Lab1 运算器与寄存器

1. 设计逻辑

(1) ALU：为了实现 ALU 的加、减、与、或、非、异或功能，用自己拟定的 4 位操作码来区分功能：

0001——加　0010——减　0011——与

0100——或　0101——非　0110——异或

用行为级建模风格电路。减法运算为减数取反加一后做加法运算

对于进位的判别：最高位有进位（此时当作无符号数）

对于借位的判别：被减数小于减数，结果符号位为 1

对于溢出的判别：作为有符号数时，加法：最高位进位和次高位不同

减法：取反加一作为加法

对于零标志的判别：结果为 6'b000000 则零标志为 1

4 位标志位 f[3]=1 零标志　f[2]=1 溢出　f[1]=1 借位　f[0]=1 进位

(2) Fibonacci 数列：利用 ALU 的加法功能和存储器的存值功能实现 Fibonacci 数列的输出。将两个 6 位初始值分别置于两个存储器清零后的输入端，存储器为时钟上沿触发，存储器输出接入 ALU 的两个输入端，限定 ALU 的操作数为 0001 执行加法运算，并将输出结果接入第二个存储器。在下一个时钟上沿，第一个存储器的值更新为第二个存储器的值，第二个存储器的值更新为 ALU 的输出结果。

2. 核心代码

```
23  module lab1_1(
24      input [5:0]a,
25      input [5:0]b,
26      input [3:0]s,  //0001加 0010减 0011与 0100或 0101非 0110异或
27      output reg [5:0]y,
28      output reg [3:0]f    //f[0]进位 f[1]借位 f[2]溢出 f[3]零
29      );
30      reg [5:0]c;
31
32      always @(*)
33      begin
34          f=4'b0000;
35          case(s)
36          4'b0000:
37          begin
38              y=6'b0; f=4'b0000;
39          end
40          4'b0001:
41          begin
42              y=a+b;
43              if(y==0)
44              begin
45                  f[3]=1;
46              end
47              else if((a[5]==0 && b[5]==0 && y[5]==1))
48              begin
49                  f[2]=1;
50                  f[0]=1;
51              end
```

ALU_code(1)

```
57            else if((a[5]==1 && b[5]==0 && y[5]==0) || (a[5]==0 && b[5]==1 && y[5]==0))
58            begin
59                f[0]=1;
60            end
61        end
62        4'b0010:
63        begin
64            c=~b;
65            c=c+1;
66            y=a+c;
67            if(a[5]==b[5])
68            begin
69                if(a<b)
70                    f[1]=1;
71            end
72            else if(a[5]==1 && b[5]==0)
73            begin
74                f[1]=1;
75                if(y[5]==0)
76                begin
77                    f[2]=1;
78                end
79            end
80            else if(a[5]==0 && b[5]==1 && y[5]==1)
81            begin
82                f[2]=1;
83            end
```

ALU_code(2)

