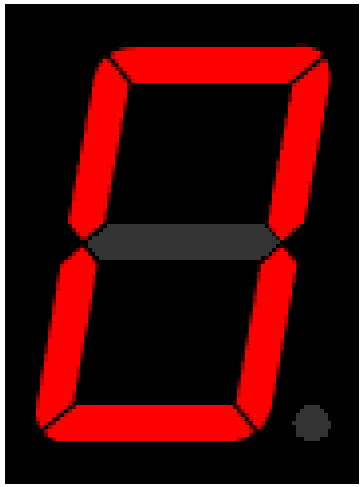
实验2：七段LED数码管



xgsun@fudan.edu.cn

孙晓光

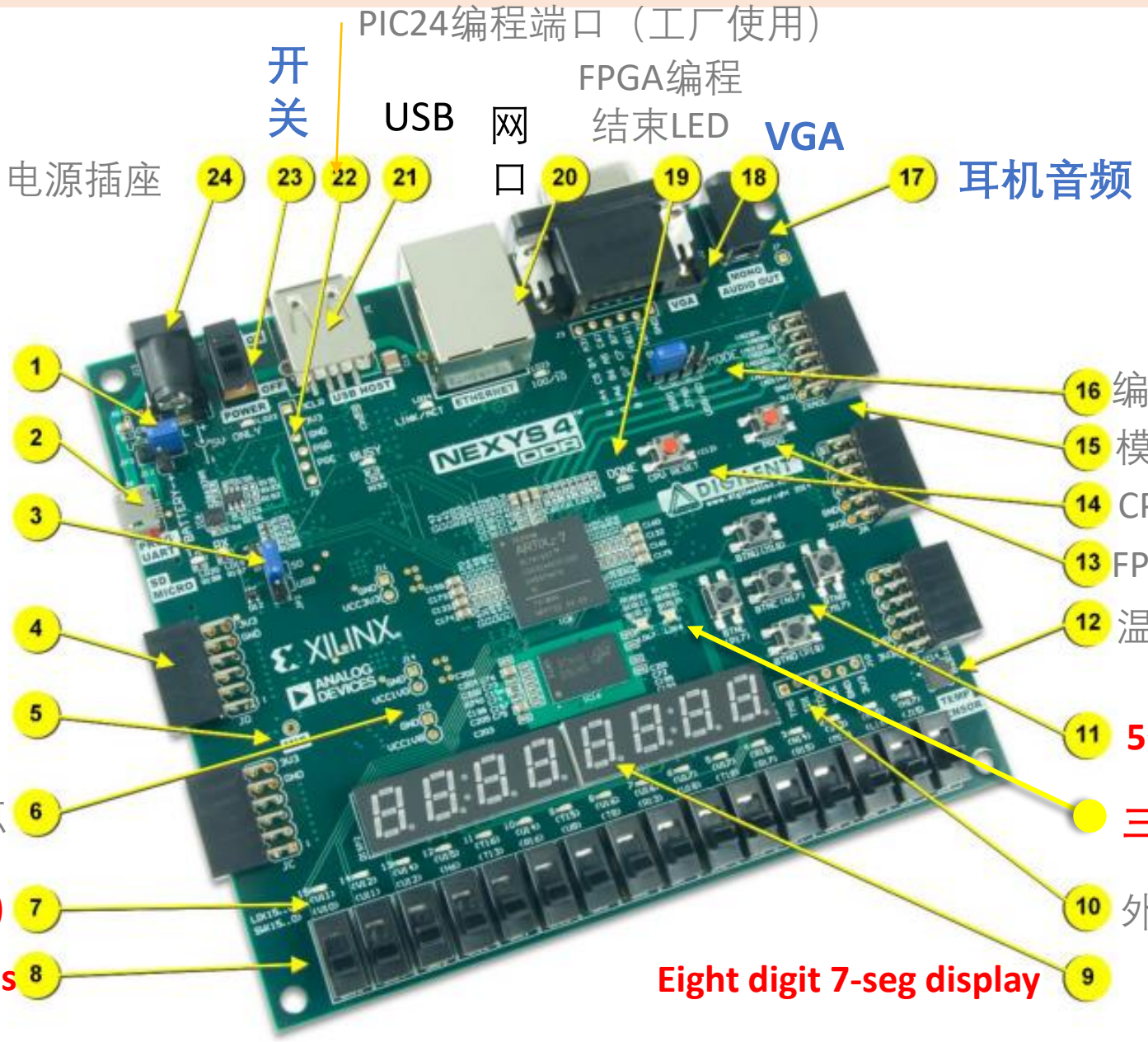
2024-9-8

1

NEXYS 4

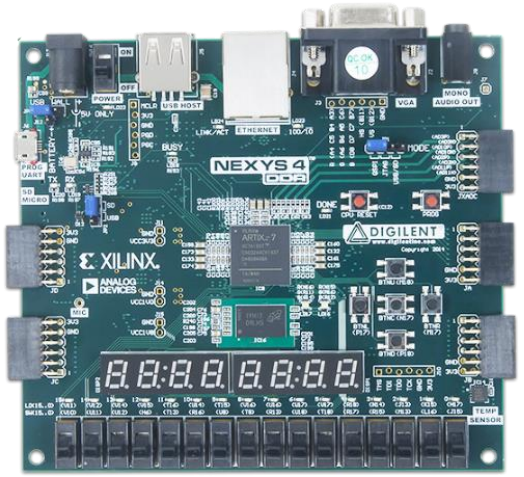
简 介

通过USB供电、编程以及串口监视，只需1根USB线



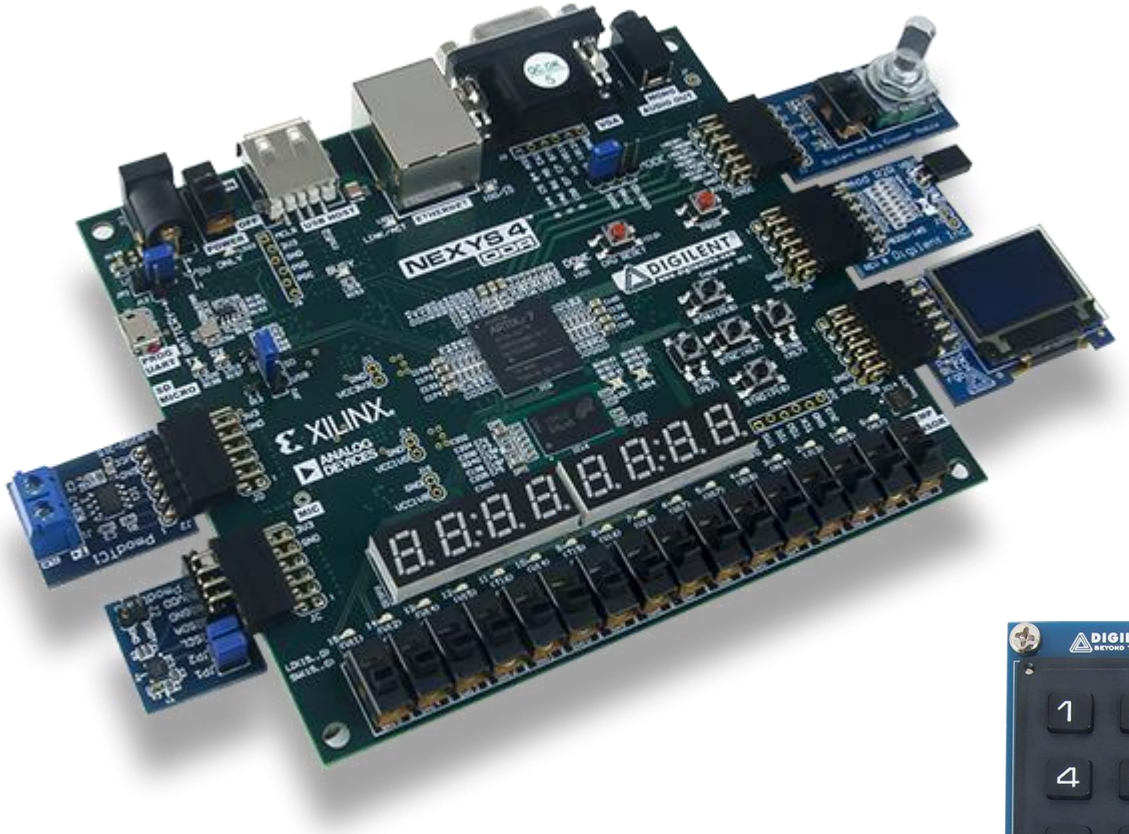
Nexys4 DDR 特点

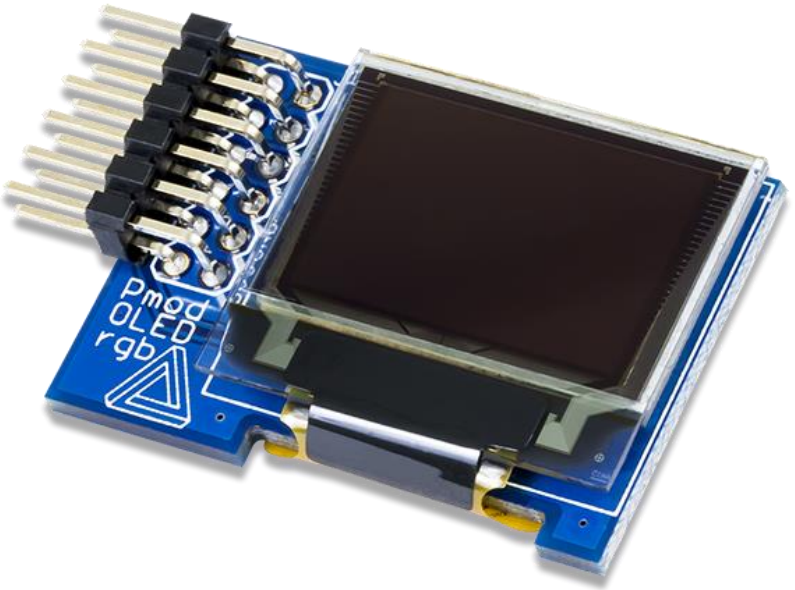
- 10/100 Ethernet PHY
- 128Mib DDR2
- SD卡插槽（可用来存储FPGA程序）
- Micro USB接口（UART和编程口共用）
- USB HID主机接口（鼠标、键盘）
- Digilent USB-JTAG port for FPGA programming & communication
- 4个Pmod接口、 Pmod for XADC signals
- USB HID Host for mice, keyboards and memory sticks
- **PWM (Pulse Width Modulation)音频输出**
- PDM（脉冲密度调制）数字麦克风
- **基本IO：16路拨码开关、16路LED、8路七段LED数码管**
  2个RGB 三色 LED、5个按钮



