94 年度大學院校 積體電路設計競賽

標準單元式(Cell-Based)競賽初賽參考解答

研究所/大學組

一、 設計結果報告(report.000)

二、 暫存器轉換階層(RTL level)設計結果

三、 模擬結果

設計結果報告(report.000)

隊號(Team number): ***CIC***

使用之 HDL 名稱(Verilog or VHDL): **Verilog**
----------RTL Simulation Level----------
RTL 檔案名稱(RTL Netlist file name): **triangle.v**

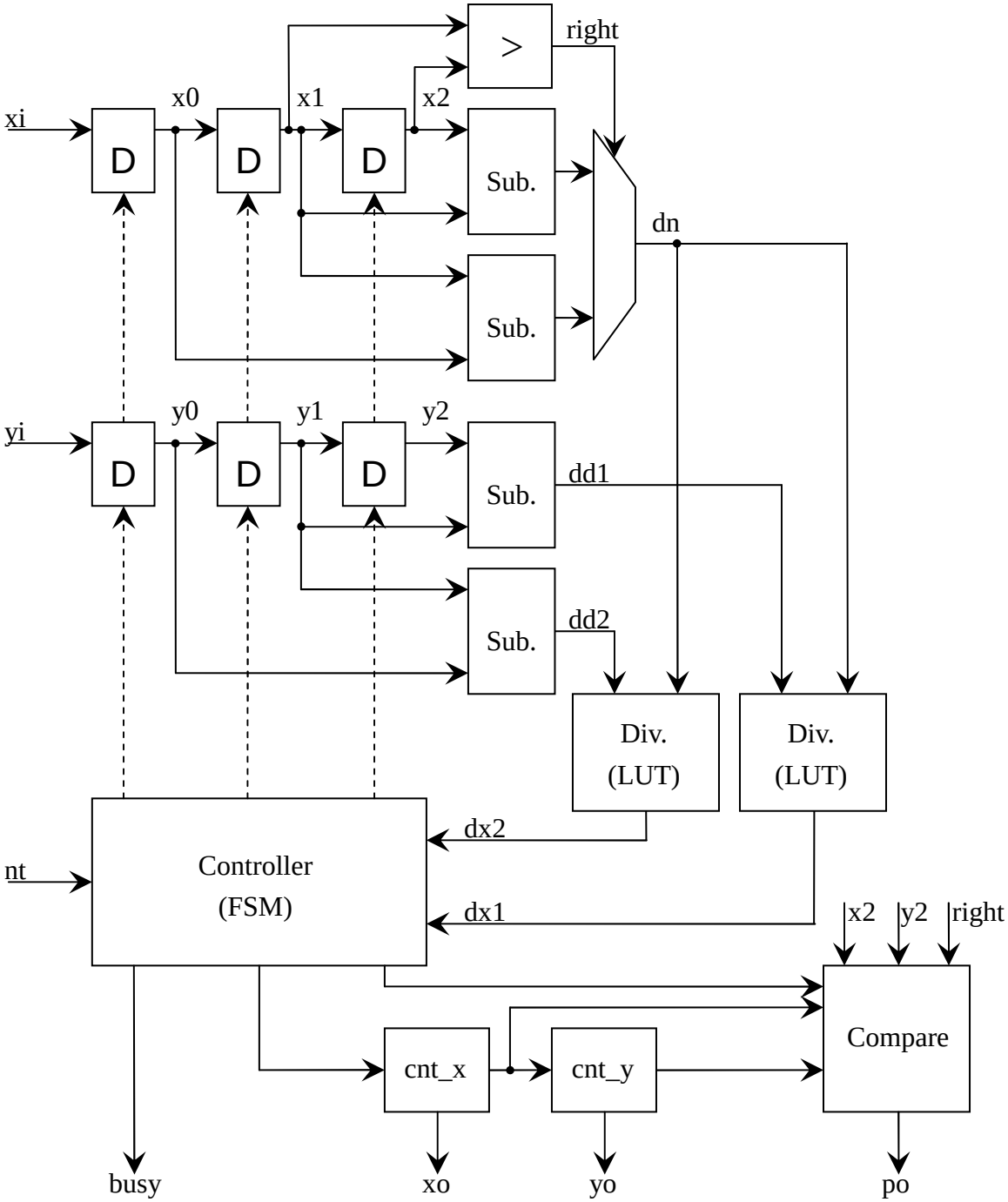
RTL simulation, 所使用最小的 CYCLE = (**5**) ns

----------Synthesis Level----------
Gate-Level 檔案名稱(Gate-Level Netlist file name):**triangle_syn.vg**
Gate-Level simulation SDF 檔案名稱(sdf file name):**triangle_syn.sdf**
Design Compiler Datebase 檔案名稱(.db):**triangle_syn.db**

GATE LEVEL simulation, 所使用最小的 CYCLE = (**4.3**) ns

其他說明事項(Any other information you want to specify:(如設計特點 ...)

