump Display len bytes of memory starting at addr.
+! Add nu to the number stored at address a-addr.
off Store false at address a-addr.
on Store true at address a-addr.
move Copy len bytes from src-addr to dest-addr.
fill Set len bytes beginning at addr to the value byte.
blank Set len bytes beginning at addr to the value 0x20.
erase Set len bytes beginning at addr to zero.
wbflips Swap the bytes within each doublet in the given region.
lbflips Reverse the bytes within each quadlet in the given region.
lwflips Swap the doublets within each quadlet in the given region.

Memory allocation

```
      alloc-mem   Allocate len bytes of memory.
      free-mem    Free memory allocated by alloc-mem.
Text input and output
   Text input
      (           Ignore immediately-following text, up to
                  closing ")".
      \           Ignore immediately-following text on this line.
      >in         Variable containing offset of next input
                  buffer character.
      parse       Parse text from the input buffer, delimited by
                  delim.
      parse-word  Parse text from the input buffer, delimited by
                  space.
      source      Return the location and size of the input
                  buffer.
      word        Parse text from the input buffer, delimited by
                  delim.
   Console input
      key?        Return true if an input character is available.
      key         Read a character from the console input device.
      expect      Get an edited input line, storing it at addr.
      span        Variable containing number of characters
                  received by expect.
      accept      Get an edited input line, storing it at addr.
   ASCII constants
      bell        ASCII code for "bell" character.
      bl          ASCII code for "space" (blank) character.
      bs          ASCII code for "backspace" character.
      carret      ASCII code for "carriage-return" character.
      linefeed    ASCII code for "linefeed" character.
      ascii       Generate ASCII code for immediately-following
                  character.
      char        Generate ASCII code for next character from
                  input buffer.
      [char]      Generate ASCII code for next character from
                  input buffer.
      control     Generate control-code for immediately-following
                  character.
   Console output
      ."          Display immediately-following text.
      .(          Display immediately-following text up to
                  delimiting ")".
      emit        Display the given ASCII character.
      type        Display text-len characters beginning at
                  address text-str.
   Output formatting
      cr          Subsequent output goes to the next line.
      space       Display a single space.
      spaces      Display cnt spaces.
      #line       Variable holding the output line number.
      #out        Variable holding the output column number.
   Display pause
      exit?       Return true when output should be terminated.
   String literals
      "           Gather the immediately-following string or hex
                  data.
      s"          Gather the immediately-following string.
   String manipulation
      count       Unpack a counted string to a text string.
      pack        Pack a text string into a counted string.
      lcc         Convert ASCII char1 to lower-case.
      upc         Convert ASCII char1 to upper-case.
      -trailing   Remove trailing spaces from string.
```

```
Numeric input and output
   Numeric-base control
      base        Variable containing the numeric conversion
                  radix.
      decimal     Set numeric conversion radix to ten.
      hex         Set numeric conversion radix to sixteen.
      octal       Set numeric conversion radix to eight.
   Numeric input
      $number     Convert a string to a number.
      >number     Convert string to a number, add to d1.
      digit       Convert a character to a digit in the given
                  base.
      d#          Interpret the following number as a decimal
                  number.
      h#          Interpret the following number as a
                  hexadecimal number.
      o#          Interpret the following number as an octal
                  number.
   Numeric output
      .           Display number, with a trailing space.
      s.          Display a signed number, with a trailing space.
      u.          Display an unsigned number, with a trailing
                  space.
      .r          Display a signed number, right-justified.
      u.r         Display an unsigned number, right-justified.
      .d          Display a signed number (and space) in decimal.
      .h          Display a signed number (and space) in hex.
      .s          Display entire stack contents, unchanged.
      ?           Display the number at address a-addr.
   Numeric output primitives
      (.)         Convert a number into a text string.
      (u.)        Convert an unsigned number into a text string.
      <#          Initialize pictured numeric output conversion.
      #           Convert a digit in pictured numeric output
                  conversion.
      #s          Convert remaining digits in pictured numeric
                  output.
      #>          End pictured numeric output conversion.
      hold        Add char in pictured numeric output conversion.
      sign        If n < 0 , insert "-" in pictured numeric
                  output.
      u#          Convert a digit in pictured numeric output
                  conversion.
      u#s         Convert remaining digits in pictured numeric
                  output.
      u#>         End pictured numeric output conversion.
Comparison operators
   <              Return true if n1 is less than n2.
   <=             Return true if n1 is less than or equal to n2.
   <>             Return true if x1 is not equal to x2.
   =              Return true if x1 is equal to x2.
