Timing Circuit Designs and Their Applications

Homework 1
Controller using successive approximation

姓名：宋易柔（Yi-Rou, Song)

學號：114061529

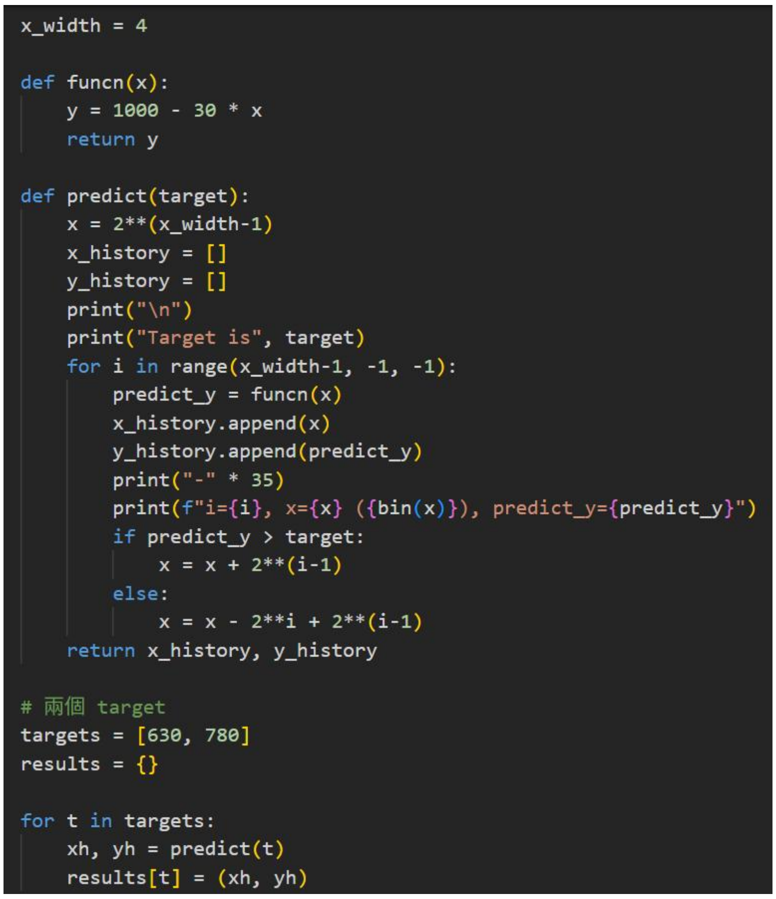
系所：電機系 (EE)

1. 題目說明

用連續逼近法(Successive approximation)鎖定特定的值。產生一個線性函數$y = 1000 - 30x$，x是 4 bit 的值，y是 10 bit 的值、$y \in [550,1000]$，設計電路找到適當的x使得對應的y (predict)會近似 target。

2. 軟體設計

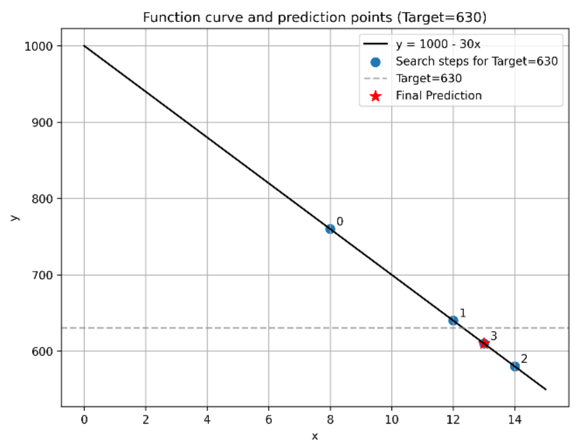
於 Google Colab 上編譯 python，以下是程式。

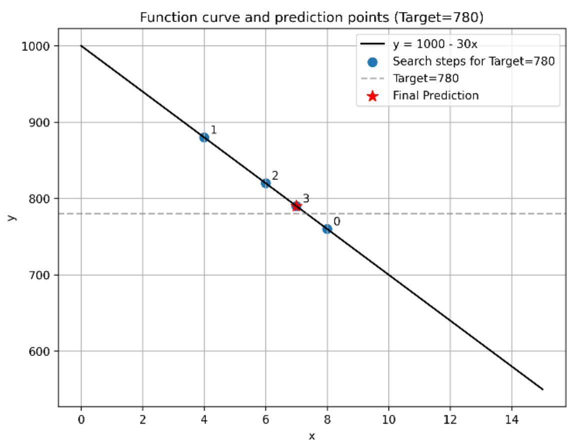


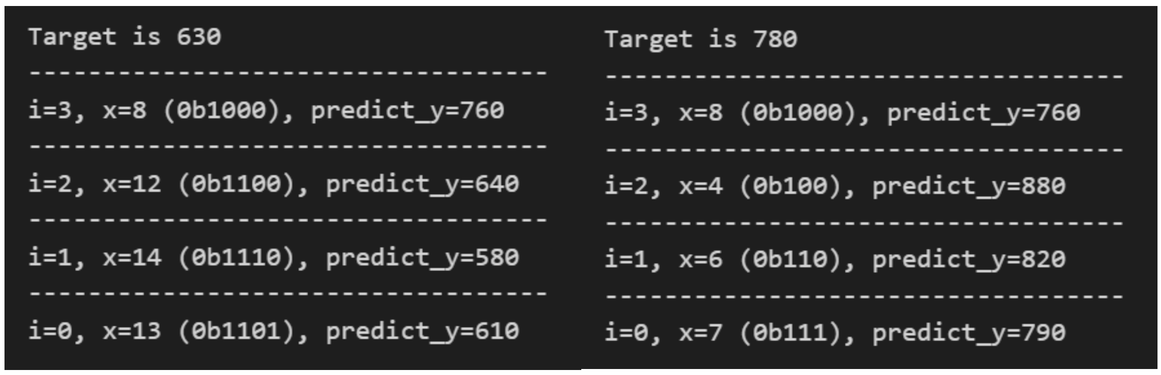
```python
x_width = 4

def funcn(x):
    y = 1000 - 30 * x
    return y

def predict(target):
    x = 2**(x_width-1)
    x_history = []
    y_history = []
    print("\n")
    print("Target is", target)
    for i in range(x_width-1, -1, -1):
        predict_y = funcn(x)
        x_history.append(x)
        y_history.append(predict_y)
        print("-" * 35)
        print(f"i={i}, x={x} ({bin(x)}), predict_y={predict_y}")
        if predict_y > target:
            x = x + 2**(i-1)
        else:
            x = x - 2**i + 2**(i-1)
    return x_history, y_history

# 兩個 target
targets = [630, 780]
results = {}

for t in targets:
    xh, yh = predict(t)
    results[t] = (xh, yh)
```

