# 數位電路期末專題報告

電機三乙 11128245 陳昱宇

# 目錄

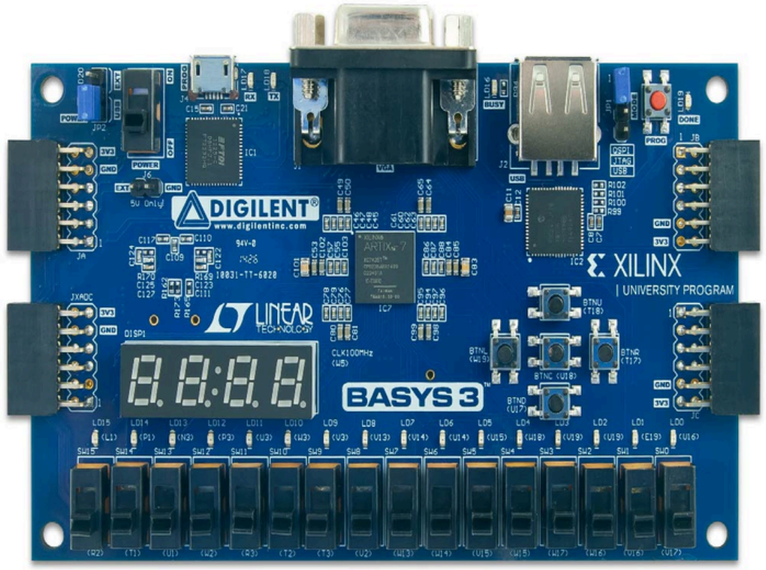
# 簡介

這次的期末專題，將整學期學習的內容整合。結合設計有限狀態機的數位控制電路、並整合暫存器和全加法進行資料的儲存與運算，最後利用解碼器輸出，將成果燒錄在BASYS 3板子呈現

# BASYS 3

BASYS 3 是由 Digilent 公司設計與製造的一款入門級 FPGA 開發板，專為數位電路設計與嵌入式系統學習而設計。

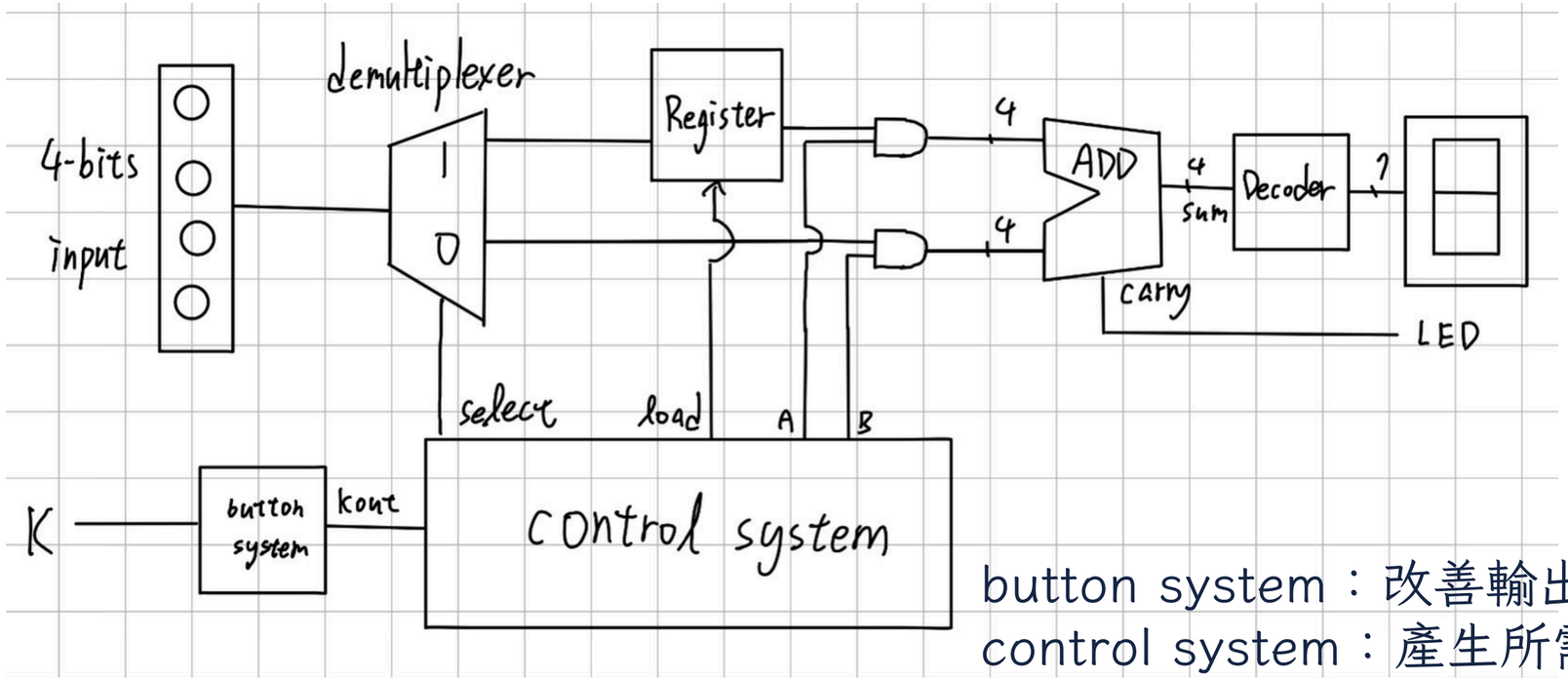
該板子採用了 Xilinx Artix-7 FPGA 作為核心處理單元，提供強大的計算能力與靈活性，適合進行各類數位系統的設計與驗證。