   >              Return true if n1 is greater than n2.
   >=             Return true if n1 is greater than or equal to
                  n2.
   between        Return true if n is between min and max,
                  inclusive.
   within         Return true if n is between min and max-1,
                  inclusive.
   0<             Return true if n is less than zero.
   0<=            Return true if n is less than or equal to zero.
   0<>            Return true if n is not equal to zero.
   0=             Return true if nu|flag is equal to zero.
```

```
   0>             Return true if n is greater than zero.
   0>=            Return true if n is greater than or equal to
                  zero.
   u<             Return true if u1 is less than u2, unsigned.
   u<=            Return true if u1 less or equal to u2,
                  unsigned.
   u>             Return true if u1 is greater than u2, unsigned.
   u>=            Return true if u1 greater or equal to u2,
                  unsigned.
Flag constants
   false          Return the value false (zero).
   true           Return the value true (negative one).
Control-flow commands
   Conditional branches
      if          If flag is true, execute following code.
      else        When if flag was false, execute following code.
      then        Terminate an if construct.
   Case statement
      case        Begin a case (multiple selection) statement.
      of          Begin of clause, execute through endof if
                  params match.
      endof       Mark end of clause, jump to end of case if
                  match.
      endcase     Mark end of a case statement.
   Conditional loops
      begin       Begin a conditional loop.
      until       End a begin...until loop. Exits loop if flag
                  is true.
      again       End an (infinite) begin...again loop.
      while       Conditional test within begin...while...repeat
                  loop.
      repeat      End a begin...while...repeat loop. Jump to
                  begin.
   Counted loops
      do          Start a counted loop, beginning index value is
                  start.
      ?do         Similar to do, but do not execute loop if
                  limit = start.
      loop        Add one to index, then return to the previous
                  do or exit the loop.
      +loop       Add delta to index, return to the previous do
                  or exit the loop.
      i           Return current loop index value.
      j           Return next outer loop index value.
      leave       Exit this do or ?do loop immediately.
      ?leave      If flag is true, exit this do or ?do loop
                  immediately.
      unloop      Discard loop control parameters.
   Other control flow commands
      eval        Synonym for evaluate.
      evaluate    Interpret Forth text from the given string.
      execute     Execute the command whose execution token is
                  xt.
      exit        Exit from the currently-executing command.
   Error handling
      quit        Abort program execution.
      abort       Abort program execution, clear stacks.
      abort"      If flag is true, display text and call abort.
      catch       Execute command indicated by xt. Return throw
                  result.
      throw       Transfer back to catch routine.
Forth dictionary
   Defining words
```

```
   constant    Create a named constant. new-name returns
               value x.
   2constant   Create a named two-number constant.
   value       Create a named variable, change with to.
   variable    Create a named variable. new-name returns
               address a-addr.
   buffer:     Creates a named data buffer. new-name returns
               address.
   :           Begin creation of a colon definition.
   ;           End creation of a colon definition.
   alias       Create a new command equivalent to an existing
               command.
   defer       Create a command with alterable behavior,
               alter with to.
   struct      Start a struct...field definition.
   field       Create new field offset specifier, named
               new-name.
   create      Create a new command, behavior set by further
               commands.
   does>       Specify run-time behavior of a created word.
   $create     Call create, new name specified by name-string.
   marker      Define a marker for subsequent dictionary
               cleanup.

Dictionary commands
   Data space allocation
      here     Return current dictionary pointer.
      allot    Allocate len bytes in the dictionary.
      align    Allocate dictionary bytes to leave top of
               dictionary var-aligned.
      c,       Compile a byte into the dictionary.
      w,       Compile a doublet w into the dictionary
               (doublet-aligned).
      l,       Compile a quadlet into the dictionary
               (doublet-aligned).
      ,        Compile a cell into the dictionary (doublet-aligned).
   Immediate words
      immediate   Declare the previous definition as
                  "immediate".
      state       Variable containing true if in compile
                  state.
      [           Enter interpret state.
      ]           Enter compile state.
      compile     Compile following command at run time.
      [compile]   Compile immediately-following command.
      literal     Compile a number, later leave it on the
                  stack.
      postpone    Delay execution of immediately-following
                  command.
      compile,    Compile the behavior of the word given by
                  xt.
   Dictionary search
      [']         Return execution token xt of a command.
      '           Return execution token xt of a command,
                  parsed later.
      find        Find command, return -1 (found), +1
                  (immediate), or 0 (not found).
   Miscellaneous dictionary
      to          Change value or defer or machine register
                  contents.
