For: dwight
Printed on: Fri, Jul 12, 1991 09:04:56
From book: guide
Document: disclaimer
Last saved on: Tue, Jul 2, 1991 13:28:18
Document: titlepage
Last saved on: Tue, Jul 2, 1991 13:27:31
Document: appa
Last saved on: Wed, Jun 26, 1991 10:06:02
Document: appb
Last saved on: Wed, Jun 26, 1991 10:51:09
Document: Index
Last saved on: Tue, Jul 2, 1991 15:23:23

Disclaimer

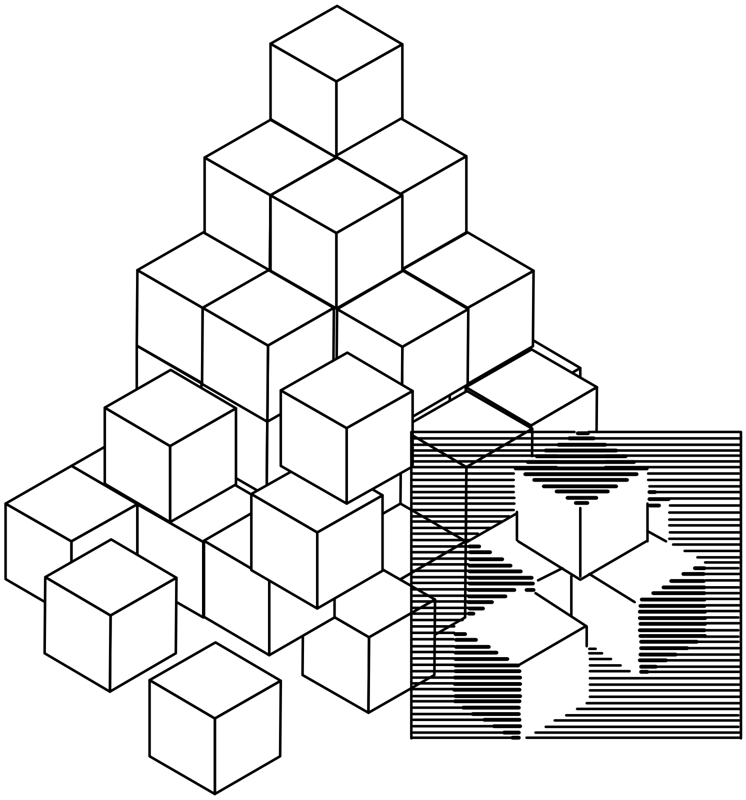
Acknowledgement

We particularly acknowledge the efforts of the Carnegie Mellon University, School of Computer Science, Warp team. This team included,Dr. H.T. Kung, his associates, and his students who provided the primary technical direction for iWarp technology.

Revision	Revision History	Date
–00A	Preliminary release of manual	7/91

Part Number 318151

Programmer's Guide



Appendix A: Quick-reference of Commands

Invoking the C Compiler: iwcc

```
iwcc [option] ... filename ...
```

iwcc Options

-#	List each phase of compilation being executed and the command options used to invoke the phase.
-##	Same as -#, but also prints the full path used to load each phase of the compilation.
-###	Same as -##, but do not execute any phases of the compilation.
-c	Suppress the link edit phase of the compilation and produce an object file for each source file.
-C	Prevent the C preprocessor from removing comments.
-D*name* [=def]	Defines *name* for the preprocessor as if by a #define directive.
-E	Run only the preprocessor on the named C programs and send the result to standard output.
-fasmwiden	Convert floating-point operands to double-precision operands for asm insert function parameters and function return types.
-fnoasmwiden	Suppress the conversion of floating-point operands to double-precision operands for asm insert function parameters and function return types.
-fnowiden	Suppress the automatic conversion of floating-point operands to double-precision operands within expressions, and suppress the conversion of floating-point parameters to functions and function return types.
-fsingle	Suppress the automatic conversion of floating-point operands to double-precision operands within expressions.
-g	Generate information needed for use of sdb.
-G*c*	Retain intermediate files from pass *c* for debugging.
-I*pathname*	Add *pathname* to the list of directories in which to search for #include files with relative pathnames (not beginning with /).
-K	Invoke PCO to generate an intra-file inlined intermediate file.
-l*x*	This option is an abbreviation for the library name lib*x*.a, where *x* is a string.
-L*dir*	Add *dir* to the list of directories containing object library routines.
-M	Merge the C and generated iWarp assembler files into a file suffixed .m.

