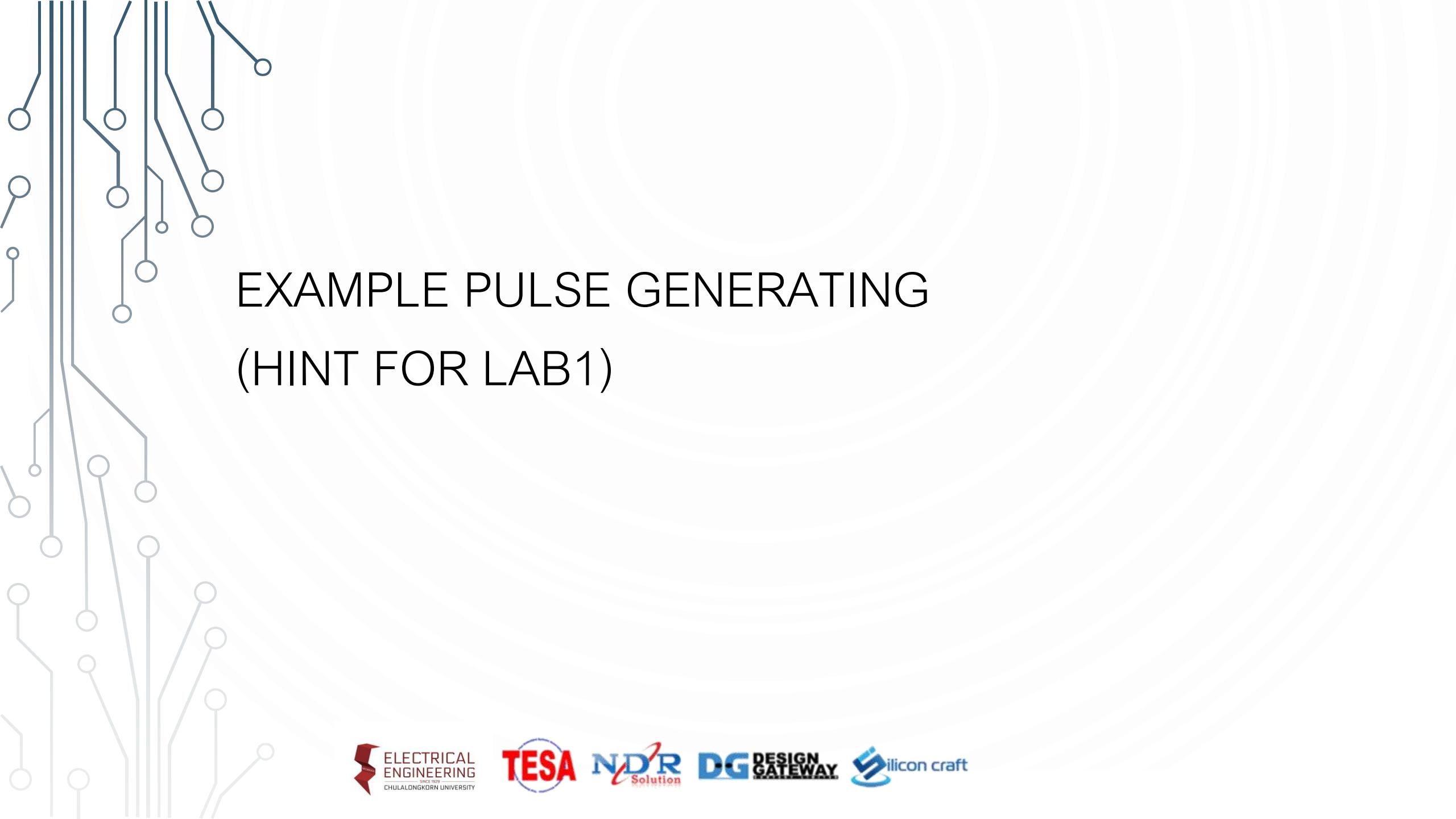


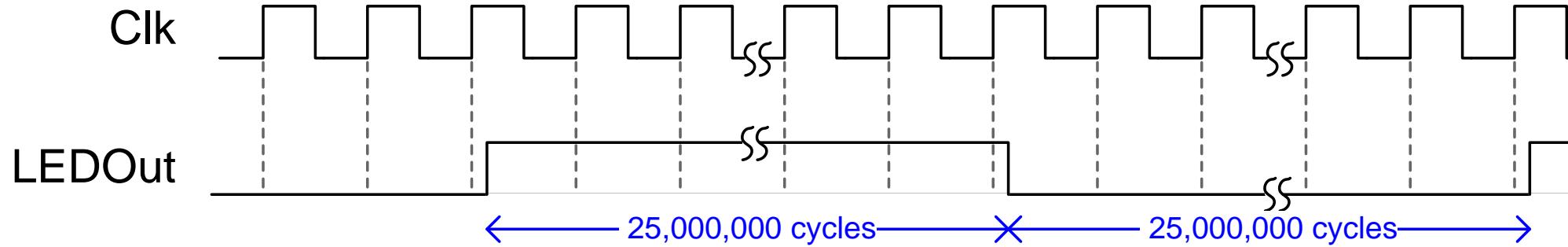
# DIGITAL DESIGN WITH FPGA CAMP

DAY 1 LOGIC DESIGN LAB



## EXAMPLE PULSE GENERATING (HINT FOR LAB1)

## EXAMPLE PROBLEM



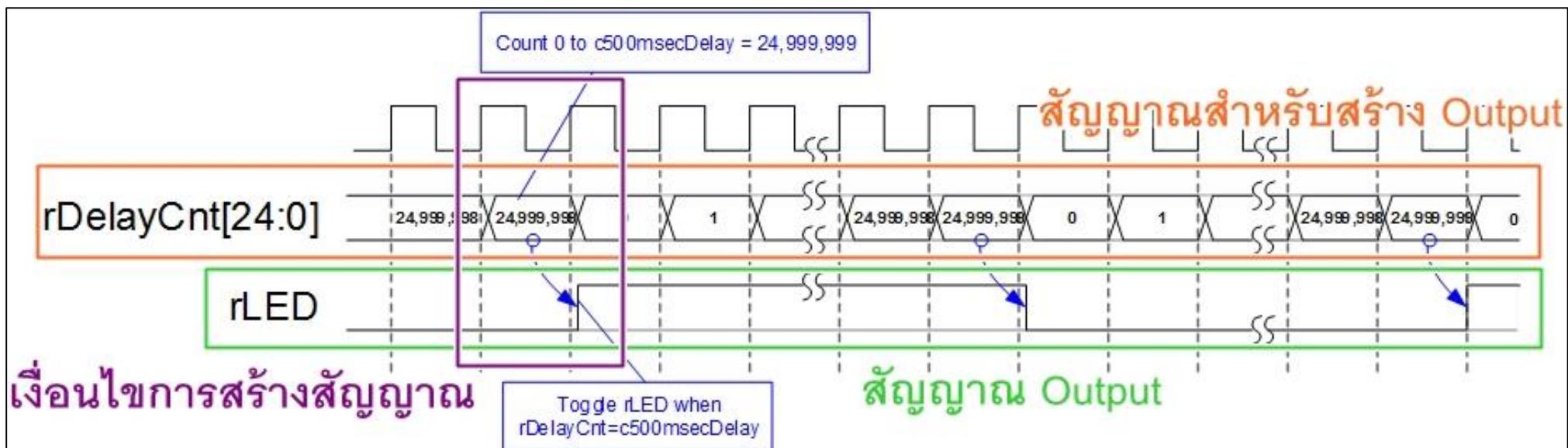
สร้างสัญญาณ OUTPUT ชื่อ LEDOut 送ไปที่หลอดไฟ LED เพื่อสร้างไฟกระพริบที่มีความเวลา 1 วินาที

- เนื่องจากสัญญาณนาฬิกาของระบบที่ใช้มีความถี่ 50 MHz ดังนั้นความถี่ของสัญญาณที่ต้องการจึงกว้าง 50,000,000 CLOCK
- ไฟกระพริบที่ออกแบบมี DUTY CYCLE เท่ากับ 50% (มีค่าเป็น '0' และ '1' อย่างละครึ่งควบเท่ากัน) ดังนั้น LEDOut จะ toggle เป็นครั้งๆ ๆ 25,000,000 CLOCK

