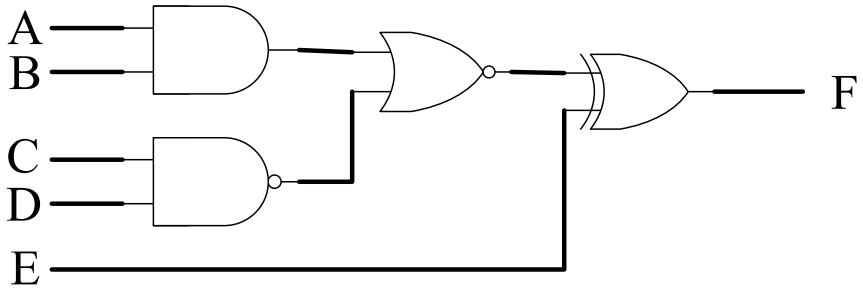
Verilog习题课

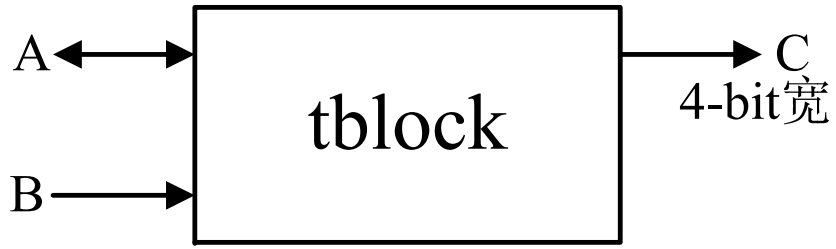
习题

- 将题中的连续赋值语句补充完整以实现对应电路的逻辑功能




```
assign  F= E ^ ( ! ((A & B) | ! (C & D) ) )
```

- 根据图中输入输出关系将Verilog模块定义补充完整，其中信号C包含4个bit宽度，其余信号为1bit宽度




```
module  tblock( A, B, C  ) ;
 inout  A;
 input  B;
 output  [3:0]  C;
……        //省略了功能描述
endmodule  //模块结束
```

习题

- 将下列测试模块补充完整，使其在仿真时输出时钟的频率为1MHz

```
`timescale 100ns/1ns
module test;
reg        clk;      //变量定义
initial    clk=0;  //初始化

always  #5   clk =~clk   //产生时钟

endmodule
```

- 你所知道的可编程逻辑器件有（至少两种）：FPGA CPLD PAL GAL
- 现有的两种主要的硬件描述语言是 Verilog HDL和 VHDL

习题

- 向空格中填入合适的内容，使P1中的Depth的值变为11

```
module Pipe (IP,OP);
parameter Option=1;
parameter Depth=1;
   . . .
endmodule
module t;
        Pipe P1 (IP1,OP1);
        defparam P1.Depth = 11;
endmodule
```

- 如下程序代码，V的8位数分别为 8'bxxxxxxxx 和 8'b0000001x

```
reg [7:0] V;
initial
begin
    V=8'bx;
    V=8'b1x;
end
```

习题

- 下面两段程序哪一段为阻塞赋值,哪一段为非阻塞赋值，画出对应的综合结果的电路图。

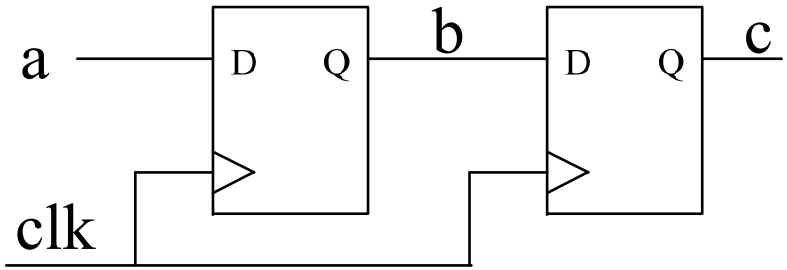
第一段：

```
always @(posedge clk)
begin
    b<=a;
    c<=b;
end
```