-MM	Same as -M, but includes the text of #include files in the listing file.
-N*letter*	*letter* can be C, L, or S. C suppresses the automatic use of a default link control file. L suppresses automatic linking with libc.a (and libg.a if -g is used). S suppresses the automatic linking with iwcrt0.o (or iwmcrt0.o if -p is used).
-o *filename*	Specifies *filename* as the name of the object file, overriding the default name a.out.
-O	Invoke all available optimizers.
-p	Produce code that counts the number of times each routine is called.
-P	Run only the preprocessor on the specified C programs and leave results in the corresponding files suffixed .i.
-Q*options*	Invoke optimizations as defined by *options* , where *options* can be: G invoke global optimizations (PCO) C compaction of basic block for Compute and Access instruction S software pipelining
-S	Compile the specified C programs and leave the assembler-language output in corresponding files suffixed .s.
-u*name*	Causes the linker to enter *name* as an undefined symbol in the symbol table.
-U*name*	Remove any initial definition of *name* , where *name* is a reserved symbol predefined by the preprocessor.
-V	Display the version of the C compiler and all invoked passes, and perform compilation as specified.
-VV	Display the version of the C compiler and all possible passes, but do not perform compilation.
-Wc, *arg1* [, *arg2* ...]	Hand off any specified arguments to pass *c*, where *c* is one of the following: p preprocessor 0 compiler first pass q PCO pass 1 compiler second pass 2 PCI a assembler l link editor m listing merger
-x	Preserve only global (non-.globl) symbols in the output symbol table. Enter only external symbols.
-Yc, dir	Get pass *c* from directory *dir* , where *c* is one of the following: p preprocessor

0	compiler first pass
q	PCO pass
1	compiler second pass
2	PCI
a	assembler
l	link editor
m	listing merger
C	link.ctl
S	startup code
L	default library directory

Invoking the Assembler: iwas

`iwas [`*option*`] ...` *`filename`*

Assembler Options

-#	Print the active command line options.
-##	Print the active command line options and a message listing permitted command line options.
-dl	Remove the line number entries from the symbol table of the object file.
-L	Create a listing file with the extension .l.
-m	Use the m4 preprocessor on the source file.
-o *filename*	Specifies *filename* as the name of the object file, overriding the default name.
-O	List intermediate file information for debugging.
-rom	Prohibit data from straddling a 64-byte boundary.
-unlink	Remove the source file on completion of assembly.
-V	Write the version number of the assembler.
-VV	Write the version number of the assembler on standard error output and suppress assembling the source file.
-W	Suppress warning messages.
-XA	Enable the use of the alternate register names r0-r127 (general purpose registers) and cs000-cs3ff.
-XD	Dump the internal tables of machine-specific names.
-XN	Suppress printing of the date on listings.

Invoking the Linker: iwld

`iwld [`*option*`] ...` *`filename`* `...`

Linker Options

-e *entry*	Define *entry* as the entry point of the program.
-l*x*	This option is an abbreviation for the library name lib*x*.a, where *x* is a string.

-L*dir* Add *dir* to the list of directories containing object library routines.

-m Produce a simple load map, listing the names of the files which will be loaded plus their start addresses, input sections, output sections, and lengths.

-o *filename* Specifies *filename* as the name of the object file, overriding the default name a.out.

-r Prevent final definitions from being given to common symbols, and suppress undefined symbol diagnostics.

-s Removes the symbol table and relocation information to save space.

-u *name* Enter *name* as an undefined symbol.

-x Preserve only global symbols in the output symbol table.

