同济大学计算机系

数字逻辑课程综合实验报告

学　　号 2352595

姓　　名 张嘉麟

专　　业 计算机科学与技术

授课老师 郭玉臣

一、实验内容

1.4 位比较器的设计与实现

设计一个 4 位比较器，能够比较两个 4 位二进制数 iData_a 和 iData_b。
当 iData_a 大于 iData_b 时，输出 oData 应为 3'b100。
当 iData_a 小于 iData_b 时，输出 oData 应为 3'b010。
当 iData_a 等于 iData_b 时，输出 oData 应等于输入 iData。

2.8 位比较器的设计与实现

设计一个 8 位比较器，用于比较两个 8 位二进制数 iData_a 和 iData_b。
输出 oData 应为 3'b100 表示 iData_a 大于 iData_b，3'b010 表示 iData_a 小于 iData_b，3'b001 表示两者相等。

3.无符号加法器的设计与实现

设计一个单比特全加器模块 FA，该模块可以处理三个输入（两个加数和一个进位），并输出一个和以及一个进位。
使用多个 FA 模块构建一个多位加法器，用于计算两个无符号数的和。

4.有符号加法器的设计与实现

在无符号加法器的基础上，扩展设计以支持有符号数的加法运算。
该模块同样需要处理两个 8 位输入 iData_a 和 iData_b，以及一个进位输入 iC，输出为 8 位结果 oData 和最终的进位 oData_C。

二、硬件逻辑图

（实验步骤中要求用 logisim 画图的实验，在该部分给出 logisim 原理图，否则该部分在实验报告中不用写）

三、模块建模

（该部分要求对实验中建模的所有模块进行功能描述，并列出各模块建模的 verilog 代码）

1.4 位比较器

```
module DataCompare4(
    input [3:0] iData_a,
    input [3:0] iData_b,
    input [2:0] iData,
    output reg [2:0] oData
```

```
);
    always@(*)
    begin
        if(iData_a > iData_b)
            oData = 3'b100;
        else if(iData_a<iData_b)
            oData = 3'b010;
        else
            oData = iData;
    end
endmodule
```

2.8 位比较器

```
module DataCompare8(
    input[7:0] iData_a,
    input[7:0] iData_b,
    output reg[2:0] oData
    );


    always@(*)
    begin
        if(iData_a > iData_b)
            oData = 3'b100;
        else if(iData_a < iData_b)
            oData = 3'b010;
        else
            oData = 3'b001;
    end

endmodule
```

3.无符号加法器

```
module FA(
    input iA,
    input iB,
    input iC,
    output oS,
    output oC
    );
```

```
    wire AandB, AxorB, Cand;

    xor (AxorB, iA, iB);
    and (Cand, iC, AxorB);
    and (AandB, iA, iB);

    or(oC, AandB, Cand);
    xor(oS, iC, AxorB);

endmodule
```

4.有符号加法器

```
module FA(
    input iA,
    input iB,
    input iC,
    output oS,
    output oC
    );

    wire AandB, AxorB, Cand;

    xor (AxorB, iA, iB);
    and (Cand, iC, AxorB);
    and (AandB, iA, iB);

    or(oC, AandB, Cand);
    xor(oS, iC, AxorB);

endmodule

module Adder(
    input[7:0] iData_a,
    input[7:0] iData_b,
    input iC,
    output[7:0] oData,
    output oData_C
    );

    wire [6:0]carrybit;
    FA FA0(iData_a[0], iData_b[0], iC, oData[0], carrybit[0]);
```

```
    FA FA1(iData_a[1], iData_b[1], carrybit[0], oData[1], carrybit[1]);
    FA FA2(iData_a[2], iData_b[2], carrybit[1], oData[2], carrybit[2]);
    FA FA3(iData_a[3], iData_b[3], carrybit[2], oData[3], carrybit[3]);
    FA FA4(iData_a[4], iData_b[4], carrybit[3], oData[4], carrybit[4]);
    FA FA5(iData_a[5], iData_b[5], carrybit[4], oData[5], carrybit[5]);
    FA FA6(iData_a[6], iData_b[6], carrybit[5], oData[6], carrybit[6]);
    FA FA7(iData_a[7], iData_b[7], carrybit[6], oData[7], oData_C);

endmodule
```

四、测试模块建模

（要求列写各建模模块的 test bench 模块代码）

1.4 位比较器

```
module DataCompare4_tb();
    reg [3:0] iData_a;
    reg [3:0] iData_b;
    reg [2:0] iData;
    wire [2:0] oData;

    initial
    begin
        iData_a = 4'b1111;
        iData_b = 4'b0000;
        iData = 3'b000;
        #5;
        iData_a = 4'b0000;
        iData_b = 4'b1111;
        #5;
        iData_a = 4'b1100;
        iData_b = 4'b1100;
        iData = 3'b100;
        #5;
        iData = 3'b010;
        #5;
        iData = 3'b001;
        #5;
    end

    DataCompare4 DataCompare4_inst(iData_a, iData_b, iData, oData);
```

```
endmodule
```

2.8 位比较器

```
module DataCompare8_tb();
    reg[7:0] iData_a;
    reg[7:0] iData_b;
    wire [2:0] oData;

    initial
    begin
        iData_a = 8'b11110000;
        iData_b = 8'b00001111;
        #20;
        iData_a = 8'b01010101;
        iData_b = 8'b10101010;
        #20;
        iData_a = 8'b11001100;
        iData_b = 8'b11001100;
        #20;
    end

    DataCompare8 DataCompare8_inst(iData_a, iData_b, oData);

endmodule
```

3.无符号加法器

```
module FA_tb;
    wire iA, iB, iC;
    wire oS, oC;
    reg [3 : 0] ABC;
    assign {iA, iB, iC} = ABC[2 : 0];

    initial
    begin
    for(ABC = 0; ABC <= 3'b111; ABC = ABC + 1)
        #5;
    end

    FA FA_inst(iA, iB, iC, oS, oC);
```

```
endmodule
```

4.有符号加法器

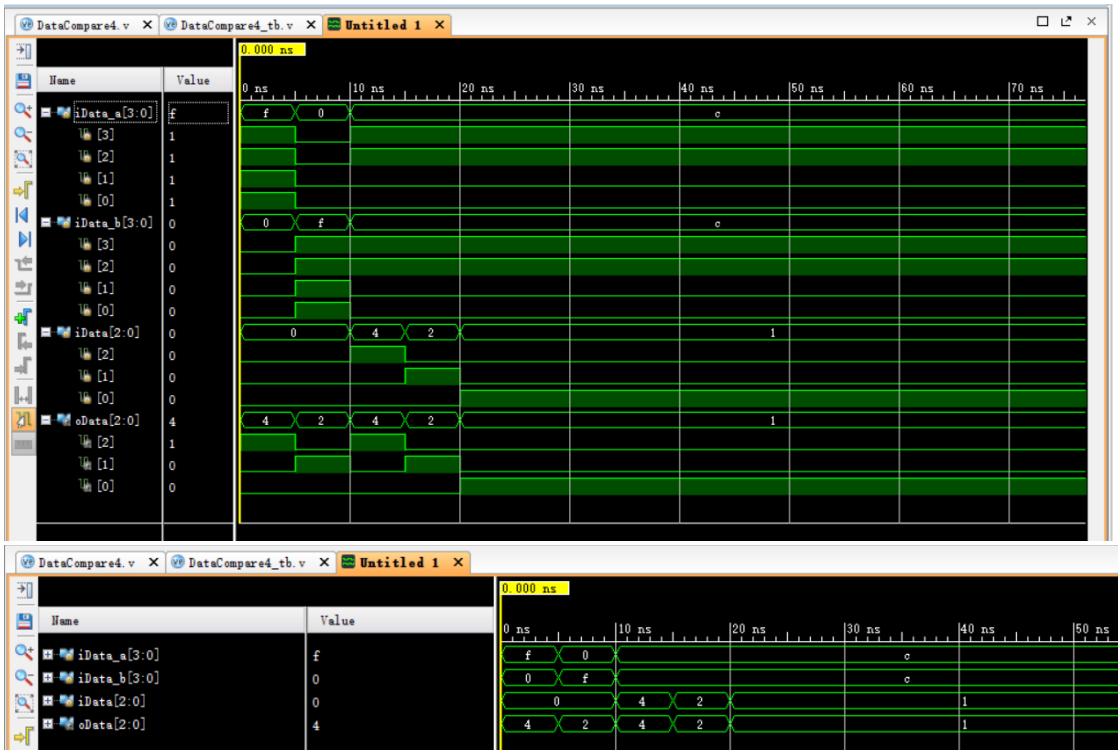
```
module Adder_tb();
    reg[8:0] iData_a;
    reg[8:0] iData_b;
    reg[1:0] iC;
    wire[7:0] oData;
    wire oData_C;

    initial
    begin
    for(iData_a = 8'b0000_1100; iData_a <= 8'b1111_1111; iData_a = iData_a + 2'b010)
        for(iData_b = 8'b1111_0000; iData_b <= 8'b1111_1111; iData_b = iData_b + 2'b010)
            for(iC = 0; iC <= 1'b1; iC = iC + 1)
                #5;
    end

    Adder Adder_inst(iData_a[7:0], iData_b[7:0], iC[0], oData, oData_C);
endmodule
```