- **12位VGA接口**
- 温度传感器
- 串行Flash
- 3轴加速度计

Pmod接口*





96x64,16位色彩分辨率RGB OLED屏

带16个按键的键盘

带有并行接口的字符型LCD

XC7A100T

Artix-7, Spartan-7, Kintex-7, Virtex-7

Xilinx(赛灵思)公司

Artix-7是 **7系列** 中低成本系列芯片，

面向成本敏感型应用FPGA，具有目前最低的成本和功耗优势，

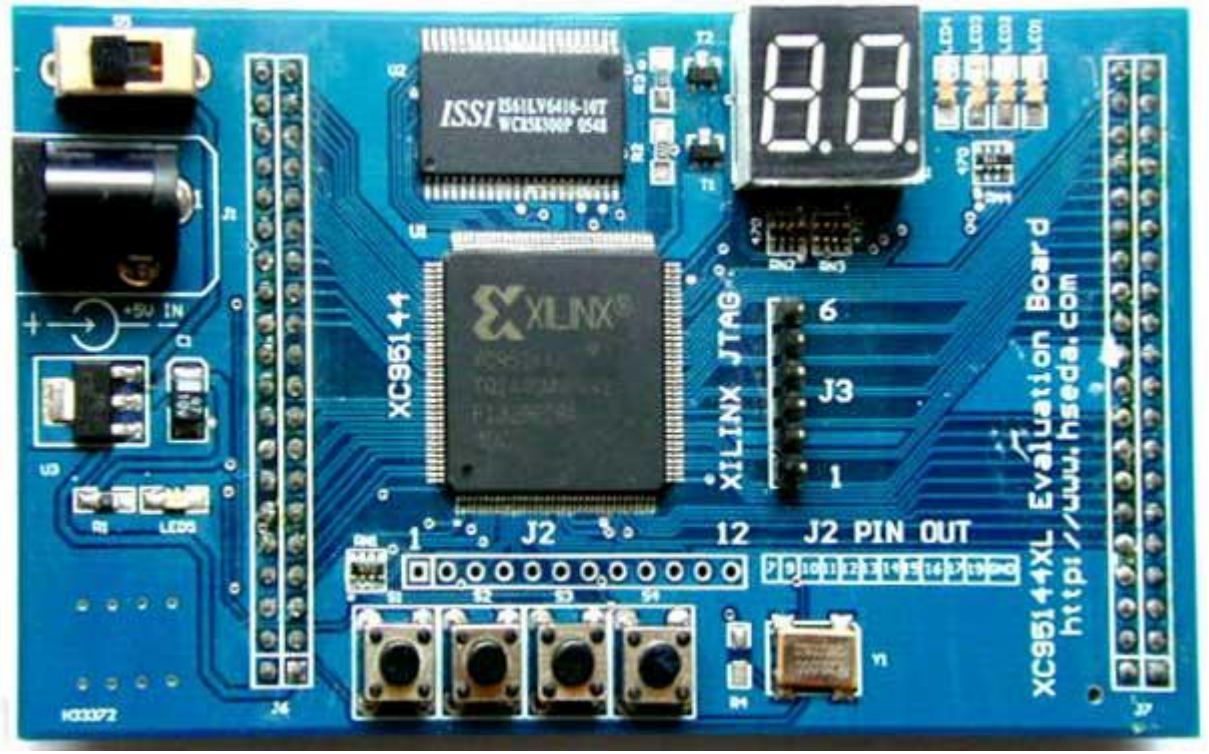
特别适用于消费类电子以及手持和便携设备。

XC7A100T是Artix-7系列中资源比较丰富的一款芯片

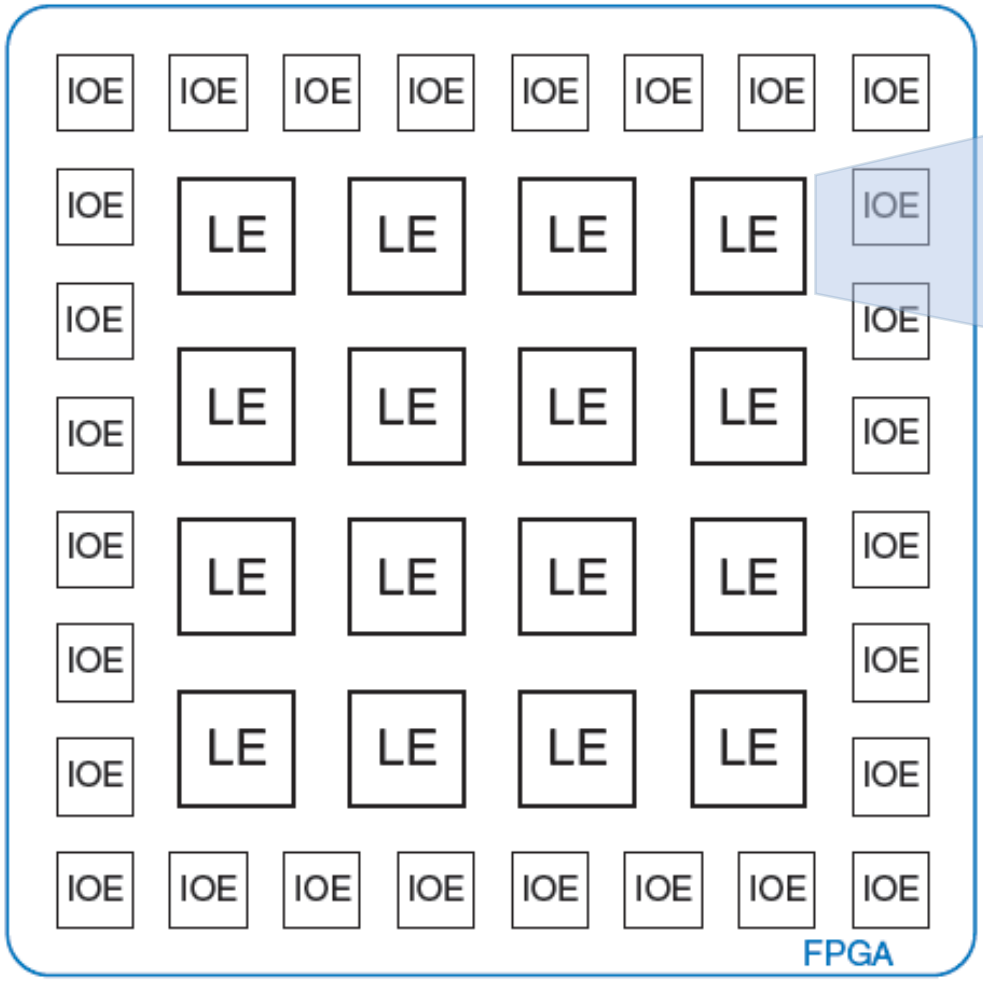
- 15,850 logic slices, each with four **6-input LUTs** and **8 flip-flops**
- 4,860 Kbits快速随机存储器RAM
- Six clock management tiles, each with phase-locked loop (PLL)
- 内部时钟速度超过450 MHz
- 240 DSP(数字信号处理) slices(切片)
- 片上模数转换器（XADC）

FPGA 现场可编程门阵列

- FPGA：可由用户现场进行编程的大规模电路。
- FPGA是80年代中期发展起来的一种可编程的。
- 特点：
  - 保密性强
  - 体积小
  - 可靠性高

