实验三 基于 FPGA 的简易计算器

2019010175 孔瑞阳 计科 91

一、设计思路

计算模块：

通过 k1k0 的取值和 a/b 的输入确定运算的结果。

选择模块：

根据 k3, k2 的值确定在哪个数码管上输出。
k3k2 为 00 时第四个数码管显示结果的后面一位。
k3k2 为 01 时第三个数码管显示结果的前面一位。
k3k2 为 10 时第二个数码管显示 b。
k3k2 为 00 时第一个数码管显示 a。

输出模块：

通过 data 的值确定数码管的各个部分的亮暗。

二、实现代码

```
module experiment(
    input [3:0] a, b, k,      // 输入代表 a,b 和 k
    output reg [7:0] light,  // 控制数码管输出
    output reg [3:0] sel,    // 选择在哪个数码管输出
    output reg sign          // 符号位 LED
);

reg [7:0] ans;               // 运算结果
reg [3:0] data;              // 输出值
```

```
always @ (a or b or k)       // 计算模块
begin
    sign = 0;
    ans = 0;
    case (k[1:0])            // 控制运算类型
        2'b00: ans = 0;
        2'b01: ans = a + b;
```

```
        2'b10:
        begin
            {sign, ans[3:0]} = a - b;
            if (sign)
                ans[3:0] = b - a;
        end
        2'b11: ans = a * b;
    endcase
end
```

```
always @ (*)                        // 选择模块
begin
    sel = 4'b1111;
    case (k[3:2])
        2'b00:
            begin
                data = ans[3:0];    // 数码管 4 代表结果的后面一位
                sel = 4'b1110;
            end
        2'b01:
            begin
                data = ans[7:4];    // 数码管 3 代表结果的前面一位
                sel = 4'b1101;
            end
        2'b10:
            begin
                data = b;           // 数码管 2 代表 b
                sel = 4'b1011;
            end
        2'b11:
            begin
                data = a;           // 数码管 1 代表 a
                sel = 4'b0111;
            end
    endcase
end
```

```
always @ (*)                        // 输出模块
begin
    light[0] = 0;
    case (data[3:0])
```

```
        4'h0: light[7:1] = 7'b1111110;
        4'h1: light[7:1] = 7'b0110000;
        4'h2: light[7:1] = 7'b1101101;
        4'h3: light[7:1] = 7'b1111001;
        4'h4: light[7:1] = 7'b0110011;
        4'h5: light[7:1] = 7'b1011011;
        4'h6: light[7:1] = 7'b1011111;
        4'h7: light[7:1] = 7'b1110000;
        4'h8: light[7:1] = 7'b1111111;
        4'h9: light[7:1] = 7'b1111011;
        4'hA: light[7:1] = 7'b1110111;
        4'hB: light[7:1] = 7'b0011111;
        4'hC: light[7:1] = 7'b1001110;
        4'hD: light[7:1] = 7'b0111101;
        4'hE: light[7:1] = 7'b1001111;
        4'hF: light[7:1] = 7'b1000111;
    endcase
end

endmodule
```

三、拓展内容

拓展内容：

1. 计算结果用十进制而不是十六进制输出。
2. 当结果为负数时，用第一个数码管显示负号而不用 LED 表示。
3. 用扫描的方式同时在四个数码管上显示结果。

实现代码：

```
module experiment(
    input [3:0] a, b, k,      // 输入代表 a,b 和 k
    input clk,
    output reg [7:0] light,  // 控制数码管输出
    output reg [3:0] sel     // 选择在哪个数码管输出
);

reg sign;
reg [1:0] cur;
reg [7:0] ans;               // 运算结果
reg [3:0] data;              // 输出值
integer cnt;                // clock 计数
```

```
always @ (a or b or k)          // 计算模块
begin
    sign = 0;
    ans = 0;
    case (k[1:0])               // 控制运算类型
        2'b00: ans = 0;
        2'b01: ans = a + b;
        2'b10:
            begin
                {sign, ans[3:0]} = a - b;
                if (sign)
                    ans[3:0] = b - a;
            end
        2'b11: ans = a * b;
    endcase
end
```

```
always @ (posedge clk)          // 计数模块
begin
    cnt = cnt + 1;
    if (cnt == 5000)
    begin
        cnt = 0;
        cur = cur + 1;
    end
end
```

```
always @ (*)                    //选择模块，十进制输出
begin
    sel = 4'b1111;
    case (cur)
        2'b00:
            begin
                data = ans % 10;
                sel = 4'b1110;
            end
        2'b01:
            begin
                data = ans / 10 % 10;
                sel = 4'b1101;
            end
```

```
        2'b10:
            begin
                data = ans / 100;
                sel = 4'b1011;
            end
        2'b11:
            begin
                data = (sign) ? 4'hA : 4'h0;
                sel = 4'b0111;
            end
    endcase
end
```

```
always @ (*)                                    // 数码管输出
begin
    light[0] = 0;
    case (data[3:0])
        4'h0: light[7:1] = 7'b1111110;
        4'h1: light[7:1] = 7'b0110000;
        4'h2: light[7:1] = 7'b1101101;
        4'h3: light[7:1] = 7'b1111001;
        4'h4: light[7:1] = 7'b0110011;
        4'h5: light[7:1] = 7'b1011011;
        4'h6: light[7:1] = 7'b1011111;
        4'h7: light[7:1] = 7'b1110000;
        4'h8: light[7:1] = 7'b1111111;
        4'h9: light[7:1] = 7'b1111011;
        4'hA: light[7:1] = 7'b0000001;     // 不会出现的 A 表示负号
    endcase
end

endmodule
```