数字部分　实验一　Verilog 代码及网表仿真

一、实验目的

(1) 学习使用 Cadence Incisive 进行 Verilog 代码仿真；

(2) 学习使用 Cadence Incisive 进行 Verilog 网表仿真；

(3) 学习使用 Cadence Incisive 和 Integrated Metrics Center 进行代码覆盖率统计。

二、实验内容

1　环境配置

1.1 登录方法

利用 Xmanager 或 MobaXterm 等远程终端软件登录服务器，协议选择 SSH，服务器 IP 地址为 202.38.81.119，端口 2122，登录进入管理节点 mgt。

然后再利用 SSH 登录计算节点 c01n01 至 c01n14，登录时请注意避开用户较多的节点。例如，登录 c01n10 节点：

```
$ ssh -X c01n10
```

注意：Linux 命令行是大小写敏感的，大小写字母代表不同的含义，请正确书写字母。

查看各节点的在线和负载情况，可以浏览如下网址：

http://202.38.81.119/ganglia/

注意：请勿直接在管理结点 mgt 上运行程序。

1.2 建立实验目录

```
$ cd
$ mkdir vlsi
```

实验目录名 `vlsi` 也可以使用其他名称，不影响后续实验。

1.3 准备实验数据

进入实验目录：

```
$ cd ~/vlsi
```

复制实验数据至当前目录：

```
$ cp ~eda/course/219004/RTLtoGDSII_3_0.tar.gz .
```

注意：执行上述命令时，不要漏掉最后表示当前目录的句点符号。

解压实验数据至当前目录：

```
$ tar xvfz RTLtoGDSII_3_0.tar.gz
```

列出实验数据目录中的内容：

```
$ ls counter_design_database_45nm
```

提示：在命令行引用目录名或文件名时，只需输入开头若干个字母，再输入<TAB>键即

可自动补全。执行上述命令时，因为当前目录下只有实验数据目录名以字母 c 开头，所以只需要输入字母 c，即可用<TAB>键补全。

实验数据目录的内容简要说明如下。

captable	互连线寄生参数模型
constraints	设计约束，SDC 文件
Equivalence_checking	等价性检查运行目录
gate_level_simulation	门级仿真运行目录
lef	物理模型，LEF 文件
lib	时序模型，LIB 文件
physical_design	物理设计运行目录
QRC_Tech	QRC 工艺文件
rtl	RTL 级代码，Verilog 文件
simulation	代码仿真运行目录
STA	静态时序分析运行目录
synthesis	逻辑综合运行目录

2 简单计数器代码仿真

2.1 设置软件环境

设置 Cadence Incisive 152 软件环境：

```
$ setdt incisive
```

注意：上述命令中的 setdt 是实验中心自定义的脚本，不是通用命令，作用是设置软件所需的路径、环境变量等。在其他服务器运行软件时，请咨询管理员或 CAD 支持人员。

2.2 进入仿真运行目录

```
$ cd counter_design_database_45nm
$ cd simulation
```

注意：请勿在指定目录之外的位置执行实验操作，以免文件相对路径错误，或实验产生的文件相互混在一起。

当前目录下有两个 Verilog 文件。

counter.v	设计源代码
counter_test.v	测试平台 (Testbench)

