数字逻辑设计

实验报告

循环计数器系统的分析与设计

组员 1 _____

组员 2 _____

组员 3 _____

组员 4 _____

电话 _____

日期 _____

一、项目目标

本项目的目标是实现一个四位的循环计数器系统，模拟二进制加减法运算，并在开发板上使用多个 LED 灯将计数器当前的值表现出来。

二、项目介绍

项目将实现一个简单的循环计数器系统。我们实践的是四位循环计数器，该计数器能够完成十进制数 16 以内，也就是四位二进制的加减法运算。在实际运算过程中，可能会减法运算后出现负数、加法运算后溢位的情况，因此我们将 0 与 15 相连，即对 0 进行减一操作会得到 15、对 15 做加一操作会得到 0，用这样的循环计数方法防止报错。

项目开发使用的硬件是 FuDan Micro 新型数字电路开发板，包括三个按键和 4 个 LED 灯。项目开发使用的软件是 Quartus 和 Procise，我们先对目标进行分析设计，在 Quartus 里进行 Verilog 代码编写，进行时序仿真及功能调试，再使用 Procise 软件进行编译和下载，最后在开发板上实现循环计数器的功能。

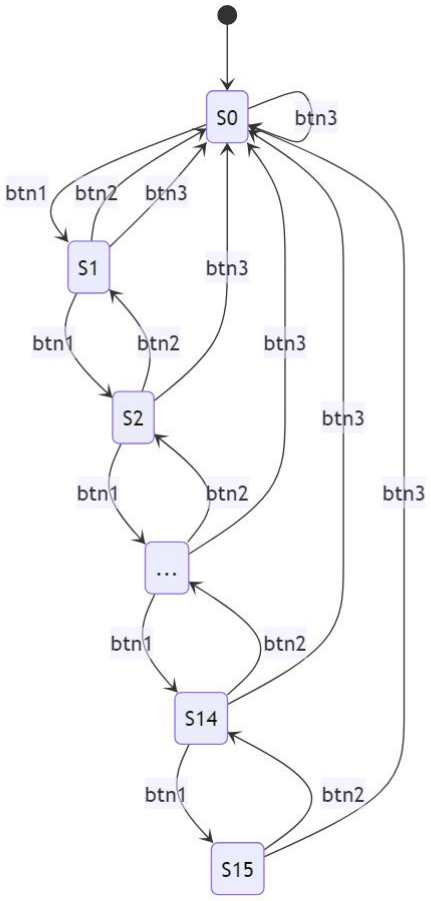
本次实验中，小组成员齐心协力，共同完成了从编写 Verilog 代码到实现在开发板上呈现正确结果的任务，每人大约分工 25%。

三、项目设计

我们小组设计的循环计数器系统需要四个输入信号，包括时钟信号 clk 及三个按键 btn1、btn2、btn3，btn1 每从低电平改为高电平一次，计数器存储的值加一，btn2 每从低电平改为高电平一次，计数器存储的值减一，btn3 每从低电平改为高电平一次，计数器存储的值清零；该系统还具有四个输出信号 led[0]、led[1]、led[2]、led[3]，分别对应循环计数器存储的二进制数的第 1 位、第二位、第三位、第四位。

该数字系统采用状态机的方式实现，拥有一个初始状态和十五个交替切换的状态，分别是 S0：（0000），S1：（0001），S2：（0010），S3：（0011），S4：（0100），S5：（0101），S6：（0110），S7：（0111），S8：（1000），S9：（1001），S10：（1010），S11：（1011），S12：（1100），S13：（1101），S14：（1110），S15：（1111）。系统开启后，将自动进入 S0 状态，此时计数器的值为 0000，若按下 btn1 按钮则计数器的值加一并进入 S1 状态，按下 btn2 按钮则会循环进入 S15 状态，系统处于其他状态时按下 btn1 也会进入对应二进制数加一的状态，按下 btn2 会进入对应二进制数减一的状态，按下 btn3 则会进行置零操作，无论处于任何状态都会进入 S0 状态。

