Lab 5 report

PB22051022王嘉宁

实验目的与内容

> 将上一次实验设计的 CPU 流水化，形成一个不考虑冒险的流水线 CPU。

逻辑设计

> 任务 2：搭建 无冒险流水线 关键代码段

CPU连接

```
module CPU (...
);
Intersegment_REG IF_ID(
    .clk        (            clk),
    .rst        (            rst),
    .en         (      global_en),
    .stall      (          stall),
    .flush      (          flush),

    .pcadd4_in      (      pcadd4_if),
    .pcadd4_out     (      pcadd4_id),
    .pc_in      (          pc_if),
    .pc_out     (          pc_id),
    .inst_in        (     imem_rdata),
    .inst_out       (        inst_id),
    .commit_in      (      commit_if),
    .commit_out     (      commit_id),

    .imm_in     (      32'd0),
    .imm_out        (       ),
    .rf_wa_in       (           5'd0),
    .rf_wa_out      (       ),
    .rf_rd0_in      (          32'd0),
    .rf_rd0_out     (       ),
    .rf_rd1_in      (          32'd0),
    .rf_rd1_out     (       ),
    .rf_we_in       (           1'd0),
    .rf_we_out      (       ),
    .rf_wd_sel_in   (           2'd0),
    .rf_wd_sel_out  (       ),
    .alu_op_in      (           5'd0),
    .alu_op_out     (       ),
    .alu_src0_sel_in    (           1'd0),
    .alu_src0_sel_out   (       ),
```

```
    .alu_src1_sel_in    (           1'd0),
    .alu_src1_sel_out   (       ),
    .dmem_access_in (           4'd0),
    .dmem_access_out    (       ),
    .br_type_in     (           4'd0),
    .br_type_out    (       ),

    .alu_res_in     (          32'd0),
    .alu_res_out    (       ),

    .dmem_rd_out_in (          32'd0),
    .dmem_rd_out_out    (       ),
    .dmem_wdata_in  (          32'd0),
    .dmem_wdata_out (       ),
    .dmem_we_in     (           1'd0),
    .dmem_we_out    (       )
);
Intersegment_REG ID_EX(
    .clk        (            clk),
    .rst        (            rst),
    .en         (      global_en),
    .stall      (          stall),
    .flush      (          flush),

    .pcadd4_in      (      pcadd4_id),
    .pcadd4_out     (      pcadd4_ex),
    .pc_in      (          pc_id),
    .pc_out     (          pc_ex),
    .inst_in        (        inst_id),
    .inst_out       (        inst_ex),
    .commit_in      (      commit_id),
    .commit_out     (      commit_ex),

    .imm_in     (     imm_id),
    .imm_out        (         imm_ex),
    .rf_wa_in       (       rf_wa_id),
    .rf_wa_out      (   rf_wa_ex),
    .rf_rd0_in      (      rf_rd0_id),
    .rf_rd0_out     (      rf_rd0_ex),
    .rf_rd1_in      (      rf_rd1_id),
    .rf_rd1_out     (      rf_rd1_ex),
    .rf_we_in       (       rf_we_id),
    .rf_we_out      (       rf_we_ex),
    .rf_wd_sel_in   (   rf_wd_sel_id),
    .rf_wd_sel_out  (   rf_wd_sel_ex),
    .alu_op_in      (      alu_op_id),
    .alu_op_out     (      alu_op_ex),
    .alu_src0_sel_in    (alu_src0_sel_id),
    .alu_src0_sel_out   (alu_src0_sel_ex),
    .alu_src1_sel_in    (alu_src1_sel_id),
    .alu_src1_sel_out   (alu_src1_sel_ex),
```

```
    .dmem_access_in ( dmem_access_id),
    .dmem_access_out    ( dmem_access_ex),
    .br_type_in     (     br_type_id),
    .br_type_out    (     br_type_ex),

    .alu_res_in     (          32'd0),
    .alu_res_out    (       ),

    .dmem_rd_out_in (          32'd0),
    .dmem_rd_out_out    (       ),
    .dmem_wdata_in  (          32'd0),
    .dmem_wdata_out (       ),
    .dmem_we_in     (     dmem_we_id),
    .dmem_we_out    (     dmem_we_ex)
);
Intersegment_REG EX_MEM(
    .clk        (             clk),
    .rst        (             rst),
    .en         (       global_en),
    .stall      (           stall),
    .flush      (            1'b0),

    .pcadd4_in      (       pcadd4_ex),
    .pcadd4_out     (      pcadd4_mem),
    .pc_in      (           pc_ex),
    .pc_out     (          pc_mem),
    .inst_in        (         inst_ex),