Invoking the Disassembler: iwdiswas

```
diwas [option] ... filename
```

Disassembler Options

-# Print version information and executes disassembly.

-## Print version information, but does not execute disassembly.

-b Produce byte mode addresses. This is the default.

-B Suppress printing of byte mode addresses.

-c Decode the given disassembly range as instructions rather than data.

-d Dump the symbol table for debugging.

-D Produce listing style output using obsolete double word addresses.

-f Treat the input file as if the object file being disassembled had been assembled with the .flags psuedo-operation.

-H *path* Define *path* as the path to the directory where the disassembler creates its symbol and hash table.

-i Force the specified disassembly range to be initialized data rather than separate data and text sections.

-l Invoke listing mode, producing hexadecimal location and object code output. This is the default.

-L *path* Define *path* as the path to the standard symbol mnemonics (registers, condition codes).

-n Create a new hash table for symbols.

-o *filename* Specify *filename* as the name of the disassembly output file, overriding the default name.

-q Suppress listing mode, causing the output to look like assembly language source code.

-r Disable symbolization of addresses.

-R *start:end* Define a range between address *start* and address *end* for disassembly.

-s	Enable symbolization of addresses. The disassembler uses what symbol table information is available to try to replace numeric addresses by labels in instructions. This is the default.
-t	Print elapsed user and system times for the phases of disassembly.
-u	Reuse the symbol hash table constructed at the last invocation of the disassembler.
-w	Produce listing style output using obsolete word addresses.
-W	Specify that the labels stored in the symbol table are given as word addresses.
-XA	Handle unknown condition codes and control space registers by using generic names.
-XD	Compile the standard symbol names from the text output produced by the assembler into a fast binary format.

Invoking the Profiler: iwprof

`iwprof [`*option*`] ... `*filename*

Profiler Options

-a	Sort by increasing symbol address.
-c	Sort by decreasing number of calls.
-h	Suppress the heading normally printed on the report.
-m *mdata*	Use the file *mdata* instead of mon0.out as the input profile file.
-n	Sort lexically by symbol name.
-o	Print each symbol address (in octal) along with the symbol.
-s	Print a summary of the monitoring parameters and statistics on the standard error output.
-t	Sort by decreasing percentage of total time.
-V	Print iwprof version information on the standard error output.
-x	Print each symbol address (in hexadecimal) along with the symbol name.
-z	Include all symbols in the profile range, even if associated with zero number of calls and zero time.

Invoking the Archiver: iwar

`iwar `*key*` [`*posname*`] [`*option*`] ... `*afile*` [`*filename*`] ...`

Archiver Keys

d	Delete the named files from the archive file.
m	Move the named files either to the end of the archive, or the position specified by a positioning argument and posname as in the case of the r key.
p	Print the named files.

q Append the named files to the end of the archive file without searching the archive for duplicate names.

r Replace the named files in the archive.

t Print a table of contents of the archive file.

x Extract the named files.

Archiver Options

-a Place new files after *posname* .

-b Place new files before *posname* .

-c Create an archive.

-i Identical to the b option.

-l Place temporary files in the current working directory rather than in the default temporary directory, TMPDIR.

-s Force the regeneration of the archive symbol table even if the archiver is not invoked with a command that will modify the archive contents.

-u Replace those files that have changed since they were put in the archive.

-v Use verbose mode.

Invoking the Symbol Table Lister: iwnm

```
iwnm [option] ... filename ...
```

Symbol Table Lister Options

-e Print only external and static symbols.

-f Print redundant symbols (.text, .data, .lib, and .bss) normally suppressed.

-h Do not display the output header data.

-n Sort external symbols by name before they are printed.

-o Print the value and size of a symbol in octal instead of decimal.

-p Produce easily parsable, terse output.

-r Prepend the name of the object file or archive to each output line.

-T Truncate every name which would otherwise overflow its column and place an asterisk as the last character in the display name to mark it as truncated.

-u Print only undefined symbols.

-v Sort external symbols by value before they are printed.

-V Print the version of the iwnm command executing on the standard error output.