状态图如下图所示：



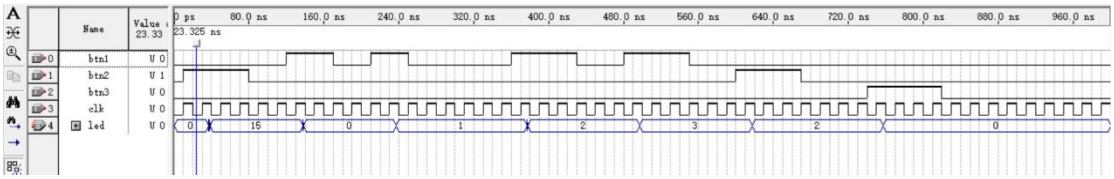
实现该数字系统的 verilog 代码如下：

```
module jishiqi (
  input clk, // 时钟信号
  input btn1, // 按键 1，用于+1
  input btn2, // 按键 2，用于-1
  input btn3, // 按键 3，用于归零
  output [3:0] led // 四个 LED 灯输出
);
reg [3:0] counter; // 计数器
reg btn1_d, btn2_d, btn3_d; // 按键状态延时寄存器，用于按键去抖
// 状态机状态定义
parameter S0 = 4'd0;
parameter S1 = 4'd1;
parameter S2 = 4'd2;
parameter S3 = 4'd3;
```

```
parameter S4 = 4'd4;
parameter S5 = 4'd5;
parameter S6 = 4'd6;
parameter S7 = 4'd7;
parameter S8 = 4'd8;
parameter S9 = 4'd9;
parameter S10 = 4'd10;
parameter S11 = 4'd11;
parameter S12 = 4'd12;
parameter S13 = 4'd13;
parameter S14 = 4'd14;
parameter S15 = 4'd15;
// 按键去抖
always @(posedge clk) begin
  btn1_d <= btn1;
    btn2_d <= btn2;
    btn3_d <= btn3;
end
// 状态机
always @(posedge clk) begin
  begin
    case (counter)
      S0: begin
        if (!btn3_d && btn3) begin
          counter <= 4'd15;
        end else if (!btn1_d && btn1) begin
          counter <= counter + 1;
        end else if (!btn2_d && btn2) begin
          counter <= counter - 1;
        end else begin
```

```
        end
      end
      S1: begin
        if (!btn3_d && btn3) begin
          counter <= 4'd15;
        end else if (!btn1_d && btn1) begin
          counter <= counter + 1;
        end else if (!btn2_d && btn2) begin
          counter <= counter - 1;
        end else begin
        end
      end
      // 同理为其他状态
      S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12, S13, S14, S15: begin
        if (!btn3_d && btn3) begin
          counter <= 4'd15;
        end else if (!btn1_d && btn1) begin
          counter <= counter + 1;
        end else if (!btn2_d && btn2) begin
          counter <= counter - 1;
        end else begin
        end
      end
    endcase
  end
end
// LED 输出
assign led = counter;
endmodule
```