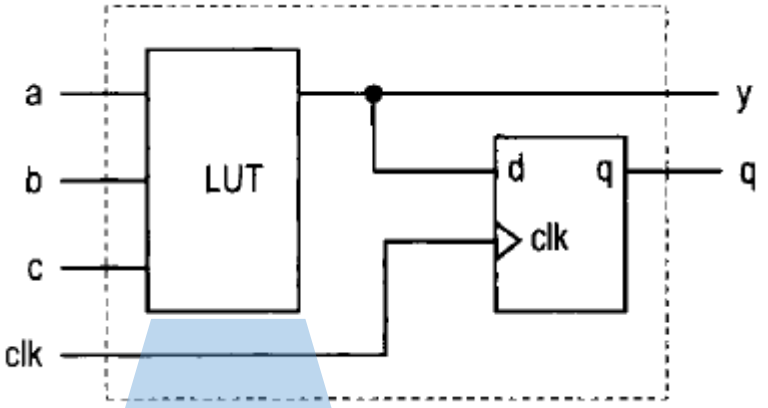


通用FPGA结构图



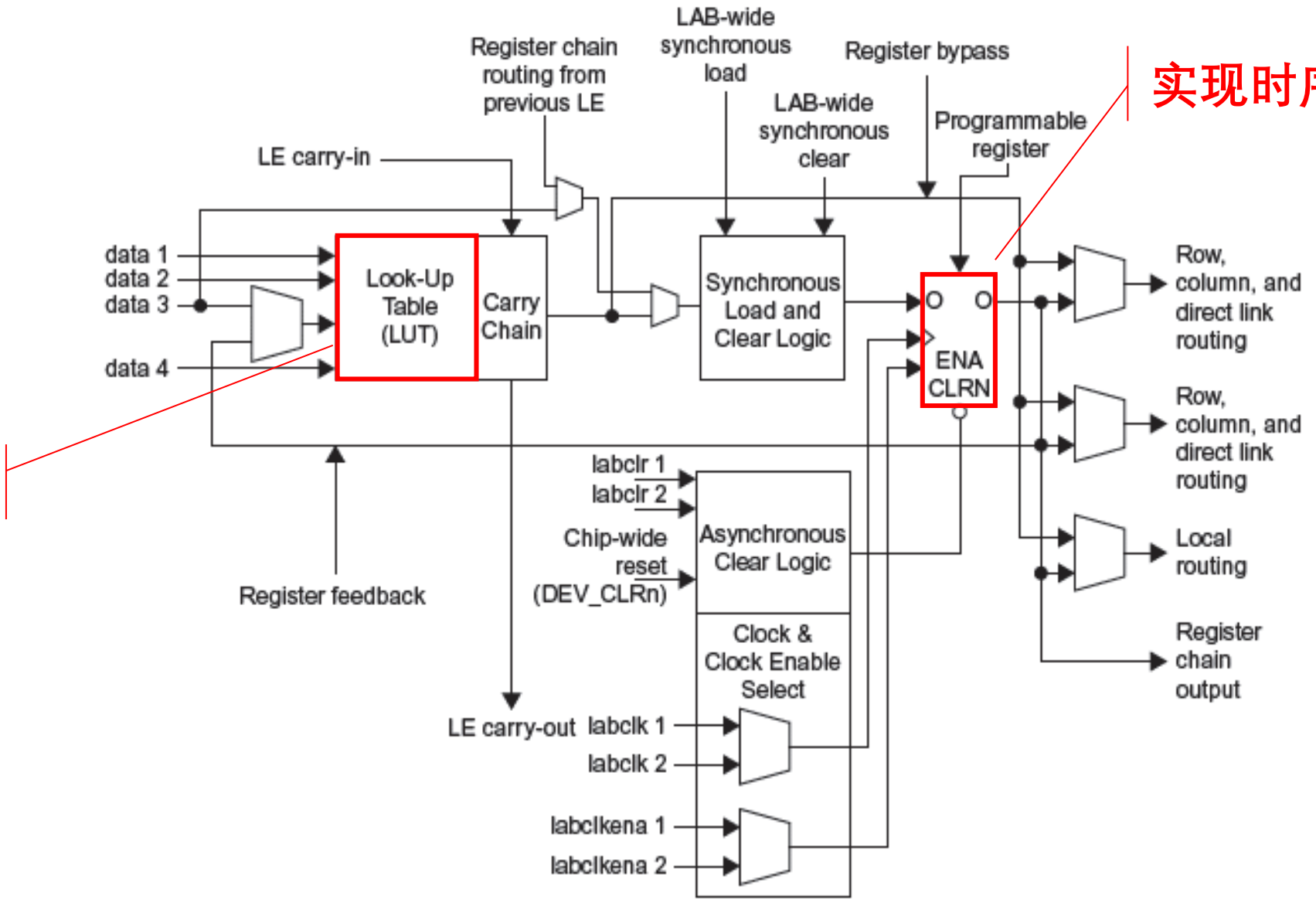
LE：逻辑单元

LUT： Look-up Table



a b c	y
0 0 0	0
0 0 1	1
0 1 0	1
0 1 1	0
1 0 0	1
1 0 1	0
1 1 0	0
1 1 1	1

Altera's Cyclone IV LE



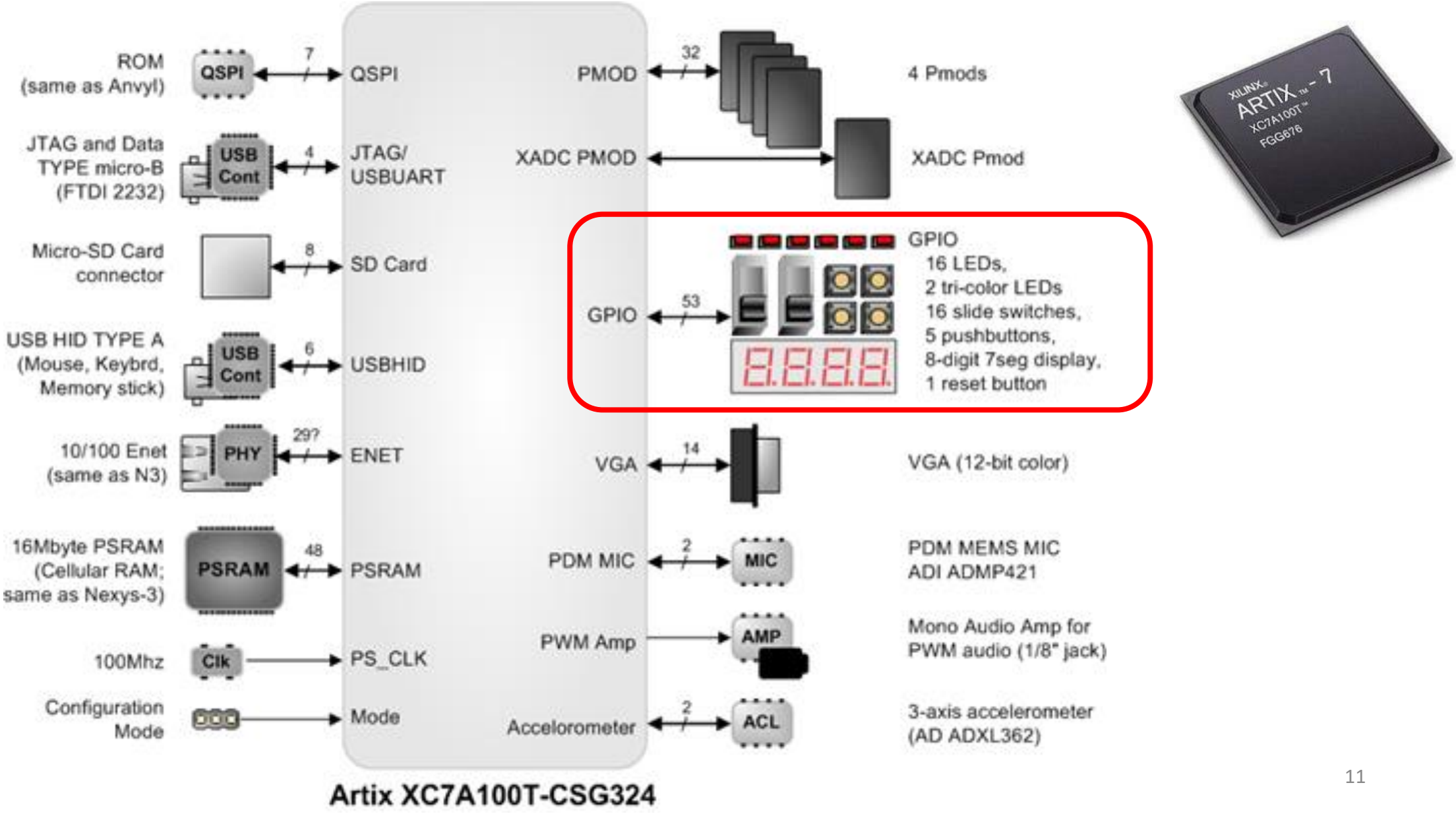
实现时序逻辑

实现组合逻辑

2

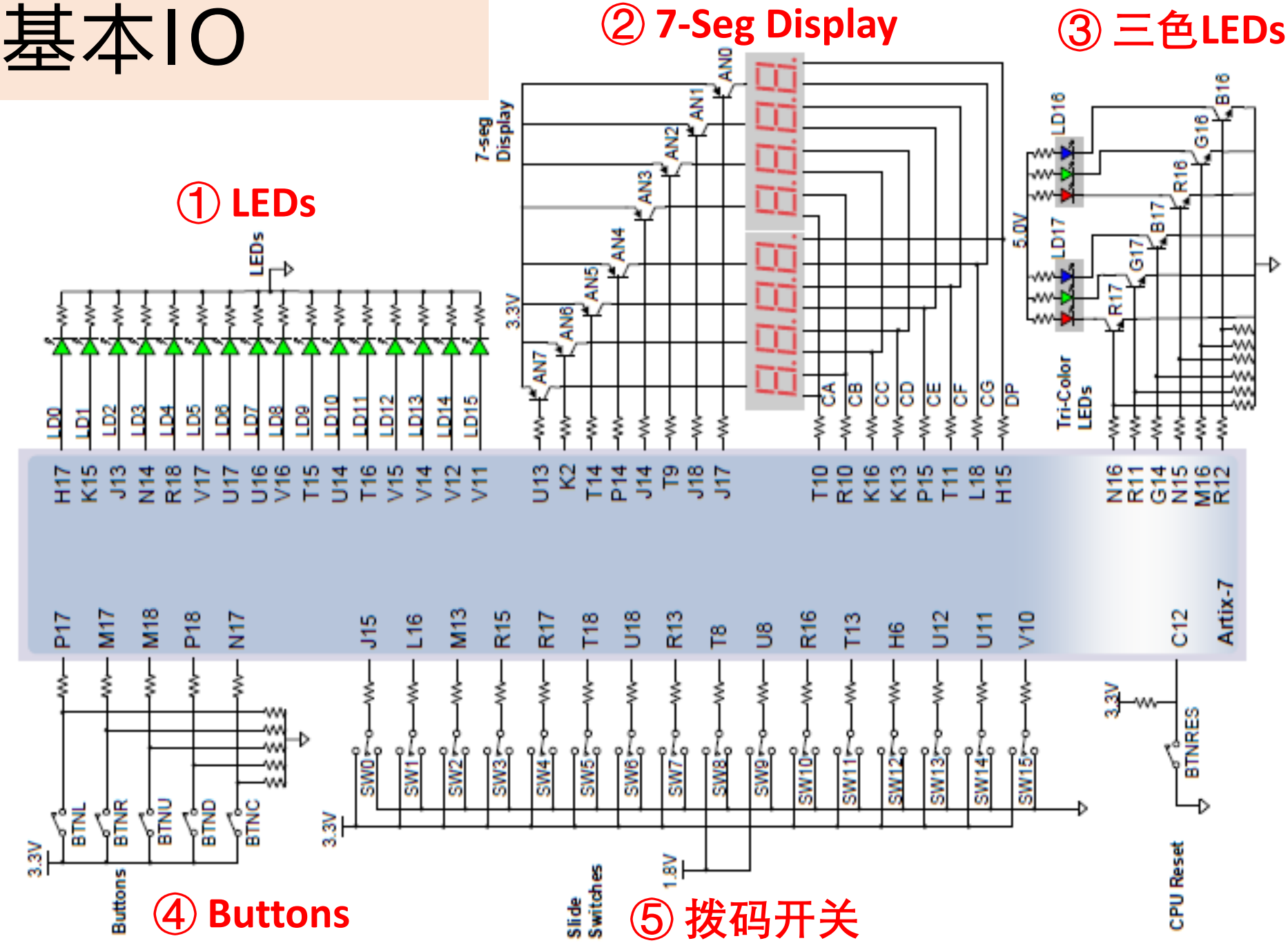
基本IO

XC7A100T

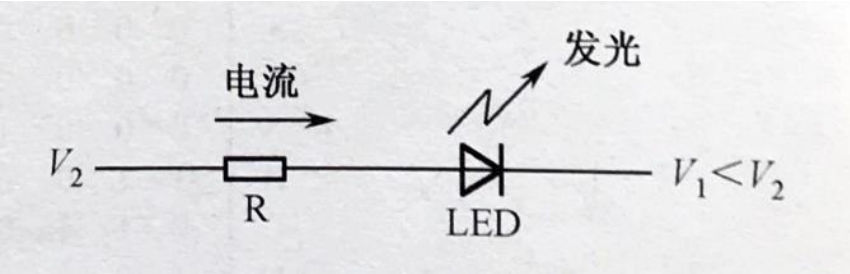

ROM
(same as Anvyl)
QSPI
7
QSPI
JTAG and Data
TYPE micro-B
(FTDI 2232)
USB
Cont
4
JTAG/
USBUART
Micro-SD Card
connector
8
SD Card
USB HID TYPE A
(Mouse, Keybrd,
Memory stick)
USB
Cont
6
USBHID
10/100 Enet
(same as N3)
PHY
29?
ENET
16Mbyte PSRAM
(Cellular RAM;
same as Nexys-3)
PSRAM
48
PSRAM
100Mhz
Clk
PS_CLK
Configuration
Mode
Mode
PMOD
32
4 Pmods
XADC PMOD
XADC Pmod
GPIO
53
GPIO
16 LEDs,
2 tri-color LEDs
16 slide switches,
5 pushbuttons,
8-digit 7seg display,
1 reset button
VGA
14
VGA (12-bit color)
PDM MIC
2
MIC
PDM MEMS MIC
ADI ADMP421
PWM Amp
AMP
Mono Audio Amp for
PWM audio (1/8" jack)
Accelorometer
2
ACL
3-axis accelerometer
(AD ADXL362)
Artix XC7A100T-CSG324

XILINX
ARTIX - 7
XC7A100T
FGG676

基本IO


① LEDs
② 7-Seg Display
③ 三色LEDs
④ Buttons
⑤ 拨码开关
LEDs
7-seg Display
Tri-Color LEDs
Buttons
Slide Switches
CPU Reset
Artix-7
3.3V
5.0V
1.8V
BTNL
BTNR
BTNU
BTND
BTNC
BTNRES

① LEDs + Switches



当一个正向电流通过发光二极管(LED)时，LED就会发光。

FPGA的I/O引脚有**2种**方式点亮LED：

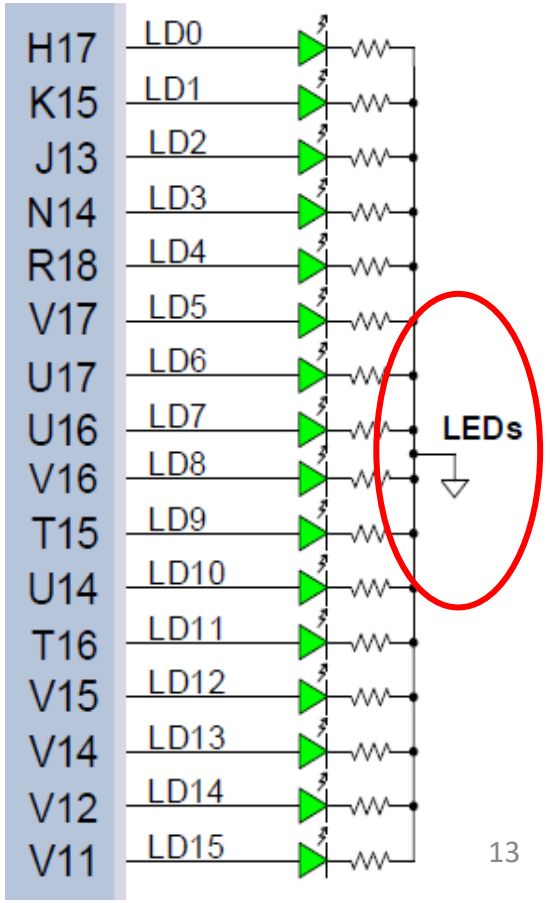
① LED右端V_1接地

- **点亮LED，V_2端为高电平(1)**
- 熄灭LED，V_2端为低电平(0)

② LED左端V_2接恒定电压

- **点亮LED，V_1端为低电平(0)**
- 熄灭LED，V_1端为高电平(1)

Artix-7

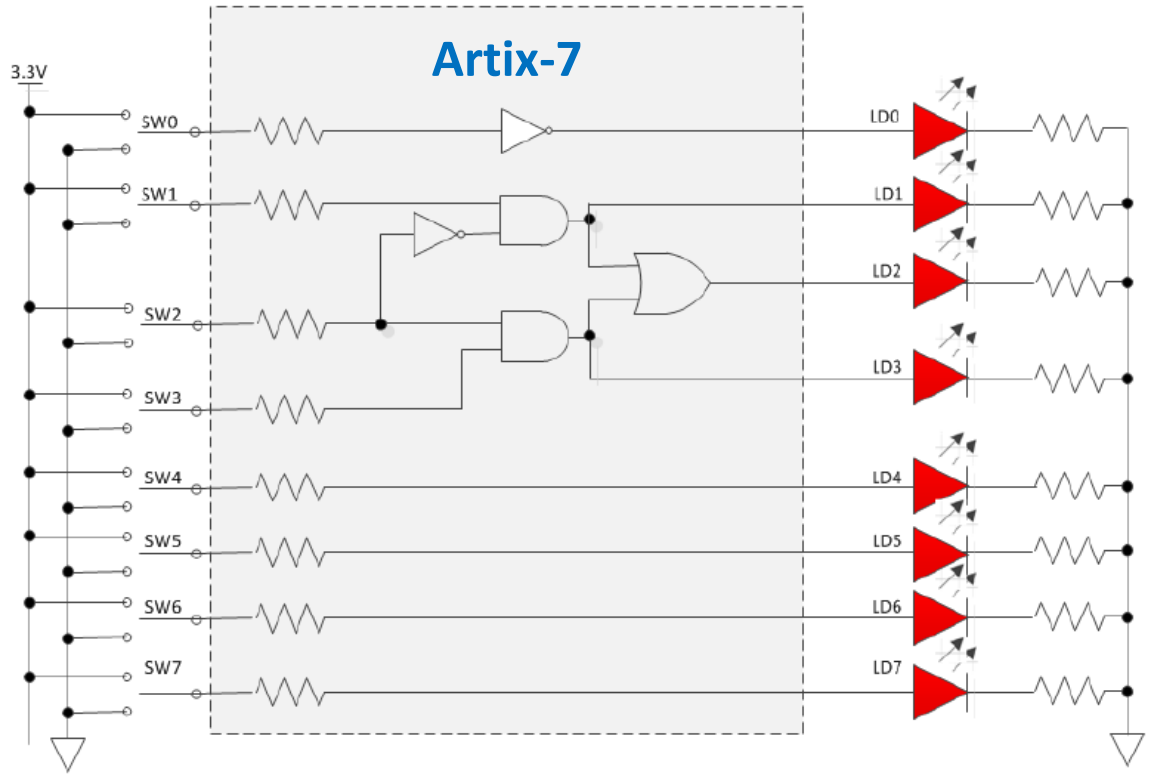


LEDs
代码

例：点亮LED

Leds.sv

```
1  module Leds(
2      input  logic [3:0] SW,
3      output logic [3:0] LED);
4
5      assign LED = SW;
6  endmodule
```