2.3 仿真步骤

Incisive 仿真步骤分为以下三步：

(1) 编译(Compilation)：检查代码的语法和语义，**ncvlog**/**ncvhdl** 命令；

(2) 细化(Elaboration)：建立设计层次，连接设计内部的信号，**ncelab** 命令；

(3) 仿真(Simulation)：利用细化得到的快照，对设计进行仿真，**ncsim** 命令。

Incisive 还提供了 **irun** 命令将上述三个步骤无缝连接起来，并给出仿真结果。

以下实验过程中，请使用 **irun** 的帮助信息学习命令行中参数的含义。

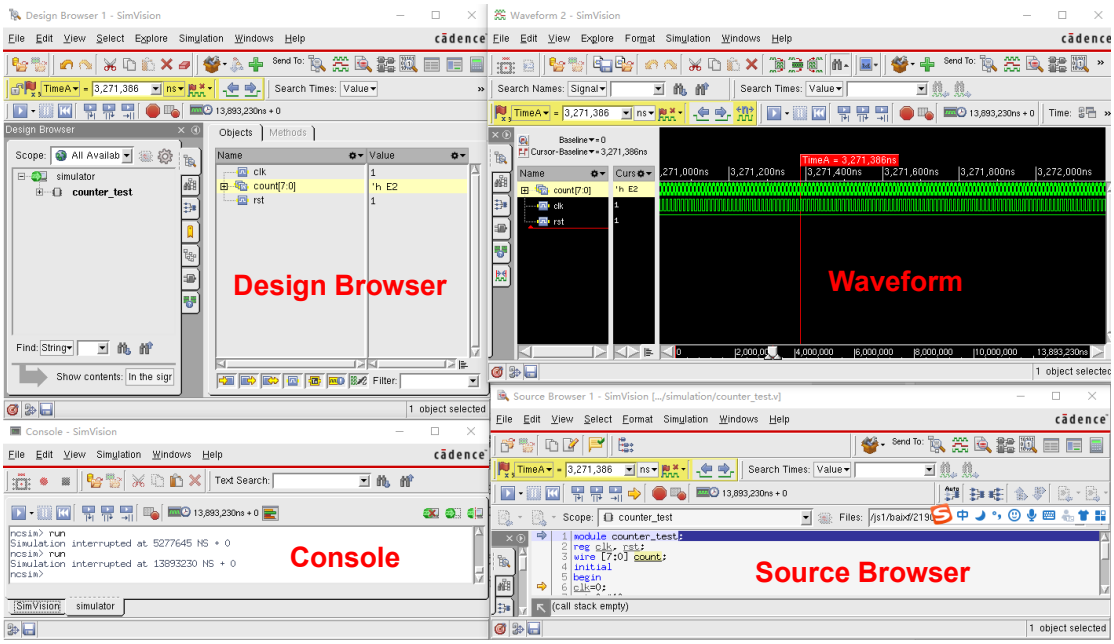
```
$ irun -help
$ irun -helpall
```

2.4 使用图形模式运行仿真

执行图形模式仿真命令：

```
$ irun counter.v counter_test.v -access +rwc -gui
```

这将打开 SimVision 工具，初始图形界面由 Design Browser 和 Console 两个窗口组成。根据需要，还可以打开 Waveform、Source Browser 等其他功能窗口。

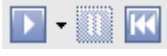


在 Console 窗口中，可以输入仿真命令，也可以通过按钮控制仿真。Run/Continue 按钮用于启动或继续仿真，若点击其右侧小箭头，则可以设置仿真时长。Interrupt 按钮用于暂停仿真。Reset 按钮用于复位仿真，重新载入细化生成的快照，仿真时间回到 0。

通过 Design Browser 窗口中部左侧的选项卡可以选择显示不同的信息，初始的状态是 Design Browser 选项卡，可以从设计的层次结构中选择任何一个实例。当选中一个实例时，右侧的 Objects 选项卡中将显示该实例中的信号名称及其当前仿真时刻的值。

在 Design Browser 选项卡中选择测试平台的顶层 **counter_test**，在右侧的 Objects 选项卡显示 **clk**, **count**, **rst** 三个信号。用鼠标选中这三个信号，点击工具条的 Send To Waveform 按钮，打开 Waveform 窗口并把这三个信号加入进去。

在 Waveform 窗口中波形区域上方工具条中，也有 Run/Continue, Interrupt, Reset 等仿真控制按钮。点击 Run/Continue 按钮启动仿真，因电路简单，仿真速度很快，仿真启动后很快即可点击 Interrupt 按钮暂停仿真。



观察波形显示区域，通过其上方工具条最右侧按钮控制显示区域和比例，也可以通过按住鼠标左键在波形区域拖曳，选择需要显示的时间范围。

点击 Design Browser 或 Waveform 窗口工具条上的 Send To Source 按钮，可以打开 Source Browser 窗口阅读源代码。若在 Design Browser 或 Waveform 窗口选中某个信号，Source Browser 窗口中的代码也会高亮显示这个信号。通过 Source Browser 窗口工具条中的 Display/Hide Values 按钮，可以打开或关闭代码中显示当前仿真时间各信号值的功能。当前仿真时间可以在 Waveform 窗口中，通过在波形上用鼠标左键点击选择，同时有红色旗标定位在选中的时间上。