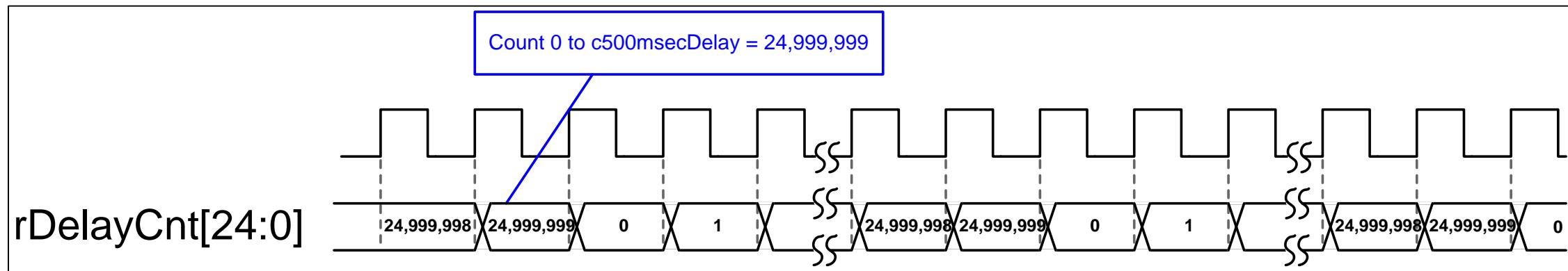
## STEP1 : DESIGN SIGNAL AND TIMING DIAGRAM

การออกแบบจึงประกอบด้วย 2 สัญญาณ ได้แก่

- 1) วงจร Counter จับเวลา 0.5 วินาทีเพื่อใช้ในการ toggle สัญญาณ LED ออกไป ตั้งชื่อว่า rDelayCnt และจำนวน bit ที่ต้องใช้สำหรับรองรับ 25,000,000 คือ 25 bit
- 2) สัญญาณ LED ที่จะสร้าง ที่จะสลับค่าระหว่าง 0 และ 1 ตั้งชื่อว่า rLED และค่อยเชื่อมสัญญาณ rLED ออกไปที่ Output



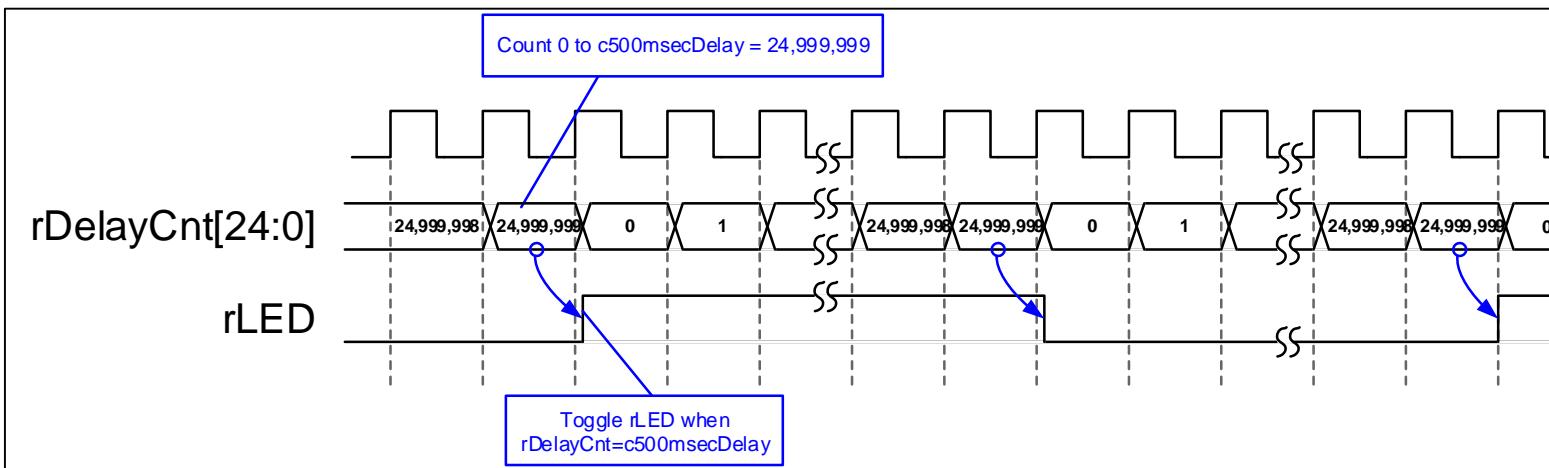
## STEP2 : HDL CODING FOLLOWING TIMING DIAGRAM (1)



```
u_rDelayCnt : Process (Clk50) Is
Begin
  if ( rising_edge(Clk50) ) then
    if ( rSysRstB='0' ) then
      rDelayCnt(24 downto 0) <= (others=>'0');
    else
      if ( rDelayCnt(24 downto 0)=c500msecDelay(24 downto 0) ) then
        rDelayCnt(24 downto 0) <= (others=>'0');
      else
        rDelayCnt(24 downto 0) <= rDelayCnt(24 downto 0) + 1;
      end if;
    end if;
  end if;
End Process u_rDelayCnt;
```

c500msecDelay นั้นเป็นค่าคงที่ (constant) ที่ประกาศไว้ใน Code ในช่วงประกาศสัญญาณ การประกาศเป็น constant จะทำให้สามารถแก้ไขค่าได้ง่ายเมื่อระบบเปลี่ยน spec เช่น ถ้าเราต้องการนี้ไปทำงานที่ Clock อื่น ๆ เราสามารถสร้าง Counter 500 msec ได้โดยเพียงค่า constant ด้านบนเท่านั้น

## STEP2 : HDL CODING FOLLOWING TIMING DIAGRAM (2)

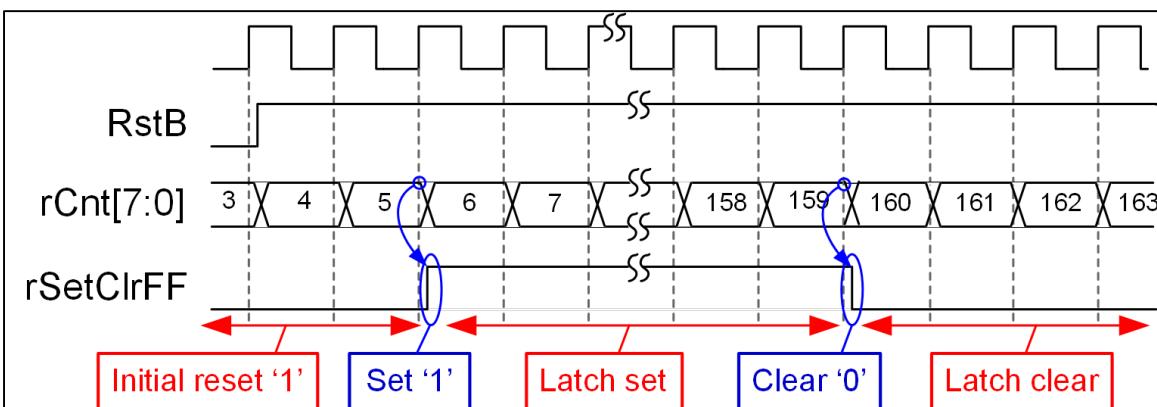


```
u_rLED : Process (Clk50) Is
Begin
  if ( rising_edge(Clk50) ) then
    if ( rSysRstB='0' ) then
      rLED    <= '0';
    else
      if ( rDelayCnt(24 downto 0)=c500msecDelay(24 downto 0) ) then
        rLED    <= not rLED;
      else
        rLED    <= rLED;
      end if;
    end if;
  end if;
End Process u_rLED;
```

ในจังหวะที่ใช้ Clear ค่า rDelayCnt ให้กลับเป็น '0' จะเป็น  
จังหวะเดียวกับที่สัญญาณ rLED นั้น toggle ค่า ดังนั้นเงื่อนไข<sup>6</sup>  
สำหรับ Clear ค่า rDelayCnt กับ toggle สัญญาณ rLED จึง<sup>6</sup>  
เหมือนกัน

# EXAMPLE TO GENERATE PULSE WITH SPECIFIC PERIOD

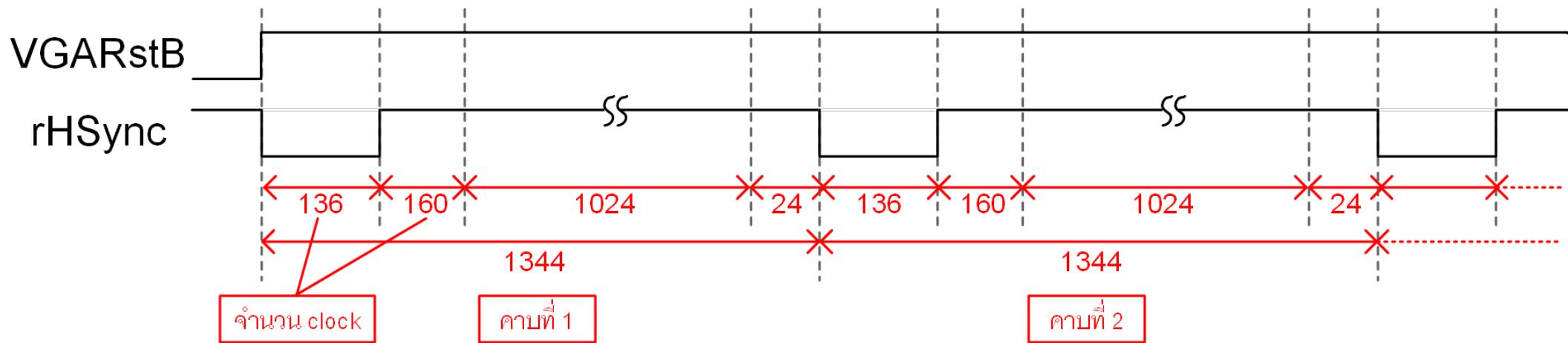
ตัวอย่างการสร้างสัญญาณ Pulse ที่มีความกว้างตามขนาดที่เราต้องการ โดยใช้ Counter เป็นตัวให้จังหวะการ set/clear ค่าของสัญญาณ



