Khối PC

```
module register
(
    input clk,
    input rst_n,
    input [31:0] pc_new,
    output reg [31:0] pc
);

always @(posedge clk or posedge rst_n)
    if(rst_n)
        pc <= {32{1'b0}};
    else
        pc <= pc_new;
endmodule

```

Khối Rom

```
module rom
(
    input [5:0] imAdrr,
    output [31:0] imData
);
    reg [31:0] rom [64 - 1:0];
    assign imData = rom[imAdrr];

    initial begin
        $readmemh("program.hex", rom);
    end
endmodule
```

Khối Register File

```
1  module register_file
2  (
3     input clk,
4     input [4:0] a1,
5     input [4:0] a2,
6     input [4:0] a3,
7     input [31:0] wd3,
8     input we3,
9     output [31:0] rd1,
10    output [31:0] rd2
11 );
12    reg [31:0] rf [31:0];
13    assign rd1 = (a1 != 0) ? rf[a1] : 32'b0;
14    assign rd2 = (a2 != 0) ? rf[a2] : 32'b0;
15    always @ (posedge clk)
16       if(we3) rf[a3] <= wd3;
17 endmodule
```

Khối Ram

```
1  module ram
2  (
3     input clk,
4     input [31:0] a,
5     input we,
6     input [31:0] wd,
7     output [31:0] rd
8  );
9     reg [31:0] ram [64 - 1:0];
10    assign rd = ram [a[31:2]];
11    always @(posedge clk)
12       if (we)
13          ram[a[31:2]] <= wd;
14 endmodule
```

Khối ALU

```
1  module alu
2  (
3      input [31:0] srcA,
4      input [31:0] srcB,
5      input [3:0] oper,
6      input [4:0] shift,
7      output zero,
8      output reg [31:0] result
9  );
10 always @ (*) begin
11     case (oper)
12         default : result = srcA + srcB;
13         4'b0000 : result = ~srcA;
14         4'b0001 : result = srcA & srcB;
15         4'b0010 : result = srcA ^ srcB;
16         4'b0011 : result = srcA | srcB;
17         4'b0100 : result = srcA - 1;
18         4'b0101 : result = srcA + srcB;
19         4'b0110 : result = srcA - srcB;
20         4'b0111 : result = srcA + 1;
21         4'b1000 : result = (srcA < srcB) ? 1 : 0;
22         4'b1010 : result = srcB << shift;
23         4'b1011 : result = srcB >> shift;
24         4'b1100 : result = (srcB << 16);
25     endcase
26 end
27     assign zero = (result == 0);
28 endmodule
```

