邏輯設計期末報告

報告題目：

汙染擴散模型

學生： 潘玠佑

學號： E94086076

授課教授姓名: 賴槿峰

一、　專題構想

一次工數上課時教到 gradient 計算濃度梯度、最大變化量及擴散方程，我便藉此發想是否可以用邏輯設計至今所學建立一個模型，用以計算汙染、亦或病毒的擴散方向，以此達到防患於未然。但由於積分器、微分器不易實作，經過深思考慮，我決定將模型化繁為簡，製作較為簡單的擴散模型。

二、　預期目標

輸入任意地形、以及任意汙染點之後，經由計算，可以列印出目前汙染的位置、時間以及受到汙染擴散災害的地區。由於我想和現今台灣疫情做關聯，因此我預設將地形初始化為台灣的模型，並且在一些地點設有北往南的火車，也就是在Ａ、Ｂ兩地，當汙染觸及點Ａ時，點Ｂ也會隨之被汙染。

三、　實作方式

地圖上的每格都是兩個 bits，00 代表未汙染點；01 是汙染點；11 則是牆壁。汙染會隨時間向四周擴散，並且無法穿透牆壁。

因此實作方式，先初始化台灣地圖，由於 ModelSim 無法使用矩陣，因此我採用有 2016 個 bits 的 reg（reg [2015:0] terrain）於 display 的時候再分割成矩陣的樣式，如右圖 1。

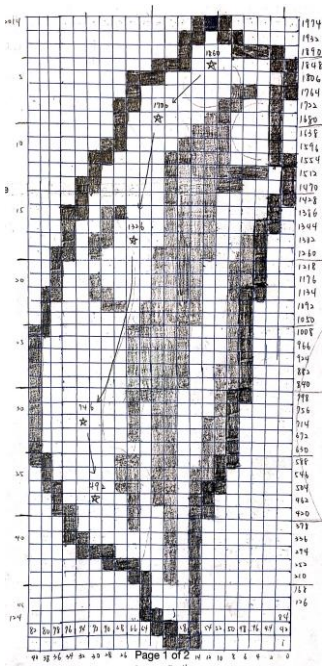


圖 1　台灣網格圖

構思完成之後，開始建立模組，如下圖 2。

[10:0] point 用來投放汙染點的座標位置；

[8:0] counter_start 用來設定日期開始時間；

[8:0] counter 用以輸出計算後的日期（月/日）；

[2015:0] result 列印出汙染點的位置和汙染後的圖。

```
1  // E94086076
2  module polution (
3  input [10:0] point,
4  input [8:0] counter_start,
5  input clk, rst,
6  output reg [2015:0] result,
7  output [8:0]counter
8  );
```

圖 2　建立 module

【註】 圖 1 中黑色方格是牆壁，星號是火車站，箭頭是指火車移動方向。圖寬 21 格（42bits）長 48 格，總共 1008 格（2016bits）。

接著，因為打一串 2016bits 的數字來初始化很容易打錯也不好看清楚，所以我使用串接來初始化地形：assign terrain = {g1,g2 … ,g48}，每個 reg g 有 42bits 分別對應每一列。

地形建置完成以後，接著設置污染點，使用 combinational circuit 使其可以任意時候立刻輸入汙染點。(汙染點可以不用在一開始輸入，也可以經過好幾個時脈以後再輸入，同時也可以多個點輸入。)

```
66  // initial points
67  always @ (*) begin
68          {terrain[point+1], terrain[point]} = 2'b01;
69  end
```

圖 3 輸入汙染點

接續使用 for loop 掃描並用 if else 找出汙染點，對汙染點四周的格子進行 OR 以此汙染四周。可以從下表 1 看出 AB 單個 bit OR 完以後再 assign 給 B (B+)

