**“电子设计自动化(EDA)”实验指导书(八)**

**一、实验课程编码：105009**

**注意**：

实验七**完全用VHDL写**，

不再用原理图的形式做顶层文件

**二、实验课程名称：电子设计自动化(EDA)**

**三、实验项目名称：**带闹钟的时分秒可调的数字钟

**四、实验目的**

深入理解硬件原理图，掌握FPGA硬件开发技能。

学会分析和设计一个功能较复杂的综合电路，数字钟，带时分秒调整,闹钟及显示。

在此过程中进一步掌握并熟练使用VHDL的各种语句。

**五、主要设备**

PC机，Quartus II软件开发平台，DE2-35实验板。

**六、实验内容**

**1**．(选做)设计一个数字钟，在上个实验的基础上，为数字钟增加闹钟功能。电路连接如图1所示(见最后一页)。从图1中，6个数码管(HEX5~HEX0)用来显示时,分,秒。Adjust 按键用来调整时,分,秒。Mode按键用来调整模式，模式的具体含义及数字钟功能要求如表1所示。Reset键用来把计数器复位到0.

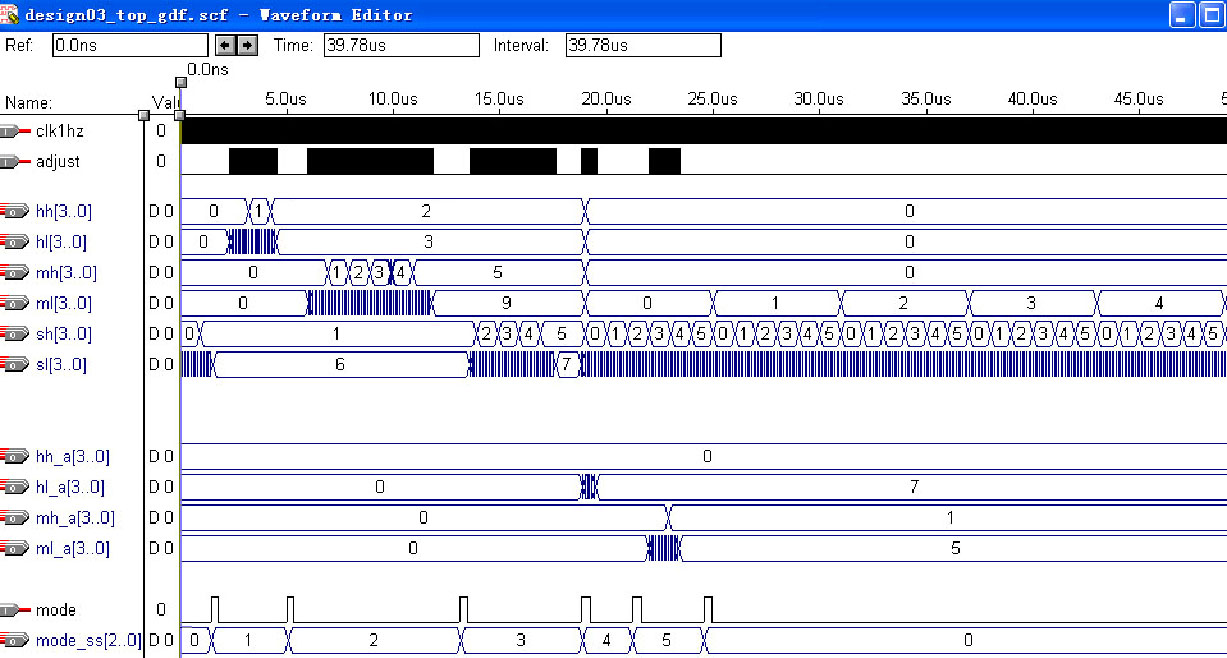
表1 数字钟功能

|  |  |
| --- | --- |
| Mode键 | 功能 |
| 按一下mode键 | 此时可调整“时”，每按一下adjust键，增加1小时，增加到23小时，再按一下adjust键，则变为00小时。调整过程中，数码管HEX5，HEX4闪烁，提示用户此时正在调整“小时”。 |
| 再按一下mode键 | 此时可调整“分”，每按一下adjust键，增加1分钟，增加到59分，再按一下adjust键，则变为00分。调整过程中，数码管HEX3，HEX2闪烁，提示用户此时正在调整“分钟”。 |
| 再按一下mode键 | 此时可调整“秒”，每按一下adjust键，增加1秒，增加到59秒，再按一下adjust键，则变为00秒。调整过程中，数码管HEX1，HEX0闪烁，提示用户此时正在调整“秒”。 |
| 再按一下mode键 | 此时可调整闹钟的“时”，每按一下adjust键，增加1小时，增加到23小时，再按一下adjust键，则变为00小时。调整过程中，数码管HEX5，HEX4用来显示闹钟的“小时”并闪烁，提示用户此时正在调整闹钟“小时”。HEX1，HEX0两个数码管（显示秒）不亮。 |
| 再按一下mode键 | 此时可调整闹钟的“分”，每按一下adjust键，增加1分钟，增加到59分，再按一下adjust键，则变为00分。调整过程中，数码管HEX3，HEX2用来显示闹钟的“分钟”并闪烁，提示用户此时正在调整闹钟“分钟”。HEX1，HEX0两个数码管（显示秒）不亮。 |
| 再按一下mode键 | 回到正常计时和显示状态 |

