

# 运算器及其应用

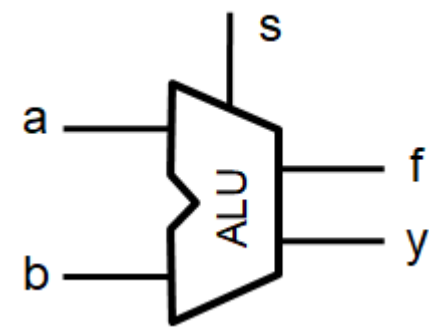
## 一.实验目的

- 掌握算数逻辑单元(ALU)的功能
- 掌握数据通路和控制器的设计方法
- 掌握组合电路和时序电路，以及参数化和结构化的 Verilog描述方法
- 了解查看电路性能和资源使用情况

## 二.逻辑设计

### 1.算数逻辑单元ALU

模块功能

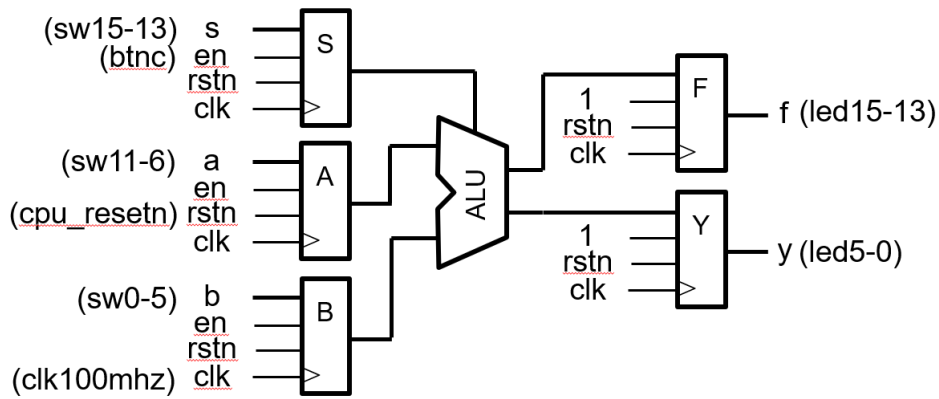


s	y	f
0	$a - b$	*
1	$a + b$	0
2	$a \& b$	0
3	$a   b$	0
4	$a \wedge b$	0
5	$a \gg b$	0
6	$a \ll b$	0
7	$a \ggg b$	0

大小关系	f		
	ltu	lt	eq
$a = b$	0	0	1
$a \neq b$	x	x	0
$a <_s b$	x	1	0
$a \geq_s b$	x	0	x
$a <_u b$	1	x	0
$a \geq_u b$	0	x	x

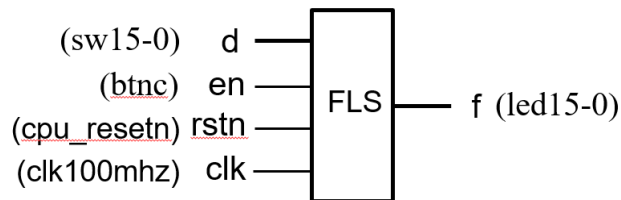
- f[0]:相等 (eq)
- f[1]:有符号数小于 (lt)
- f[2]:无符号数小于 (ltu)
- \*: 表示根据运算结果设置
- x: 表示与比较结果无关