仿真结果如下：



可以看出，仿真得到的时序图与预期结果一致，编写的 Verilog 代码可以实现项目所需的功能。

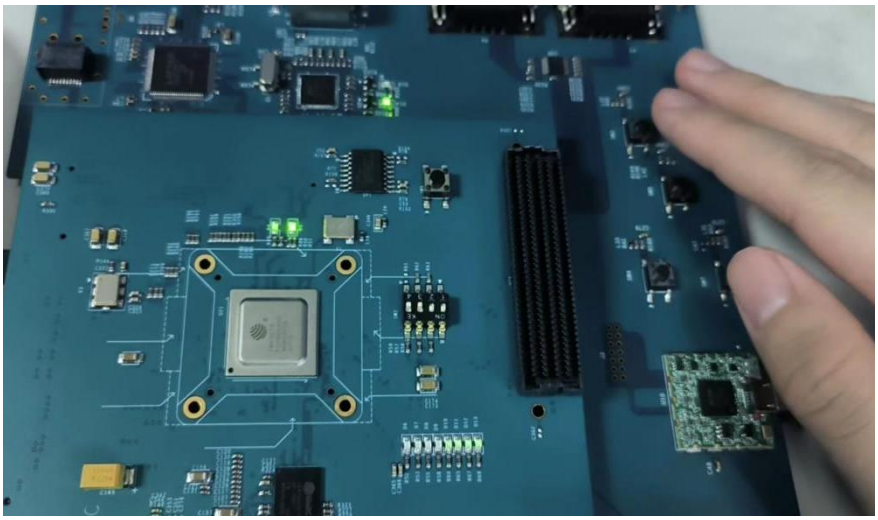
根据老师所给的 FuDan Micro 新型数字电路开发板的原理图，我们找到了实验所需的三个按钮及时钟信号的引脚，并基于此编写了约束文件，约束文件的代码如下：

```
NET "clk" LOC=W17;

NET "led[0]" LOC=M17;
NET "led[1]" LOC=N17;
NET "led[2]" LOC=P18;
NET "led[3]" LOC=P19;

NET "btn1" LOC=T20;
NET "btn2" LOC=U20;
NET "btn3" LOC=T19;

NET "clk" TNM_NET=clk;
TIMESPEC TS_clk=PERIOD "clk" 10 ms HIGH 50%;
```