这里只放 ALU 加法与减法的代码，其他运算过于简单，略过

Fibonacci 代码：

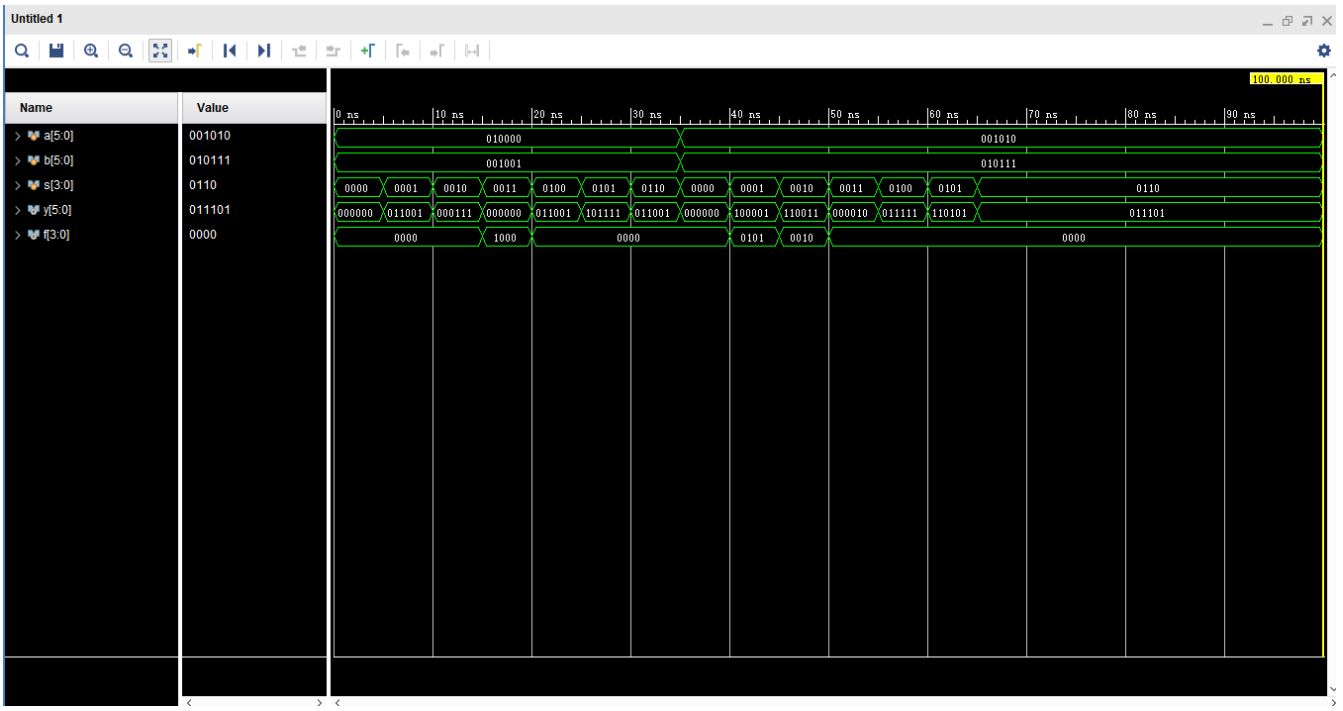
```
module lab1_3_C(
    input [5:0]f0,
    input [5:0]f1,
    input rst,
    input clk,
    output wire [15:0]fn
    );
    wire [3:0]s;
    reg [15:0]f2;
    reg [15:0]f3;
    assign s=4'b0001;

    ALU FIB(.a(f2),.b(f3),.s(s),.y(fn));
    always @(posedge clk)
    begin
        if(rst)
        begin
            f2<=f0;
            f3<=f1;
        end
        else
        begin
            f2<=f3;
            f3<=fn;
        end
    end
endmodule