      behavior    Retrieve execution behavior of a defer word.
      >body       Convert execution token to data field
                  address.
```

```
            body>       Convert data field address to execution
                        token.
            noop        Do nothing.
            recursive   Make current definition visible, for
                        recursive call.
            recurse     Compile recursive call to the command being
                        compiled.
            forth       Make Forth the context vocabulary.
            environment?   Return system information based on input
                           keyword.
      Assembler
         code        Begin creation of machine-code command called
                     new-name.
         label       Begin machine-code sequence, leave addr on
                     stack.
         c;          End creation of machine-code command, will
                     return to caller.
         end-code    End creation of machine-code sequence.
Administration command group
   Help
      help   Provide information for category or specific command.
   System startup
      Power On Self Test (POST)
      System Initialization
      Evaluate the script (if use-nvramrc? is true)
      probe-all (evaluate FCode)
      install-console
      banner
      Secondary Diagnostics
      Default boot (if auto-boot? is true)
      Suppression of "probe-all install-console banner" sequence
       when either banner or suppress-banner is executed from the
       script.
   Booting
      Device and argument selection controlled by configuration
      variables
         auto-boot?
         boot-command
         diagnostic-mode?
         boot-device
         boot-file
         diag-device
         diag-file
      Argument passing
         "bootpath" property in the /chosen node.
         "bootargs" property in the /chosen node.
      User commands for booting
         Resolves boot command ambiguity
            If the word following boot on the command line begins
             with a slash ( / ) character, it is a device-path and,
             thus, a device-specifier.
            Otherwise, if there is a device alias matching that
             word, the word is a device-specifier.
            If that word is neither a device-path nor a known alias,
             the default boot device is used and the word is
             included in the boot arguments.
         User commands
            boot             Load and execute a specified program.
            diagnostic-mode? If true, boot from diag sources,
                             perform longer selftests.
            diag-switch?     If true, diagnostic-mode? returns true.
            boot-device      Default boot device-name when
                             diagnostic-mode? is false.
```

```
         boot-file        Default boot arguments when
                          diagnostic-mode? is false.
         diag-device      Default boot device-name when
                          diagnostic-mode? is true.
         diag-file        Default boot arguments when
                          diagnostic-mode? is true.
         auto-boot?       If true, boot automatically after
                          power-on or reset-all.
Non-volatile memory
   Configuration variables
      Data Types
         integer
         bytes
         string
         boolean
         security-mode
      Commands to inspect and modify configuration variables.
         setenv            Set the specified configuration
                           variable to the specified value.
         $setenv           Set the specified configuration
                           variable to the specified value.
         printenv          Display current, default value of
                           configuration variable (or all).
         set-default       Set specified configuration variable
                           to default value.
         set-defaults      Reset most configuration variables to
                           their default values.
         nodefault-bytes   Create custom configuration variable.
      Open Firmware Configuration Variables
         Not all variables apply to all platforms.
         auto-boot?        If true, boot automatically after
                           power on or reset.
         boot-command      Command that is executed if
                           auto-boot? is true.
         boot-device       Device from which to boot if
                           diagnostic-mode? is false.
         boot-file         Arguments passed to booted program if
                           diagnostic-mode? is false.
         diag-device       Device from which to boot if
                           diagnostic-mode? is true.
         diag-file         Arguments passed to booted program if
                           diagnostic-mode? is true.
         diag-switch?      If true, run in diagnostic mode.
         fcode-debug?      If true, include name fields for
                           plug-in device FCodes.
         input-device      Console input device (usually
                           keyboard, ttya, or ttyb).
         nvramrc           Contents of NVRAMRC.
         oem-banner        Custom OEM banner (enabled by
                           oem-banner? true).
         oem-banner?       If true, use custom OEM banner.
         oem-logo          Byte array custom OEM logo (enabled
                           by oem-logo? true). Displayed in
                           hexadecimal.
         oem-logo?         If true, use custom OEM logo (else,
                           use default system logo).
         output-device     Console output device (usually
                           screen, ttya, or ttyb).
         screen-#columns   Number of on-screen columns
                           (characters/line).
         screen-#rows      Number of on-screen rows (lines).
         security-#badlogins   Number of incorrect security
                               password attempts.
```