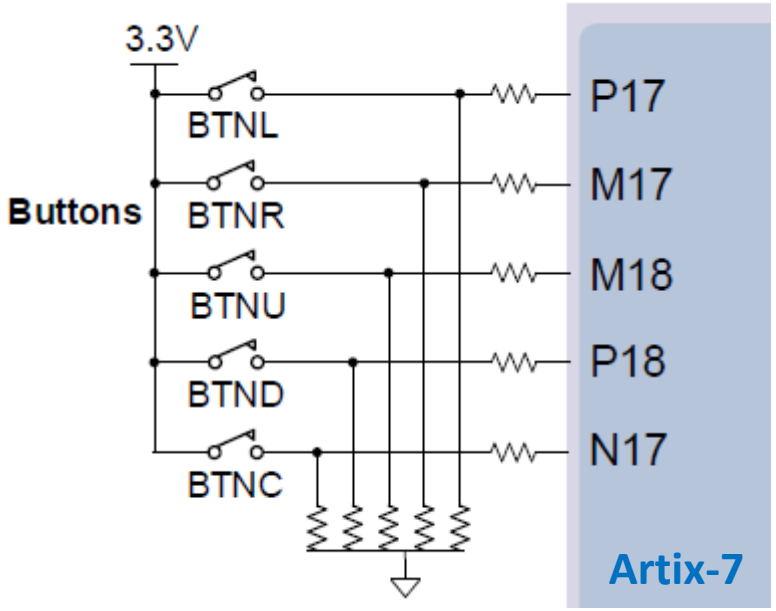
Nexys4DDR_Master.xdc

```
1  ##Switches
2  set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { SW[0] }]; #IO_L24N_T3_RS0_15 Sch=sw[0]
3  set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { SW[1] }]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
4  set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { SW[2] }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]
5  set_property -dict { PACKAGE_PIN R15   IOSTANDARD LVCMOS33 } [get_ports { SW[3] }]; #IO_L13N_T2_MRCC_14 Sch=sw[3]
6
7  ## LEDs
8  set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { LED[0] }]; #IO_L18P_T2_A24_15 Sch=led[0]
9  set_property -dict { PACKAGE_PIN K15   IOSTANDARD LVCMOS33 } [get_ports { LED[1] }]; #IO_L24P_T3_RS1_15 Sch=led[1]
10 set_property -dict { PACKAGE_PIN J13   IOSTANDARD LVCMOS33 } [get_ports { LED[2] }]; #IO_L17N_T2_A25_15 Sch=led[2]
11 set_property -dict { PACKAGE_PIN N14   IOSTANDARD LVCMOS33 } [get_ports { LED[3] }]; #IO_L8P_T1_D11_14 Sch=led[3]
```

② Buttons 示例代码

Button.sv

```
1  module Button(
2      input  logic       BTNC,
3      output logic [0:0] LED);
4
5      assign LED[0] = BTNC;
6  endmodule
```



Nexys4DDR_Master.xdc

```
1  ## LEDs
2  set_property -dict { PACKAGE_PIN H17   IOSTANDARD LVCMOS33 } [get_ports { LED[0] }]; #IO_L18P_T2_A24_15 Sch=led[0]
3
4  ##Buttons
5  set_property -dict { PACKAGE_PIN N17   IOSTANDARD LVCMOS33 } [get_ports { BTNC }]; #IO_L9P_T1_DQS_14 Sch=btnc
6  #set_property -dict { PACKAGE_PIN M18   IOSTANDARD LVCMOS33 } [get_ports { BTNU }]; #IO_L4N_T0_D05_14 Sch=btnu
7  #set_property -dict { PACKAGE_PIN P17   IOSTANDARD LVCMOS33 } [get_ports { BTNL }]; #IO_L12P_T1_MRCC_14 Sch=btnl
8  #set_property -dict { PACKAGE_PIN M17   IOSTANDARD LVCMOS33 } [get_ports { BTNR }]; #IO_L10N_T1_D15_14 Sch=btnr
9  #set_property -dict { PACKAGE_PIN P18   IOSTANDARD LVCMOS33 } [get_ports { BTND }]; #IO_L9N_T1_DQS_D13_14 Sch=btnd
10
```

ColorLED

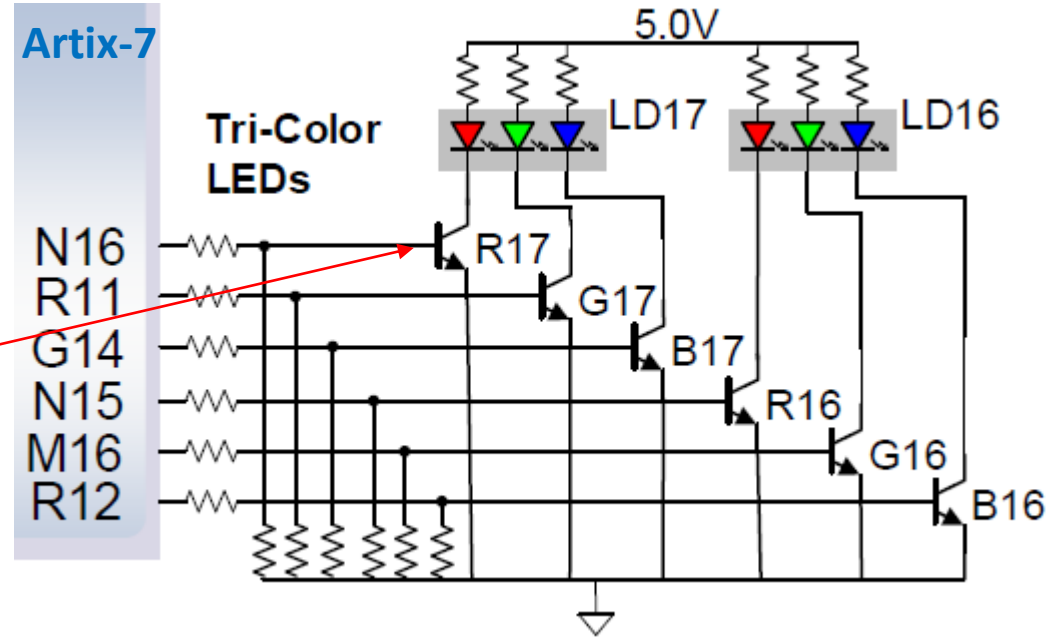
③ Tri-Color LEDs 代码

ColorLED.sv

C:/Users/Sam/Documents/Vivado2015/ColorLED

```
1  module colorLED(
2      input logic [2:0] SW,
3      output logic LED16_B,
4      output logic LED16_G,
5      output logic LED16_R);
6
7      assign LED16_B = SW[0];
8      assign LED16_G = SW[1];
9      assign LED16_R = SW[2];
10 endmodule
```

三极管基极
高电压时导通



Nexys4DDR_Master.xdc

C:/Users/Sam/Documents/Vivado2015/ColorLED/ColorLED.srcs/constrs_1/imports/01 XUP Nexys4 DDR Board/Nexys4DDR_Master.xdc

```
1 ##Switches
2 set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { SW[0] }]; #IO_L24N_T3_RS0_15 Sch=sw[0]
3 set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { SW[1] }]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
4 set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { SW[2] }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]
5
6 ## LEDs
7 set_property -dict { PACKAGE_PIN R12   IOSTANDARD LVCMOS33 } [get_ports { LED16_B }]; #IO_L5P_T0_D06_14 Sch=led16_b
8 set_property -dict { PACKAGE_PIN M16   IOSTANDARD LVCMOS33 } [get_ports { LED16_G }]; #IO_L10P_T1_D14_14 Sch=led16_g
9 set_property -dict { PACKAGE_PIN N15   IOSTANDARD LVCMOS33 } [get_ports { LED16_R }]; #IO_L11P_T1_SRCC_14 Sch=led16_r
```

3

七段数码管

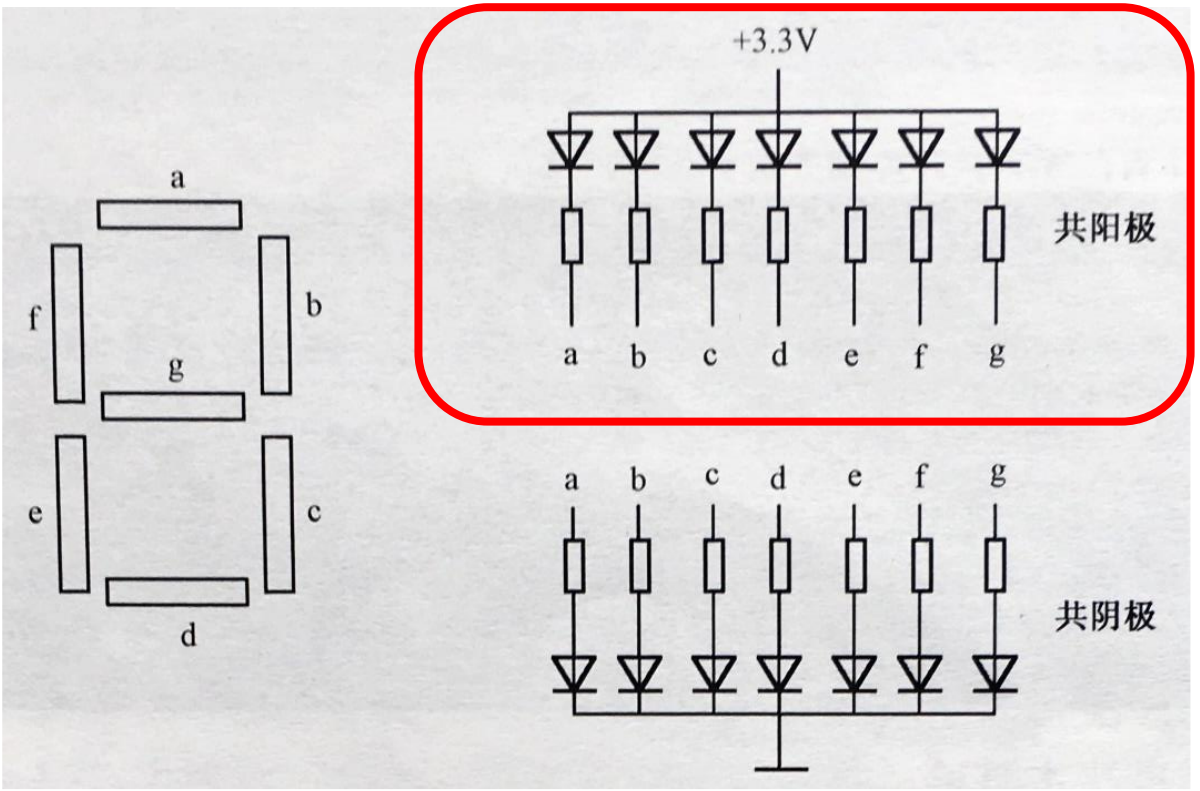
④ 七段LED数码管

① 共阳极

NEXYS4 (DDR)

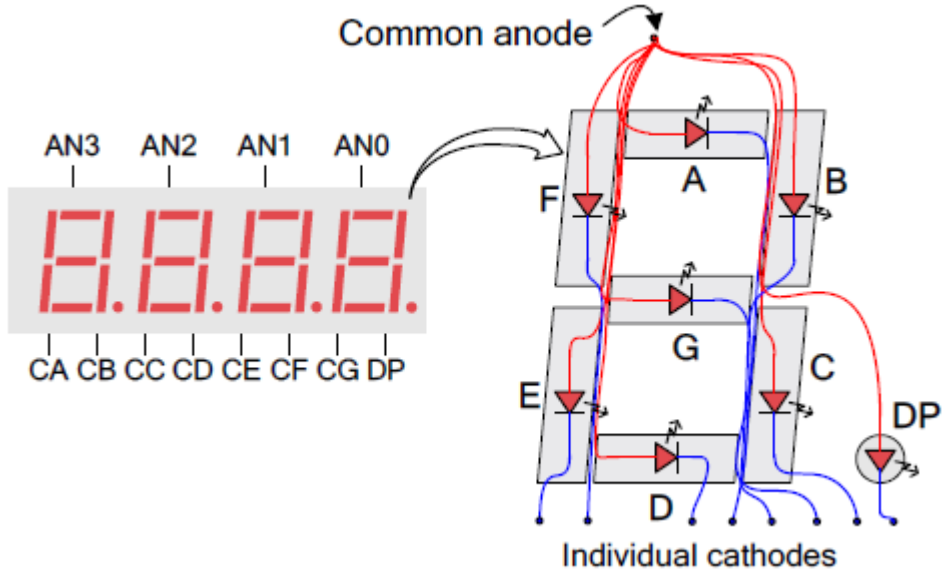
- 点亮，引脚输出为低电平(0)
- 熄灭，引脚输出为高电平(1)

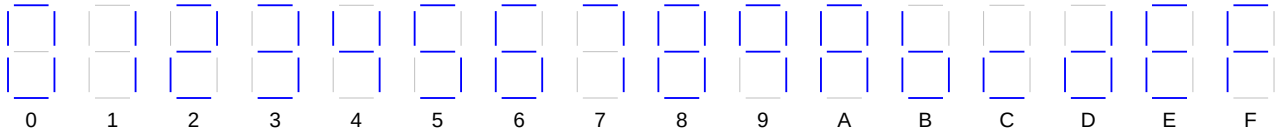
② 共阴极



AN0..7取反了

Both the **AN0..7** and the **CA..AG/DP** signals are driven **low** when active.