Khối Control

```
module control_unit
(
    input [5:0] opcode,
    input [5:0] funct,
    output reg condZero,
    output reg branch,
    output reg regWrite,
    output reg [1:0] regDst,
    output reg ALUSrc,
    output reg [3:0] ALUOp,
    output reg [1:0] pcSrc,
    output reg memWrite,
    output reg [1:0] memToReg
);

always @(*) begin
      case(opcode)
         6'b0: begin
            case(funct)
               6'b100001: begin
                  ALUOp = 4'b0101;
                  regDst = 2'b01;
                  regWrite = 1;
                  branch = 0;
                  condZero = 0;
                  ALUSrc = 0;
                  memWrite = 0;
                  memToReg = 2'b00;
                  pcSrc = 2'b00;
               end
               6'b100011: begin
                  ALUOp = 4'b0110;
                  regDst = 2'b01;
                  regWrite = 1;
```

```
35                branch = 0;
36                condZero = 0;
37                ALUSrc = 0;
38                memWrite = 0;
39                memToReg = 2'b00;
40                pcSrc = 2'b00;
41            end
42            6'b100100: begin
43                ALUOp = 4'b0001;
44                regDst = 2'b01;
45                regWrite = 1;
46                branch = 0;
47                condZero = 0;
48                ALUSrc = 0;
49                memWrite = 0;
50                memToReg = 2'b00;
51                pcSrc = 2'b00;
52            end
53            6'b100101: begin
54                ALUOp = 4'b0011;
55                regDst = 2'b01;
56                regWrite = 1;
57                branch = 0;
58                condZero = 0;
59                ALUSrc = 0;
60                memWrite = 0;
61                memToReg = 2'b00;
62                pcSrc = 2'b00;
63            end
64            6'b100110: begin
65                ALUOp = 4'b0010;
66                regDst = 2'b01;
67                regWrite = 1;
```

```
68                  branch = 0;
69                  condZero = 0;
70                  ALUSrc = 0;
71                  memWrite = 0;
72                  memToReg = 2'b00;
73                  pcSrc = 2'b00;
74              end
75              6'b101011: begin
76                  ALUOp = 4'b1000;
77                  regDst = 2'b01;
78                  regWrite = 1;
79                  branch = 0;
80                  condZero = 0;
81                  ALUSrc = 0;
82                  memWrite = 0;
83                  memToReg = 2'b00;
84                  pcSrc = 2'b00;
85              end
86              6'b0: begin
87                  ALUOp = 4'b1010;
88                  regDst = 2'b01;
89                  regWrite = 1;
90                  branch = 0;
91                  condZero = 0;
92                  ALUSrc = 0;
93                  memWrite = 0;
94                  memToReg = 2'b00;
95                  pcSrc = 2'b00;
96              end
97              6'b000010: begin
```

```
 97                  6'b000010: begin
 98                      ALUOp = 4'b1011;
 99                      regDst = 2'b01;
100                      regWrite = 1;
101                      branch = 0;
102                      condZero = 0;
103                      ALUSrc = 0;
104                      memWrite = 0;
105                      memToReg = 2'b00;
106                      pcSrc = 2'b00;
107                  end
108                  6'b001000: begin
109                      ALUOp = 4'b0110;
110                      regDst = 2'b01;
111                      regWrite = 0;
112                      branch = 0;
113                      condZero = 0;
114                      ALUSrc = 0;
115                      memWrite = 0;
116                      memToReg = 2'b00;
117                      pcSrc = 2'b10;
118                  end
119                  default: begin
120                       ALUOp    = 4'b0000;
121                       regDst   = 2'b00;
122                       regWrite = 0;
123                       branch   = 0;
124                       condZero = 0;
125                       ALUSrc   = 0;
126                       memWrite = 0;
127                       memToReg = 2'b00;
```

```
128                     pcSrc     = 2'b00;
129                 end
130             endcase
131         end
132         6'b001001: begin
133             ALUOp = 4'b0101;
134             regDst = 2'b00;
135             regWrite = 1;
136             branch = 0;
137             condZero = 0;
138             ALUSrc = 1;
139             memWrite = 0;
140             memToReg = 2'b00;
141             pcSrc = 2'b00;
142         end
143         6'b001100: begin
144             ALUOp = 4'b0001;
145             regDst = 2'b00;
146             regWrite = 1;
147             branch = 0;
148             condZero = 0;
149             ALUSrc = 1;
150             memWrite = 0;
151             memToReg = 2'b00;
152             pcSrc = 2'b00;
153         end
154         6'b001101: begin
155             ALUOp = 4'b0011;
156             regDst = 2'b00;
157             regWrite = 1;
```

```
158             branch = 0;
159             condZero = 0;
160             ALUSrc = 1;
161             memWrite = 0;
162             memToReg = 2'b00;
163             pcSrc = 2'b00;
164         end
165         6'b001011: begin
166             ALUOp = 4'b1000;
167             regDst = 2'b00;
168             regWrite = 1;
169             branch = 0;
170             condZero = 0;
171             ALUSrc = 1;
172             memWrite = 0;
173             memToReg = 2'b00;
174             pcSrc = 2'b00;
175         end
176         6'b001111: begin
177             ALUOp = 4'b1100;
178             regDst = 2'b00;
179             regWrite = 1;
180             branch = 0;
181             condZero = 0;
182             ALUSrc = 1;
183             memWrite = 0;
184             memToReg = 2'b00;
185             pcSrc = 2'b00;
186         end
187         6'b000100: begin
```

```
187             6'b000100: begin
188                 ALUOp = 4'b0110;
189                 regDst = 2'b00;
190                 regWrite = 0;
191                 branch = 1;
192                 condZero = 1;
193                 ALUSrc = 1;
194                 memWrite = 0;
195                 memToReg = 2'b00;
196                 pcSrc = 2'b00;
197             end
198             6'b000101: begin
199                 ALUOp = 4'b0110;
200                 regDst = 2'b00;
201                 regWrite = 0;
202                 branch = 1;
203                 condZero = 0;
204                 ALUSrc = 1;
205                 memWrite = 0;
206                 memToReg = 2'b00;
207                 pcSrc = 2'b00;
208             end
209             6'b100011: begin
210                 ALUOp = 4'b0101;
211                 regDst = 2'b00;
212                 regWrite = 1;
213                 branch = 0;
214                 condZero = 0;
```

```
215             ALUSrc = 1;
216             memWrite = 0;
217             memToReg = 2'b01;
218             pcSrc = 2'b00;
219         end
220         6'b101011: begin
221             ALUOp = 4'b0101;
222             regDst = 2'b00;
223             regWrite = 0;
224             branch = 0;
225             condZero = 0;
226             ALUSrc = 1;
227             memWrite = 1;
228             memToReg = 2'b00;
229             pcSrc = 2'b00;
230         end
231         6'b000010: begin
232             ALUOp = 4'b0110;
233             regDst = 2'b00;
234             regWrite = 0;
235             branch = 0;
236             condZero = 0;
237             ALUSrc = 0;
238             memWrite = 0;
239             memToReg = 2'b00;
240             pcSrc = 2'b01;
241         end
242         6'b000011: begin
243             ALUOp = 4'b0110;
244             regDst = 2'b10;
```

```
            regWrite = 1;
            branch = 0;
            condZero = 0;
            ALUSrc = 0;
            memWrite = 0;
            memToReg = 2'b10;
            pcSrc = 2'b01;
        end
        default: begin
            ALUOp    = 4'b0000;
            regDst   = 2'b00;
            regWrite = 0;
            branch   = 0;
            condZero = 0;
            ALUSrc   = 0;
            memWrite = 0;
            memToReg = 2'b00;
            pcSrc    = 2'b00;
        end
    endcase
  end
endmodule
```