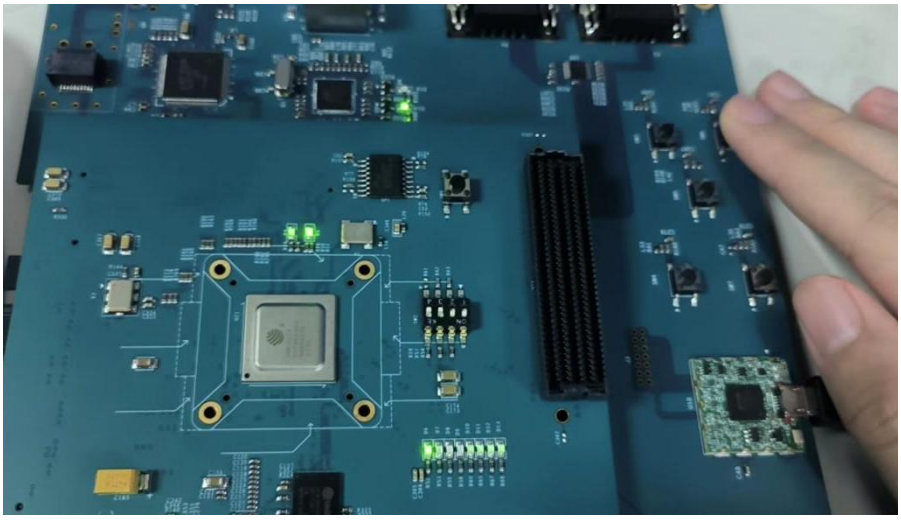
四、结果分析与讨论

完成编译综合和编译下载，在开发板上观察到的实验结果如下：

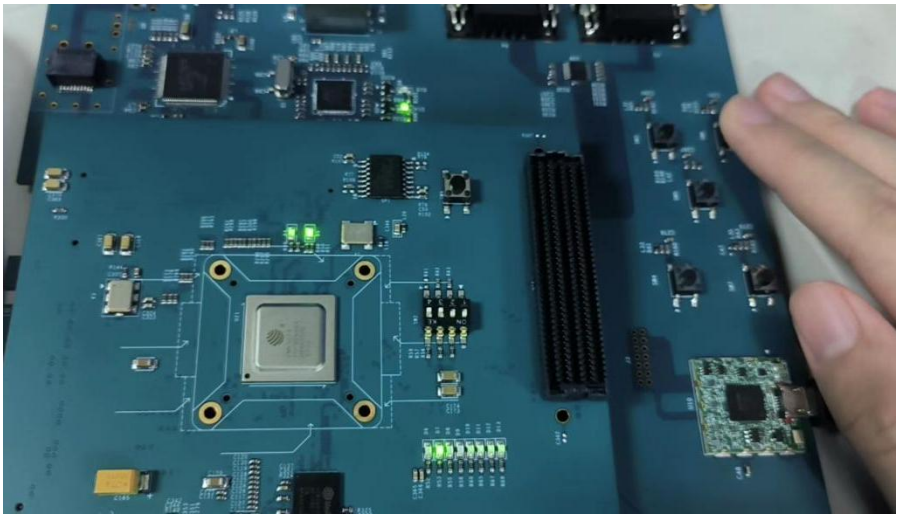
初始状态 S0 状态下，所有 LED 灯均熄灭：



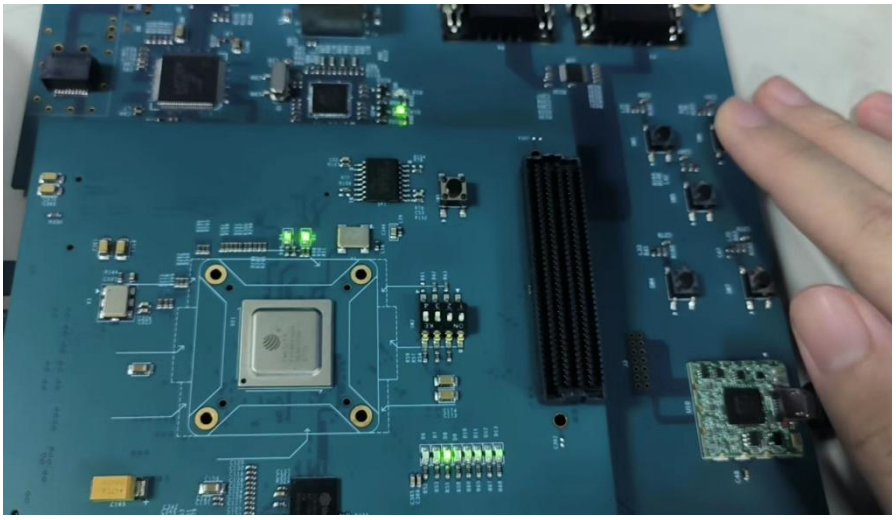
按下 btn1 按钮，进入 S1 状态，对应计数器二进制数第一位的 LED 进入常亮状态：



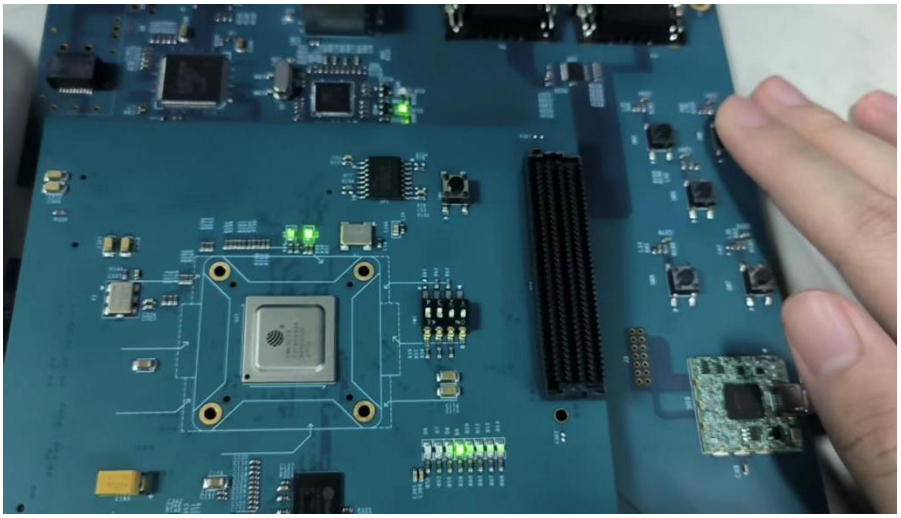
再次按下 btn1 按钮，进入 S2 状态，对应计数器二进制数第二位的 LED 进入常亮状态：



