计算机组成原理实验报告

- **实验题目：数据通路与状态机**
- **实验日期：2019年3月29日**
- **姓名：张劲暾**
- **学号：PB16111485**
- **成绩：**

实验目的：

1. 排序：s0 ~ s3是x0 ~ x3的排序结果
2. 除法运算：x / y = q ... r

实验设计简述与核心代码：

四数排序单元设计（SORT.v）

```
module SORT(
    //======================================================================
    input [3:0] x0,
    input [3:0] x1,
    input [3:0] x2,
    input [3:0] x3,
    input rst,
    input clk,
    //======================================================================
    output reg [3:0] s0,
    output reg [3:0] s1,
    output reg [3:0] s2,
    output reg [3:0] s3,
    output reg done
    //======================================================================
    );
    reg [1:0] cmp_count;
    // cmp_count
    always@(posedge clk or posedge rst) // 状态机，3种状态
        begin
            if(rst)
                begin
                    cmp_count <= 2'b00;
                    done <= 0;
```

```
                end
            else if(cmp_count == 2'b10)
                begin
                    cmp_count <= 2'b00;
                    done <= 1;
                end
            else
                begin
                    cmp_count <= cmp_count + 2'b01;
                end
        end
    //====================================================================
    // compare
    always@(posedge clk or posedge rst)
        begin
            if(rst)
                begin
                    {s0[3:0],s1[3:0],s2[3:0],s3[3:0]} <= {x0[3:0],x1[3:0],x2[3:0],x3[3:0]};
                end
            else
                begin
                    case(cmp_count)
                        2'b00:              // 状态一，1-2比较，3-4比较
                            begin
                 {s0[3:0], s1[3:0]} <= (s0[3:0] > s1[3:0]) ? {s0[3:0], s1[3:0]} : {s1[3:0], s0[3:0]};
                 {s2[3:0], s3[3:0]} <= (s2[3:0] > s3[3:0]) ? {s2[3:0], s3[3:0]} : {s3[3:0], s2[3:0]};
                            end
                        2'b01:              // 状态二，1-3比较，2-4比较
                            begin
                   {s0[3:0], s2[3:0]} <= (s0[3:0] > s2[3:0]) ? {s0[3:0], s2[3:0]} : {s2[3:0], s0[3:0]};
                   {s1[3:0], s3[3:0]} <= (s1[3:0] > s3[3:0]) ? {s1[3:0], s3[3:0]} : {s3[3:0], s1[3:0]};
                            end
                        2'b10:              // 状态三，2-3比较
                            begin
                   {s1[3:0], s2[3:0]} <= (s1[3:0] > s2[3:0]) ? {s1[3:0], s2[3:0]} : {s2[3:0], s1[3:0]};
                            end
                    endcase
                end
        end
    //====================================================================
endmodule
```

除法器设计(DIV.v)

```
module DIV(
    input[3:0] x,
    input[3:0] y,
    input rst,
    input clk,
```

```
    input input_confer,
    output reg [3:0] q,
    output reg [3:0] r,
    output reg error,
    output reg done
);
    reg [3:0] tempx;
    reg [3:0] tempy;
    reg [7:0] temp_x;
    reg [7:0] temp_y;
    reg [3:0] div_count;    // 状态计数
    reg div_on_going;       // 运行标志，对应结束标志done
    //------------------------------------------------
    //division vaild
    always@(*)
        begin
            error = ~(|y);
        end
    //------------------------------------------------
    //operator store
    always@(posedge clk or posedge rst)
        begin
            if(rst)                 begin   tempx <= 4'b0;  tempy <= 4'b0;  end
            else if(input_confer)   begin   tempx <= x;     tempy <= y;     end
            else                    begin   tempx <= tempx; tempy <= tempy; end
        end
    //------------------------------------------------
    //division on going flag
    always@(posedge clk or posedge rst)
        begin
            if(rst)
                begin
                    div_on_going <= 1'b0;
                    done <= 1'b0;
                end
            else if(input_confer && div_on_going == 1'b0)// 状态机，八种状态，四个移位，四个比较-减法，交替出现
                div_on_going <= 1'b1;
            else if(div_count == 4'd8)
                begin
                    div_on_going <= 1'b0;
                    done <= 1'b1;
                end
            else
                div_on_going <= div_on_going;
        end
    //------------------------------------------------
    //division counter
    always@(posedge clk or posedge rst)
        begin
            if(rst)                 div_count <= 4'b0;
```

```
56            else if(div_on_going)   div_count <= div_count + 4'b1;
57            else                    div_count <= 4'b0;
58        end
59    //-----------------------------------------------
60    //division
61    always@(posedge clk or posedge rst)
62        begin
63            if(rst)
64                begin
65                    temp_x = 8'b0;
66                    temp_y = 8'b0;
67                end
68            else if(div_on_going)
69                begin
70                    if(div_count == 4'b0)                   // 四个移位状态
71                        begin
72                            temp_x <= {4'b0,tempx};
73                            temp_y <= {tempy,4'b0};
74                        end
75                    else if(div_count[0] == 1'b1)           // 四个比较-减法状态
76                        temp_x <= {temp_x[6:0],1'b0};
77                    else
78                        temp_x <= (temp_x[7:4] >= temp_y[7:4]) ? (temp_x - temp_y + 1) : temp_x;
79                end
80            else
81                begin
82                    temp_x = 8'b0;
83                    temp_y = 8'b0;
84                end
85        end
86    //-----------------------------------------------
87    //result output
88    always@(posedge done or posedge clk or posedge rst)
89        begin
90            if(rst)
91                begin
92                    q = 4'b0;
93                    r = 4'b0;
94                end
95            else if(done)
96                begin
97                    q = temp_x[3:0];
98                    r = temp_x[7:4];
99                end
100           else
101               begin
102                   q = 4'b0;
103                   r = 4'b0;
104               end
105       end
106   //-----------------------------------------------
```

```
endmodule