A	B	B+
01	00	01
01	01	01
01	11	11

表 1 真值表

```
integer i;
always @ (posedge clk or posedge rst) begin
        if (rst) begin
                for(i = 0; i < 2015; i = i + 2) begin
                        if (terrain[i + 1] != 1'b1 && terrain[i] != 1'b1) begin
                                {terrain[i+1],terrain[i]} = 2'b0;
                        end
                end
        end
        else begin
                for (i = 0; i < 2015; i = i + 2) begin
                        if (terrain[i + 1] == 1'b0 && terrain[i] == 1'b1) begin
                                {terrain[i+42 + 1], terrain[i+42]} <= {terrain[i+1] | terrain[i+1+42], terrain[i] | terrain[i+42]};     //up
                                {terrain[i-42 + 1], terrain[i-42]} <= {terrain[i+1] | terrain[i+1-42], terrain[i] | terrain[i-42]};     //down
                                {terrain[i+ 2 + 1], terrain[i+ 2]} <= {terrain[i+1] | terrain[i+1+ 2], terrain[i] | terrain[i+ 2]};     //left
                                {terrain[i- 2 + 1], terrain[i- 2]} <= {terrain[i+1] | terrain[i+1- 2], terrain[i] | terrain[i- 2]};     //right
                                {terrain[i+44 + 1], terrain[i+44]} <= {terrain[i+1] | terrain[i+1+44], terrain[i] | terrain[i+44]};     //up-left
                                {terrain[i+40 + 1], terrain[i+40]} <= {terrain[i+1] | terrain[i+1+40], terrain[i] | terrain[i+40]};     //up-rignt
                                {terrain[i-40 + 1], terrain[i-40]} <= {terrain[i+1] | terrain[i+1-40], terrain[i] | terrain[i-40]};     //down-left
                                {terrain[i-44 + 1], terrain[i-44]} <= {terrain[i+1] | terrain[i+1-44], terrain[i] | terrain[i-44]};     //down-right
                        end
                end
        end
end
```

圖 5 汙染點擴散

設置火車，使用 case。A、B 兩點就好比是兩地的火車站。當 A 點有人感染時，B 點的人也會跟著感染。為了接近真實情形，所以將火車設置為由北部駛往南部。並且火車站設置在台北、桃園、台中、台南和高雄。

```
97  // train
98  always @ (posedge clk or posedge rst) begin
99          if (rst) begin
100                 for(i = 0; i < 2015; i = i + 2) begin
101                         if (terrain[i + 1] != 1'b1 && terrain[i] != 1'b1) begin
102                                 {terrain[i+1],terrain[i]} = 2'b0;
103                         end
104                 end
105         end
106         else begin
107                 for(i = 0; i < 2015; i = i + 2) begin
108                         if ({terrain[i+1], terrain[i]} == 2'b01) begin
109                                 case (i)
110                                         11'd1860: begin
111                                                 {terrain[1701], terrain[1700]} = 2'b01;
112                                         end
113                                         11'd1700: begin
114                                                 {terrain[1327], terrain[1326]} = 2'b01;
115                                         end
116                                         11'd1326: begin
117                                                 {terrain[747], terrain[746]} = 2'b01;
118                                         end
119                                         11'd746: begin
120                                                 {terrain[493], terrain[492]} = 2'b01;
121                                         end
122                                 endcase
123                         end
124                 end
125         end
126 end
```

圖 6 火車

最後設置日期 counter 計算日期，為了方便我設計只有月跟日，所以只能計算今年，因此 2 月只有 28 天。同時也是期許疫情今年就可以完全結束。counter 有 9bits 是月和日的串接。前 4bits 是月，後 5bits 是日。counter 使用 case 包含 if else 做判斷，數到月底以後，日便會歸零，月會加一。

```
150 always @ (posedge clk or posedge rst) begin
151         if (rst) begin
152                 day <= 1;
153                 month <= 1;
154         end
155         else begin
156                 case(month)
157                         4'd2: begin
158                                 if (day == 5'd28) begin
159                                         day <= 5'd1;
160                                         month <= month + 1'b1;
161                                 end
162                         end
163                         4'd3: begin
164                                 if (day == 5'd30) begin
165                                         day <= 5'd1;
166                                         month <= month + 1'b1;
167                                 end
168                         end
169                         4'd5: begin
170                                 if (day == 5'd30) begin
171                                         day <= 5'd1;
172                                         month <= month + 1'b1;
173                                 end
174                         end
175                         4'd7: begin
176                                 if (day == 5'd30) begin
177                                         day <= 5'd1;
178                                         month <= month + 1'b1;
179                                 end
180                         end
181                         4'd11: begin
182                                 if (day == 5'd30) begin
183                                         day <= 5'd1;
184                                         month <= month + 1'b1;
185                                 end
186                         end
187                         4'd12: begin
188                                 if (day == 5'd31) begin
189                                         day <= 5'd1;
190                                         month <= 1'b1;
191                                 end
192                         end
193                         default: begin
194                                 if (day == 5'd31) begin
195                                         day <= 5'd1;
196                                         month <= month + 1'b1;
197                                 end
198                         end
199                 endcase
200                 day <= day + 1'b1;
201         end
202 end
```

圖 7 counter

四、 成果展示

reset 以後輸入汙染點並且從 6/1（011/000001）開始數，delay100 以後停止。

```
11  initial begin
12  clk = 0;
13  rst = 0;
14  #5 rst = 1;
15  #2 rst = 0;
16  counter_start = 9'b011000001;
17  point = 11'd1944;
18
19  #100 $stop;
```