    .inst_out       (        inst_mem),
    .commit_in      (       commit_ex),
    .commit_out     (      commit_mem),

    .imm_in     (      imm_ex),
    .imm_out        (         imm_mem),
    .rf_wa_in       (        rf_wa_ex),
    .rf_wa_out      (       rf_wa_mem),
    .rf_rd0_in      (       rf_rd0_ex),
    .rf_rd0_out     (      rf_rd0_mem),
    .rf_rd1_in      (       rf_rd1_ex),
    .rf_rd1_out     (      rf_rd1_mem),
    .rf_we_in       (        rf_we_ex),
    .rf_we_out      (       rf_we_mem),
    .rf_wd_sel_in   (    rf_wd_sel_ex),
    .rf_wd_sel_out  (   rf_wd_sel_mem),
    .alu_op_in      (       alu_op_ex),
    .alu_op_out     (      alu_op_mem),
    .alu_src0_sel_in    ( alu_src0_sel_ex),
    .alu_src0_sel_out   (alu_src0_sel_mem),
    .alu_src1_sel_in    ( alu_src1_sel_ex),
    .alu_src1_sel_out   (alu_src1_sel_mem),
    .dmem_access_in (  dmem_access_ex),
    .dmem_access_out    ( dmem_access_mem),
```

```
    .br_type_in     (      br_type_ex),
    .br_type_out    (     br_type_mem),

    .alu_res_in     (      alu_res_ex),
    .alu_res_out    (     alu_res_mem),

    .dmem_rd_out_in (           32'd0),
    .dmem_rd_out_out    (        ),
    .dmem_wdata_in  (           32'd0),
    .dmem_wdata_out (            ),
    .dmem_we_in     (      dmem_we_ex),
    .dmem_we_out    (     dmem_we_mem)
);
Intersegment_REG MEM_WB(
    .clk        (             clk),
    .rst        (             rst),
    .en         (       global_en),
    .stall      (           stall),
    .flush      (            1'b0),

    .pcadd4_in      (      pcadd4_mem),
    .pcadd4_out     (       pcadd4_wb),
    .pc_in      (          pc_mem),
    .pc_out     (           pc_wb),
    .inst_in        (        inst_mem),
    .inst_out       (         inst_wb),
    .commit_in      (      commit_mem),
    .commit_out     (       commit_wb),

    .imm_in     (         imm_mem),
    .imm_out        (          imm_wb),
    .rf_wa_in       (       rf_wa_mem),
    .rf_wa_out      (        rf_wa_wb),
    .rf_rd0_in      (      rf_rd0_mem),
    .rf_rd0_out     (       rf_rd0_wb),
    .rf_rd1_in      (      rf_rd1_mem),
    .rf_rd1_out     (       rf_rd1_wb),
    .rf_we_in       (       rf_we_mem),
    .rf_we_out      (        rf_we_wb),
    .rf_wd_sel_in   (   rf_wd_sel_mem),
    .rf_wd_sel_out  (    rf_wd_sel_wb),
    .alu_op_in      (      alu_op_mem),
    .alu_op_out     (       alu_op_wb),
    .alu_src0_sel_in    (alu_src0_sel_mem),
    .alu_src0_sel_out   ( alu_src0_sel_wb),
    .alu_src1_sel_in    (alu_src1_sel_mem),
    .alu_src1_sel_out   ( alu_src1_sel_wb),
    .dmem_access_in ( dmem_access_mem),
    .dmem_access_out    (  dmem_access_wb),
    .br_type_in     (     br_type_mem),
    .br_type_out    (      br_type_wb),
```

```
    .alu_res_in     (     alu_res_mem),
    .alu_res_out    (      alu_res_wb),

    .dmem_rd_out_in ( dmem_rd_out_mem),
    .dmem_rd_out_out    (  dmem_rd_out_wb),
    .dmem_wdata_in  (  dmem_wdata_mem),
    .dmem_wdata_out (   dmem_wdata_wb),
    .dmem_we_in     (     dmem_we_mem),
    .dmem_we_out    (      dmem_we_wb)
);
MUX2 mux2(
    .src0   (     pcadd4_wb),
    .src1   (    alu_res_wb),
    .src2   (dmem_rd_out_wb),
    .src3   (         32'd0),
    .sel    (  rf_wd_sel_wb),
    .res    (      rf_wd_wb)
);
always @(posedge clk) begin
    if (rst) begin ...
    end
    else if (global_en) begin
        commit_reg          <= commit_wb;
        commit_pc_reg       <= pc_wb;
        commit_inst_reg     <= inst_wb;
        commit_halt_reg     <= inst_wb == `HALT_INST;
        commit_reg_we_reg   <= rf_we_wb;
        commit_reg_wa_reg   <= rf_wa_wb;
        commit_reg_wd_reg   <= rf_wd_wb;
        commit_dmem_we_reg  <= dmem_we_wb;
        commit_dmem_wa_reg  <= alu_res_wb;
        commit_dmem_wd_reg  <= dmem_wdata_wb;
    end
end
endmodule
```