五、实验结果

（该部分可截图说明，要求 logisim 逻辑验证图、modelsim 仿真波形图、以及下板后的实验结果贴图（实验步骤中没有下板要求的实验，不需要下板贴图））

modelsim 仿真波形图

1.4 位比较器

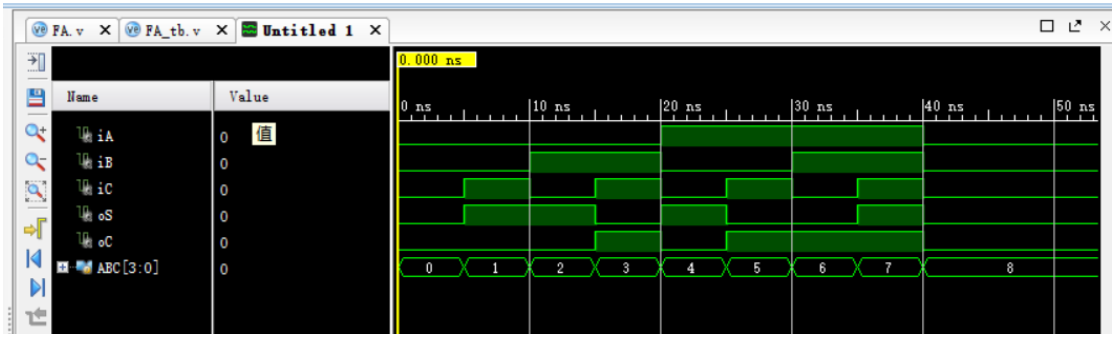


观察到当 a>b 时，输出为 3’b100；当 a<b 时，输出为 3’b010；当 a=b 时，输出与级联输入相同，分别为 3’b100、3’b010、3’b001。

2.8 位比较器

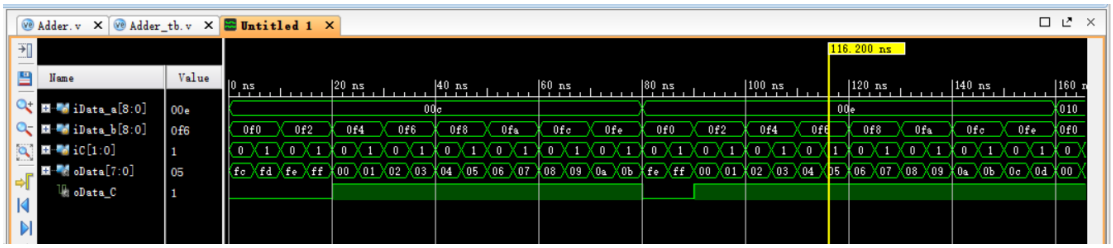
观察到当 a>b 时，输出为 3’b100；当 a<b 时，输出为 3’b010；当 a=b 时，输出为 3’b001。

3.无符号加法器



观察到 oC、oS 分别为加法结果的向高位进位与该位和结果，符合预期。

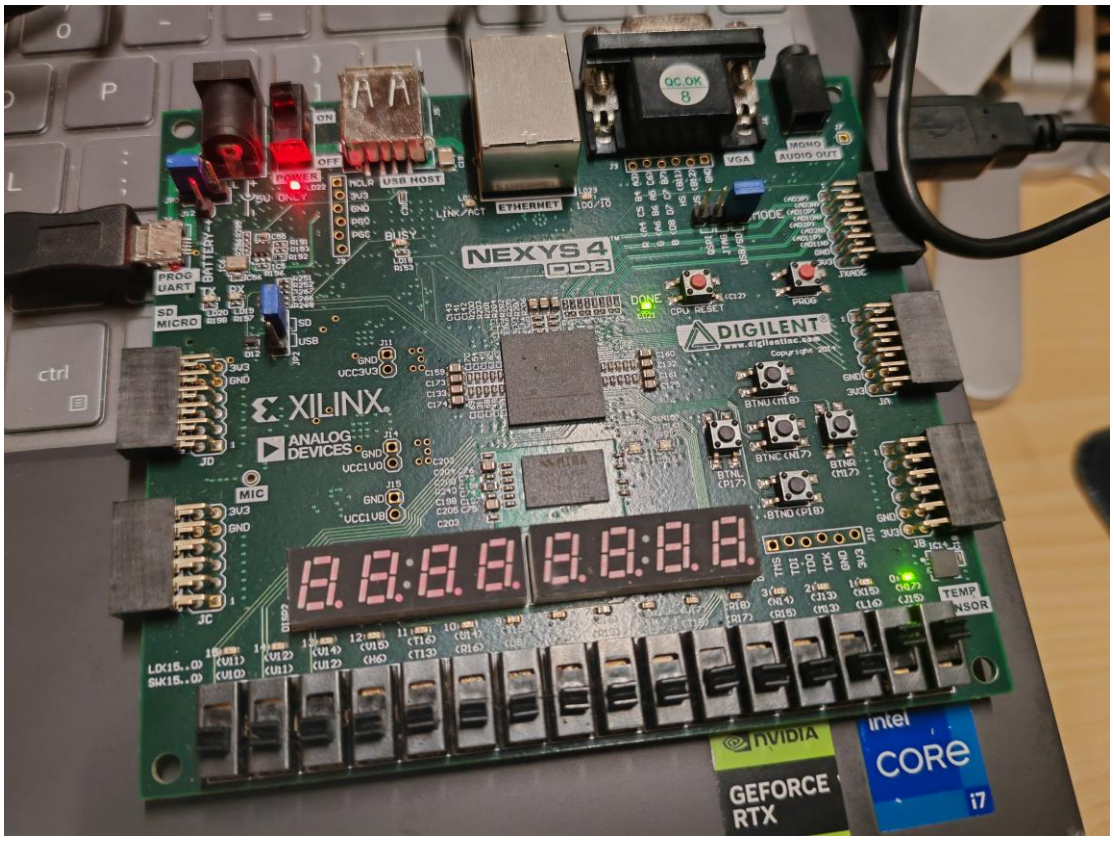
4.有符号加法器



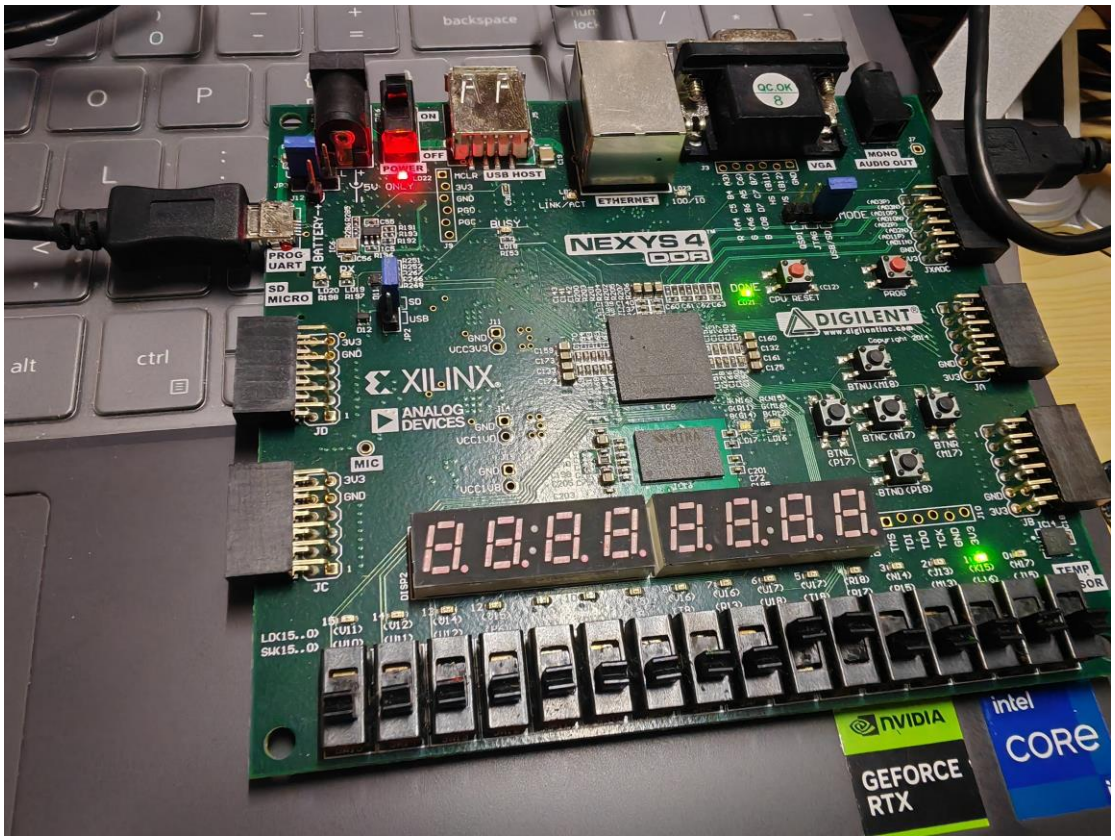
oData,oData_C 分别于 8 为 a 与 b 假发后得到的结果与向高位进位值，仿真正确。