預測時，x的值由左至右決定，初始x設為1000_2即 4-bit value 的一半，將x代入線性函數得到的值為當次猜測的y，如果猜測的y大於目標的y，x就會在下一個 bit 位置設為 1；反之，x會讓現在 bit 位置的 1 變 0，並將下一個 bit 位置設為 1。

軟體輸出結果：







```
Target is 630
-----------------------------------
i=3, x=8 (0b1000), predict_y=760
-----------------------------------
i=2, x=12 (0b1100), predict_y=640
-----------------------------------
i=1, x=14 (0b1110), predict_y=580
-----------------------------------
i=0, x=13 (0b1101), predict_y=610
```

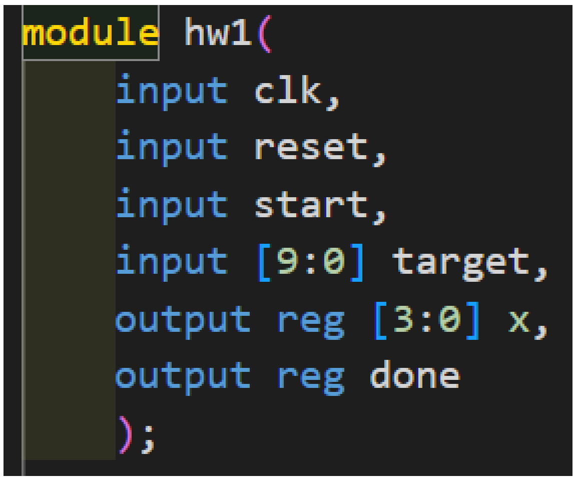
```
Target is 780
-----------------------------------
i=3, x=8 (0b1000), predict_y=760
-----------------------------------
i=2, x=4 (0b100), predict_y=880
-----------------------------------
i=1, x=6 (0b110), predict_y=820
-----------------------------------
i=0, x=7 (0b111), predict_y=790
```

請注意圖中的 0~3 數字標記代表是第幾次 iteration，下方文字輸出的 i 代表是哪一個 bit。圖中以紅色星星標記的為最終輸出結果。左圖中，即使第一個 iteration 預測的 y 是最接近 target 的，但此演算法依然會以最後一個 iteration 的輸出結果決定。

3. RTL 設計

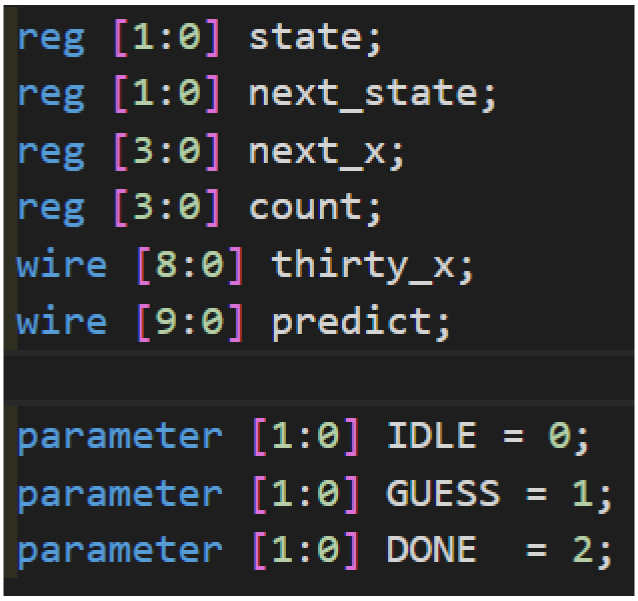
Signal Name	I/O	Width	Simple Description
reset	I	1	低準位非同步(active low asynchronous)之系統重置信號。
clk	I	1	時脈訊號。
start	I	1	start 拉起時表示開始預測。
target	I	10	目標值。
x	O	4	用以運算 predict。
done	O	1	當預測完成時，將 done 設為 high 表示完成。。

定義 I/O 名稱及寬度。



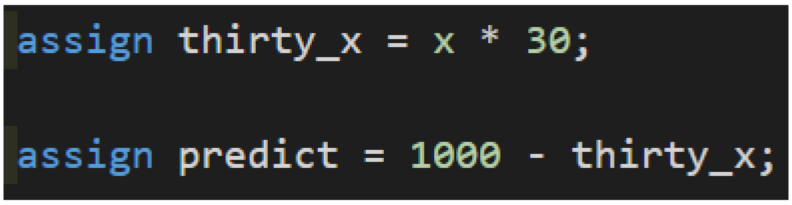
```
module hw1(
    input clk,
    input reset,
    input start,
    input [9:0] target,
    output reg [3:0] x,
    output reg done
    );
```

Registers and wires:
共有三個狀態 IDLE、GUESS、DONE，因此需要 2 的 state 與 next_state，count 是用來決定新的 x 值。



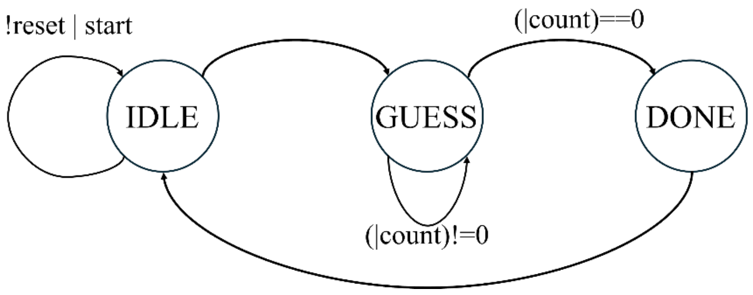
```
reg [1:0] state;
reg [1:0] next_state;
reg [3:0] next_x;
reg [3:0] count;
wire [8:0] thirty_x;
wire [9:0] predict;

parameter [1:0] IDLE = 0;
parameter [1:0] GUESS = 1;
parameter [1:0] DONE  = 2;
```

Linear function:
The wire “thirty_x” is x multiplied by 30 and the wire “predict” is correspond to the function of 1000-30x.