要求：

1. 闹钟所设置的时间到了之后，LEDG[7]绿灯亮。
2. 图1(见最后一页)增加一个复位按键 Reset, 连接到日常的counter24,counter60,counter60计数器的rst端口, 按一下Reset按键，三个计数器全部复位到零。闹钟的counter24和counter60的rst端直接连到VCC ( 固定接高电平’1’)即可, 不需要复位。

3）设计好之后，建议先仿真，如下面的图2所示(见下页)，看看是否和自己期望的功能一样。做仿真时，建议撤掉分频器，直接用clk1Hz作为总输入，否则很难仿真。仿真成功之后，再把分频器加上。(实验报告中要用自己的仿真图，不要用下图)



闹钟的“小时”

闹钟的“分钟”

内部状态机，名称不确定

图2 仿真结果(不带分频器)

思考题：

1. 整个Project 编译成功后，数字钟运行正确后，查看compilation report, 问，  
   a) 芯片EP2C35F672C6内部共有多少资源，换句话说，共有多少逻辑单元(Logic element)? 多少个寄存器(register)? 数字钟用了多少逻辑单元(Logic element)? 用了总资源的百分之几？  
   b) 数字钟用了多少寄存器(register)? 用了总资源的百分之几？

其中counter24、 counter60、fdivider、mode\_adjust 模块，各用了多少寄存器(regjister)? 可以通过Tools->Netlist viewers-> RTL viewer ,查看编译器根据VHDL文件分析综合后，合成的逻辑框图，双击各个子框图 (例如counter60模块) 数一数用了多少寄存器。然后统计各个模块所用的寄存器的总和，是否与compilation report统计的一样。

1. Project编译过程中，编译器做了哪几步重要的工作？分别是什么？可以通过仔细查看下面图3中的编译任务(Tasks)看出来，列表中列出了编译的重要步骤。

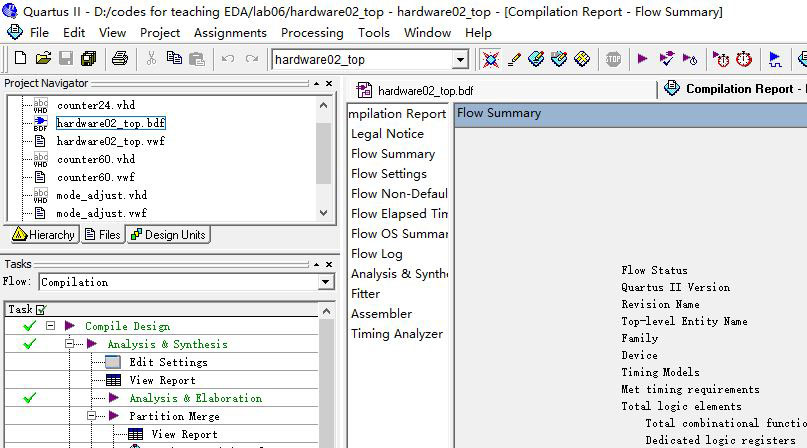


图3 查看编译报告

**2**．（必做）用VHDL语言设计一个奇偶校验器，如图4所示。有16个输入信号d15~d0, 一个输出信号y. 当检测到d15~d0中有偶数个1的时候, y输出1(表示检测到了偶数个1)。要求使用For Loop语句，要求使用变量。本题不用硬件实验箱，只要求软件仿真。

# Parity\_check

# y

# d15

# d14

# ……

# d1

# d0

图4 奇偶校验器

思考题：变量variable 和信号signal有什么区别？请说出三个区别。

4.（必做） 用VHDL设计一个数据选通器，如下面图7所示，功能表如表3所示。要求使用三种方法(三种赋值语句)设计并仿真验证。建议只建立一个project, 设计3个结构体(architecture), 用configuration语句选择不同的结构体。

