**微算機系統**

**小組專案報告**

實驗九

組別： 18

班級、姓名與學號：

資工二 蕭耕宏 110590005

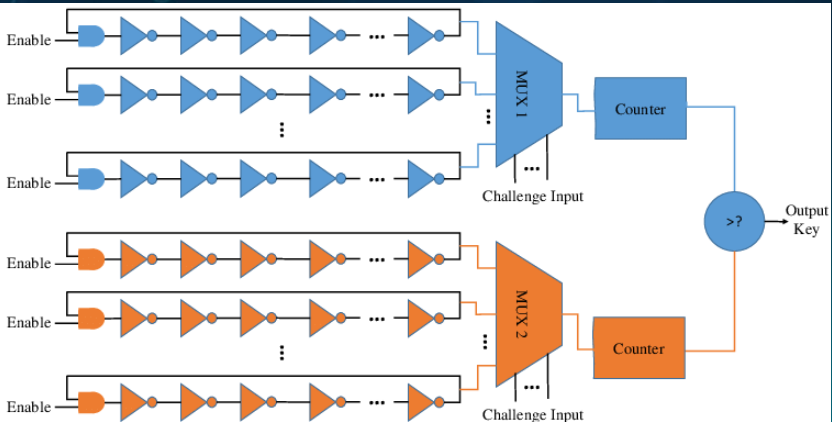
資工二 楊榮鈞 110590034

日期： 2023.01.10

1. 實驗內容：

本次實驗是，透過RO，實作一個PUF(Physically Unclonable Function)，中文為物理不可仿製功能。環形振盪器的階層數從7階到21階，使用50MHZ來產生1HZ的輸出頻率。

1. 實驗過程及結果：

PUF架構圖

實驗的結果

|  |  |
| --- | --- |
| 110590005->11059000->11057000->  9 9 7 17 21 7 7 7  430095011->30095011->30175111->  13 7 9 21 17 9 9 9 | LED(7 downto 0) (red)  LED6和LED3會暗  最左邊的綠色LED是reset  reset旁邊的兩個綠色LED是  count1和count2 |
|  |  |
| sw = 000, reset = 1 | sw = 011, reset = 1 |
|  |  |
| sw = 011, reset = 0 | sw = 110, reset = 1 |

