Chapter 3
JForth Disk Organization

JForth is released on three disks named *JForth*, *Extras* and *JTools* [The Freeware version is released in a single archive.]

Each disk contains several directories, each containing a specific set of related files. Only four of the directories do not contain JForth program source files; COM: holds only executable binary images, FD: stores information used by the JForth compiler to interface with the Amiga ROM Kernel, and MOD: holds precompiled code.

Directory Nicknames

JForth utilizes features provided by the AmigaDOS ASSIGN command to create *nicknames* for each of the normal directory pathnames. By executeing (another AmigaDOS command) the *assigns* file that is on the root of each disks, you can *teach* your system these nicknames. This need only be done once per session, when you first insert the disks in your system. If you install JForth on your hard disk as described in chapter one, then these assigns will be done automatically.

There are a number of benefits to using assigned names:

1) They short and require less typing then the full path names.

2) You can always use the same name regardless of the actual pathname of the directory. You can always reference JU:LOCALS with the same name regardless of whether JU: is JForth:Util or WORK:Programming/JForth_Dir/JForth/Util.

Please read about ASSIGN in the Amiga DOS manual if you are not familiar with this very powerful command.

If you enter an assigned name and Amiga DOS asks you to enter a disk with the logical name, then your assigns are not set properly. Check the installation procedure to make sure you have executed the assigns properly. If you are not using a hard disk you can simply enter:

```
EXECUTE JFORTH:ASSIGNS
EXECUTE EXTRAS:ASSIGNS
EXECUTE JTOOLS:ASSIGNS
```

Once this has been done, JForth will be able to locate all required source files.

The following is a description of each JForth directory in alphabetical order. Be sure to read the description of JU: since it contains the most generally useful utilities.

CL: - Extras:Clone - Clone Recompiler

CLONE is compiled by loading CL:TOPFILE. Clone is used to generate stand-alone royalty free applications.

COM: - Extras:Com - Executable Command Images

JKERNAL - This image is the basic Forth engine and primitives. Once executed from AmigaDOS, files from the 'JF:' (Extras:SysGen) directory are INCLUDEd to generate a new, fully-operational, JForth image. Since a completely compiled development image is supplied (see 'JForth', below), this need only be done if the files in the 'JF:' directory have been altered by the programmer.

*** JFORTH** - This is the main JForth executable image. This file is comprised of the kernel image (described above) and utilities compiled from the JF:' directory, notably the disassembler, timing words, a pattern-matching 'VLIST', Hashing, History and others. See the chapter on "System Internals" for more information on recompiling COM:JForth.

FD: - JForth:fd.files

This directory contains all the information that the JForth compiler needs to create the proper *glue* routines to interface to any and all Amiga library calls. As supplied, the files in this directory describe the 2.0 version of the AmigaDOS operating system.

JA: - Extras:Appls - Applications

We have provided several complete applications written in JForth. They can be used either directly from JForth or they can be cloned and used as CLI commands. Some of them demonstrate how to parse the command line, how to handle input errors and printing "help" information. The file names all end with ".f" to distinguish the Forth source file from the cloned image file.

These are described in more detail in an appendix. Please see the chapter on Clone for instructions on how to clone these applications.

JARP: - JTools:JARP - ARP Interface

ARP stands for Amiga DOS Replacement. It is a popular library of calls

ARP.J - Include file for interfacing to ARP library.

ARPFileRequest.f - Simple call to ARP's file requester.

JD: - Extras:Demos

This directory contains the JForth source code for all of the supplied demo programs. You can include each demo individually or include most of them at once by INCLUDEing JD:LOAD_DEMOS. At the end of the compile, instructions on how to execute the demos is printed.

These demos illustrate how to use various aspects of the JForth programming environment and the Amiga. You may find these useful as a starting point for your own programs. Feel free to modify these demos and include parts of them in your commercial products.

BENCH_PRIMITIVES - benchmark programs for primitive operations. Use this to compare JForth with other Forths.

DEMO_BOXES - This program is similar to the BOXES demo that comes with the Amiga. It shows you how to open a window and draw random rectangles. It also illustrates using the XOR mode of drawing, and changing colors.

DEMO_CLICK - Shows how to detect double clicks.

DEMO_CIRCSQ - Graphical pattern generator using algebraic functions.

DEMO_DBUF - Shows how to create a double buffered video display.