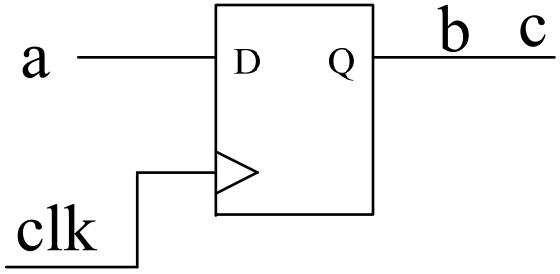
第二段：

```
always @(posedge clk)
begin
    b=a;
    c=b;
end
```

非阻塞赋值

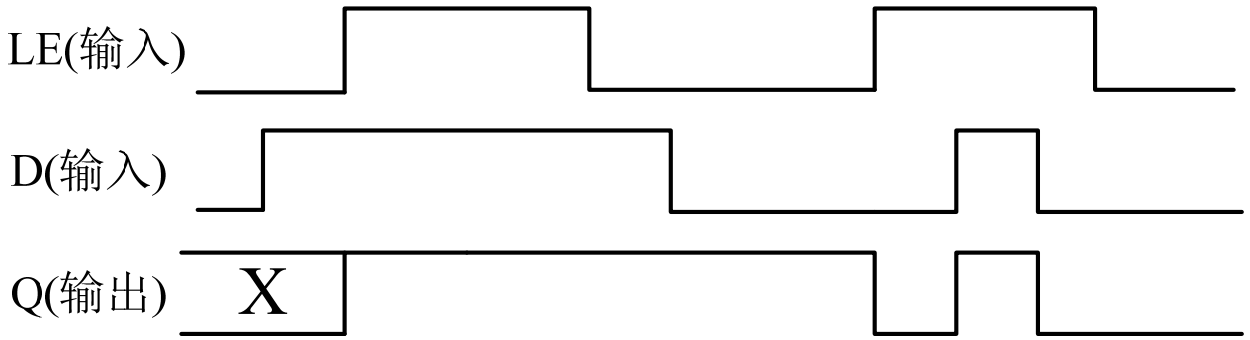


阻塞赋值



习题

- 根据时序图设计与之功能匹配的电路模块（透明锁存器）

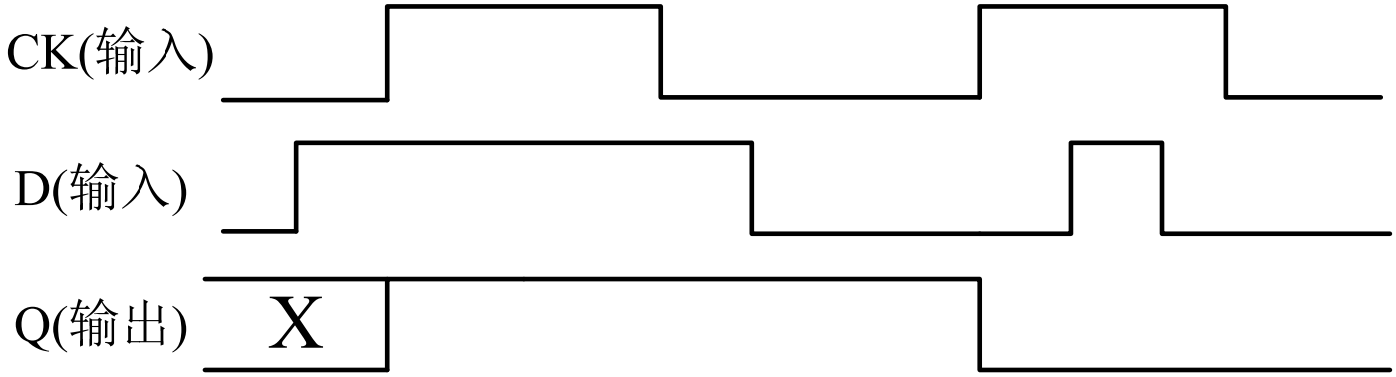


- 编程处：

```
module latch(LE, D, Q);
input LE, D;
output  Q;
reg Q;
always @(LE or D)
if(le)    Q = D;
else      Q = Q;
endmodule
```

