題目：A Simple Processor

1. 實驗程式碼

```
module LAB6 (KEY, LEDR, HEX5, HEX4, HEX3, HEX2, HEX1, HEX0);
   input [2:0] KEY;
   output reg [7:0] LEDR;
   output [6:0] HEX5, HEX4, HEX3, HEX2, HEX1, HEX0;
   wire [4:0] addr;
   wire [7:0] instruction, R1, R0;

   rom1 ROM(
      .address(addr),
      .clock(~KEY[1]),
      .q(instruction));

   always @(posedge ~KEY[2]) begin
      LEDR = instruction;
   end

   Program_Counter PC(~KEY[1], KEY[0], addr);
   Processor Proc(instruction, KEY[0], ~KEY[2], R1, R0);
   two_digit_seg R0_display(R0, HEX3, HEX2);
   two_digit_seg R1_display(R1, HEX1, HEX0);
   two_digit_seg instruction_display(instruction, HEX5, HEX4);

endmodule
```

```
module Program_Counter (clk, Resetn, address);
   input clk, Resetn;
   output reg [4:0] address;

   always @(posedge clk or negedge Resetn) begin
      if(~Resetn) begin
         address = 0;
      end
      else begin
         if(address >= 32) begin
            address = 0;
         end
         else begin
            address = address + 1;
         end
      end
   end
endmodule
```

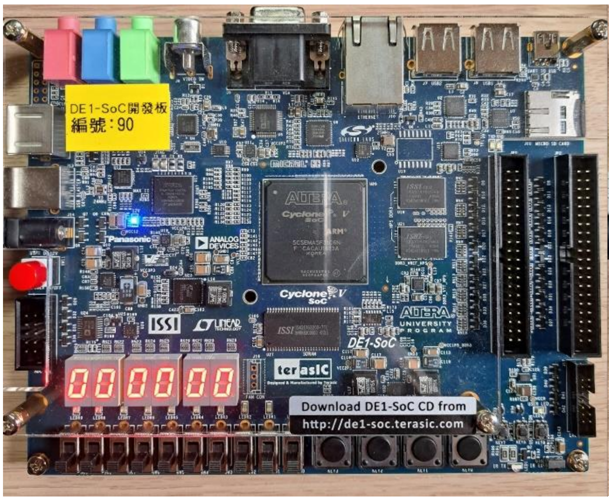
```
module Processor (DIN, Resetn, clk, R1, R0);
   input [7:0] DIN;
   input Resetn, clk;
   output [7:0] R1, R0;
   reg [7:0] register[0:7];
   reg flag = 0;
   reg [2:0] index;
   integer i;
   parameter mv = 2'b00, mvi = 2'b01, add = 2'b10, sub = 2'b11;

   assign R0 = register[0];
   assign R1 = register[1];

   always @(posedge clk or negedge Resetn) begin
      if(~Resetn) begin
         for(i = 0; i < 8; i = i + 1)
            register[i] <= 0;
      end
      else begin
         if(flag) begin
            register[index] = DIN;
            flag = 0;
         end
         else begin
            case (DIN[7:6])
               mv:
               begin
                  register[DIN[5:3]] = register[DIN[2:0]];
                  flag = 0;
               end
```

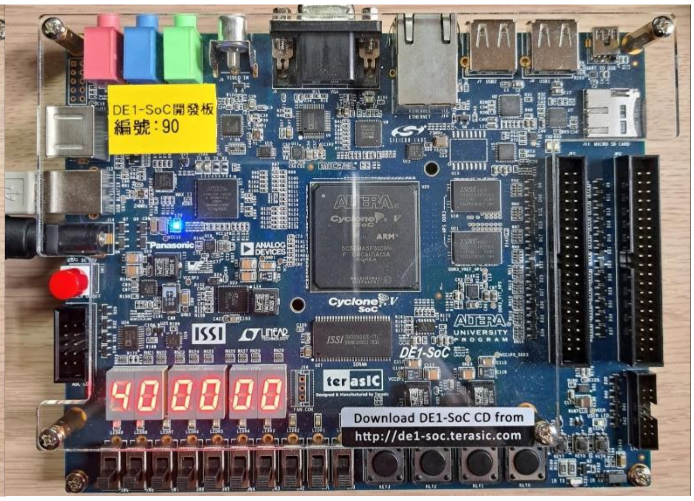
```
                mvi:
                begin
                    index = DIN[5:3];
                    flag = 1;
                end
                add:
                begin
                    register[DIN[5:3]] = register[DIN[5:3]] + register[DIN[2:0]];
                    flag = 0;
                end
                sub:
                begin
                    register[DIN[5:3]] = register[DIN[5:3]] - register[DIN[2:0]];
                    flag = 0;
                end
            endcase
        end
    end
  end
endmodule
```

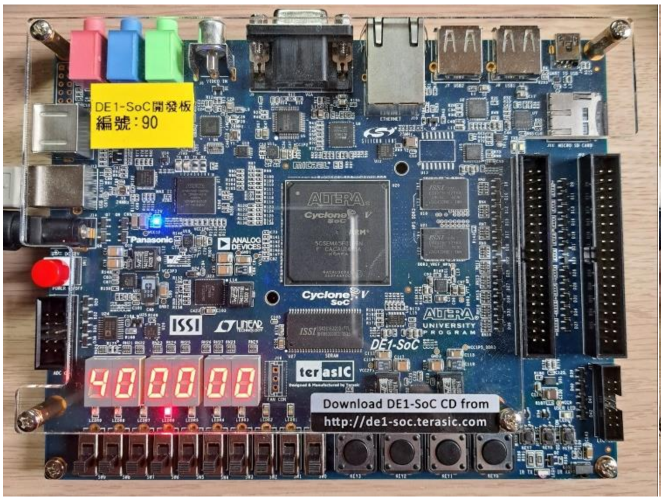
```
//bch to seg
module seg_decoder (bch, seg);
    input [3:0] bch;
    output reg [6:0] seg;

    always @(bch) begin
        case (bch)
            0 : seg = 7'b1000000;
            1 : seg = 7'b1111001;
            2 : seg = 7'b0100100;
            3 : seg = 7'b0110000;
            4 : seg = 7'b0011001;
            5 : seg = 7'b0010010;
            6 : seg = 7'b0000010;
            7 : seg = 7'b1111000;
            8 : seg = 7'b0000000;
            9 : seg = 7'b0010000;
            10 : seg = 7'b0001000;  //A
            11 : seg = 7'b0000011;  //b
            12 : seg = 7'b1000110;  //C
            13 : seg = 7'b0100001;  //d
            14 : seg = 7'b0000110;  //E
            15 : seg = 7'b0001110;  //F
            //lights out
            default : seg = 7'b1111111;
        endcase
    end
endmodule
```