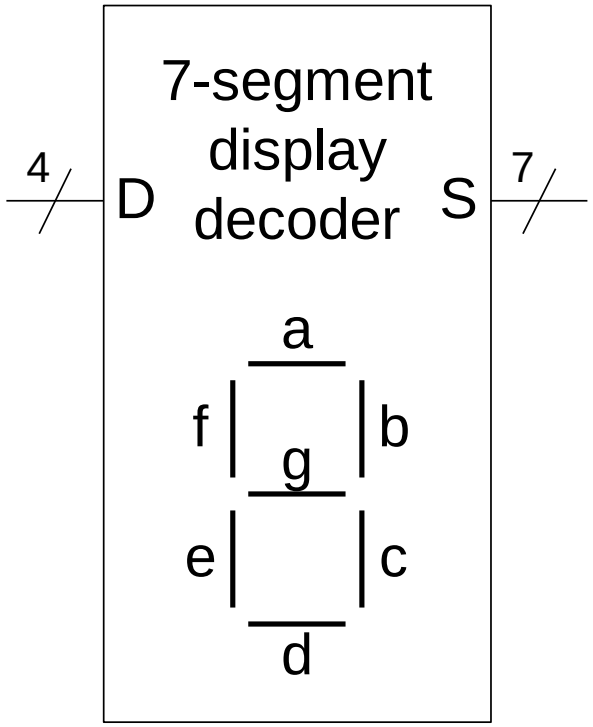


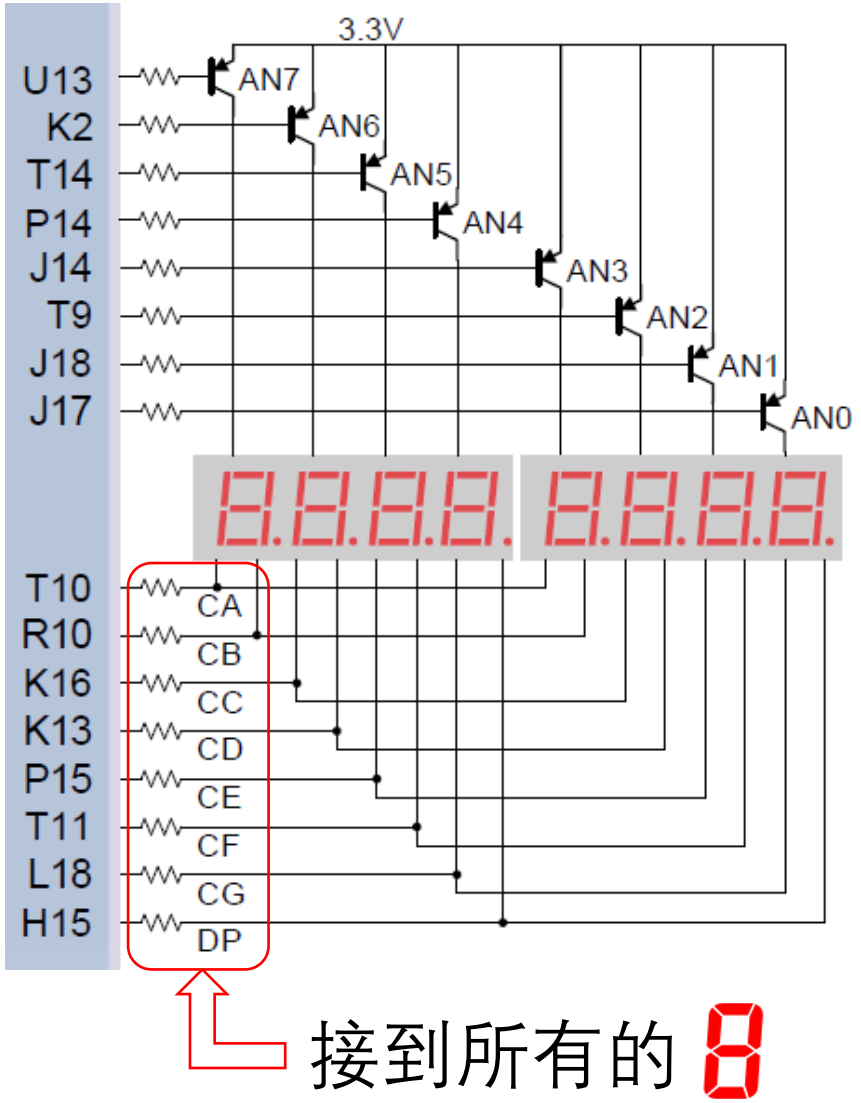


七段数码管-方案1

共阳极

- **点亮，引脚输出为低电平(0)**
- 熄灭，引脚输出为高电平(1)






```
1  module SevenSegLED(
2      input   logic [3:0] SW,
3      output  logic [3:0] AN,    //使能
4      output  logic       DP,    //小数点
5      output  logic [6:0] A2G );
6
7      assign AN = 4'b0000;// 右侧4个全亮
8      assign DP = 1;       // DP off
9      // A2G format {a, b, c, d, e, f, g}
10     always_comb
11         case (SW)
12             'h0: A2G = 7'b0000001;
13             'h1: A2G = 7'b1001111;
14             'h2: A2G = 7'b0010010;
15             'h3: A2G = 7'b0000110;
16             'h4: A2G = 7'b1001100;
17             'h5: A2G = 7'b0100100;
18             'h6: A2G = 7'b0100000;
19             'h7: A2G = 7'b0001111;
20             'h8: A2G = 7'b0000000;
21             'h9: A2G = 7'b0000100;
22             'hA: A2G = 7'b0001000;
23             'hB: A2G = 7'b1100000;
24             'hC: A2G = 7'b0110001;
25             'hD: A2G = 7'b1000010;
26             'hE: A2G = 7'b0110000;
27             'hF: A2G = 7'b0111000;
28         default: A2G = 7'b0000001; //0
29         endcase
30 endmodule
```

16进制七段译码器

7SegmentLED

七段数码管-方案1

都显示同一个数字？

Nexys4DDR_Master.xdc

```
1  ##Switches
2  set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { SW[0] }]; #IO_L24N_T3_RS0_15 Sch=sw[0]
3  set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { SW[1] }]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
4  set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { SW[2] }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]
5  set_property -dict { PACKAGE_PIN R15   IOSTANDARD LVCMOS33 } [get_ports { SW[3] }]; #IO_L13N_T2_MRCC_14 Sch=sw[3]
6
7  ##7 segment display
8  set_property -dict { PACKAGE_PIN T10   IOSTANDARD LVCMOS33 } [get_ports { A2G[6] }]; #IO_L24N_T3_A00_D16_14 Sch=ca
9  set_property -dict { PACKAGE_PIN R10   IOSTANDARD LVCMOS33 } [get_ports { A2G[5] }]; #IO_25_14 Sch=cb
10 set_property -dict { PACKAGE_PIN K16   IOSTANDARD LVCMOS33 } [get_ports { A2G[4] }]; #IO_25_15 Sch=cc
11 set_property -dict { PACKAGE_PIN K13   IOSTANDARD LVCMOS33 } [get_ports { A2G[3] }]; #IO_L17P_T2_A26_15 Sch=cd
12 set_property -dict { PACKAGE_PIN P15   IOSTANDARD LVCMOS33 } [get_ports { A2G[2] }]; #IO_L13P_T2_MRCC_14 Sch=ce
13 set_property -dict { PACKAGE_PIN T11   IOSTANDARD LVCMOS33 } [get_ports { A2G[1] }]; #IO_L19P_T3_A10_D26_14 Sch=cf
14 set_property -dict { PACKAGE_PIN L18   IOSTANDARD LVCMOS33 } [get_ports { A2G[0] }]; #IO_L4P_T0_D04_14 Sch=cg
15 set_property -dict { PACKAGE_PIN H15   IOSTANDARD LVCMOS33 } [get_ports { DP }]; #IO_L19N_T3_A21_VREF_15 Sch=dp
16 set_property -dict { PACKAGE_PIN J17   IOSTANDARD LVCMOS33 } [get_ports { AN[0] }]; #IO_L23P_T3_FOE_B_15 Sch=an[0]
17 set_property -dict { PACKAGE_PIN J18   IOSTANDARD LVCMOS33 } [get_ports { AN[1] }]; #IO_L23N_T3_FWE_B_15 Sch=an[1]
18 set_property -dict { PACKAGE_PIN T9    IOSTANDARD LVCMOS33 } [get_ports { AN[2] }]; #IO_L24P_T3_A01_D17_14 Sch=an[2]
19 set_property -dict { PACKAGE_PIN J14   IOSTANDARD LVCMOS33 } [get_ports { AN[3] }]; #IO_L19P_T3_A22_15 Sch=an[3]
```

```
1  module SevenSegLED(
2      input   logic [3:0] SW,
3      output  logic [3:0] AN,    //使能
4      output  logic       DP,    //小数点
5      output  logic [6:0] A2G );
6
7      assign AN = 4'b0000;// 右侧4个全亮
8      assign DP = 1;      // DP off
9      // A2G format {a, b, c, d, e, f, g}
10     always_comb
11         case (SW)
12             'h0: A2G = 7'b0000001;
13             'h1: A2G = 7'b1001111;
14             'h2: A2G = 7'b0010010;
15             'h3: A2G = 7'b0000110;
16             'h4: A2G = 7'b1001100;
17             'h5: A2G = 7'b0100100;
18             'h6: A2G = 7'b0100000;
19             'h7: A2G = 7'b0001111;
20             'h8: A2G = 7'b0000000;
21             'h9: A2G = 7'b0000100;
22             'hA: A2G = 7'b0001000;
23             'hB: A2G = 7'b1100000;
24             'hC: A2G = 7'b0110001;
25             'hD: A2G = 7'b1000010;
26             'hE: A2G = 7'b0110000;
27             'hF: A2G = 7'b0111000;
28         default: A2G = 7'b0000001; //0
29         endcase
30 endmodule
```

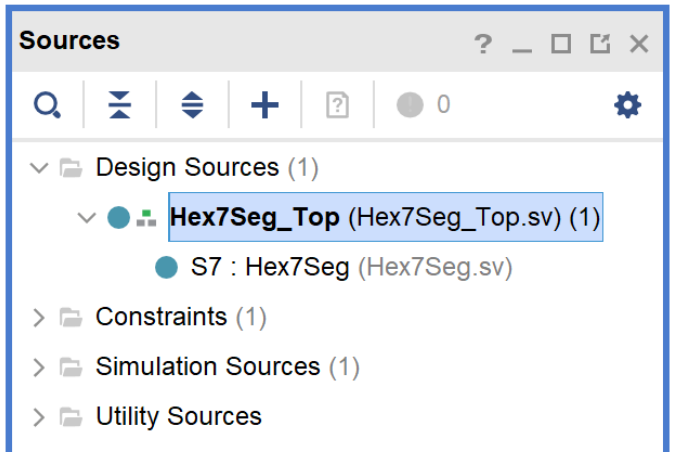
16进制七段译码器

Hex7Seg

7段数码管 - 方案2

顶层文件

```
1  module Hex7Seg_Top(
2      input  logic [3:0]SW,
3      output logic [6:0]A2G,
4      output logic [3:0]AN,
5      output logic DP );
6
7      assign AN = 4'b0000;     // 右侧4个亮
8      assign DP = 1;           // all dp off
9
10     // 实例化 7段数码管
11     Hex7Seg S7(.data(SW), .a2g(A2G));
12 endmodule
```