DEMO_DOTS - Simple random pixels.

DEMO_FONTS - Shows how to open and use Amiga fonts.

DEMO_GADGET - Shows how to use most of the gadget types available in a simple "do-nothing"

program.

DEMO_GSORT - An animated demonstration of using the Batcher's sort. It generates random length rows and sorts them visually.

DEMO_HAM - This illustrates opening a CUSTOM SCREEN in HAM mode. A BACKDROP window is then opened within it and some random 'art' is generated.

DEMO_INTERRUPT - Shows how to install an interrupt handler. Uses the Vertical Blanking Interrupt to generate a 60 hertz clock. See also **HIGH_INTERRUPT** which shows a way to call high level Forth code from an interrupt (not recommended).

DEMO_LINES - This is similar to the LINES demo from Commodore, except it stays within the window. It generates sweeping line patterns in different colors.

DEMO_MENUS - The use of the EZMENU system is shown here. A set of menus that demonstrate mutual exclusion, checking, setting of command keys, etc. is built and used.

DEMO_MSG - Open a message port and send a message.

DEMO_PAINT - This is a simple paint program that demonstrates the use of graphics input events. A call to EV.GETCLASS is made. The colors are cycled automatically. Detects and responds to double clicks.

DEMO_POLYGON - Use the AREADRAW routines to draw polygons.

DEMO_RAIN - This doesn't teach anything that BOXES doesn't already show but it looks interesting so here it is.

DEMO_RAWKEY - Shows how to get RAWKEY events from a window and convert them.

DEMO_RGB - Cycle colors by changing the color map of a screen.

DEMO_SCROLL - The demo shows you how to make a direct call to an Amiga graphics library routine, ScrollRaster. It also uses line drawing.

DEMO_SPEAK - This small demo illustrates the use of the Amiga Translator library and Narrator device to create a synthetic dialog. You should also examine the file JU:SPEAK for an example of opening a device and calling DoIO and other Exec calls.

DEMO__SPRITE - This demonstrates the creation and use of a hardware sprite. It bounces around on the screen leaving a textile like pattern. If you let it go for a long time it will slowly change the color scheme. You can modify the width and height to get different patterns.

DEMO_STRIP - No this is not pornography! Displays a simple running strip chart.

Fibonacci - Generates the Fibonacci numeric sequence.

HIGH_INTERRUPT - How to call high level Forth code from a 68000 interrupt. Be careful.

LOAD_DEMOS - This contains a menu based program that integrates all of the other demos. This could be used by retailers for customer demos.

SIEVE - The legendary Sieve of Eratosthenes. Generates prime numbers which is sometimes useful for something besides benchmarking compilers.

JDEV: - JTools:DevTools - Development Tools

BLOCK - Support for old style Forth BLOCK or SCREEN environment.

BLOCK2TEXT - Convert an old style Forth BLOCK based file to a normal text file.

*** DEBUGGER** - Source level debugger. This allows you to single step through your code while examining the stack, etc.

EDITOR - Old style FIG Forth line editor. As in Leo Brodie's starting Forth. We recommend using

the more modern text editors or SCRED.

PROFILE - Analyses code performance so you know what to optimize.

SCRED - Screen based block editor, WYSIWYG. Almost enough to make you want to use BLOCKs.

UNUSED - Mark words as used when compiling a file. Handy for stripping old extraneous code out of a system.

WORDCACHE - Caches recently entered words for word completion using a function key.

JF: - Extras:Sysgen - System Generation

The JF: files are necessary for any JForth image...they are loaded on top of the com:JKernal image to produce the com:JForth image, which may be further customized as the programmer sees fit for his particular programming needs.

These files allow the programmer to modify the JForth system at a relatively low level; he may alter and regenerate all but the bottom 23K of code which was written in assembler.

Here is a partial listing of those files.

$TABLE - String Table, string arrays used by disassembler.

.IF - Conditional compilation.

@BITS -

AJF_DICT - low level support for C_STRUCT and ODE.

ANSI - Support for the ANSI terminal escape sequences for fancy screen displays. Change colors, underline, italics, move cursor, etc.

ASM - Reverse Polish 68000 assembler. In a precompiled module.

AUTO - Contains an AUTO.INIT that provides a way of passing a command to JForth from the CLI.

BUILDSYS - lowest level Forth code. Everything below this was written in assembler. Contains IF ELSE THEN and other important stuff.

