Hw 2 40243118S 王擎天

1. 設計電路說明(20%):

這次的電路主要透過shift register來用1bit的訊號產生4 bits的訊號,其中第一個元素的訊號來自D\_in,其他皆來自前一個的訊號,這些訊號皆在clock event發生且為rising edge時來寫入.接下來4 bit訊號將移至multiplexer來選擇應輸出的3 bit訊號.

2. 程式碼說明(30%):

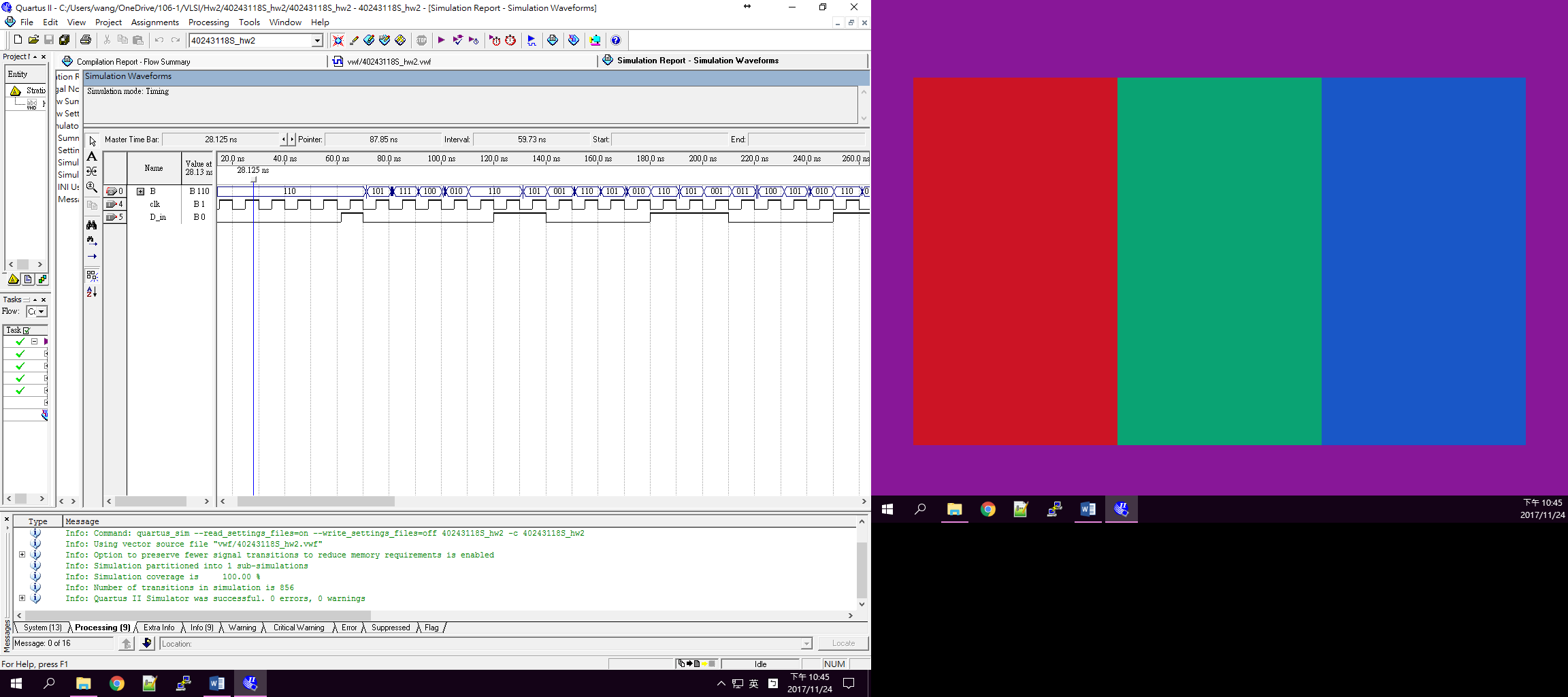
shift register我用4 bit的訊號組成,第一個元素的訊號來自D\_in,並利用for迴圈造成shift的現象,結果將輸入至訊號A,藉由訊號A來選擇適當的訊號輸出到B.

