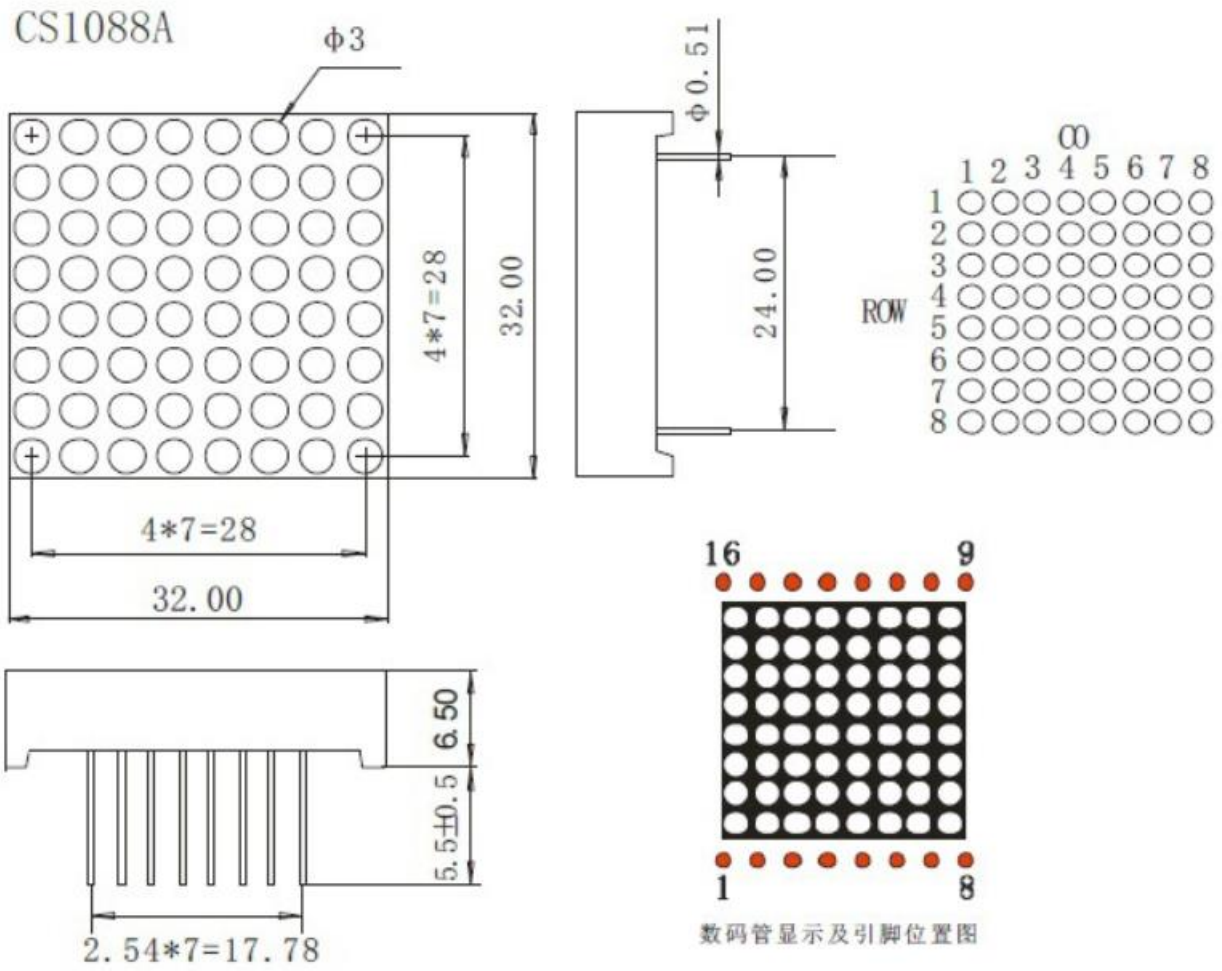
8x8点阵模块驱动

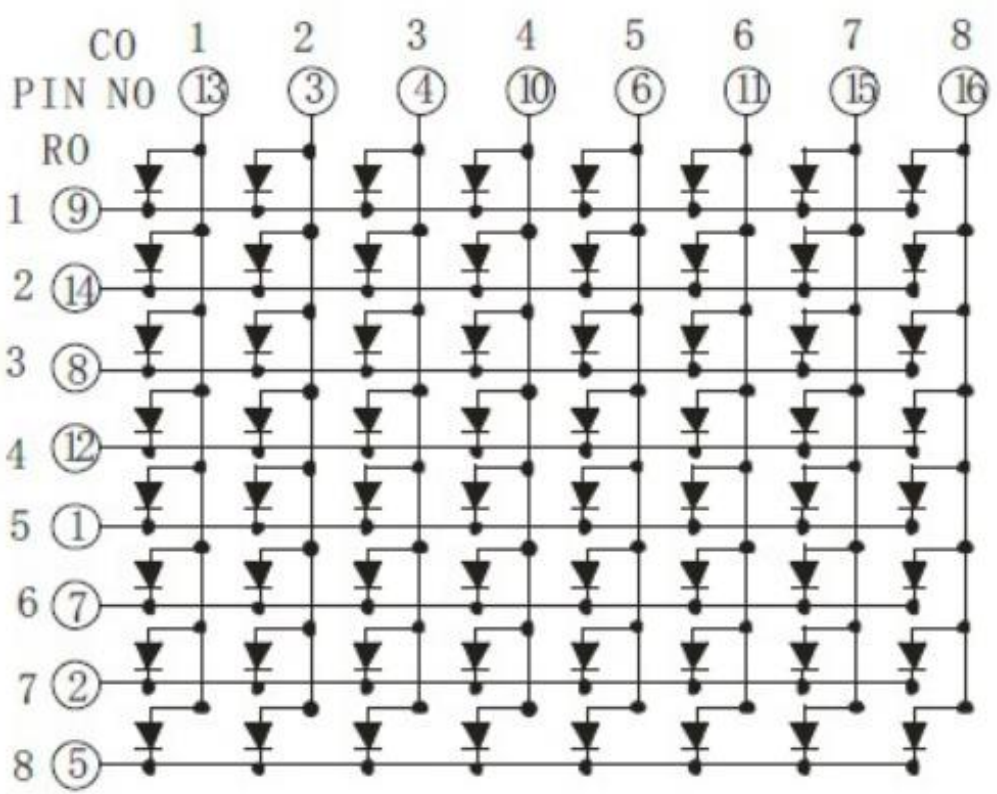
王安然

STEP FPGA

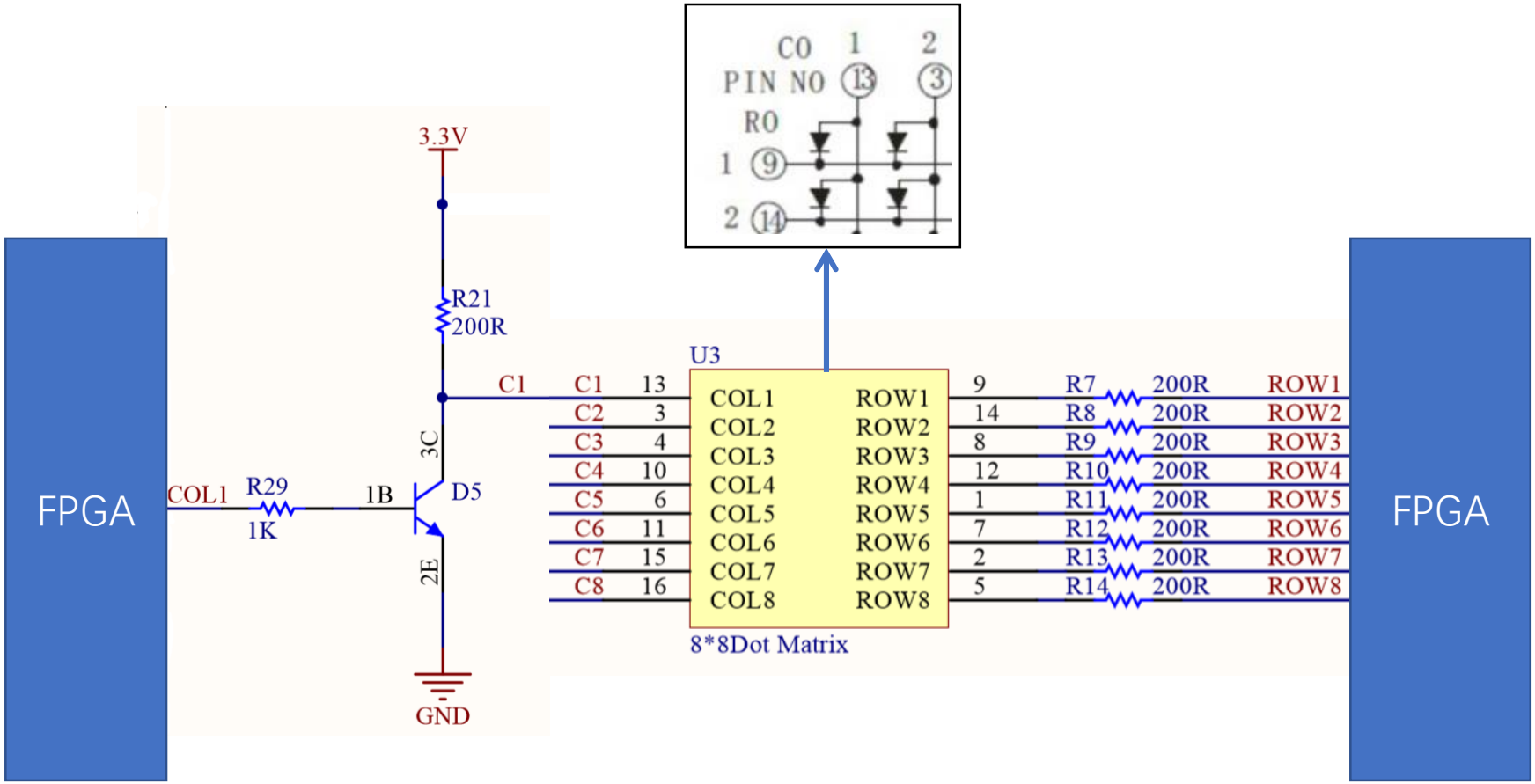
8x8点阵封装


CS1088A
φ3
φ0.51
4*7=28
32.00
24.00
4*7=28
32.00
COL
1 2 3 4 5 6 7 8
ROW
1
2
3
4
5
6
7
8
6.50
5.5±0.5
2.54*7=17.78
16
9
1
8
数码管显示及引脚位置图

