C! (n addr --) "c store"

Store the lower 8 bits of N at address ADDR in memory. The higher bits are discarded. Used for storing ASCII characters or other 8 bit values. Odd addresses can be used with this word.

Related Words: C@ ABSC! W! !

C@ (addr -- n) "c fetch"

Fetch a byte from address ADDR in memory and leave it on the stack as a 32 bit value with the high 24 bits set to zero. Usually used for accessing ASCII characters and other 8 bit values

Related Words: C! @ ABSC@ W@

CALL (various-params <library> <routine> -- result)

Call an Amiga Library routine. CALL is an IMMEDIATE word, which can ONLY be used in a colon definition. At compile time it will look up the calling parameters in the FD file and build the appropriate code that, at run time, will:

1) Load the correct registers in order from the stack. (The order the arguments appear in source text is the same as listed in the AmigaDOS technical manuals, and with the ARGS command.)

2) Load Amiga register A6 with the correct library base pointer.

3) Call the correct location, offset from the base pointer.

4) Use register D0 as a 'universal' return code, moving it to top of stack.

Please note that addresses passed to the Amiga must be converted from RELATIVE to ABSOLUTE! Please see the words >ABS and CALL>ABS for this. Please also see the chapter on Calling Amiga Library Routines for more information about CALL.

Related Words: CALLVOID CALL>ABS AREGS>ABS :LIBRARY DCALL >REL >ABS

CALL>ABS (various-params <lib> <routine> -- result)

Call an Amiga Library Routine. This differs from CALL by automatically converting any parameters passed in an address register to absolute before calling. NULL values are left untouched. This saves you from having to use IF>ABS . Use ARGS to see which parameters are passed in address registers. Generally routines in GRAPHICS, INTIUITION and EXEC work with CALL>ABS while routines in the DOS library do not because they pass addresses in data registers!

Please see the chapter on Calling Amiga Library Routines for more information about CALL.

Related Words: CALL AREGS>ABS ARGS CALLVOID

CALLADR, (addr-of-code --) "call address comma"

CALLADR, will cause the compiler to generate, at HERE, an optimized indirect reference to the address on the stack. The address MUST contain executable code.

The indirect reference will be assembled in whichever of the following addressing modes is both possible and most efficient:

```
16-bit relative (BSR).
Address-register indirect with offset (JSR).
Long absolute  (JSR).
```

Refer to the Motorola 68000 Technical Manual for more information on addressing modes.

Related Words: CFA, HERE [ ]

CALLVOID (various-parameters <library> <routine> --)

Call an Amiga Library routine without returning a result. As an example the DOS library Delay() routine does not have a result. Here is an example of calling it.

```
: DELAY()  ( n -- , delay for N/50 seconds )
    CALLVOID DOS_LIB DELAY ;
100 DELAY() ( wait two seconds )
```

Please see the chapter on Calling Amiga Library Routines for more information about CALL.

Related Words: CALL AREGS>ABS ARGS CALL>ABS

CANCELKEY? (-- n)

See the chapter on CLONE.

CANCELNOW? (--)

See the chapter on CLONE.

CARRAY (numbytes <name> --) "c array"

Same as ARRAY except that the size of each element is one byte. Creates 1 dimensional array of length N byte-sized values starting at next available dictionary location . Bytes are initialized at compile time to zero's. The word that is created will have the following stack diagram:

MY-CARRAY (index -- addr)

Related Words: ALLOT ALLOCBLOCK ARRAY

CASE (n -- n)

CASE is a Forth construct for selecting between several actions based on integer value. It is used as follows:

```
: EXAMPLE   ( N -- )
   CASE   0 OF  ." None at all! "    ENDOF
      " Still checking... "
      1 OF  ." Only one "        ENDOF
      13 19 RANGEOF  ." In the teens!"       ENDOF
      dup 5,000 > ?OF  ." Lots and lots!"  ENDOF
      ." Unexpected number = " DUP . CR
   ENDCASE   ( ENDCASE does a DROP )
;
```

When executed, EXAMPLE will check the top number on the stack against each of the numbers before each of the "OF"'s. If it matches, it will drop the number that was being checked then execute the program after the OF up to the ENDOF , then jump to the code after the ENDCASE . if the number does not match, it skips to the code after the ENDOF , leaving the number still on the stack.

Similar to switch() in 'C'.

Related Words: OF ?OF RANGEOF ENDCASE ENDOF ELSE

CD (<directory-name> --) "c d"

CD is used to change the current directory, an operation not possible with the DOS function. This is a JForth program, given the same name as its AmigaDOS counterpart.

CD is the only AmigaDOS function that is completely redefined in the JForth environment. Other AmigaDOS commands are available via DOS.