```

实验结果：

现场烧录检查：已通过

实现资源消耗与性能统计：

SORT:

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	117	63400	0.18
FF	35	126800	0.03
IO	35	210	16.67
BUFG	1	32	3.13

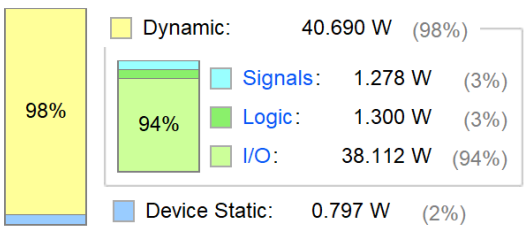
Summary

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**41.487 W (Junction temp exceeded!)**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**125.0°C**
Thermal Margin:	-129.3°C (-27.8 W)
Effective ϑJA:	4.6°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power



Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	inf	Worst Hold Slack (WHS):	inf	Worst Pulse Width Slack (WPWS):	NA
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	NA
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	NA
Total Number of Endpoints:	119	Total Number of Endpoints:	119	Total Number of Endpoints:	NA

There are no user specified timing constraints.

DIV:

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	50	63400	0.08
FF	34	126800	0.03
IO	21	210	10.00
BUFG	1	32	3.13

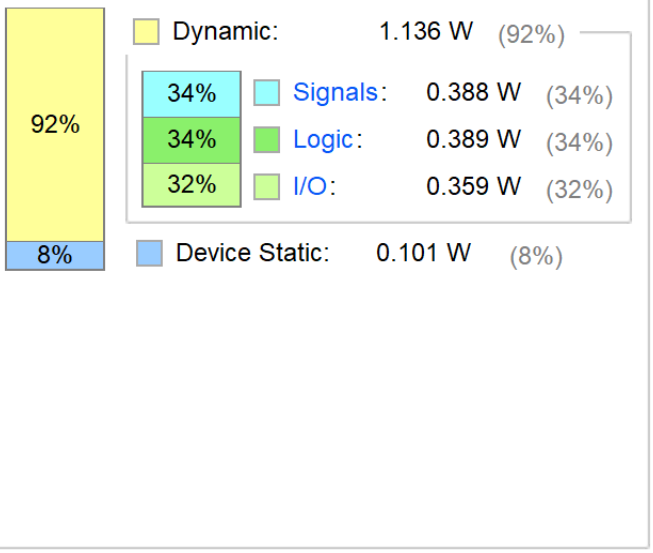
Summary

Power analysis from Implemented netlist. Activity derived from constraints files, simulation files or vectorless analysis.

Total On-Chip Power:	**1.237 W**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**30.6°C**
Thermal Margin:	54.4°C (11.8 W)
Effective ϑJA:	4.6°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power



Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	inf	Worst Hold Slack (WHS):	inf	Worst Pulse Width Slack (WPWS):	NA
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	NA
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	NA
Total Number of Endpoints:	98	Total Number of Endpoints:	98	Total Number of Endpoints:	NA

There are no user specified timing constraints.