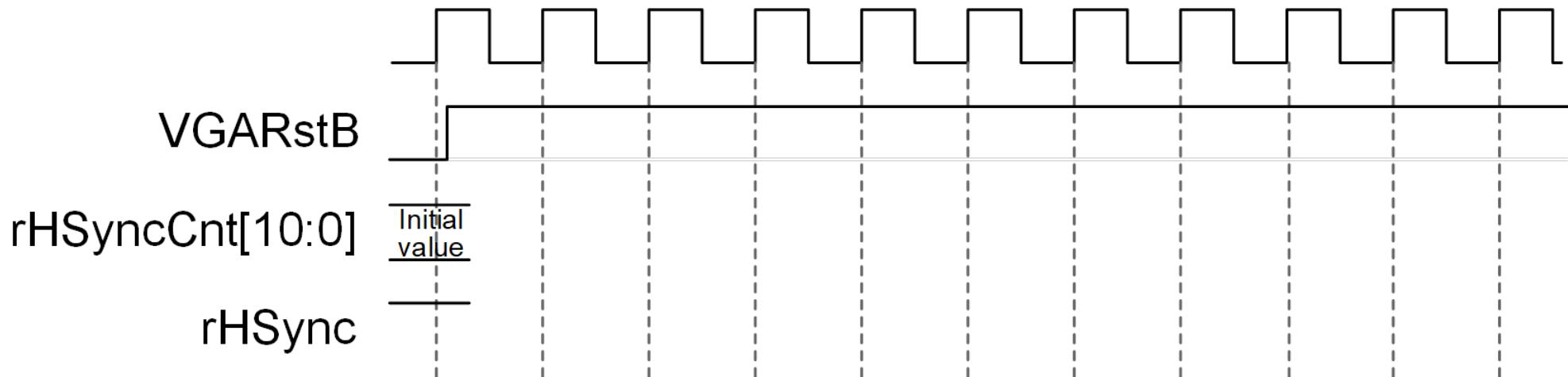
```
u_SetClrFF : Process (Clk) Is
Begin
  if ( rising_edge(Clk) ) then
    if ( RstB='0' ) then
      rSetClrFF  <= '0'; Initial reset
    else
      if ( rCnt(7 downto 0)=x"5" ) then Set '1'
        rSetClrFF  <= '1';
      elsif ( rCnt(7 downto 0)=x"9F" ) then Clear '0'
        rSetClrFF  <= '0';
      else
        rSetClrFF  <= rSetClrFF; Latch
      end if;
    end if;
  end if;
End Process u_SetClrFF;
```

# LAB1: HSYNC TIMING DIAGRAM AND CODING

# LAB1 : OUTPUT SPECIFICATION : HSYNC



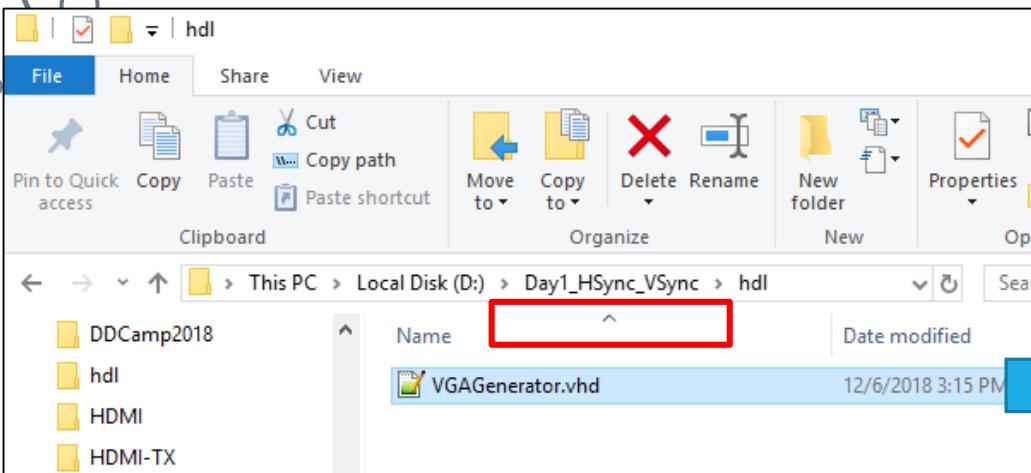
# LAB1 : DRAW TIMING DIAGRAM OF HSYNC



## คำแนะนำ

- สร้าง Counter ชื่อ rHSyncCnt มาช่วยนับจังหวะเพื่อสร้าง rHSync
- วาด Timing Diagram โดยย่อส่วน/ข้ามค่าของ Counter ที่ไม่สำคัญ (ไม่ใช่จังหวะที่สัญญาณ rHSync มีการเปลี่ยนแปลงค่า) เพื่อให้รูปสามารถ fit ลงในหน้ากระดาษได้

# LAB1 : CODING HSYNC



1) เปิดโปรเจ็ค Day1\_Hsync\_Vsync เข้า hdl folder  
แล้วแก้ไขไฟล์ VGAGenerator.vhd

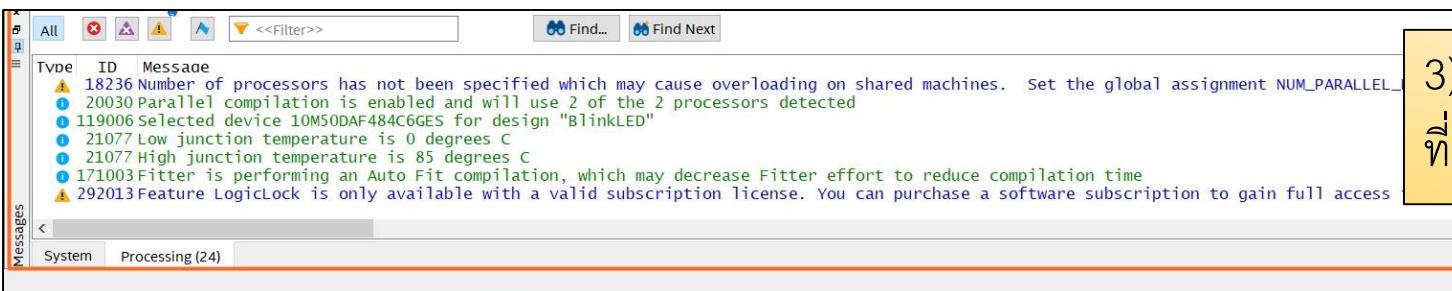
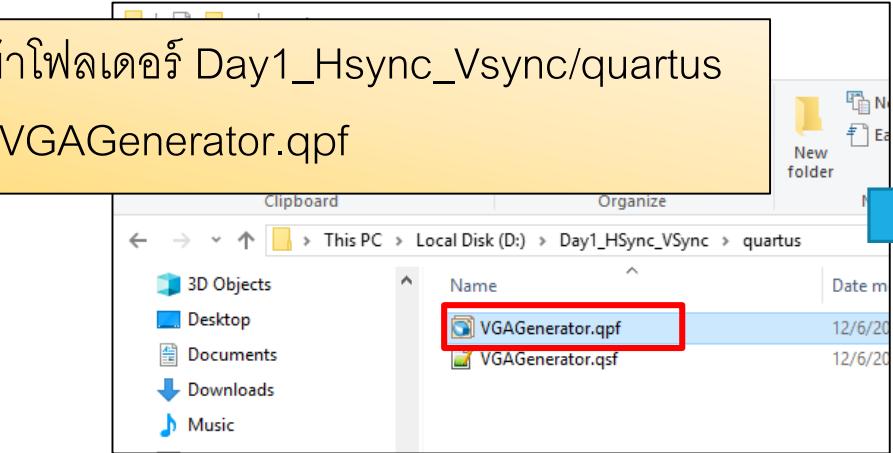
```
71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99  
-- [[ LBA 1 : CODING HSYNC(1) ]]  
  
u_rHSyncCnt : Process (VGAClk) Is  
Begin  
  if ( rising_edge(VGAClk) ) then  
    if ( VGARstB='0' ) then  
      -- coding here  
      -- initial value  
    else  
      -- coding here  
      -- behaviour  
    end if;  
  end if;  
End Process u_rHSyncCnt;  
  
u_rHSync : Process (VGAClk) Is  
Begin  
  if ( rising_edge(VGAClk) ) then  
    if ( VGARstB='0' ) then  
      -- coding here  
      -- initial value  
    else  
      -- coding here  
      -- behaviour  
    end if;  
  end if;  
End Process u_rHSync;
```