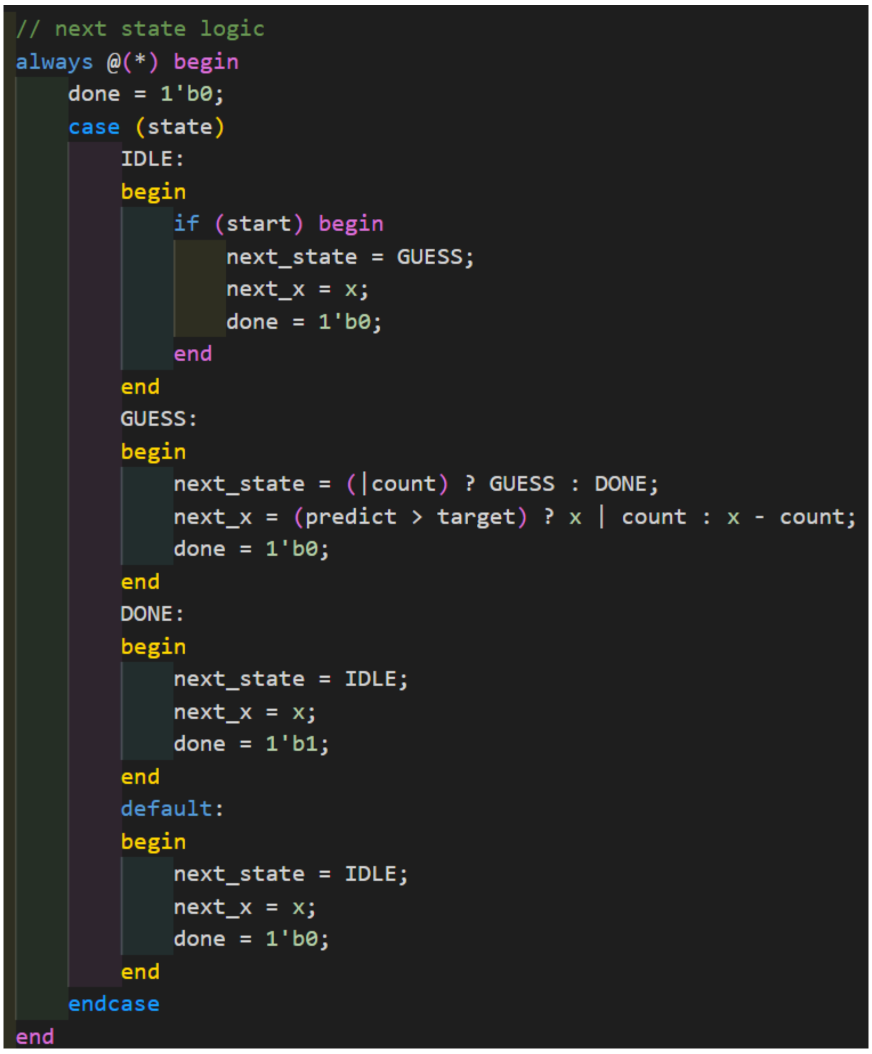


```
assign thirty_x = x * 30;

assign predict = 1000 - thirty_x;
```

FSM:

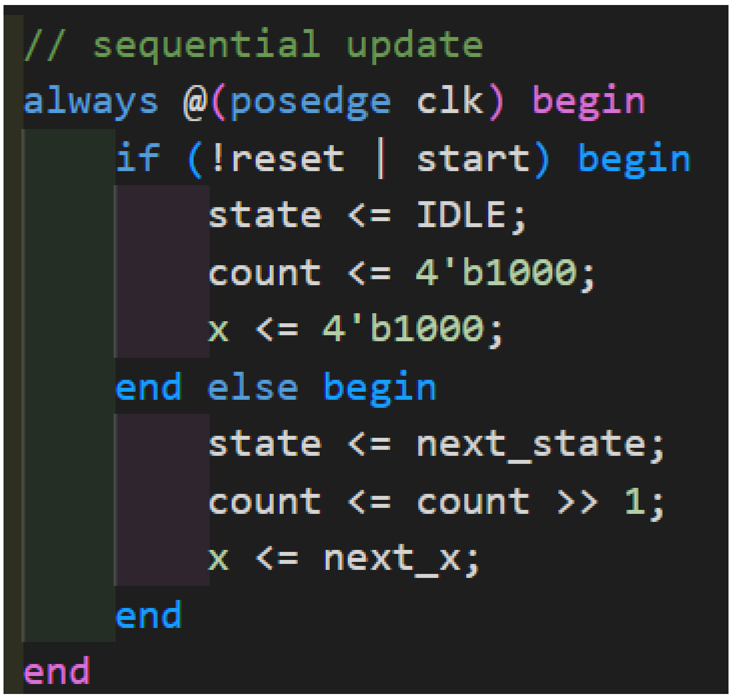


In IDLE state, get ready for the GUESS state.
In GUESS state, we use a mux to decide whether to overestimate or underestimate.
If it overestimates target, then bitwise or x and count. In this way, we can save the cost of an adder from x + count.
Otherwise, if it underestimates target, the next_x will be x substrate count (x – count). Here we'll have a substractor.
And if "count" down to 0, that represent all bits of x have been decided to make it the closest y.
In DONE state, pull up the" done" signal tell the user the output is valid.