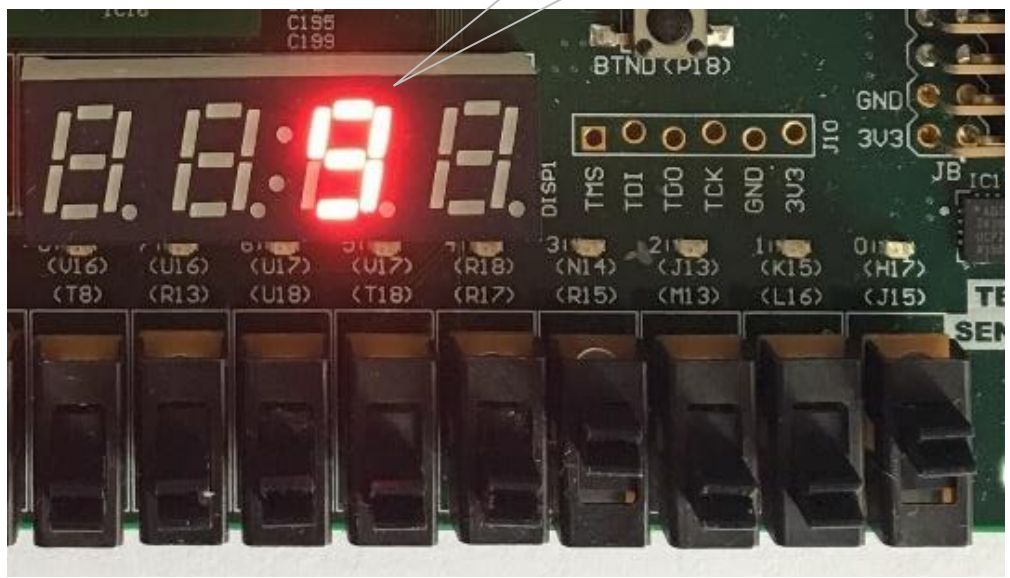

```
1  module Hex7Seg(
2      input  logic [3:0] data,
3      output logic [6:0] a2g );
4
5  //    assign AN = 4'b0000;// 右侧4个全亮
6  //    assign DP = 1;      // DP off
7
8      // a2g format {a, b, c, d, e, f, g}
9      always_comb
10         case (data)
11             'h0: a2g = 7'b0000001;
12             'h1: a2g = 7'b1001111;
13             'h2: a2g = 7'b0010010;
14             'h3: a2g = 7'b0000110;
15             'h4: a2g = 7'b1001100;
16             'h5: a2g = 7'b0100100;
17             'h6: a2g = 7'b0100000;
18             'h7: a2g = 7'b0001111;
19             'h8: a2g = 7'b0000000;
20             'h9: a2g = 7'b0000100;
21             'hA: a2g = 7'b0001000;
22             'hB: a2g = 7'b1100000;
23             'hC: a2g = 7'b0110001;
24             'hD: a2g = 7'b1000010;
25             'hE: a2g = 7'b0110000;
26             'hF: a2g = 7'b0111000;
27         default: a2g = 7'b0000001; //0
28         endcase
29 endmodule
```

16进制七段译码器

七段数码管 - 方案3

```
assign AN = 4'b1101;
```

只显示右侧第2个

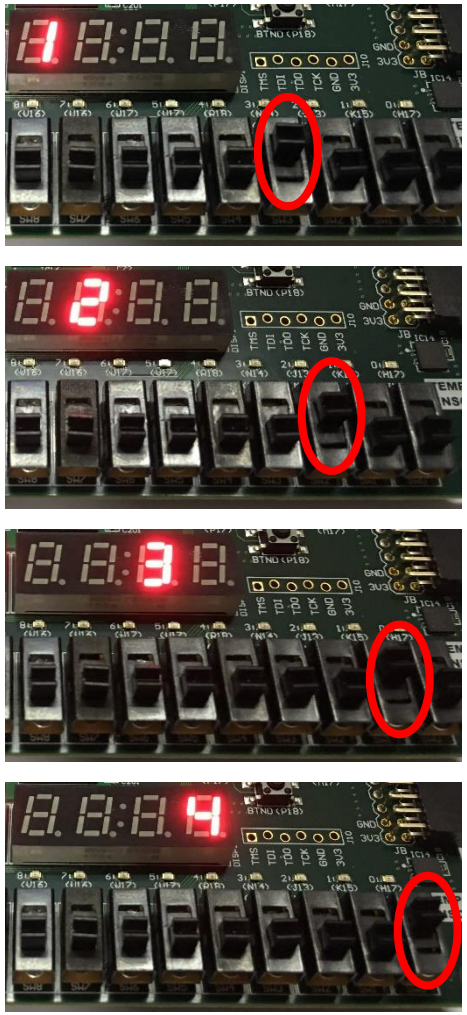



```
1  module SevenSegLED(
2      input   logic [3:0] SW,
3      output  logic [3:0] AN,    //使能
4      output  logic       DP,    //小数点
5      output  logic [6:0] A2G );
6
7      assign AN = 4'b1101;// 只点亮右侧第2个
8      assign DP = 1;          // DP off
9      // A2G format {a, b, c, d, e, f, g}
10     always_comb
11         case (SW)
12             'h0: A2G = 7'b0000001;
13             'h1: A2G = 7'b1001111;
14             'h2: A2G = 7'b0010010;
15             'h3: A2G = 7'b0000110;
16             'h4: A2G = 7'b1001100;
17             'h5: A2G = 7'b0100100;
18             'h6: A2G = 7'b0100000;
19             'h7: A2G = 7'b0001111;
20             'h8: A2G = 7'b0000000;
21             'h9: A2G = 7'b0000100;
22             'hA: A2G = 7'b0001000;
23             'hB: A2G = 7'b1100000;
24             'hC: A2G = 7'b0110001;
25             'hD: A2G = 7'b1000010;
26             'hE: A2G = 7'b0110000;
27             'hF: A2G = 7'b0111000;
28         default: A2G = 7'b0000001; //0
29         endcase
30 endmodule
```

16进制七段译码器

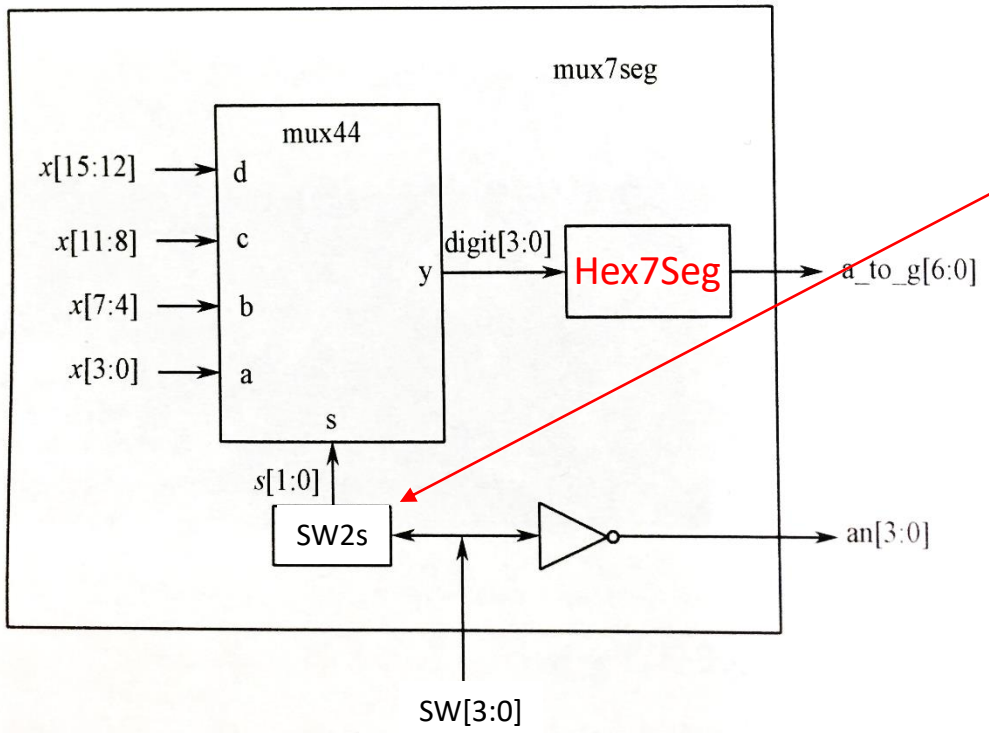
mux7seg

方案4：复用七段数码管



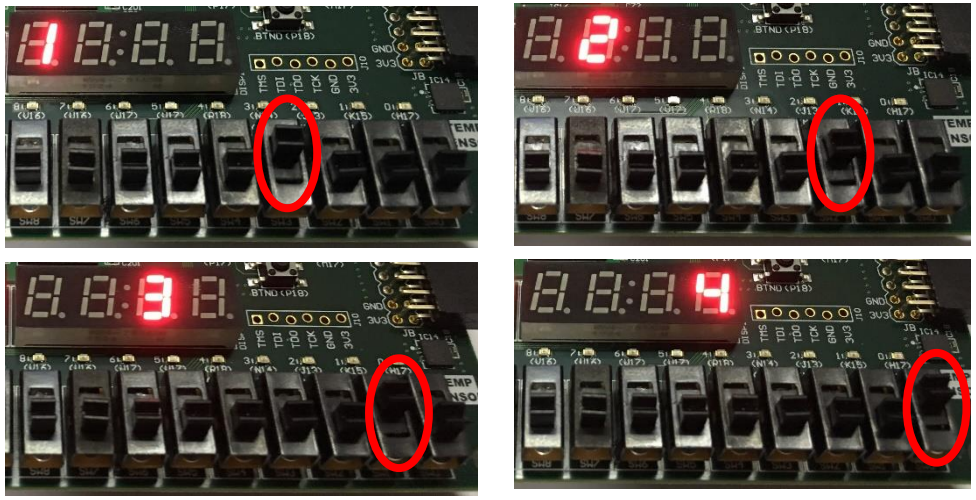
拨码开关控制哪个数码管亮

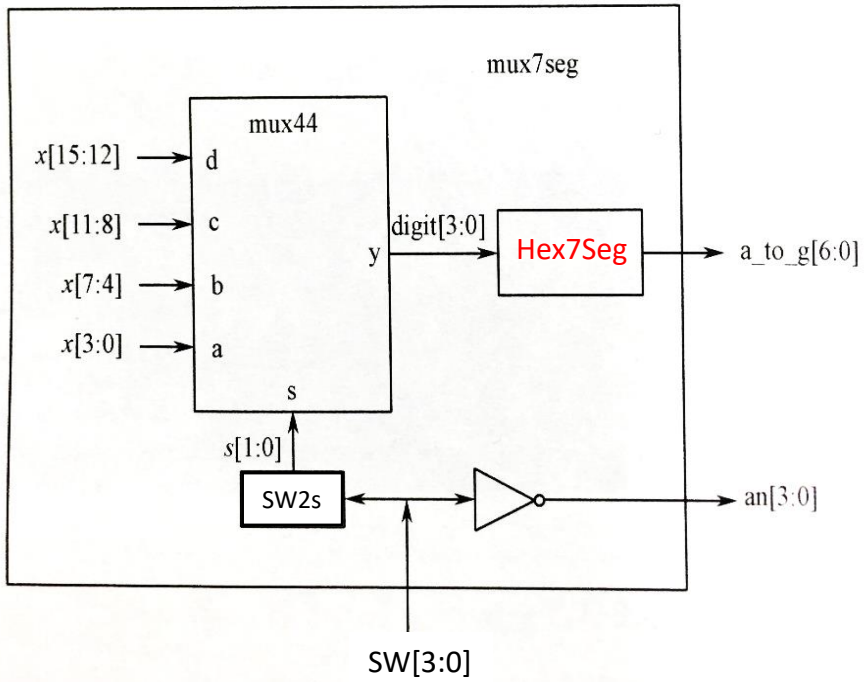
SW[3]	SW[2]	SW[1]	SW[0]	s[1]	s[0]
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

