2021《FPGA 应用实验》实验报告

实验编号：实验二　　实验时间：2021.3.23

实验名称：利用旋转编码器控制 8 个发光二极管

班级：F1803603　学号：518021910534　姓名：尹俊同

1、实验平台

采用 Xilinx 公司的 FPGA 集成开发环境 Xilinx ISE Design Suite 10.1 sp3，实验开发板为 Xilinx Spartan-3E FPGA Starter Kit。

2、实验设计要求：

利用在 Spartan - 3E FPGA Starter Kit Board 上的旋转开关（Rotary Push Button Switch）编码控制开发板上的 8 个发光二极管（LED7 ～ LED0）。

（0）滑杆开关（SW3）为显示模式控制开关，

（1）当 SW3 设置为（0）时，只有一个 LED 点亮：

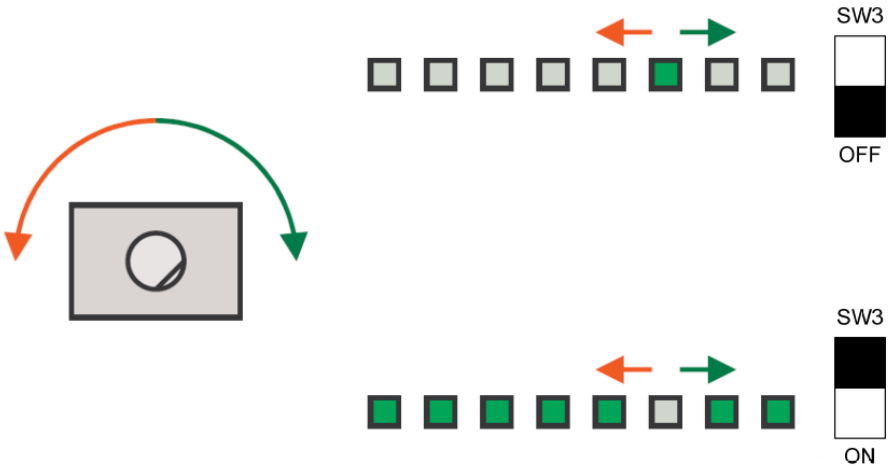
a）向右旋转 Rotary Switch 时，点亮的 LED 从右向左移动，即，旋转一步旋转开关，当前点亮 LED 关闭，其左边的 LED 点亮，当 LED7 点亮时，关闭 LED7 后，再从 LED0 开始。

b）向左旋转 Rotary Switch 时，点亮的 LED 从左向右移动，即，旋转一步旋转开关，当前点亮 LED 关闭，其右边的 LED 点亮，当 LED0 点亮时，关闭 LED0 后，再从 LED7 开始。

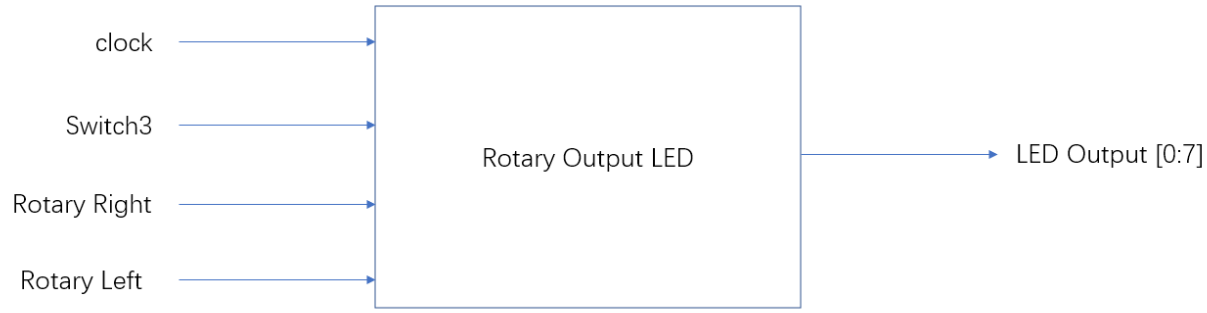
（2）当 SW3 设置为（1）时，只有一个 LED 关闭：

关闭的 LED 移动方式与（1）相同。

旋转开关控制 8 个发光二极管点亮与关闭示意图如下图所示：



3、模块设计框图



4、实验原理：

1、LED 原理

Spartan-3e FPGA Starter Kit 板有 8 个独立的表面安装 led 位于滑动开关上方，如图 2-10 所示。led 标记为 LED0 到 LED7。LED7 是最左边的 LED, LED0 是最右边的 LED。