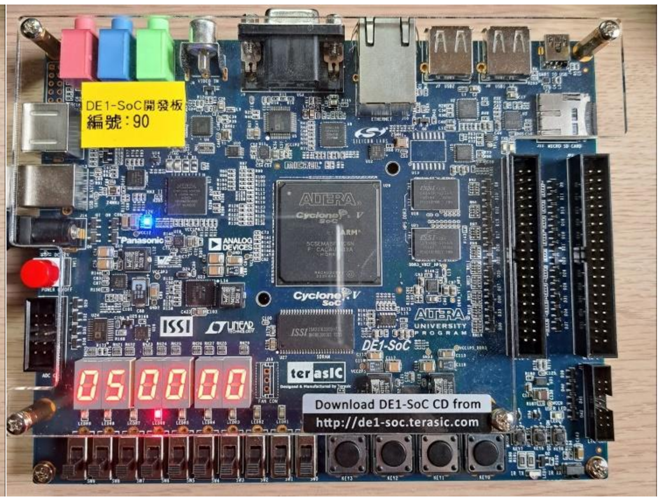
```
//display two digit in seg
module two_digit_seg (num, seg1, seg0);
   input  [7:0] num;
   output  [6:0] seg1, seg0;

   seg_decoder de1(num[7:4], seg1);
   seg_decoder de0(num[3:0], seg0);
endmodule
```