```

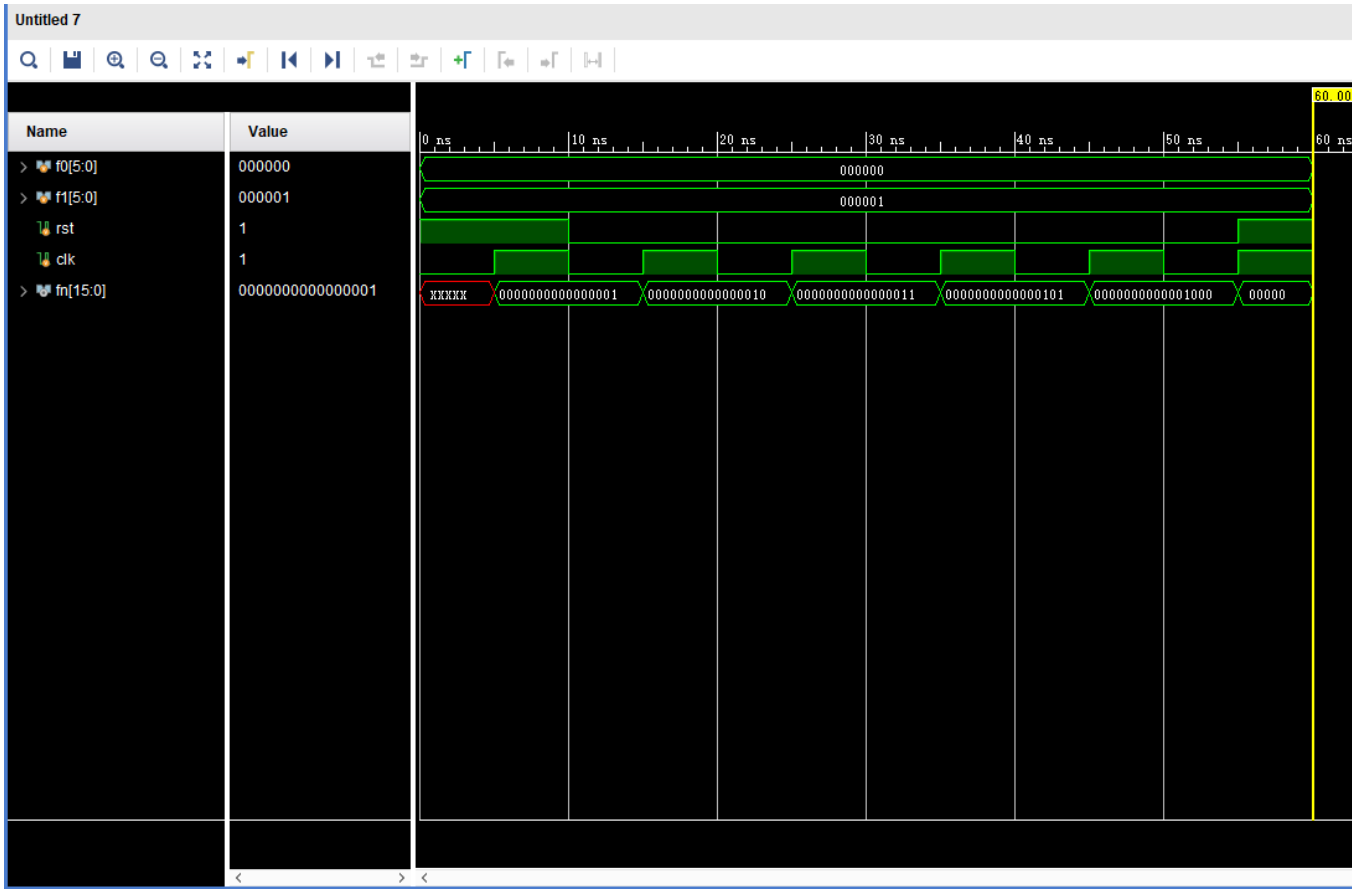
Fibonacci_code

3. 仿真结果与下载结果:



ALU_simulation

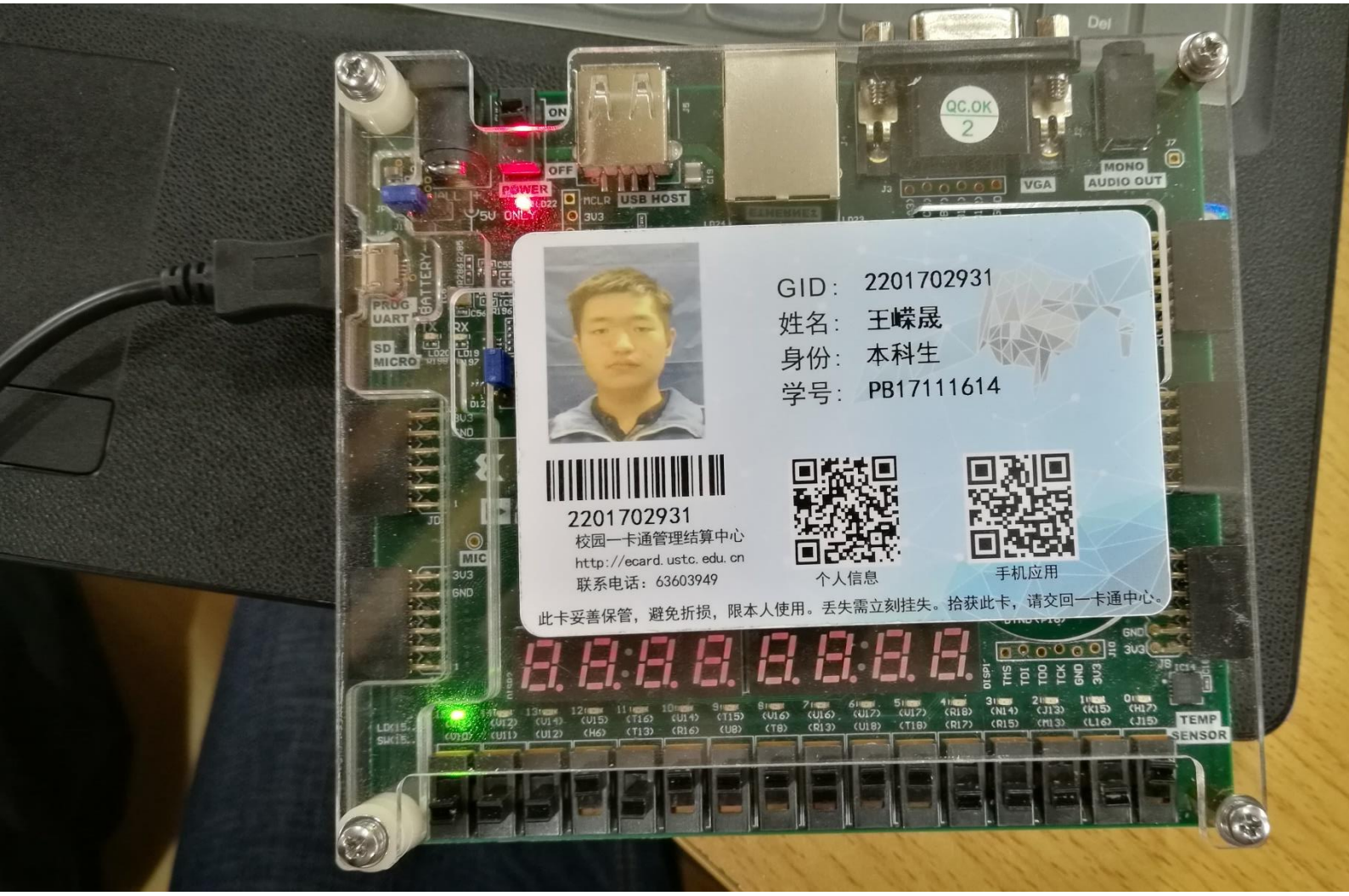
做 ALU 的仿真时，输入了两组变量：16 9 和 8 23，分别进行六种运算得到结果