-x Print the value and size of a symbol in hexadecimal instead of decimal.

Invoking the Dumper: iwdump

```
iwdump [option] ... filename ...
```

Dumper Options

-a	Dump the archive header of each member of each archive file specified.
-c	Dump the string table.
-f	Dump the header file.
-g	Dump the global symbols in the symbol table of an archive.
-h	Dump section headers.
-l	Dump line number information.
-L	Interpret and print the contents of the .lib sections.
-o	Dump each optional header.
-r	Dump relocation information.
-s	Dump sections contents.
-t	Dump symbol table entries.
-z *name*	Dump line number entries for the named function.

Dumper Option Modifiers

-d *number*	Dump the section number, number, or the range of sections starting at number and ending the number specified by +d.
+d *number*	Dump sections in the range either beginning with first section or beginning with section specified by -d.
-n *name*	Dump information pertaining only to the named entity. This modifier can be used with -h, -s, -r, -l, -t
-p	Suppress printing of the headers.
-t *index*	Dump only the indexed symbol table entry. The -t used in conjunction with +t, specifies a range of symbol table entries.
+t *index*	Dump the symbol table entries in the range ending with the indexed entry. The range begins at the first symbol table entry or at the entry specified by the -t modifier.
-u	Underline the name of the file for emphasis
-v	Dump information in symbolic representation rather than numeric. This modifier can be used with all the above options except -s and -o options of iwdump.
-z *name, number*	Dump line number entry or range of line numbers at *number* for the function *name* .
-+z *number*	Dump line numbers starting at either function *name* or *number* specified by -z, up to *number* specified by +z.

Invoking the Sizer: iwsize

iwsize [*option*] ... *filename ...*

Sizer Options

-f	Print the size of every loaded section, followed by the section name in parentheses.

-n Include NOLOAD sections in the size.

-o Print numbers in octal.

-V Supply version information on the iwsize command.

-x Print numbers in hexadecimal.

Invoking the Monitor: iwmon

`iwmon [sibnum]`

Monitor commands

?	Lists all available commands offered by *iwmon.*
boot *filename*	Loads the monitor on each cell in the array.
ex *cellid wordaddr count*	Examine cell memory. Displays *count* number of words of memory on cell *cellid* from memory location at word address *wordaddr* .
exr *cellid wordaddr count*	Examine general-purpose registers. Displays *count* general purpose registers on cell *cellid* starting at the register with a word address of *wordaddr* .
exc *cellid wordaddr count*	Examine control space register. Displays *count* control space registers on cell *cellid* starting at the register with a word address of *wordaddr.*
dep *cellid wordaddr count data*	Deposit into cell memory. Deposits *data* on cell *cellid* in *count* memory locations starting at word address *wordaddr* .
depr *cellid wordaddr count data*	Deposit into general purpose registers. Deposits *data* on cell *cellid* in *count* general purpose registers starting at the register with a word address of *wordaddr* .
depc *cellid wordaddr count data*	Deposit into control space register. Deposits *data* on cell *cellid* in *count* control space registers starting at the register with a word address of *wordaddr* .
load *cellid filename*	Load a program into cell memory. Loads the COFF file *filename* on cell *cellid* at the location determined by the COFF information in the file.
run *cellid wordaddr*	Run a program. Starts execution of a program on cell *cellid.* The execution starts at word address *wordaddr* .
sbp *cellid wordaddr bp*	Set a breakpoint. Sets a breakpoint at word address *wordaddr* on cell *cellid.* *bp* is the breakpoint number which can later be used to unset the breakpoint.
ubp *cellid bp*	Unset a breakpoint. Unsets breakpoint *bp* on cell *cellid.*

adr	Toggle the address mode for output (word or byte).
exit	Exit the iWarp monitor.
quit	Same as exit.

Appendix B: PathLib Routines

This appendix contains copies of the manual pages describing the PathLib communication routines. The manual pages include the following:

- connections
- express
- marks
- messages
- plerror
- plinit
- spool
- streamgates

Index

Q

R

S

T