三，下板结果

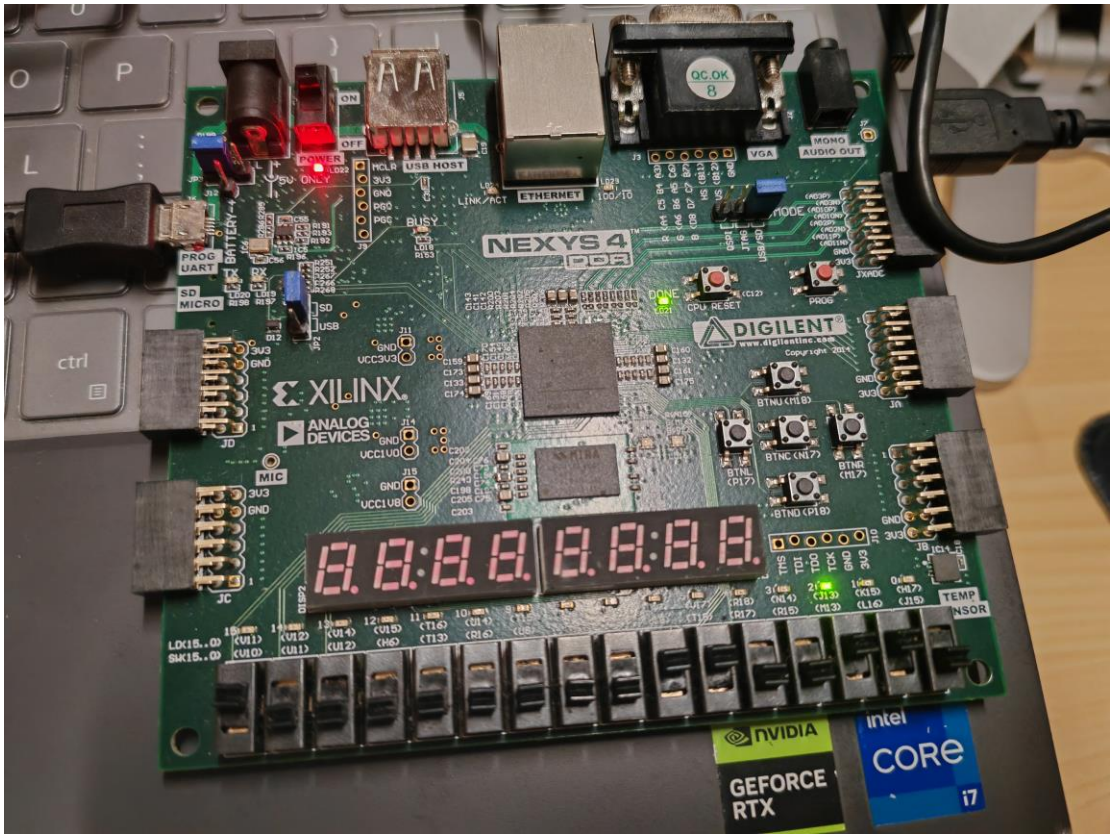
1.4 位比较器



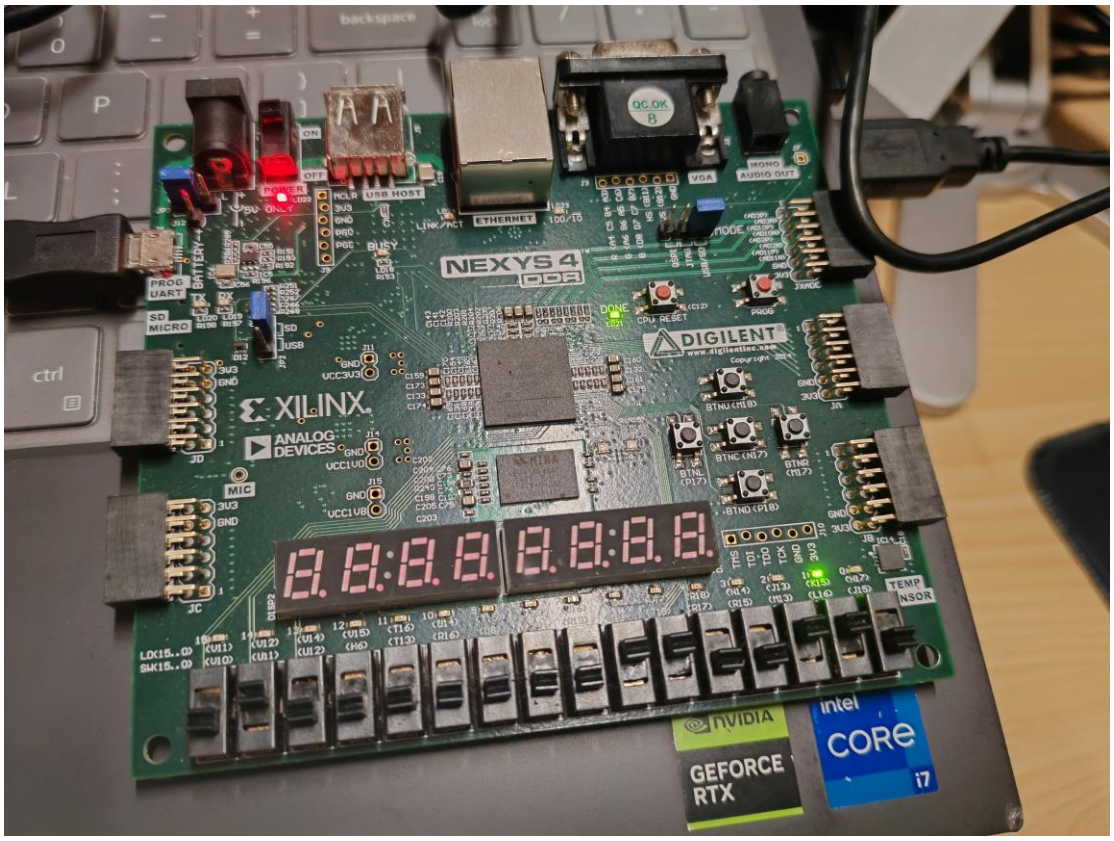
当仅有 L16、J15 打开时，a>b；仅 H17 亮起。



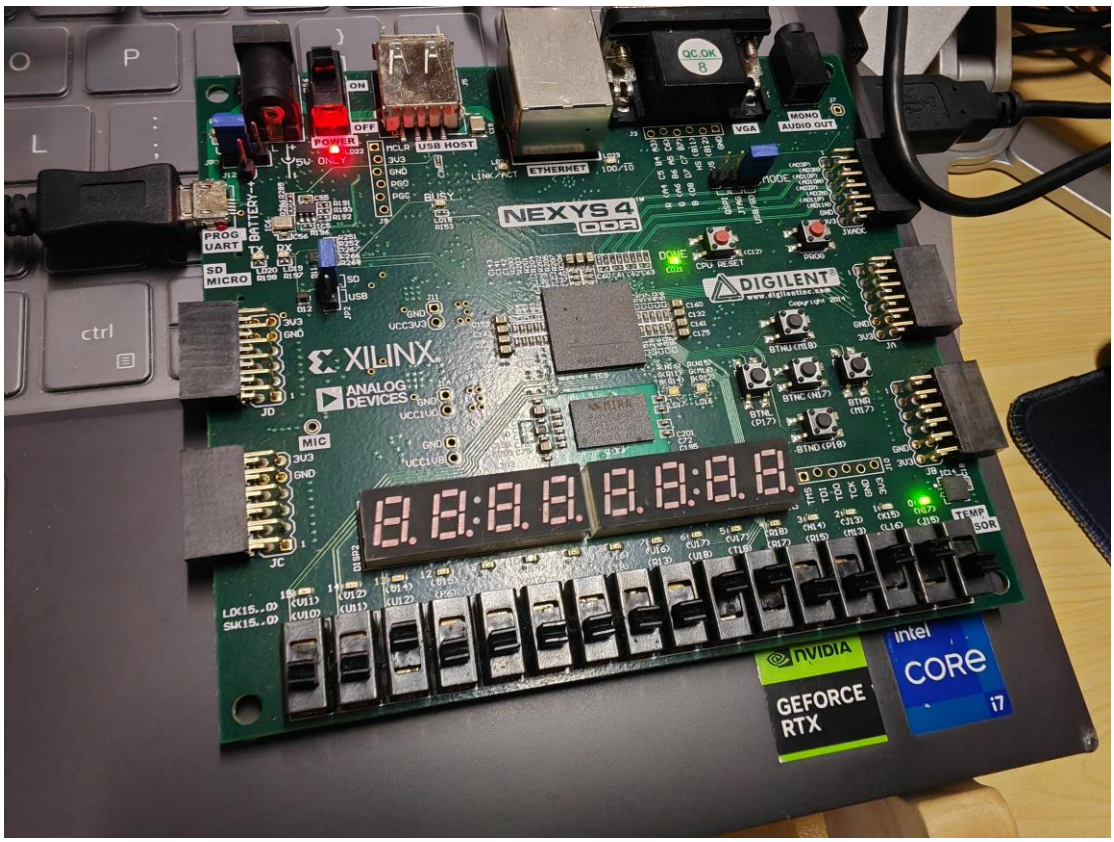
当仅有 U18、T18 打开时，a<b；仅 K15 亮起。



当 M13、L16、U18、T18 打开时，a=b；V10 打开，仅 J13 亮起。

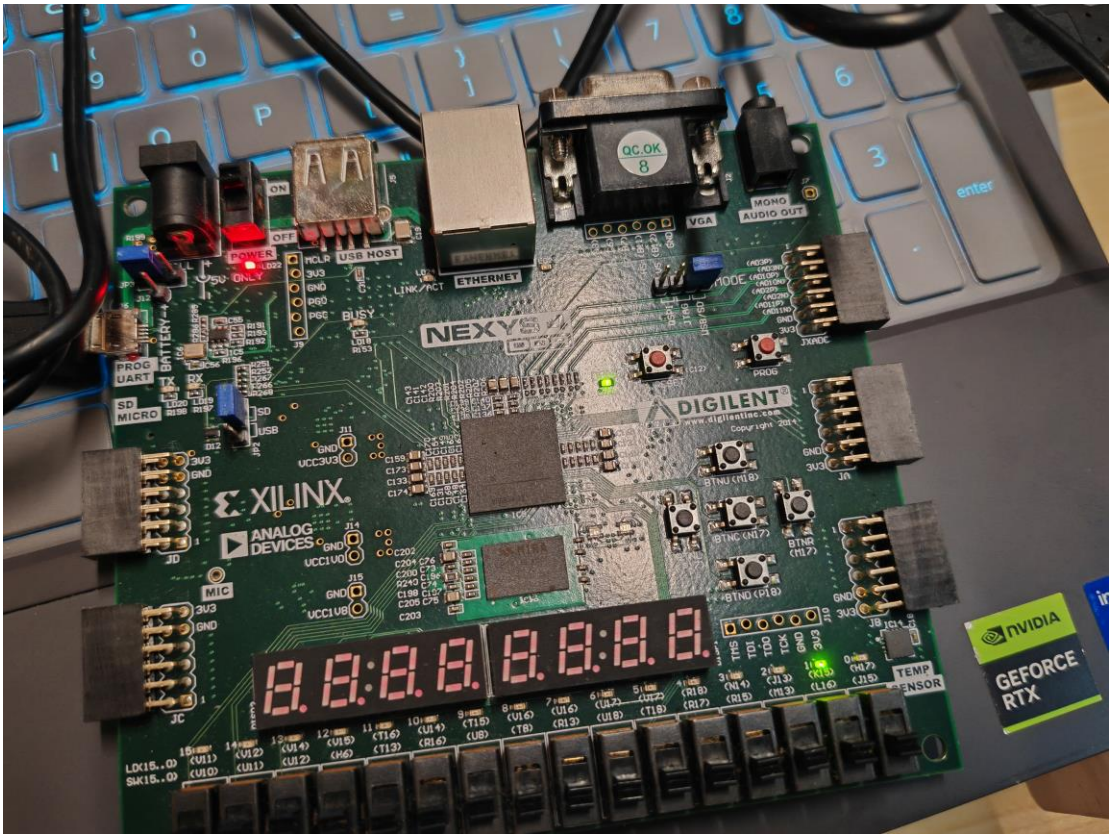