1、 此電路的設計方法是使用兩個 counter，分別用來計數 x 軸與 y 軸座標。當 x 軸的計數器(cnt_x)與 y 軸的計數器(cnt_y)落在三角形之中，便將 po 輸出為 high，表示(x, y)座標是正確值。此電路是先計算出 x 座標的 boundary(x_bound)，當 cnt_x 計數的值落在 x_bound 之中，便將 po 升為 high。
2、 整體架構如圖一所示。首先，shift register 透過 controller 的控制可將三角形的座標值適時地存在暫存器之中，直到運算結束。
3、 其動作說明如下：透過第二個座標 x 值，可得知三角形的頂點是否在右邊(right=1)，再經由 right 來選擇適當的被除數(正值或負值)，使 controller 來計算出 x_bound。在計算 x_bound 的斜率時，還需要 y 值，因此就透過兩個減法器來找出 y 值，經由兩個除法器就可以計算出斜率來送給 controller。隨著 y 座標的計數，po 會在 x 座標符合條件時，輸出為 high。
4、 由於此題目所要求的運算範圍很小，所以在計算斜率的除法器上是使用查表(LUT)的方式來實現，這樣可以大大提升運算速度。


xi
x0
x1
x2
D
D
D
>
right
Sub.
Sub.
dn
yi
y0
y1
y2
D
D
D
Sub.
Sub.
dd1
dd2
Div.
(LUT)
Div.
(LUT)
nt
Controller
(FSM)
dx2
dx1
x2
y2
right
Compare
cnt_x
cnt_y
busy
xo
yo
po

圖一　架構圖

二、暫存器轉換階層(RTL level)設計結果 (Verilog HDL)

```
module triangle (clk, reset, nt, xi, yi, busy, po, xo, yo);
  input clk, reset, nt;
  input [2:0] xi, yi;
  output busy, po;
  output [2:0] xo, yo;

  reg       busy;
  reg  [1:0] state;
  reg  [2:0] cnt_y, cnt_x;
  reg  [5:0] x_tmp;
  wire [2:0] x0, x1, x2;
  wire [2:0] y0, y1, y2;
  wire [2:0] dn, dd1, dd2;
  wire [2:0] x_bound;
  wire [5:0] dx1, dx2;
  wire       right;

  div_lut div1(dn, dd1, dx1);
  div_lut div2(dn, dd2, dx2);
  fr x0_reg(clk, reset, &(state), xi, x0);
  fr x1_reg(clk, reset, &(state), x0, x1);
  fr x2_reg(clk, reset, &(state), x1, x2);
  fr y0_reg(clk, reset, &(state), yi, y0);
  fr y1_reg(clk, reset, &(state), y0, y1);
  fr y2_reg(clk, reset, &(state), y1, y2);

  assign right = (x1 > x2) ? 1'b1 : 1'b0;
  assign xo = cnt_x;
  assign yo = cnt_y;
  assign x_bound = ((!right && |(x_tmp[2:0]))) ? x_tmp[5:3] + 1'b1 : x_tmp[5:3] ;
  assign dn = right ? x1 - x2 : x2 - x1;
  assign dd1 = y1 - y2;
  assign dd2 = y0 - y1;
  assign po = (&(state) && cnt_y >= y2 && cnt_y <= y0 && right && cnt_x >= x2 &&
              cnt_x <= x_bound) ? 1'b1 : (&(state) && cnt_y >= y2 && cnt_y <= y0
              && !right && cnt_x >= x_bound && cnt_x <= x2) ? 1'b1 : 1'b0;

  always @(posedge clk)
  begin : FSM
    if (reset) begin
      state <= 2'd0;
      busy <= 1'b0;
    end else begin
      case (state)
        0: begin
          if (nt)
            state <= 2'd1;
        end
        1: begin
          state <= 2'd2;
          busy <= 1'b1;
        end
        2:
          state <= 2'd3;
        3: begin
          if (&{cnt_y, cnt_x}) begin
            state <= 2'd0;
            busy <= 1'b0;
```

```
            end
          end
          default:
            state <= 0;
        endcase
      end
    end

    always @(posedge clk)
    begin : cal_xbond
      if (reset)
        x_tmp <= 6'd0;
      else if (state == 2'b10)
        x_tmp <= {xi, 3'b000};
      else if (&(cnt_x))
        if (cnt_y < y1 && cnt_y >= y2)
          if (right)
            x_tmp <= x_tmp + dx1;
          else
            x_tmp <= x_tmp - dx1;
        else if (cnt_y >= y1)
          if (right)
            x_tmp <= x_tmp - dx2;
          else
            x_tmp <= x_tmp + dx2;
    end

    always @(posedge clk)
    begin : counter_y
      if (reset)
        cnt_y <= 3'd0;
      else if (&(cnt_x))
        cnt_y <= cnt_y + 1;
    end

    always @(posedge clk)
    begin : counter_x
      if (reset)
        cnt_x <= 3'd0;
      else if (&(state))
        cnt_x <= cnt_x + 1;
    end

endmodule

module fr(clk,rst,stall,in,out); //feedback register
input       clk,rst,stall;
input  [2:0] in;
output [2:0] out;

wire [2:0] mux;
reg  [2:0] out;

assign mux = (stall) ? out : in;

always @(posedge clk)
begin : fb_reg
  if (rst)
    out <= 3'd0;
  else
    out <= mux;
```

```
end

endmodule

module div_lut(dn, dd, div_out);  //divider
   input [2:0]  dn;  //dividend, range: 1~6
   input [2:0]  dd;  //dividor, range: 1~6
   output [5:0] div_out;

   reg [5:0] div_out;

   always @(dn or dd) begin
      case (dd)
         3'd1: begin
            div_out = {dn, 3'b000};
         end
         3'd2: begin
            div_out = {1'b0, dn, 2'b00};
         end
         3'd3: begin
            case (dn)
               3'd1: div_out = 6'b000_010;
               3'd2: div_out = 6'b000_101;
               3'd3: div_out = 6'b001_000;
               3'd4: div_out = 6'b001_010;
               3'd5: div_out = 6'b001_101;
               3'd6: div_out = 6'b010_000;
            endcase
         end
         3'd4: begin
            div_out = {2'b00, dn, 1'b0};
         end
         3'd5: begin
            case (dn)
               3'd1: div_out = 6'b000_001;
               3'd2: div_out = 6'b000_011;
               3'd3: div_out = 6'b000_100;
               3'd4: div_out = 6'b000_110;
               3'd5: div_out = 6'b001_000;
               3'd6: div_out = 6'b001_001;
            endcase
         end
         3'd6: begin
            case (dn)
               3'd1: div_out = 6'b000_001;
               3'd2: div_out = 6'b000_010;
               3'd3: div_out = 6'b000_100;
               3'd4: div_out = 6'b000_101;
               3'd5: div_out = 6'b000_110;
               3'd6: div_out = 6'b001_000;
            endcase
         end
      endcase
   end

endmodule
```