|  |
| --- |
| entity hw2 is  port(  D\_in,clk:in bit;  B:out bit\_vector(2 downto 0)  );  end entity hw2;  architecture hw2\_arch of hw2 is  signal reg:bit\_vector(3 downto 0):="0000";  signal A:bit\_vector(3 downto 0):="0000";  begin  process(clk,A,D\_in,reg) is  begin  if clk'EVENT and clk='1'  then  reg(0)<=D\_in;  for i in 3 downto 1 loop  reg(i)<=reg(i-1);  end loop;  end if;  A<=reg;  case A is  when "0000"=>B<="110";  when "0001"=>B<="101";  when "0010"=>B<="111";  when "0011"=>B<="001";  when "0100"=>B<="100";  when "0101"=>B<="101";  when "0110"=>B<="110";  when "0111"=>B<="011";  when "1000"=>B<="010";  when "1001"=>B<="000";  when "1010"=>B<="111";  when "1011"=>B<="001";  when "1100"=>B<="101";  when "1101"=>B<="010";  when "1110"=>B<="100";  when others => B<="111";  end case;  end process;  end architecture hw2\_arch; |

3. 模擬結果說明(30%):

紅框代表A的狀態

1.A:0000,0001,0010,0100,1000



0001

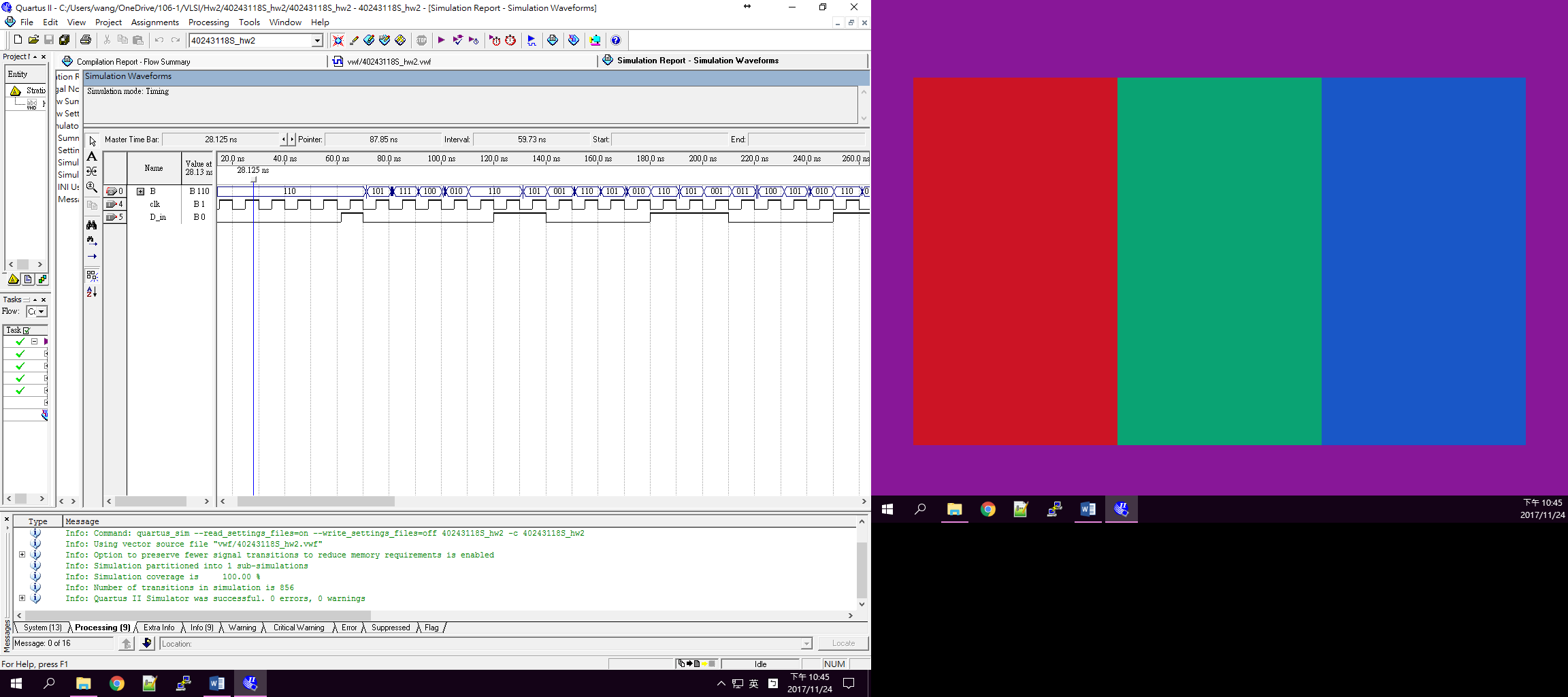
0000

0010

0100

1000

2.A:0011,0110,1100,0111,1110



0011

0001

0110

1100

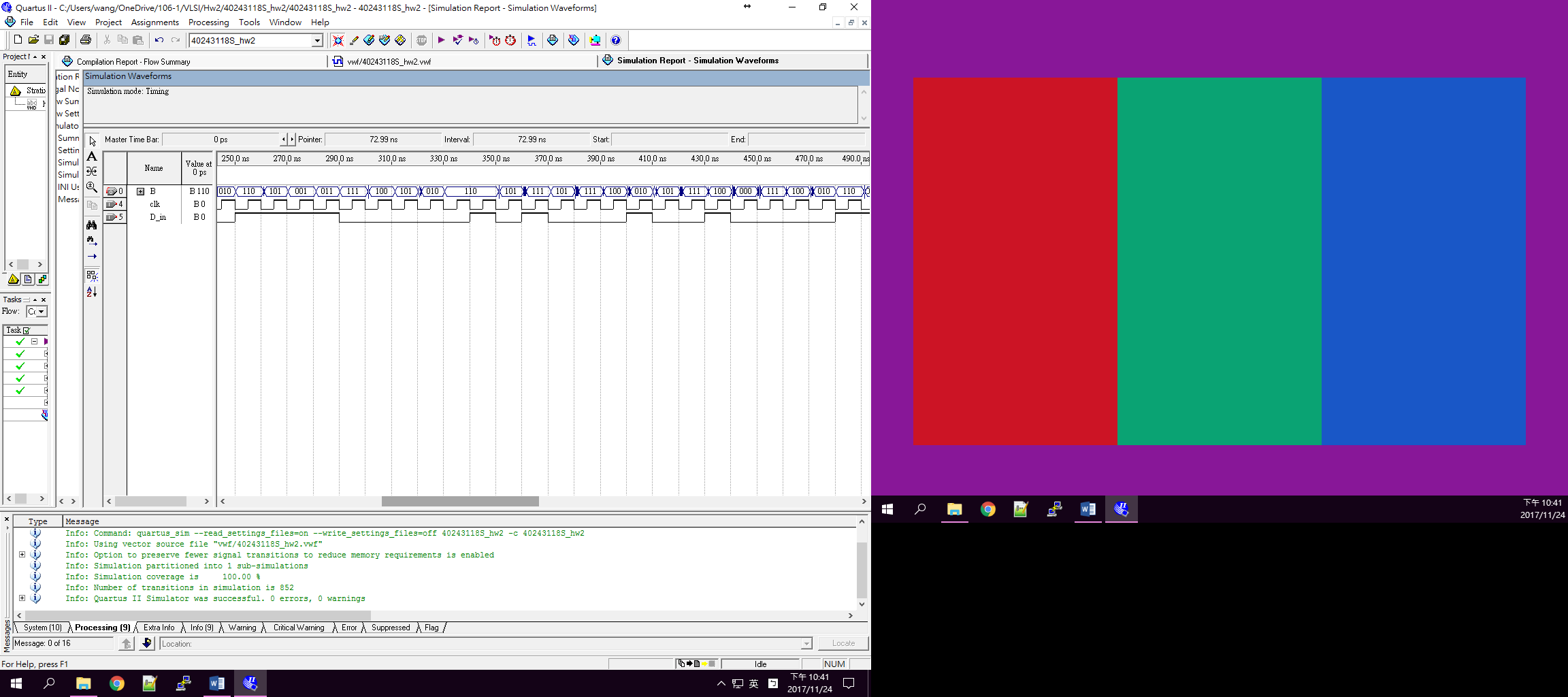