当 M13、L16、U18、T18 打开时，a=b；U11 打开，仅 K15 亮起。

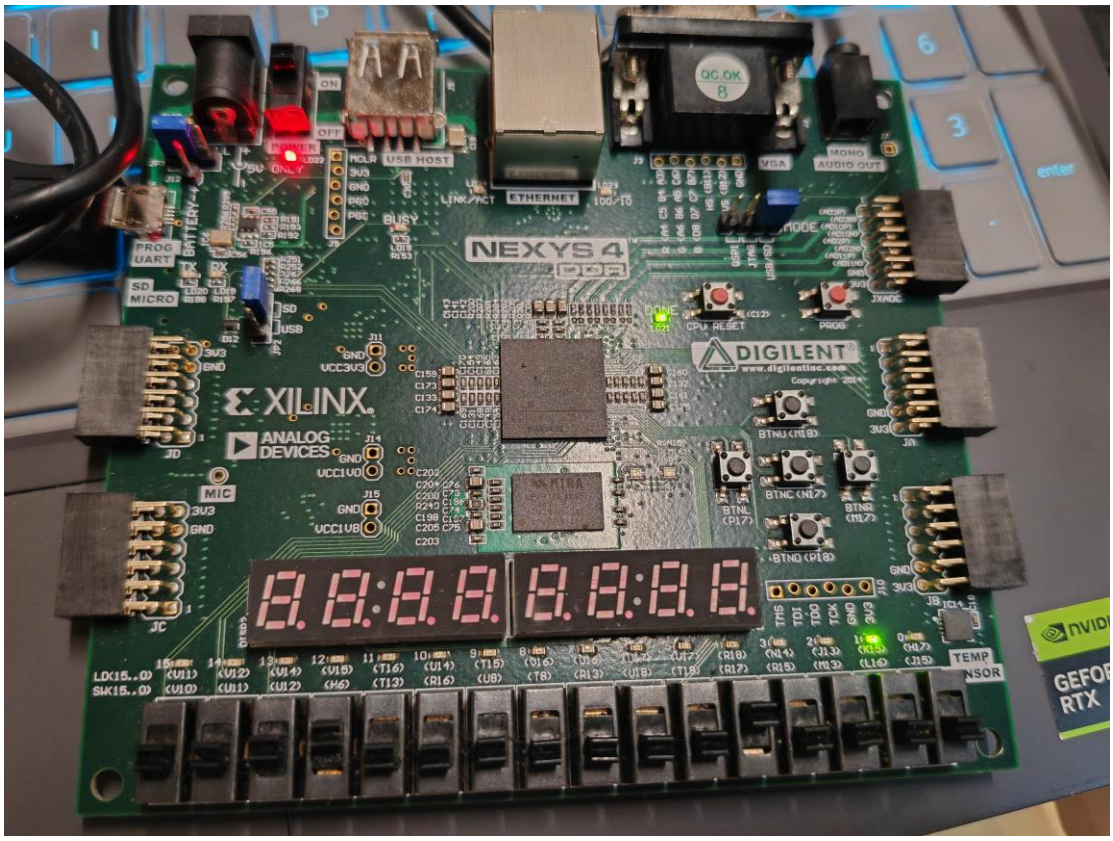


当 M13、L16、U18、T18 打开时，a=b；U12 打开，仅 H17 亮起。

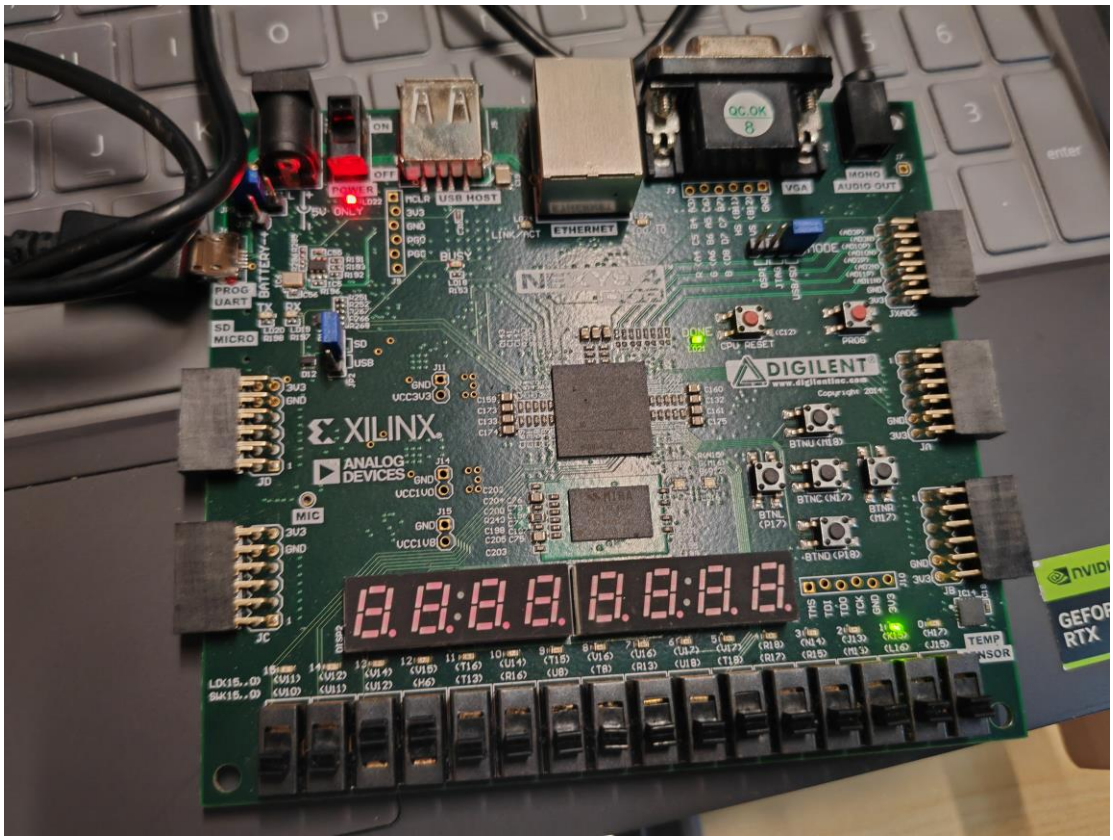
2.8 位比较器



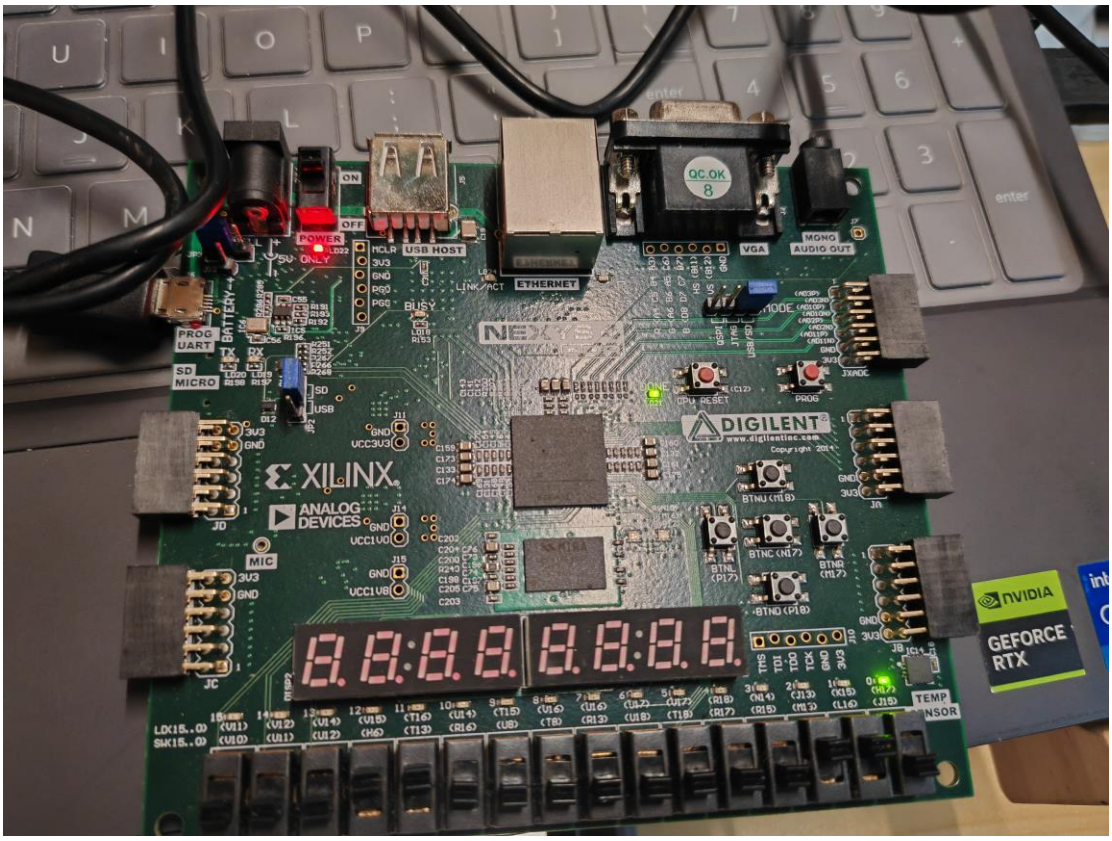
所有开关关闭，a=b；仅 K15 亮起。