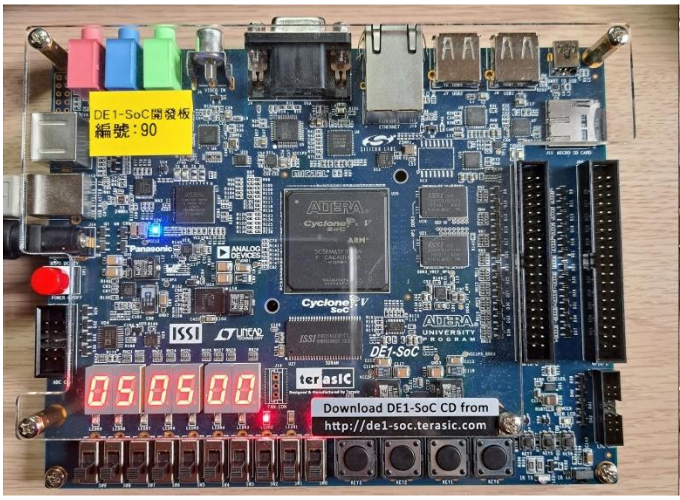
2. 實驗結果

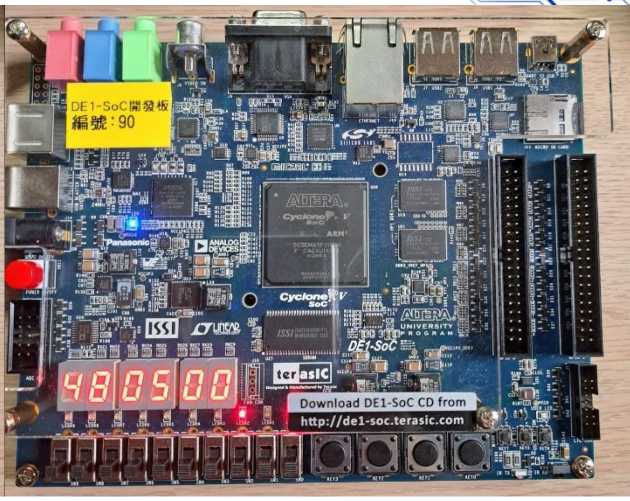


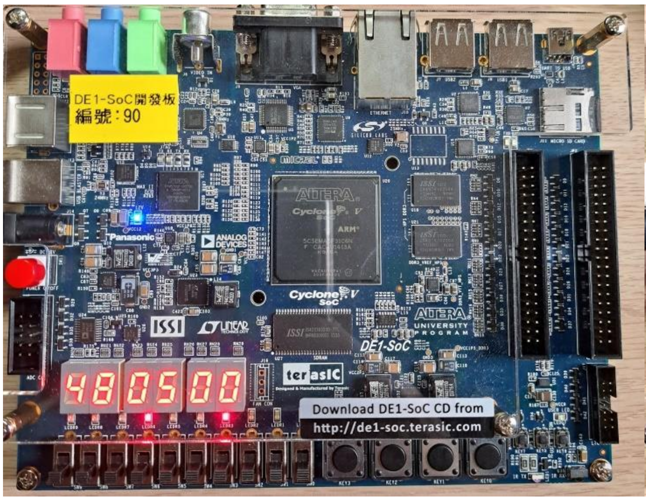


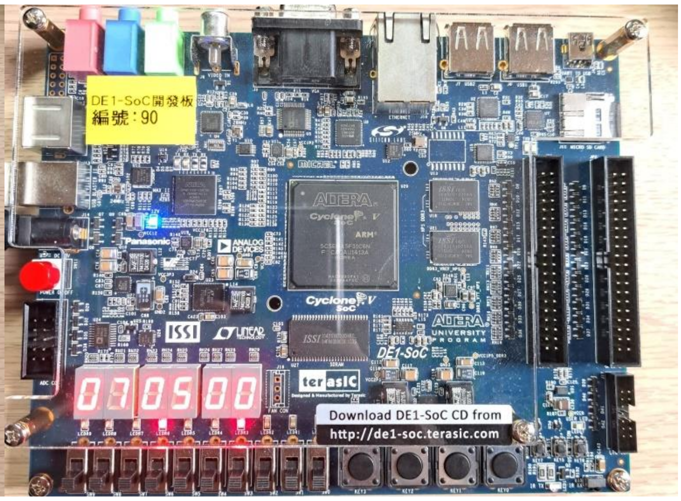


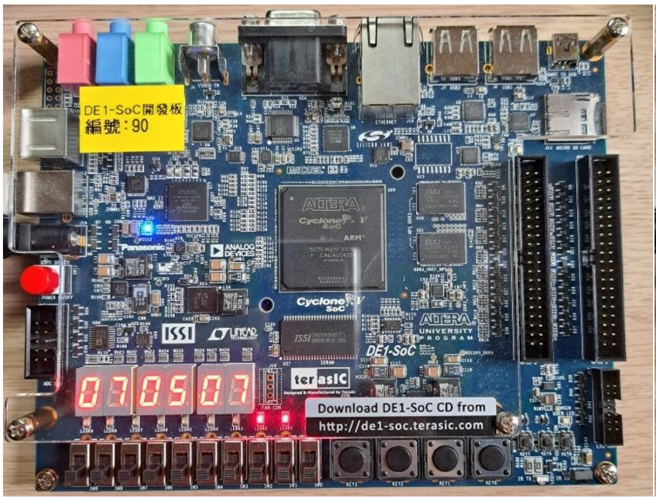


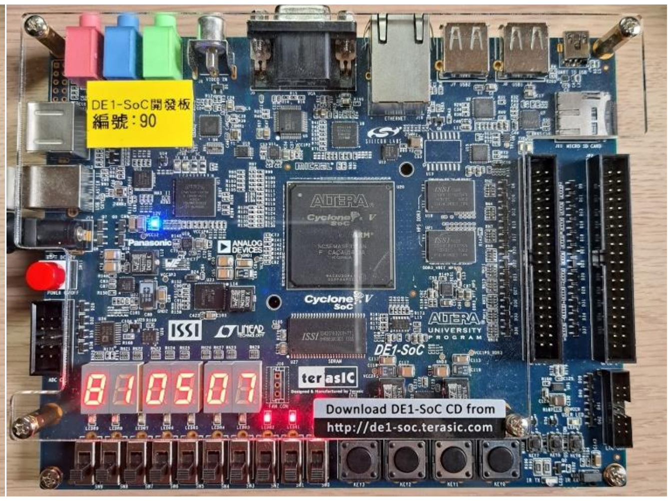

DE1-SoC開發板
編號：90
050500
Download DE1-SoC CD from
http://de1-soc.terasic.com

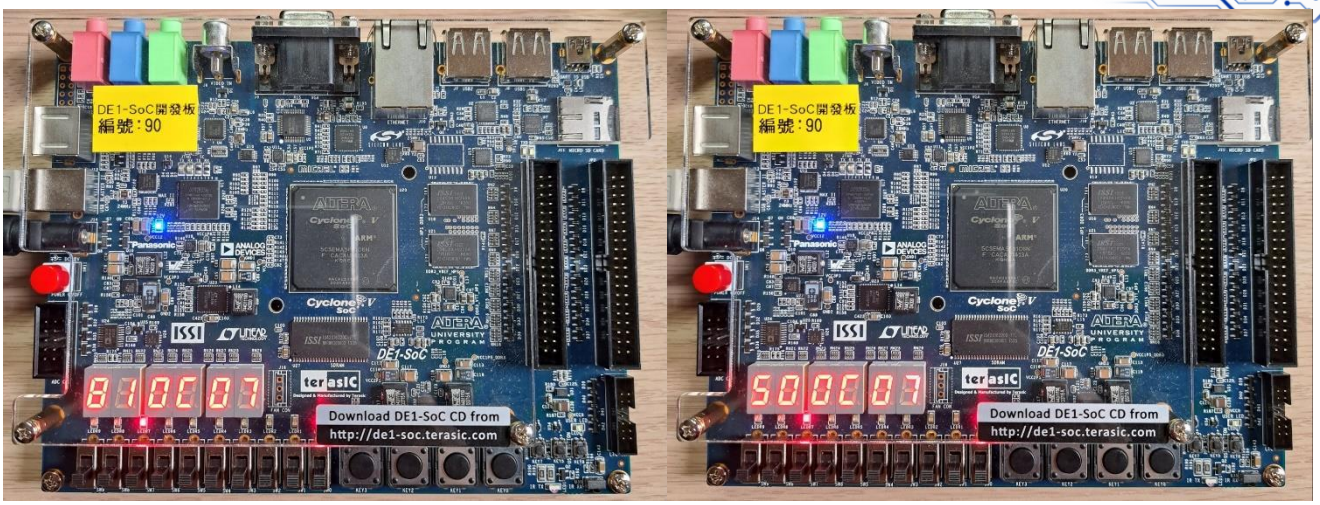

DE1-SoC開發板
編號：90
480500
Download DE1-SoC CD from