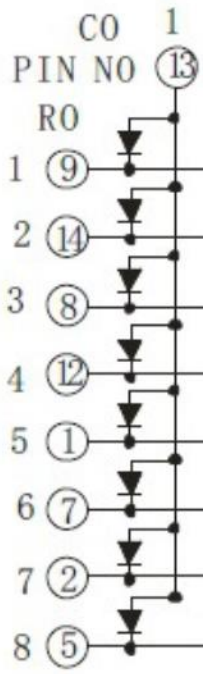
8x8点阵结构



点阵电路连接


FPGA
3.3V
R21
200R
C1
COL1
R29
1K
1B
3C
D5
2E
GND
U3
C1 13
C2 3
C3 4
C4 10
C5 6
C6 11
C7 15
C8 16
COL1
COL2
COL3
COL4
COL5
COL6
COL7
COL8
ROW1
ROW2
ROW3
ROW4
ROW5
ROW6
ROW7
ROW8
9
14
8
12
1
7
2
5
R7 200R ROW1
R8 200R ROW2
R9 200R ROW3
R10 200R ROW4
R11 200R ROW5
R12 200R ROW6
R13 200R ROW7
R14 200R ROW8
8*8Dot Matrix
C0 1 2
PIN NO 13 3
R0
1 9
2 14
FPGA

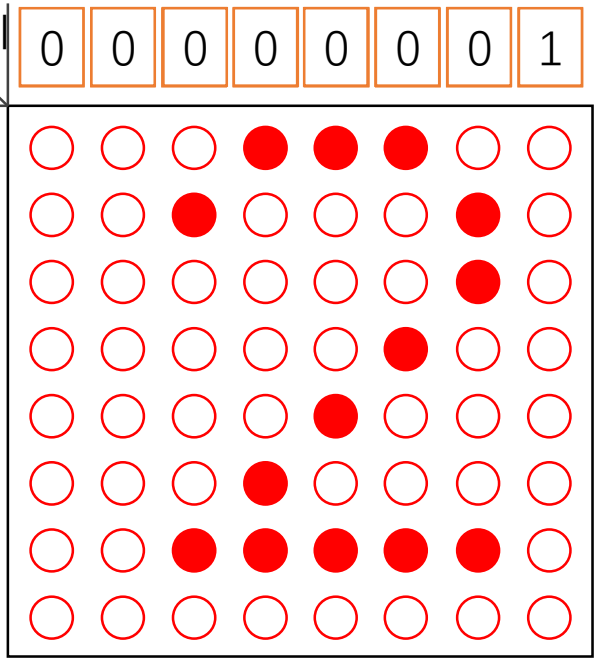
点阵扫描显示原理



2的字库数据

1	1	0	0	0	1	1	1
1	0	1	1	1	0	1	1
1	0	1	1	1	1	1	1
1	1	0	1	1	1	1	1
1	1	1	0	1	1	1	1
1	1	1	1	0	1	1	1
1	0	0	0	0	0	1	1
1	1	1	1	1	1	1	1