圖 8 initial begin

將結果以每列 42bits display 出來，同時顯示時間。81 行的 for loop 會 display 當下汙染的點，並且將他們加總起來算出總共污染的點。

```
23  integer i, counter_p = 0;
24  always #5 clk = ~clk;
25  always @(posedge clk) begin
26
27  $display ("%d/%d\tresult=\n%42b", counter[8:5], counter[4:0], result[2015:1974]);
28  $display ("%42b", result[1973:1932]);
29  $display ("%42b", result[1931:1890]);
30  $display ("%42b", result[1889:1848]);
31  $display ("%42b", result[1847:1806]);
32  $display ("%42b", result[1805:1764]);
33  $display ("%42b", result[1763:1722]);
34  $display ("%42b", result[1721:1680]);
35
48  $display ("%42b", result[1259:1218]);
49  $display ("%42b", result[1217:1176]);
50  $display ("%42b", result[1175:1134]);
51  $display ("%42b", result[1133:1092]);
52  $display ("%42b", result[1091:1050]);
53  $display ("%42b", result[1049:1008]);
54  $display ("%42b", result[1007:966]);
55  $display ("%42b", result[965:924]);
56  $display ("%42b", result[923:882]);
57  $display ("%42b", result[881:840]);
46  $display ("%42b", result[1301:1260]);
```

```
59  $display ("%42b", result[839:798]);
60  $display ("%42b", result[797:756]);
61  $display ("%42b", result[755:714]);
62  $display ("%42b", result[713:672]);
63  $display ("%42b", result[671:630]);
64  $display ("%42b", result[629:588]);
65  $display ("%42b", result[587:546]);
66  $display ("%42b", result[545:504]);
67  $display ("%42b", result[503:462]);
68  $display ("%42b", result[461:420]);
69
70  $display ("%42b", result[419:378]);
71  $display ("%42b", result[377:336]);
72  $display ("%42b", result[335:294]);
73  $display ("%42b", result[293:252]);
74  $display ("%42b", result[251:210]);
75  $display ("%42b", result[209:168]);
76  $display ("%42b", result[167:126]);
77  $display ("%42b", result[125: 84]);
78  $display ("%42b", result[ 83: 42]);
79  $display ("%42b", result[ 41:  0]);
80
81  for (i = 0; i < 2015; i = i + 2) begin
82          if ({result[i+1], result[i]} == 2'b01) begin
83                  $display ("ponit[%4d]", i);
84                  counter_p = counter_p + 1;
85          end
86  end
87  $display ("total point: %4d", counter_p);
88  counter_p = 0;
89  end
```

圖 9 display result

Simulation 的波形圖，可以看出 rst（reset）完以後開始 run。

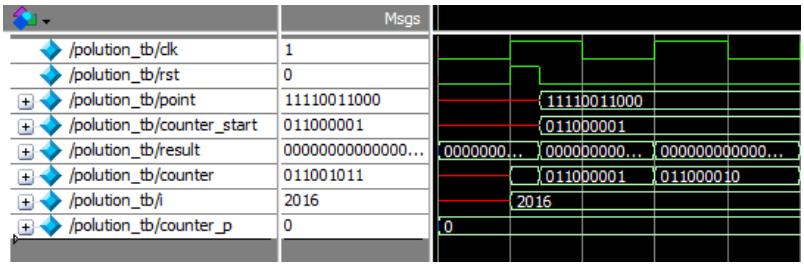
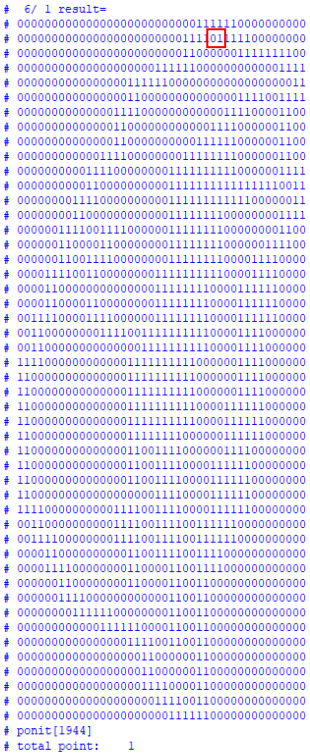
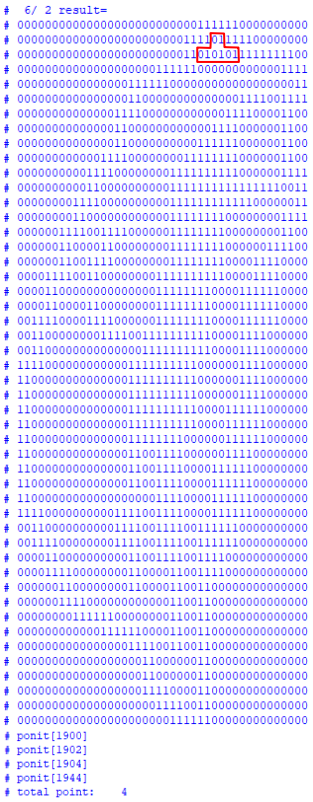
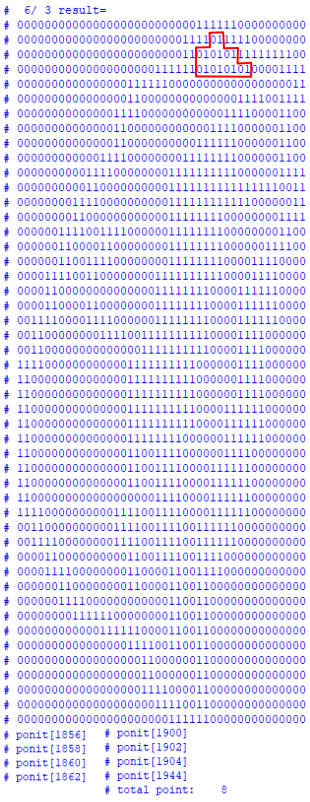


