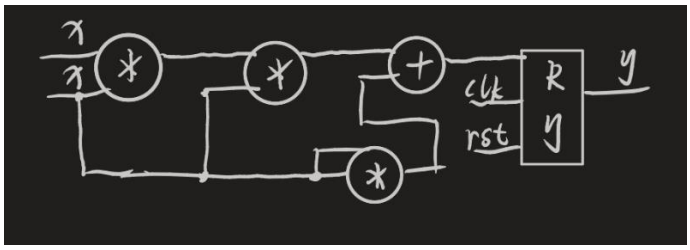
Functional electronic circuits Lab2

Student Name: CAO Xinyang
Student ID: 20321308
Variant: 2

One-cycle device

The picture with microarchitecture:



The timing diagram with simulation results:

One-cycle calculation time: 10 ns

Code of the testbench and the device:

device:

```
module lab2_oc(
    input [31:0] x,
    input rst,
    input clk,
    output reg [31:0] y
);

always @(posedge clk) begin
    if (rst)
        y <= 0;
    else
        y <= x*x*x + x*x;
end
```

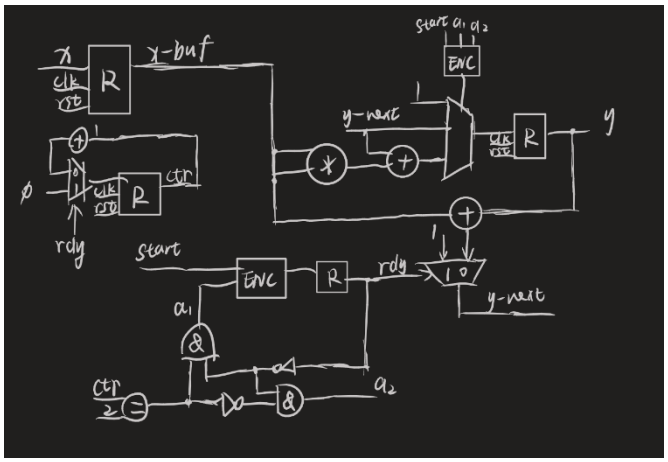
```
endmodule
```

testbench:

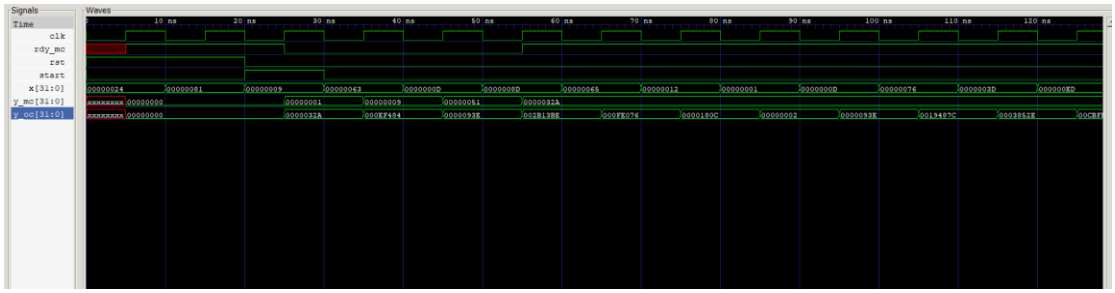
```
`timescale 1ns/1ps

module lab2_tb;

reg [31:0] x;
reg rst, clk;
wire [31:0] y_oc;

lab2_oc uut_oc(
    .clk(clk),
    .rst(rst),
    .x(x),
    .y(y_oc)
);

always #5 clk = ~clk;

always@(negedge clk) begin
    x = ($random % 256) & 8'hFF;
end

initial begin

    $dumpfile("time.vcd");
    $dumpvars(1, lab2_tb);

    clk = 0;
    rst = 1;
    x = 0;

    #20
    rst = 0;

    #100
    $finish;

end
endmodule
```

Multi-cycle device

The picture with microarchitecture:



The timing diagram with simulation results:



Multi-cycle calculation time: 50 ns

Code of the testbench and the device:

device:

```
module lab2_mc(
    input [31:0] x,
    input rst,
    input clk,
    input start,
    output reg [31:0] y,
    output reg rdy
);

reg [31:0] x_buf;
reg [2:0] ctr;

wire [2:0] ctr_next = (rdy)? 0: ctr + 1;
wire [31:0] y_next = (rdy)? 1: y*x_buf;

always @(posedge clk) begin
    if (rst) begin
```

```
        y <= 0;
        rdy <= 1;
        x_buf <= 0;
        ctr <= 0;
    end
    else begin
        ctr <= ctr_next;

        if (start) begin
            y <= 1;
            rdy <= 0;
            x_buf <= x;
        end

        if (!rdy)
            if (ctr == 2) begin
                y <= y_next + x_buf * x_buf;
                rdy <= 1;
            end else
                y <= y_next;
    end

end
endmodule
```