2) ภายใน VGAGenerator.vhd ให้เขียน code  
สำหรับสัญญาณ rHSyncCnt และ rHSync ใน  
Process u\_rHSyncCnt และ u\_rHSync  
ตามลำดับ

# LAB1 : ANALYSIS AND SYNTHESIS

เมื่อเขียน code เสร็จแล้วทํางานตรวจสอบ Syntax Error ด้วยการ Synthesis ใน Quartus

1) เข้าโฟลเดอร์ Day1\_Hsync\_Vsync/quartus  
เปิด VGAGenerator.qpf

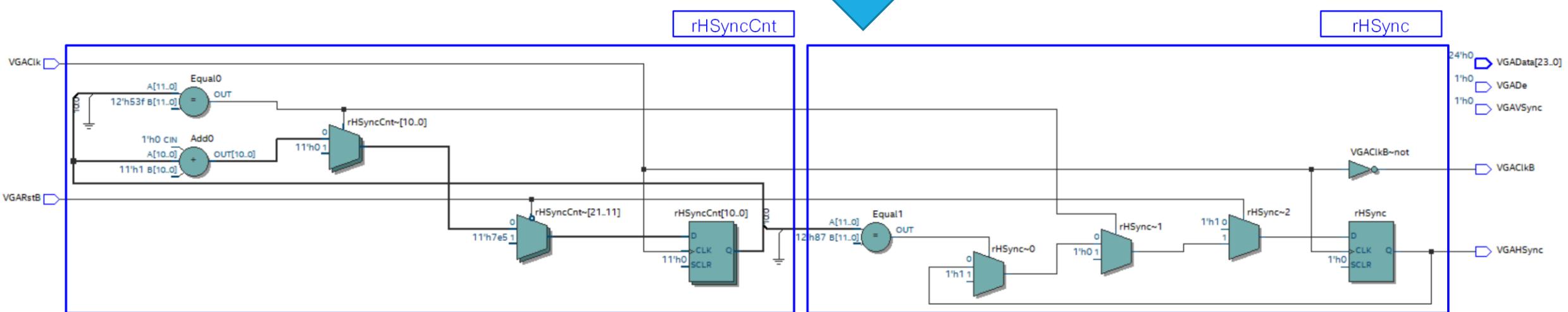
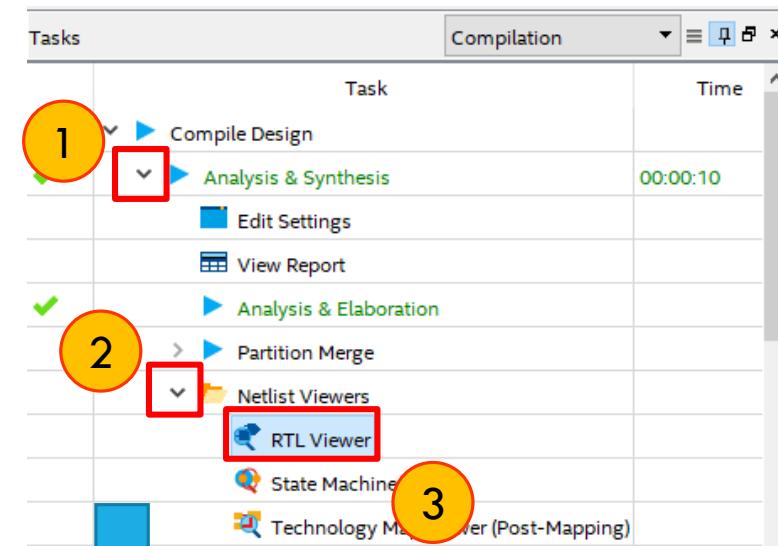


2) ที่ด้านซ้ายของโปรแกรมกด Analysis & Synthesis

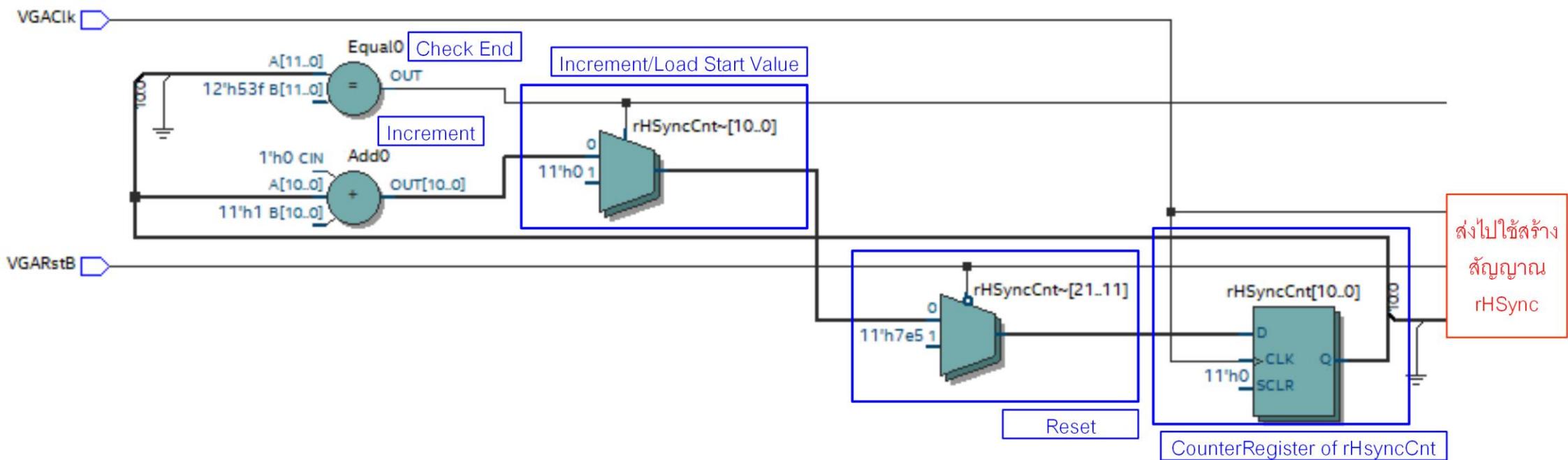
3) ตรวจสอบ error หรือ warning  
ที่หน้าต่าง message ด้านล่าง

# LAB1 : RTL VIEWER RESULT

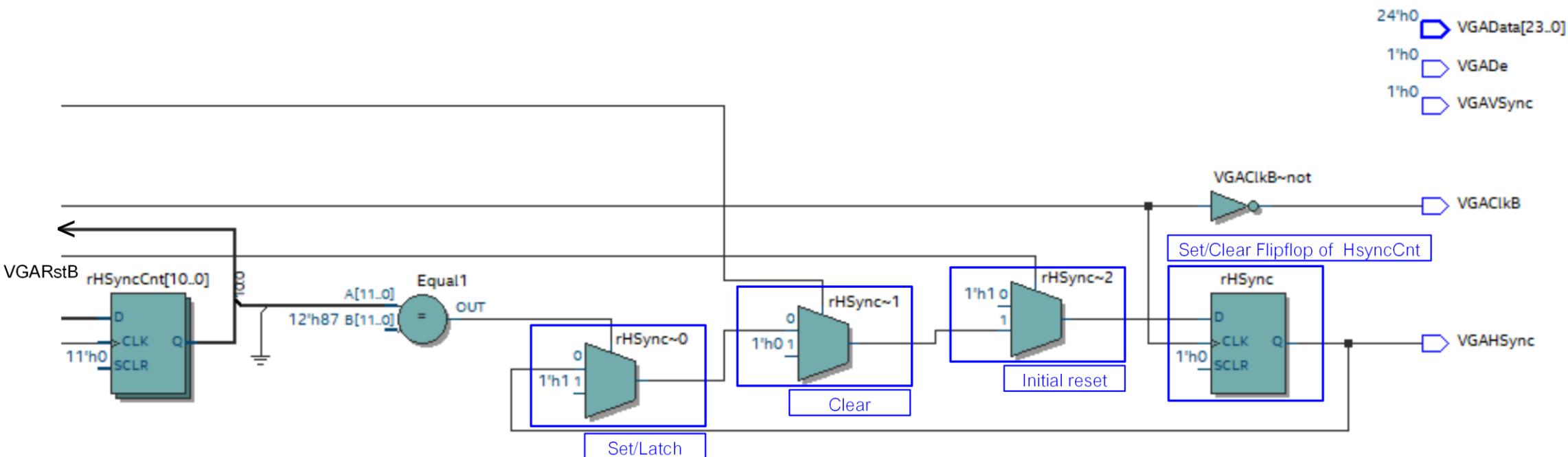
หลังจาก Synthesis เสร็จสิ้น ให้ตรวจสอบวงจรที่ได้จากการ Synthesis ด้วย RTL Viewer