习题

- 根据时序图设计与之功能匹配的电路模块（边沿锁存器）



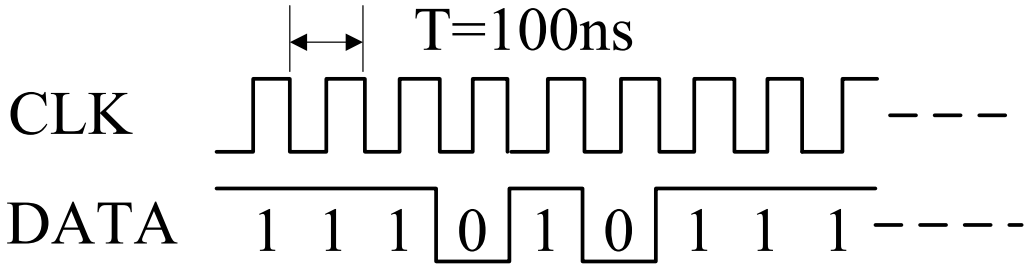
- 编程处：

```
module DFF(CK, D, Q);
input CK, D;
output  Q;

reg Q;
always @(posedge CK)
Q <= D;
endmodule
```

习题

- 使用Verilog HDL描述图中所示波形（在仿真工具中仿真可生成如下波形）。数据流中的数据共有256字节（256X8 Bits），每个时钟周期产生1bit，字节的高有效位(bit7)先产生；数据的第0字节为8’hEB，第1字节为8’h90，以后依次是8’h02, 8’h03….8’hFF（即数据内容与字节序号相同）：



  - 编程处：

习题

```
`timescale 1ns/1ns
module datagen;
reg CLK,rstn;
initial begin
    CLK = 1'b0;
     rstn = 1'b0;
# 200;   rstn = 1'b1;
end
always #50 CLK = ~ CLK;

reg [2:0] bit_cnt;
always @ (negedge CLK)
if(!rstn) bit_cnt <= 1'b0;
else bit_cnt <= bit_cnt + 1'b1;

reg [7:0] byte_cnt;
always @ (negedge CLK)
if(!rstn) byte_cnt <= 1'b0;
else if(bit_cnt == 3'd7)
    byte_cnt <= byte_cnt + 1'b1;

reg [2:0] data_byte;
always @ (negedge CLK)
if(!rstn) data_byte <= 8'hEB;
else if(byte_cnt == 2'd0)
     data_byte <= 8'hEB;
else if(byte_cnt == 2'd1)
     data_byte <= 8'h90;
else
     data_byte <= byte_cnt;

wire [2:0] cnt;
assign cnt = 3'd7 - bit_cnt;
wire DATA;
assign DATA = data_byte[cnt];
endmodule
```

习题

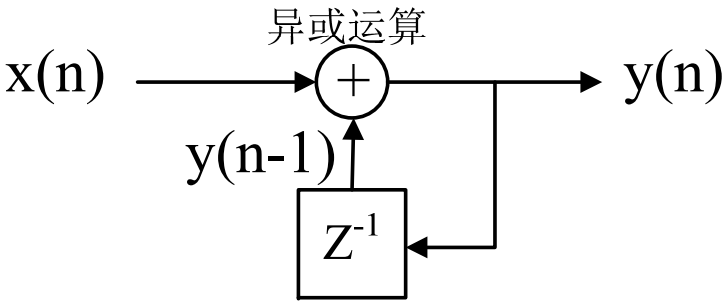
- 使用Verilog HDL设计一个3—8译码电路。输入in为3-bit信号，输出out为8-bit信号，当输入信号in为i时，使输出的第i位为1，其余位为0。（要求使用可综合风格的描述）