三、 模擬結果 (Gate-level)

(a) 模擬之 log 檔

```
ncverilog: 05.50-p004: (c) Copyright 1995-2005 Cadence Design Systems, Inc.
TOOL:     ncverilog 05.50-p004: Started on Jun 15, 2006 at 16:18:24 CST
ncverilog
     +maxdelays
     testfixture.v
     triangle_syn.v
     -v
     ../tsmc18_neg.v
     +access+r
Recompiling... reason: file './testfixture.v' is newer than expected.
     expected: Sat Jul 15 16:17:56 2006
     actual:   Sat Jul 15 16:18:21 2006
file: testfixture.v
     module worklib.test:v
          errors: 0, warnings: 0
file: triangle_syn.v
file: ../tsmc18_neg.v
   ...
Loading snapshot worklib.test:v ..................... Done
ncsim> source /usr/cad/cadence/IUS/cur/tools/inca/files/ncsimrc
ncsim> run

****** START to VERIFY the Triangle Rendering Engine OPERATION ******

Waiting for the rendering operation of the triangle points with:
(x1, y1)=(1, 0)
(x2, y2)=(7, 2)
(x3, y3)=(1, 7)
Waiting for the rendering operation of the triangle points with:
(x1, y1)=(6, 1)
(x2, y2)=(0, 3)
(x3, y3)=(6, 6)
Waiting for the rendering operation of the triangle points with:
(x1, y1)=(0, 1)
(x2, y2)=(4, 5)
(x3, y3)=(0, 6)
Waiting for the rendering operation of the triangle points with:
(x1, y1)=(5, 2)
(x2, y2)=(2, 5)
(x3, y3)=(5, 6)
PASS! All data have been generated successfully!
-------------------------------------------
Total delay:      11180 ns
-------------------------------------------
Simulation complete via $finish(1) at time 1126600 PS + 0
./testfixture.v:144      $finish;
```

```
ncsim> exit
TOOL:     ncverilog 05.50-p004: Exiting on Jun 15, 2006 at 16:18:28 CST  (total: 00:00:04)
```

(b) 波形圖(部分)