security-mode Firmware security level (options: none, command, or full).
security-password Firmware security password
selftest-#megs Megabytes of RAM to test. Ignored in diagnostic mode.
use-nvramrc? If true, execute commands in NVRAMRC during system start-up.
pci-probe-list Which PCI bus device numbers to probe and in what order.
little-endian? If true, the endian mode of the machine is little-endian.
real-mode? If true, the address translation mode of Open Firmware is Real-Mode. If false, the address translation mode is Virtual-Mode.
real-base This integer variable defines the starting physical address to be used by Open Firmware.
real-size This integer variable defines the size of the physical address space which can be used by Open Firmware.
virt-base This integer variable defines the starting virtual address which can be used by Open Firmware.
virt-size This integer variable defines the size of the virtual address space which can be used by Open Firmware.
load-base This integer variable defines the default load address for client programs when using the load method. The default value is implementation dependent.
reboot-command-address This integer variable defines the (real) address of the reboot command as defined in Section 4.1 of the PowerPC binding.

The script

Editor keystroke command differences

^c Exits the script editor, returning to the Open Firmware command interpreter. The temporary buffer is preserved, but is not written back to the script. (Use nvstore afterwards to write it back.)
<cr> Inserts a newline at the cursor position and advances to the next line.
^o Inserts a newline at the cursor position and stays on the current line.
^k If at the end of a line, joins the next line to the current line (i.e., deletes the newline).
^n Moves to the next line of the script editing buffer.
^p Moves to the previous line of the script editing buffer.
^l Displays the entire contents of the editing buffer.

Miscellaneous

nvramrc Contents of the script.
use-nvramrc? If true, the script is evaluated at system start-up.