Intersegment_REG

```
module Intersegment_REG (...
);

always @(posedge clk) begin
    if (rst) begin
        pcadd4_out          <=  32'h1c00_0004   ;
        pc_out              <=  32'h1c00_0000   ;
        inst_out            <=  32'h0       ;
        rf_rd0_out          <=  32'h0       ;
        rf_rd1_out          <=  32'h0       ;
```

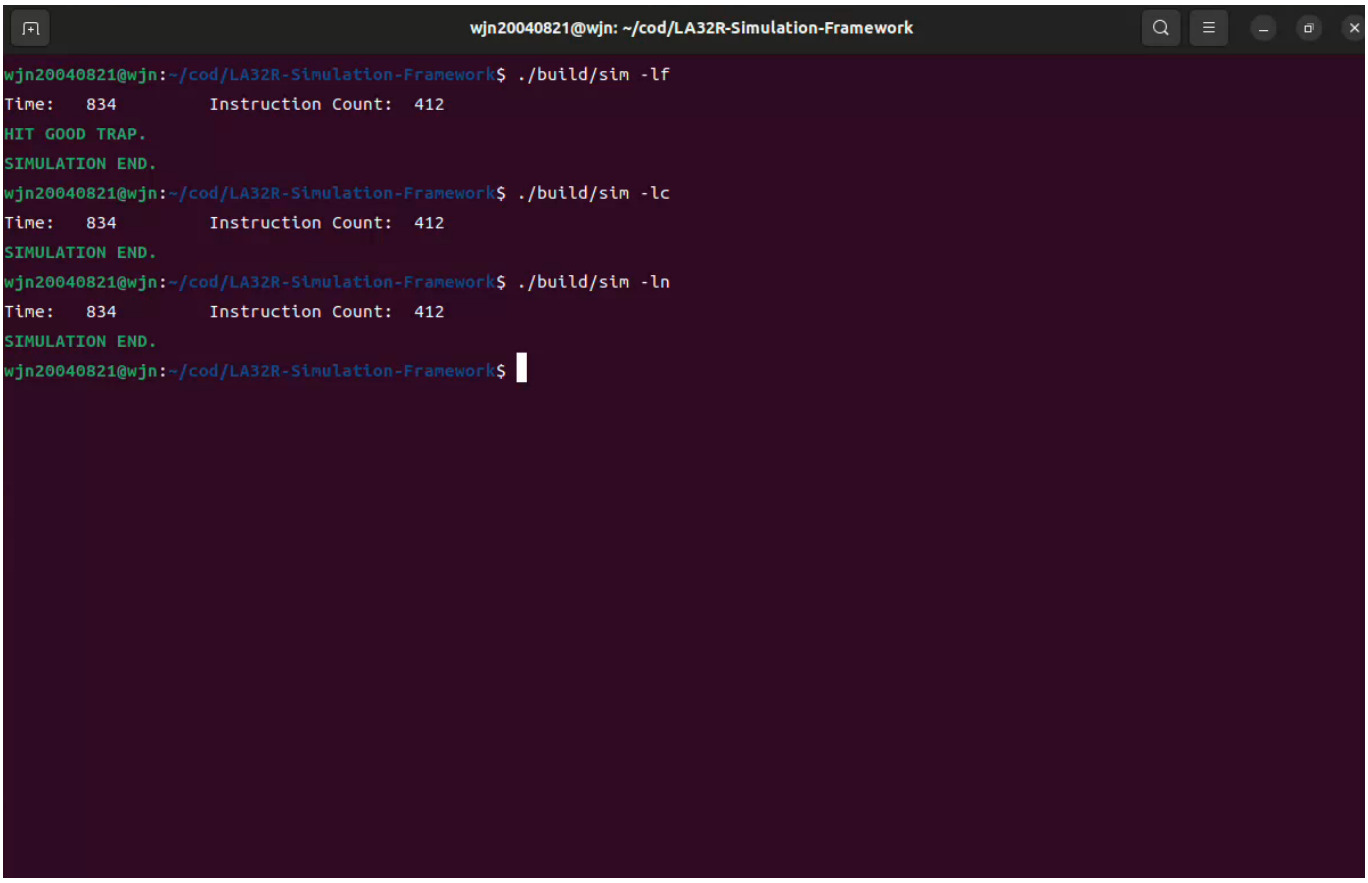
```
            imm_out             <=  32'h0       ;
            rf_wa_out           <=   5'h0       ;
            rf_we_out           <=   1'h0       ;
            rf_wd_sel_out       <=   2'h0       ;
            alu_op_out          <=   5'h0           ;
            alu_src0_sel_out    <=   1'h0       ;
            alu_src1_sel_out    <=   1'h0       ;
            dmem_access_out     <=   4'h0           ;
            br_type_out         <=   4'h0           ;
            alu_res_out         <=  32'h0       ;
            dmem_rd_out_out     <=  32'h0       ;
            dmem_we_out         <=   1'h0       ;
            commit_out          <=   1'h0       ;
            dmem_wdata_out      <=  32'h0       ;
        end
        else if (en) begin
            if (flush) begin
                pcadd4_out          <=  32'h1c00_0004   ;
                pc_out              <=  32'h1c00_0000   ;
                inst_out            <=  32'h0000_0000   ;
                rf_rd0_out          <=  32'h0       ;
                rf_rd1_out          <=  32'h0       ;
                imm_out             <=  32'h0       ;