http://de1-soc.terasic.com

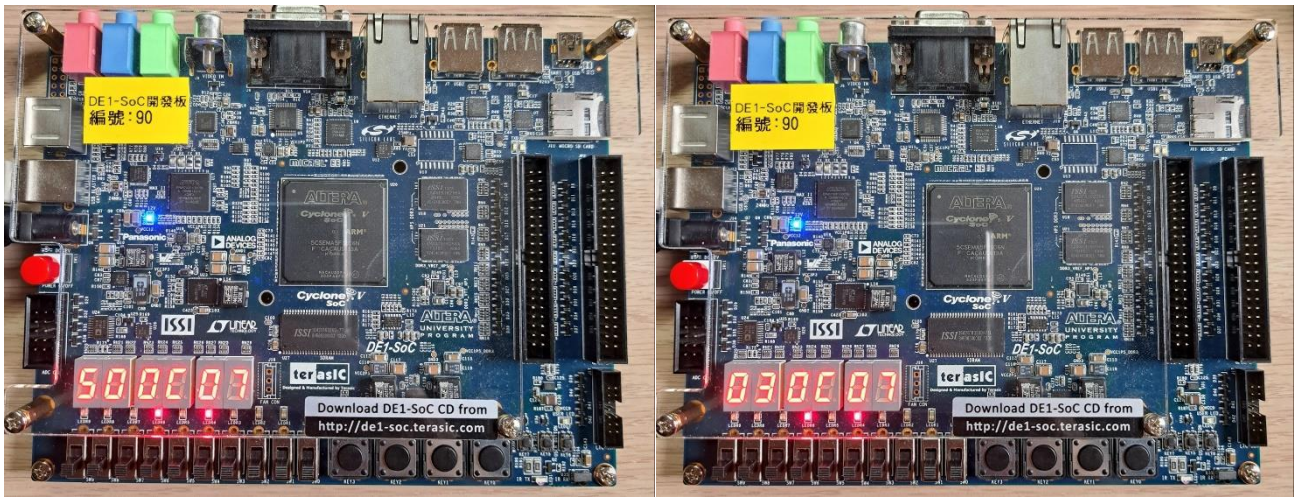

DE1-SoC開發板
編號：90
480500
Download DE1-SoC CD from
http://de1-soc.terasic.com

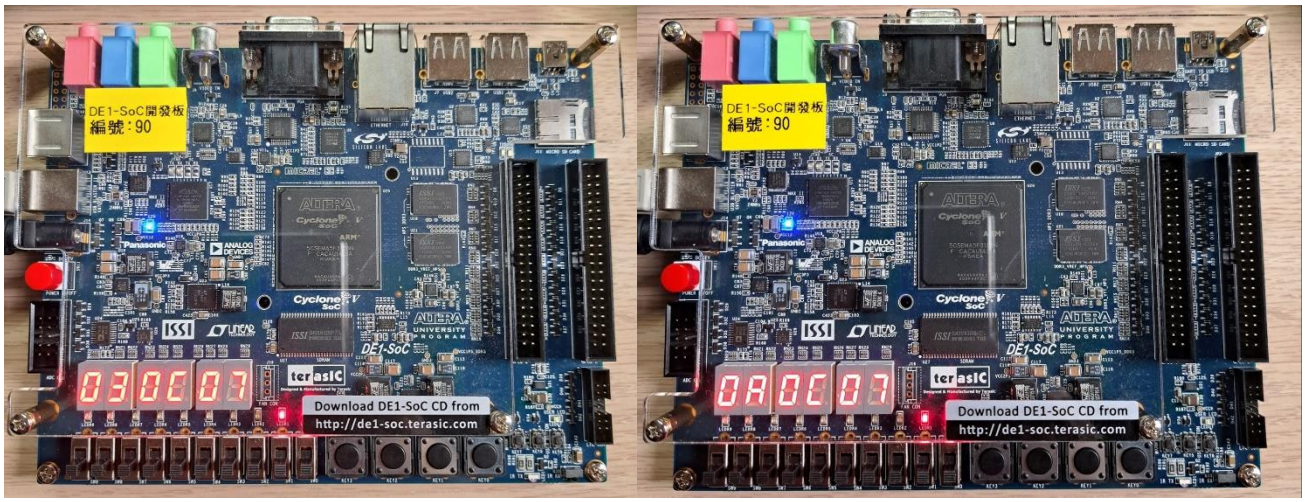

DE1-SoC開發板
編號：90
070500
Download DE1-SoC CD from
http://de1-soc.terasic.com

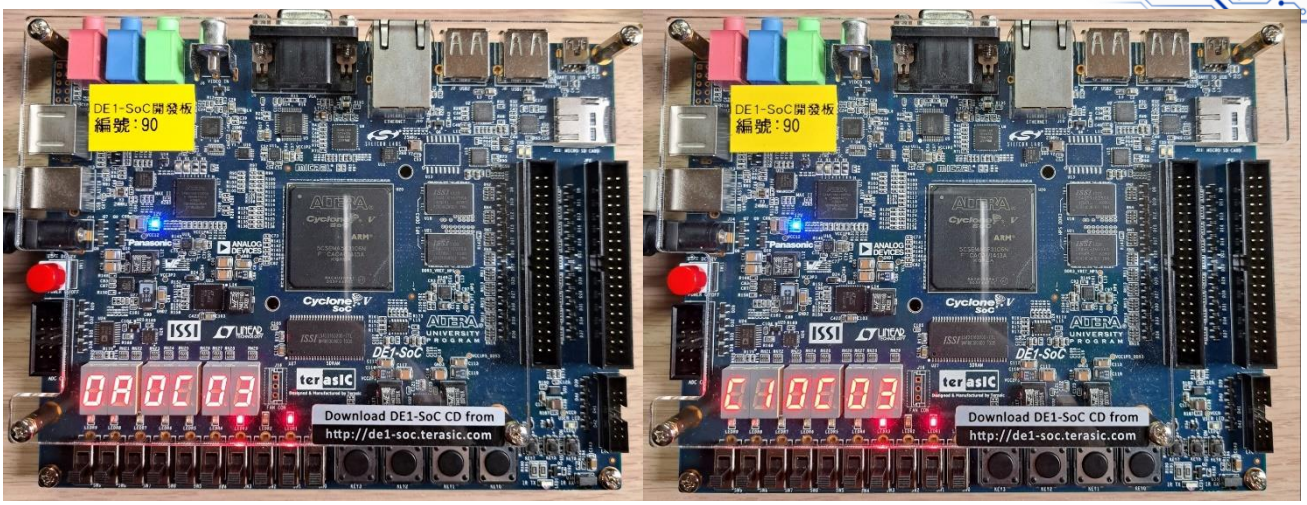

DE1-SoC開發板
編號：90
070507
Download DE1-SoC CD from
http://de1-soc.terasic.com

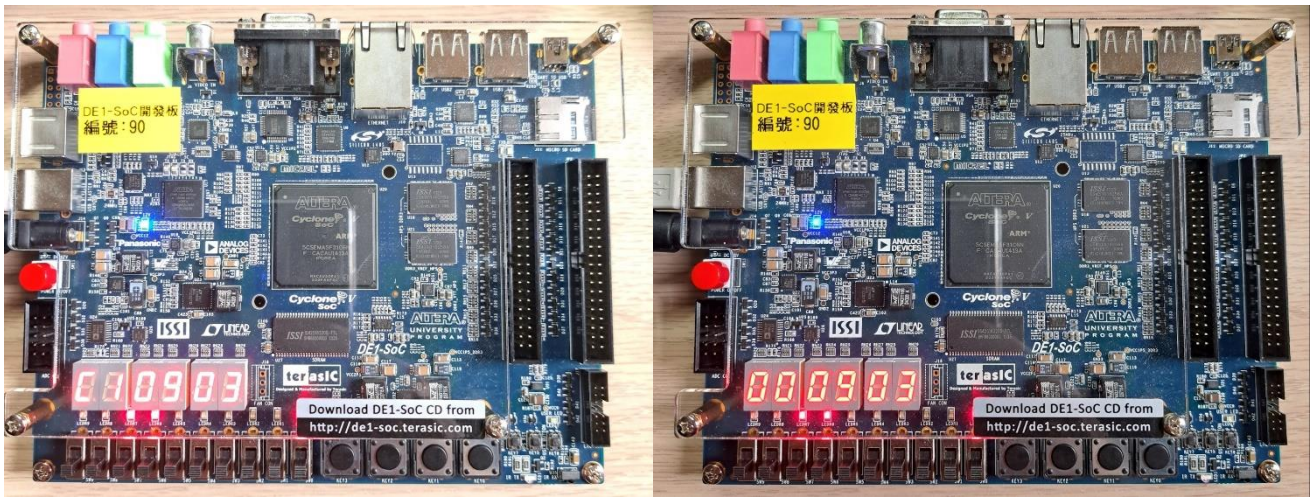

DE1-SoC開發板
編號：90
810507