Editor commands

nvedit Enter script editor (exit with ^c).
nvstore Copy contents of nvedit temporary buffer

```
                        into the script.
         nvquit         Discard contents of nvedit temporary
                        buffer.
         nvrecover      Attempt to recover lost script contents.
         nvrun          Execute the contents of the nvedit
                        temporary buffer.
I/O control
   General
      Creates "stdin" property in the /chosen node.
      Creates "stdout" property in the /chosen node.
      Uses serial port as diagnostic output device.
      Implementation-dependent action taken on failure to open
       an input or output device.
   Facilities
      input-device      Default console input device.
      output-device     Default console output device.
      stdin             Variable containing the ihandle of the
                        console input device.
      stdout            Variable containing the ihandle of the
                        console output device.
      screen-#columns   Maximum number of columns on console
                        output device.
      screen-#rows      Maximum number of rows on console output
                        device.
      install-console   Select and activate console input and
                        output devices.
      input             Select the indicated device for console
                        input.
      output            Select the indicated device for console
                        output.
      io                Select the indicated device for console
                        input and output.

Security
   password             Prompt user to set security password.
   security-mode        Contains level of security access
                        protection.
   security-password    Contains security password text string.
   security-#badlogins  Contains total count of invalid security
                        access attempts.

Reset
   reset-all      Reset the machine as if a power-on reset had
                  occurred.

Selftest
   test               Invoke the selftest routine for the
                      specified device.
   test-all           Invoke "selftest" routines at and below
                      specified node.
   selftest-#megs     Number of megabytes of memory to test.
   diagnostic-mode?   If true, boot from diag sources, perform
                      longer selftests.
   diag-switch?       If true, diagnostic-mode? returns true.

Client program callback
   callback       Execute specified client program callback
                  routine.
   $callback      Execute specified client program callback
                  routine.
   sync           Flush system file buffers, after a program
                  interrupt.
```

```
   Banner
      banner            Display the system power-on banner.
      suppress-banner   Abbreviate system startup sequence after
                        the script.
      oem-logo?         If true, banner displays custom logo in
                        oem-logo.
      oem-logo          Contains custom logo for banner, enabled by
                        oem-logo?.
      oem-banner?       If true, banner displays custom message in
                        oem-banner.
      oem-banner        Contains custom banner text, enabled by
                        oem-banner?.

   Device tree
      show-devs      Show all devices beneath the indicated node.

      Device alias
         devalias    Create device alias, or display current
                     alias(es).
         nvalias     Create non-volatile device alias, edit the
                     script.
         $nvalias    Create non-volatile device alias, edit the
                     script.
         nvunalias   Delete non-volatile device alias, edit the
                     script.
         $nvunalias  Delete non-volatile device alias, edit the
                     script.
         "screen"    Standard string for alias created by
                     install-console.

      Device tree browsing
         dev           Make the specified device node the active
                       package.
         find-device   Make the device node dev-string the active
                       package.
         device-end    Unselect the active package, leaving none
                       selected.
         pwd           Display the device-path that names the
                       active package.
         ls            Display the names of the active package's
                       children.
         .properties   Display names and values of properties of
                       the active package.

      Device probing
         probe-all   Probe for all available plug-in devices.

Firmware debugging command group
   Automatic stack display
      showstack      Turn on automatic stack display.
      noshowstack    Turn off showstack (automatic stack display).

   Serial download
      dl     Download and execute Forth text, end with ^d.

   Dictionary
      Dictionary search
         .calls      Display all commands which use the execution
                     token xt.
         $sift       Display all command-names containing
                     text-string.
         sifting     Display all command-names containing specified
                     text.
```

```
      words       Display the names of methods or commands.

   Decompiler
      see         Decompile the Forth command old-name.
      (see)       Decompile the Forth command whose execution
                  token is xt.

   Patch
      patch       Change contents of specified word.
      (patch)     Change contents of word specified by xt.

   Forth source-level debugger
      <space>     Executes the word just displayed and proceeds to
                  the next word.
      d           Goes "down a level", i.e., mark for debugging
                  the word whose name was just displayed and
                  execute it.
      u           Goes "up a level", i.e., unmark the word being
                  debugged, mark its caller for debugging and
                  finish executing the word that was previously
                  being debugged.
      c           "continue"; switch from stepping to tracing,
                  thus tracing the remainder of the execution of
                  the word being debugged.
      f           Starts a subordinate Forth interpreter. Forth
                  commands may be executed normally. When the
                  resume command is encountered, the interpreter
                  exits and control returns to the debugger at the
                  place where the f keystroke was executed.
      q           "Quits", i.e., aborts the execution of the word
                  being debugged and all its callers, returning to
                  the command interpreter.
      debug       Mark the command old-name for debugging.
      (debug      Mark the command indicated by xt for debugging.
      stepping    Set "step mode" (default) for Forth source-level
                  debugging.
      tracing     Set "trace mode" for Forth source-level
                  debugging.
      debug-off   Turn off the Forth source-level debugger.
      resume      Exit from a "subordinate interpreter" back to
                  the stepper.

Client program debugging command group
   Supports all standard Forth capabilities.  In addition:
   Registers display
      Saves the complete state of the machine whenever a
       machine-language program execution is suspended.
      CPU registers may be examined and modified.
      Registers may be written or read individually, or read as a
       group.
      For registers other than floating point registers, the
       register access commands operate on memory copies of the
       register values, instead of directly on the processor
       registers themselves.
      Floating point registers are accessed "in-place".
      Processor registers are copied to the saved-program-state
       memory area when a running program transfers control to the
       Open Firmware as a result of a user abort, a program
       breakpoint, or a severe system crash (resulting in a
       watchdog reset).
      When execution of the suspended program is resumed with the
       go command, the processor registers are reloaded from the
       saved-program-state area and any modifications that the user
```

has made prior to resumption of the program will then take effect.