# 電路設計

設計電路功能：
使用指撥開關，透過控制彈跳開關(K)，將數值依次輸入，加數、被加數，最後使其相加，將結果顯示在七段顯示器以及LED上

電路圖：




button system：改善輸出過長的高電位信號
control system：產生所需控制訊號

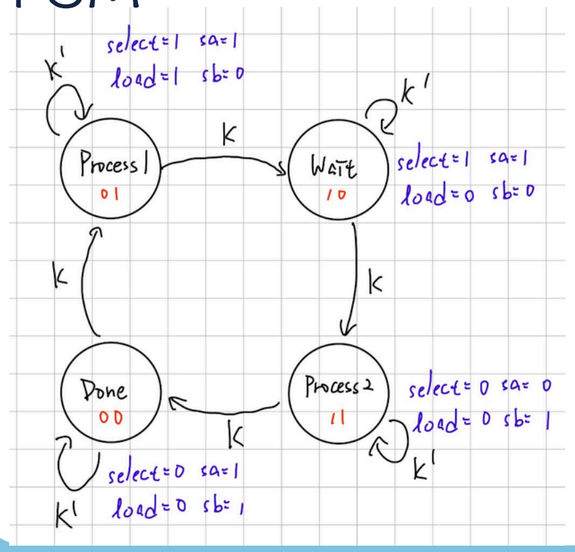
# 狀態機設計

有限狀態機初始狀態為Process 1(選擇被加數)
按下按鈕切換為下個狀態Wait(此時顯示被加數)(選擇加數)
再按下按鈕切換狀態為Process 2(此時顯示加數)
再度按下按鈕會將加法結果呈現出來(Done)
再度按下按鈕(以上不斷迴圈)

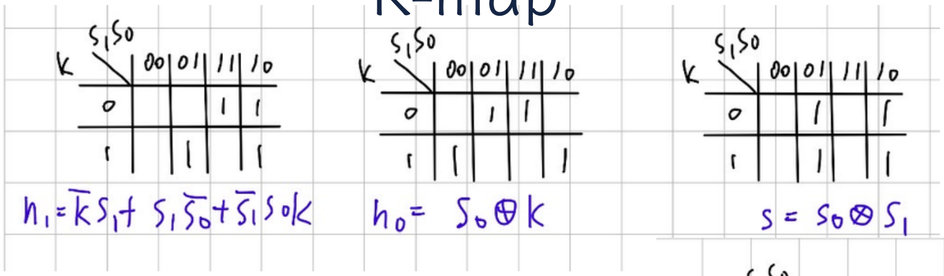
## state table

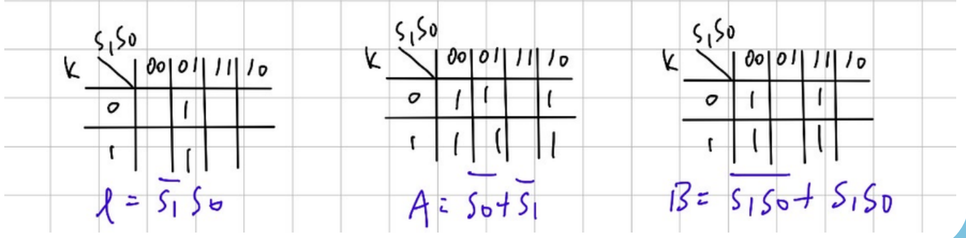
| Input | | | Output | | | | | |
|---|---|---|---|---|---|---|---|---|
| $S_1$ | $S_0$ | k | $n_1$ | $n_0$ | s | l | A | B |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |

## FSM




## K-map



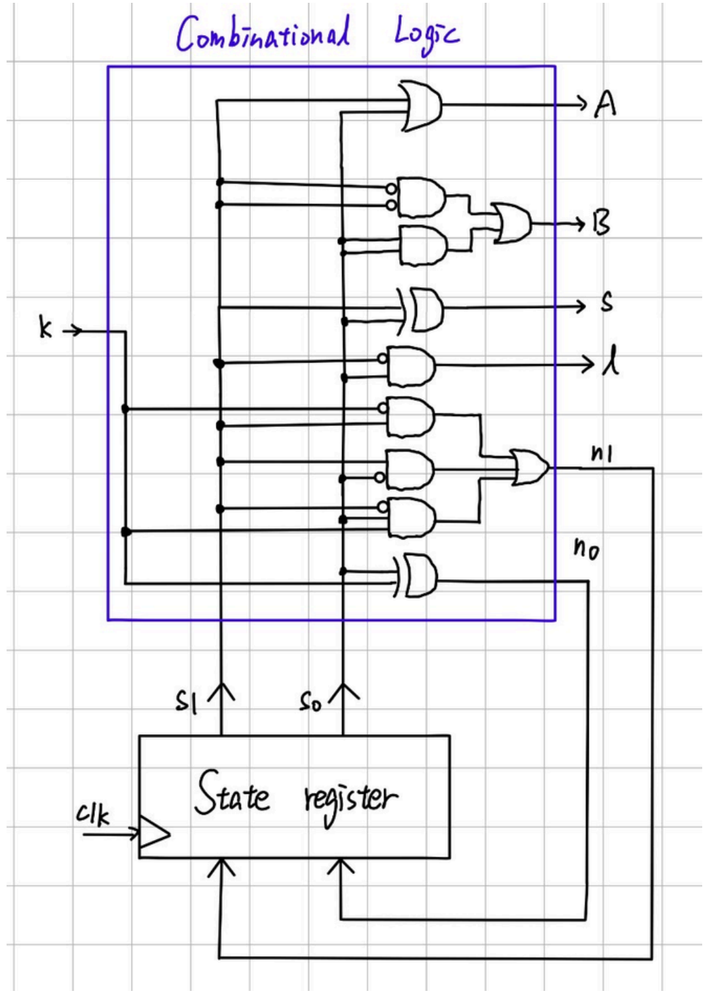



# 狀態機設計

## Verilog code

```
`timescale 1ns / 1ps
module FSM(clock, reset, kout, select, load, n0,n1,s0,s1,sa,sb);
    input clock;
    input reset;
    input kout;
    output select;
    output load;
    output sa, sb;
    output n0, n1, s0, s1;
    wire n0,n1,s0,s1;

    assign n1=(~kout&s1 | s1&~s0 | ~s1&s0&kout);
    assign n0=(s0^kout);
    assign select=s1^s0;
    assign load=~s1&s0;
    assign sa=~s0 |~s1;
    assign sb=(~s1&~s0 | s1&s0);

    FDCE#
    (.INIT(1'b1) )FDCE_inst_1(
    .Q(s0),
    .C(clock),
    .CE(1'b1),
    .CLR(reset),
    .D(n0));

    FDCE FDCE_inst_2(
    .Q(s1),
    .C(clock),
    .CE(1'b1),
    .CLR(reset),
    .D(n1));

endmodule
```

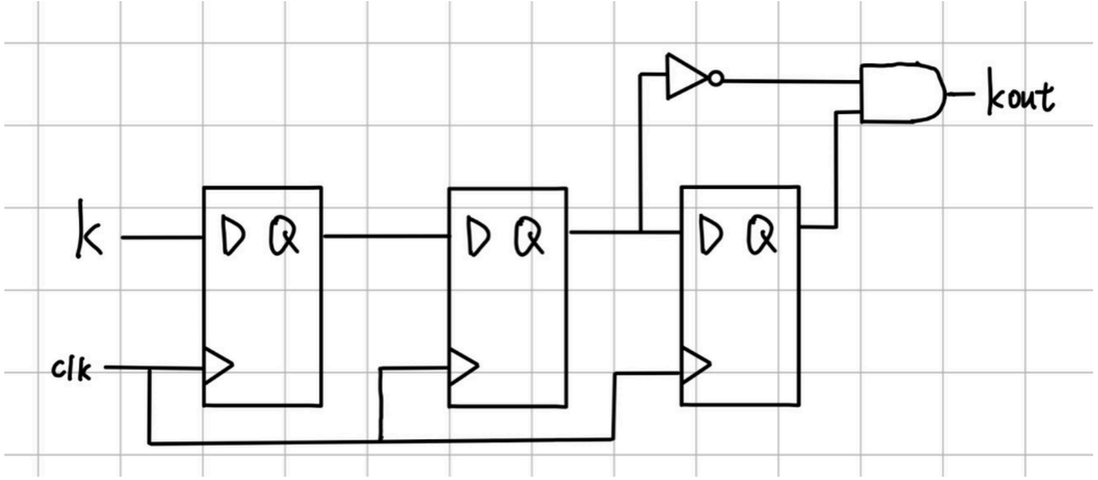
## Implement combinational logic



# 模組電路(KOUT)

此電路是為了改善當按鈕按下時FSM的快速切換狀態，用來改善輸出過長的高電位信號

logic circuit




Verilog code

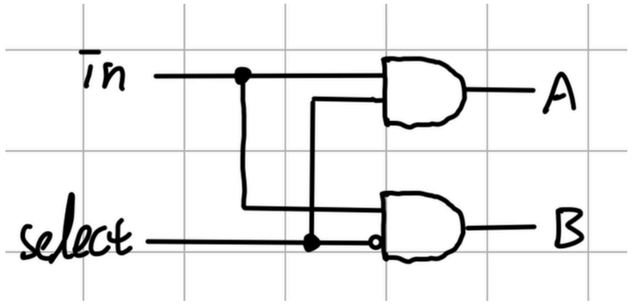
```
`timescale 1ns / 1ps
module KOUT(clock,reset,k,s0,s1,s2,kout);
    input clock,k,reset;
    output s0, s1, s2;
    output kout;

    assign kout=(~s1&s2);

    FDCE FDCE_inst_1(
     .Q(s0),
     .C(clock),
     .CE(1'b1),
     .CLR(reset),
     .D(k));

    FDCE FDCE_inst_2(
     .Q(s1),
     .C(clock),
     .CE(1'b1),
     .CLR(reset),
     .D(s0));

     FDCE FDCE_inst_3(
     .Q(s2),
     .C(clock),
     .CE(1'b1),
     .CLR(reset),
     .D(s1));

endmodule
```