## 数据通路

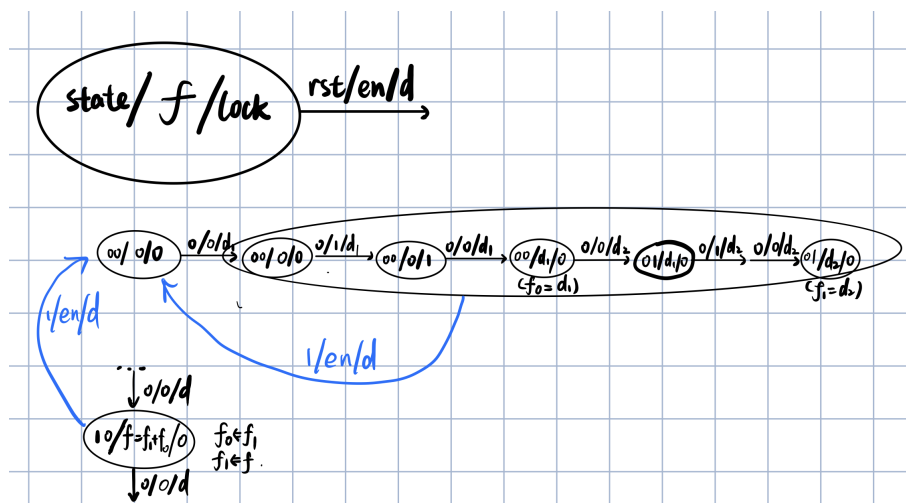


## 2. ALU 应用：计算斐波那契—卢卡斯数列 (Fibonacci Lucas Series)

## 数据通路



## 状态图



### 三.核心代码

## ALU

```
/*parameter SUB = 3'b000,
parameter ADD = 3'b001,
parameter AND = 3'b010,
parameter OR = 3'b011,
parameter XOR = 3'b100,
parameter RS = 3'b101,
parameter LS = 3'b110,
parameter RRS= 3'b111*/
always@(*)begin
```

```

case (s)
  A_SUB:begin
    y=a-b;
    if(!y) f=3'b001;
    else begin//处理不同情况的大小比较数
      f[0]=0;
      f[1]=(a<b)? 1:0;
      if(a[WIDTH-1]==1&&b[WIDTH-1]==0) f[2]=1;
      else if(a[WIDTH-1]==0&&b[WIDTH-1]==1) f[2]=0;
      else if(a[WIDTH-1]==0 && b[WIDTH-1]==0) f[2]=(a<b)? 1:0;
      else f[2]=(a>b)? 1:0;
    end
  end
  A_ADD:begin
    y=a+b;f=0;
  end
  A_AND:begin
    y=a&b;f=0;
  end
  A_OR:begin
    y=a|b; f=0;
  end
  A_XOR:begin
    y=a^b;f=0;
  end
  A_RS:begin
    y=a>>b;f=0;
  end
  A_LS:begin
    y=a<<b;f=0;
  end
  A_RRS:begin
    y=a>>>b; f=0;
  end
  default:begin
    y=0;f=0;
  end
endcase
end

```

## FLS

```

reg lock=0;//协助en来控制d值的输入
reg [31:0] cnt;
reg[WIDTH-1:0] f0,f1,nextf;//永远把nextf赋给f
wire[WIDTH-1:0] f2;
reg[1:0] curState,nextState; //curState nextState起到了状态机的作用

//用ALU模块去生成预赋值的信号
alu alu(.a(f0),.b(f1),.s(3'b001),.y(f2));

//初始化
initial begin
  nextState <= 2'b00;
  curState<=2'b00;

```

```

    f1<=16'b0;
    f0<=16'b0;
    cnt <= 0;
    f<=0;
end
//用于控制时间频率,便于在板上观察信号
always @(posedge clk) begin
    if (cnt == 'd80000000) begin
        cnt <= 0;
    end
    else begin
        cnt <= cnt + 1;
    end
end
end

//每当输入的值 rst,en有所改变时,就应该产生相对应的变化
//只更新nextState nextf lock
always@(*)begin
    if(!rst) begin //当rst=0时,重置。
        nextState = 2'b00;
        nextf=16'b0;
        lock=1'b0;
    end else if(!en)begin//当en=0时,考虑是否可以赋值
        case(curState)
            2'b00:begin
                if(lock)begin
                    nextf=d;
                    nextState=2'b01;
                    lock=0;
                end
            end
            2'b01:begin
                if(lock)begin
                    nextf=d;
                    nextState=2'b10;
                    lock=0;
                end
            end
            2'b10: begin
                nextf=f2;
                nextState=2'b11;
            end
        endcase
    end else begin
        lock=1; //当按下
        nextf=f; //说明当rst=0, en=0时,保持f的值不变;
    end
end
end

//时钟上升沿赋值
always @(posedge clk)begin
    if (cnt == 16'b0) begin
        if(!rst) begin
            curState<=2'b00;
            f0<=16'b0;
            f1<=16'b0;