The register names vary with processor type. For PowerPC they are:

%f0 through %f31
: Return the value in the specified floating point register.

%fpscr
: Return the value in the floating point status and control register.

%r0 through %r31
: Return the value in the specified fixed-point register.

%sprg0 through %sprg3
: Return the value in the specified SPRG register.

%srr0 and %srr1
: Return the value in the specified Save/Restore register.

%pc
: An alias for %srr0.

%cr %ctr %lr %msr %xer
: Return the value in the specified register.

Execution of a register name pushes the value contained in that register (or its memory copy) onto the stack. The value may be changed with the to command.

ctrace	Display saved call stack (subroutines calls and arguments).
.registers	Display values in %r0 through %r31, %sprg0 through %sprg3, %srr0 and %srr1, plus %cr, %ctr, %lr, %msr, %xer.
.fregisters	Display the values in %f0 through %f31.
.pc	Display the value in %pc.
to	Change the value stored in any of the above registers.

Program download and execute

load	Load a program, specified by params.
go	Execute or resume execution of a program in memory.
state-valid	Variable, true if saved-program-state is valid.
init-program	Initialize saved-program-state.

Abort and resume

Control-Break	Suspend the currently executing program, saving processor state in the saved-program-state memory area, and enter the Open Firmware command interpreter.
go	Execute or resume execution of a program in memory.

Disassembler

dis	Begin disassembling at the given address.
+dis	Continue disassembling where dis or +dis last stopped.