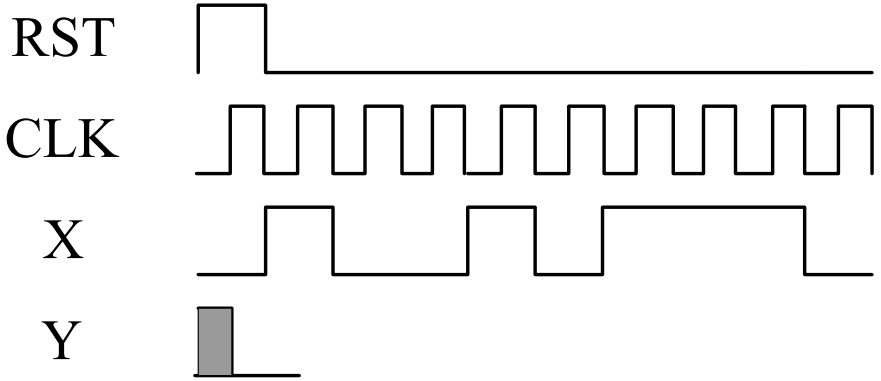
  - 编程处：

```
module decoder(in,out)
wire   [7:0] out;
assign out = 1'b1 << in;
endmodule
```

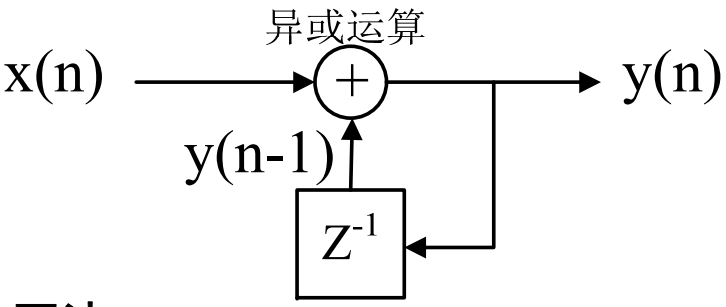
习题

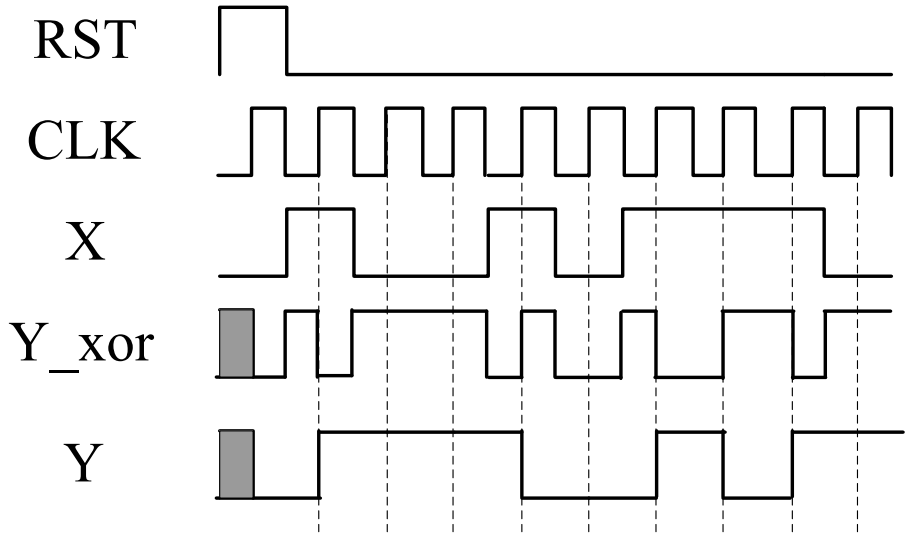
- 使用Verilog HDL 设计如下码型变换器（从非归零码(NRZ-L)至非归零传号码(NRZ-M)）。并将图中输出信号的波形补充完整。输入信号RST(复位信号)，CLK（时钟信号，与数据X同步），X（输入待处理信号），Y（输出）。（异或运算：两个输入相同则输出0，不同则输出1）。注：复位之前输出信号的状态即波形图中黑色区域为未知状态）。（要求使用可综合风格的描述）