CALLS - support calls to Amiga libraries.

CASE - CASE OF ENDOF ENDCASE conditionals.

C_STRUCT - Support for the Amiga 'C' structures.

CONDITION - Exotic conditional construct similar to CASE.

DISM - Disassembler. Shows you the 68000 code of a word.

DOSCOMMANDS - Support easy DOS calls from Forth, eg. DIR , COPY

FORWARD-ASM - Motorola style forward assembler. This uses the more familiar 68000 assembler syntax.

HASHING - Hashes the Forth dictionary for faster compilation.

HISTORY - Provide Command Line History and hot Function Keys.

LOADJFORTH - Loads the com:JForth image from com:Minimum. You may want to customize this and recompile.

MAKEINCLUDES - Compile the includes module. You may want to add more to this.

MEASURE - Measure how long it takes to execute something. Great for running benchmarks.

MEMBER - Support for structure members , ..@ and ..@

MODULE - Support for precompiled modules of code like the INCLUDES.

SELECT - Positional case statement. Warning - will not clone! Use CASE or a run-time initialized jump table.

STRING-INTERPRET - Interpret a string containing Forth code.

STRIP-PATHNAME - Strip the path, if any, from a filename.

TRAPS - Trap 68000 exceptions to eliminate some GURU meditation errors.

WORDS-LIKE - Searches dictionary words that contain a substring. Very handy if you can't remember exactly how a word was spelled or for finding related words.

Y-OR-QUIT - Ask the user to answer yes, no or quit.

JFLT: - Extras:Floats - Floating Point

FLOAT.FFP - Single Precision Floating Point Math

FLOAT.DOUBLE - Double Precision Floating Point Math

JI: - JForth:Include

The Include directory contains the JForth-equivalent of the conventional C-language ".h" files. The same file and directory structure has been maintained; the only difference is that the filename suffix is '.j'. The files may be compiled via the normal JForth 'include' command; they are referenced with the nickname 'ji:'. Some of these files have been precompiled into a module called INCLUDES.

JIFF: - Extras:IFF - Interchange File Format

This directory contains the files to support the IFF standard. This is the standard that is used for exchanging pictures, samples or other information between different programs. See chapters 21 and 22. See also JANIM: in this chapter.

DOUBLE_BUFFER - Support for creating double buffered displays which can result in smoother animations.

IFF.J - Structure definitions and Chunk ID definitions, eg. 'FORM', are defined here.

IFF_SUPPORT - Miscellaneous tools for recursively parsing an IFF file.

ILBM_MAKER - Tools for creating an InterLeaved BitMap file.

ILBM_PARSER - Tools for parsing/reading an InterLeaved BitMap file.

LOAD_PIC - Loads Picture system.

PACKING - Routines for packing Bitmaps into BODY chunks and for packing CTABLES into CMAP chunks.

PIC_EFFECTS - Special effects for IFF Pictures like wipes and fades.

PIC_FLIP - reverse and flip a picture.

PICTURES - Picture Animation that allows you to load IFF pictures and perform simple animations, page flipping, transparent blitting, etc.

SHOW_IFF - Example program to display an ILBM graphics picture, plus tools for transparently copying brush bitmaps to a screen.

UNPACKING - The assembly routine for unpacking a Run Length Encoded bitmap, an Uncompressed bitmap, and a CMAP.

TEST_PIC - Simple test/demo of picture system.

JO: - Extras:ODE - Object Development Environment

This directory contains source code for ODE, the Object Oriented Development Environment. Please see the chapter on ODE.

JPICS: - JTools:JPics - Pictures for tutorials

These IFF pictures are used by the test programs and tutorials of the IFF and ANIM Support systems.

JRX: - JTools:JARexx - Arexx Interface

Tools for writing ARexx compatible applications. See chapter on ARexx for a description of the files.

JTX: - JTools:Textra_Dir - Text Editor

Textra is a public domain text editor written in JForth by Mike Haas. It can be integrated with JForth using ARexx. See the docs on disk and the chapter on ARexx interfacing. A subdirectory in JTX: contains scripts which can be copied to your REXX: directory for use with Textra.

JU: - JForth:Util - Utilities

This directory contains a wide variety of tools and utilities. We have divided them into two categories, general purpose Forth utilities and Amiga specific utilities. Files that we think are particularly important are marked with a single asterisk, '*'.

