

Phạm Việt Hùng 2113592

Bài tập: Tìm giá trị max, min trong vùng nhớ 1K

$$1KB = 2^{10} B \quad 1024 B \text{ nhỏ}$$

1. Khi start tích cực nan giá trị trong ô nhớ đầu tiên của vùng nhớ 2K vào đồng thời vào 2 biến Max, Min là nó lưu trữ

2 giá trị max min cần tìm. Khởi tạo biến C (counter) = 0

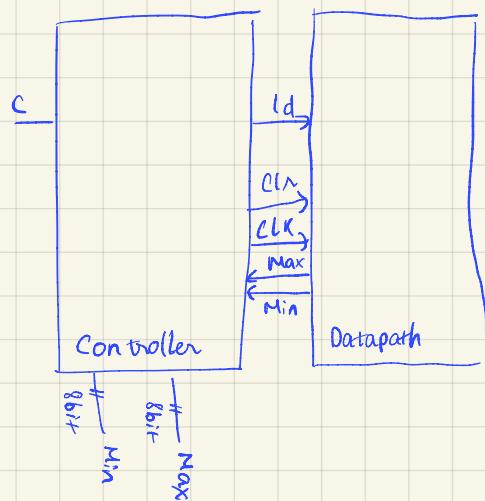
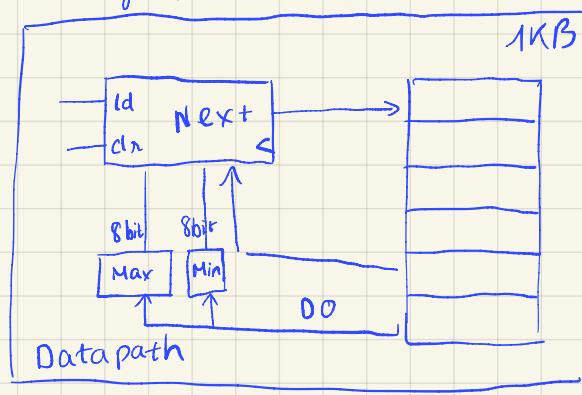
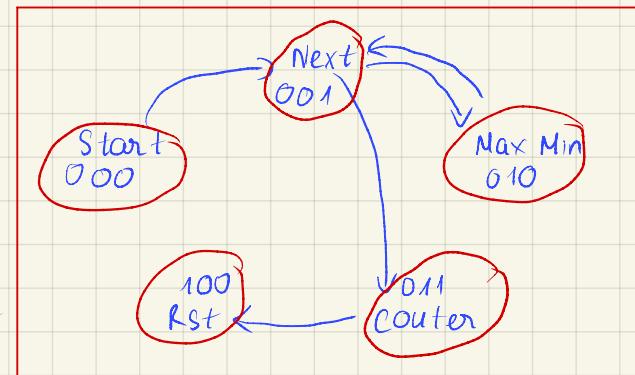
2. Trỏ vào ô nhớ tiếp theo ngay trên ô nhớ ban đầu load giá trị vào biến Next

3. So sánh giá trị trong biến next với 2 biến max, min nếu biến trong Next lớn hơn Max hoặc bé hơn min thì cập nhật

giá trị trong Next vào Max hoặc Min và tăng biến đếm C thêm 1

4. Sử dụng vòng lặp thực hiện lại bước 2 cho tới Khi biến C = 1024 thời điểm 3 bit biến diễn tả trạng thái thoát khỏi vòng lặp và tích cực biến Rst để kết thúc chương trình