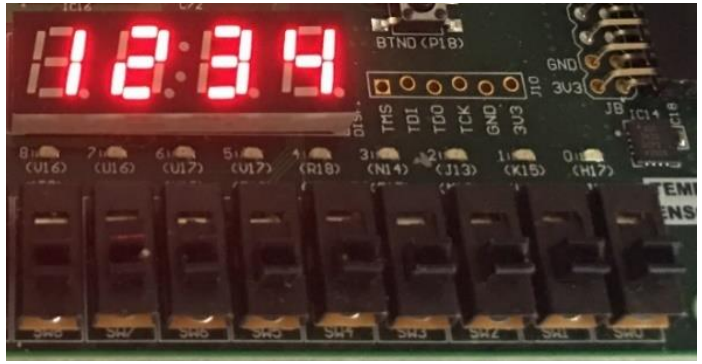



```
module mux7seg( input  logic [3:0] SW,
                output logic [6:0] A2G,
                output logic [3:0] AN,
                output logic       DP);

    logic [15:0] x;
    logic [3:0]  digit;

    assign x = 'h1234;
    assign AN = ~SW;
    assign DP = 1;     // DP off

    logic [1:0]  s;
    assign s[1] = SW[2] | SW[3];
    assign s[0] = SW[1] | SW[3];

    //4位 4选1 MUX: mux44
    always_comb
        case(s)
            0: digit = x[3:0];
            1: digit = x[7:4];
            2: digit = x[11:8];
            3: digit = x[15:12];
            default: digit = x[3:0];
        endcase

    //实例化 7段数码管
    Hex7Seg s7(.x(digit), .a2g(A2G));
endmodule
```

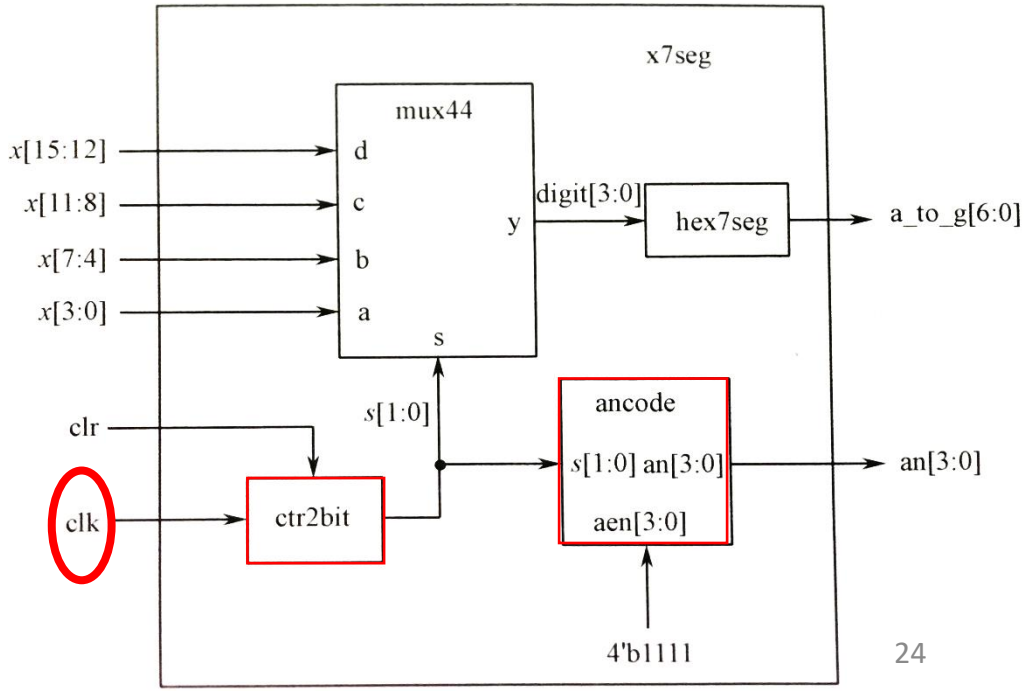
手工复用 vs 分时复用







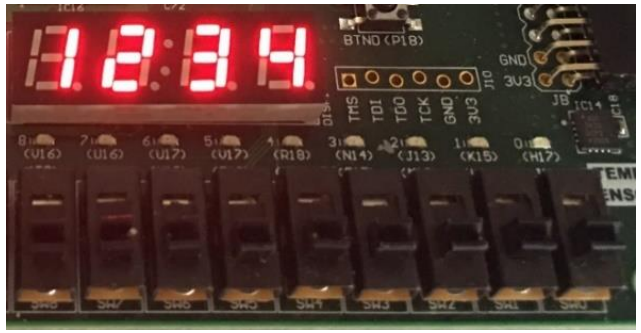
用clk快速、反复代替SW[3:0]



x7seg

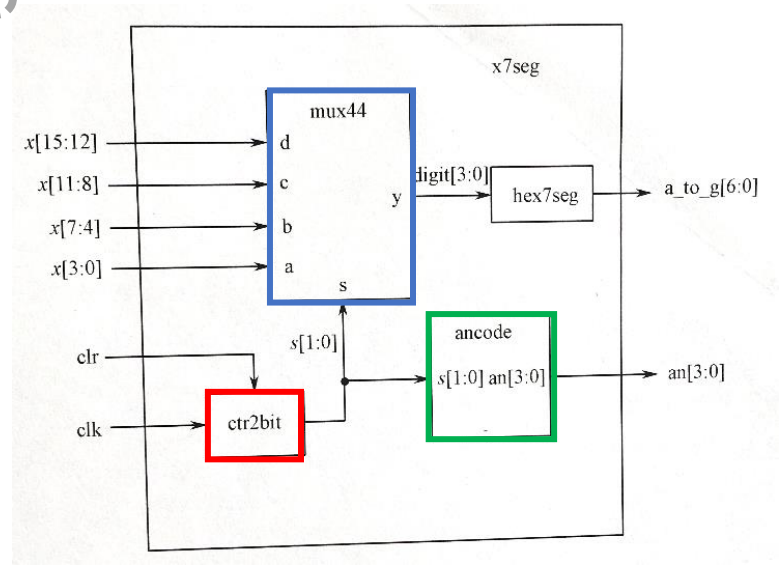
方案5：分时显示

```
module x7seg( input  logic [15:0] data,
              input  logic        clk,
              input  logic        clr,
              output logic [6:0]  a2g,
              output logic [3:0]  an,  //数码管使能
              output logic        dp ); //小数点

    logic [1:0] s;      //选择哪个数码管
    logic [3:0] digit;
    logic [19:0] clkdiv;

    assign dp = 1;              // DP off
    assign s = clkdiv[19:18];// 190Hz

    //4个数码管 4选1 (MUX44)
    always_comb
        case(s)
            0:  digit = data[3:0];
            1:  digit = data[7:4];
            2:  digit = data[11:8];
            3:  digit = data[15:12];
            default: digit = data[3:0];
        endcase
```



4个七段数码管分别显示，

每个数字每秒刷新190次




```
    //4个数码管的使能
    always_comb
        case(s)
            0:  an = 4'b1110;
            1:  an = 4'b1101;
            2:  an = 4'b1011;
            3:  an = 4'b0111;
            default: an = 4'b1110;
        endcase

    // 时钟分频器（20位二进制计数器）
    always @(posedge clk, posedge clr)
      if(clr == 1) clkdiv <= 0;
      else         clkdiv <= clkdiv + 1;

    //实例化 7段数码管
    Hex7Seg H7(.digit(digit), .a2g(a2g));
endmodule
```

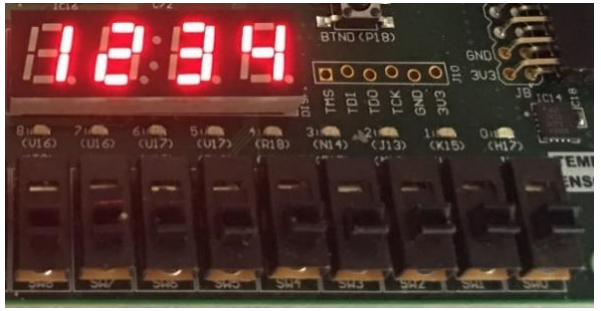
x7seg

方案5：分时显示

```
module x7seg( input  logic [15:0] data,
              input  logic        clk,
              input  logic        clr,
              output logic [6:0]  a2g,
              output logic [3:0]  an, //数码管使能
              output logic        dp ); //小数点

    logic [1:0] s;     //选择哪个数码管
    logic [3:0] digit;
    logic [19:0] clkdiv;

    assign dp = 1;           // DP off
    assign s = clkdiv[19:18];// count every 10.4ms

    //4个数码管 4选1 (MUX44)
    always_comb
        case(s)
            0:  digit = data[3:0];
            1:  digit = data[7:4];
            2:  digit = data[11:8];
            3:  digit = data[15:12];
            default: digit = data[3:0];
        endcase
```

```
module x7seg_Top(
    input  logic        CLK100MHZ,
    input  logic [0:0] SW,
    output logic [6:0] A2G,
    output logic [3:0] AN,
    output logic        DP );

    logic [15:0] x;
    assign x = 'h1234; //test value

    x7seg X7(.data(x),
             .clk (CLK100MHZ),
             .clr (SW[0]),
             .a2g (A2G),
             .an  (AN),
             .dp  (DP) );
endmodule
```

顶层文件

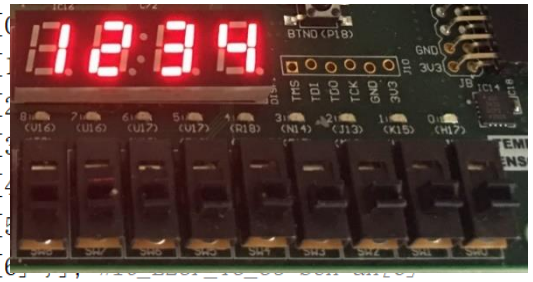


x7seg

方案5：分时显示

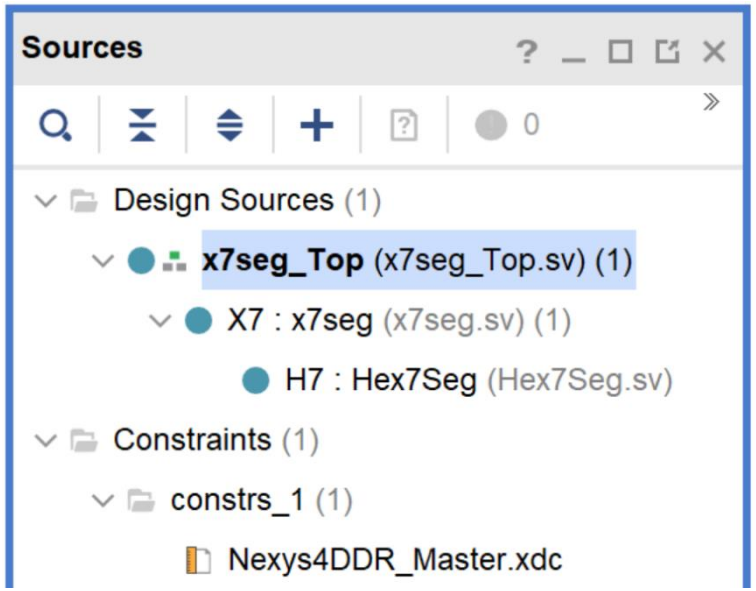
约束文件

```
## Clock signal
set_property -dict { PACKAGE_PIN E3    IOSTANDARD LVCMOS33 } [get_ports { CLK100MHZ }]; #IO_L12P_T1_MRCC_35 Sch=clk100mhz
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports {CLK100MHZ}];

##Switches
set_property -dict { PACKAGE_PIN J15   IOSTANDARD LVCMOS33 } [get_ports { SW[0] }]; #IO_L24N_T3_RS0_15 Sch=sw[0]
#set_property -dict { PACKAGE_PIN L16   IOSTANDARD LVCMOS33 } [get_ports { SW[1] }]; #IO_L3N_T0_DQS_EMCCLK_14 Sch=sw[1]
#set_property -dict { PACKAGE_PIN M13   IOSTANDARD LVCMOS33 } [get_ports { SW[2] }]; #IO_L6N_T0_D08_VREF_14 Sch=sw[2]
#set_property -dict { PACKAGE_PIN R15   IOSTANDARD LVCMOS33 } [get_ports { SW[3] }]; #IO_L13N_T2_MRCC_14 Sch=sw[3]

##7 segment display
set_property -dict { PACKAGE_PIN T10   IOSTANDARD LVCMOS33 } [get_ports { A2G[0] }]; #IO_L24N_T3_A00_D16_14 Sch=ca
set_property -dict { PACKAGE_PIN R10   IOSTANDARD LVCMOS33 } [get_ports { A2G[1] }]; #IO_25_14 Sch=cb
set_property -dict { PACKAGE_PIN K16   IOSTANDARD LVCMOS33 } [get_ports { A2G[2] }]; #IO_25_15 Sch=cc
set_property -dict { PACKAGE_PIN K13   IOSTANDARD LVCMOS33 } [get_ports { A2G[3] }]; #IO_L17P_T2_A26_15 Sch=cd
set_property -dict { PACKAGE_PIN P15   IOSTANDARD LVCMOS33 } [get_ports { A2G[4] }]; #IO_L13P_T2_MRCC_14 Sch=ce
set_property -dict { PACKAGE_PIN T11   IOSTANDARD LVCMOS33 } [get_ports { A2G[5] }]; #IO_L19P_T3_A10_D26_14 Sch=cf
set_property -dict { PACKAGE_PIN L18   IOSTANDARD LVCMOS33 } [get_ports { A2G[6] }]; #IO_L4P_T0_D04_14 Sch=cg
set_property -dict { PACKAGE_PIN H15   IOSTANDARD LVCMOS33 } [get_ports { DP }]; #IO_L19N_T3_A21_VREF_15 Sch=dp
set_property -dict { PACKAGE_PIN J17   IOSTANDARD LVCMOS33 } [get_ports { AN[
set_property -dict { PACKAGE_PIN J18   IOSTANDARD LVCMOS33 } [get_ports { AN[
set_property -dict { PACKAGE_PIN T9    IOSTANDARD LVCMOS33 } [get_ports { AN[
set_property -dict { PACKAGE_PIN J14   IOSTANDARD LVCMOS33 } [get_ports { AN[
set_property -dict { PACKAGE_PIN P14   IOSTANDARD LVCMOS33 } [get_ports { AN[
set_property -dict { PACKAGE_PIN T14   IOSTANDARD LVCMOS33 } [get_ports { AN[
set_property -dict { PACKAGE_PIN K2    IOSTANDARD LVCMOS33 } [get_ports { AN[
set_property -dict { PACKAGE_PIN U13   IOSTANDARD LVCMOS33 } [get_ports { AN[7] }]; #IO_L23N_T3_A02_D18_14 Sch=an[7]
```