习题





写法一

```
module code(clk, rst, x, y) ;
input clk, rst, x;
output y;
reg y;
always @ (posedge clk)
if(rst)    y<= 1'b0;
else       y <= y_xor;

wire       y_xor;
assign  y_xor = x^y;
endmodule
```

写法二

```
module code(clk, rst, x, y) ;
input clk, rst, x;
output y;
reg y;
always @ (posedge clk)
if(rst)    y<= 1'b0;
else       y <= x^y;
endmodule
```

注意：图中的Y应当为y(n-1)，不是y(n)，右图中的符号应当在左图中标出，减少误解

习题

- 设计一个10分频电路，使输出信号的频率为输入信号频率的1/10，且占空比为1：1（高低电平各占50%）。（输入信号为rst，clk,输出为out）

◆ 编程处：

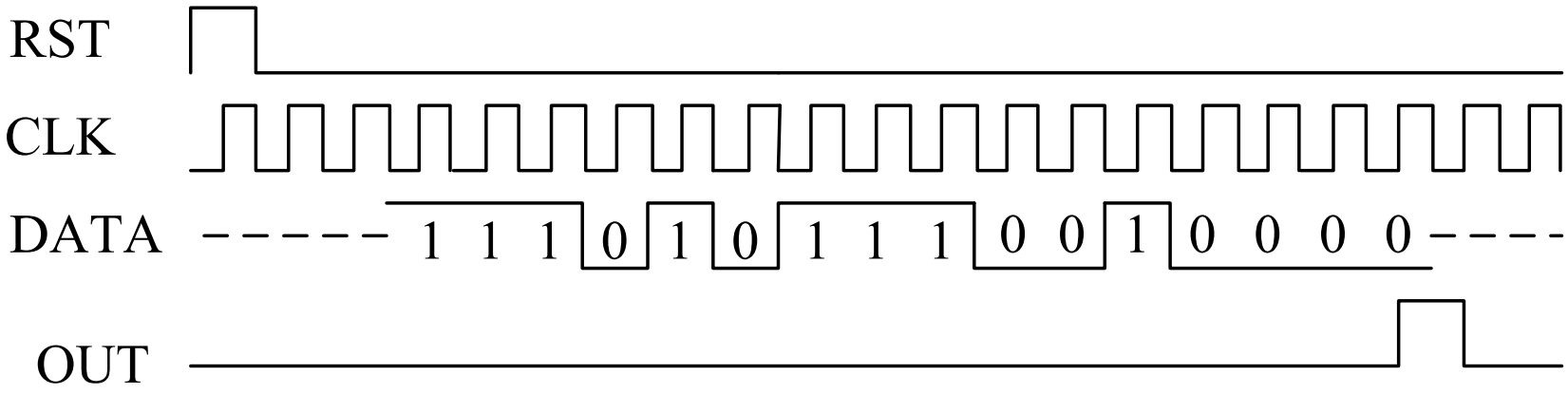
```
module Freq_gen(clk,rstn, clk_out);
input clk, rstn;
output clk_out;

reg [2:0] clk_cnt;
always @ (posedge clk)
if(!rstn)
    clk_cnt <= 1'b0;
else if(clk_cnt == 4'd4)
    clk_cnt <= 1'b0;
else
    clk_cnt <= clk_cnt + 1'b1;
```

```
reg  clk_out;
always @ (posedge CLK)
if(!rstn)
    clk_out <= 1'b0;
else if(clk_cnt == 3'd4 )
    clk_out <= ~clk_out;
```

习题

- 设计一个序列检测电路，当数据流中有’b1110_1011_1001_0000这一特殊序列时输出一个时钟周期的高电平。如图所示：



◆ 编程处：

```
module seq_det(rst, clk ,data, out);
input rst, clk, data;
output out;
reg  [15:0] data_reg;
always @(posedge clk)
if (rstn)  data_reg <= 5’d0;
else  data_reg <= {data_reg[14:0], data};
wire out;
assign out = (data_reg
== 16’heb90);

endmodule
```