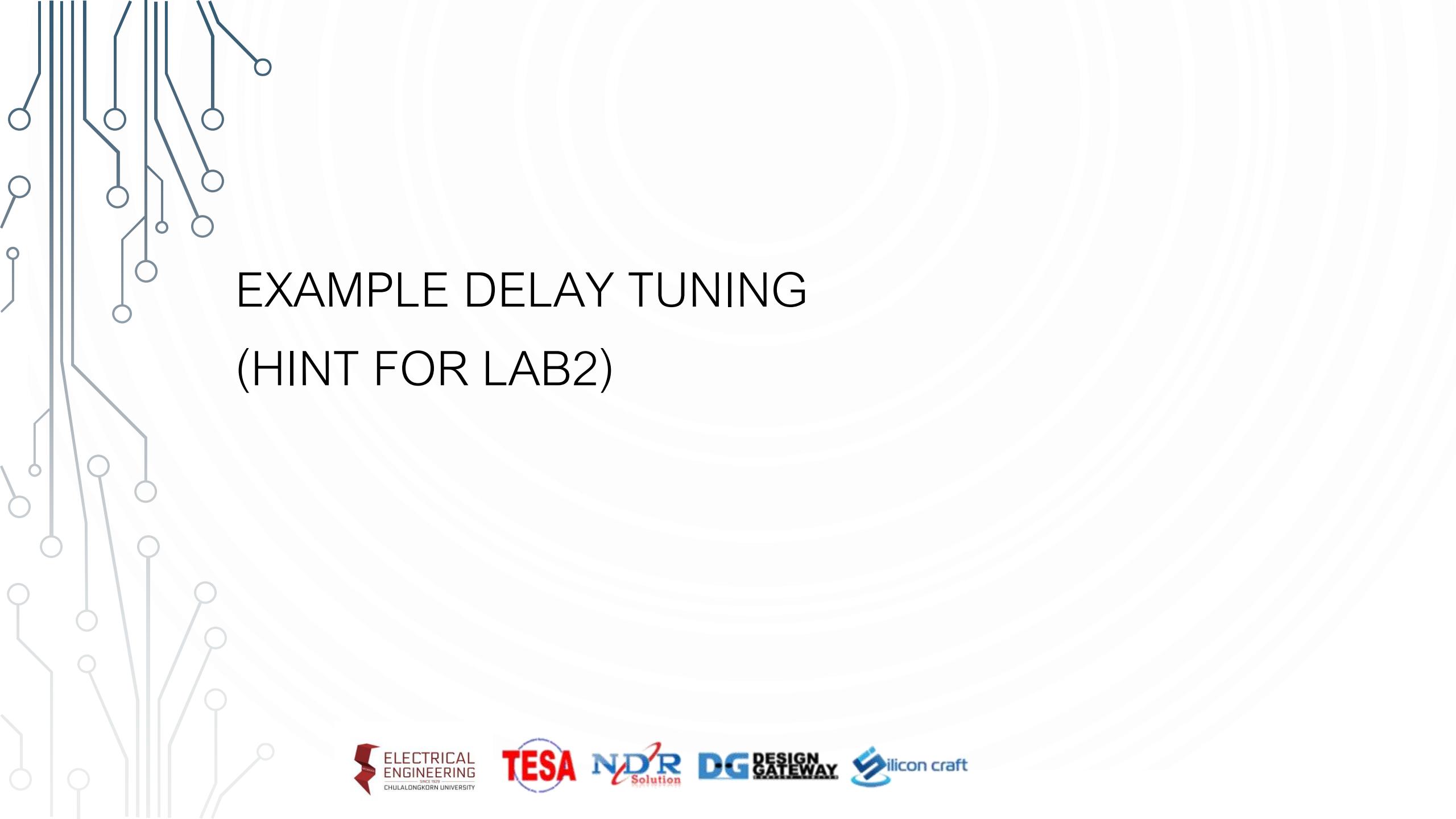


# LAB1 : RTL VIEWER RESULT OF RHSYNCNT



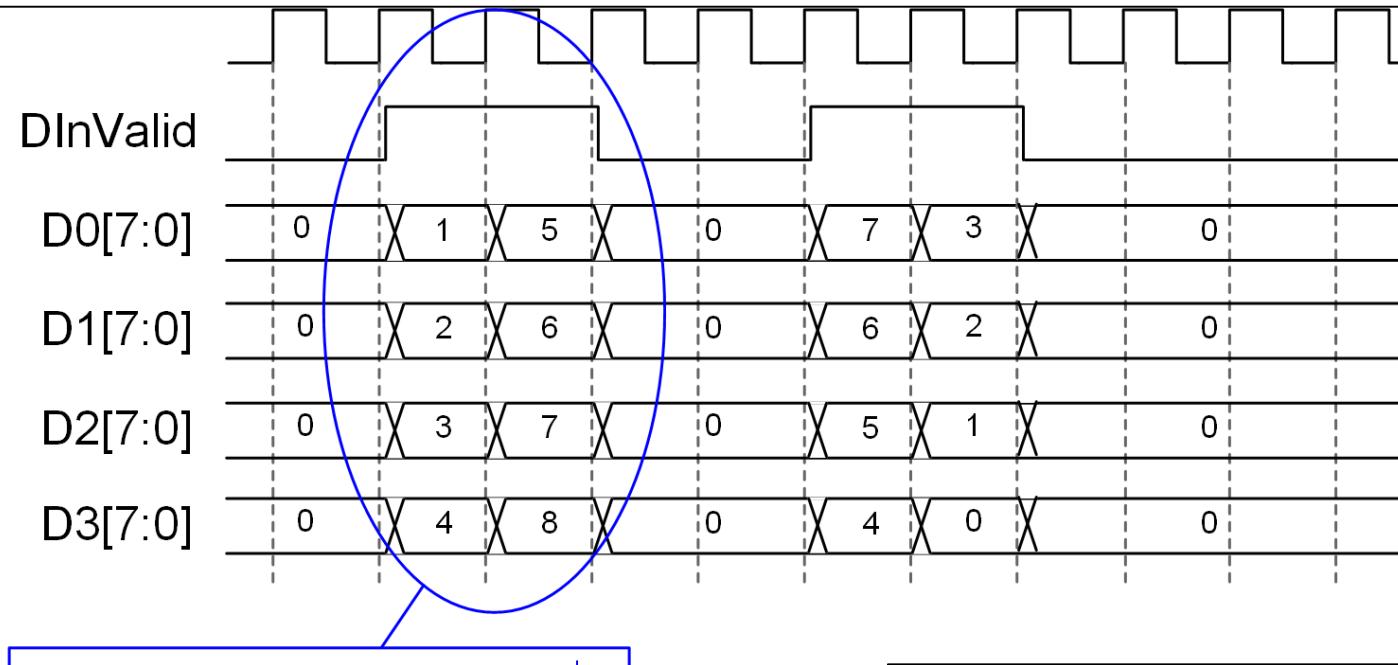
# LAB1 : RTL VIEWER RESULT OF HSYNC





## EXAMPLE DELAY TUNING (HINT FOR LAB2)

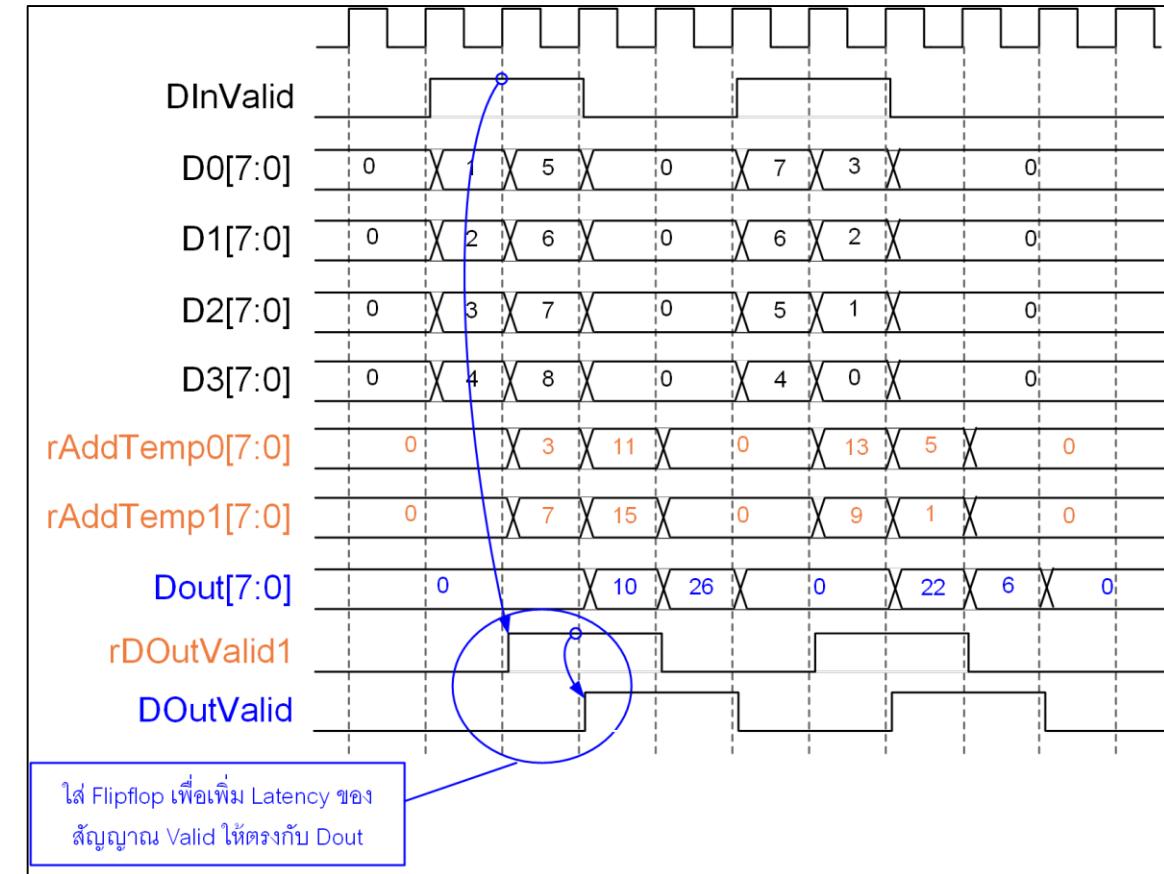
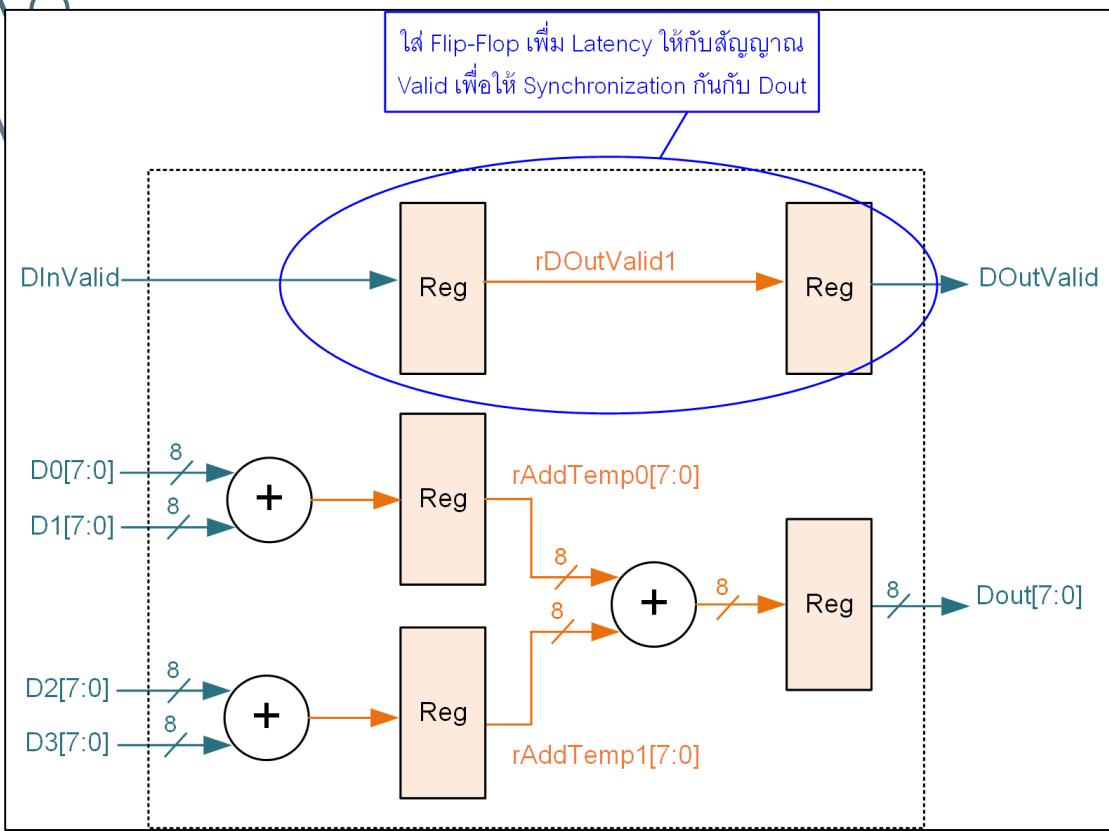
# EXAMPLE DESIGN TO TUNE OUTPUT LATENCY (1)



D0,D1,D2, และ D3 จะถือว่า Valid ก็ต่อเมื่อ  
DInValid มีค่าเป็นหนึ่ง ซึ่งเรียกว่าร่วงเวลา  
D0,D1,D2 และ D3 เป็นร่วงเวลาที่  
