```
module CORE(
    input clk,
    input  [15:0]data_DM_core,//from data mem to proc(MDDR)
    input  [15:0]data_IM_core,
    input  [15:0]proId,
    output reg [15:0]data_core_IM,
    output reg [15:0]data_core_DM,//from processor(MDDR) to data mem (data)
    output reg [0:0]DM_en,//from processor to data mem (control signal)
    output reg [0:0]IM_en,
    output reg [15:0]AR_out,
    output reg finish,
    output reg [15:0]bus,
    output reg [7:0]current_micro_instruction

    );
    wire [7:0]current_micro_instruction_wire;
    wire to_DM;
    wire to_IM;
    wire [0:0]wire_decoder_registers [16:0];
    wire [15:0] bus_wire;
    wire [15:0] MIDR_wire;
    wire [15:0] wire_iram_mddr;
    wire [15:0] wire_dram_mddr;
    wire [15:0] Base_A_wire;
    wire [15:0] Base_B_wire;
    wire [15:0] Base_C_wire;
    wire [15:0] data_alu_ac;
    wire [15:0] wire_registers_busMultiplexer [12:0];
    wire [0:0] PC_inc_wire;
    wire [0:0] iflag_wire;
    wire [0:0] jflag_wire;
    wire [0:0] kflag_wire;
    wire [0:0] zflag_wire;
    wire [15:0] i_ref_wire;
    wire [15:0] j_ref_wire;
    wire [15:0] k_ref_wire;
    wire [3:0] opcode_wire;
    wire [2:0] ac_cu_wire;
    wire [5:0] wire_cu_decoder;//select reg by cu
    wire [5:0] wire_regSelector_decoder;//select read/write by reg selector m
    wire [4:0] bus_selector_from_cu; //
    wire [4:0] bus_selector_from_register_selector;
    wire [0:0] en_register_selector;//form CU - enable
    wire [0:0] rw_register_selector;//from cu - read/wrie select
    wire [0:0] finish_wire;//from cu - read/wrie select

    assign wire_iram_mddr = data_IM_core;
    assign wire_dram_mddr = data_DM_core;

    always@(*)
     begin
        current_micro_instruction <= current_micro_instruction_wire;
        bus <= bus_wire;
        IM_en <= to_IM;
```

```
55          DM_en <= to_DM;
56          AR_out <= wire_registers_busMultiplexer[0];
57          data_core_IM <= wire_registers_busMultiplexer[2];
58          data_core_DM <= wire_registers_busMultiplexer[2];
59          finish <= finish_wire ;
60          if(finish==1'd1) $display("finish");
61          //$display("AR:- %d , I:- %d, I_ref:- %d, J:- %d, J_ref:- %d, base_A:
    ,wire_registers_busMultiplexer[0],wire_registers_busMultiplexer[4],i_ref_wire
62      end
63
64      GENERAL_PURPOSE_REGISTER ID (.clk(clk), .write(1'd1), .data_in(proId),.d
65
66      GENERAL_PURPOSE_REGISTER AR (.clk(clk), .write(wire_decoder_registers[0]
67
68      PC PC (.clk(clk), .write(wire_decoder_registers[1]), .incpc(PC_inc_wire)
69
70      GENERAL_PURPOSE_REGISTER MIDR (.clk(clk), .write(wire_decoder_registers[
71
72      MDDR MDDR (.clk(clk), .write_ins(wire_decoder_registers[4]),.write_data(
73          .write_bus(wire_decoder_registers[3]),.instr_iram_mddr(wire_iram_mddr
74          .data_bus_mddr(bus_wire),.data_out(wire_registers_busMultiplexer[2]))
75
76      GENERAL_PURPOSE_REGISTER BASE_C (.clk(clk), .write(wire_decoder_register
77
78      REG_X I (.clk(clk), .write_x(wire_decoder_registers[7]),.write_xref(wire_
79          .data_out_x(wire_registers_busMultiplexer[4]), .data_out_xref(i_ref_w
80
81      REG_X J (.clk(clk), .write_x(wire_decoder_registers[10]),.write_xref(wir
82          .xflag(jflag_wire), .data_out_x(wire_registers_busMultiplexer[5]), .d
83
84      REG_X K (.clk(clk), .write_x(wire_decoder_registers[13]),.write_xref(wir
85          .xflag(kflag_wire), .data_out_x(wire_registers_busMultiplexer[6]), .d
86
87      GENERAL_PURPOSE_REGISTER BASE_A (.clk(clk), .write(wire_decoder_register
88          .data_out(Base_A_wire));
89
90      GENERAL_PURPOSE_REGISTER BASE_B (.clk(clk), .write(wire_decoder_register
91          .data_out(Base_B_wire));
92
93      MAT_X MAT_A (.clk(clk), .X_Ref(j_ref_wire), .X(wire_registers_busMultipl
94          .Y(wire_registers_busMultiplexer[4]),.Mat_data_out(wire_registers_busM
95