0001

0011

0111

1110

3.1111



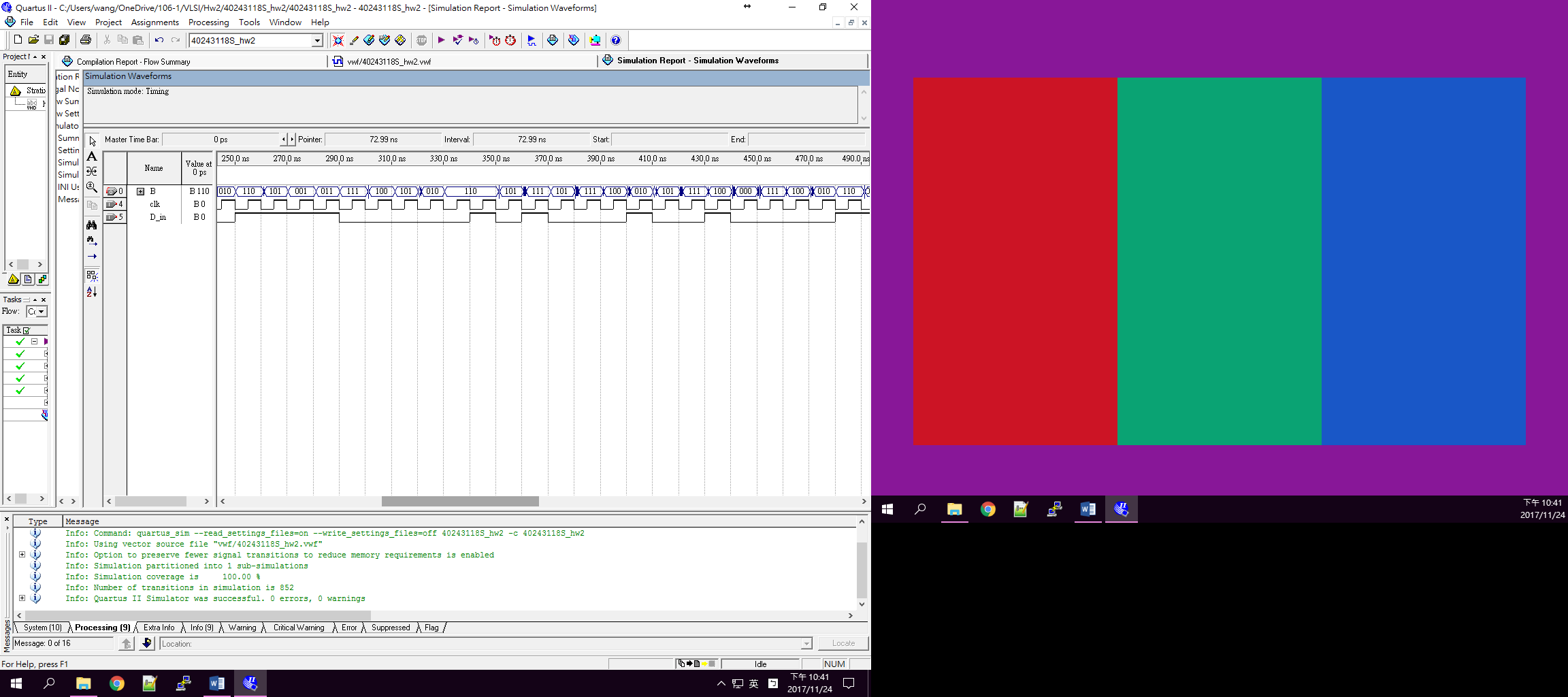
0001

0011

0111

1111

4.0101,1010



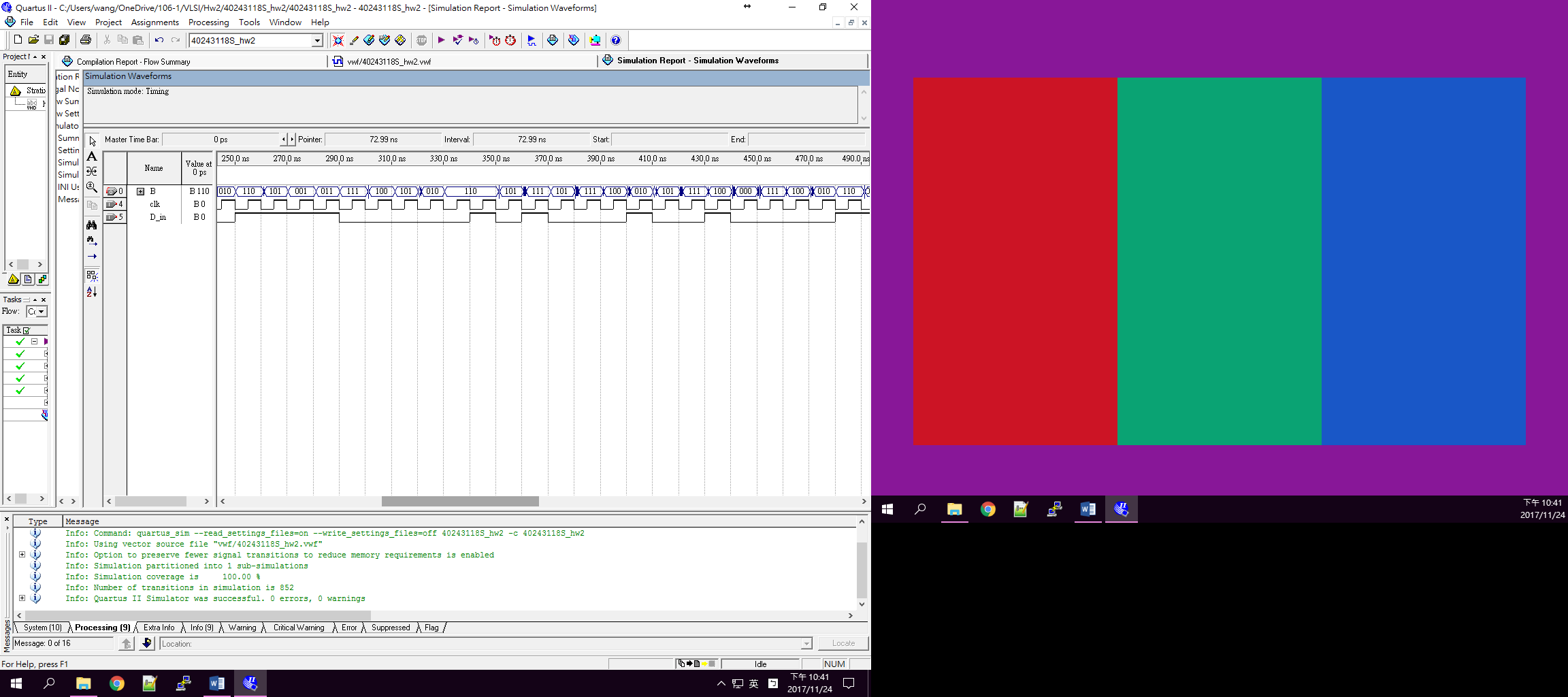
0001

0010

0101

1010

5.1001



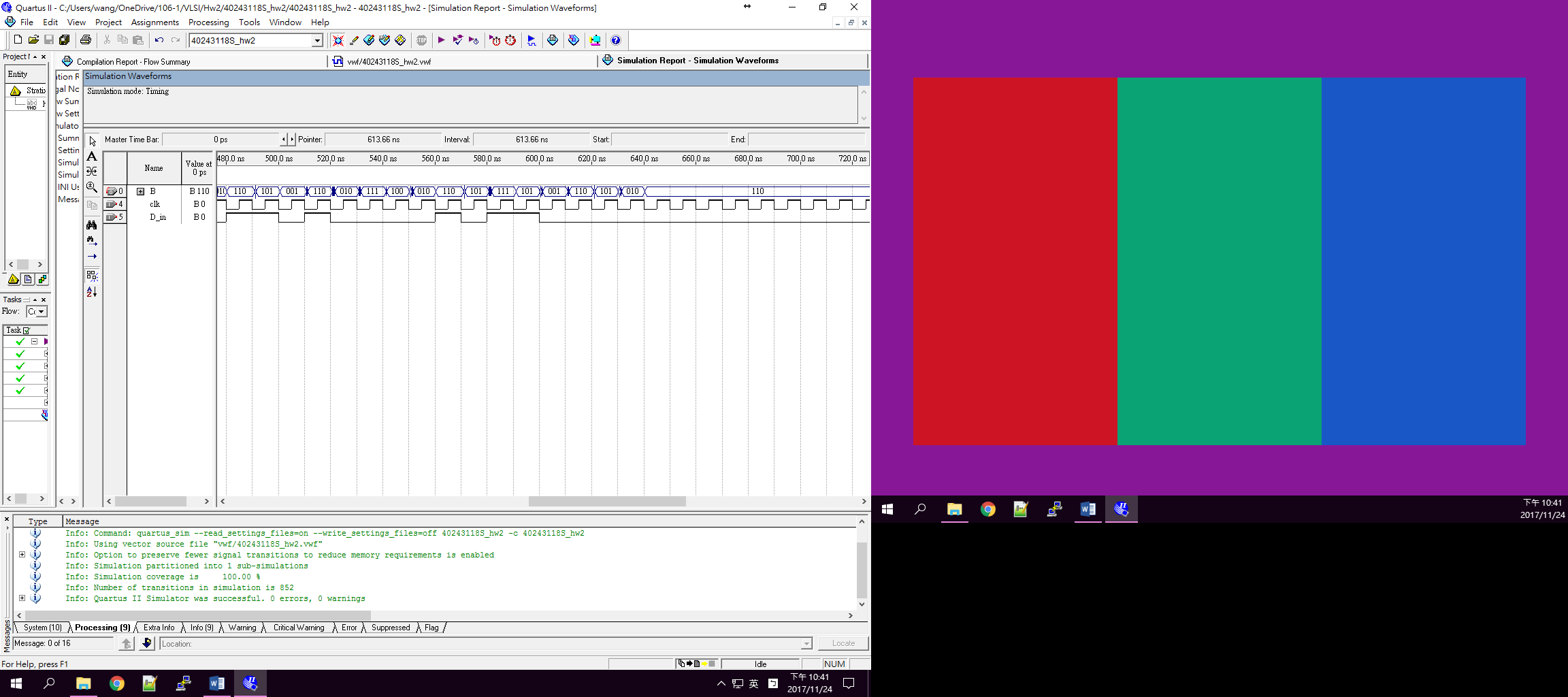
0001

0010

0100

1001

6.1101