打开开关 R17、H6，a=b；仅 K15 亮起。

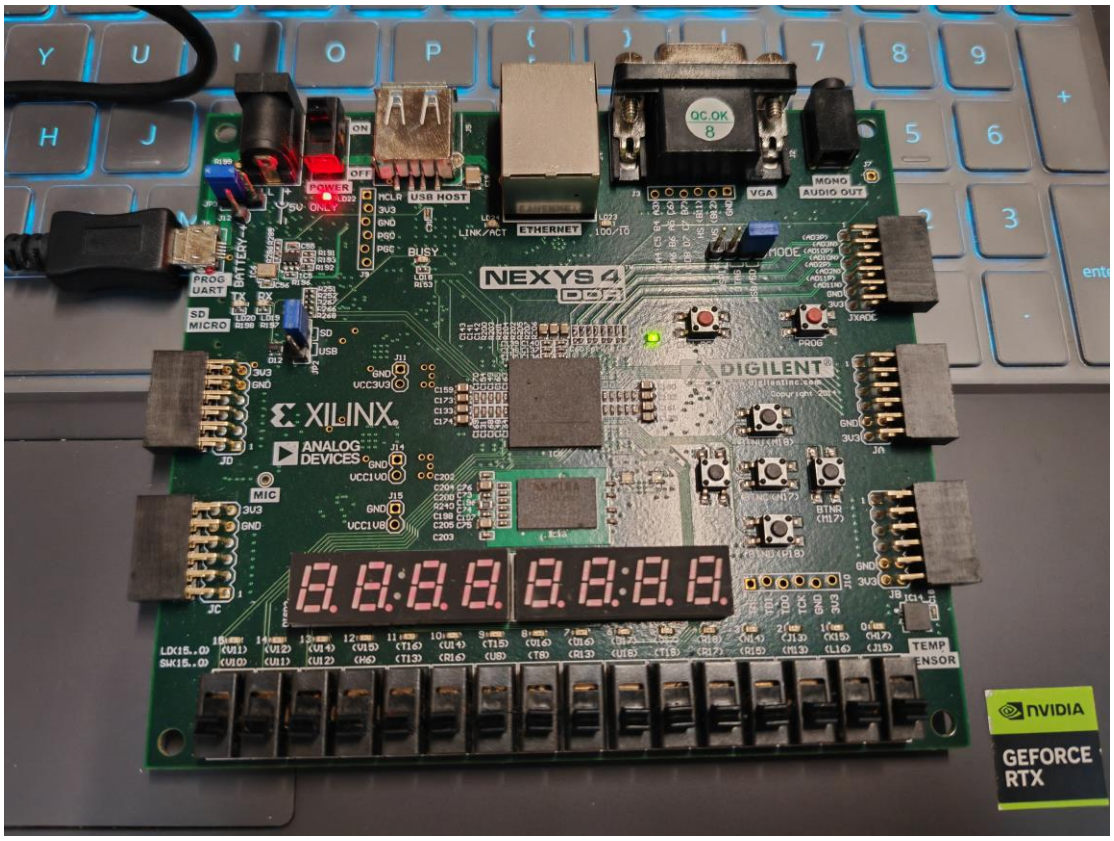


打开开关 U12、H6，a<b；仅 K15 亮起

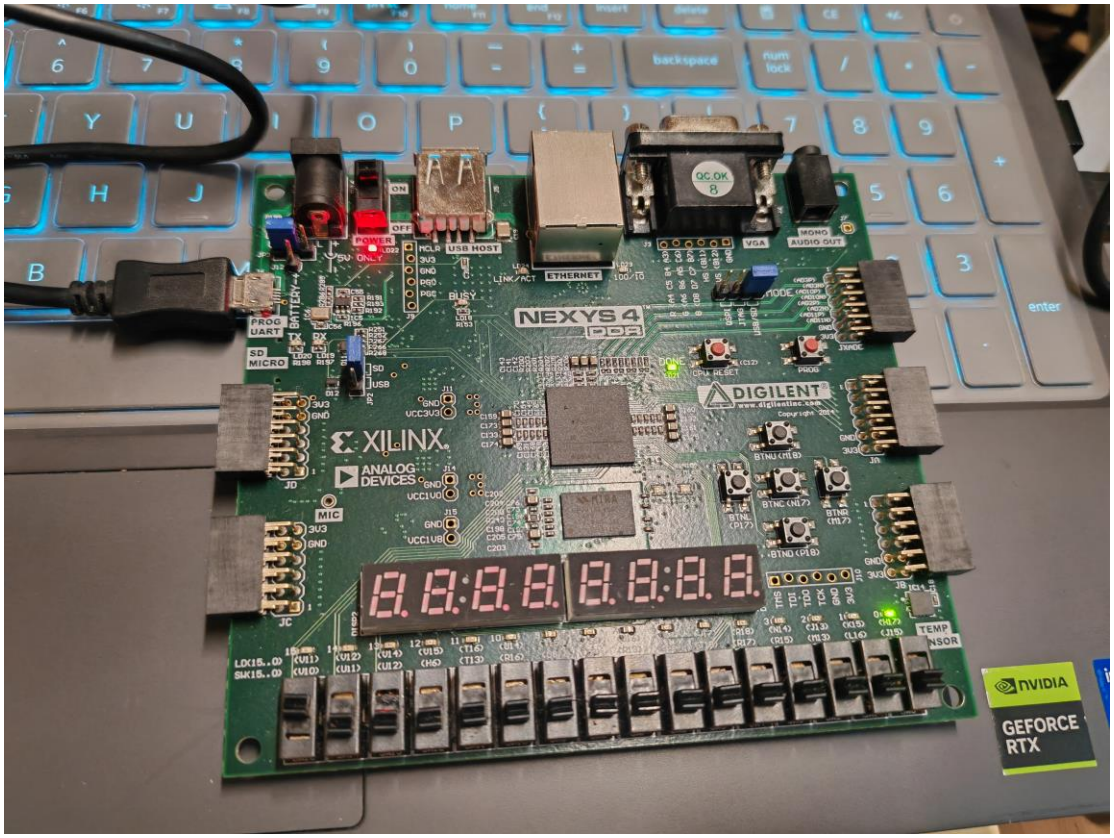


打开开关 H6、T13、M13、L16，a<b；仅 H17 亮起

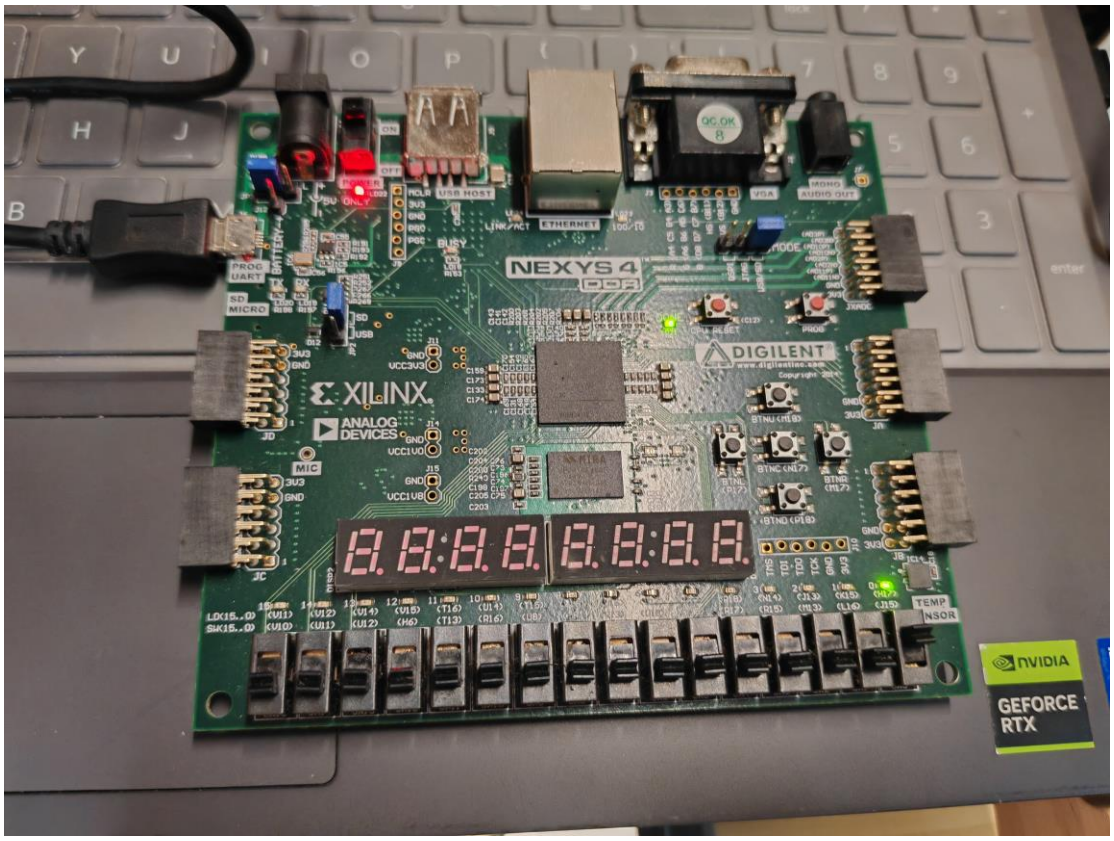
3.无符号加法器



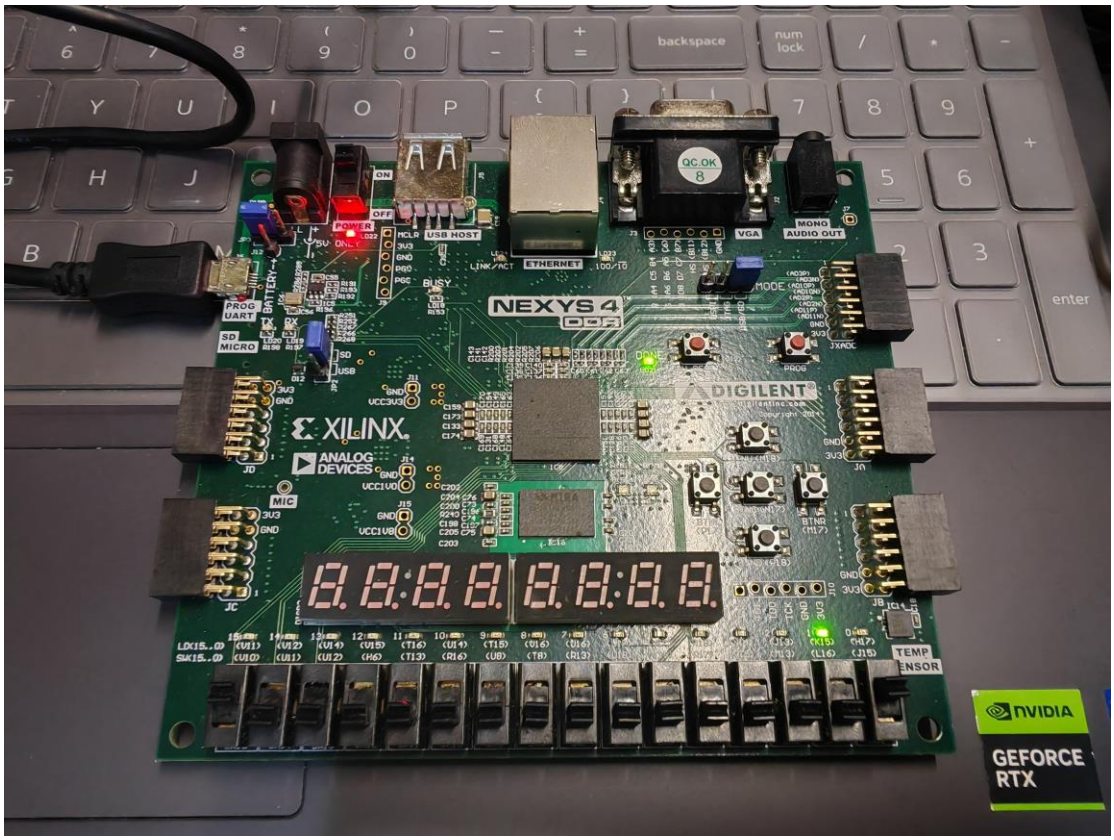
所有开关关闭，无灯亮。