Download DE1-SoC CD from
http://de1-soc.terasic.com

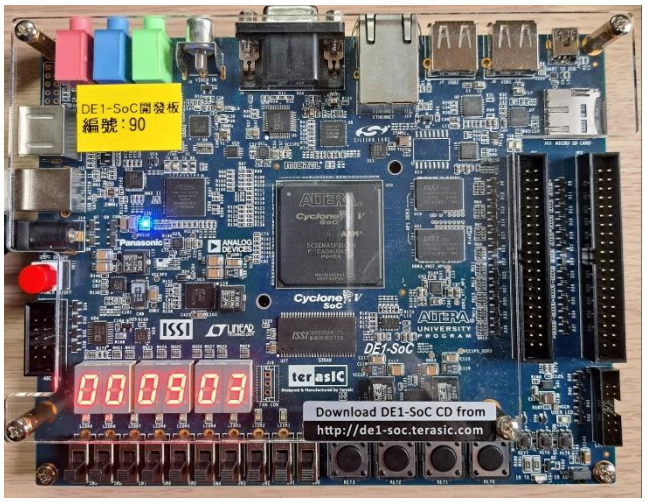

DE1-SoC開發板
編號：90
Download DE1-SoC CD from
http://de1-soc.terasic.com


DE1-SoC開發板
編號：90
Download DE1-SoC CD from
http://de1-soc.terasic.com

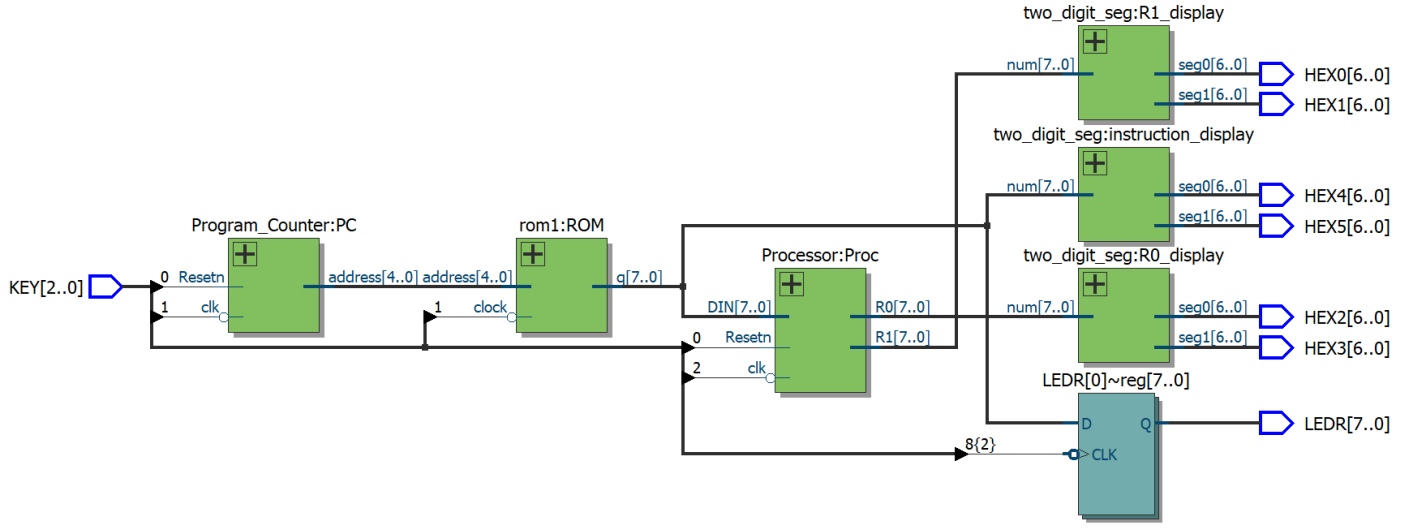

DE1-SoC開發板
編號：90
Download DE1-SoC CD from
http://de1-soc.terasic.com

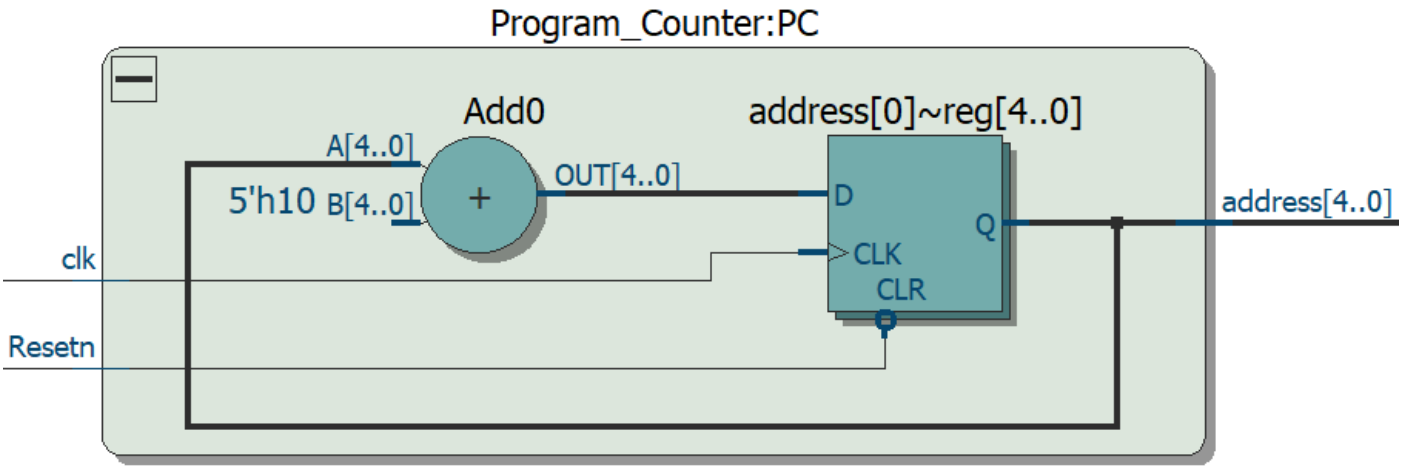

DE1-SoC開發板
編號：90
Download DE1-SoC CD from
http://de1-soc.terasic.com