Fib_simulation

做 Fib 的仿真时，设定初始输入的值为 0 和 1，在 reset 后启动

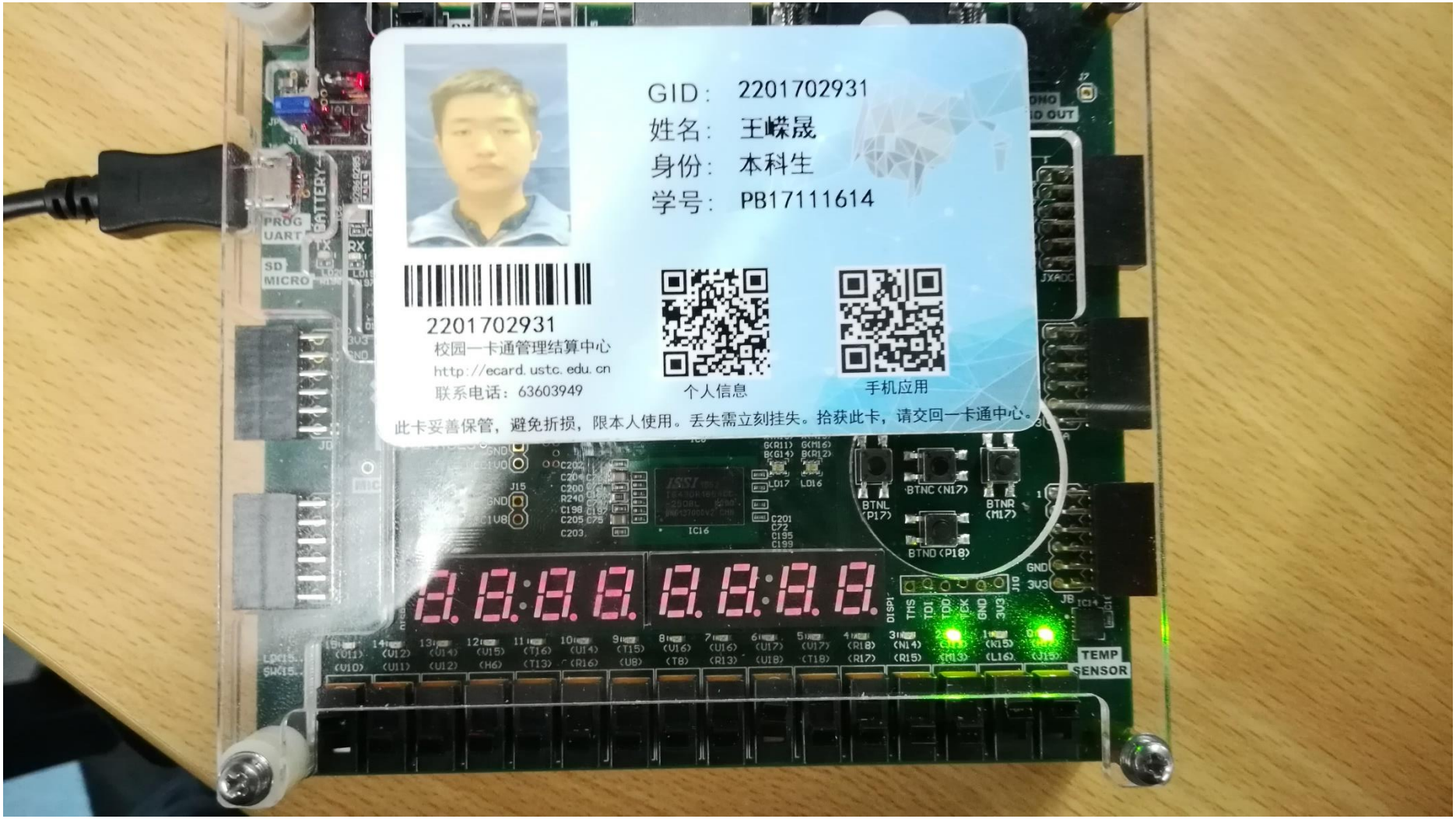
下载照片：



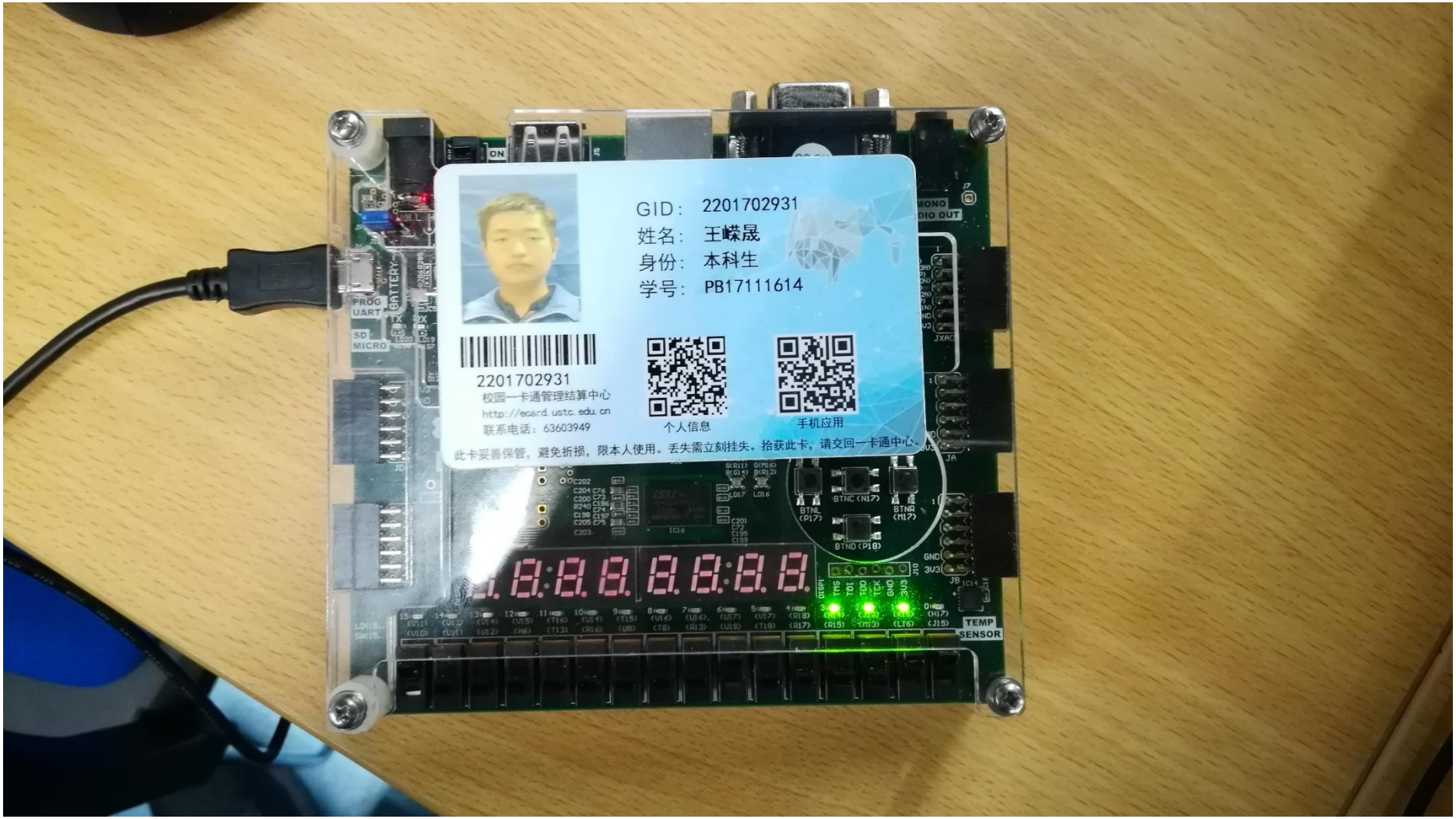
ALU 下载照片（1）


GID: 2201702931
姓名: 王嵘晟
身份: 本科生
学号: PB17111614
2201702931
校园一卡通管理结算中心
http://ecard.ustc.edu.cn
联系电话：63603949
个人信息
手机应用
此卡妥善保管，避免折损，限本人使用。丢失需立刻挂失。拾获此卡，请交回一卡通中心。

ALU 下载照片（2）


GID: 2201702931
姓名: 王嵘晟
身份: 本科生
学号: PB17111614
2201702931
校园一卡通管理结算中心
http://ecard.ustc.edu.cn
联系电话：63603949
个人信息
手机应用
此卡妥善保管，避免折损，限本人使用。丢失需立刻挂失。拾获此卡，请交回一卡通中心。
BTNC (N17)
BTNL (P17)
BTNR (M17)
BTND (P18)
TEMP SENSOR


GID: 2201702931
姓名: 王嵘晟
身份: 本科生
学号: PB17111614
2201702931
校园一卡通管理结算中心
http://ecard.ustc.edu.cn