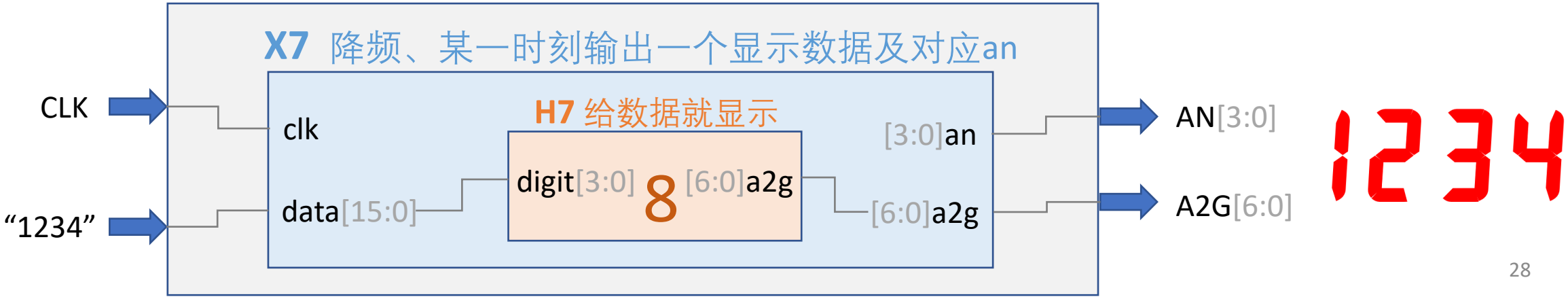


```
module x7seg_Top(
    input  logic       CLK100MHZ,
input  logic [0:0] SW,
utput logic [6:0] A2G,
utput logic [3:0] AN,
utput logic       DP );

logic [15:0] x;
assign x = 'h1234; //test value

x7seg X7(.data(x),
         .clk (CLK100MHZ),
         .clr (SW[0]),
         .a2g (A2G),
         .an  (AN),
         .dp  (DP) );
odule
```

方案5：分时显示



Top 与板子相连



时钟分频器

q(i)	频率 Hz	周期 ms	q(i)	频率 Hz	周期 ms
In	100M	0.000 01	12	12 207.03	0.081 92
0	50M	0.000 02	13	6 103.52	0.163 84
1	25M	0.000 04	14	3 051.76	0.327 68
2	12.5M	0.000 08	15	1 525.88	0.655 36
3	6.25M	0.000 16	16	762.94	1.310 72
4	3.125M	0.000 32	17	381.47	2.621 44
5	1.5625M	0.000 64	18	190.73	5.242 88
6	781.25K	0.001 28	19	95.37	10.485 76
7	390.625K	0.002 56	20	47.68	20.010 24
8	195.3125K	0.005 12	21	**23.84**	41.943 04
9	97 656.25	0.010 24	22	11.92	83.886 08
10	48 828.13	0.020 48	23	5.96	167.772 16
11	24 414.06	0.040 96	24	2.98	335.544 32

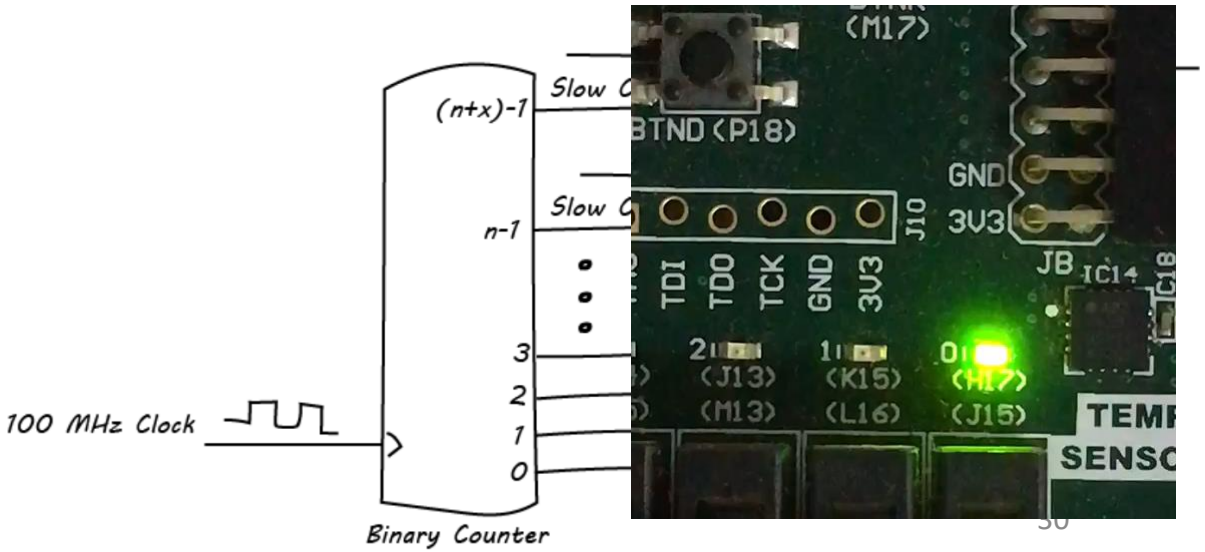
```
1  module clkdiv(
2      input mclk,
3      input clr,
4      output clk190,//190Hz
5      output clk48);// 48Hz
6
7      reg [24:0] q; //25位计数器
8
9      always @(posedge mclk or posedge clr)
10     begin
11       if(clr==1)
12          q <= 0;
13       else
14          q <= q + 1;
15     end
16
17     assign clk190 = q[18];
18     assign clk48  = q[20];
19 endmodule
```

blinkLED

闪烁的LED

q(i)	频率 Hz	周期 ms	q(i)	频率 Hz	周期 ms
In	100M	0.000 01	12	12 207.03	0.081 92
0	50M	0.000 02	13	6 103.52	0.163 84
1	25M	0.000 04	14	3 051.76	0.327 68
2	12.5M	0.000 08	15	1 525.88	0.655 36
3	6.25M	0.000 16	16	762.94	1.310 72
4	3.125M	0.000 32	17	381.47	2.621 44
5	1.5625M	0.000 64	18	190.73	5.242 88
6	781.25K	0.001 28	19	95.37	10.485 76
7	390.625K	0.002 56	20	47.68	20.010 24
8	195.3125K	0.005 12	21	23.84	41.943 04
9	97 656.25	0.010 24	22	11.92	83.886 08
10	48 828.13	0.020 48	23	5.96	167.772 16
11	24 414.06	0.040 96	24	2.98	335.544 32

```
1  module blinkLED(
2      input logic CLK100MHZ,
3      output logic [1:0] LED   );
4
5      logic [30:0] blinkCount;
6
7      always @(posedge CLK100MHZ)
8         blinkCount <= blinkCount + 1;
9
10     assign LED[0] = blinkCount[25]; //Slow
11     assign LED[1] = blinkCount[24]; //Fast
12 endmodule
```



参考资料

- Nexys4-DDR_Reference Manual.pdf

- Nexys4-DDR_sch.PDF

- lab1 Vivado Design Flow.pdf

- lab1 Vivado Design Flow.zip

引脚约束文件

- Nexys4DDR_Master.xdc

https://www.xilinx.com/

https://china.xilinx.com/

*扩展模块 PmodKYPD

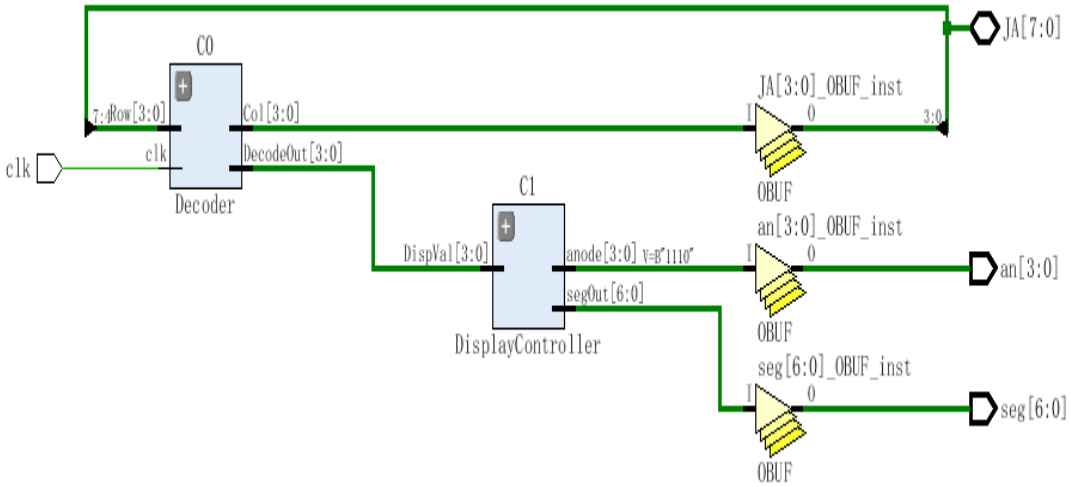
a 16 button keypad

Decoder Behavior

The Decoder determines which key, if any, was pressed on the PmodKYPD by cycling through each column pin with a logic low. After the Decoder sets a **column pin low**, it checks for **a logic low in the row pins**. A low in a row pin signifies that a button has been pressed. Once the Decoder has both the row and column of the key, it can determine the corresponding **value** to output to the **DisplayController**. The decoder will change columns every **1ms** in its cycle.

DisplayController Behavior

The DisplayController is used to display the output of the Decoder onto the seven segment display on a Nexys4 board. In this project only the rightmost digit on the seven segment display is used. Before any key was pressed, the seven segment display shows a '0' on the rightmost digit. The keypad is represented in hex values.



keyboard

*USB Keyboard

1) Initial state

Initially, the 7 segment display will show all 0‘s.

2) Key Press

When a button is pressed, the value of the scan code will shift onto the 7 segment display. In the picture, '**a**' is pressed, so a scan code of “**1C**” is displayed.

3) Key Release

When the 'a' key is released, a scan code of “F01C” is shifted onto the 7 segment display. The initial scan code of “1C” is shifted to the left display.

4) Multi key press

When multiple keys are pressed their scan codes are shifted in order. In this case, Q (“15”) was pressed, followed by W (“1D”).

ESC 76		F1 05	F2 06	F3 04	F4 0C		F5 03	F6 0B	F7 83	F8 0A		F9 01	F10 09	F11 78	F12 07
` ~ 0E	1 ! 16	2 @ 1E	3 # 26	4 $ 25	5 % 2E	6 ^ 36	7 & 3D	8 * 3E	9 (46 | 0) 45	- _ 4E	= + 55	BackSpace ← 66		
TAB 0D	Q 15	W 1D	E 24	R 2D	T 2C	Y 35	U 3C	I 43	O 44	P 4D	[{ 54	] } 5B	\ | 5D		
Caps Lock 58	A 1C	S 1B	D 23	F 2B	G 34	H 33	J 3B	K 42	L 4B	; : 4C	' " 52	Enter ↵ 5A			
Shift 12	Z 1Z	X 22	C 21	V 2A	B 32	N 31	M 3A	, < 41	. > 49	/ ? 4A	Shift 59				
Ctrl 14		Alt 11	Space 29							Alt E0 11		Ctrl E0 14			

Figure 9. Keyboard scan codes.