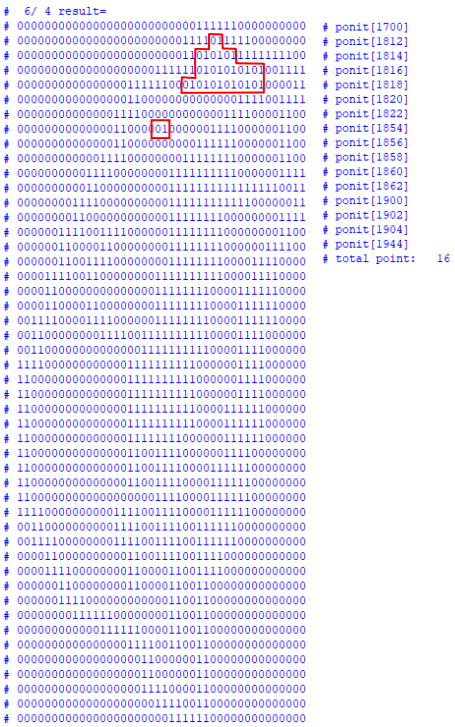
圖 10 wave

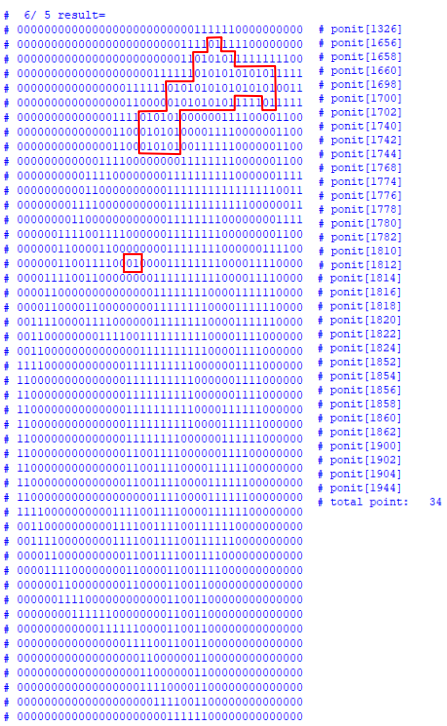
display 的圖。以下僅列出從 6/1 到 6/5 的圖來表示實作結果，並用紅色線條框出汙染的點。











【註】6/3 台北搭乘火車，因此 6/4 板橋汙染。

6/4 板橋搭乘火車，因此 6/5 台中汙染。

五、 未來展望

針對地形可以使用更多 register 存取數值以及增加解析度，可以更完整呈現盆地、平原、台地、丘陵以及高山，甚至可以引入積分器、微分器真實運算病毒、汙染的最大變化趨勢。

另外，為了使擴散模型更真實且使人更容易閱讀，可以在各個地區增加 delay 和 counter。delay 負責延遲擴散的速度，像是花蓮、台東的擴散速度會遠小於台北、新北；counter 負責加總各個區域的感染總數並互相比較數量之後回報哪個區域較為嚴重。

train 也可以做的更為真實，擴建大眾運輸系統，包含公車、捷運、火車、高鐵、飛機等。可以經由 delay 顯得更為真實。

另外還有許多可以擴建的功能像是人體模型，對每個人的行為模式輸入參數，可以更真實的模擬出病毒帶原者的移動範圍。期許未來引進各項運算系統後未來倘若遇到汙染或是病毒，可以更清楚如何防範。