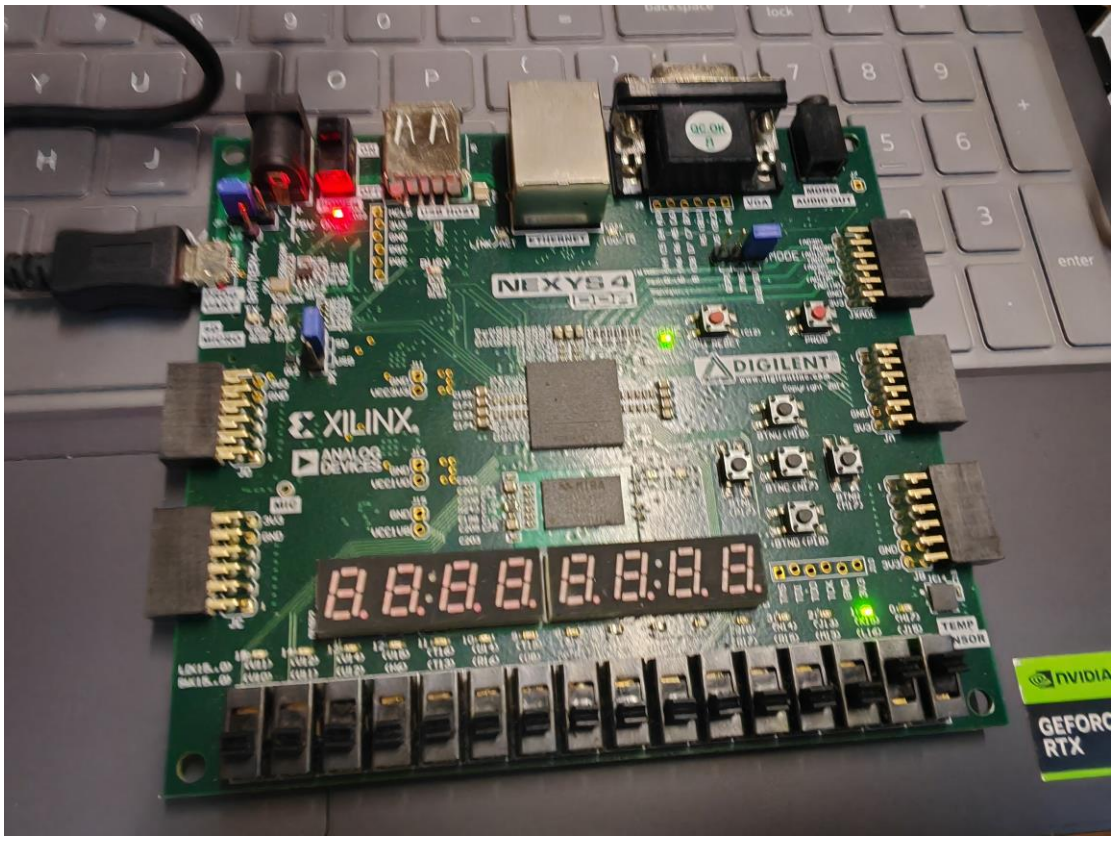
打开 V10，ab 均为 0，c 为 1，H17 亮



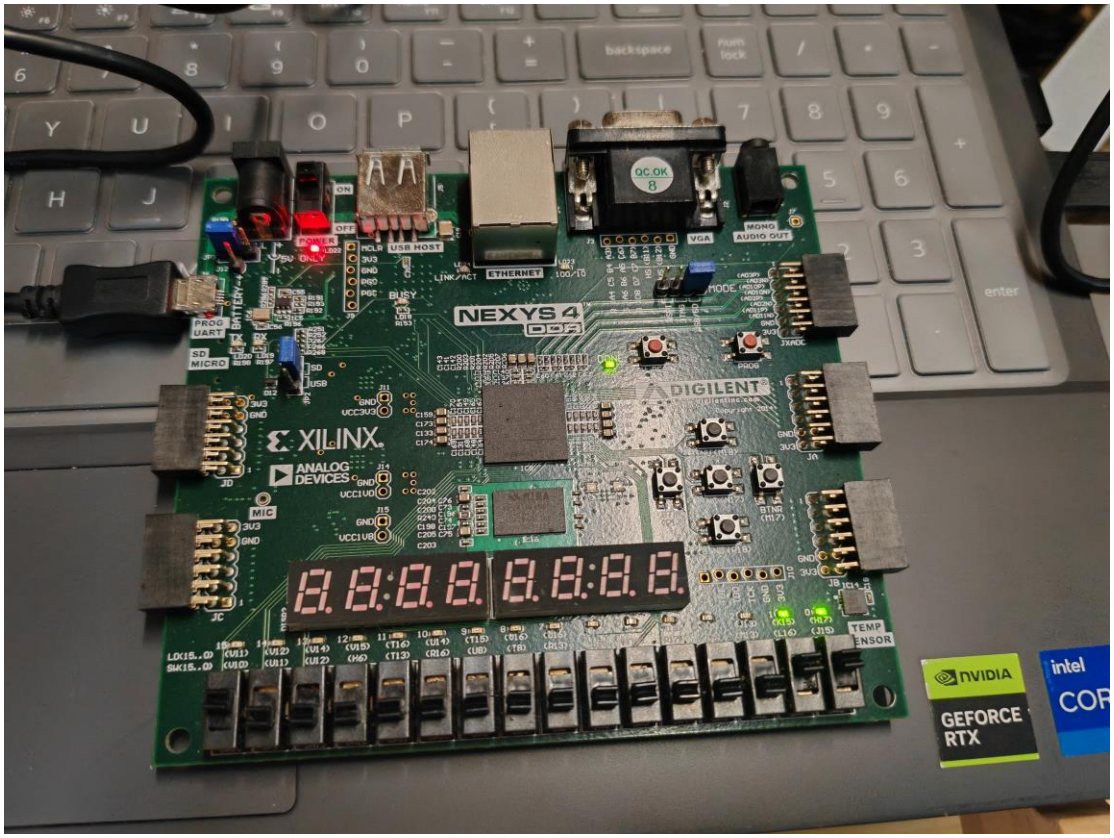
打开 J15，a=1,b=0,c=0,H17 亮



打开 J15 V10，a=1;b=0;c=1;K15 亮

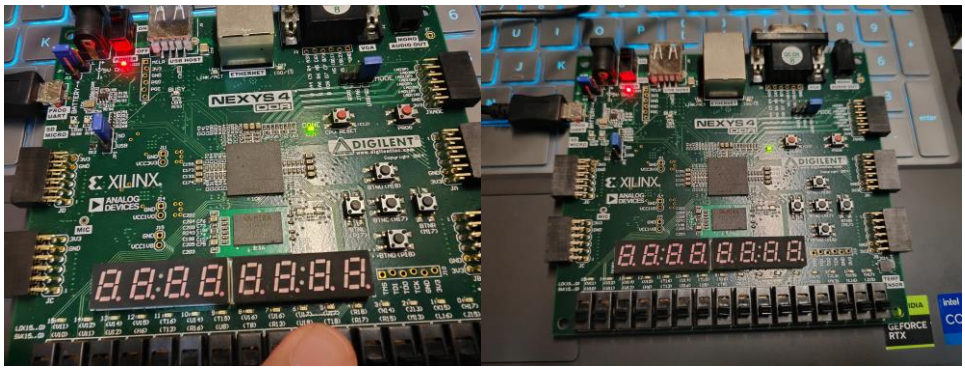


打开 J15 L16,a=1=b, c=0,K15 亮

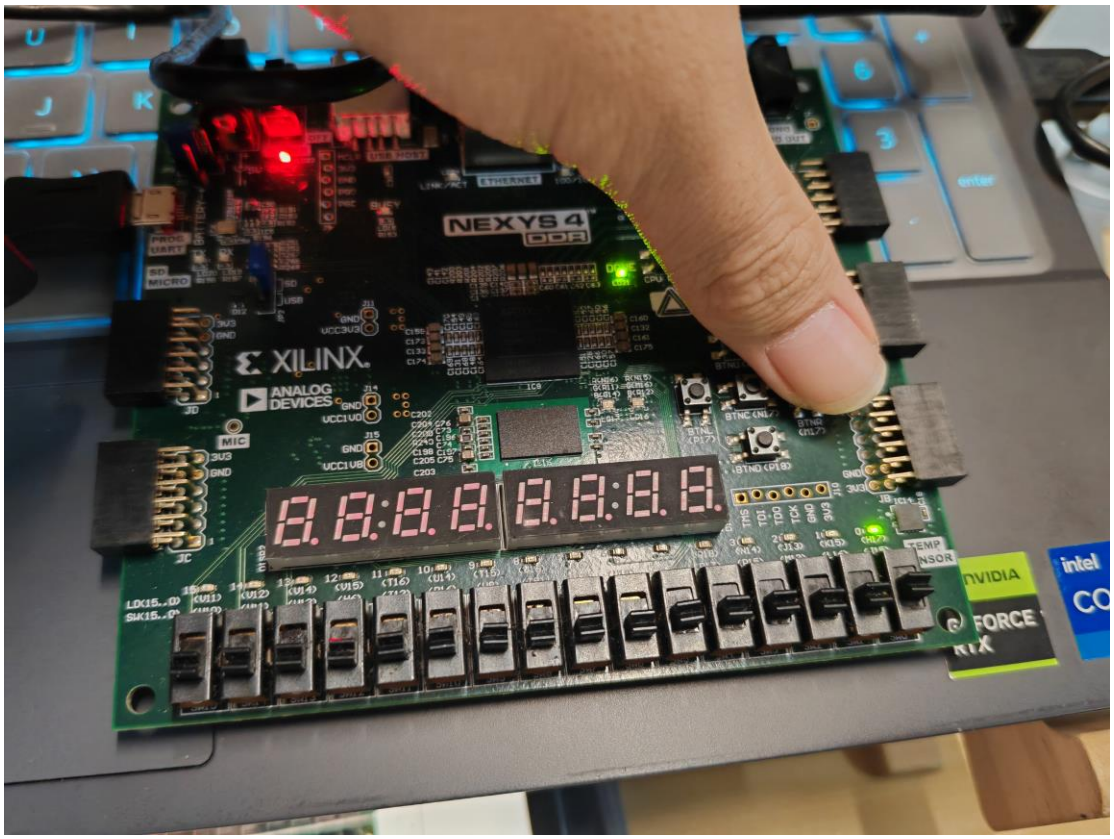