1. 程式碼

|  |
| --- |
| 進階題 |
| Lab9.vhd |
| library ieee;  use ieee.std\_logic\_1164.all;  --use ieee.std\_logic\_arith.all;  use ieee.numeric\_std.all;  use ieee.std\_logic\_unsigned.all;  use work.lab9\_package.all;  --7 9 11 13 15 17 19 21 <- 階層  --N = 2\*i + 7  --11059000 ->110590005  --30095011 ->110590034  --30195111 ->repeat +1  -- n -> 0~7 (8,9 -> 7)  -- 0=7, 1=9, 2=11, 3=13, 4=15, 5=17, 6=19, 7=21  --11057000  --30175111  entity Lab9 is  port( clk\_50M : in std\_logic;  reset : in std\_logic;  sw : in std\_logic\_vector(2 downto 0);  led1 : out std\_logic;  led2 : out std\_logic;  LED : buffer std\_logic\_vector(8 downto 0)  );  end Lab9;  architecture behavior of Lab9 is  signal clk\_1hz : std\_logic;  signal count1 : std\_logic\_vector(127 downto 0);  signal count2 : std\_logic\_vector(127 downto 0);  signal comp : std\_logic;  signal n1 : std\_logic;  signal n2 : std\_logic;  type t\_N is array (0 to 7, 0 to 1) of std\_logic;  signal N\_arr : t\_N;  begin    stage0 : divider\_n port map(clk\_50M, clk\_1hz);    ro00 : R\_O\_n generic map(7) port map(reset, clk\_50M, N\_arr(0,0));  ro01 : R\_O\_n generic map(7) port map(reset, clk\_50M, N\_arr(1,0));  ro02 : R\_O\_n generic map(7) port map(reset, clk\_50M, N\_arr(2,0));  ro03 : R\_O\_n generic map(21) port map(reset, clk\_50M, N\_arr(3,0));  ro04 : R\_O\_n generic map(17) port map(reset, clk\_50M, N\_arr(4,0));  ro05 : R\_O\_n generic map(7) port map(reset, clk\_50M, N\_arr(5,0));  ro06 : R\_O\_n generic map(9) port map(reset, clk\_50M, N\_arr(6,0));  ro07 : R\_O\_n generic map(9) port map(reset, clk\_50M, N\_arr(7,0));    ro10 : R\_O\_n generic map(9) port map(reset, clk\_50M, N\_arr(0,1));  ro11 : R\_O\_n generic map(9) port map(reset, clk\_50M, N\_arr(1,1));  ro12 : R\_O\_n generic map(9) port map(reset, clk\_50M, N\_arr(2,1));  ro13 : R\_O\_n generic map(17) port map(reset, clk\_50M, N\_arr(3,1));  ro14 : R\_O\_n generic map(21) port map(reset, clk\_50M, N\_arr(4,1));  ro15 : R\_O\_n generic map(9) port map(reset, clk\_50M, N\_arr(5,1));  ro16 : R\_O\_n generic map(7) port map(reset, clk\_50M, N\_arr(6,1));  ro17 : R\_O\_n generic map(13) port map(reset, clk\_50M, N\_arr(7,1));    select0 : mux8to1 port map( N\_arr(0,0),  N\_arr(1,0),  N\_arr(2,0),  N\_arr(3,0),  N\_arr(4,0),  N\_arr(5,0),  N\_arr(6,0),  N\_arr(7,0),  sw(2 downto 0),  n1);  select1 : mux8to1 port map( N\_arr(0,1),  N\_arr(1,1),  N\_arr(2,1),  N\_arr(3,1),  N\_arr(4,1),  N\_arr(5,1),  N\_arr(6,1),  N\_arr(7,1),  sw(2 downto 0),  n2);    cal0 : cal\_freq port map(reset, clk\_1hz, n1, count1);  cal1 : cal\_freq port map(reset, clk\_1hz, n2, count2);    cp : compare port map(count1, count2, comp);    process(clk\_50M)  begin  if(reset = '0') then  LED(7 downto 0) <= (others => '1');  LED(8) <= reset;  elsif (reset = '1') then  LED(8) <= reset;  if (count1 > count2) then  LED(to\_integer(unsigned(sw))) <= '1';  elsif (count1 <= count2) then  LED(to\_integer(unsigned(sw))) <= '0';  end if;  end if;  end process;  led1 <= '1' when to\_integer(unsigned(count1)) mod 5000000 > 2500000 else  '0';    led2 <= '1' when to\_integer(unsigned(count2)) mod 5000000 > 2500000 else  '0';    end behavior; |
| R\_O\_n.vhd |
| library ieee;  use ieee.std\_logic\_1164.all;  use ieee.std\_logic\_arith.all;  use ieee.std\_logic\_unsigned.all;  entity R\_O\_n is  generic (n : integer := 7); --N階環形震盪  port( enable : in std\_logic;  clk\_50M : in std\_logic;  ck\_out : out std\_logic  );  end R\_O\_n;  architecture behavior of R\_O\_n is  signal chain : std\_logic\_vector(n-1 downto 0);  attribute syn\_keep:boolean;  attribute syn\_keep of chain:signal is true;    begin  process(clk\_50M)  begin  if(clk\_50M'event and clk\_50M = '1') then    for i in 1 to n-1 loop  chain(i) <= not chain(i-1);  end loop;  end if;  end process;  chain(0) <= enable and (not chain(n-1));  ck\_out <= chain(n-1);  end behavior; |
| mux\_8to1.vhd |
| library ieee;  use ieee.std\_logic\_1164.all;  entity mux8to1 is  port( w0,w1,w2,w3,w4,w5,w6,w7 : in std\_logic;  s : std\_logic\_vector(2 downto 0);  f : out std\_logic  );  end mux8to1;    architecture behavior of mux8to1 is  begin  with s select  f <= w0 when "000",  w1 when "001",  w2 when "010",  w3 when "011",  w4 when "100",  w5 when "101",  w6 when "110",  w7 when others;  end behavior; |
| divider\_n.vhd |
| library ieee;  use ieee.std\_logic\_1164.all;  use ieee.std\_logic\_unsigned.all;  use ieee.std\_logic\_arith.all;  entity divider\_n is  generic(N : integer :=50000000);  port( clk\_50M : in std\_logic;  clk\_out : out std\_logic);  end divider\_n;  architecture behavior of divider\_n is  signal count2 : std\_logic;  shared variable count1 : integer range 0 to N := 1;  shared variable temp : integer range 0 to N;    begin  process(clk\_50M)  begin  temp := N/2;  if clk\_50M 'event and clk\_50M='1' then  if count1 = N then  count1 := 1;  else  count1 := count1 + 1;  end if;  end if;    if clk\_50M 'event and clk\_50M='1' then  if ((count1 = temp)or(count1 = N)) then  count2 <= not count2;  end if;  end if;  clk\_out <= count2;  end process;  end behavior; |
| cal\_freq.vhd |
| library ieee;  use ieee.std\_logic\_1164.all;  --use ieee.std\_logic\_arith.all;  use ieee.numeric\_std.all;  use ieee.std\_logic\_unsigned.all;  entity cal\_freq is  port( res : in std\_logic;  clk\_1Hz : in std\_logic;  clk\_nHz : in std\_logic;  nHz : buffer std\_logic\_vector(127 downto 0)  );  end cal\_freq;  architecture behavior of cal\_freq is  begin  process (clk\_nhz, res)  begin  if (res = '0') then  nHz <= (others => '0');  elsif (clk\_nhz'event and clk\_nhz='1') then  -- if (clk\_1hz = '0') then  nHz <= nHz+1;  -- end if;  end if;  end process;  end behavior; |
| compare.vhd |
| library ieee;  use ieee.std\_logic\_1164.all;  use ieee.std\_logic\_unsigned.all;  use ieee.numeric\_std.all;  entity compare is  port( val\_a : in std\_logic\_vector(127 downto 0);  val\_b : in std\_logic\_vector(127 downto 0);  result : out std\_logic  );  end compare;  architecture behavior of compare is  begin  result <= '1' when (unsigned(val\_a) > unsigned(val\_b)) else '0';  end behavior; |
| Lab9\_package.vhd |
| library ieee;  use ieee.std\_logic\_1164.all;  package lab9\_package is  component R\_O\_n  generic (n : integer := 7);  port( enable : in std\_logic;  clk\_50M : in std\_logic;  ck\_out : out std\_logic  );  end component R\_O\_n;    component mux8to1  port( w0,w1,w2,w3,w4,w5,w6,w7 : in std\_logic;  s : std\_logic\_vector(2 downto 0);  f : out std\_logic  );  end component mux8to1;    component divider\_n  port( clk\_50M : in std\_logic;  clk\_out : out std\_logic  );  end component divider\_n;    component cal\_freq  port( res : in std\_logic;  clk\_1Hz : in std\_logic;  clk\_nHz : in std\_logic;  nHz : buffer std\_logic\_vector(127 downto 0)  );  end component cal\_freq;    component compare  port( val\_a : in std\_logic\_vector(127 downto 0);  val\_b : in std\_logic\_vector(127 downto 0);  result : out std\_logic  );  end component compare;    end lab9\_package; |