```

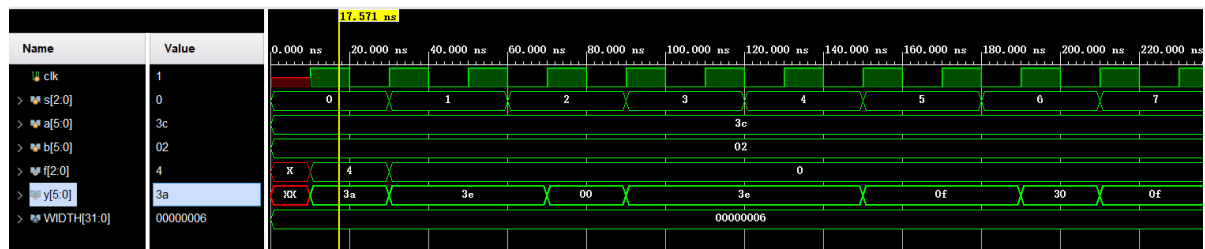
```

        f<=16'b0;
    end else if(!en)begin
        case(curState)
            2'b00:begin
                f0<=nextf;
                f<=nextf;
                curState<=nextState;//curState改变为下一状态
            end
            2'b01:begin
                f1<=nextf;
                f<=nextf;
                curState<=nextState;
            end
            default:begin
                f0<=f1;//斐波那契数列数值传递
                f1<=nextf;
                f<=nextf; //保证每次f的最终值由nextf决定。
            end
        endcase
    end else f<=nextf;
end
end

