

# 数字系统设计作业

2018 年秋季学期

说明：这一部分中的练习题，**为要提交的作业题。**

**最晚 2018 年 12 月 28 日上传至课件 ftp。**

## 第二部分、Verilog 设计

**2.1、**设计一个 Verilog 模块，产生图 1 所示的波形。

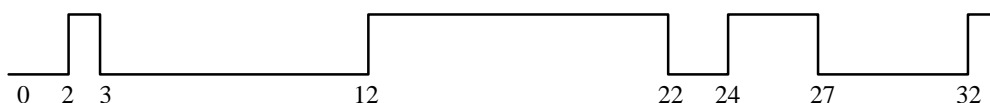


图 1、习题 1 波形图

**要求：**

- (1) 时间单位为：10ns；时间精度为：1ns；
- (2) 使用 initial 语句；模块名为：

**wavegen ()**

**2.2、**8:3 编码器的真值表如表 1 所示，使用 always—case

结构，设计一个 8:3 编码器。并设计一个测试平台，对电路进行仿真。

**要求：**

- (1) 在模块设计中要求明确标明基数格式。；
- (2) 设计模块名为：

**Encoder8x3 (code, data)**

测试平台的模块名为：

**tb\_Encoder8x3 ()**

表 1、8:3 编码器真值表

输入 data[7:0]	输出 code[2:0]
0000_0001	0
0000_0010	1
0000_0100	2
0000_1000	3
0001_0000	4
0010_0000	5
0100_0000	6
1000_0000	7

**2.3、** a) 如图 2 所示，使用 bufif0 和 bufif1 设计一个二选一路选择器，并给出测试激励模块，和仿真测试结果。

要求：

设计模块名为：

**mux2x1(dout, sel, din)**

测试平台的模块名为：

**tb\_mux2x1()**

b) 以 (a) 设计的 2 选 1 多路选择器为底层模块，通过调用 2 选 1 多路选择器模块，设计一个 4 选 1 多路选择器，并给出仿真测试结果。

要求：

设计模块名为：

**mux4x1(dout, sel, din)**

测试平台的模块名为：

**tb\_mux4x1()**

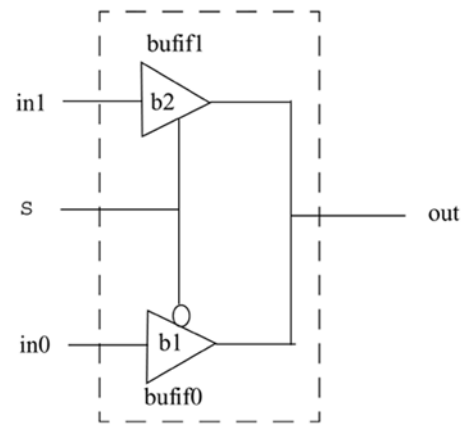


图 2、二选一路选择器

**2.4、** 设计一个 Verilog 模块，描述如图 3 所示的电路原理图表示的电路。并设计一个测试平台，对电路进行仿真。

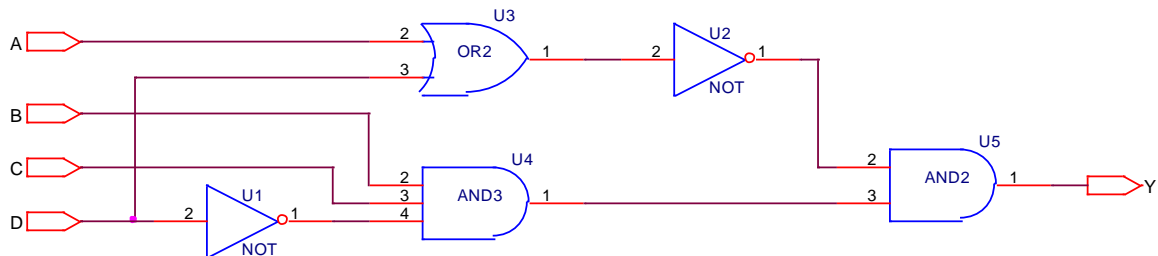


图 3、习题 4 电路原理图

要求：

(1) 利用 Verilog 的基本门（门级原语），采用结构（Structural）描述方式设计；此时，模块名为：

**comb\_str(Y, A, B, C, D)**

(2) 利用连续赋值语句，采用数据流（Dataflow）方式设计；此时，模块名为：

**comb\_dataflow(Y, A, B, C, D)**

(3) 利用 always 过程语句，采用行为和算法（Behavioral or algorithmic）方式设计；此时，模块名为：

**comb\_behavior(Y, A, B, C, D)**

(4) 利用用户定义原语（UDP），使用真值表描述方式设计；此时，模块名为：

**comb\_prim(Y, A, B, C, D)**

(5) 测试平台的模块名为：**testbench\_comb()**；注意，测试平台模块没有端口。

(6) 对电路进行全面仿真，提供仿真波形以证明设计的正确性；并使用系统任务 \$monitor 监控仿真结果，在 ModelSim 的 **Transcript Window** 中输出文本表示的仿真结果。