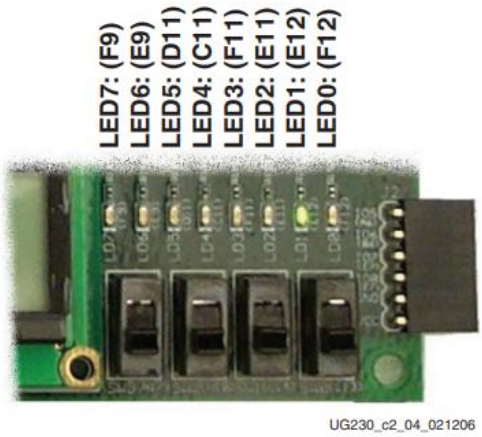


Figure 2-10: **Eight Discrete LEDs**

操作：每个 LED 的一端连接到地，另一端通过 390 欧限流电阻连接到 Spartan-3e 设备上的引脚。为了点亮单个 LED，需驱动相关的 FPGA 控制信号为高。

UCF 位置约束：图 2-11 提供了四个按钮开关的 UCF 约束，包括 I/O 引脚分配，使用的 I/O 标准，输出回转率和输出驱动电流。

```
NET "LED<7>" LOC = "F9" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<6>" LOC = "E9" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<5>" LOC = "D11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<4>" LOC = "C11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<3>" LOC = "F11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<2>" LOC = "E11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<1>" LOC = "E12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED<0>" LOC = "F12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
```

Figure 2-11: **UCF Constraints for Eight Discrete LEDs**

2、50MHz 时钟

如图 3-1 所示，Spartan-3E FPGA Starter Kit 单板支持三个主时钟输入源，它们都位于 Xilinx 标识的下方，靠近 spartan-3e 标识。该板包括一个板载 50 MHz 时钟振荡器。时钟可以通过 SMA 风格的连接器提供。FPGA 也可以通过 SMA 风格的连接器产生时钟信号或其他高速信号。可选安装一个单独的 8 针 DIP 风格时钟振荡器在提供的插座。

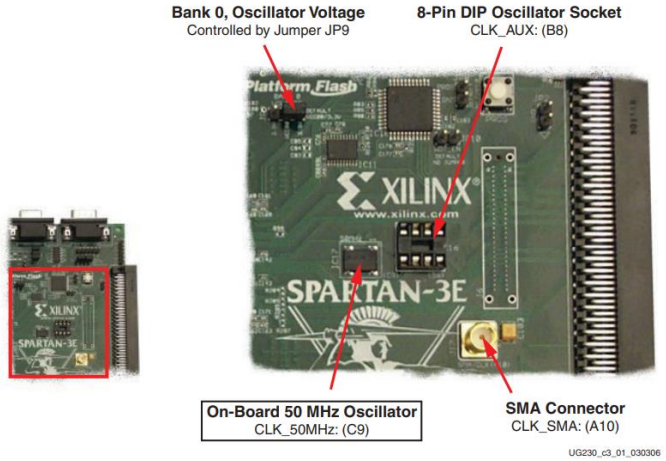


Figure 3-1: **Available Clock Inputs**

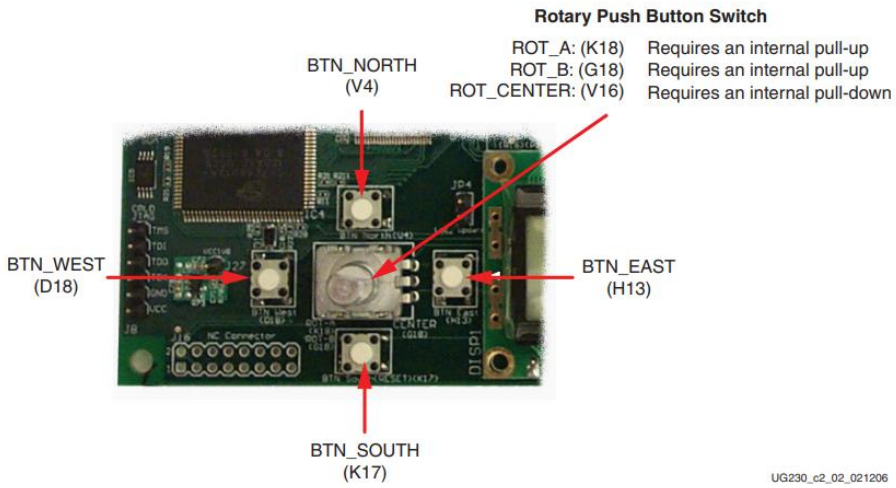
50MHz 时钟振荡器的 UCF 约束如下图。

```
NET "CLK_50MHZ" LOC = "C9"  | IOSTANDARD = LVCMOS33 ;
NET "CLK_SMA"   LOC = "A10" | IOSTANDARD = LVCMOS33 ;
NET "CLK_AUX"   LOC = "B8"  | IOSTANDARD = LVCMOS33 ;
```

