##在 Mac 上编写 Verilog 代码

## **##前言**

本文将会介绍在 Mac 上如何编写，编译和仿真你的 Verilog 代码，来完成冯爱民老师《计算机组成原理A》课程的实验内容，我将会介绍一款免费的文本编辑器 Sublime Text ，一个自由软件[Icarus Verilog](http://iverilog.icarus.com/)，一个免费的波形文件查看软件[Scansion](http://www.logicpoet.com/scansion/)。在这里也呼吁大家力所能及的使用正版软件或者免费的自由软件。破解收费软件是不道德并且违法的行为。

## **使用 Sublime Text 编写代码**

你可能需要一个有语法高亮和更加聪明的缩进功能的文本编辑器。如果你现在没有趁手的，我推荐 Sublime Text，然后安装 Sublime Verilog 插件。按照以下步骤：

* 从 Sublime Text [官方网站](http://www.sublimetext.com/)下载安装。
* 按照[这个页面](https://sublime.wbond.net/installation) 的指导安装Package Contro，重启 Sublime Text。
* 打开 Sublime ，使用快捷键 Command+Shift+P（对Windows 和 Linux用户是Ctrl+Shift+P），一个窗口或者菜单就出现了。
* 输入 install 然后按回车。
* 输入 verilog 然后按回车。
* 现在随便打开一份 Verilog 代码源文件，可以看到已经正确的 进行了高亮，Sublime主界面的左下角也提示 Verilog 字样。

下一步，你需要在 Mac, Linux 或者 Windows上安装 Icarus Verilog 用来进行编译和仿真。

## **##安装Icarus Verilog**

首先，让我们安装软件：

* 确保你安装了 Xcode 和 Developer Tools，[参见](http://guide.macports.org/" \l "installing.xcode).(从 App store 或苹果开发者网站安装[Xcode](https://developer.apple.com/xcode/),然后安装Xcode command line tools, 在终端.app 中运行命令```xcode-select --install

- 原文推荐使用 MacPorts，但是我推荐大家选择 [Homebrew](http://brew.sh/)作为自己的包管理工具。点击上面链接按照官网说明安装，或者在终端运行下面的命令。

```

ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"

打开控制台输入

brew install icarus-verilog

使用Homebrew安装软件包时，会自动先下载软件包，然后解压安装，但有时候下载会卡住，或者很慢，这个时候你可以通过其他工具先将所需的软件包下载下来，注意版本一定要对应，下载的链接可以在 brew 在终端中的输出看到。例如

ftp://icarus.com/pub/eda/verilog/v0.9/verilog-0.9.7.tar.gz

Homebrew放置软件包源码的路径为\*/Library/Caches/Homebrew/，只要你将所需要的软件包下载正确的版本，放置在此目录下，那么再使用brew install icarus-verilog的时候, brew 就能直接安装了，注意软件包的命名，一般是name-version.suffix\*,参考以下下命令：

cp ~/Desktop/verilog-0.9.7.tar.gz /Library/Caches/Homebrew/icarus-verilog-0.9.7.tar.gz

当你安装成功，在终端运行命令iverilog可以看到如下的输出:

iverilog: no source files. Usage: iverilog [-ESvV] [-B base] [-c cmdfile|-f cmdfile] [-g1995|-g2001|-g2005] [-g] [-D macro[=defn]] [-I includedir] [-M depfile] [-m module] [-N file] [-o filename] [-p flag=value] [-s topmodule] [-t target] [-T min|typ|max] [-W class] [-y dir] [-Y suf] source\_file(s) See the man page for details.

##编译和仿真

---

- 下载[例子代码](http://www.swarthmore.edu/NatSci/mzucker1/e15\_f2014/tutorial1.zip)，解压后放在桌面，文件夹命名为tutorial1

- 在终端中打开文件夹。

```

cd Desktop/tutorial1

```

- 下一步编译例子代码，运行命令：

```

iverilog -o example\_3\_1.vvp example\_3\_1\_tb.v

```

- 成功编译后可以看到生成了example\_3\_1.vvp文件，你可以在 Finder 中查看，也可以使用命令：

>

```

ls

```

>可以看到输出

>

```

example\_3\_1.v example\_3\_1\_tb.v

example\_3\_4\_tb.v example\_3\_5\_tb.v

example\_3\_1.vvp example\_3\_4.v

example\_3\_5.v

```

- 现在就可以运行了，使用命令：

>

```

vvp example\_3\_1.vvp

```

>或者直接运行可执行文件example\_3\_1.vvp（其实是脚本文件，vvp 文件内容第一行指明了使用 vpp 解释）。

>

```

./example\_3\_1.vvp

```

>然后得到输出：

>

```

VCD info: dumpfile example\_3\_1.vcd opened for output.

