## NAME

iverilog-vpi - Compile front end for VPI modules

## SYNOPSIS

**iverilog-vpi** [--name=*name*] *sourcefile...*

## DESCRIPTION

*iverilog-vpi* is a tool to simplify the compilation of VPI modules for use with Icarus Verilog. It takes on the command line a list of C or C++ source files, and generates as output a linked VPI module. See the **vvp**(1) man page for a description of how the linked module is loaded by a simulation.

The output is named after the first source file. For example, if the first source file is named *foo.c*, the output becomes *foo.vpi*.

## OPTIONS

*iverilog-vpi* accepts the following options:

**-l***library* Include the named library in the link of the VPI module. This allows VPI modules to further reference external libraries.

**--name=***name*

Normally, the output VPI module will be named after the first source file passed to the command. This flag sets the name (without the .vpi suffix) of the output vpi module.

**--install-dir**

This flag causes the program to print the install directory for VPI modules, then exit. It is a convenience for makefiles or automated plug-in installers.

## PC-ONLY OPTIONS

The PC port of *iverilog-vpi* includes two special flags needed to support the more intractable development environment. These flags help the program locate parts that it needs.

**-mingw=***path*

Tell the program the root of the Mingw compiler tool suite. The **vvp** runtime is compiled with this compiler, and this is the compiler that *iverilog-vpi* expects to use to compile your source code. This is normally not needed, and if you do use it, it is only needed once. The compiler will save the *path* in the registry for use later.

**-ivl=***path*

Set for the use during compilation the root if the Icarus Verilog install. This is the place where you installed Icarus Verilog when you ran the installer. This flag is also only needed once, and the path is stored in the registry for future use.

## UNIX-ONLY OPTIONS

The UNIX version of *iverilog-vpi* includes additional flags to let Makefile gurus peek at the configuration of the *iverilog* installation. This way, Makefiles can be written that handle complex VPI builds natively, and without hard-coding values that depend on the system and installation. If used at all, these options must be used one at a time, and without any other options or directives.

**--cflags** Print the compiler flags (CFLAGS or CXXFLAGS) needed to compile source code destined for a VPI module.

**--ldflags** Print the linker flags (LDFLAGS) needed to link a VPI module.

**--ldlibs** Print the libraries (LDLIBS) needed to link a VPI module.

**-m32** On 64bit systems that support it (and support vvp32) this flag requests a 32bit vpi binary instead of the default 64bit binary.

Example GNU makefile that takes advantage of these flags:

```
CFLAGS = -Wall -O $(CFLAGS_$@)
VPI_CFLAGS := $(shell iverilog-vpi --cflags)
CFLAGS_messagev.o = $(VPI_CFLAGS)
CFLAGS_fifo.o = $(VPI_CFLAGS)
messagev.o fifo.o: transport.h
messagev.vpi: messagev.o fifo.o
        iverilog-vpi $^
```

## AUTHOR

Steve Williams (steve@icarus.com)

## SEE ALSO

iverilog(1), vvp(1), **<http://www.icarus.com/eda/verilog/>**, **<http://www.mingw.org>**,

## COPYRIGHT