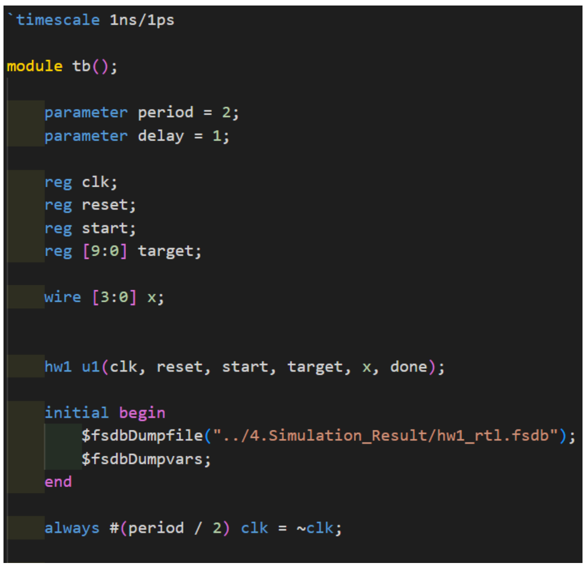
```
// next state logic
always @(*) begin
    done = 1'b0;
    case (state)
        IDLE:
        begin
            if (start) begin
                next_state = GUESS;
                next_x = x;
                done = 1'b0;
            end
        end
        GUESS:
        begin
            next_state = (|count) ? GUESS : DONE;
            next_x = (predict > target) ? x | count : x - count;
            done = 1'b0;
        end
        DONE:
        begin
            next_state = IDLE;
            next_x = x;
            done = 1'b1;
        end
        default:
        begin
            next_state = IDLE;
            next_x = x;
            done = 1'b0;
        end
    endcase
end
```

If reset is down or start is high, we reset the value and turn to IDLE state.
Every positive edge, count shift to right for 1 bit. For example, 1000 → 0100 → 0010 → 0001 → 0000 。

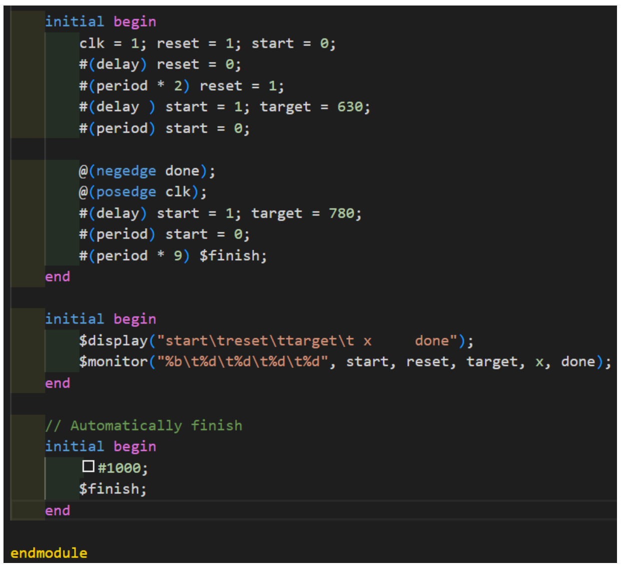


```
// sequential update
always @(posedge clk) begin
    if (!reset | start) begin
        state <= IDLE;
        count <= 4'b1000;
        x <= 4'b1000;
    end else begin
        state <= next_state;
        count <= count >> 1;
        x <= next_x;
    end
end
```

Testbench:



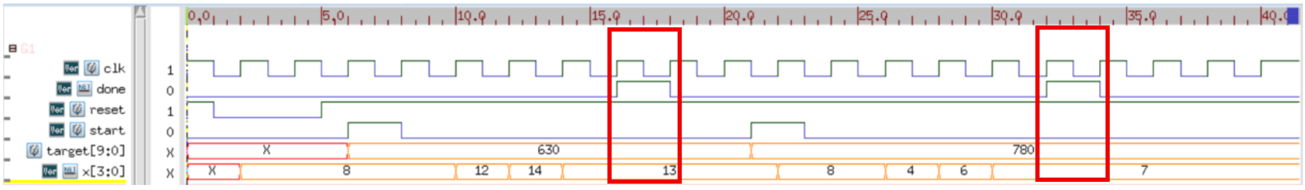
```
`timescale 1ns/1ps

module tb();

    parameter period = 2;
    parameter delay = 1;

    reg clk;
    reg reset;
    reg start;
    reg [9:0] target;

    wire [3:0] x;


    hw1 u1(clk, reset, start, target, x, done);

    initial begin
        $fsdbDumpfile("../4.Simulation_Result/hw1_rtl.fsdb");
        $fsdbDumpvars;
    end

    always #(period / 2) clk = ~clk;
```



```
    initial begin
        clk = 1; reset = 1; start = 0;
        #(delay) reset = 0;
        #(period * 2) reset = 1;
        #(delay ) start = 1; target = 630;
        #(period) start = 0;

        @(negedge done);
        @(posedge clk);
        #(delay) start = 1; target = 780;
        #(period) start = 0;
        #(period * 9) $finish;
    end

    initial begin
        $display("start\treset\ttarget\t x     done");
        $monitor("%b\t%d\t%d\t%d\t%d", start, reset, target, x, done);
    end

    // Automatically finish
    initial begin
        □#1000;
        $finish;
    end

endmodule
```

First we set a reset to low and then start to high, input the target with value 630.
After the done is high, we set start to high again, and the second input of target is 780. The done will turn to 1 again.