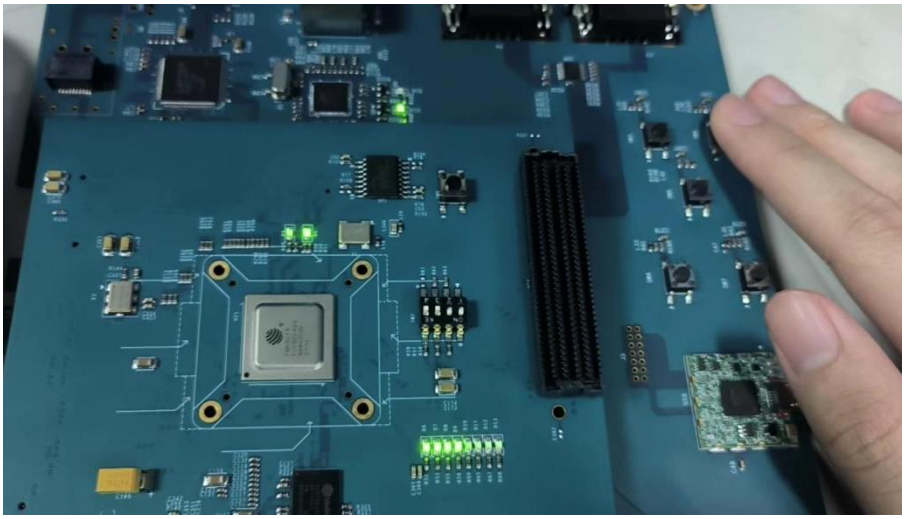
再按两次 btn1 按钮，进入 S4 状态，对应计数器二进制数第三位的 LED 进入常亮状态：