                rf_wa_out           <=   5'h0       ;
                rf_we_out           <=   1'h0       ;
                rf_wd_sel_out       <=   2'h0       ;
                alu_op_out          <=   5'h0           ;
                alu_src0_sel_out    <=   1'h0       ;
                alu_src1_sel_out    <=   1'h0       ;
                dmem_access_out     <=   4'h0           ;
                br_type_out         <=   4'h0           ;
                alu_res_out         <=  32'h0       ;
                dmem_rd_out_out     <=  32'h0       ;
                dmem_we_out         <=   1'h0       ;
                commit_out          <=   1'h0       ;
                dmem_wdata_out      <=  32'h0       ;
            end
            else if (!stall) begin
                pcadd4_out          <=  pcadd4_in   ;
                pc_out              <=  pc_in       ;
                inst_out            <=  inst_in     ;
                rf_rd0_out          <=  rf_rd0_in   ;
                rf_rd1_out          <=  rf_rd1_in   ;
                imm_out             <=  imm_in      ;
                rf_wa_out           <=  rf_wa_in    ;
                rf_we_out           <=  rf_we_in    ;
                rf_wd_sel_out       <=  rf_wd_sel_in    ;
                alu_op_out          <=  alu_op_in   ;
                alu_src0_sel_out    <=  alu_src0_sel_in ;
                alu_src1_sel_out    <=  alu_src1_sel_in ;
                dmem_access_out     <=  dmem_access_in  ;
```