Synchronization กันกับ Valid

ตัวอย่าง เมื่อวงจรต้องรับข้อมูลเข้ามาทั้งหมด 4 ตัวพร้อมกันเมื่อ DInValid='1'  
ส่วนในช่วงที่ DInValid='0' ก็จะข้ามไปไม่ใช้ข้อมูลมาประมวลผล

## EXAMPLE DESIGN TO TUNE OUTPUT LATENCY (2)

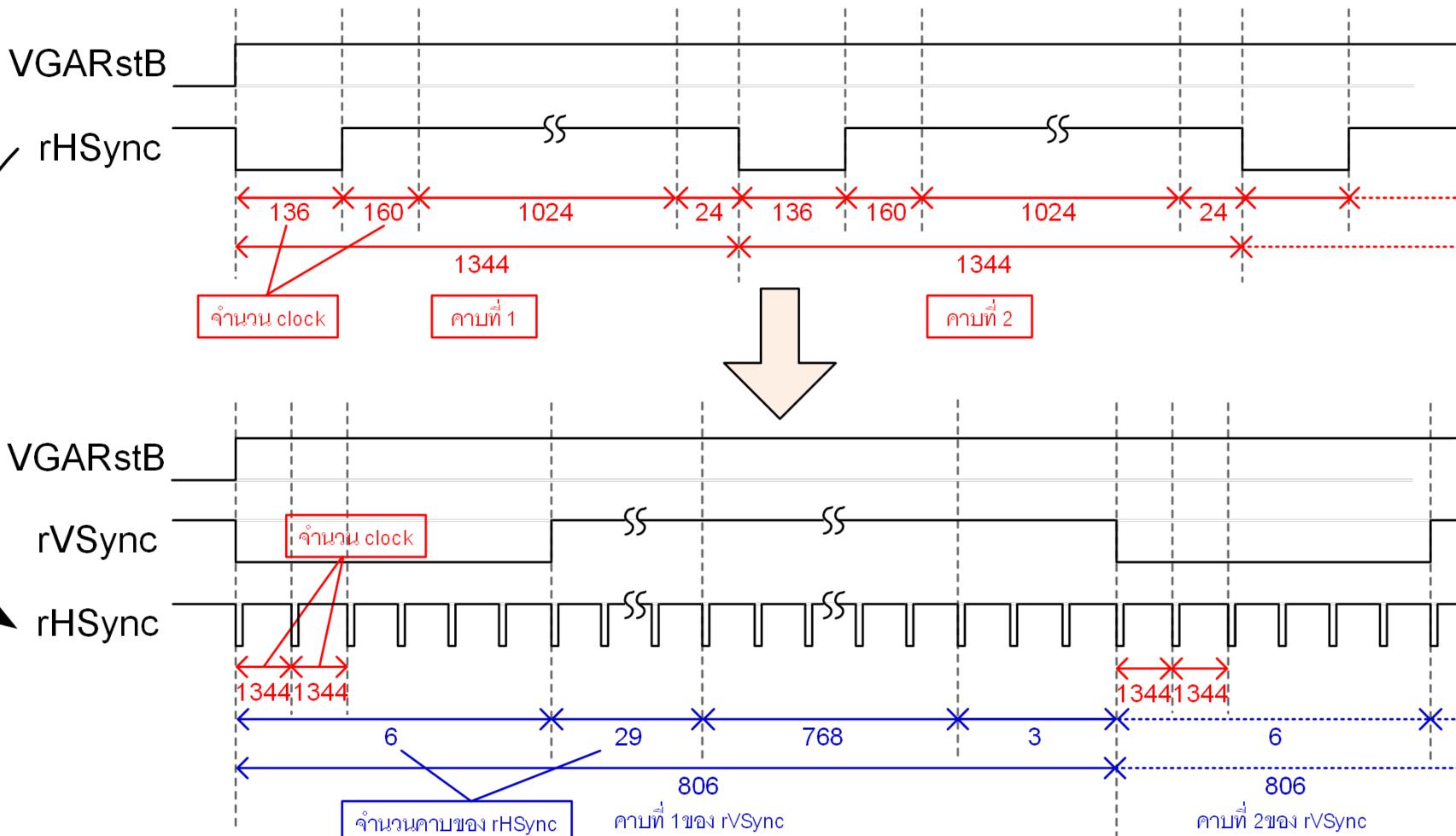


เมื่อ Data path (D0-D3) ต้องถูกประมวลผลผ่านวงจรบวกไป 2 ชั้น และผ่าน DFF 2 ชั้น การสร้าง DOutValid จึงต้องถูกสร้างจาก DInValid โดยผ่าน DFF 2 ตัว เพื่อทำให้จังหวะสุดท้ายของข้อมูลข้าอกันนี้ มี timing ที่ตรงกัน