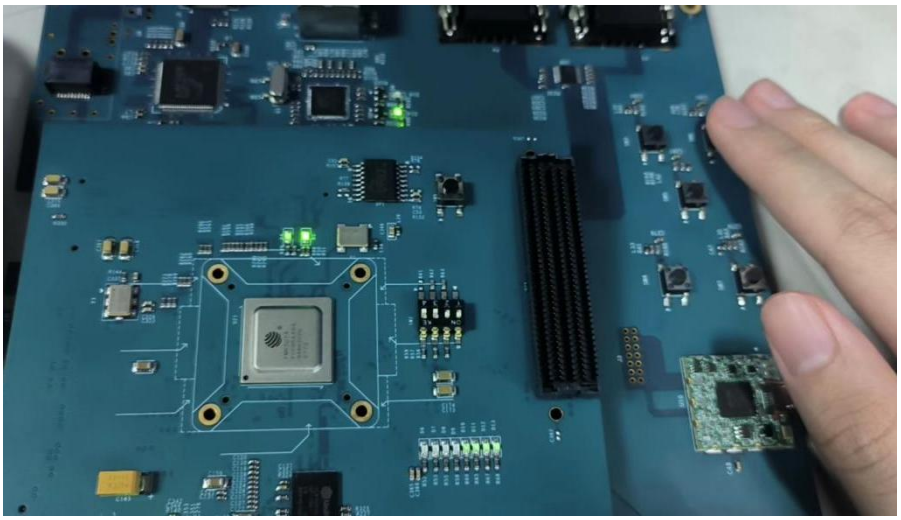


再按四次 btn1 按钮，进入 S8 状态，对应计数器二进制数第四位的 LED 进入常亮状态：



当处于 S15 状态时，由于循环计数器的计数范围是 0~15,故再次按下 btn1 按钮时转为 S0 状态：

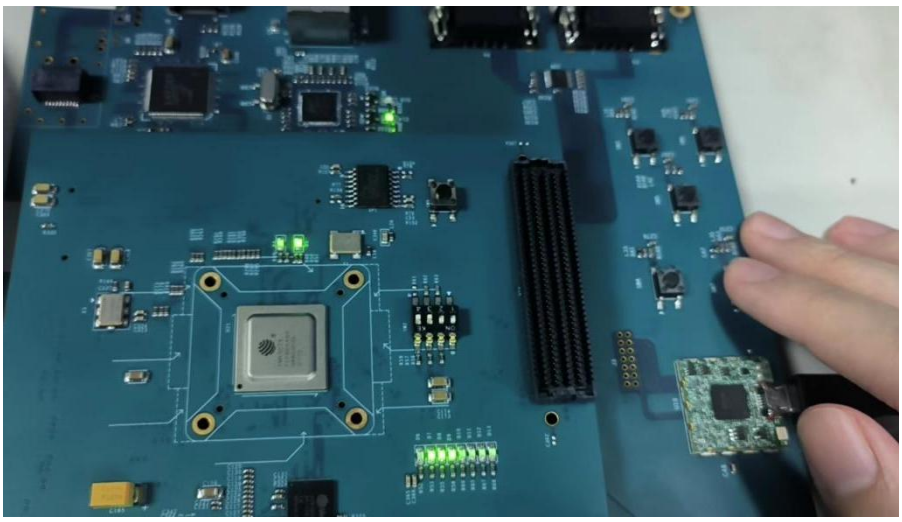




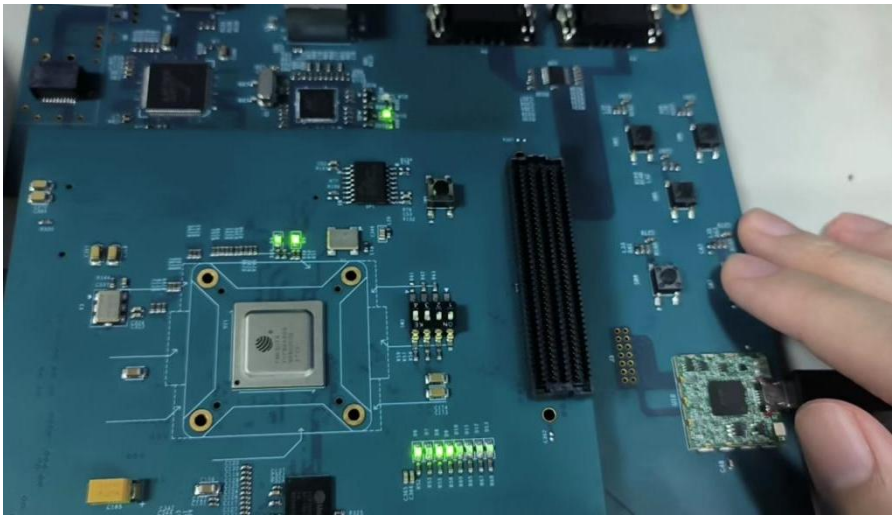