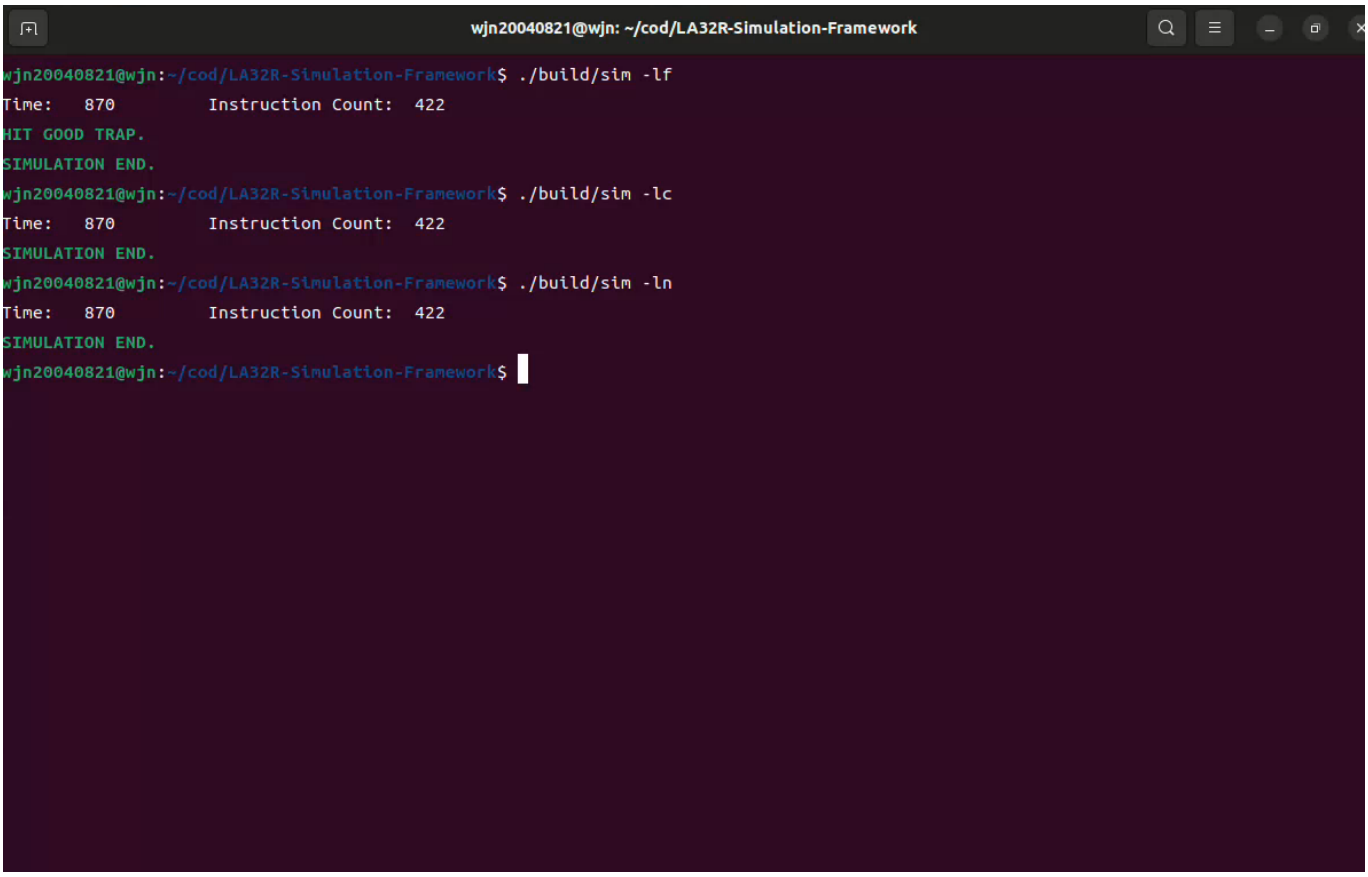
```
            br_type_out         <=  br_type_in  ;
            alu_res_out         <=  alu_res_in  ;
            dmem_rd_out_out     <=  dmem_rd_out_in  ;
            dmem_we_out         <=  dmem_we_in  ;
            commit_out          <=  commit_in   ;
            dmem_wdata_out      <=  dmem_wdata_in   ;
        end
    end
end

endmodule
```

REGFILE

```
module REGFILE(...
);
    reg [31 : 0] reg_file [0 : 31];

    integer i;
    initial begin
        for (i = 0; i < 32; i = i + 1)
            reg_file[i] = 0;
    end

    always @(posedge clk) begin
        if(rf_we && rf_wa != 0) begin
            reg_file[rf_wa] <= rf_wd;
        end
    end
    always @(*)begin
        if(rf_we && rf_ra0 == rf_wa)begin
            rf_rd0 = (rf_wa == 0) ? 32'd0 : rf_wd;
        end
        else begin
            rf_rd0 = reg_file[rf_ra0];
        end
    end
    always @(*)begin
        if(rf_we && rf_ra1 == rf_wa)begin
            rf_rd1 = (rf_wa == 0) ? 32'd0 : rf_wd;
        end
        else begin
            rf_rd1 = reg_file[rf_ra1];
        end
    end
    always @(*) begin
        dbg_reg_rd = reg_file[dbg_reg_ra];
    end
endmodule
```