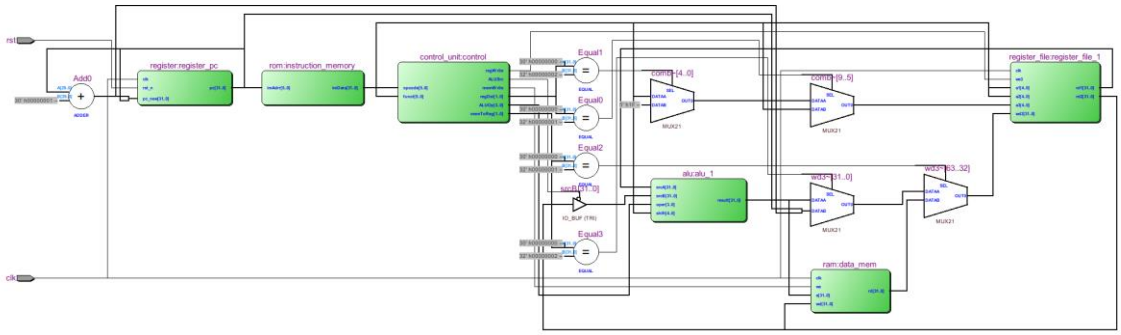
MIPS

```
module mips
(
    input clk,
    input rst
);
    wire [31:0] pc, instr, pcNext;
    wire [5:0] imAddr;
    wire regWrite, ALUSrc;
    wire [1:0] regDst, memToReg;
    wire [3:0] ALUOp;
    wire [31:0] rd1, rd2;
    wire [4:0] a3 = (regDst == 1) ? instr[15:11] : (regDst == 2) ? 5'b11111 : instr[20:16];
    wire [31:0] wd3;
    wire [31:0] RDmem;
    wire [31:0] srcB = ALUSrc ? 32'bZ : rd2;
    wire [31:0] aluResult;
    wire aluZero;

    assign imAddr = pc >> 2;
    assign wd3 = (memToReg == 1) ? RDmem : (memToReg == 2) ? pcNext : aluResult;
    assign pcNext = pc + 4;
    register register_pc(clk, rst, pcNext,pc);
    register_file register_file_1(clk, instr[25:21], instr[20:16], a3, wd3, regWrite, rd1, rd2);
    rom instruction_memory(imAddr, instr);
    ram data_mem(clk, aluResult, memWrite, rd2, RDmem);
    alu alu_1(rd1, srcB, ALUOp, instr[10:6], aluZero, aluResult);
    control_unit control(instr[31:26], instr[5:0], , , regWrite, regDst, ALUSrc, ALUOp, , memWrite, memToReg);
endmodule

```

RTL


rst
clk
Add0
register:register_pc
rom:instruction_memory
control_unit:control
Equal1
Equal0
Equal2
Equal3
comb[4..0]
comb[9..5]
alu:alu_1
wd3[31..0]
wd3[63..32]
ram:data_mem
register_file:register_file_1
MUX21
ADDER
EQUAL
IO_BUF (TRI)