DE1-SoC開發板
編號：90
Download DE1-SoC CD from
http://de1-soc.terasic.com

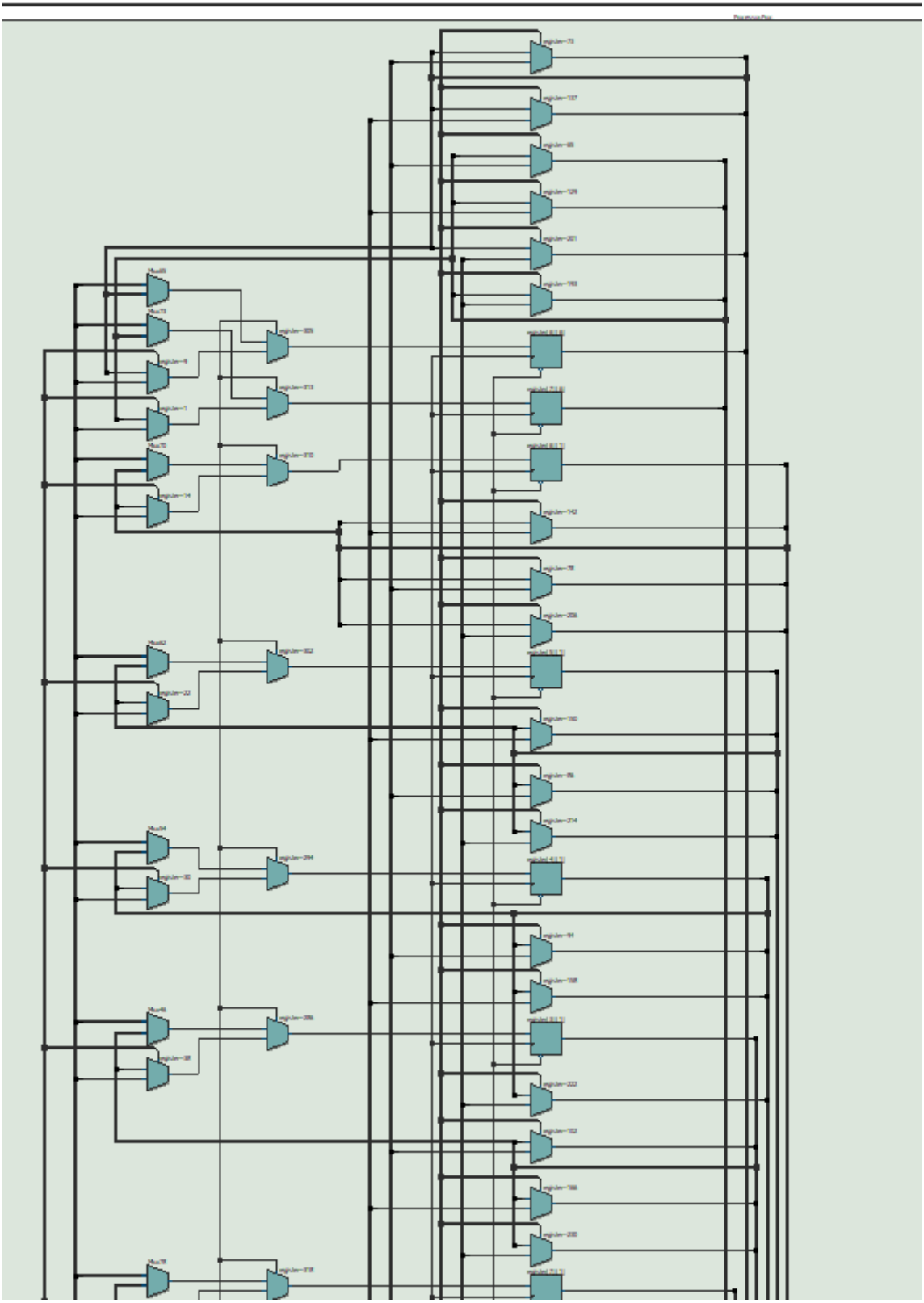

DE1-SoC開發板
編號：90
Download DE1-SoC CD from
http://de1-soc.terasic.com

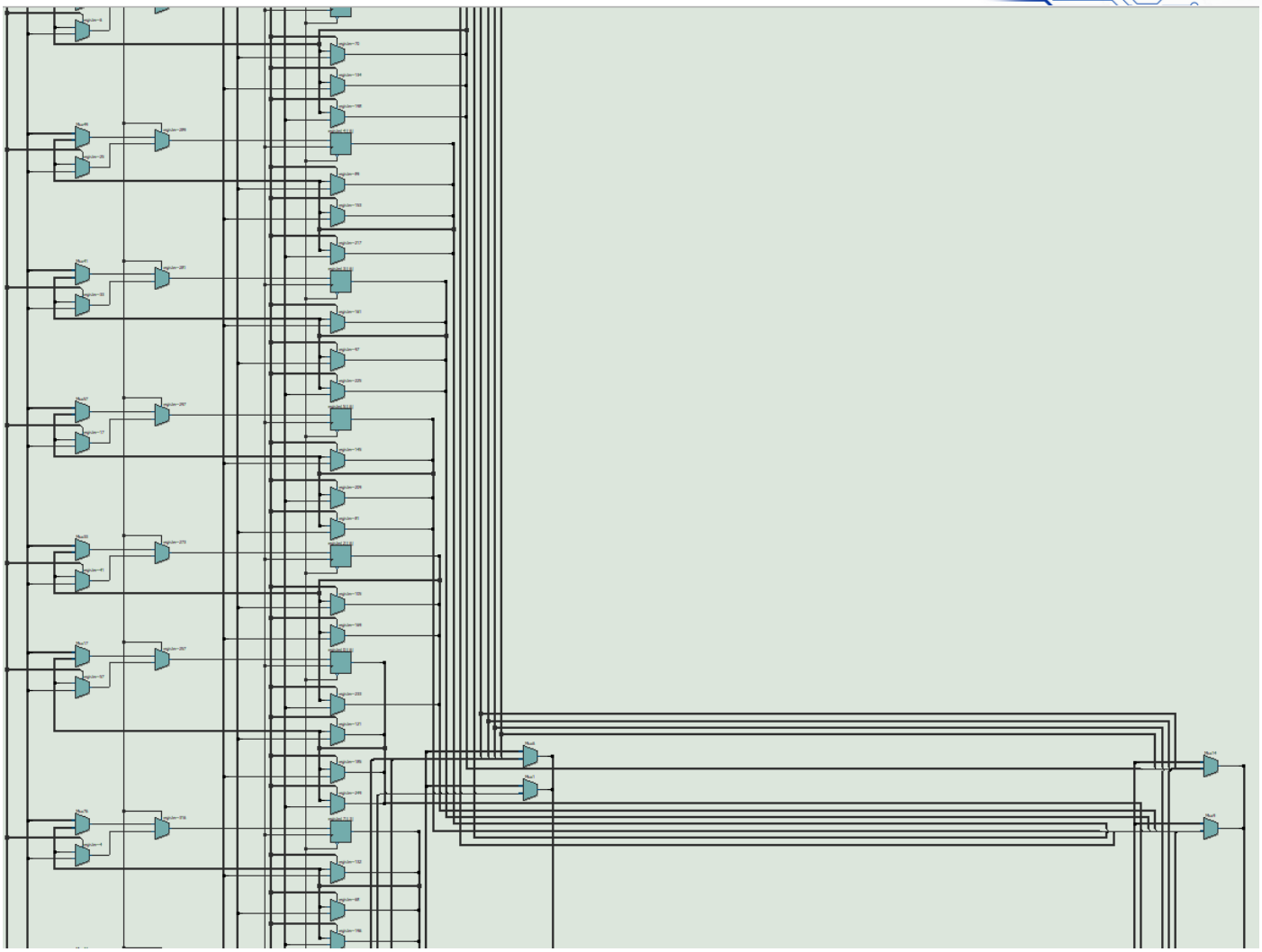
3. **RTL 佈局**

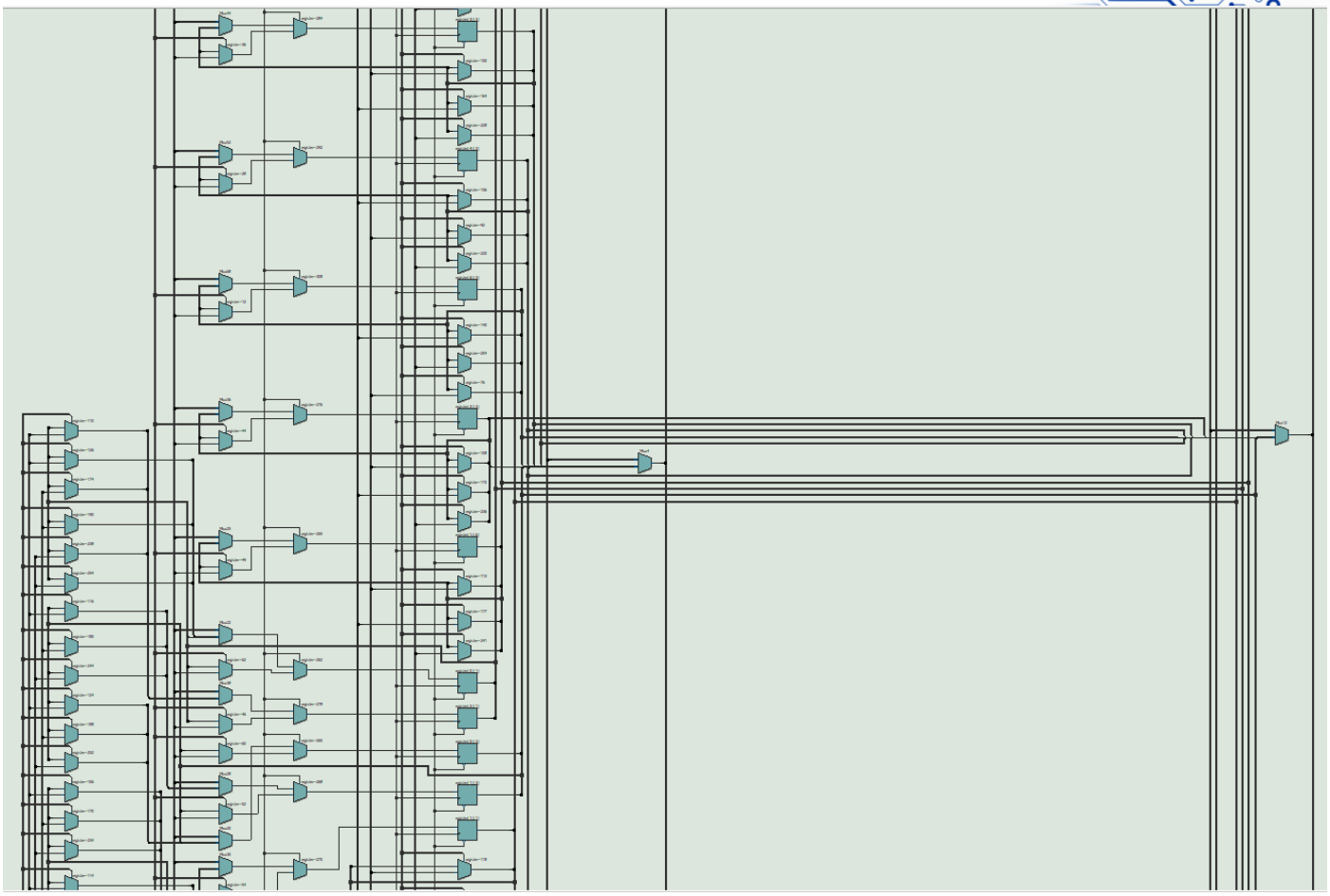

two_digit_seg:R1_display
num[7..0]
seg0[6..0]
seg1[6..0]
HEX0[6..0]
HEX1[6..0]
two_digit_seg:instruction_display
num[7..0]
seg0[6..0]
seg1[6..0]
HEX4[6..0]