NOTE: for memory considerations, the JForth 'CD' does not display the current directory if typed in with no pathname; this can still be done via "DOS CD".

```
CD JU:     ( to actually change directories )
```

Related Words: DOS

CELL (-- 4) "cell"

CELL is a constant equal to the number of bytes in a stack item. JForth uses 32 bit, 4 byte stack items so CELL = 4.

CELL allows for code transportability between Forths of different sizes, i.e. 16 bit vs. 32 bit.

A convenient method for defining CELL is:

```
SP@ SP@ - ABS CONSTANT CELL
```

Related Words: CELL+ CELL- CELLS CELL/

CELL+ (n -- n+cell) "cell plus"

Add CELL to the value on the stack. Use for address incrementing , making transportable code .

Related Words: CELL

CELL- (n -- n-cell) "cell minus"

Subtract CELL from the value on the stack.

Related Words: CELL

CELL/ (n -- n/cell) "cell slash"

Divide the value on the stack by CELL.

Related Words: CELL

CELLS (n -- n*cell) "cells"

Multiply the value on the stack by CELL.

Related Words: CELL

CFA, (cfa --) "c f a comma"

This is a deferred word used by the compiler to compile a reference to a Forth word. See (CFA,) .

CHOOSE (n -- r)

Choose a random number R between 0 and N-1 inclusive.

Handy for games, statistics, random art and music. Only good for N <= 32767.

CHOOSE is only pseudo-random but is good enough for most simple applications. You can even get a pseudo-gaussian distribution if you add up the results of several CHOOSE operations, say 7.

CHOOSE uses the value in RAND-SEED for its seed. CHOOSE will always give you the same "random" sequence for the same seed. In a program where you want a different random sequence every time, try setting RAND-SEED to the current system time when you start the program.

```
INCLUDE? CHOOSE JU:RANDOM
: ROLL.DIE   ( -- 1-6 )
    6 choose 1+
;
```

Related Words: RANDOM WCHOOSE RAND-SEED

CLIST? (--)

```
CLIST_LIB
CLIST_NAME
```

Used to manage the CLIST library. See :LIBRARY.

CLONE (<wordname> --)

Clone is used to generate a Royalty Free executable image from your Forth programs. It is a two-pass program which first creates a call dependency tree for <wordname>, then uses the tree to build a separate program image of minimum size.

See the chapter on CLONE.

CLOSEALLLIBS (--) "close all libs"

This word will close all of the standard Amiga Libraries, except for EXEC and DOS, which are closed when BYE is called.

Note that this word is not usually needed; the preferred programming technique is to use the XXX? and -XXX words, where XXX is the name of the library. See :LIBRARY.

Note also that CLOSEALLLIBS will not close any additional libraries the programmer has defined with :LIBRARY. It is the responsibility of the calling program to insure these custom libraries are closed.

CLOSEALLLIBS is normally only used by BYE .

Standards: JForth unique

Related Words: BYE LIBRARY

CLOSEFILES (addr --)

CLOSEFILES accepts the address of a memory block (gotten via ALLOCBLOCK) which has been used to store a list of opened files, and will close each file still there.

CLOSEFILES will not free the memory block.

CLOSEFVREAD (var-addr --) "close f v read"

CLOSEFVREAD accepts a variable or user-variable address that contains the location of a virtual buffer which has been used for reading ONLY.

The memory-block serving as the buffer will be freed, and the var-address will be cleared.

See chapter on File I/O.

Related Words: OPENFV READLINE

CLOSEFVWRITE (file-pointer var-addr --)

CLOSEFVWRITE accepts a variable or user-variable address that contains the location of a virtual buffer which has been used for writing, and the file-pointer associated with the buffer.

If the FREEBYTE-counter for the memory-block is non zero indicating data is present, it is written to the file. Then, the memory-block serving as the buffer will be freed, and the var-address will be cleared.

See chapter on File I/O.

Related Words: OPENFV F, FREEBYTE

CLR-BIT (n bit# -- n') "clear bit"

Force a bit in N to zero. BIT# determines which bit is set to zero. BIT# = 0 refers to the least significant bit.

Related Words: SET-BIT

CLRCHAR (-- var-addr) "clear character"

This is a VARIABLE which contains the ASCII value used by CLS to clear the screen.

CLS (--) "c l s"

Clear the screen. Fetch the ASCII value contained in the VARIABLE CLRCHAR, and send it to the standard EMIT device.

Related Words: EMIT ASCII CLRCHAR

CMOVE (source-addr dest-addr count --) "c move"

Move count bytes from source to destination. The move will proceed from low to high address. If the destination region overlaps the source region and the destination is at a higher address you may want to use CMOVE> or MOVE instead to avoid destroying data.