通过以上功能观察波形，验证仿真结果的正确性。完成后，退出 SimVision。

SimVision 有丰富的调试功能，请在保证完成后续实验的前提下，自行探索其他菜单和按钮的功能。

2.5 使用批处理模式运行仿真

清理仿真运行目录：

```
$ irun -clean
$ rm -rf irun.* waves.shm
```

执行批处理模式仿真命令：

```
$ irun counter.v counter_test.v -access +rwc
```

得到以下运行结果：

```
irun(64): 15.20-p001: (c) Copyright 1995-2016 Cadence Design Systems, Inc.
file: counter.v
        module worklib.counter:v
                errors: 0, warnings: 0
file: counter_test.v
        module worklib.counter_test:v
                errors: 0, warnings: 0                          Compilation
                Caching library 'worklib' ....... Done
        Elaborating the design hierarchy:
        Top level design units:
                counter_test
        Building instance overlay tables: .................... Done
        Generating native compiled code:
                worklib.counter:v <0x432cbccb>
                        streams:   2, words:   491
                worklib.counter_test:v <0x17220295>
                        streams:   4, words:  1320
        Building instance specific data structures.
        Loading native compiled code:     ..................... Done
        Design hierarchy summary:
                                Instances  Unique
                Modules:                2       2
                Registers:              3       3
                Scalar wires:           2       -
                Vectored wires:         1       -
                Always blocks:          2       2
                Initial blocks:         1       1
                Pseudo assignments:     2       2               Elaboration
        Writing initial simulation snapshot: worklib.counter_test:v
Loading snapshot worklib.counter_test:v ..................... Done
ncsim> source /soft1/cadence/INCISIVE152/tools/inca/files/ncsimrc
ncsim> run                                                      Simulation
```

使用 Ctrl-C 组合键中断仿真后，可以通过命令调用 SimVision：

```
ncsim> simvision
```

后续操作与 2.4 节的 SimVision 图形界面操作相同。

在 Console 窗口或 Waveform 窗口中，点击工具条中的 Reset 按钮复位仿真，再点击  Run/Continue 按钮重新启动仿真，即可看到波形。

完成后，退出 SimVision。

3 简单计数器门级仿真

3.1 进入门级仿真目录

```
$ cd ..
$ cd gate_level_simulation
```

当前目录下有以下文件。

counter_netlist.v	逻辑综合或物理设计生成的门级网表
counter_test.v	测试平台
delays.sdf	逻辑综合或物理设计生成的延时文件
slow_vdd1v0_basicCells.v	标准单元库 Verilog 仿真模型

3.2 延时数据反标

门级仿真用于逻辑综合或物理设计之后，验证所生成的门级网表功能是否正确。

使用文本编辑器打开门级网表文件 counter_netlist.v，观察门级网表中的内容。

门级网表中描述的是标准单元实例之间的连接关系，为了完成门级仿真，还需要读入标准单元的 Verilog 仿真模型，并且将延时数据反标到设计中。使用文本编辑器打开测试平台文件 counter_test.v，注意其中与延时反标有关的部分。

```
`ifdef SDF_TEST
initial
begin
$sdf_annotate("delays.sdf",counter_test.counter1,,"sdf.log",
"MAXIMUM");
end
`endif
```

以上语句在定义了宏 SDF_TEST 的条件下，使用系统调用函数$sdf_annotate 将延时文件 delays.sdf 中的延时数据反标至网表中，仿真结果将能够反映这些延时信息。

3.3 网表仿真

网表仿真命令如下：

```
$ irun -timescale 1ns/10ps counter_netlist.v counter_test.v -v
slow_vdd1v0_basicCells.v -access +rwc -define SDF_TEST -mess -gui
```