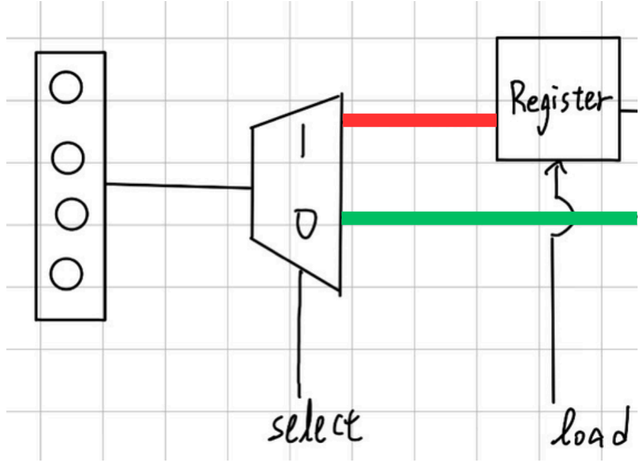
# 模組電路(解多公器)

此電路能將輸入切換到不同的線路輸出
並藉由select控制

//當select = 1 輸出為紅色
//當select = 0 輸出為綠色

logic circuit




Verilog code

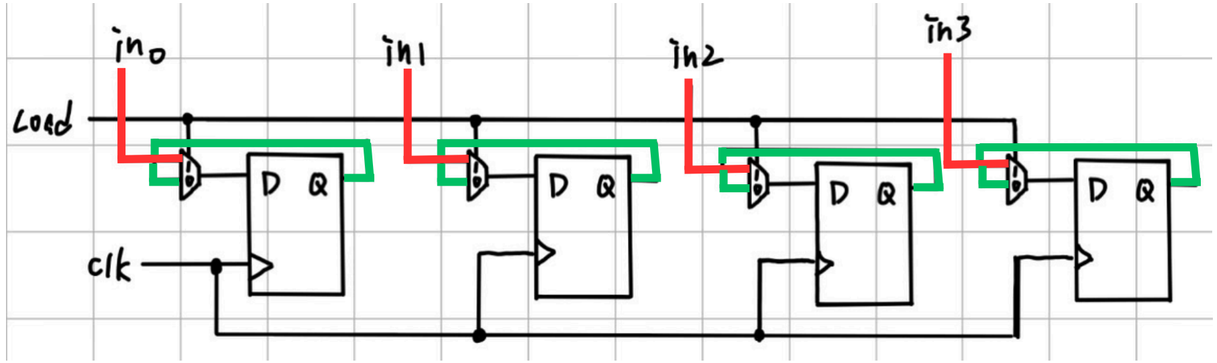
```
`timescale 1ns / 1ps
module demultiplexer(in,select,A,B );
    input [3:0]in;
    input select;
    output [3:0]A;
    output [3:0]B;

    assign A[0] = in[0] & select;
    assign A[1] = in[1] & select;
    assign A[2] = in[2] & select;
    assign A[3] = in[3] & select;
    assign B[0] = in[0] & ~select;
    assign B[1] = in[1] & ~select;
    assign B[2] = in[2] & ~select;
    assign B[3] = in[3] & ~select;
endmodule
```

# 模組電路(暫存器)

此電路能將輸入儲存並不斷輸出
藉由load控制資料存儲

//當load = 1 為紅色
//當load = 0 為綠色

logic circuit



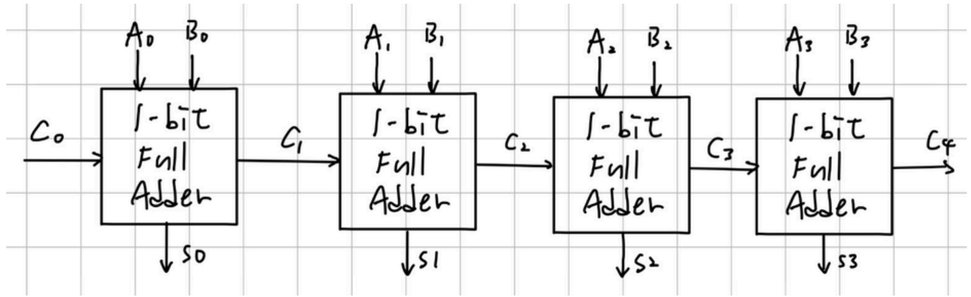

## Verilog code

```
`timescale 1ns / 1ps
module register(I,load,clock,reset,R );
    input [3:0]I;
    input load,clock,reset;
    output [3:0]R;

    assign D0=(I[0]&load)|(R[0]&~load);
    assign D1=(I[1]&load)|(R[1]&~load);
    assign D2=(I[2]&load)|(R[2]&~load);
    assign D3=(I[3]&load)|(R[3]&~load);

    FDCE FDCE_inst_1(
    .Q(R[0]),
    .C(clock),
    .CE(1'b1),
    .CLR(reset),
    .D(D0));

    FDCE FDCE_inst_2(
    .Q(R[1]),
    .C(clock),
    .CE(1'b1),
    .CLR(reset),
    .D(D1));

    FDCE FDCE_inst_3(
    .Q(R[2]),
    .C(clock),
    .CE(1'b1),
    .CLR(reset),
    .D(D2));

    FDCE FDCE_inst_4(
    .Q(R[3]),
    .C(clock),
    .CE(1'b1),
    .CLR(reset),
    .D(D3));

endmodule
```