For large amounts of data MOVE is much faster and will handle any overlaps automatically.

Related Words: MOVE CMOVE> $MOVE

CMOVE> (source-addr dest-addr count --) "c move back"

Move COUNT bytes from source to destination. Unlike CMOVE , start with the last byte and work towards the beginning. Useful for overlapping memory areas.

Related Words: MOVE CMOVE

CNT>RANGE (n count -- n+count n) "count to range"

Convert a standard VALUE COUNT argument pair to a LIMIT INDEX argument pair (suitable for entry into a DO-LOOP).

```
: FOO 10 5  CNT>RANGE  DO I . LOOP ;
( will print: 10 11 12 13 14 )
```

CODE (-- , <wordname>)

Invokes the RPN Assembler to create a new word called <wordname>.

See the chapter on 68000 ASSEMBLY.

COLD (--)

Initialize Forth. COLD will reset most of JForth to the same state it was in on startup or when FREEZE was last called. The parts of JForth affected include:

All USER-variables below SPARE (those defined in the kernal)

Vocabulary and dictionary structures.

All MODULEs are DETACHed.

Note that the arrangement of files, memory areas and libraries will not be affected.

Users should be careful not to execute COLD if the system is currently executing definitions (via DEFERred words) which will be purged; the preferred method is to remove these vectors, replacing them with those which exist in the frozen image.

Users can define an AUTO.INIT word to reset their own code when COLD is called. See AUTO.INIT .

Related Words: ABORT COLDEXEC FREEZE

COLDEXEC (--)

COLDEXEC is a DEFERed word; the JForth default contents is NOOP.

COLDEXEC may be set (via IS) to execute a programmer-defined word during COLD, just prior to the call to ABORT. This is called when you first bring up JForth. You can use this to have special initialization automatically occur on startup. It is necessary to FREEZE the image after setting COLDEXEC to have it execute at the next COLD. (This is done automatically by SAVE-FORTH).

```
' MY-INIT-WORD IS COLDEXEC     FREEZE COLD
```

We recommend the use of the new AUTO.INIT facility instead of COLDEXEC.

Related Words: COLD ABORT FREEZE AUTO.INIT

COMPARE (addr1 addr2 #bytes -- result)

Compare two strings, S1 starting at address ADDR1 , the other, S2 starting at address ADDR2 , paying attention to the case of alphabetic characters. The number of bytes to be compared is on top of the stack.

Results are returned as follows:

```
String relationship      Result
     s1 = s2 .............  0
     s1 > s2 .............  1
     s1 < s2 ............. -1
```

An example that uses a compiled string and a string at PAD follows:

```
: BIGGERTHANLIFE? ( $string -- result )
    COUNT DROP " LIFE" COUNT COMPARE ;
" MARYLIN" BIGGERTHANLIFE? . ( print 1 )
```

Related Words: TEXT=? $= MATCH? SCAN $-

COMMAS (--)

Turn on the use of commas in the outputting of numbers. If enabled, commas occur every 3 digits in decimal and every 4 digits otherwise.

```
12345 . ( prints 12,345 )
```

Related Words: (COMMAS) NO-COMMAS DIGS/,

COMPILE (<name> --)

COMPILE is an immediate, compile-only word, the use of which may be illustrated by a sample definition:

```
VARIABLE USE-32BIT
: COMPILE-SIZED-FETCH  ( -- )
    USE-32BIT @
    IF    COMPILE  @
    ( 4 bytes long, compile a @ )
    ELSE  COMPILE w@
    ( 2 bytes long, compile a w@ )
    THEN  ; IMMEDIATE
```

```
VARIABLE VAR1
USE-32BIT OFF
: GETVAL  VAR1 COMPILE-SIZED-FETCH ;
```

Our hypothetical example assumes that we may be compiling for one of two memory systems...one that will need 16-bit fetches when it runs, another at 32 bits. The variable USE-32BIT has been set to the appropriate state for the one we are doing.

The job of the above definition is not to FETCH the correct size at compile-time, but rather to COMPILE the correctly-sized operator to be executed later, when the application actually runs.

Related Words: IMMEDIATE [COMPILE] LITERAL

COMPILING? (-- flag)

Returns a flag reflecting true (if COMPILING) or false (if not COMPILING) based on the value of the variable STATE .

Related Words: STATE ?COMP ?EXEC

CONSOLE?

```
CONSOLE_LIB
CONSOLE_NAME
```

These are used to manage the Amiga console library . See :LIBRARY .