方法1，Process 加 IF语句实现。

方法2，Conditional Signal Assignment （Implied 隐含的 IF statement ）

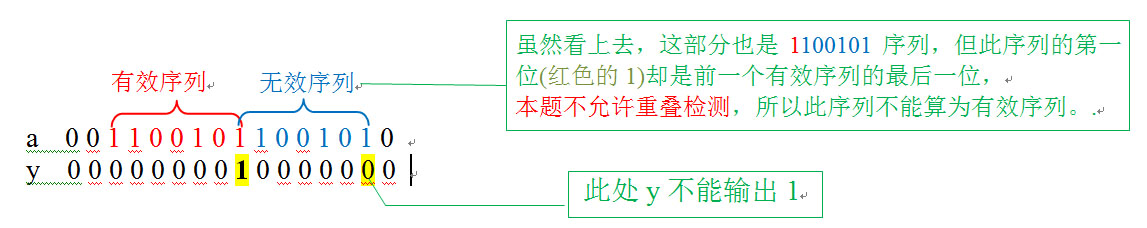
方法3. Selected Signal Assignment（Implied 隐含的 CASE statement ）

表3 数据选通器功能表

|  |  |
| --- | --- |
| **Sel [1..0]** | **y** |
| 00 | a |
| 01 | b |
| 10 | c |
| 其他 | 高阻态 |

图7. 数据选通器

1. （必做）用VHDL状态机设计一个1100101序列信号检测器，如图8所示。a为检测器的输入端口，待测的串行信号从a端口进入检测器。clk为时钟输入，y为输出信号(指示信号)。检测器的功能是：每来一个时钟，检测器检测一位输入信号，当检测到一个完整的1100101序列时，检测器就输出一个指示信号 y=1。工作原理波形图示意如下（未画时钟，每个时钟检测一位）。不允许重叠检测(关于不允许重叠检测，参见下图说明)。另外，检测器还有一组输出端 sm[3..0], 也是一组指示信号，用于指示检测器现在正处于第几个状态，这样仿真时很容易观察。本题不用硬件实验箱，只要求软件仿真。



# sm[3..0]

# check

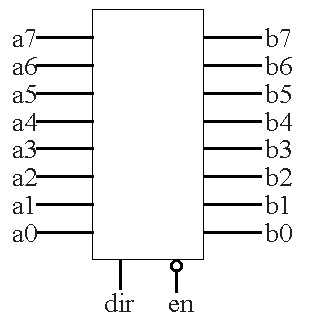
# a

# y

# clk

图8 序列信号检测器

**3.** （选做）用VHDL设计一个8bit双向总线接口，如图5所示，其功能如表2所示。

表2 8位双向总线接口

|  |  |
| --- | --- |
| 控制输入  en dir | 功能 |
| 1. 1 | a向b传输 |
| 1. 0 | b向a传输 |
| 1 x | a,b都为高阻态Z |

图5 8位双向总线接口

设计好了之后，做仿真，结果如下面的图6所示。图中，aresult是a端口上测量到信号(例如用示波器测量)，bresult是b端口上测量到信号。aresult和bresult是仿真后由仿真软件自动添加的，用户不用自己添加。用户只需要添加en, dir, a, b等输入信号及其波形。回答问题：aresult和bresult为什么会有这样的结果？