$$m=5 \leq 2^3 \Rightarrow n=3$$

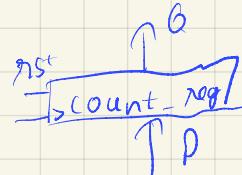


xem bài tập sau S2 K0 có ngõ ra

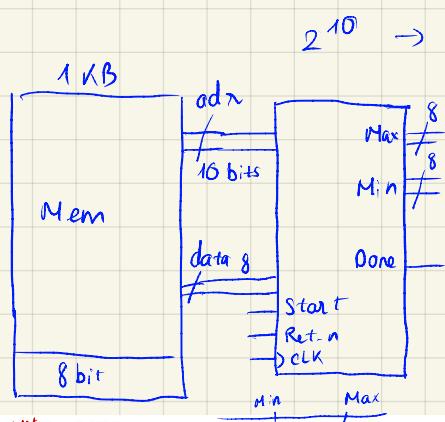
s

bên trong C là thành phần

$$\text{count} = \text{count} + 1$$



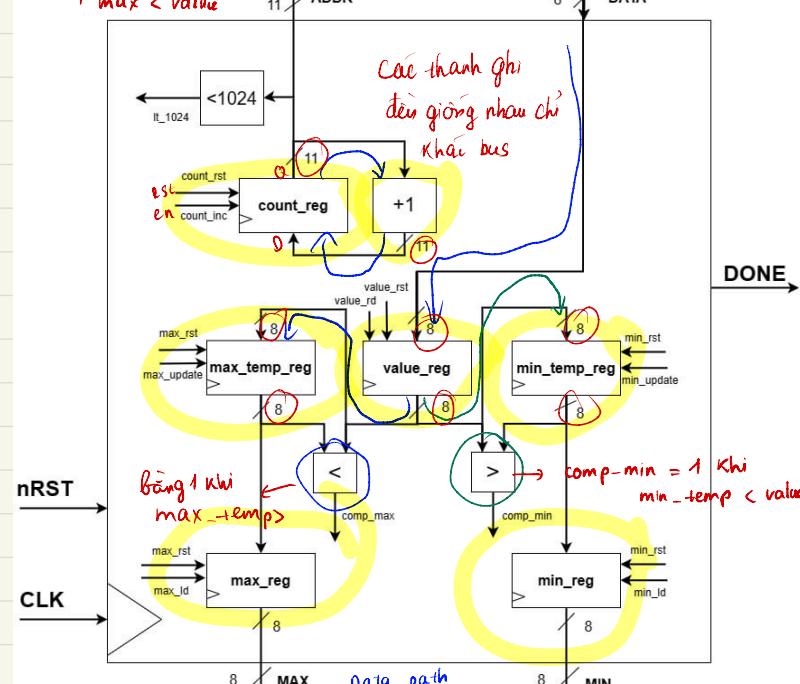
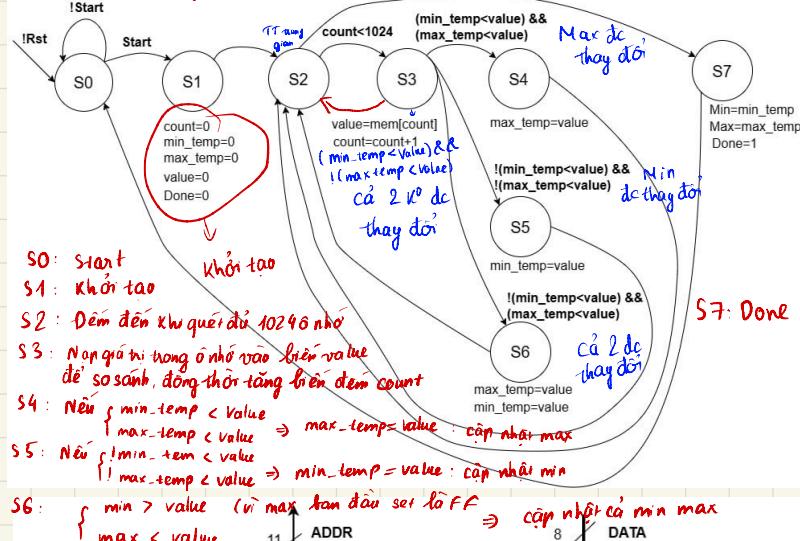
Bài sửa



$2^{10} \rightarrow 10$ đg địa chỉ từ $A0 \div A9 \rightarrow$ tay sai 2^{10} ô nhớ
1024 ô nhớ 8 bit