**2.5**、根据下面的布尔方程，设计两个组合逻辑电路模块：

i)  $Y1(A,B,C) = \Sigma m(1,2,4,5)$

ii)  $Y2(A,B,C,D) = \Sigma m(4,5,6,7,11,12,13)$

**要求：**

采用数据流（Dataflow）方式，利用连续赋值语句设计，并设计测试平台进行仿真验证。

(1) 两个设计模块名为：

**comb\_Y1(Y, A, B, C)、comb\_Y2(Y, A, B, C, D)**

测试平台的模块名分别为：

**tb\_comb\_Y1(), tb\_comb\_Y2()**

(2) 对电路进行全面仿真，提供仿真波形以证明设计的正确性；并使用系统任务\$monitor 监控仿真结果。

**2.6**、设计一个 Verilog 模块，使用 4 位输出码表示 8 位输入字中 1 的个数。并设计测试平台进行仿真验证。

**要求：**

(1) 设计模块名为：

**ones\_count(count, dat\_in)**

这里，count 是 4 位输出端口，dat\_in 为 8 位输入端口；

测试平台的模块名为：

**tb\_ones\_count()**

(2) 对电路进行全面仿真，提供仿真波形以证明设计的正确性；并使用系统任务\$monitor 监控仿真结果。

**2.7**、设计一个 10 进制计数器的 Verilog 模块。

**要求：**

(1) 计数器从 0 到 10 计数，然后返回到 0 重新开始计数；采用同步复位；

(2) 设计测试模块对其进行仿真；

(3) 设计模块名为：

**dec\_counter(count, clk, reset)**

测试平台的模块名为：

**tb\_dec\_counter()**

**2.8**、采用结构（Structural）描述方式设计一个 Verilog 模块，描述图 4 所示电路原理图表示的电路。

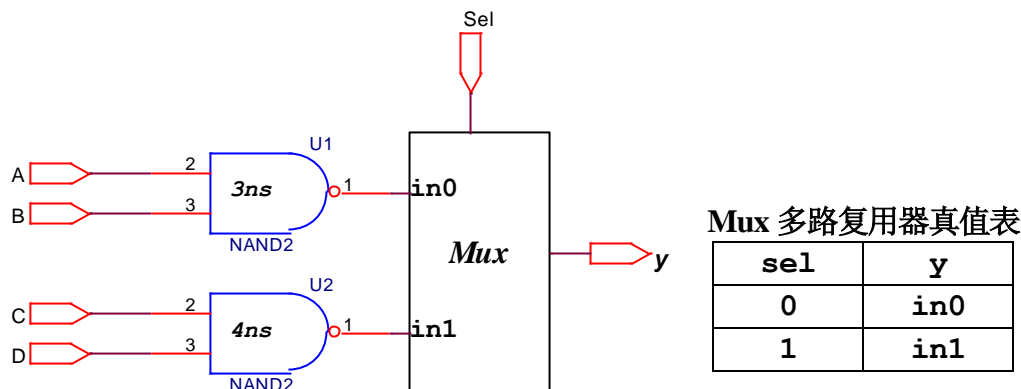


图 4、习题 8 电路原理图

**要求:**

(1) 设计模块名为:

`comb_str(y, sel, A, B, C, D)`

测试平台的模块名为:

`tb_comb_str()`

(2) 对电路进行全面仿真，提供仿真波形以证明设计的正确性；并使用系统任务\$monitor 监控仿真结果。

**2.9**、根据下面本源多项式公式，设计一个线性反馈移位寄存器，要求使用内部反馈方式设计。

$$P(x) = x^{26} + x^8 + x^7 + x + 1$$

这里:

输出: 26 位伪随机数 q

输入: clk —— 时钟信号

rst\_n —— 同步复位信号，低电平有效

load —— 加载控制，当 load=1' b1,

din —— 26 位输入，不全为 0 时，din → q

**要求:**

(1) 设计模块名为:

```
module LFSR( output reg [1:26] q, // 26 bit data output.
            input clk,           // Clock input.
            input rst_n,         // Synchronous reset input.
            input load,          // Synchronous load input.
            input [1:26] din     // 26 bit parallel data input.
            );
```