HEX5[6..0]
two_digit_seg:R0_display
num[7..0]
seg0[6..0]
seg1[6..0]
HEX2[6..0]
HEX3[6..0]
Program_Counter:PC
KEY[2..0]
0 Resetn
1 clk
address[4..0]
address[4..0]
rom1:ROM
1 clock
q[7..0]
Processor:Proc
DIN[7..0]
R0[7..0]
R1[7..0]
0 Resetn
2 clk
LEDR[0]~reg[7..0]
D
Q
CLK
LEDR[7..0]
8{2}

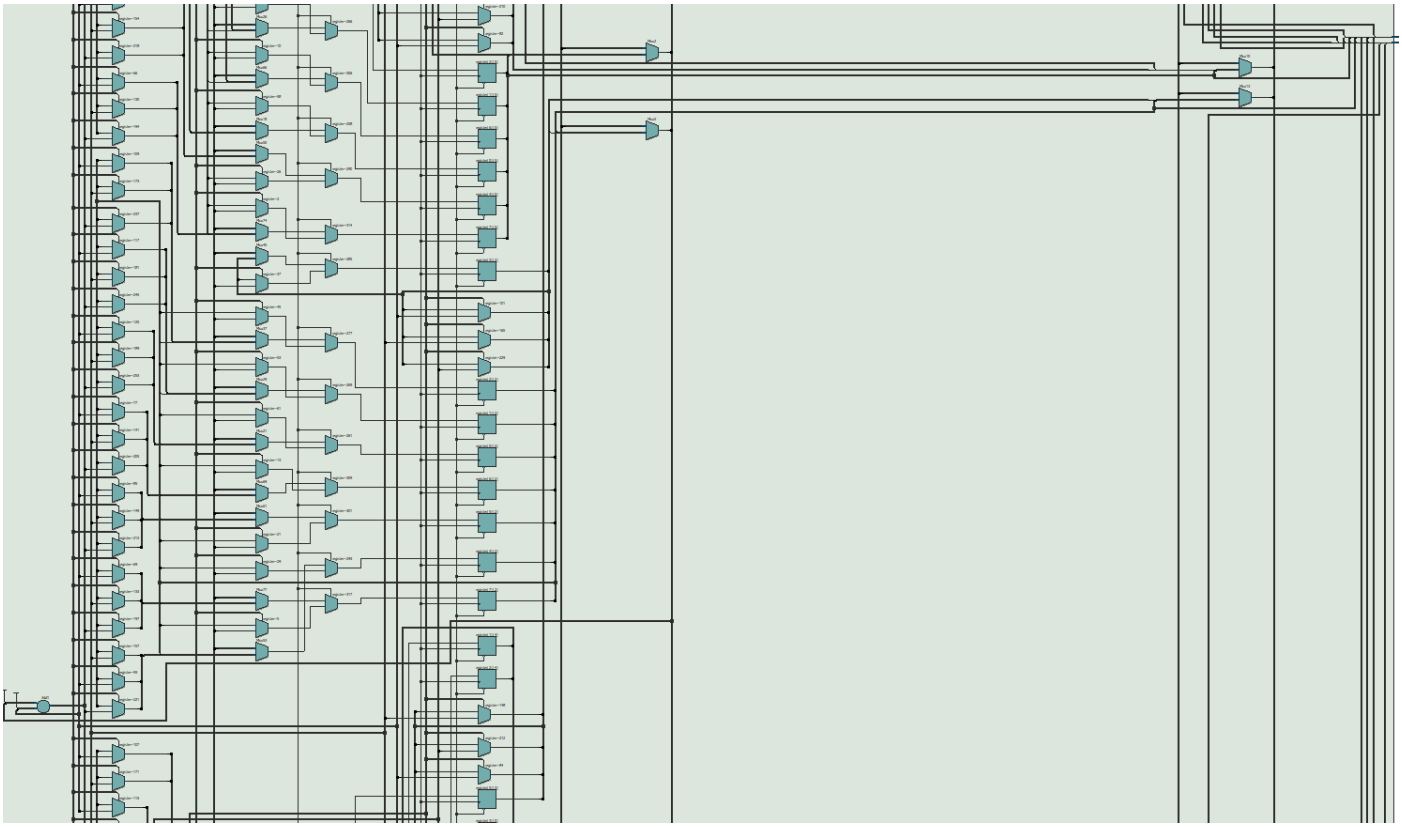

Program_Counter:PC
Add0
A[4..0]
5'h10 B[4..0]
+
OUT[4..0]
address[0]~reg[4..0]
D
Q
CLK
CLR
address[4..0]
clk
Resetn