A9 A8 A7 A6 A5 A4 A3 A2 A1 A0

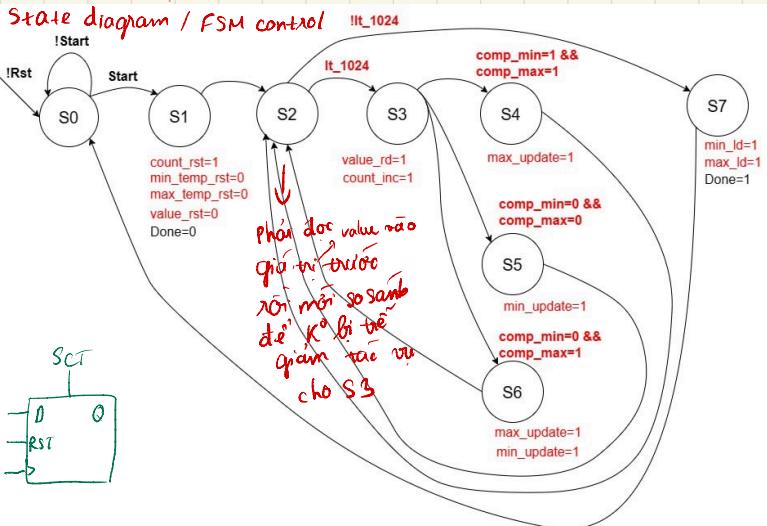
State diagram / output



```

1 module adder #(parameter WIDTH=10)
2   (A,S);
3   input  logic [WIDTH-1:0]A;
4   output logic [WIDTH-1:0]S;
5
6   always_comb
7   begin
8     S = A + 'd1;
9   end
10
11 endmodule
  
```

Khởi tạo Max = 00, Min = FF



Tín hiệu truyền vào sẽ thay đổi
còn mặc định là 8

```

module register #(parameter WIDTH=8)
  (rst, set, en, clk, D, Q);
  input logic rst;
  input logic set;
  input logic en;
  input logic clk;
  input logic [WIDTH-1:0]D;
  output logic [WIDTH-1:0]Q;

  always_ff@(posedge clk)
    begin
      if (rst)
        Q <= 0;
      else if (set)
        Q <= 'hFF;
      else if (en)
        Q <= D;
    end
endmodule
  
```

Verilog
→ System Verilog
→ logic
always
always_ff
always_comb

Files:
- register.sv
- Exercise1.sv
- adder.sv
- compare_A_It_B.sv
- FSM.sv
- Exercise1_tb.sv
- Exercise1.sdc

```

1 module compare_A_It_B #(parameter WIDTH=8)
2   (A,B,A_It_B);
3   input  logic [WIDTH-1:0]A;
4   input  logic [WIDTH-1:0]B;
5   output logic A_It_B;
6
7   always_comb
8   begin
9     if (A < B)
10       A_It_B = 1;
11     else
12       A_It_B = 0;
13   end
14
15 endmodule
  
```

```

Files
register.sv
Exercise1.sv
adder.sv
compare_A_lt_B.sv
FSM.sv
Exercise1_db.sv
Exercise1_sdc

Tasks Compilation
Task
④ Compile Design
④ Analysis & Synthesis
④ Fitter (Place & Route)
④ Assembler (Generate programming)
④ Timing Analysis
④ EDA Netlist Writer
Edit Settings
Program Device (Open Programmer)

1 module Exercise1 #(parameter ADDR_WIDTH=10,
                      parameter DATA_WIDTH=8)
  (rst_n,clk,start,ADDR,DATA,MAX,MIN,done);
  ...
endmodule

```

```

1 module Exercise1 #(parameter ADDR_WIDTH=10,
                      parameter DATA_WIDTH=8)
  (rst_n,clk,start,ADDR,DATA,MAX,MIN,done);
  ...
endmodule

```

đề code 1 bộ sequential chia ra 2 khối

- ④ Khối chuyển trạng thái: FF
- ④ Khối quyết định trạng thái kế tiếp: combinational

ngõ vào: state
ngõ ra : next_state