测试平台的模块名为:

`tb_LFSR()`

(2) 并给出测试模块和测试分析结果。

**2.10**、设计一个 Verilog 模块，描述如图 5 所示的电路原理图表示的电路。并设计一个测试平台，对电路进行仿真。要求：使用单一模块设计。

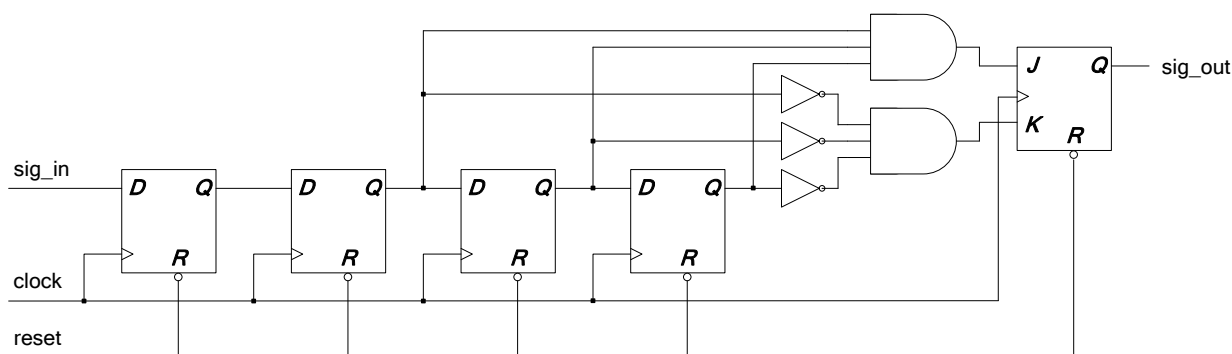


图 5、习题 10 电路原理图

要求：

(1) 设计模块名为：

**filter(sig\_out, clock, reset, sig\_in)**

测试平台的模块名为：

**tb\_filter()**

(2) 并给出测试模块和测试分析结果。

**2.11**、设计一个能够递增和递减的 8 位双向循环计数器，计数器的示意图如图 6 所示。

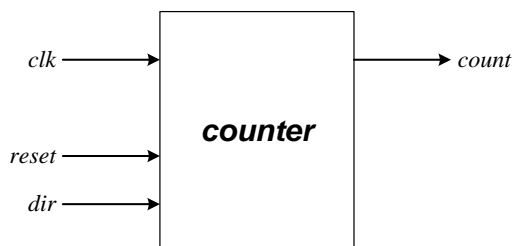


图 6、双向循环计数器

要求：

(1) 采用异步复位，复位后从第一个有效时钟的上跳沿开始计数；如果此时 **dir=1**，则递增计数，否则，递减计数。

(2) 输出 count 为 8 位；

(3) 对电路进行全面仿真。

(4) 设计模块名为：

**counter8b\_updown(count, clk, reset, dir)**

测试平台的模块名为：

**tb\_counter8b\_updown()**

**2.12**、设计一个 8 位算术逻辑单元 (ALU)，该单元的输入为操作数 a 和 b，以及操作码 oper，输出为 { c\_out, sum }，并且具有如下表所示的功能。

操作码	功能
and	$a + b + c\_in$
subtract	$a + \sim b + c\_in$
subtract_a	$b + \sim a + \sim c\_in$
or_ab	{ 1'b0, $a \mid b$ }
and_ab	{ 1'b0, $a \& b$ }
not_ab	{ 1'b0, $(\sim a) \& b$ }
exor	{ 1'b0, $a \wedge b$ }
exnor	{ 1'b0, $a \sim \wedge b$ }

- 要求:
- (1) 使用 always-case 结构设计 ALU 模块，并设计测试模块对其进行仿真验证;
  - (2) 设计模块名为:  
`ALU( c_out, sum, oper, a, b, c_in )`  
 测试平台的模块名为:  
`tb_ALU()`

**2.13**、设计一个具有图 7 所示功能的计数器模块。

- 要求:
- (1) 采用同步复位，复位后回到初始状态，然后从一个有效时钟的上跳沿开始计数;
  - (2) 设计测试模块对其进行仿真验证;
  - (2) 设计模块名为:  
`shift_counter( count, clk, reset )`  
 测试平台的模块名为:  
`tb_shift_counter()`