打开 J15\L16\V10,a=b=c=1;K15,H17 亮

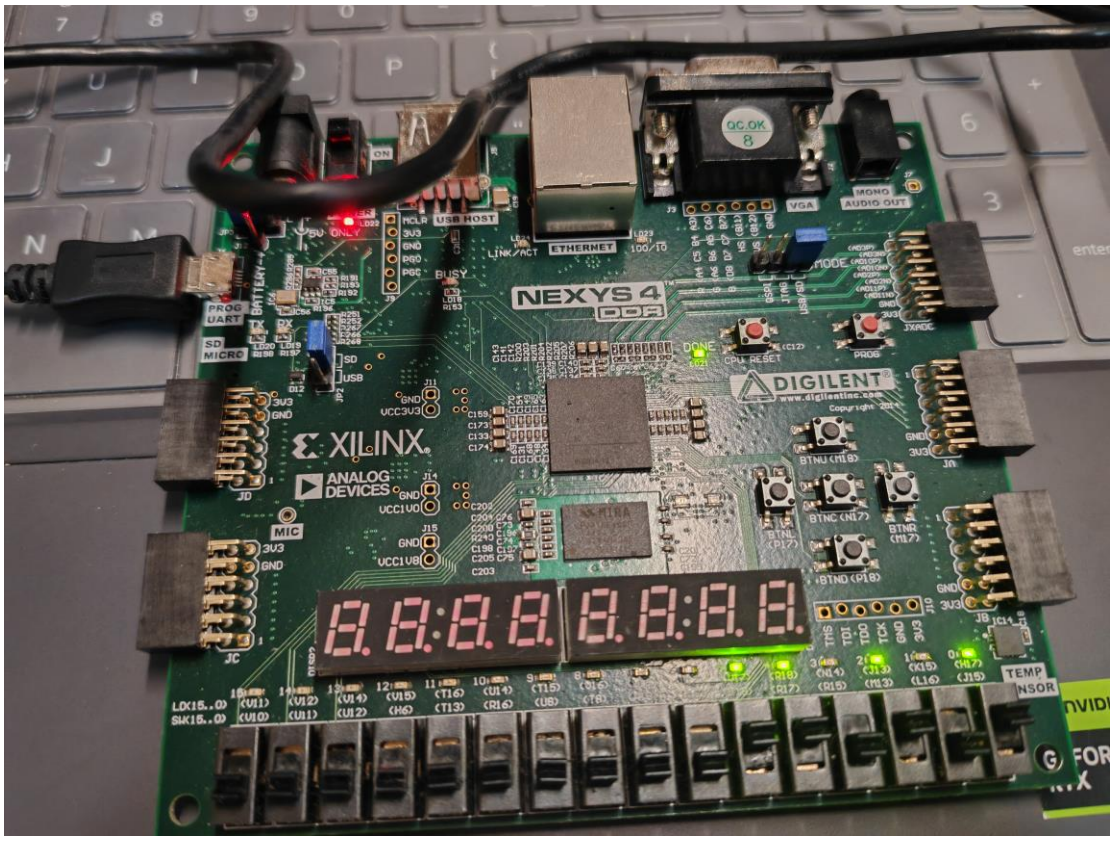
4.有符号加法器



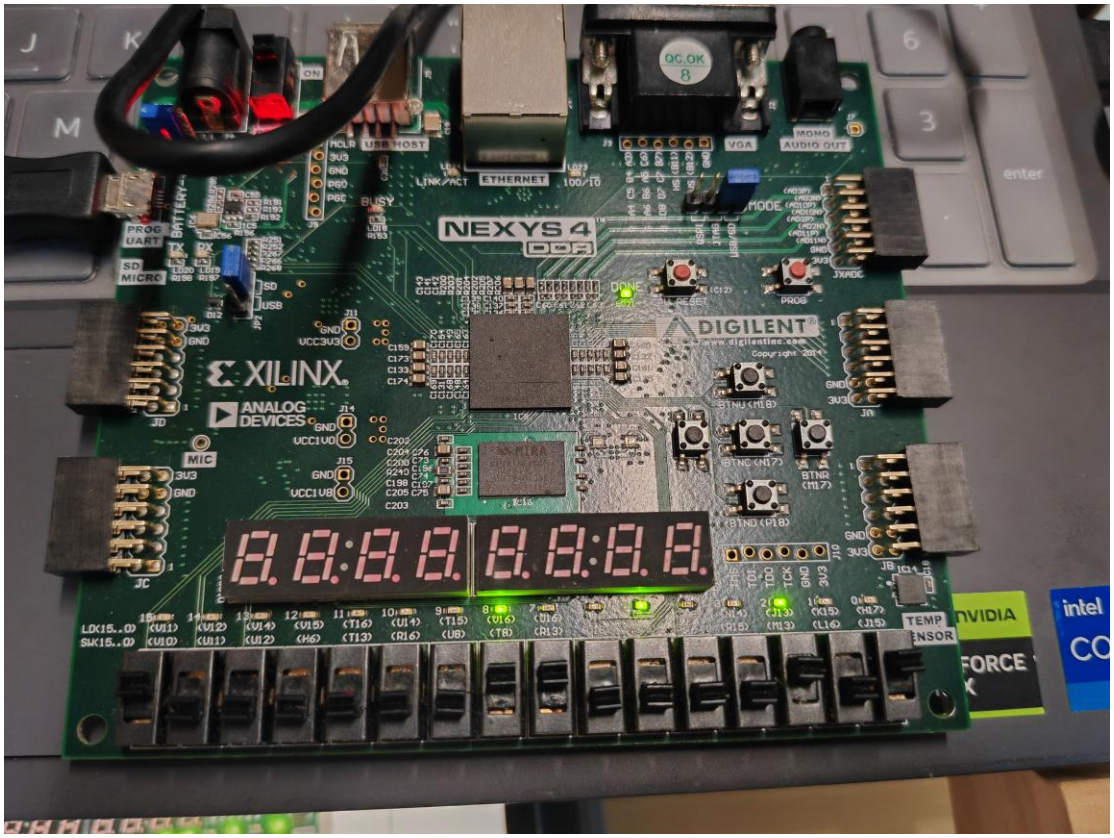
开关均关闭，a、b 均为 0；无灯亮起。
（板子 U18 和 T18 的开关有点接触不良，有的时候需要用力往下按才能保持断的状态）



开关均关闭、按下 M17；a、b 均为 0，产生低位进位；H17 亮起。

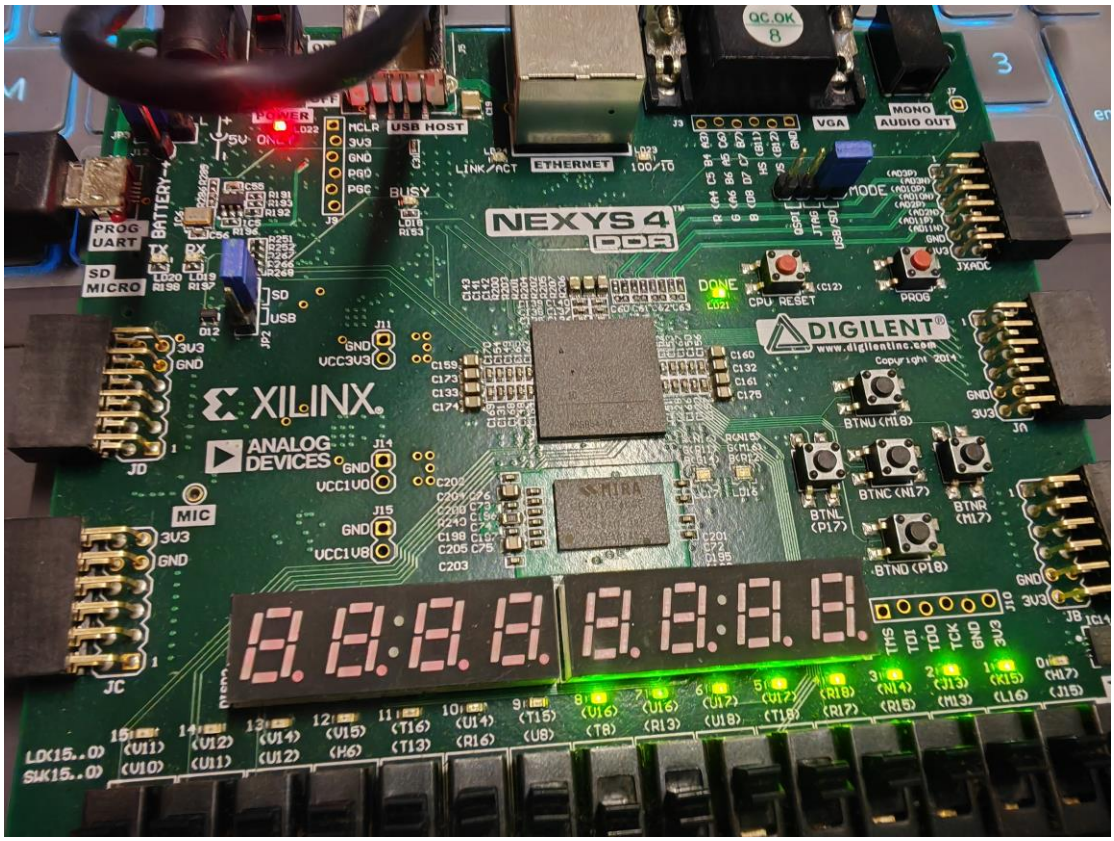