```

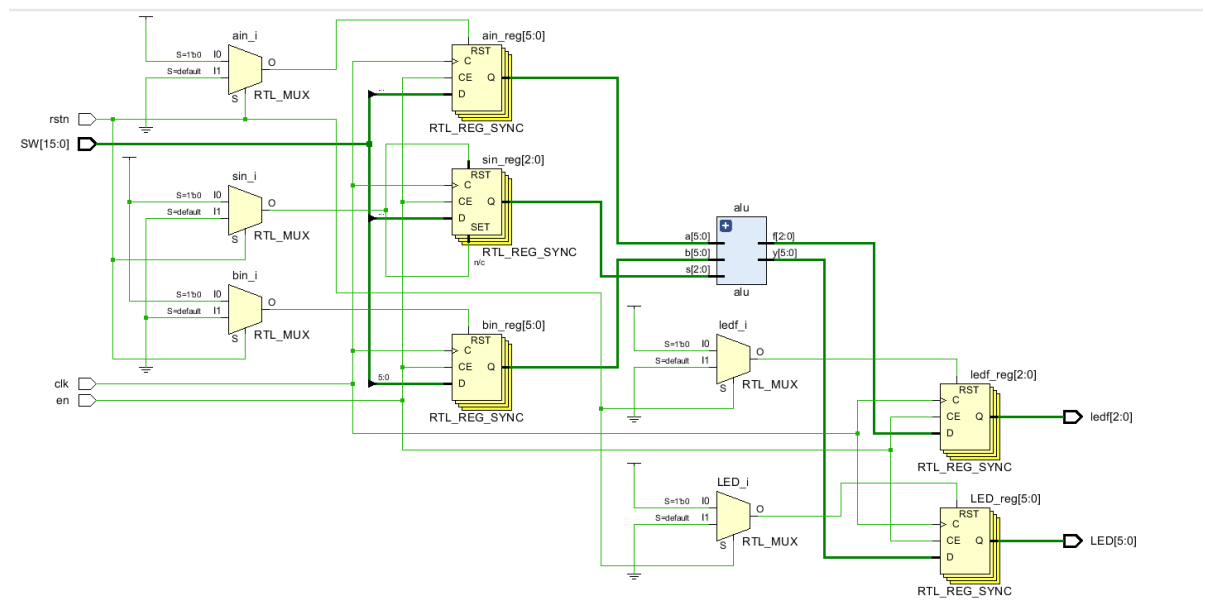
## 四.实验结果

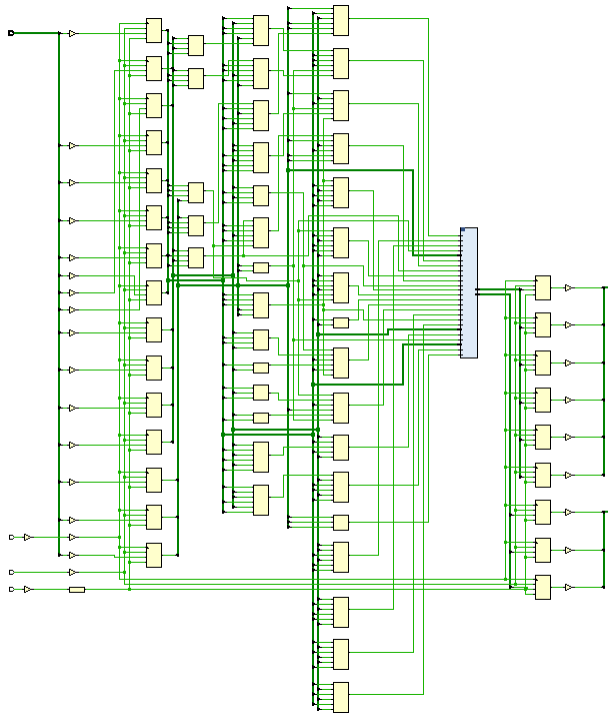
1.下图演示了ALU模块仿真的结果，输入a=6'b111100,b=6'b000010，不断切换s的值



2.下图演示ALU模块电子路资源

Vivado生成电路：RTL电路和综合电路





ALU电路资源使用

Summary

Resource	Utilization	Available	Utilization %
LUT	58	63400	0.09
FF	15	126800	0.01
IO	27	210	12.86

LUT 1%

FF 1%

IO 13%

Utilization (%)

Name	Slice LUTs (63400)	Slice Registers (126800)	Bonded IOB (210)	BUFGCTRL (32)
main	58	24	27	1
alu (alu)	28	0	0	0

3.下图演示ALU电路性能

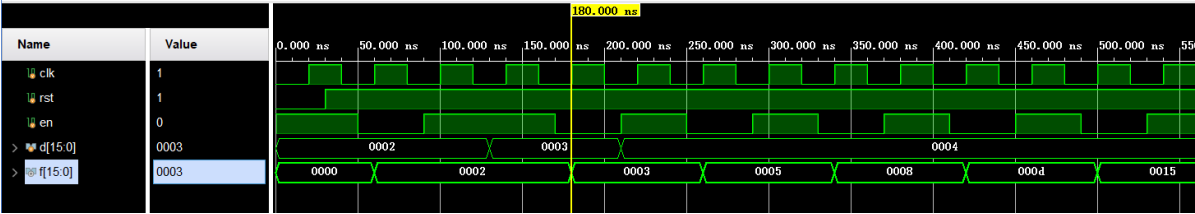
Intra-Clock Paths - sys\_clk\_pin

Clock: sys\_clk\_pin

Statistics

Type	Worst Slack	Total Violation	Failing Endpoints	Total Endpoints
Setup	6.146 ns	0.000 ns	0	9
Hold	0.220 ns	0.000 ns	0	9
Pulse Width	4.500 ns	0.000 ns	0	25

4.下图演示了FLS模块仿真的结果，在两次切换en后，输入得到



注：烧写板下载结果不易在报告中展示，待实验检查时现场演示

## 五.结果分析

仿真与下载结果基本符合预期，综合性能良好，结构清晰，满足实验要求

## 六.实验总结

本学期计算机组成原理第一次实验，复习了Verilog HDL语法，熟悉了Vivado 开发环境和仿真、下载、性能测试，通过FSL实验，熟悉了Moore型FSM，将数据通路结构化，提高了自己逻辑设计水平。