由上验证了实验用到的每一个 LED 灯均可正常运行，且加法部分

正常运行。

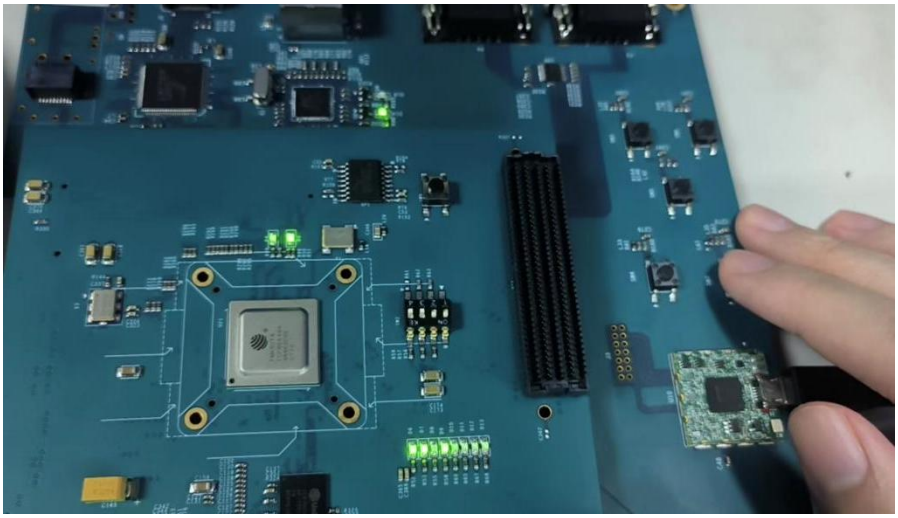
减法部分的验证类似，在 S15 状态按下 btn2 后进入 S14 状态，对应计数器二进制数第一位的 LED 熄灭：



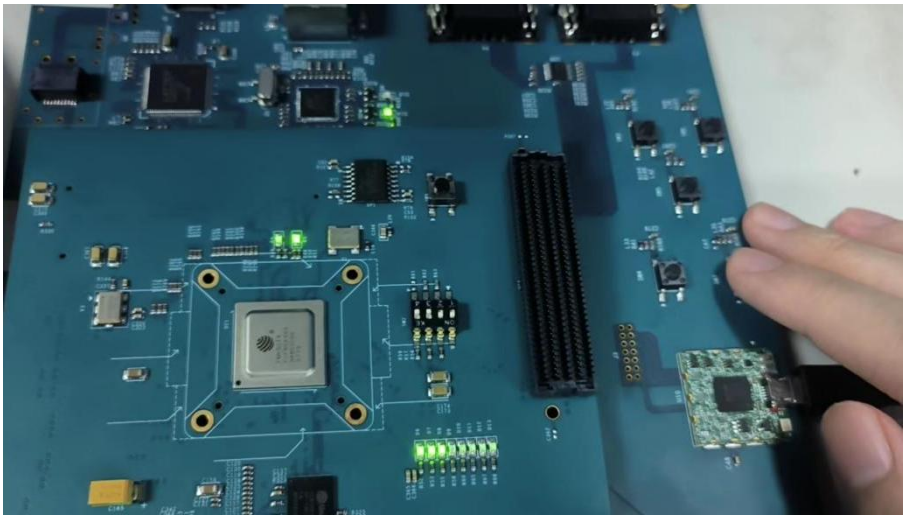
再次按下 btn2 按钮，进入 S13 状态，对应计数器二进制数第二位的 LED 熄灭：