8个时刻，扫描显示



点阵扫描速率

点阵可以用扫描的方式显示，主要的原理是利用人眼的滞留效应，当闪烁的频率足够高时，人眼看到的就是连续的，例如当扫描频率等于100Hz时，则在人眼看到的点阵显示就是连续的。

整个点阵扫描频率为100Hz，周期为10ms，每一列的扫描周期为1.25ms，我们可以使用800Hz的时钟触发完成每一列的扫描。

```
reg [12:0] cnt;
always @(posedge clk or negedge rst_n)
if(!rst_n) cnt <= 1'b0;
else if(cnt >= 13'd7499) cnt <= 1'b0;
else cnt <= cnt + 1'b1;

reg clk_800hz;
always @(posedge clk or negedge rst_n)
if(!rst_n) clk_800hz <= 1'b0;
else if(cnt == 13'd7499) clk_800hz <= ~clk_800hz;
else clk_800hz <= clk_800hz;
```

字库存储器实现

使用取模软件将数字1~9转换成8X8字库，网上也有很多现成的字库数据，下面就是我们在网上找到的5X7的字库数据（适合共阳极点阵使用），我们可以调整成8X8字库数据，使用时按位取反即可。

```
reg [63:0] mem [9:0];
always @(negedge rst_n) begin
   mem[0]= {8'h00, 8'h00, 8'h3E, 8'h51, 8'h49, 8'h45, 8'h3E, 8'h00}; // 0
   mem[1]= {8'h00, 8'h00, 8'h00, 8'h42, 8'h7F, 8'h40, 8'h00, 8'h00}; // 1
   mem[2]= {8'h00, 8'h00, 8'h42, 8'h61, 8'h51, 8'h49, 8'h46, 8'h00}; // 2
   mem[3]= {8'h00, 8'h00, 8'h21, 8'h41, 8'h45, 8'h4B, 8'h31, 8'h00}; // 3
   mem[4]= {8'h00, 8'h00, 8'h18, 8'h14, 8'h12, 8'h7F, 8'h10, 8'h00}; // 4
   mem[5]= {8'h00, 8'h00, 8'h27, 8'h45, 8'h45, 8'h45, 8'h39, 8'h00}; // 5
   mem[6]= {8'h00, 8'h00, 8'h3C, 8'h4A, 8'h49, 8'h49, 8'h30, 8'h00}; // 6
   mem[7]= {8'h00, 8'h00, 8'h01, 8'h71, 8'h09, 8'h05, 8'h03, 8'h00}; // 7
   mem[8]= {8'h00, 8'h00, 8'h36, 8'h49, 8'h49, 8'h49, 8'h36, 8'h00}; // 8
   mem[9]= {8'h00, 8'h00, 8'h06, 8'h49, 8'h49, 8'h29, 8'h1E, 8'h00}; // 9
end
```