Breakpoints

.bp	Display a list of all locations which are breakpoints.
+bp	Add the given address to the breakpoint list.
-bp	Remove the breakpoint at the given address.
--bp	Remove most recently-set breakpoint (repeat if desired).
bpoff	Remove all breakpoints from the breakpoint list.
step	Executes a single machine-code instruction.

```
      steps          Execute step n times.
      hop            Execute single instruction, or entire
                     subroutine call.
      hops           Execute hop n times.
      go             Execute or resume execution of a program in
                     memory.
      gos            Execute go n times.
      till           Execute until the given address. Equivalent to:
                        +bp go
      return         Execute until return from this subroutine.
      .breakpoint    Action performed when breakpoint occurs.
      .step          Action performed when a single step occurs.
      .instruction   Display next pending address and instruction.

   Symbolic debugging
      .adr           Display symbolic form for the given address.
      sym            Return value of specified client program
                     symbol.
      sym>value      Defer word to resolve symbol names.
      value>sym      Defer word to resolve symbol values.

FCode debugging command group
      external       Newly-created functions will be visible.
      headerless     Newly-created functions will be invisible.
      headers        Newly-created functions will be optionally
                     visible.
      fcode-debug?   If true, save names for FCodes with headers.
      open-dev       Open device (and parents) named by given
                     device-specifier.
      begin-package  Set up device tree, before creating new node.
      close-dev      Close device and all of its parents.
      end-package    Close the device tree entry set up with
                     begin-package.
      execute-device-method
                     Execute the named method in the specified
                     package.
      apply          Execute named method in the specified package.
      decode-bytes   Decode a byte array from a prop-encoded-array.

Tokenizer
   Host systems
      Unix
      MS-DOS(r)
      Windows(tm) NT
   Tokenizer Behavior
      Read an FCode text file, one word at a time.
      If the word read is an existing FCode name (with an assigned
       FCode#), generate the appropriate FCode# and append that number
       to the FCode binary file.
      If the word read is a standard Tokenizer macro (indicated by a
       type-code of "T"), generate the appropriate series of FCode#'s
       as specified in the description of the command. Some macros
       will cause more words to be read from the input FCode text
       file. These usually have a stack comment including [...], i.e.
       [text<delim>].
      If the word which was read is a Tokenizer-only command (these
       are listed later), perform the appropriate action as specified.
      Otherwise, print an error message that the particular word is not
       recognized.
      Tokenizing behavior continues until the end of the FCode text
       file is encountered. If there were no errors, the FCode binary
       file is created.
   Tokenizer-only commands
```

```
      Manual Tokenizer output
         tokenizer[   Enter Tokenizer-escape mode, allowing manual
                      FCode generation.
         ]tokenizer   Exit Tokenizer-escape mode, resumes FCode
                      interpretation.
         emit-byte    Output given FCode#, only in Tokenizer-escape
                      mode.

      File inclusion
         fload        Insert the specified file at this point.

FirmWorks Experience-Driven Extensions
   Additional Keyboard Chords
      Control-Alt-D   Enter diagnostic mode
      Control-Alt-N   Reset NVRAM contents to default values

   Additional tools for working with configuration variables
      editenv   Displays the current value of the specified variable
                and enables use of normal line editing keystrokes.  If
                variable does not exist, creates it.

   Additional control over the dictionary
      definitions   Make the current vocabulary the context vocabulary.
      only          Resets search order to initial state.
      also          Duplicates the first vocabulary in the search
                    order.
      previous      Drops the first vocabulary in the search order.
      order         Displays the current search order.
      sift-devs     Similar to sifting, but searches the methods in
                    the device tree's nodes instead of the dictionary.

   File system extensions
      x:          Change the current drive to x, where x is a function
                  of the drives available on a given system.  Typical
                  systems have a:, c: and maybe d: or e:.
      df          Displays the number of bytes available on the
                  current drive.
      pcwd        Display the name of the current working directory.
      chdir <directory-name>  Change the current working directory to
                              the directory specified by
                              <directory-name>.
      chdir" directory-name"  Similar to chdir except that
                              directory-name is delimited with a
                              trailing " .

      dir [pattern] Display the statistics of the files and/or
                    directories specified by pattern. pattern may
                    contain the wildcard characters listed in the
                    table below. If pattern is null, display the
                    current working directory. If pattern explicitly
                    specifies the name of a directory, display the
                    statistics of the contents of that directory.
      dir" pattern" Similar to dir except that pattern is delimited
                    with a trailing " .
      mkdir directory-name    Create the directory specified by
                              directory-name. If directory-name
                              contains no pathname components, create
                              directory-name in the current working
                              directory.
      mkdir" directory-name"  Similar to mkdir except that
                              directory-name is delimited with a
                              trailing " .
      rmdir directory-name    Delete the directory specified by
```

	directory-name. If directory-name contains no pathname components, delete directory-name from the current working directory.
rmdir" directory-name"	Similar to rmdir except that directory-name is delimited with a trailing " .

Some file system commands allow the use of wildcard characters in their filename arguments; wildcard characters are not permitted in directory names. The available wildcard characters are:

Syntax	Function
.	Separates filenames from file extensions as in foo.bat
*	Matches zero or more characters.
?	Matches exactly one character including periods (i.e. f?????? matches foo.bat).

All other characters match a single occurrence of themselves.

copy filename1 filename2	Make a copy of the file named filename1 in a new file named filename2.
copy pattern directory	Make a copy of all files matching pattern in the directory named directory.
delete pattern	Delete all file(s) matching pattern. If pattern contains no pathname components, delete all files matching pattern from the current working directory.
del pattern	An alias for delete.
rm pattern	An alias for delete.
delete" pattern"	Similar to delete except that pattern is delimited with a trailing " .
del" pattern"	An alias for delete".
rm" pattern"	An alias for delete".
rename filename1 filename2	Change the name of the file named filename1 to filename2.
ren filename1 filename2	An alias for rename.
mv filename1 filename2	An alias for rename.