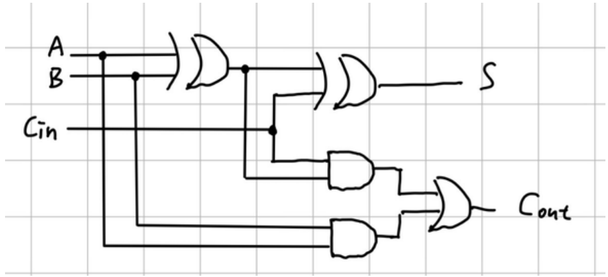
# 模組電路(加法器)

此電路能將輸入相加後輸出

## logic circuit




## full adder




## Verilog code

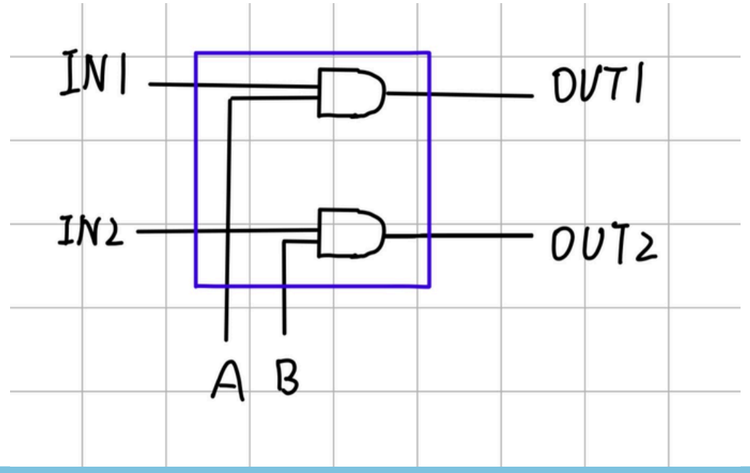
```
`timescale 1ns / 1ps
module adder (A, B, Cin, Sum, Cout);
    input A, B, Cin;
    output Sum, Cout;
    assign Sum = A ^ B ^ Cin;
    assign Cout = (A & B) | (Cin & (A ^ B));
endmodule

module adder4b (A, B, Ci, S, Co);
    input [3:0]A, B;
    input Ci;
    output [3:0] S;
    output Co;
    wire c_out_wire0, c_out_wire1, c_out_wire2;

    adder FA0 (.A(A[0]), .B(B[0]), .Cin(Ci),          .Sum(S[0]), .Cout(c_out_wire0));
    adder FA1 (.A(A[1]), .B(B[1]), .Cin(c_out_wire0), .Sum(S[1]), .Cout(c_out_wire1));
    adder FA2 (.A(A[2]), .B(B[2]), .Cin(c_out_wire1), .Sum(S[2]), .Cout(c_out_wire2));
    adder FA3 (.A(A[3]), .B(B[3]), .Cin(c_out_wire2), .Sum(S[3]), .Cout(Co));
endmodule
```

# 模組電路(多工器系統)

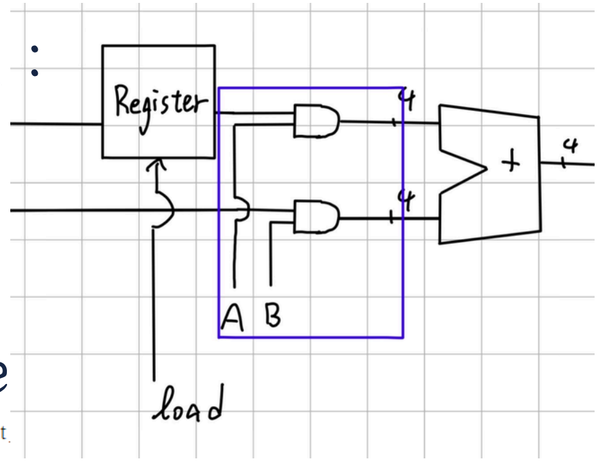
輸入資料：有兩筆資料 IN1 和 IN2
控制端口：A、B
輸出端口：有兩個輸出端口 OUT1 和 OUT2
狀態：這個設計有三種可能的輸出組合：

- 狀態1：當 OUT1 = IN1, OUT2 = 0　(A=1、B=0)
- 狀態2：當 OUT1 = 0　, OUT2 = IN2(A=0、B=1)
- 狀態3：當 OUT1 = IN1, OUT2 = IN2(A=1、B=1)

logic circuit




電路位置：




Verilog code

```
wire c_out_wire0, c_out
wire [3:0]AA,BB ;

assign AA[0]=A[0]&sa;
assign AA[1]=A[1]&sa;
assign AA[2]=A[2]&sa;
assign AA[3]=A[3]&sa;
assign BB[0]=B[0]&sb;
assign BB[1]=B[1]&sb;
assign BB[2]=B[2]&sb;
assign BB[3]=B[3]&sb;

adder FA0 (.A(AA[0]), .B(BB[0]), .Cin(Ci),          .Sum(S[0]), .Cout(c_out_wire0));
adder FA1 (.A(AA[1]), .B(BB[1]), .Cin(c_out_wire0), .Sum(S[1]), .Cout(c_out_wire1));
adder FA2 (.A(AA[2]), .B(BB[2]), .Cin(c_out_wire1), .Sum(S[2]), .Cout(c_out_wire2));
adder FA3 (.A(AA[3]), .B(BB[3]), .Cin(c_out_wire2), .Sum(S[3]), .Cout(Co));
endmodule
```