CONSOLE! (file-pointer --) "console store"

This word accepts an AmigaDOS file-pointer, and installs it into the CONSOLEIN and CONSOLEOUT variables, causing the file to become the console through which EMIT and KEY work.

Usually, the file-pointer represents an AmigaDOS console window of either type RAW: CON: or NEWCON:.

CONSOLE@ (-- file-pointer) "console fetch"

This word returns the AmigaDOS file-pointer currently installed as the CONSOLEOUT window.

Usually, the file-pointer represents an AmigaDOS console window of either type RAW: CON: or NEWCON:.

Related Words: EMIT KEY FOPEN

CONSOLEIN (-- var-addr) "console in"

CONSOLEIN is a variable that usually contains the AmigaDOS file-pointer through which KEY will receive characters.

Usually, the file-pointer represents an AmigaDOS console window of either type RAW: CON: or NEWCON:.

Related Words: CONSOLE! CONSOLEOUT CONSOLE@

CONSOLEOUT (-- var-addr) "console out"

CONSOLEOUT is a variable that usually contains the AmigaDOS file-pointer through which EMIT will output characters.

Usually, the file-pointer represents an AmigaDOS console window of either type RAW: CON: or NEWCON:.

Related Words: CONSOLE! CONSOLEIN CONSOLE@

CONSTANT (value <name> --)

Define a Forth word that will return the value when called. The use of constants is recommended whenever possible in your code because it makes it easier to change and easier to understand.

```
123 CONSTANT MYVAL
MYVAL . ( print 123 )
```

Related Words: VALUE VARIABLE CREATE DOES>

CONTEXT (-- addr)

A user-variable containing a pointer to the vocabulary which is first to be searched by FIND. CONTEXT may be set simply by declaring the name of the vocabulary you wish to access.

The function VLIST will display the words in the CONTEXT vocabulary.

See the section on Vocabularies.

Related Words: CURRENT VLIST ALSO PREVIOUS ORDER VOCS FIND WORDS

CONVERT (d1 addr1 -- d2 addr2)

```
d1 = value accumulator
addr1 = address 1 char before numeric string
d2 = d1 + <addr1+1>
addr2 = address of first non-digit
```

Convert a string at address ADDR1+1 to a double number and add it to D1 . Leave the sum D2 on the stack and push the address ADDR2 of the first non-digit found on top of the stack. You can now process the non digit and continue with another call to CONVERT .

Related Words: NUMBER NUMBER?

COUNT ($string -- addr count)

Take a Forth text string and return the address of the first character and the number of characters.

Definition: : COUNT (addr -- addr+1 byte) DUP 1+ SWAP C@ ;

Also used as a C@ with auto increment. See '83 handy reference.

```
" Hello" COUNT TYPE
```

CR (--) carriage return

CR is a DEFERed execution word; see (CR). Transmits a carriage return and line feed to the selected output device . Also flushes the FASTEMIT buffer if FAST I/O mode (line-buffered) is in effect.

Related Words: (CR) CR? >NEWLINE FLUSHEMIT FAST EMIT

CR? (--) carriage return ?

Does a carriage return and line feed if the input text is near the end of a line .

Related Words: CR (CR) OUT >NEWLINE

CREATE (<name> --)

Define a new word in the dictionary.

CREATE creates a "header" structure in the dictionary comprised of:

1) a name field -- built from the next text in the input stream.

2) a link field -- used to tie the headers together for searching. The link field points to the name field of the previous definition.

3) a size field -- used by the compiler to determine compile-time characteristics.

4) a code field -- contains actual 68000 assembly language which determines the run time behavior of the word.

Words that use CREATE are called DEFINING-WORDS. CREATE is used by all defining words, such as CONSTANT , : , VARIABLE , etc... Also used with DOES> to create new data type. See the tutorial for more information and examples.

Related Words: CREATECHAR CONSTANT : VARIABLE DOES> (CREATE) DO-DOES-SIZE SKIP-WORD

CREATECHAR (-- var-addr)

This is a user-variable that holds the character that is used to parse the input stream when a dictionary name-field is being CREATEd. This character is used as a delimiter.

Related Words: CREATE

CSP (-- addr) "c s p"

Variable used by many of the compiling words to check the stack depth. !CSP will set CSP to the current SP value; ?CSP will later verify CSP as unchanged, or generate an error. SP = stack pointer.

Related Words: CSP! ?CSP DEPTH

CURRENT (-- addr)

CURRENT is a variable containing a pointer to the vocabulary to which newly-created words will be attached. It is set to point to a specific vocabulary by executing that vocabulary name, followed by DEFINITIONS.

Related Words: ALSO ONLY VOCABULARY PREVIOUS ORDER VOCS FORTH ABORT COLD CONTEXT