测试结果与分析





上板结果

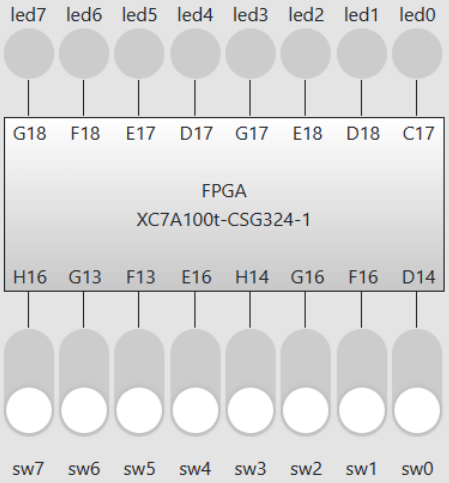
FPGAOL NG DEV PAGE

Bitstream File

Select file | example bitstream | C:\fakepath\TOP.bit | Program!

Program success!

FPGA interface



uart show>

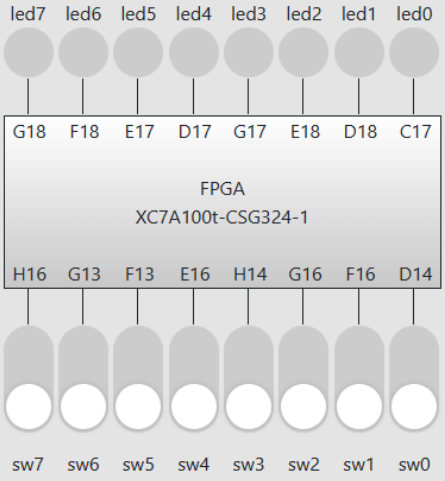
```
FPGAOL UART xterm.js 1.1

USTC COD Project
User: R;
---------- CPU ----------


User: RR 0 32;
00000000 1C00064C 5FB208BC 1C0005F8
00000000 8DEC18DA 00000000 1C00064C
00000000 1C801999 83998EA4 9F999793
00000000 00000000 00000000 1C8020D6
00000000 1C8018B8 00000000 1C80174F
CB8CDA5E 00000000 00000000 00000000
F2089F91 CA329FA7 6BC44DC7 1C013600
1C800C67 0D954809 1C80164D 1C8022B6


User:
```

uart pins: cts rts rxd txd
xdc sym: D3 E5 D4 C4
baud rate: 115200

RR 0 32;\n

input

FPGA interface



uart show>

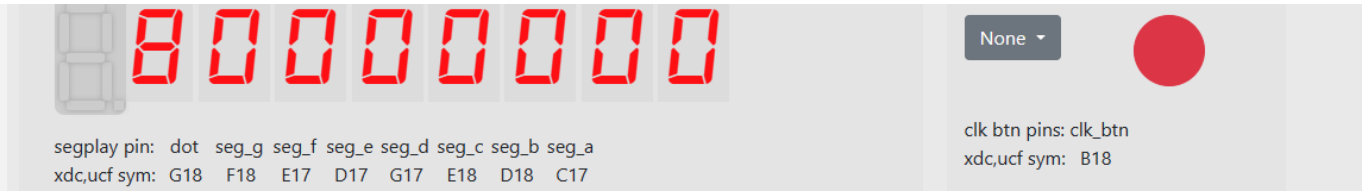
```
User: RR 0 32;
00000000 00000000 000000B4 00000000
00000000 FFE753B8 1939E000 1939E000
FFFFFFF9 3266B82D 00000001 00000000
B432D000 B432D000 000000B4 00000001
0D49774C 188DB25C 000001AD 00000000
00000000 F89ECD69 00000001 B4F217ED
00000000 000007AA 00000000 00000000
014BE630 00000001 00000001 419D7600


User: RR 0 32;
00000000 00000000 000000B4 00000000
00000000 FFE753B8 1939E000 1939E000
FFFFFFF9 3266B82D 00000001 00000000
B432D000 B432D000 000000B4 00000001
0D49774C 188DB25C 000001AD 00000000
00000000 F89ECD69 00000001 B4F217ED
00000000 000007AA 00000000 00000000
014BE630 00000001 00000001 419D7600


User:
```

uart pins: cts rts rxd txd
xdc sym: D3 E5 D4 C4
baud rate: 115200

RR 0 32;\n

input

segplay(sharing with led) hexplay soft clock button



总结

> 将上一次实验设计的 CPU 流水化，形成了一个不考虑冒险的流水线 CPU。