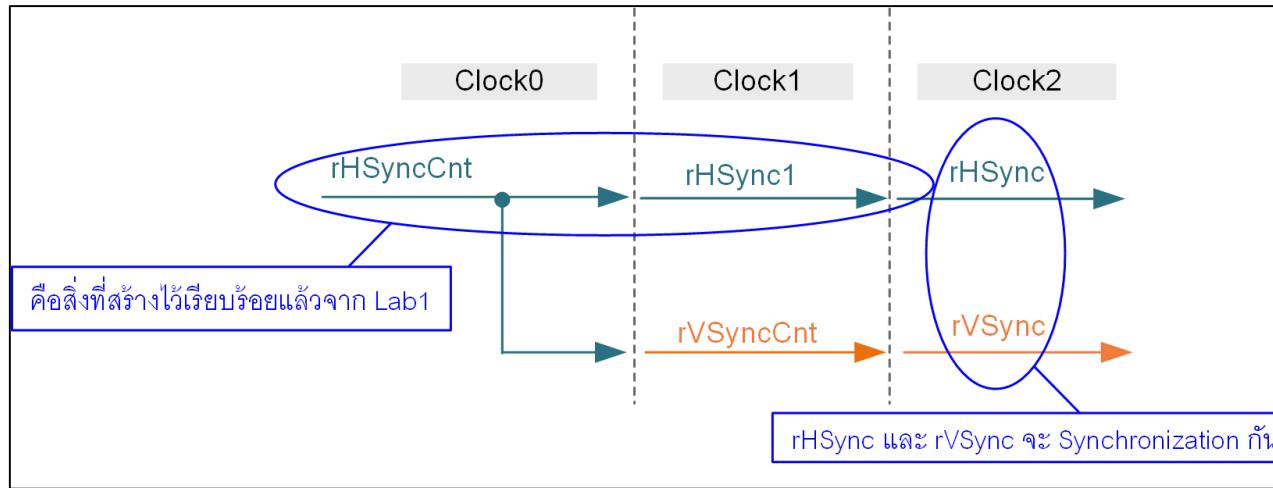
เมื่อมีสัญญาณหลาย ๆ ตัวขึ้นมา เช่น สัญญาณข้อมูลภาพที่ประกอบด้วยแม่สี 3 สี คือ R G และ B เพื่อ  
บอกค่าของสัญญาณแต่ละสี ในแต่ละ pixel พร้อมกับมีสัญญาณ Valid เพื่อบอกว่าเป็นข้อมูลภาพ 3 สี  
นั้นพร้อมแล้ว หรือยังไม่พร้อมใช้งาน ด้วยเงื่อนไขเหล่านี้ เราจะต้องระวังและออกแบบให้สัญญาณตั้งแต่  
input ไปถึง output ของแต่ละสัญญาณ ผ่านจำนวน FlipFlop เท่า ๆ กัน เพื่อให้ latency รวมทั้งหมด  
ของทุกสัญญาณพร้อมในจังหวะ Clock เดียวกัน

# LAB2: VSYNC TIMING DIAGRAM AND CODING

# LAB2 : OUTPUT SPECIFICATION : VSYNC

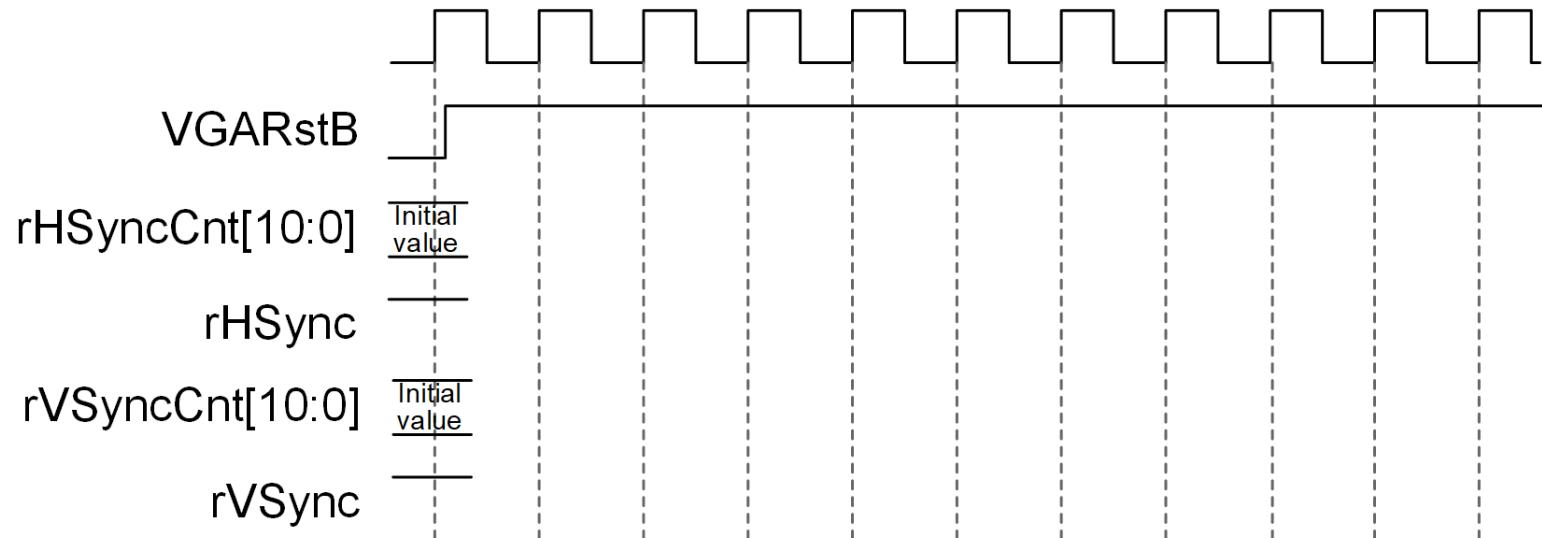


## LAB2 : TUNE DATA LATENCY (SYNCHRONIZATION)



- จากโจทย์ของ Lab2 สัญญาณ VSync นั้นจะมีความสัมพันธ์กับจำนวน HSync ที่สร้างไปแล้ว ดังนั้นจึงต้องมีการสร้าง Counter สำหรับนับจำนวน HSync เพิ่มขึ้นมาให้ชื่อว่า rVSyncCnt
- สัญญาณ rVSyncCnt นั้นถูกสร้างมาจาก rHSyncCnt และจึงนำ rVSyncCnt ไปสร้างสัญญาณ VSync อีกที
- เดิมสัญญาณ HSync ถูกสร้างจาก rHSyncCnt โดยตรง จึงเห็นได้ว่าผ่านจำนวน Register มากกว่า VSync
- ต้องเติม Register ให้กับ path ที่ใช้สร้าง HSync เพื่อให้สูดท้ายจังหวะของ HSync กับ VSync นั้นตรงกับที่เราต้องการ