//這裡將邏輯電路直接串在加法器前

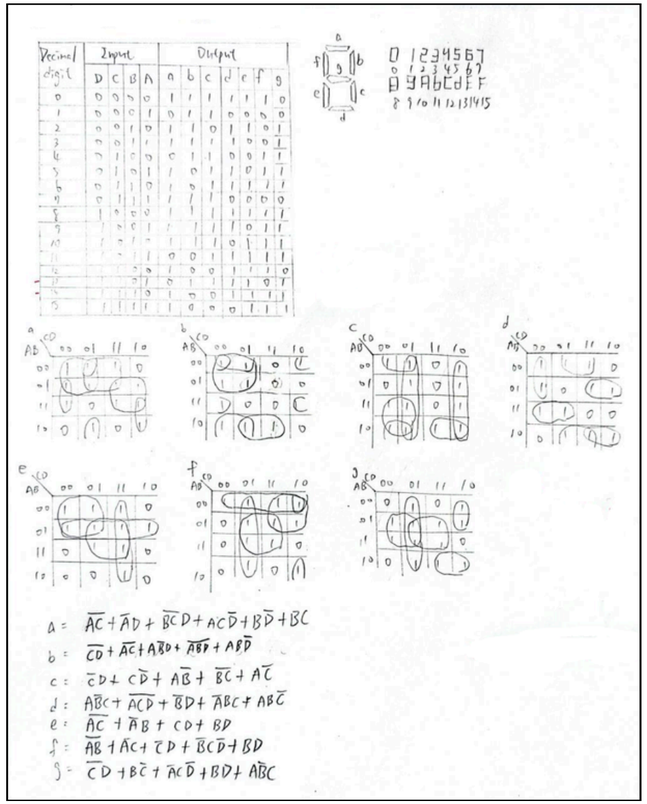
# 模組電路(解碼器)

此電路能將二進制的輸入
轉換成七段顯示器的輸出

## Verilog code

```
`timescale 1ns / 1ps
module decoder( binary_in,segment_out);
    input [3:0] binary_in;
    output [6:0] segment_out;
    wire a, b, c, d;
    assign {d, c, b, a} = binary_in;

    assign segment_out[0] = ~((~a & ~c) | (~a & d) | (~b & ~c & d) | (a & c & ~d) | (b & ~d) | (b & c));
    assign segment_out[1] = ~((~c & ~d) | (~a & ~c) | (a & ~b & d) | (~a & ~b & ~d) | (a & b &~d));
    assign segment_out[2] = ~((~c & d) | (c & ~d) | (a & ~b) | (~b & ~c) | (a & ~c));
    assign segment_out[3] = ~((a & ~b & c) | (~a & ~c & ~d) | (~b & d) | (~a & b & c) | (a & b & ~c));
    assign segment_out[4] = ~((~a & ~c) | (~a & b) | (c & d) | (b & d));
    assign segment_out[5] = ~((~a & ~b) | (~a & c) | (~c & d) | (~b & c & ~d) | (b & d));
    assign segment_out[6] = ~((~c & d) | (b & ~c) | (~a & c & ~d) | (b & d) | (a & ~b & c));
endmodule
```

設計七段顯示器的解碼器
下圖為共陰極(code設計為陽極)



# 模組電路(TOP.v)

Verilog code

```
`timescale 1ns / 1ps
module TOP(
    input clock,
    input reset,
    input k,
    input [3:0]in,
    output [6:0] seg_sum,
    output [6:0] seg_carry,
    output Co
);
    wire select, load;
    wire [3:0] A, B, R;
    wire [3:0] S;
    wire sa,sb;
```

```
KOUT kout_inst (
    .clock(clock),
    .reset(reset),
    .kout(kout),
    .k(k)
);

FSM fsm_inst (
    .clock(clock),
    .reset(reset),
    .kout(kout),
    .select(select),
    .load(load),
    .sa(sa),
    .sb(sb)
);
```

```
demultiplexer demux_inst (
    .in(in),
    .select(select),
    .A(A),
    .B(B)
);

register reg_inst (
    .I(A),
    .load(load),
    .clock(clock),
    .reset(reset),
    .R(R)
);
```

```
adder4b adder_inst (
    .A(R),
    .B(B),
    .Ci(1'b0),
    .S(S),
    .Co(Co),
    .sa(sa),
    .sb(sb)
);
        decoder seg_sum_inst (
            .binary_in(S),
            .segment_out(seg_sum)
        );

        decoder seg_carry_inst (
            .binary_in({3'b000, Co}).
            .segment_out(seg_carry)
        );

    endmodule
```