96      MAT_X MAT_B (.clk(clk), .X_Ref(k_ref_wire), .X(wire_registers_busMultipl
97          .Y(wire_registers_busMultiplexer[5]),.Mat_data_out(wire_registers_busM
98
99      MAT_X MAT_C (.clk(clk), .X_Ref(k_ref_wire), .X(wire_registers_busMultipl
100         .Y(wire_registers_busMultiplexer[4]),.Mat_data_out(wire_registers_busM
101
102     GENERAL_PURPOSE_REGISTER P (.clk(clk), .write(wire_decoder_registers[15]
103         .data_out(wire_registers_busMultiplexer[7]));
104
105     GENERAL_PURPOSE_REGISTER R (.clk(clk), .write(wire_decoder_registers[16]
106         .data_out(wire_registers_busMultiplexer[8]));
107
108     MAIN_ALU ALU (.clk(clk), .data_bus_alu(bus_wire), .data_ac_alu(wire_regi
```

```
109         .op_code(opcode_wire),.out(data_alu_ac), .zflag(zflag_wire));
110 
111      AC AC (.clk(clk), .write(ac_cu_wire),
112         .data_alu_ac(data_alu_ac), .data_out(wire_registers_busMultiplexer[9]
113 
114      DECODER DECODER (
115         .clk(clk),
116         .from_cu(wire_cu_decoder),
117         .from_selector(wire_regSelector_decoder),
118         .out_AR(wire_decoder_registers[0]),
119         .out_PC(wire_decoder_registers[1]),
120         .out_MIDR(wire_decoder_registers[2]),
121         .out_MDDR_BUS(wire_decoder_registers[3]),
122         .out_MDDR_IM(wire_decoder_registers[4]),
123         .out_MDDR_DM(wire_decoder_registers[5]),
124         .out_BASE_C(wire_decoder_registers[6]),
125         .out_I(wire_decoder_registers[7]),
126         .out_I_Ref(wire_decoder_registers[8]),
127         .out_Base_A(wire_decoder_registers[9]),
128         .out_J(wire_decoder_registers[10]),
129         .out_J_Ref(wire_decoder_registers[11]),
130         .out_Base_B(wire_decoder_registers[12]),
131         .out_K(wire_decoder_registers[13]),
132         .out_K_Ref(wire_decoder_registers[14]),
133         .out_P(wire_decoder_registers[15]),
134         .out_R(wire_decoder_registers[16])
135     );
136     BUS_MULTIPLEXER BUS_MULTIPLEXER(
137         .clk(clk),
138         .from_selector(bus_selector_from_register_selector),
139         .from_cu(bus_selector_from_cu),
140         .AR(wire_registers_busMultiplexer[0]),
141         .PC(wire_registers_busMultiplexer[1]),
142         .R(wire_registers_busMultiplexer[8]),
143         .P(wire_registers_busMultiplexer[7]),
144         .Mat_A(wire_registers_busMultiplexer[10]),
145         .Mat_B(wire_registers_busMultiplexer[11]),
146         .MDDR(wire_registers_busMultiplexer[2]),
147         .AC(wire_registers_busMultiplexer[9]),
148         .I(wire_registers_busMultiplexer[4]),
149         .I_ref(i_ref_wire),
150         .J(wire_registers_busMultiplexer[5]),
151         .J_ref(j_ref_wire),
152         .K(wire_registers_busMultiplexer[6]),
153         .K_ref(k_ref_wire),
154         .Mat_C(wire_registers_busMultiplexer[3]),
155         .proId(wire_registers_busMultiplexer[12]),
156         .out(bus_wire)
157     );
158     CONTROL_UNIT CONTROL_UNIT(
159         .clk(clk),
160         .z(zflag_wire),
161         .i(iflag_wire),
162         .j(jflag_wire),
```

```
        .k(kflag_wire),
        .instruction(MIDR_wire),
        .MDDR(wire_registers_busMultiplexer[2]),
        .to_ALU(opcode_wire),
        .to_DM(to_DM),
        .to_IM(to_IM),
        .to_Decoder(wire_cu_decoder),
        .to_BUS(bus_selector_from_cu),
        .to_PC(PC_inc_wire),
        .to_AC(ac_cu_wire),
        .en_RegSelector(en_register_selector),
        .rw_RegSelector(rw_register_selector),
        .finish(finish_wire),
        .current_micro_instruction(current_micro_instruction_wire)
    );
    REGISTER_SELECTOR REGISTER_SELECTOR(
        .clk(clk),
        .en(en_register_selector),
        .rw(rw_register_selector),
        .fromMDDR(wire_registers_busMultiplexer[2]),
        .toDecoder(wire_regSelector_decoder),
        .toBus(bus_selector_from_register_selector)
    );

endmodule
```