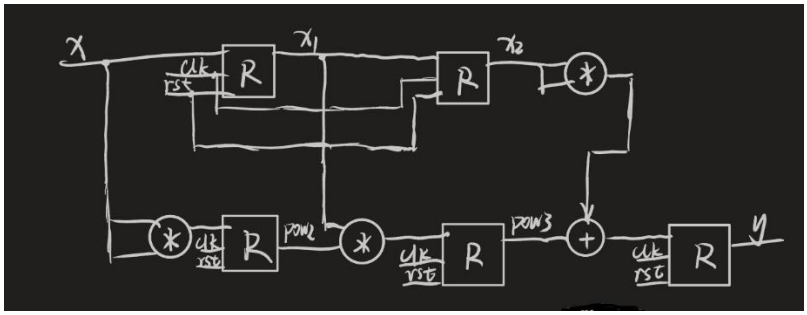
testbench:

```
`timescale 1ns/1ps

module lab2_tb;

reg [31:0] x;
reg rst, clk, start;
wire [31:0] y_oc, y_mc;
wire rdy_mc;

lab2_oc uut_oc(
    .clk(clk),
    .rst(rst),
    .x(x),
    .y(y_oc)
);

lab2_mc uut_mc(
```

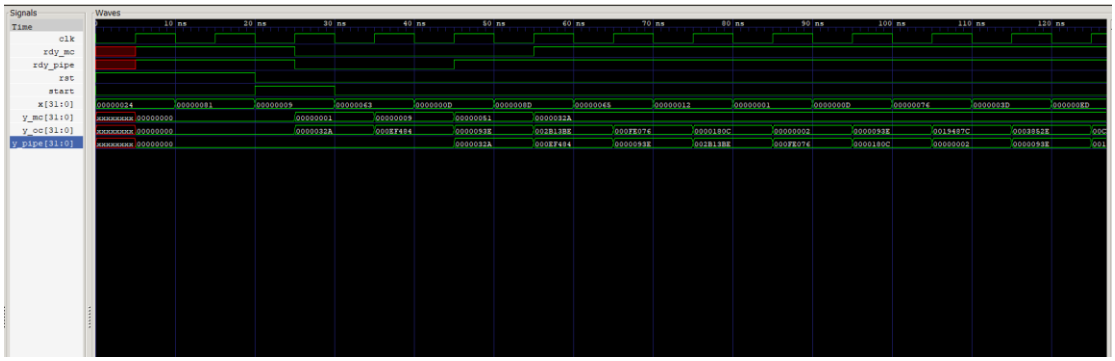
```
    .clk(clk),
    .rst(rst),
    .start(start),
    .x(x),
    .rdy(rdy_mc),
    .y(y_mc)
);

always #5 clk = ~clk;

always@(negedge clk) begin
    x = ($random % 256) & 8'hFF;
end

initial begin

    $dumpfile("time.vcd");
    $dumpvars(1, lab2_tb);

    clk = 0;
    rst = 1;
    x = 0;
    start = 0;

    #20
    rst = 0;
    start = 1;

    #10
    start = 0;

    #100
    start = 1;

    #10
    start = 0;

    #100
    $finish;

end
endmodule
```

Pipelined device

The picture with microarchitecture:



The timing diagram with simulation results:



Pipeline calculation time for first result is 20 ns

for second is 10 ns

Code of the testbench and the device:

device:

```
module lab2_pipe(
    input [31:0] x,
    input rst,
    input clk,
    input start,
    output reg [31:0] y,
    output reg rdy
);

reg [31:0] pow2, pow3;
reg [31:0] x1, x2;
reg rdy1, rdy2;

always @(posedge clk) begin
    if (rst) begin
        y <= 0;
        rdy <= 1;
```

```
        rdy1 <= 0;
        rdy2 <= 0;
        pow2 <= 0;
        pow3 <= 0;
        x1 <= 0;
        x2 <= 0;

    end else begin
        if (start) begin
            rdy <= 0;
            rdy1 <= 1;
        end

        rdy2 <= rdy1;
        rdy <= rdy2;

        x1 <= x;
        x2 <= x1;

        pow2 <= x*x;
        pow3 <= pow2 * x1; //x*x*x
        y <= pow3 + x2 * x2; // x*x*x + x*x
    end
end
endmodule
```

testbench:

```
`timescale 1ns/1ps

module lab2_tb;

reg [31:0] x;
reg rst, clk, start;
wire [31:0] y_oc, y_mc, y_pipe;
wire rdy_mc, rdy_pipe;

lab2_oc uut_oc(
    .clk(clk),
    .rst(rst),
    .x(x),
    .y(y_oc)
);

lab2_mc uut_mc(
```

```
    .clk(clk),
    .rst(rst),
    .start(start),
    .x(x),
    .rdy(rdy_mc),
    .y(y_mc)
);

lab2_pipe uut_pipe(
    .clk(clk),
    .rst(rst),
    .start(start),
    .x(x),
    .rdy(rdy_pipe),
    .y(y_pipe)
);

always #5 clk = ~clk;

always@(negedge clk) begin
    x = ($random % 256) & 8'hFF;
end

initial begin

    $dumpfile("time.vcd");
    $dumpvars(1, lab2_tb);

    clk = 0;
    rst = 1;
    x = 0;
    start = 0;

    #20
    rst = 0;
    start = 1;

    #10
    start = 0;

    #100
    start = 1;

    #10
```

```
    start = 0;

    #100
    $finish;

end
endmodule
```