RTL simulation waveform:



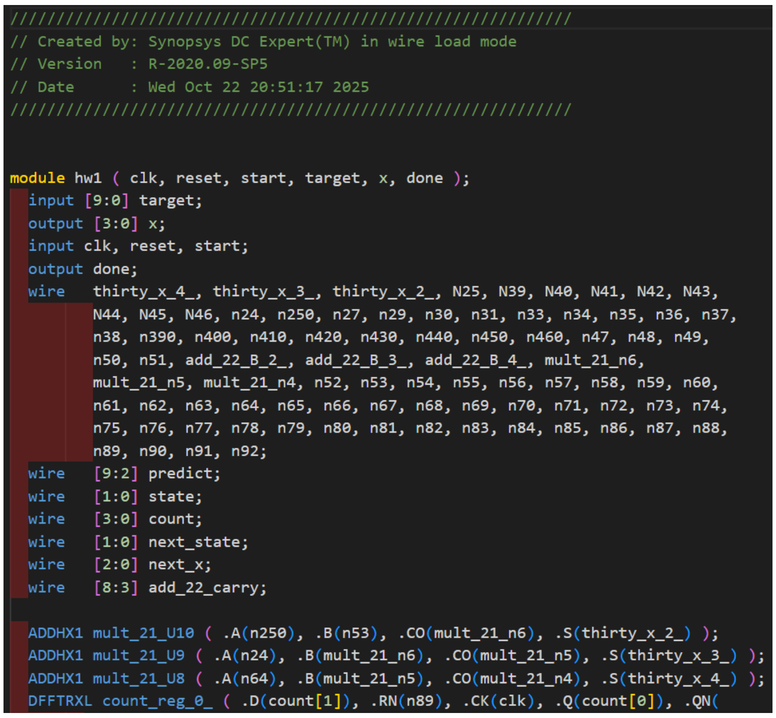
The above result is the same as the software output.

Final x[3:0] when target is 630 is 13, and the final x[3:0] when target is 780 is 7.

And every search of x is the same as the python output.

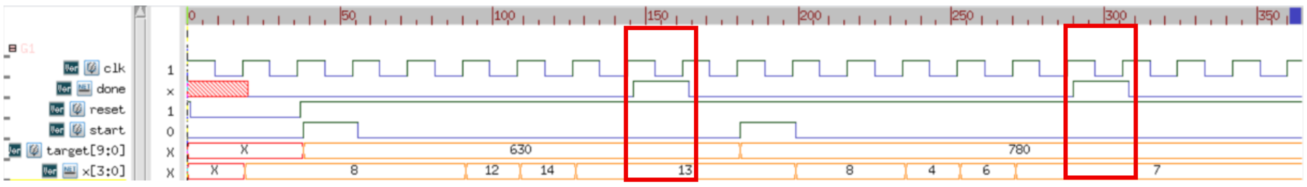
4. Synthesis and Report

Gate-level netlist:



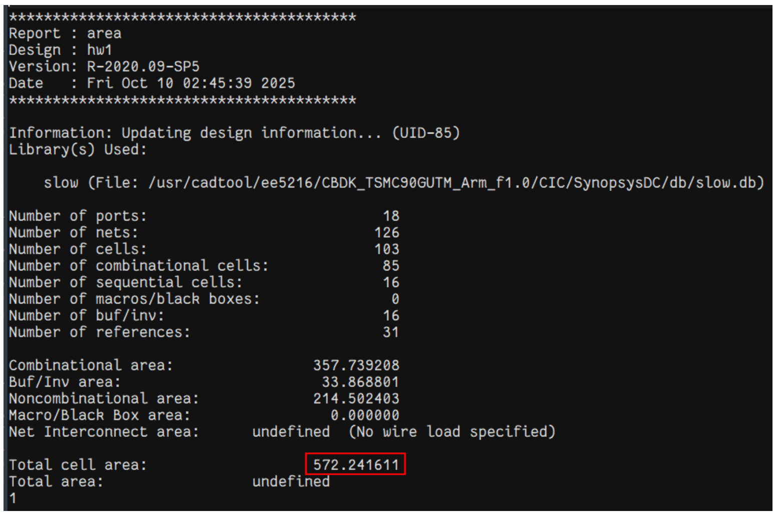
```
/////////////////////////////////////////////////////////////
// Created by: Synopsys DC Expert(TM) in wire load mode
// Version   : R-2020.09-SP5
// Date      : Wed Oct 22 20:51:17 2025
/////////////////////////////////////////////////////////////


module hw1 ( clk, reset, start, target, x, done );
  input [9:0] target;
  output [3:0] x;
  input clk, reset, start;
  output done;
  wire   thirty_x_4_, thirty_x_3_, thirty_x_2_, N25, N39, N40, N41, N42, N43,
         N44, N45, N46, n24, n250, n27, n29, n30, n31, n33, n34, n35, n36, n37,
         n38, n390, n400, n410, n420, n430, n440, n450, n460, n47, n48, n49,
         n50, n51, add_22_B_2_, add_22_B_3_, add_22_B_4_, mult_21_n6,
         mult_21_n5, mult_21_n4, n52, n53, n54, n55, n56, n57, n58, n59, n60,
         n61, n62, n63, n64, n65, n66, n67, n68, n69, n70, n71, n72, n73, n74,
         n75, n76, n77, n78, n79, n80, n81, n82, n83, n84, n85, n86, n87, n88,
         n89, n90, n91, n92;
  wire   [9:2] predict;
  wire   [1:0] state;
  wire   [3:0] count;
  wire   [1:0] next_state;
  wire   [2:0] next_x;
  wire   [8:3] add_22_carry;

  ADDHX1 mult_21_U10 ( .A(n250), .B(n53), .CO(mult_21_n6), .S(thirty_x_2_) );
  ADDHX1 mult_21_U9 ( .A(n24), .B(mult_21_n6), .CO(mult_21_n5), .S(thirty_x_3_) );
  ADDHX1 mult_21_U8 ( .A(n64), .B(mult_21_n5), .CO(mult_21_n4), .S(thirty_x_4_) );
  DFFTRXL count_reg_0_ ( .D(count[1]), .RN(n89), .CK(clk), .Q(count[0]), .QN(
```

合成完的波形:



輸出與 RTL waveform 比對結果一致。當 target 是 630 時，最終 x 是 13；當 target 是 780 時，最終 x 是 7。

Area report:



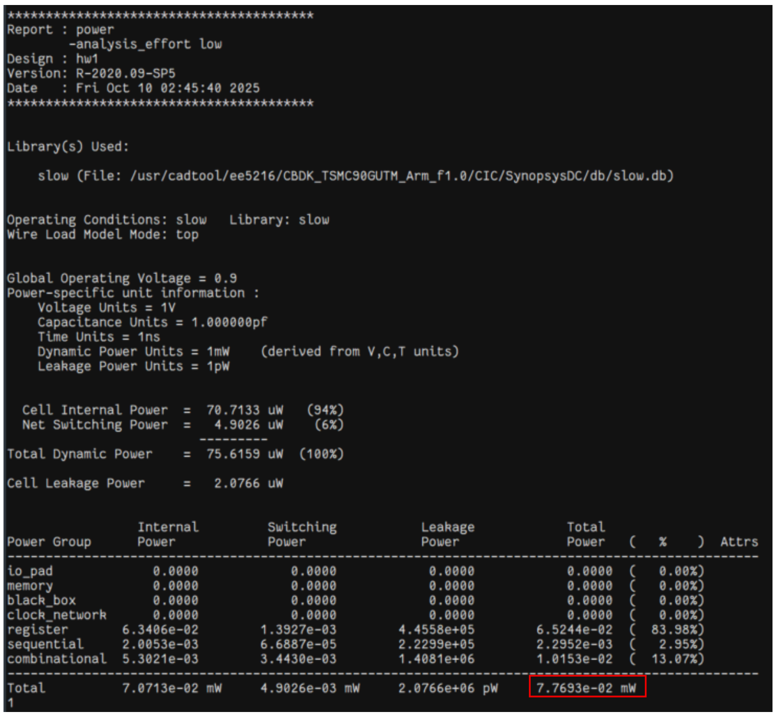
```
****************************************
Report : area
Design : hw1
Version: R-2020.09-SP5
Date   : Fri Oct 10 02:45:39 2025
****************************************

Information: Updating design information... (UID-85)
Library(s) Used:

    slow (File: /usr/cadtool/ee5216/CBDK_TSMC90GUTM_Arm_f1.0/CIC/SynopsysDC/db/slow.db)

Number of ports:                           18
Number of nets:                           126
Number of cells:                          103
Number of combinational cells:             85
Number of sequential cells:                16
Number of macros/black boxes:               0
Number of buf/inv:                         16
Number of references:                      31

Combinational area:                357.739208
Buf/Inv area:                       33.868801
Noncombinational area:             214.502403
Macro/Black Box area:                0.000000
Net Interconnect area:      undefined  (No wire load specified)

Total cell area:                   572.241611
Total area:                 undefined
1
```

Total cell area is **572.241611(μm^2).**

Area of 90 nm NAND2X1 is $2.52 \times 1.12\ (\mu m^2) = 2.8224\ (\mu m^2)$

Gate count is 572.241611÷2.8224=**202.75 of NAND2 gate**.

Power report:



```
****************************************
Report : power
        -analysis_effort low
Design : hw1
Version: R-2020.09-SP5
Date   : Fri Oct 10 02:45:40 2025
****************************************


Library(s) Used:

    slow (File: /usr/cadtool/ee5216/CBDK_TSMC90GUTM_Arm_f1.0/CIC/SynopsysDC/db/slow.db)


Operating Conditions: slow   Library: slow
Wire Load Model Mode: top


Global Operating Voltage = 0.9
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000pf
    Time Units = 1ns
    Dynamic Power Units = 1mW    (derived from V,C,T units)
    Leakage Power Units = 1pW


  Cell Internal Power  =  70.7133 uW   (94%)
  Net Switching Power  =   4.9026 uW    (6%)
                         ---------
Total Dynamic Power    =  75.6159 uW  (100%)

Cell Leakage Power     =   2.0766 uW


                 Internal         Switching           Leakage            Total
Power Group      Power            Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000            0.0000            0.0000            0.0000  (   0.00%)
memory             0.0000            0.0000            0.0000            0.0000  (   0.00%)
black_box          0.0000            0.0000            0.0000            0.0000  (   0.00%)
clock_network      0.0000            0.0000            0.0000            0.0000  (   0.00%)
register       6.3406e-02        1.3927e-03        4.4558e+05        6.5244e-02  (  83.98%)
sequential     2.0053e-03        6.6887e-05        2.2299e+05        2.2952e-03  (   2.95%)
combinational  5.3021e-03        3.4430e-03        1.4081e+06        1.0153e-02  (  13.07%)
--------------------------------------------------------------------------------------------------
Total          7.0713e-02 mW     4.9026e-03 mW     2.0766e+06 pW     7.7693e-02 mW
1
```

Internal power is $7.0713 \times 10^{-2}(mW)$. (94%)

Switching power is $4.9026 \times 10^{-3}(mW)$. (6%)

Leakage power is $2.0766 \times 10^{-3}(mW)$.

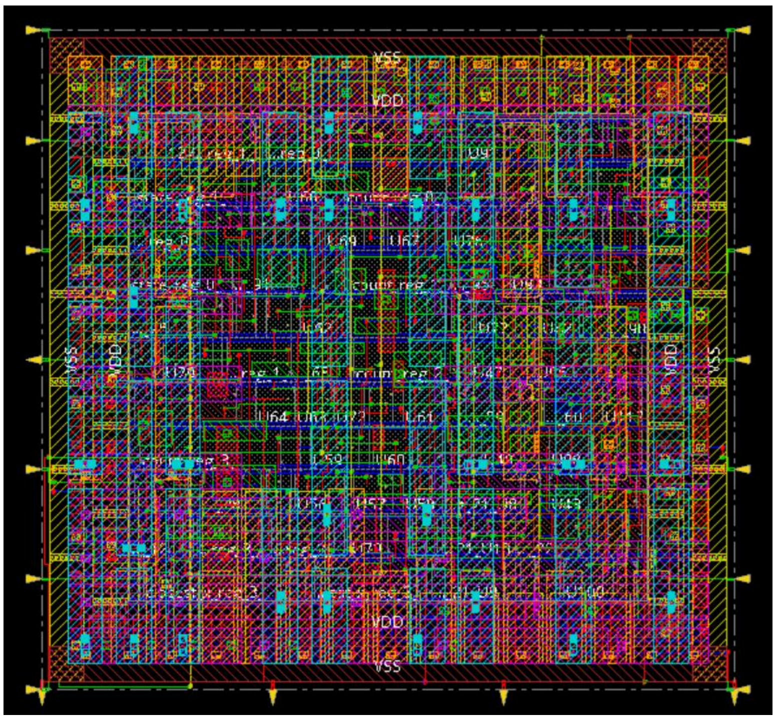
Total power is $\mathbf{7.7693 \times 10^{-2}}(\boldsymbol{mW})$.