联系电话：63603949
个人信息
手机应用
此卡妥善保管，避免折损，限本人使用。丢失需立刻挂失。拾获此卡，请交回一卡通中心。
PROG UART
SD MICRO
BATTERY
BTNC (N17)
BTNL (P17)
BTNR (M17)
BTND (P18)
TEMP SENSOR


GID: 2201702931
姓名: 王嵘晟
身份: 本科生
学号: PB17111614
2201702931
校园一卡通管理结算中心
http://ecard.ustc.edu.cn
联系电话：63603949
个人信息
手机应用
此卡妥善保管，避免折损，限本人使用。丢失需立刻挂失。拾获此卡，请交回一卡通中心。
BTNC (N17)
BTNL (P17)
BTNR (M17)
BTND (P18)
TEMP SENSOR

上图为输出 Fibonacci 数列的抓拍，初始值为 2 3

4. 结果分析

   ALU 运算结果准确，Fibonacci 数列输出准确

5. 实验总结

   在近 3 个月没有写 verilog 代码后，对语法有些生疏。还好在编写代码时慢慢找回了感觉。设计 ALU 以及用 ALU 和存储器的组合输出 Fibonacci 数列以实现 ALU 的应用，算不上难度很大的实验，大概主要为了适应。在 Fib 仿真时发现代码逻辑的诸多问题，但已成功修改。

6. 意见/建议：

   希望在今后的实验中实验要求可以给的更明确，防止引起歧义而带来没有必要的时间浪费。