点阵扫描显示实现

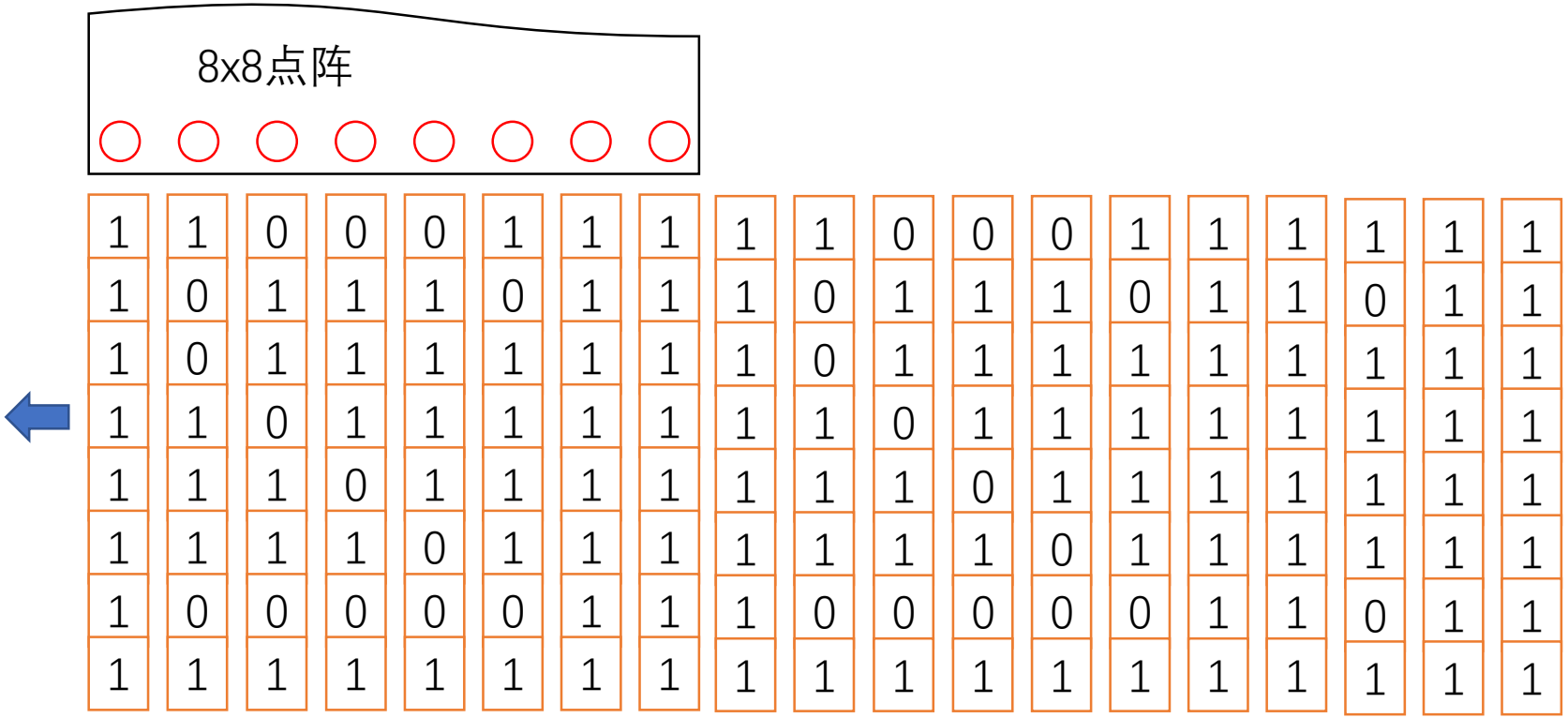
使用状态机完成点阵的扫描实现，这里我们用的一段式，将sw接口分配给FPGA核心板的拨码开关，波动拨码开关控制点阵显示数据。

```
input [3:0] sw;

reg [2:0] state;
always @(posedge clk_800hz or negedge rst_n)
if(!rst_n) begin col <= 8'hff; row = 8'hff; state <= s0; end
else
case(state)
  s0: begin col <= 8'b1111_1110; row = ~mem[sw][56+:8]; state <= s1; end
  s1: begin col <= 8'b1111_1101; row = ~mem[sw][48+:8]; state <= s2; end
  s2: begin col <= 8'b1111_1011; row = ~mem[sw][40+:8]; state <= s3; end
  s3: begin col <= 8'b1111_0111; row = ~mem[sw][32+:8]; state <= s4; end
  s4: begin col <= 8'b1110_1111; row = ~mem[sw][24+:8]; state <= s5; end
  s5: begin col <= 8'b1101_1111; row = ~mem[sw][16+:8]; state <= s6; end
  s6: begin col <= 8'b1011_1111; row = ~mem[sw][ 8+:8]; state <= s7; end
  s7: begin col <= 8'b0111_1111; row = ~mem[sw][ 0+:8]; state <= s0; end
  default: begin state <= s0; col <= 8'hff; row = 8'hff; end
endcase
```

点阵流动显示原理

字符串的字库可以由字符字库拼接组成，8x8点阵就像是一个窗口，想要流动显示只需要将字符串字库数据以列为单位依次左移。



字库存储器实现

字库数据的移位速度与点阵显示的移动速度成正比，本设计以5Hz的频率移位，需要设计产生5Hz的时钟信号，分频实现，这里就不多说了；将所有要显示的字符字库拼接成一组数据，然后以5Hz的频率左移，每次左移8位（对应一列）。

```
reg [479:0] mem_r;
always @(posedge clk_800hz or negedge rst_n)
 if(!rst_n)
   mem_r <= {mem[0],mem[1],mem[2],mem[3],mem[4],mem[5],mem[6],mem[7],mem[8],mem[9]};
 else if(clk_5hz)
   mem_r <= {mem_r[471:0],mem_r[479:472]};
 else
   mem_r <= mem_r;
```

Thanks

扫描二维码
关注小脚丫微信公众号
了解更多FPGA知识