Figure 3-2: **UCF Location Constraints for Clock Sources**

3、旋转按钮开关

位置和标签：旋转按钮开关位于四个独立按钮开关的中心位置，如图 2-3 所示。交换机产生三个输出。两轴编码器输出为 ROT A 和 ROT b，中心按钮开关为 ROT center。



操作：旋转按钮开关集成了两种不同的功能。开关轴旋转和输出值每当轴转动。轴也可以按下，作为一个按钮开关。按钮开关按下旋转/按钮开关上的旋钮，将相应的 FPGA 引脚连接到 3.3V，如图 2-6 所示。使用 FPGA 引脚内部的下拉电阻来产生逻辑低。图 2-9 显示了如何在 UCF 内指定一个下拉电阻。

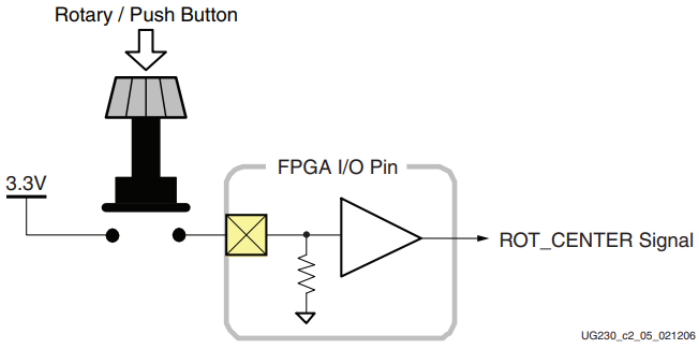


Figure 2-6: **Push-Button Switches Require Internal Pull-up Resistor in FPGA Input Pin**

从原理上讲，转轴编码器的行为很像一个凸轮，连接到中心轴转动转轴，然后运行两个按钮开关，如图 2-7 所示根据转轴旋转的方向，一个开关先于另一个打开。同样地，随着旋转的继续，一个开关在另一个之前关闭。然而，当轴是静止的，也称为止动位置，两个开关都是关闭的。

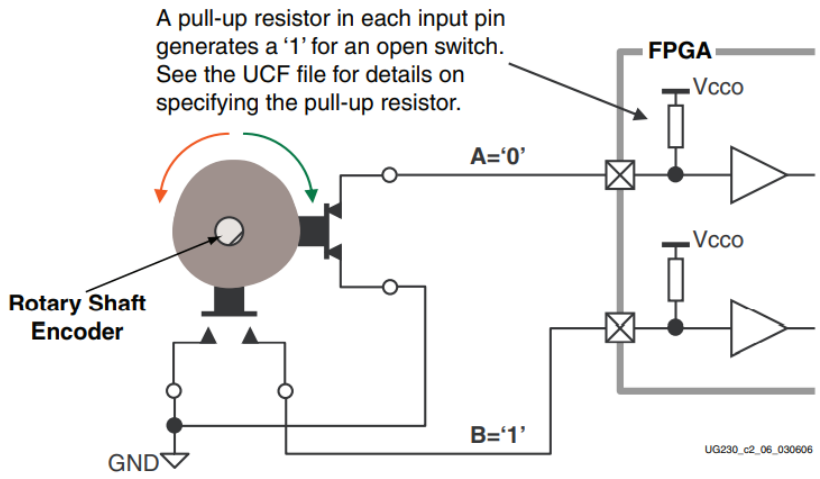


Figure 2-7: **Basic example of rotary shaft encoder circuitry**

关闭开关将其与地连接，产生逻辑低位。当开关打开时，FPGA 引脚内的上拉电阻将信号拉至逻辑高。图 2-9 UCF 约束描述了上拉电阻的定义。FPGA 解码'A'和'B'输入的电路很简单，但必须考虑输入上的机械开关噪声，也称为颤振。如图 2-8 所示，颤振可以错误地指示额外的旋转事件，甚至指示相反的旋转。