$chdir	Change to the specified directory.
$copy	Copy the contents of file1 to file2.
$create-file	Create the specified file returning the handle with which to access it.
$delete	Delete the specified file.
$delete-all	Delete all files matching the specified pattern. See table above for the wildcard characters and their meanings.
$df	Return as a double number the number of available bytes on the specified file system (e.g. " a:").
$dir	Display the statistics of the file(s) and/or directory(ies) matching the specified pattern. See above table for the wildcard characters and their meanings. If pattern is null, display the current working directory. If pattern explicitly specifies the name of a directory, display the statistics of the contents of that directory.
$disk-size	Return as a double number the total size in bytes

```
                of the specified file system (e.g. " a:" ).
   $mkdir       Create the specified directory.
   $rename      Change the name file1 to file2. Pattern characters
                are allowed in the specification of file1.
   $rmdir       Delete the specified directory. Pattern characters
                are allowed in the specification of file1.

File loading extensions
   dlfcode     Downloads tokenized FCode over a serial link.
   dlbin       Downloads a binary file over a serial link.

Breakpoint extensions
   finish-loop   Execute until the end of this loop.
   returnl       Execute until the end of this leaf subroutine.
   skip          Do not execute (skip) the current instruction.

Source level debugger extensions
   g   Turn off the debugger and continue execution.
   s   Decompile the word being debugged.
   $   Display the address,len on top of stack as a text string.
   h   Display documentation on source debugger keystroke commands.
   ?   Display brief documentation on source debugger keystroke
       commands.
   (   Moves the beginning of the debug region to the current position
       in the word being debugged.
   )   Moves the end of the debug region to the current position in the
       word being debugged.
   *   Expands the debug region to include the entire word.
   <   Moves the beginning of the debug region to just after the current
       position in the word being debugged, and moves the end of the
       debug region to the end of that word.  (Useful for skipping past
       the end of a loop. Step to the word that ends the loop and type
       "<".)
   debug-me
       Compile debug-me into a word to cause the debugger to debug the
       word containing debug-me once debug-me is first encountered.
   debug(
       Compile debug( into a word to invoke the debugger, and make the
       debugger's scope begin just after debug( and continue to the end
       of the word containing debug(.
   )debug
       Compile )debug into a word to invoke the debugger, and make the
       debugger's scope end just after the call of )debug.

Debugging hooks and tools
   Special vocabularies
      magic-properties     Can be used to add custom diagnostic
                           messages to name and other words.  Can
                           be used to snoop property creation.
      magic-device-types   Can be used to snoop device_type
                           creation.

   Device method debugging
      select             select reads a pathname from the input
                         stream and creates an instance chain to
                         the specified node.
      select-dev         Like select except that the pathname is
                         passed as a string on the stack.
      begin-select       Like select except that it doesn't execute
                         the open method at the end of the pathname.
      begin-select-dev   Like select-dev except that it doesn't
                         execute the open method at the end of the
                         pathname.
```

FCode debugging

fcode-verbose?	A flag variable which when true causes extremely verbose FCode compilation during probing and FCode evaluation.

Hooks for expanding diagnostic message detail

fm-hook	Expands "unimplemented package method" diagnostic message.
include-hook	Causes the system to print the name of each file as it is opened.
cif-error-hook	Writes a diagnostic message when the client interface is requested to perform a non-existant service.

Expansion of client program interface diagnostic messages

verbose-cif	Causes a diagnostic message to be written to the output device each time a client interface service is called. The display includes the name of the service, the input parameters to the service and the result(s) from the service.
silent-cif	Turns off the diagnostic messages enabled by verbose-cif.

Exception diagnostic detail

ftrace	Shows the sequence of Forth words that were being executed at the time of the last exception.

Decompiler extensions

see-chain	Decompiles all of the entries in a "chained" word e.g. stand-init .

Tokenizer extensions

pci-header	Outputs a PCI Expansion ROM header using default values for "vital product data pointer", "revision level" and "indicator".
pci-header-end	Computes the value of the "image length" field of the PCI Expansion ROM header and updates the field.
set-rev-level	Modifies the default value used by pci-header for the "vital product data pointer" field.
set-vpd-offset	Modifies the default value used by pci-header for the "revision level" field.
not-last-image	Modifies the default value used by pci-header for the "indicator" field.