在执行上述命令之前，请先运行 irun 帮助命令了解各参数的意义。

```
$ irun -help
```

```
$ irun -helpall
```

运行仿真命令后，可以观察到输出信息中增加了与延时反标、时序检查有关的信息。

因为网表文件中加入了扫描设计，所以在仿真时应把扫描功能关闭，否则将影响正常模式的功能仿真。在 SimVision 的 Console 窗口输入命令：

```
ncsim> force SE 0
```

在 Design Browser 窗口中选择 counter_test 模块，在 Objects 选项卡中选择 SE, clk, count, rst 等信号，点击 Send To Waveform 按钮，加入 Waveform 窗口。

在 Waveform 窗口中，点击 Run/Continue 按钮，运行仿真，得到仿真波形。

放大波形，观察因延时造成的 count 信号变化与 clk 上升沿之间的时间差，以及 count 信号各位变化不同步产生的中间状态。

完成后，退出 SimVision。

4　参数化通用计数器代码设计与仿真

4.1 准备实验数据

```
$ cd ~/vlsi
$ cp ~eda/course/219004/VLA_26_0.tar.gz .
$ tar xvfz VLA_26_0.tar.gz
$ cd VLA_26_0/lab10-cntr
```

4.2 计数器设计要求

参数化通用计数器，在实例化调用时可以设定位宽 WIDTH，时钟上升沿触发。

在时钟上升沿时：若同步复位信号 rst 为高，则计数值复位为 0；否则，若装载信号 load 为高，则计数值设置为计数器输入 cnt_in；否则，若使能信号 enab 为高，则计数值增加一。

参数和端口定义如下。

参数	WIDTH	位宽，整型，默认值为 5
输入信号	clk	时钟，上升沿触发
输入信号	rst	同步复位，高有效
输入信号	load	装载，高有效
输入信号	enab	使能，高有效
输入信号	cnt_in	宽度为 WIDTH，计数器输入
输出信号	cnt_out	宽度为 WIDTH，计数器输出

4.3 计数器代码设计

参数和端口定义代码范例如下，请按照要求完成代码设计。

```
module counter #(parameter integer WIDTH=5)(
    input  wire clk ,
```

```
    input  wire rst ,
    input  wire load,
    input  wire enab,
    input  wire [WIDTH-1:0] cnt_in,
    output reg  [WIDTH-1:0] cnt_out
    );
    ...
endmodule
```

代码编写可以使用 VIM 或 gedit 等文本编辑器，文件名为 **counter.v**。

特别提醒：若无法在实验规定时间内完成 4.3 节代码设计，可以使用实验数据中自带的范例代码完成 4.4 节实验内容，并认真阅读学习范例代码。

```
$ cd ~/vlsi/VLA_26_0/lab10-cntr
$ cp ../solutions/lab10-cntr/counter.v .
```

4.4 计数器代码仿真和调试

使用实验数据中提供的测试平台文件 **counter_test.v** 对计数器代码进行批处理模式仿真验证，若发现错误，则调用 SimVision 观察波形，定位错误位置，对代码进行修改调试，直至仿真结果正确。

使用文本编辑器打开 **counter_test.v**，阅读其中的测试代码，尝试修改其中的参数 **WIDTH** 为其他数值，并同步修改测试数据的位宽，再次进行仿真验证。

5 代码覆盖率分析

5.1 覆盖率的概念

覆盖率(Coverage)衡量测试平台对设计的检验程度，包括代码覆盖率(Code Coverage)和功能覆盖率(Functional Coverage)两种类型。

代码覆盖率用以衡量对代码的检验程度，并指示覆盖率未达标的代码区域，根据覆盖率信息，可以增加特定的仿真激励，以改善覆盖率和仿真检验效果。

代码覆盖率分为以下几种类型：块覆盖率(Block Coverage)、分支覆盖率(Branch Coverage)、语句覆盖率(Statement Coverage)、表达式覆盖率(Expression Coverage)、翻转覆盖率(Toggle Coverage)、有限状态机覆盖率(FSM Coverage)。

5.2 进入仿真运行目录

```
$ cd ~/vlsi
$ cd counter_design_database_45nm
$ cd simulation
```

5.3 运行仿真

```
$ irun counter.v counter_test.v -access +rwc -coverage all -gui
```

