2022-2023学年秋季学期

第四部分-B
组合逻辑电路高级内容
实验部分

授课团队：宋威
助　　教：马浩

实验环境和工具

- 实验环境
  - 默认环境：
    学生自己的笔记本电脑
    Windows 10，Intel 64-bit处理器
  - 其他可用环境：
    任何Linux环境
- 所需软件
  - Git-Bash：提供git支持，基本的Linux bash命令环境
  - Icarus Verilog：Verilog HDL的仿真器
  - GTKWave：VCD波形文件的查看器

默认环境下的软件安装

- 安装Git-Bash

  从 https://git-scm.com/download/win 下载
  “64-bit Git for Windows Setup”
  然后安装该软件（全部使用默认配置）

- 安装其他的工具
  - 启动Git-Bash（进入一个命令行窗口）

  在命令行中键入：

  ```
  git clone https://github.com/wsong83/verilog-demo
  ```

  或者

  ```
  git clone https://bitbucket.org/wsong83/verilog-demo
  ```

  该命令完成后，所有的必须工具和实验代码应该都被下载到：
  C:\Users\你的用户名\verilog-demo

  如果该命令执行失败，也可以下载
  https://github.com/wsong83/verilog-demo/archive/master.zip
  然后将其解压缩到对应位置。

Linux下的安装（以Ubuntu为例）

- 安装Git

  sudo apt-get install git

- 安装Icarus Verilog

  sudo apt-get install iverilog

  (如果失败，访问 https://sourceforge.net/projects/iverilog/files/iverilog/10.0/ 下载对应文件并安装)

- 安装GTKWave

  sudo apt-get install gtkwave

  (如果失败，访问 https://sourceforge.net/projects/gtkwave/files/ 下载对应文件并安装)

- 安装实验文件

  git clone https://github.com/wsong83/verilog-demo

  如果该命令执行失败，也可以下载https://github.com/wsong83/verilog-demo/archive/master.zip
  然后将其解压缩到对应位置。

文件结构

- 安装完毕之后，verilog-demo目录下应该有这些内容：

bin, include, install, lib, share	iverilog的软件目录
gtkwave	GTKWave的软件目录
src, test	实验示例目录
ex1	作业目录
experiments.sh	实验示例的编译脚本
set_env.sh	实验环境的初始化脚本

- 初始化环境（windows）

  如果使用windows，记得每次开始实验之前，在Git-Bash命令行中，进入verilog-demo文件夹，然后执行set_env.sh脚本：

  ```
  cd verilog-demo
  source set_env.sh
  ```