Operating speed:

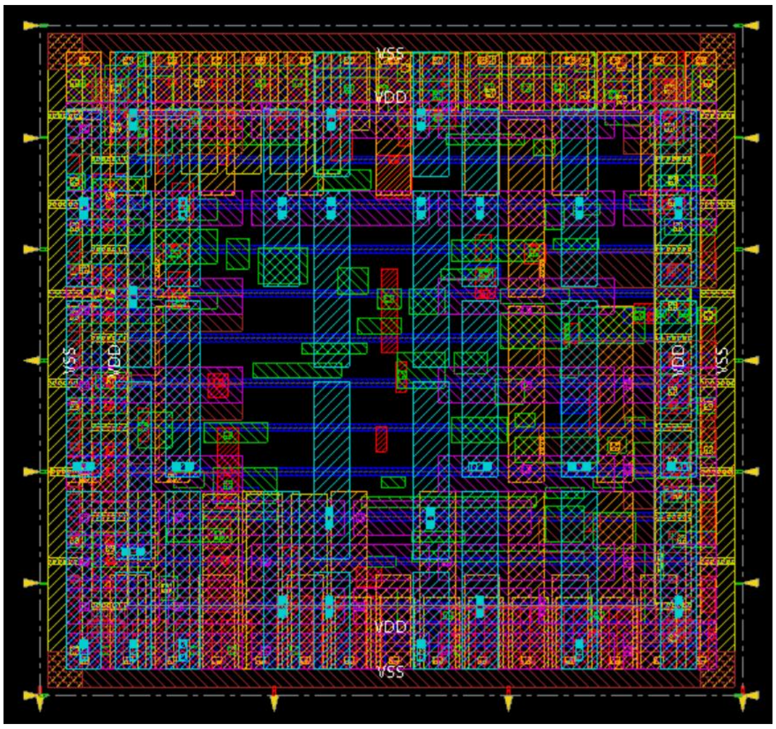
The maximum operating speed is $\frac{1}{18ns} = \mathbf{55.55MHz}$.

5. APR

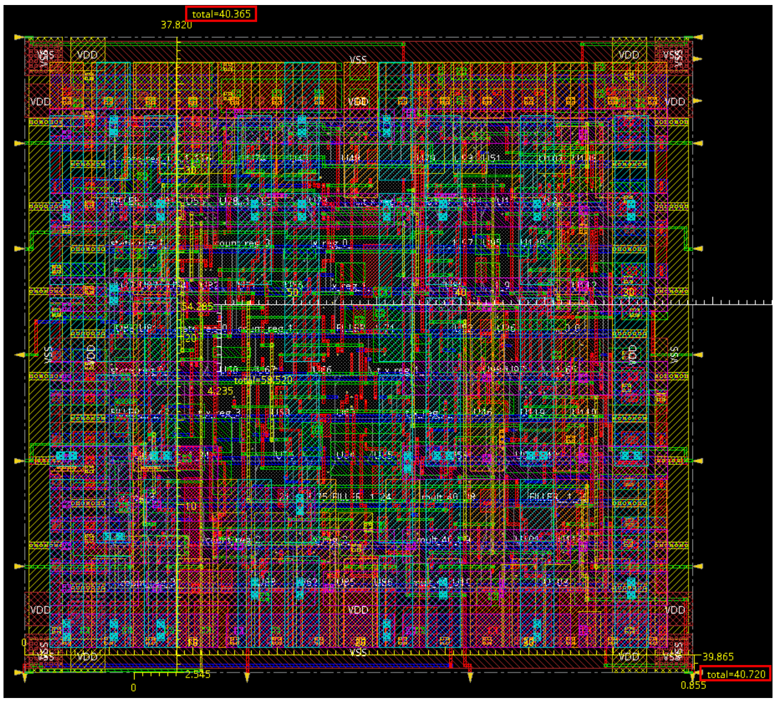
Physical view:



Floorplan view:

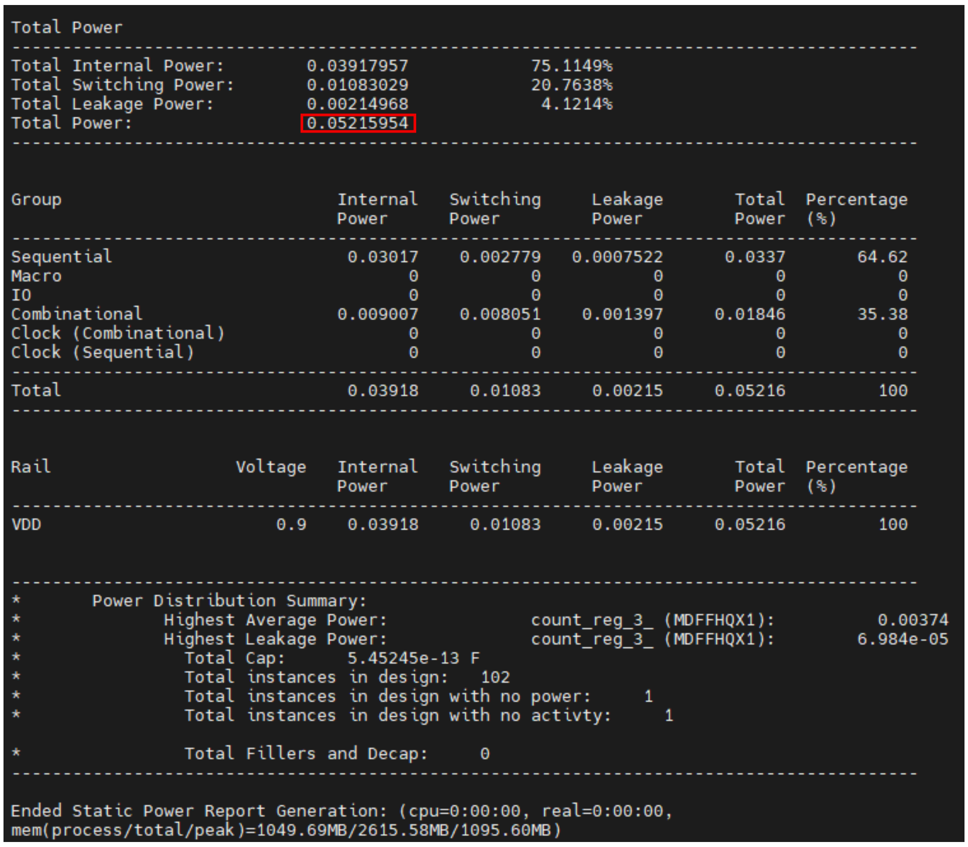


Measured Area:



Total area is $40.720(\mu m) \times 40.365(\mu m) = \mathbf{1643.6628(\mu m^2)}$.

Power report:



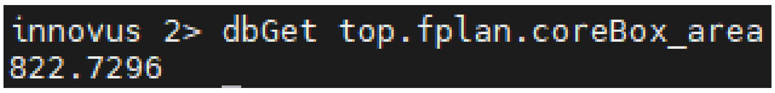
```
Total Power
-----------------------------------------------------------------------------------------
Total Internal Power:        0.03917957            75.1149%
Total Switching Power:       0.01083029            20.7638%
Total Leakage Power:         0.00214968             4.1214%
Total Power:                 0.05215954
-----------------------------------------------------------------------------------------


Group                           Internal   Switching     Leakage       Total  Percentage
                                Power      Power         Power         Power  (%)
-----------------------------------------------------------------------------------------
Sequential                       0.03017    0.002779   0.0007522      0.0337       64.62
Macro                                  0           0           0           0           0
IO                                     0           0           0           0           0
Combinational                   0.009007    0.008051    0.001397     0.01846       35.38
Clock (Combinational)                  0           0           0           0           0
Clock (Sequential)                     0           0           0           0           0
-----------------------------------------------------------------------------------------
Total                            0.03918     0.01083     0.00215     0.05216         100
-----------------------------------------------------------------------------------------


Rail                  Voltage   Internal   Switching     Leakage       Total  Percentage
                                Power      Power         Power         Power  (%)
-----------------------------------------------------------------------------------------
VDD                       0.9    0.03918     0.01083     0.00215     0.05216         100


-----------------------------------------------------------------------------------------
*       Power Distribution Summary:
*              Highest Average Power:          count_reg_3_ (MDFFHQX1):        0.00374
*              Highest Leakage Power:          count_reg_3_ (MDFFHQX1):      6.984e-05
*                Total Cap:      5.45245e-13 F
*                Total instances in design:   102
*                Total instances in design with no power:     1
*                Total instances in design with no activty:     1

*                Total Fillers and Decap:     0
-----------------------------------------------------------------------------------------

Ended Static Power Report Generation: (cpu=0:00:00, real=0:00:00,
mem(process/total/peak)=1049.69MB/2615.58MB/1095.60MB)
```

Internal power is $0.03917957(mW)$.
Switching power is $0.01083029(mW)$.
Leakage power is $0.00214968(mW)$.
Total power is $\mathbf{0.05215954(mW)}$.

Operating speed:

The maximum operating speed is $\frac{1}{2ns} = \mathbf{500MHz}$.

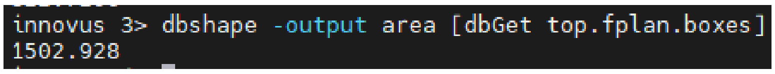
Core area:



```
innovus 2> dbGet top.fplan.coreBox_area
822.7296
```

Core area is $\mathbf{822.7296}$ $(\boldsymbol{\mu m^2})$.

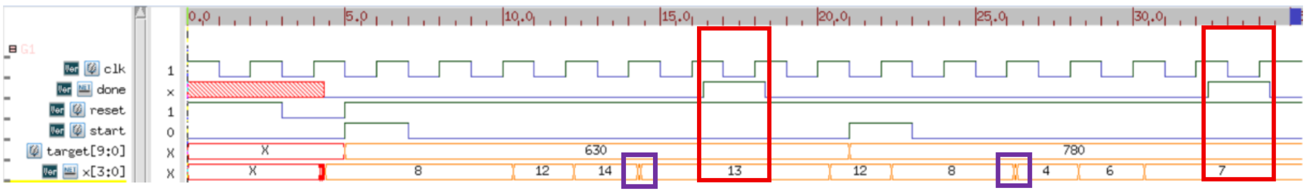
Chip area:



```
innovus 3> dbshape -output area [dbGet top.fplan.boxes]
1502.928
```

Chip area is $\mathbf{1502.928}$ $(\boldsymbol{\mu m^2})$.

Post-layout simulation waveform:



The result is the same as the previous two simulations.

Compare:

Compare the difference between post-layout simulation and two pre-layout simulation waveforms.

In the waveform of synthesized gate-level netlist and post-layout simulation, the “done” signal is still unknown before reset, also these two have glitch when signal values change.

We can see in purple blocks marked on the above figure, from 14 to 13 and from 8 to 4, there are some other values between them. These are the transition of x, and they’re not stable in these intervals. In realistic, we hope these intervals are as small as possible.