仿真命令中增加了`-coverage all`参数，这将在仿真过程中准备好覆盖率分析所需数据，并存放在仿真运行目录中的`cov_work`子目录中。

仿真命令带有`-gui`参数，这将自动调用 SimVision。

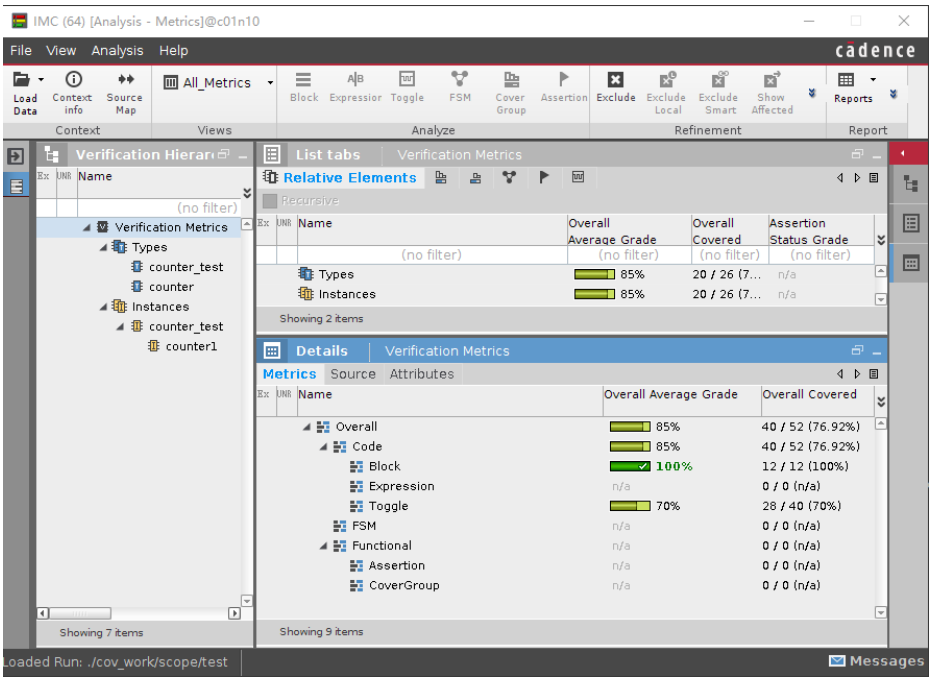
如果仿真运行目录下已经存在覆盖率数据，可以在仿真命令中增加`-covoverwrite`选项覆盖原有数据。

5.4 覆盖率分析

首先仿真 1 us，在 SimVision 的 Console 窗口输入命令：

```
ncsim> run 1us
```

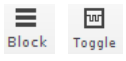
仿真完成后，点击 Coverage Analysis 图标，调用 Integrated Metrics Center (IMC)。



IMC 窗口显示内容和当前选择的覆盖率类型有关，由工具栏开始开出，窗口初始状态是 All_Metrics，即所有指标。将当前显示的覆盖率数值记录在思考题(3)表格中。

通过工具栏选择 Metrics_Block 和 Metrics_Toggle 选项，观察 IMC 窗口内容的变化。

在窗口左侧 Verification Hierarchy 中选择电路模块或实例，通过工具栏选择 Block 或 Toggle，观察详细的覆盖率数据，并分析未覆盖的情况，记录在思考题(3)表格中。



关闭 IMC 窗口。

在 SimVision 中继续运行仿真，每次 1 us，重复两次以上过程，完成思考题(3)。

关闭 SimVision 和 IMC。

三、思考题

(1) 简述对实验内容 2.5 和 4.4 所使用的测试平台的区别有何体会。

(2) 简述对 Incisive 图形模式仿真和批处理模式仿真之间的区别有何体会。

(3) 记录实验内容 5.4 中观察到的覆盖率数据，并分析未覆盖的情况。

仿真时间	1 us	2 us	3 us
Code Coverage			
Block Coverage			
Toggle Coverage			
未覆盖的情况			

要求：思考题解答以电子版形式发给数字设计助教老师，文件名为：学号+姓名。