done testing case 0

done testing case 1

done testing case 2

done testing case 3

done testing case 4

done testing case 5

done testing case 6

done testing case 7

```

- 在[这个页面](http://www.logicpoet.com/scansion/)下载Scansion，并安装。

- 使用 Scansion 查看仿真产生的波形文件。可以在 Finder 中双击 vcd 文件，也可以使用命令：

```

open -a Scansion example\_3\_1.vcd

```

恭喜你，你已经熟练的掌握了工具了，接下来赶快去写作业吧。

##一些更复杂的应用和我的体会

---

###如何使用多文件组织

本部分内容参考[Icarus Verilog官方 Wiki 的某个页面](http://iverilog.wikia.com/wiki/Getting\_Started)

举个例子，对于两个文件，实现了计数器 count model 在 counter.v文件

module counter(out, clk, reset);

parameter WIDTH = 8;

output [WIDTH-1 : 0] out; input clk, reset;

reg [WIDTH-1 : 0] out; wire clk, reset;

always @(posedge clk) out <= out + 1;

always @reset if (reset) assign out = 0; else deassign out;

endmodule // counter

并且 test bench model 的实现在 counter\_tb.v文件中。

module test;

/\* Make a reset that pulses once. \*/ reg reset = 0; initial begin # 17 reset = 1; # 11 reset = 0; # 29 reset = 1; # 11 reset = 0; # 100 $stop; end

/\* Make a regular pulsing clock. \*/ reg clk = 0; always #5 clk = !clk;

wire [7:0] value; counter c1 (value, clk, reset);

initial $monitor("At time %t, value = %h (%0d)", $time, value, value); endmodule // test

你当然可以使用 include 宏。这里我们介绍的是另外的方法。

iverlog 命令通过两种方式支持多文件组织。第一种是在命令的末尾列出所有的源代码文件。像这样：

% iverilog -o my\_design counter\_tb.v counter.v % vvp my\_design

提醒一下 PC 上可执行文件一般要.exe后缀，但是在 Linux 上我们习惯把有执行权限的文件命名成没有后缀的。我们更倾向于命名为my\_design，而不是my\_design.vvp

这个方法在文件很少的时候非常灵活好用，但是对于实验，以为我为例，我写了11个.v 文件，这时候就要用到第二种方法，使用命令的-c 参数。官方叫做“use a commandfile”，创建一个文本文件，列出要编译的所有文件名。对于上面的例子，我们创建一个“file\_list.txt”,每行列出一个文件名，文本文件内容如下：

counter.v

counter\_tb.v

运行下面的命令来编译和运行：

% iverilog -o my\_design -c file\_list.txt % vvp my\_design

在编译时，还可以使用 -s 参数指定稍后进行仿真的顶层模块，例如：

% iverilog -s main -o hello hello.v

###学习和调试的体会

1. 熟悉工具以前不要开始写实验内容，先写几个老师推荐的译者是夏宇闻的那本教材中的例子，例如：模16计数器，RS 数据锁存器等等。

2. 看书很重要，$dumpfile,$dumpvars,$display,$monitor,$stop,$finish一定要搞清楚，想要仿真正常结�束并且生成波形文件，一定在 test\_bench 的一个 initial块中正确的设置$dumpvars,并且在适当的周期后使用$finish。用$stop的话 vpp 会进入交互的调试的功能。

3. 使用编辑器+命令行运行仿真的方式调试要多多借助$display 和 $monitor 这两个宏。注意$monitor只有最后一次的设置会生效。

4. 我在写 mips 模块的时候，把很多模块用 wire 连起来的过程中出现了拼写错误，比如 Branch 写成 branch，这一点跟 c 语言不同，编译器是不会报错提示未定义的 wire，而是会生成空信号，我就坑了一把。其实应该早点用Scansion看看，空信号的是蓝色的线！特别明显啊！一查一个准啊！

5. 调试输出善用```[%m]```参数，高层模块引用底层模块可以使用点语法。

```

eg1 : 输出 time = 890 [testbench.\_mips.\_regfile]Register[16] <= 00007fff

eg2 : 代码 $display("Register[s%d] = %h",i-16,\_mips.\_regfile.rf[i]);

## **##结语**

本文面向的是第一次在 Mac 上配置开发环境的同学，已经熟悉命令行的同学，直接使用```brew install icarus-verilog

文章参考了

- [swarthmore.edu的一份资料](http://www.swarthmore.edu/NatSci/mzucker1/e15\_f2014/iverilog.html)

- [Icarus Verilog的官方 Wiki 页面](http://iverilog.wikia.com/wiki/Main\_Page)

本文面向使用 Mac OSX 的同学，对使用 Linux 的同学，本文和参考资料也会给你带来帮助。

如果有问题欢迎联系：李建霖 lijianlin1995@iCloud.com Wechat:lijianlin1995