1. 本次實驗過程說明與解決方法:

實驗過程:

一開始先按照老師ppt的code打出來，然後開始燒錄。燒好之後，發現LED燈會亂跳，我們認為是cal\_freq的地方錯誤，所以我們把邏輯計算頻率的條件改成1HZ和nHZ皆為1 (高態)的時候加一。然後重新燒錄之後，結果變成每個燈都不亮。發現是reset的邏輯有錯，調整好reset之後，我們再重新測試一次，發現我們R\_O\_n的地方的輸出只有0，然後沒有變化(就是輸出恆0)。

於是我們跟其他做完的組別討論後，發現我們的R\_O\_n不會動的原因，有可能是因為我們的值沒有變化，所以他只會執行一次就停下來，因此我們改成用process去實作環形振盪器。在解決完R\_O\_n的問題之後，我們發現我們的cal\_freq計算的頻率有問題，於是我們讓他的計數條件變為nHz觸發上緣時count再加一。在處理完這些bug之後，我們發現我們的比較器出現問題，原因不明(時間不夠讓我們找完為什麼出現bug)，所以我們直接在主程式用process的if-else去比較頻率大小。

改完這些bug之後就成功了。

主要問題:

R\_O\_n的輸出只有0沒有變化，cal\_freq的計算出現很奇怪的數字，比較器出現不明的bug。

解決方法:

R\_O\_n增加clk觸發，讓clk觸發時R\_O\_n會執行。

cal\_freq的計算條件變為讓R\_O\_n輸出的nHz觸發上緣，count就要加一。

利用process的if-else代替我們自己做的比較器的component做比較的動作。