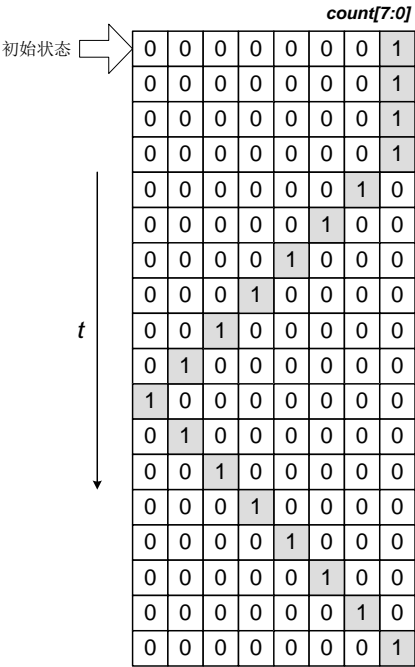


图 7、习题 13 计数器计数模式

**2.14**、图 8 是一个  $256 \times 8\text{bits}$  的 SRAM 的框图， $\text{din}[7:0]$  是 8 条数据输入线， $\text{dout}[7:0]$  是 8 条数据输出线。 $\text{wr}$  为写控制线， $\text{rd}$  为读控制线， $\text{cs}$  为片选控制线。其工作方式为：

- (1) 当  $\text{cs} = 1$ ， $\text{wr}$  信号由低变高（上升沿）时， $\text{din}$  上的数据将写入由  $\text{addr}$  所指定的存储单元；
- (2) 当  $\text{cs} = 1$ ， $\text{rd} = 0$  时，由  $\text{addr}$  所指定的存储单元的内容将从  $\text{dout}$  的数据线上输出。

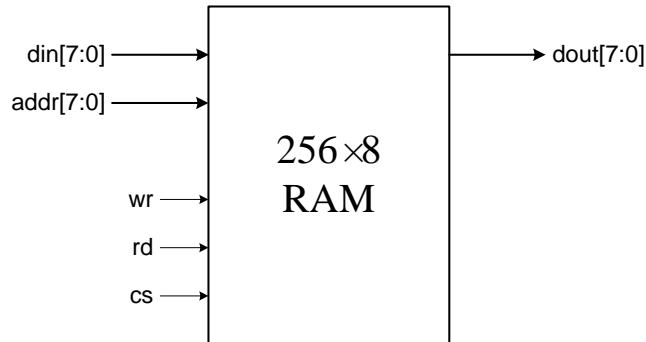


图 8、 $256 \times 8\text{bits}$  的 SRAM 框图

**要求：**

- (1) 设计一个 Verilog 模块描述一个  $256 \times 8\text{bits}$  的 SRAM。
- (2) 设计测试模块对其进行仿真验证；
- (3) 设计模块名为：

**sram( dout, din, addr, wr, rd, cs )**

测试平台的模块名为：

**tb\_sram()**

**2.15**、设计一个序列检测器，在时钟的每个下降沿检查数据。当检测到输入序列  $\text{din}$  中出现 1101 或 0110 时，输出  $\text{flag}$  为 1，否则输出为 0。

**要求：**

- (1) 画出序列检测器的状态转移图，根据状态转移图设计一个 Verilog 模块描述序列检测器。
- (2) 下面分别是设计模块和仿真测试模块。这里：

**flag**: 输出端口；当检测到指定序列时，输出为 1，否则，输出为 0；

**din**: 串行序列输入；

**clk**: 时钟信号输入；在时钟的每个下降沿检查数据

**rst\_n**: 同步复位信号，低电平有效，复位时， $\text{flag}$  输出为 0。

设计模块为：

**seq\_detect( output reg flag, input din, clk, rst\_n )**

测试平台的模块名为：

**tb\_seq\_detect()**

**2.16**、设计一个能在串行输入比特流中检测到序列 10101010 的状态机。这里，输入序列的到达方式为最低有效位 (LSB) 先到，即：第一个 0 先到，然后是 1，接着，又是第二个 0，...，等等。当检测到输入序列 `din` 中出现 10101010 时，输出 `Ready` 为 1，否则输出为 0。

**要求：**

(1) 分别采用 Mealy 和 Moore 型状态机设计，首先，画出相应类型状态机的状态转移图，然后，根据状态转移图设计 Verilog 模块。

(2) 下面分别是设计模块和仿真测试模块。这里：

`Ready`：输出端口；当检测到指定序列时，输出为 1，否则，输出为 0；

`din`：串行序列输入；

`clk`：时钟信号输入；在时钟的上升沿检查数据

`rst`：异步复位信号，高电平有效，复位时，`Ready` 输出为 0。

设计模块为：

(a) `mealy( output reg flag, input din, clk, rst )`

(b) `moore( output reg flag, input din, clk, rst )`

测试平台的模块名为：

`top()`