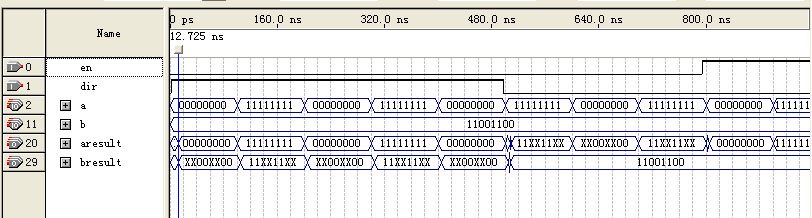


图6 8bit双向总线接口的仿真波形

**PIN\_Y18**

**FPGA**

**Cyclone II，**

**EP2C35F672C6**

**HEX0**

**HEX1**

**HEX2**

**HEX3**

**HEX4**

**HEX5**

**HEX6**

**HEX7**

**VCC**

**KEY** [**1**]

**PIN\_N23**

**PIN\_G26**

# Counter24

# Segment4-7

# HH[3..0]

# HL[3..0]

# cout

# en

# clk

# Segment4-7

**0**

**1**

**2**

**3**

**4**

**5**

**6**

0

1

2

3

4

5

6

VCC

CA

**0**

**1**

**2**

**3**

**4**

**5**

**6**

0

1

2

3

4

5

6

VCC

CA

**0**

**1**

**2**

**3**

**4**

**5**

**6**

0

1

2

3

4

5

6

VCC

CA

**0**

**1**

**2**

**3**

**4**

**5**

**6**

0

1

2

3

4

5

6

VCC

CA

**0**

**1**

**2**

**3**

**4**

**5**

**6**

0

1

2

3

4

5

6

VCC

CA

**0**

**1**

**2**

**3**

**4**

**5**

**6**

0

1

2

3

4

5

6

VCC

CA

**0**

**1**

**2**

**3**

**4**

**5**

**6**

0

1

2

3

4

5

6

VCC

CA

**0**

**1**

**2**

**3**

**4**

**5**

**6**

0

1

2

3

4

5

6

VCC

CA

**HEX7** [**0**]

**HEX7** [**1**]

161

**HEX7** [**2**]

161

**HEX7** [**3**]

161

**HEX7** [**4**]

161

**HEX7** [**5**]

161

**HEX7** [**6**]

161

**PIN\_L3**

**PIN\_L2**

**PIN\_L9**

**PIN\_L6**

**PIN\_L7**

**PIN\_P9**

**PIN\_N9**

**PIN\_R2**

**PIN\_P4**

**PIN\_P3**

**PIN\_M2**

**PIN\_M3**

**PIN\_M5**

**PIN\_M4**

**PIN\_T2**

**PIN\_P6**

**PIN\_P7**

**PIN\_T9**

**PIN\_R5**

**PIN\_R4**

**PIN\_R3**

**PIN\_U9**

**PIN\_U1**

**PIN\_U2**

**PIN\_T4**

**PIN\_R7**

**PIN\_R6**

**PIN\_T3**

**PIN\_Y23**

**PIN\_AA25**

**PIN\_AA26**

**PIN\_Y26**

**PIN\_Y25**

**PIN\_U22**

**PIN\_W24**

**PIN\_AB23**

**PIN\_V22**

**PIN\_AC25**

**PIN\_AC26**

**PIN\_AB26**

**PIN\_AB25**

**PIN\_Y24**

**PIN\_V20**

**PIN\_V21**

**PIN\_W21**

**PIN\_Y22**

**PIN\_AA24**

**PIN\_AA23**

**PIN\_AB24**

**PIN\_AF10**

**PIN\_AB12**

**PIN\_AC12**

**PIN\_AD11**

**PIN\_AE11**

**PIN\_V14**

**PIN\_V13**

**HEX6** [**0**]

**HEX6** [**1**]

161

**HEX6** [**2**]

161

**HEX6** [**3**]

161

**HEX6** [**4**]

161

**HEX6** [**5**]

161

**HEX6** [**6**]

161

**HEX5** [**0**]

**HEX5** [**1**]

161

**HEX5** [**2**]

161

**HEX5** [**3**]

161

**HEX5** [**4**]

161

**HEX5** [**5**]

161

**HEX5** [**6**]

161

**HEX4** [**0**]

**HEX4** [**1**]

161

**HEX4** [**2**]

161

**HEX4** [**3**]

161

**HEX4** [**4**]

161

**HEX4** [**5**]

161

**HEX4** [**6**]

161

**HEX3** [**0**]

**HEX3** [**1**]

161

**HEX3** [**2**]

161

**HEX3** [**3**]

161

**HEX3** [**4**]

161

**HEX3** [**5**]

161

**HEX3** [**6**]

161

**HEX2** [**0**]

**HEX2** [**1**]

161

**HEX2** [**2**]

161

**HEX2** [**3**]

161

**HEX2** [**4**]

161

**HEX2** [**5**]

161

**HEX2** [**6**]

161

**HEX1** [**0**]

**HEX1** [**1**]

161

**HEX1** [**2**]

161

**HEX1** [**3**]

161

**HEX1** [**4**]

161

**HEX1** [**5**]

161

**HEX1** [**6**]

161

**HEX0** [**0**]

**HEX0** [**1**]

161

**HEX0** [**2**]

161

**HEX0** [**3**]

161

**HEX0** [**4**]

161

**HEX0** [**5**]

161

**HEX0** [**6**]

161

# Counter60

# Segment4-7

# MH[3..0]

# ML[3..0]

# clk

# Segment4-7

# Counter60

# Segment4-7

# SH[3..0]

# SL[3..0]

# clk

# Segment4-7

**VCC**

# cout

# en

# cout

# en

**CLK50M**

**PIN\_N2**

**Mode**

**Adjust**

**Clock50MHz**

# Hour 00-23

# Minute 00-59

# Second 00-59

# Counter24

# HH[3..0]

# HL[3..0]

# cout

# en

# clk

# Counter60

# MH[3..0]

# ML[3..0]

# clk

# cout

# en

**KEY** [0]

# fdivider

# Clk1Hz

# Blink2Hz

# clkh

# clkm

# clks

# clkh\_a

# clkm\_a

# Mode\_adjust\_alarm

# MUX\_compare

# sel

# alarm

**LEDG [7]**

161

rst

rst

rst

rst

rst

**PIN\_P23**

**VCC**

**Reset**

**KEY** [2]

**VCC**

**VCC**

图1 DE2开发板 FPGA芯片的引脚连接