无低位进位，灯亮起示数为 a 和 b 开关输入结果之和，符合预期。



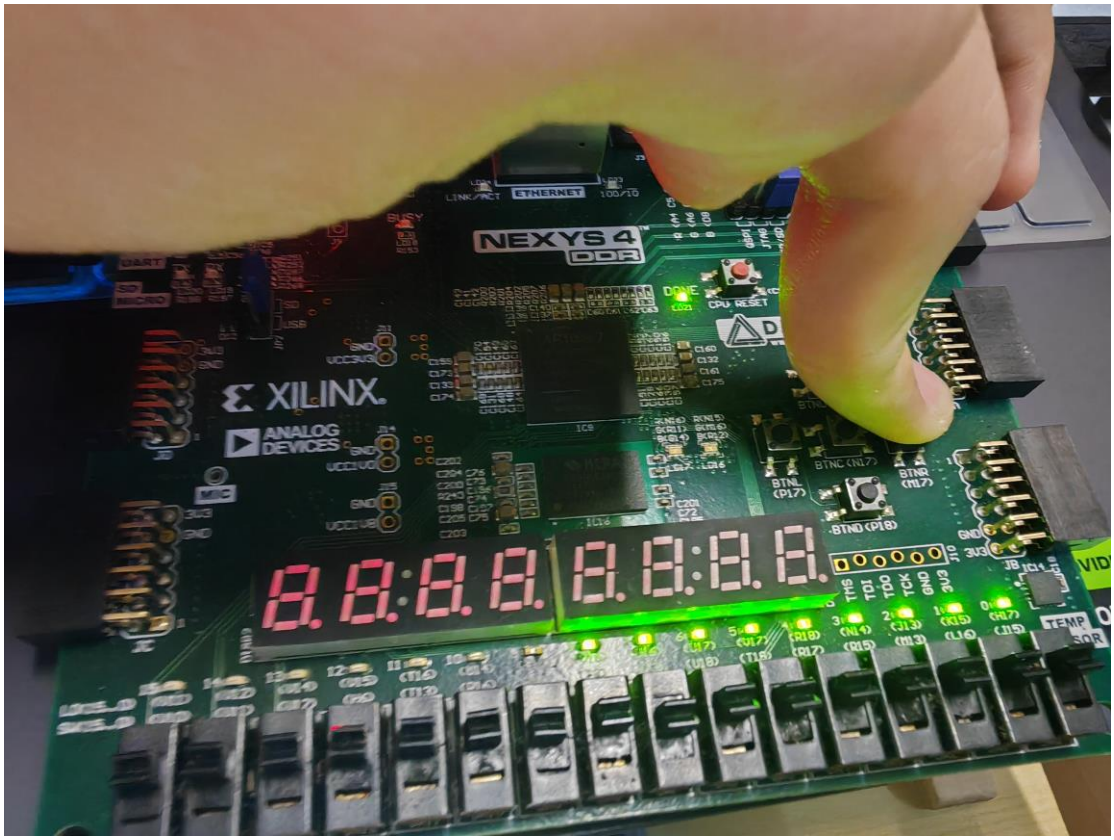
无低位进位，灯亮起示数为 a 和 b 开关输入结果之和，符合预期；

产生高位进位，灯 V16 亮起。



开关均打开，a、b 为 8’b11111111；

相加得 8’b11111110， U16 到 K15 亮起；产生进位，V16 亮起。



开关均打开，a、b 为 8’b11111111；按下 M17，产生低位进位；
相加得 8’b11111111，U16 到 H17 亮起；产生进位，V16 亮起。