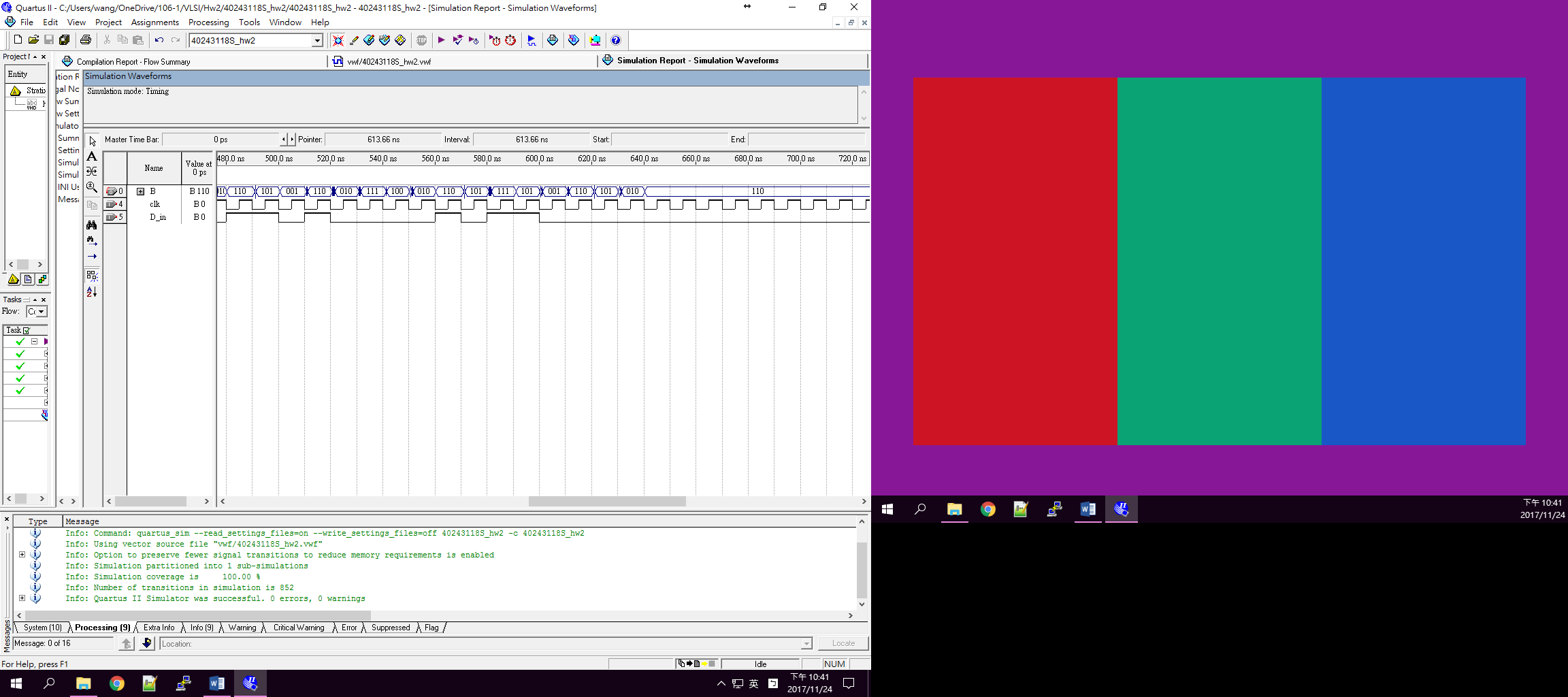
0001

0011

0110

1101

7.1011



0001

0010

0101

1011

4. 結論及心得(20%):

從上述的圖可以得知16種訊號A(紅色框)都有產生正確的3 bit訊號B,其中4 bit的訊號A確實可以透過1 bit的D\_in和for迴圈產生出來.這次的線路跟上次不同,這次多了register可以儲存狀態.藉由register的幫忙我可以將需要的狀態儲存起來以供未來使用,這將擴增我可以實做出來的線路種類.