再按两次 btn2 按钮，进入 S11 状态，对应计数器二进制数第三位的 LED 熄灭：

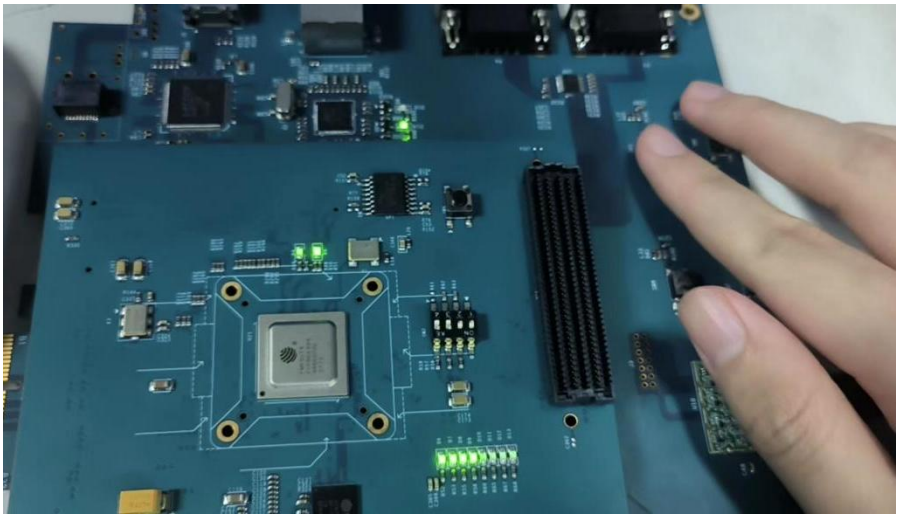


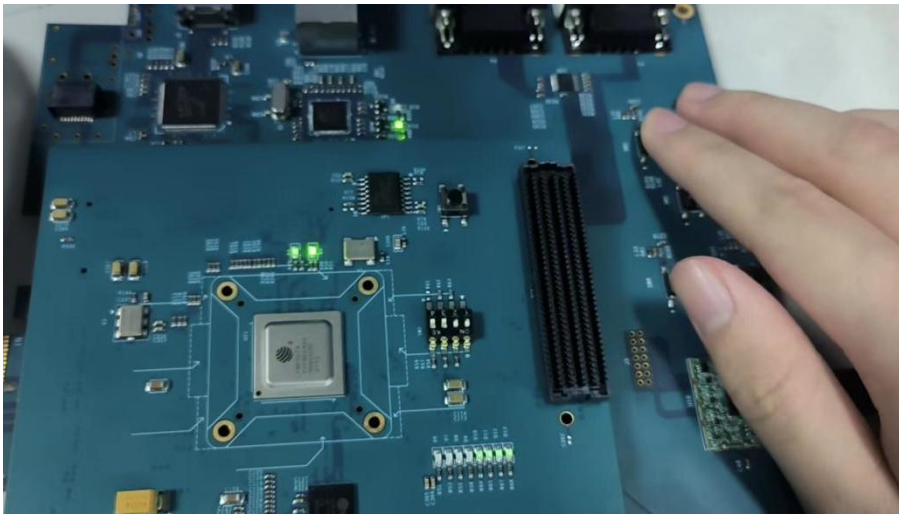
再按四次 btn2 按钮，进入 S7 状态，对应计数器二进制数第四位的 LED 熄灭：



至此计数器减法部分的正确性也得以验证。

将系统状态调至除 S0 以外的任一状态，本文以 S15 状态为例，按下 btn3 按钮，计数器置零，无论在任何状态都会进入 S0 状态：





综上所述，实验结果与预期结果相符，开发板上的 LED 灯根据预期点亮方式点亮，重置按键可以正常工作，循环计数器系统得到实现。

五、项目总结

经过长达 8 个小时的实验，我们小组独立设计了一个四位循环计数器，并运用到 FuDan Micro 新型数字电路开发板中，该循环计数器能够通过使用者按下对应加减法信号的按钮进行计数，也能通过按下重置按钮重置计数器。从编写选择项目方向开始，到最终在开发板上实现计数器这一功能的过程中，我们小组遇到了诸如长时间无法编写

出正确的 Verilog 代码并完成时序仿真（经过大量的讨论与修改代码后成功编写出了理想情况的代码）、仿真文件正常运行但将代码下载到开发板后开发板的 LED 灯处于常暗状态且并不会因为按下按键而发生改变（后发现是因为重置信号始终处于低电平，改变信号电平后问题得以解决）等很多问题，但小组成员没有放弃，齐心协力、共同思考，最终终于在给定的实验时间内完成了本次实验。美中不足的是，由于时间原因，我们实现的循环加速器只能对四位及以内的二进制数进行计数，但通过课后实践，我们发现要想实现更多位数的计数器，仅需增加 Verilog 代码中的状态，并在 Procise 里的约束文件中找到对应 LED 灯的引脚对其进行相关约束即可，计数器本身的原理并未发生改变，因此理论上使用我们的代码实现更高位的计数器是没有问题的。最后，感谢张老师在实验期间对我们小组的帮助，感谢张老师的悉心教导，让我们小组能够顺利完成本次实验！