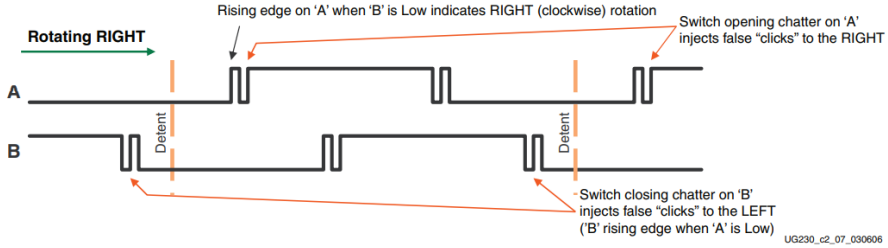


Figure 2-8: **Outputs from Rotary Shaft Encoder May Include Mechanical Chatter**

```
NET "ROT_A"      LOC = "K18" | IOSTANDARD = LVTTL | PULLUP   ;
NET "ROT_B"      LOC = "G18" | IOSTANDARD = LVTTL | PULLUP   ;
NET "ROT_CENTER" LOC = "V16" | IOSTANDARD = LVTTL | PULLDOWN ;
```

Figure 2-9: **UCF Constraints for Rotary Push-Button Switch**

5、Verilog 模块设计

```
//rotary_out.v
module rotary_out (output [7:0] ledout, input rotary_a, rotary_b, sw, clk);
   reg [1:0] rotary_in;
   reg rotary_q1, rotary_q2;
   reg delay_rotary_q1;
   reg rotary_event, rotary_left;
   reg [7:0] LED = 8'b0000_0001;

   always @(posedge clk) begin : rotary_filter
      // concatinate rotary input signals to form vector for case construct.
```

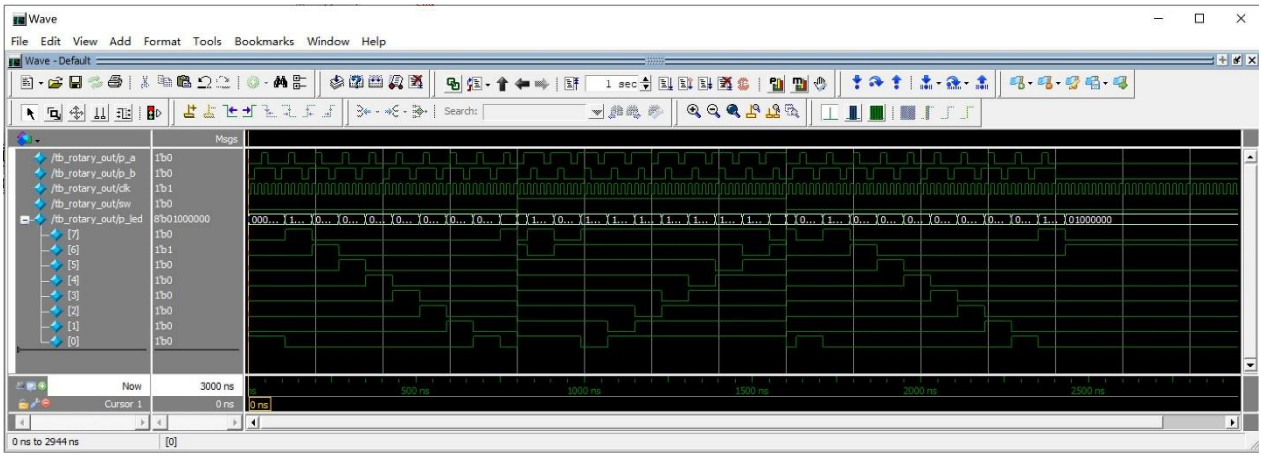
```
    rotary_in <= {rotary_b, rotary_a};
    case (rotary_in)
        2'b00: rotary_q1 <= 1'b0;
        2'b01: rotary_q2 <= 1'b0;
        2'b10: rotary_q2 <= 1'b1;
        2'b11: rotary_q1 <= 1'b1;
        default:
            begin
                rotary_q1 <= 1'b0;
                rotary_q2 <= 1'b0;
            end
    endcase
end

always @(posedge clk) begin : direction
    delay_rotary_q1 <= rotary_q1;

    if (rotary_q1 && (!delay_rotary_q1))
        begin
            rotary_event <= 1'b1;
            rotary_left <= rotary_q2;
        end
    else
        rotary_event <= 1'b0;
end

always @(posedge clk)
begin
    if(rotary_event)
    begin
        if(rotary_left)
            LED<={LED[0], LED[7:1]};

        else
            LED<={LED[6:0], LED[7]};
    end
 end

assign ledout=(~sw)?LED:(~LED);      //若 sw 为 1 则将 LED 按位取反
```

```
endmodule


//rotary_out.ucf
Net "rotary_a" LOC = "K18" | IOSTANDARD=LVTTL | PULLUP ;
Net "rotary_b" LOC = "G18" | IOSTANDARD=LVTTL | PULLUP ;
NET "ledout<7>" LOC = "F9" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 4 ;
NET "ledout<6>" LOC = "E9" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 4 ;
NET "ledout<5>" LOC = "D11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 4 ;
NET "ledout<4>" LOC = "C11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 4 ;
NET "ledout<3>" LOC = "F11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 4 ;
NET "ledout<2>" LOC = "E11" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 4 ;
NET "ledout<1>" LOC = "E12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 4 ;
NET "ledout<0>" LOC = "F12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE =4 ;
NET "clk" LOC = "C9" | IOSTANDARD = LVTTL;
NET "sw" LOC = "N17" | IOSTANDARD = LVTTL | PULLUP ;


//仿真代码
`include "rotary_out.v"
module tb_rotary_out;
  reg p_a,p_b,clk;
  reg sw;
  wire [7:0]p_led;

  rotary_out led(.ledout(p_led),.rotary_a(p_a),.rotary_b(p_b), .sw(sw),.clk(clk));

  initial
  begin
    clk=0;
    sw=0;
    p_a=0;
    p_b=0;
    forever #10 clk=~clk;
  end

  initial
    begin
    repeat(10)
    begin
```

```
            #20 p_b=1;
            #20 p_a=1;
            #20 p_a=0;
            #20 p_b=0;
    end
//开关为 1，只有一个灯灭
    sw=1;
    repeat(10)
    begin
            #20 p_a=1;
            #20 p_b=1;
            #20 p_b=0;
            #20 p_a=0;
    end

//开关为 0，只有一个灯亮
    sw=0;
    repeat(10)
    begin
            #20 p_b=1;
            #20 p_a=1;
            #20 p_a=0;
            #20 p_b=0;
    end
    end

  initial
#3000 $stop;

  initial
$monitor("switch=%d,time=%d,ledout=%b",sw,$time,p_led);

endmodule
```