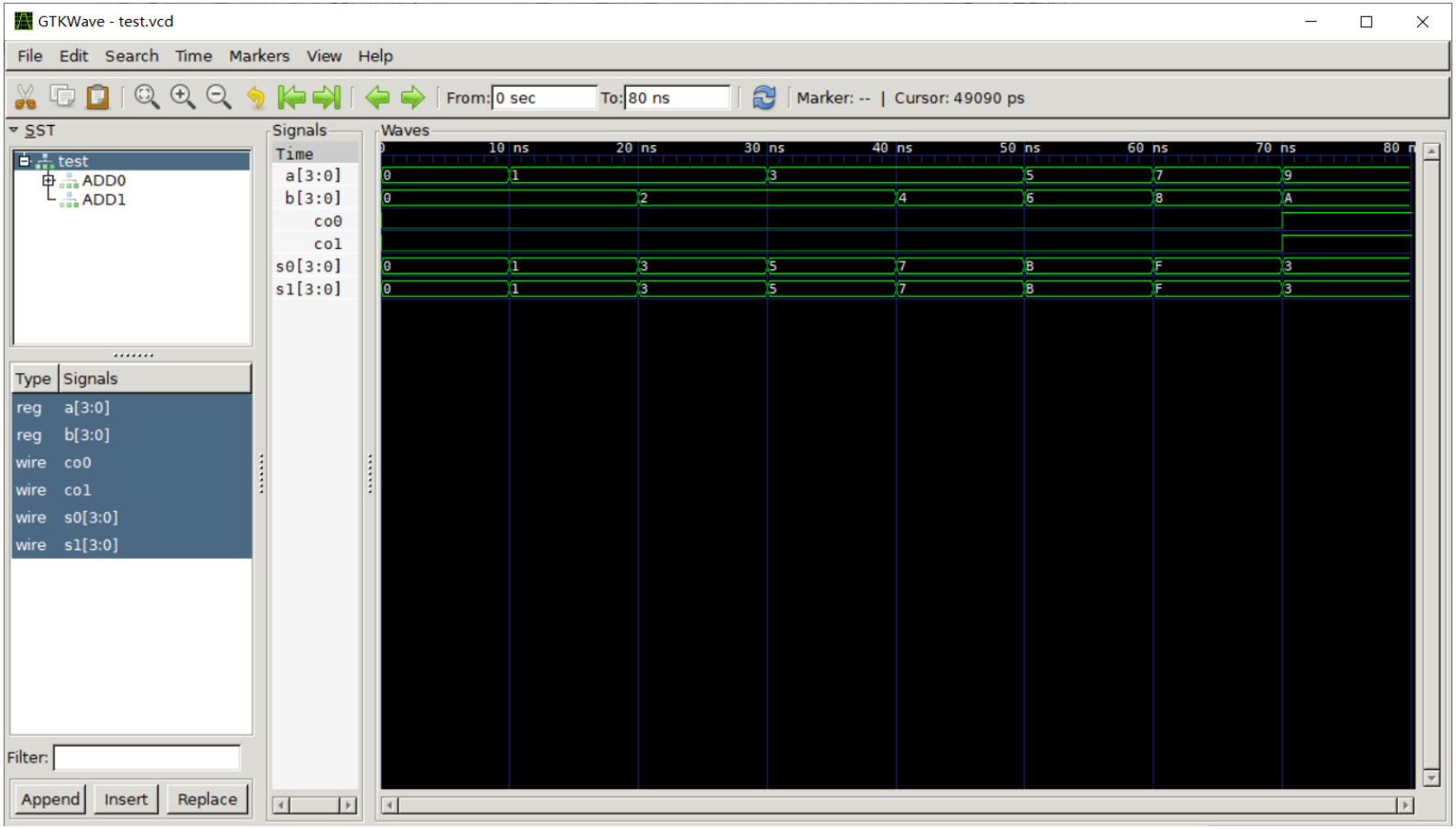
  Linux环境中不需要这一步！

实验一：加法器

- 测试串行加法器和并行加法器并比较结果

```
# 查看设计
cat src/adder.v
cat test/adder_test.v

# 编辑设计
notepad src/adder.v &

# 编译
iverilog -s test -o adder_test src/adder.v test/adder_test.v

# 仿真
vvp adder_test

# 查看波形
gtkwave test.vcd &
```

实验一：加法器测试波形


GTKWave - test.vcd
File Edit Search Time Markers View Help
From: 0 sec
To: 80 ns
Marker: -- | Cursor: 49090 ps
SST
test
ADD0
ADD1
Type Signals
reg a[3:0]
reg b[3:0]
wire co0
wire co1
wire s0[3:0]
wire s1[3:0]
Filter:
Append Insert Replace
Signals
Time
a[3:0]
b[3:0]
co0
co1
s0[3:0]
s1[3:0]
Waves
10 ns 20 ns 30 ns 40 ns 50 ns 60 ns 70 ns 80 n
0 1 3 5 7 9
0 2 4 6 8 A
0 1 3 5 7 B F 3
0 1 3 5 7 B F 3

测试加法器

```
`timescale 1ns/1ps

module test;
  reg  [3:0] a, b;
  wire [3:0] s0, s1;
  wire       co0, co1;

  add4   ADD0(a, b, s0, co0);
  adder4 ADD1(a, b, s1, co1);

  initial begin
    a = 0;
    b = 0;
    #10 a = 1;
    #10 b = 2;
    #10 a = 3;
    . . . . .
    #10;
    $finish();
  end

  initial begin
    $dumpfile("test.vcd");
    $dumpvars(0);
  end
endmodule
```

设定 时间单位/时间精度

测试顶层

测试激励

实例化被测模块

设置测试激励

输出波形文件

iverilog：编译器

```
$ iverilog -h
Usage: iverilog [-ESvV] [-B base] [-c cmdfile|-f cmdfile]
                [-g1995|-g2001|-g2005|-g2005-sv|-g2009|-g2012] [-g<feature>]
                [-D macro[=defn]] [-I includedir]
                [-M [mode=]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)

-o filename
Place output in the file filename. If no output file name is specified, iverilog
uses the default name a.out.
```

指定输出目标文件的名称，默认为a.out

```
-s topmodule
Specify the top level module to elaborate. Icarus Verilog will by default choose
modules that are not instantiated in any other modules, but sometimes that is not
sufficient, or instantiates too many modules. If the user specifies one or more
root modules with -s flags, then they will be used as root modules instead.
```

指定顶层模块的名称。Icarus Verilog会自动寻找顶层模块，一般是没有被其他模块实例化的模块。当存在多个顶层模块时，一般需要指定一个顶层模块。

```
Eg: iverilog -s test -o adder_test src/adder.v test/adder_test.v
```

vvp：仿真器

```
$ vvp -h
Usage: vvp [options] input-file [+plusargs...]
Options:
 -h              Print this help message.
 -i              Interactive mode (unbuffered stdio).
 -l file         Logfile, '-' for <stderr>
 -M path         VPI module directory
 -M -            Clear VPI module path
 -m module       Load vpi module.
 -n              Non-interactive ($stop = $finish).
 -N              Same as -n, but exit code is 1 instead of 0
 -s              $stop right away.
 -v              Verbose progress messages.
 -V              Print the version information.
```

我们暂时还不需要使用任何特殊的参数

Eg: vvp adder_test

所有预置实验

```
# adder
iverilog -o adder_test -s test src/adder.v test/adder_test.v
vvp adder_test
gtkwave test.vcd &

# comparator
iverilog -o ic74hc85_test -s test src/logic_modules.v test/ic74hc85_test.v
vvp ic74hc85_test
gtkwave test.vcd &

# multiplexer
iverilog -o ic74hc153_test -s test src/logic_modules.v test/ic74hc153_test.v
vvp ic74hc153_test
gtkwave test.vcd &

# decoder and encoder
iverilog -o ic74hc147_138_test -s test src/logic_modules.v test/ic74hc147_138_test.v
vvp ic74hc147_138_test
gtkwave test.vcd &
```

课堂实验

- 实现一个多功能计算器：该计算器有两个4位数据输入A和B，一个3位功能选择端S，和一个5位的结果输出Y。计算器的功能如下：

S的取值	Y的输出结果
0	无符号A+B
1	无符号A-B
2	有符号A+B
3	有符号A-B
4	无符号{0,0, A>B, A=B, A<B}
5	有符号{0,0, A>B, A=B, A<B}
其他	0

- 模板在ex1/calculator.v
- 请使用模板完成设计，编写测试，完成仿真并验证功能。