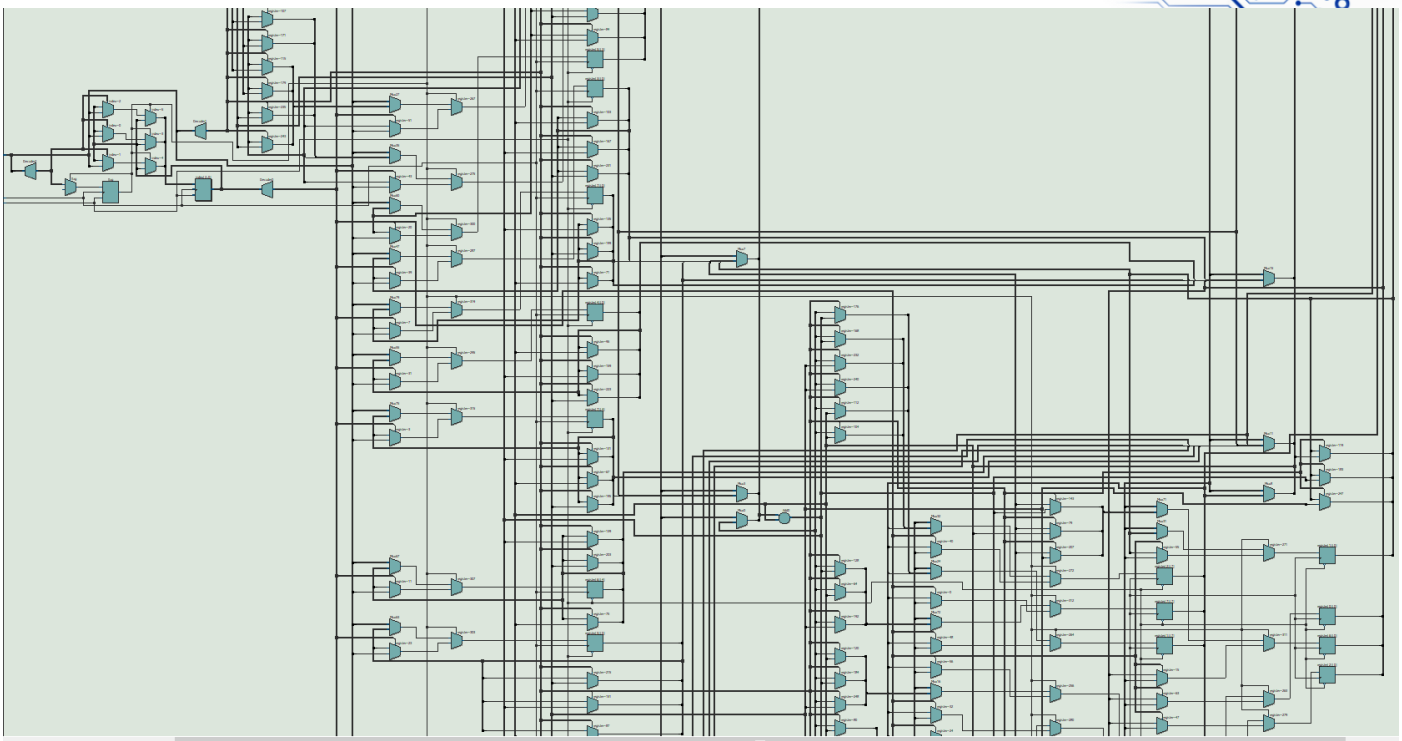
Processor

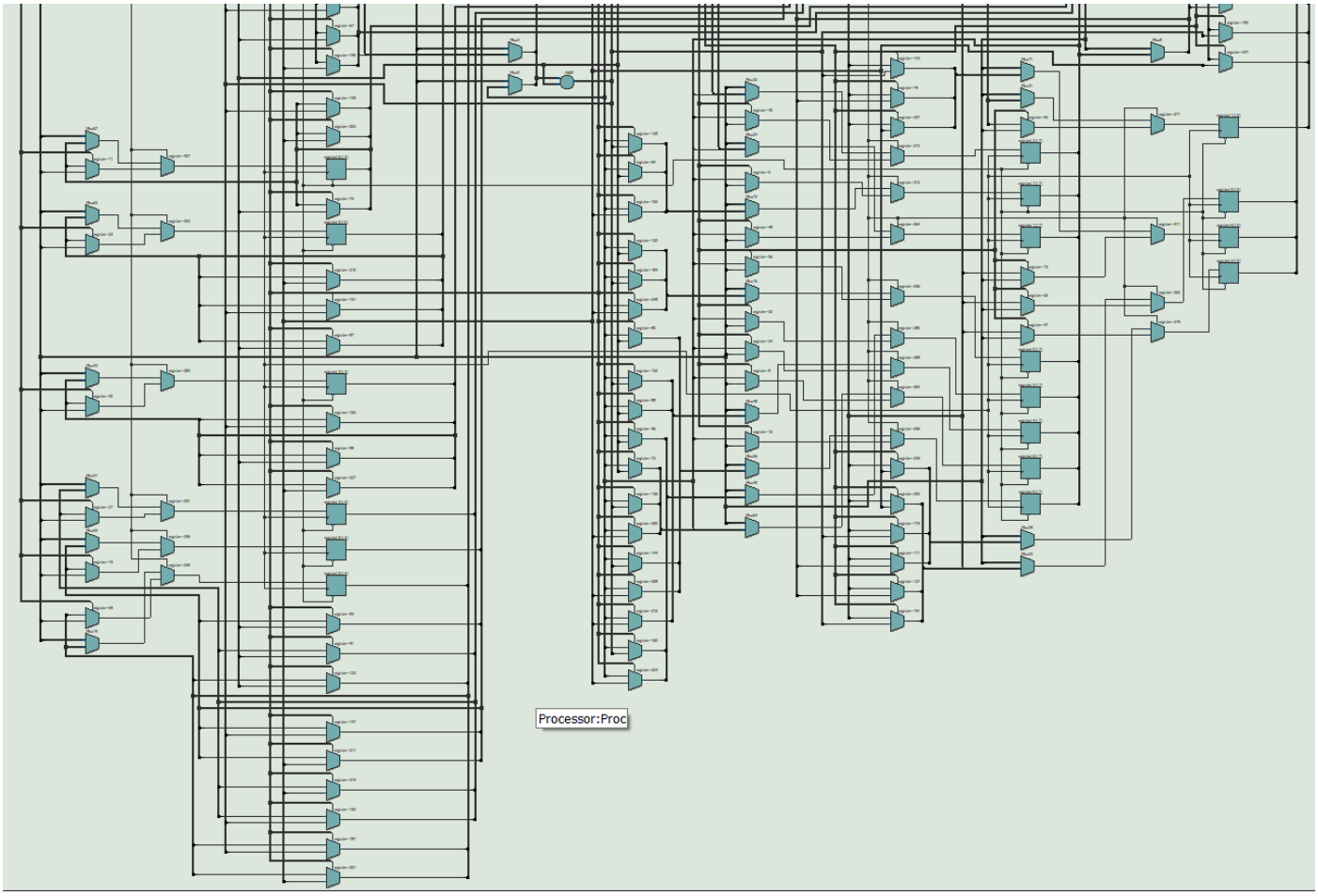












4. **問題與討論**

- 由於 LEDR 是配合 PClock，所以我將 LEDR 宣告為 reg，在 PClock posedge 才觸發，而不是直接連接 DIN。

- Processor 內部有八個暫存器儲存指令計算的結果，其中把 0 號和 1 號暫存器拉出去作為輸出，flag 為判斷是否為 mvi 指令。