# 模組電路(tb_TOP.v)

Verilog code

```
`timescale 1ns / 1ps
module tb_TOP;
    reg clock;
    reg reset;
    reg k;
    reg [3:0] in;
    wire [6:0] seg_sum;
    wire [6:0] seg_carry;

    final uut (
        .clock(clock),
        .reset(reset),
        .k(k),
        .in(in),
        .seg_sum(seg_sum),
        .seg_carry(seg_carry)
    );
```

```
    initial begin
        reset = 1;
        k = 0;
        in = 4'b0000;
        #50 reset = 0;
        #10 k = 0;
        #50 in = 4'b1000;
        #10 k = 1;
        #50 k = 0;
        #60 k = 1; in = 4'b1010;
        #50 k = 0;
        #40 k=1;
        #50 k=0;
        #150
        $finish;
    end
endmodule
```

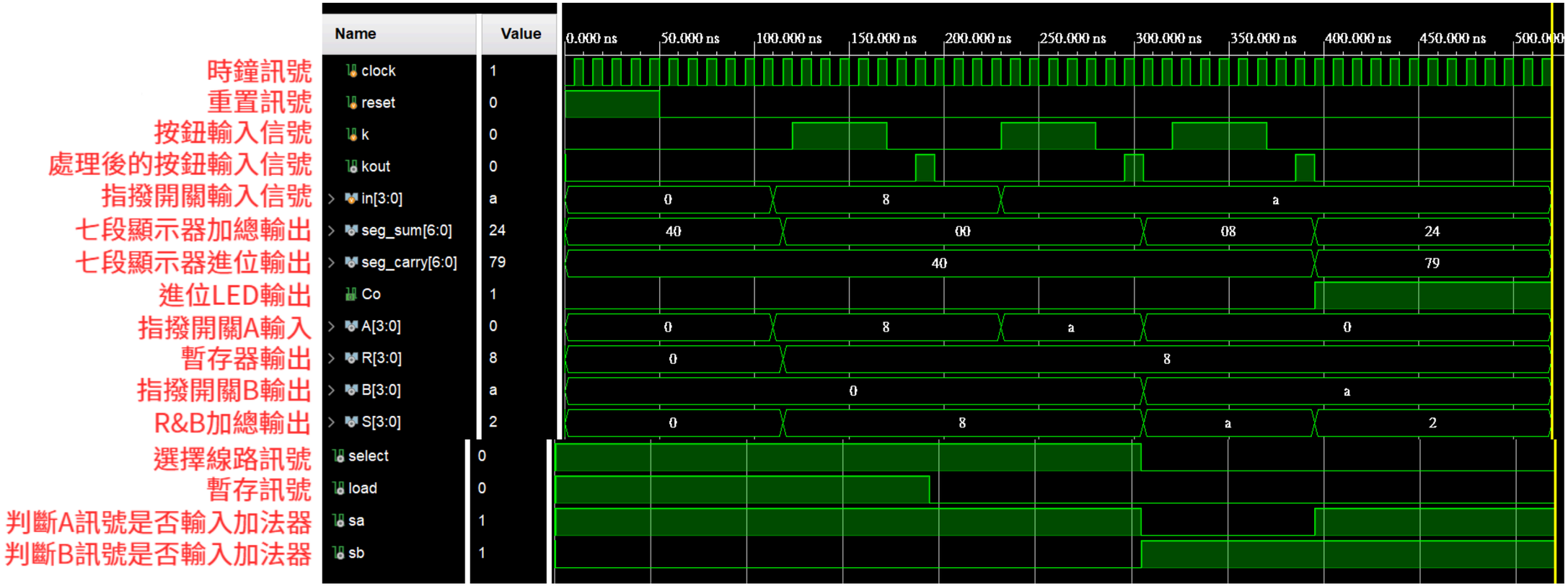
```
initial begin
    clock = 0;
    forever #5 clock = ~clock;
end
```

# 約束檔腳位設置

```
## Clock signal
set_property PACKAGE_PIN W5 [get_ports clock]
set_property IOSTANDARD LVCMOS33 [get_ports clock]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clock]

## Switches
set_property PACKAGE_PIN V17 [get_ports {in[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {in[0]}]
set_property PACKAGE_PIN V16 [get_ports {in[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {in[1]}]
set_property PACKAGE_PIN W16 [get_ports {in[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {in[2]}]
set_property PACKAGE_PIN W17 [get_ports {in[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {in[3]}]

## LEDs
set_property PACKAGE_PIN U16 [get_ports {pulse}]
set_property IOSTANDARD LVCMOS33 [get_ports {pulse}]
set_property PACKAGE_PIN E19 [get_ports {Co}]
    set_property IOSTANDARD LVCMOS33 [get_ports {Co}]
```