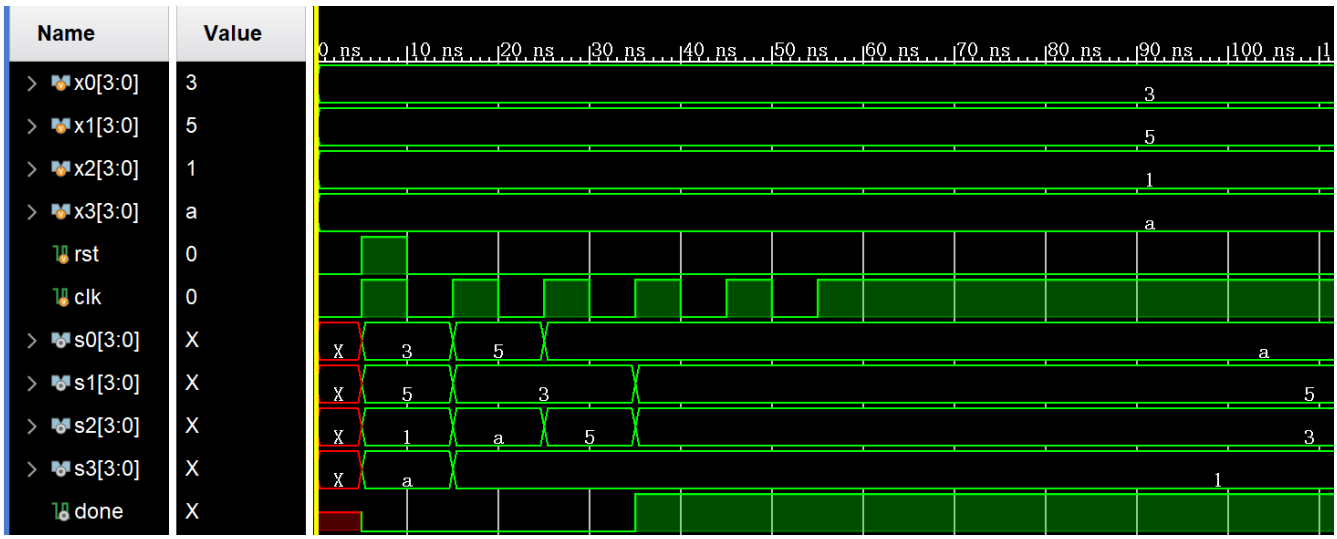
仿真测试结果：

SORT仿真（test_sort.v）

```
module test_sort(

    );
    reg [3:0] x0,x1,x2,x3;
    reg rst,clk;
    wire [3:0] s0,s1,s2,s3;
    wire done;
    SORT s(.x0(x0),.x1(x1),.x2(x2),.x3(x3),.rst(rst),.clk(clk),.s0(s0),.s1(s1),.s2(s2),.s3(s3),.done(done));
    initial
        begin
            x0 = 4'd3;
            x1 = 4'd5;
            x2 = 4'd1;
            x3 = 4'd10;
            rst = 0;
            clk = 0;
            #5
            rst = 1;
            clk = 1;
            #5
            rst = 0;
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
```

```
            #5
            clk = 1;
        end
endmodule

```

结果正确：



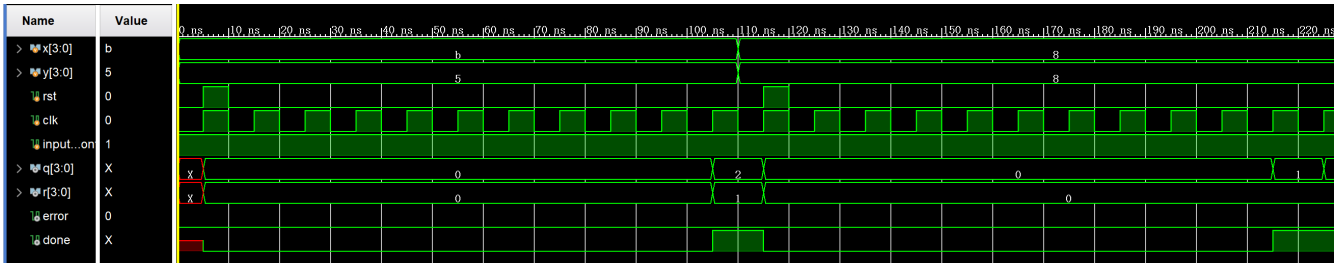
DIV仿真（test_div.v）

```
module test_div(

    );
    reg [3:0] x,y;
    reg rst;
    reg clk;
    reg input_confer;
    wire [3:0] q,r;
    wire error, done;
    DIV d(.x(x), .y(y), .rst(rst), .clk(clk), .input_confer(input_confer), .q(q), .r(r), .error(error), .done(done));
    initial
        begin
            x = 4'd11; // 测试 11 / 5 = 2 ... 1
            y = 4'd5;
            input_confer = 1;
            clk = 0;
            rst = 0;
            #5
            clk = 1;
            rst = 1;
            #5
            clk = 0;
            rst = 0;
            #5
            clk = 1;

```

```
        #5
        clk = 0;
        #5
        clk = 1;
        #5
        clk = 0;
        #5
        clk = 1;
        #5
        clk = 0;
        #5
        clk = 1;
        #5
        clk = 0;
        #5
        clk = 1;
        #5
        clk = 0;
        #5
        clk = 1;
        #5
        clk = 0;
        #5
        clk = 1;
        #5
        clk = 0;
        #5
        clk = 1;
        #5
        clk = 0;
        #5
        clk = 1;
        #5
        clk = 0;
        #5
        clk = 1;
        #5
        //====================================
        x = 4'd8;   // 测试 8 / 8 = 1 ... 0
        y = 4'd8;
        clk = 0;
        rst = 0;
        #5
        clk = 1;
        rst = 1;
        #5
        clk = 0;
        rst = 0;
        #5
        clk = 1;
        #5
```

```
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
            #5
            clk = 1;
            #5
            clk = 0;
        end
endmodule
```

结果正确：



实验总结与感想：

1. 通过实验了解了数据通路与状态机的设计实现，了解了数据通路与状态机的简单应用。
2. 复习了Verilog语法，提高了编程实践能力。