Note: Some of the files that were in JU: in Version 2.0 had to be moved because of disk space constraints. Look in JF:, JDEV:, and JFLT: for these files.

General Forth Utilities

.RSTACK - Dump return stack, like .S .

*** BSEARCH** - Binary search using a deferred word.

*** BSORT** - Batcher sort for sorting anything pretty quick.

CATCH - A way to trap errors and jump back to the caller.

*** CHAR-MACROS** - 'C' like character routines, ISDIGIT , TOUPPER

*** DOLINES** - Provides a simple way of processing the lines of a file using a deferred word.

*** LOCALS** - Local variables. These are handy for reducing the amount of stack manipulation in your program, ie. fewer DUPs , SWAPs and ROTs.

*** LOGTO** - Copy the output of JForth to a file. This can be used for echoing to the printer if you use PRT: as the file.

MORE-ARRAYS - Different types of arrays, 2D arrays, record arrays, etc.

MSEC - Millisecond timer, not very accurate.

MODULEFIND - More module support.

*** MULTISTANDARD** - Provides compatibility with Forth-79, FIG and Forth '83 standards. Look here for common words you expected to find in JForth.

RAM-WORDS - Handy word for using RAM: when editing large files.

*** RANDOM** - 16 bit pseudo-random number generator.

RELOAD - Reload a part of a program that uses multiple files.

SFA_BITS - defines bits in Size Field

SHOWHUNKS - Dump out the hunks in an Amiga binary file using the Disassembler. Great for exploring how things work.

SQRT - Calculate integer square roots fast.

STACKUTILS - Dynamically expanding stacks. A useful data structure.

TURNKEY - Save a JForth image in a royalty free format. These files are big so only use this as a last resort if your program won't Clone.

*** UNRAVEL** - Examine the return stack and tell you who called who. Useful for debugging.

*** VALUE** - Self fetching variable. Like a constant you can change.

Amiga Specific Utilities

*** AMIGA_EVENTS** - Support for getting IDCMP events.

*** AMIGA_GRAPH** - Library of primitive graphics routines to get you started.

*** AMIGA_MENUS** - The EZMenu system which makes it much easier to use text based Intuition pull down menus.

ASL_SUPPORT - Calls to Application Specific Library plus special support for a file and font requester. These only work under Amiga DOS 2.0.

ANSISUPPORT - More general support for ANSI code.

AUTO_REQUEST - Easy way to put up an Amiga Auto Requestor.

CIAB_RSRC.F - Calls into CIAB resource for parallel port interfacing.

COLORDUMP -

CONSOLESUPPORT - Many tools for handling Console devices and RAW windows.

DEVICE-CALLS - The special device calls, BEGINIO() and ABORTIO()

DOS-SUPPORT - Some handy DOS tools, Lock() , Examine() , ExNext().

*** DUMP_STRUCT** - Dump the contents of a structure with the member names.

ERROR_CODES - Defines some sequential error return codes.

EXAMINE - Print information about a file, eg. protection, size, date.

EXEC_LISTS - Linked list macros.

*** EXEC_SUPPORT** - Various EXEC library calls, eg. DoIO() , FindTask() , WaitPort() , etc. Also the 'C' routines that are not in a library like CreatePort() , DeletePort() , CreateExtIO() , etc.

EZALERT - Simple interface to Amiga Alert. Looks like a GURU message.

FDUMP - Dump the contents of a file interactively. Like DUMP but in a file.

FILE-TOOLS - More DOS file tools like FSIZE which gets file size. Read warning in file before using.

GADGET_SUPPORT - Tools for initializing and using Gadgets.

GETCC - Commodore approved way of getting 68000 condition code register contents.

GOTO_ERROR - Provides a jump to an ERROR: label in a colon definition.

*** GRAPH_SUPPORT** - Miscellaneous graphics calls, bitmap tools.

ICON-SUPPORT - Calls for Free Get and PutDiskObject()

POLYGON - Use Area draw routines to draw polygons, GR.TRIANGLE

SCREEN_SUPPORT - NewScreen.Setup , OpenScreen() , Bitmap>Screen, Screen>View, etc.

SERIAL - Support for RS232 serial device.

SET-ICON - Associate an action with an icon.

SPEAK - Simple interface to the narrator and translator libraries.

SPRITES - Support for hardware sprites.