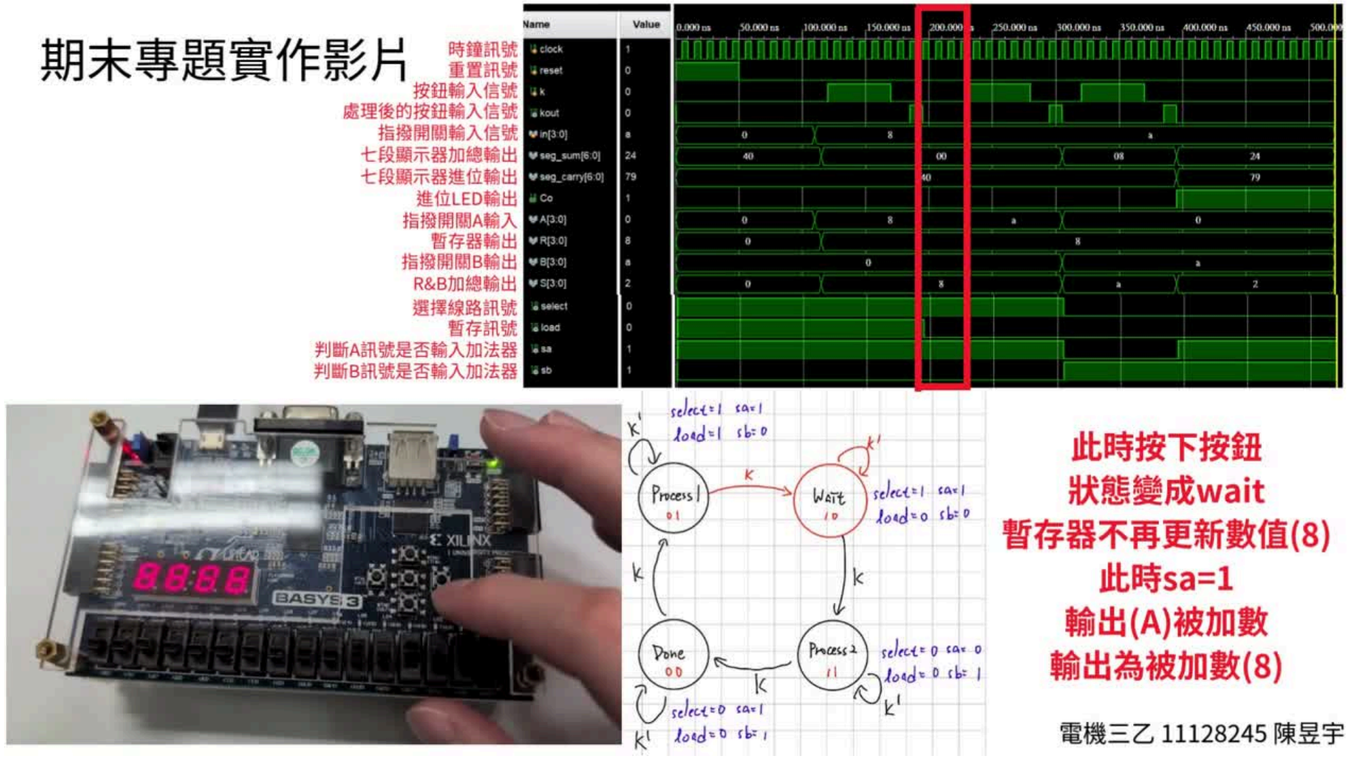
```
##7 segment display
set_property PACKAGE_PIN W7 [get_ports {seg_sum[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg_sum[0]}]
set_property PACKAGE_PIN W6 [get_ports {seg_sum[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg_sum[1]}]
set_property PACKAGE_PIN U8 [get_ports {seg_sum[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg_sum[2]}]
set_property PACKAGE_PIN V8 [get_ports {seg_sum[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg_sum[3]}]
set_property PACKAGE_PIN U5 [get_ports {seg_sum[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg_sum[4]}]
set_property PACKAGE_PIN V5 [get_ports {seg_sum[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg_sum[5]}]
set_property PACKAGE_PIN U7 [get_ports {seg_sum[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {seg_sum[6]}]

##Buttons
set_property PACKAGE_PIN U18 [get_ports reset]
set_property IOSTANDARD LVCMOS33 [get_ports reset]
set_property PACKAGE_PIN T18 [get_ports k]
    set_property IOSTANDARD LVCMOS33 [get_ports k]
```

# 波型模擬


時鐘訊號
重置訊號
按鈕輸入信號
處理後的按鈕輸入信號
指撥開關輸入信號
七段顯示器加總輸出
七段顯示器進位輸出
進位LED輸出
指撥開關A輸入
暫存器輸出
指撥開關B輸出
R&B加總輸出
選擇線路訊號
暫存訊號
判斷A訊號是否輸入加法器
判斷B訊號是否輸入加法器
Name
Value
clock 1
reset 0
k 0
kout 0
in[3:0] a
seg_sum[6:0] 24
seg_carry[6:0] 79
Co 1
A[3:0] 0
R[3:0] 8
B[3:0] a
S[3:0] 2
select 0
load 0
sa 1
sb 1
0.000 ns
50.000 ns
100.000 ns
150.000 ns
200.000 ns
250.000 ns
300.000 ns
350.000 ns
400.000 ns
450.000 ns
500.000

# 專題成果

期末專題實作影片




https://youtu.be/hRMxEYwLPM8

# 心得與結論

本次的期末專題，將整個課程學習的內容整合，讓我更熟悉軟體的操作。無論是在撰寫程式、燒錄板子，中間除錯的過程，還有在最後一次上課中重新調整電路的壓力，這些吸收到的經驗，都令我受益良多。

# THANK YOU!