6、试验仿真结果和分析

如下图，当 Switch=1 时，根据 clock 变化，只有一个灯亮且亮灯顺序指向下标数字大的方向；当 Switch=0 时，根据 clock 变化，只有一个灯灭且灭灯顺序指向下标数字大的方向。达到实验功能要求。




```
VSIM 16> run
# switch=0,time=                    0,ledout=00000001
# switch=0,time=                  110,ledout=10000000
# switch=0,time=                  190,ledout=01000000
# switch=0,time=                  270,ledout=00100000
# switch=0,time=                  350,ledout=00010000
# switch=0,time=                  430,ledout=00001000
# switch=0,time=                  510,ledout=00000100
# switch=0,time=                  590,ledout=00000010
# switch=0,time=                  670,ledout=00000001
# switch=0,time=                  750,ledout=10000000
# switch=1,time=                  800,ledout=01111111
# switch=1,time=                  830,ledout=10111111
# switch=1,time=                  910,ledout=01111111
# switch=1,time=                  990,ledout=11111110
# switch=1,time=                 1070,ledout=11111101
# switch=1,time=                 1150,ledout=11111011
# switch=1,time=                 1230,ledout=11110111
# switch=1,time=                 1310,ledout=11101111
# switch=1,time=                 1390,ledout=11011111
# switch=1,time=                 1470,ledout=10111111
# switch=1,time=                 1550,ledout=01111111
# switch=0,time=                 1600,ledout=10000000
# switch=0,time=                 1630,ledout=00000001
# switch=0,time=                 1710,ledout=10000000
# switch=0,time=                 1790,ledout=01000000
# switch=0,time=                 1870,ledout=00100000
# switch=0,time=                 1950,ledout=00010000
# switch=0,time=                 2030,ledout=00001000
# switch=0,time=                 2110,ledout=00000100
# switch=0,time=                 2190,ledout=00000010
# switch=0,time=                 2270,ledout=00000001
# switch=0,time=                 2350,ledout=10000000
# switch=0,time=                 2430,ledout=01000000
```