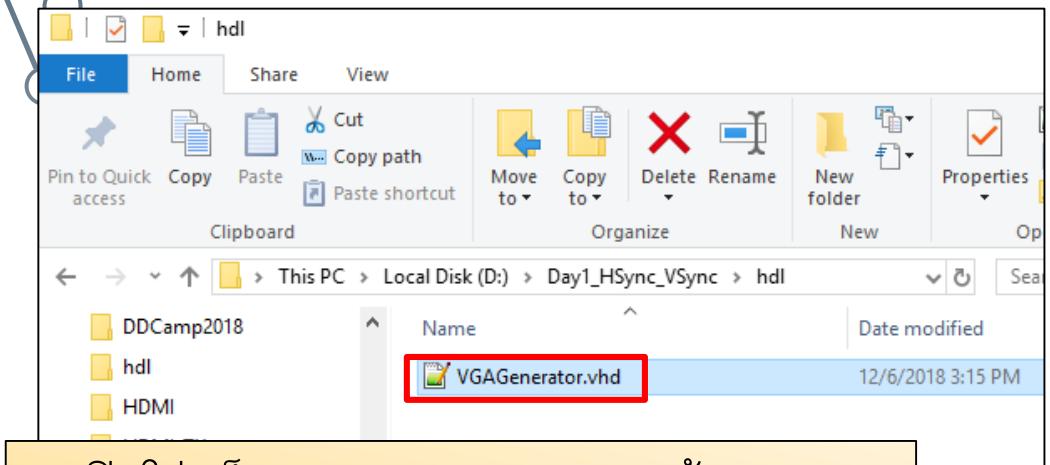
## LAB2 : DRAW TIMING DIAGRAM OF VSYNC



### คำแนะนำ

- สร้าง counter ชื่อ rVSyncCnt โดยใช้ rHSyncCnt มาช่วยนับจังหวะ และสร้าง rVSync ตามค่าของ rVSyncCnt/rHSyncCnt
- วาด timing diagram โดยข้ามค่าของ rHSyncCnt ที่ไม่ทำให้ rHSync เปลี่ยนค่าไป และเน้นว่าด้วยที่ rHSync กำลังจะเปลี่ยนค่าอย่างละเอียดแทน
- วาด timing diagram โดยการข้ามค่า rVSyncCnt ที่ไม่ทำให้ rVSync เปลี่ยนค่าไป และเน้นว่าด้วยที่ rVSync กำลังจะเปลี่ยนค่าอย่างละเอียดแทน

# LAB2 : CODING VSYNC

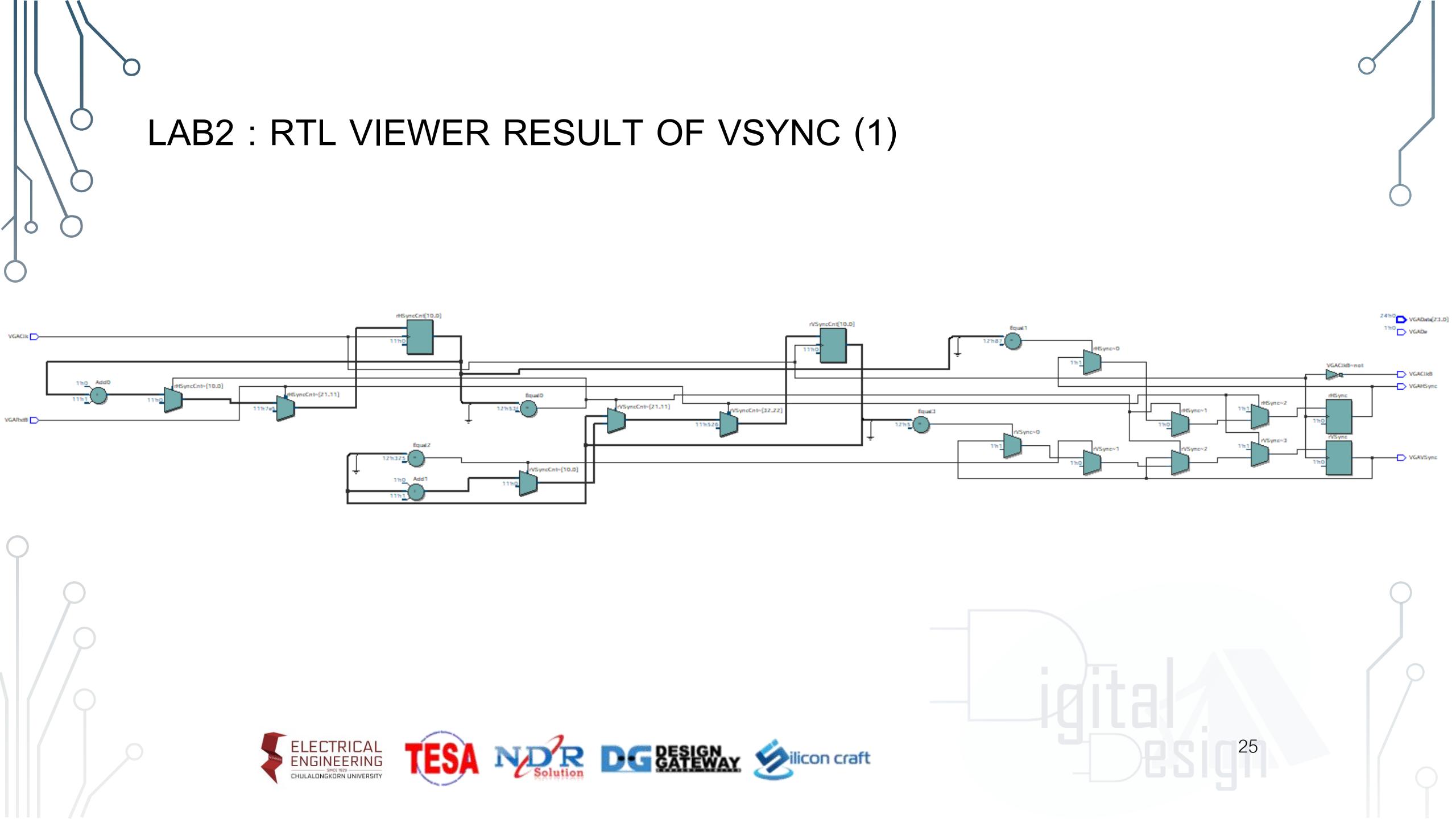


1) เปิดโปรเจ็ค Day1\_Hsync\_Vsync เข้า hdl  
folder แล้วแก้ไขไฟล์ VGAGenerator.vhd

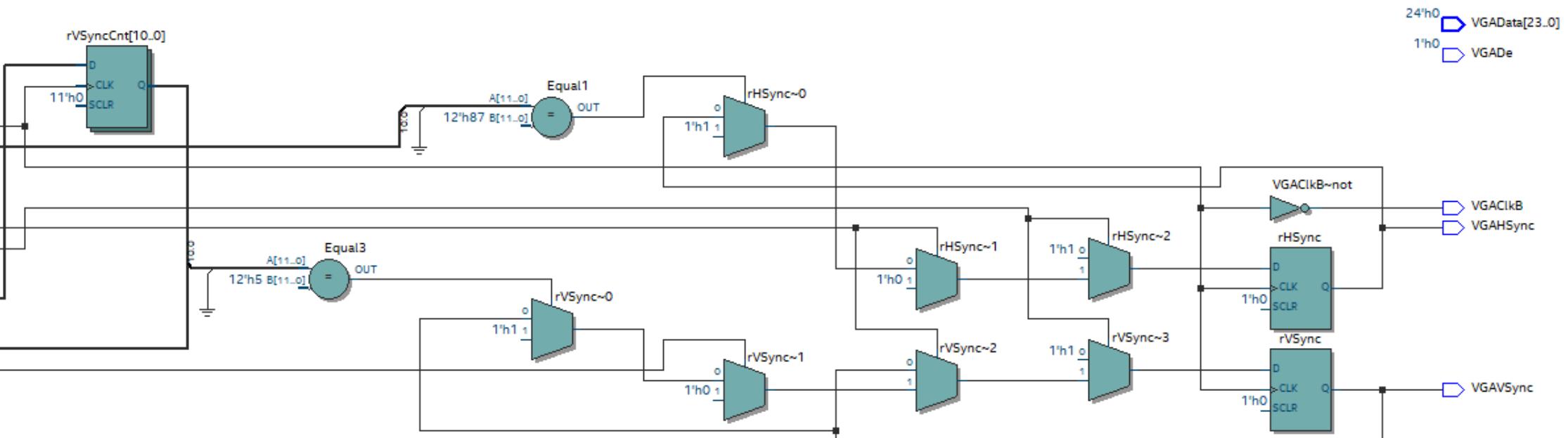
```
100
101  [[ LBA 2 : CODING VSYNC(1) ]]
102
103  u_rVSyncCnt : Process (VGAClk) Is
104  Begin
105      if ( rising_edge(VGAClk) ) then
106          if ( VGARstB='0' ) then
107              -- coding here
108              -- initial value
109          else
110              -- coding here
111              -- behaviour
112          end if;
113      end if;
114  End Process u_rVSyncCnt;
115
116  u_rVSync : Process (VGAClk) Is
117  Begin
118      if ( rising_edge(VGAClk) ) then
119          if ( VGARstB='0' ) then
120              -- coding here
121              -- initial value
122          else
123              -- coding here
124              -- behaviour
125          end if;
126      end if;
127  End Process u_rVSync;
```

2) ภายใน VGAGenerator.vhd เขียน code  
สำหรับสัญญาณ rVSyncCnt และ rVSync  
ใน Process u\_rVSyncCnt และ u\_rVSync  
ตามลำดับ

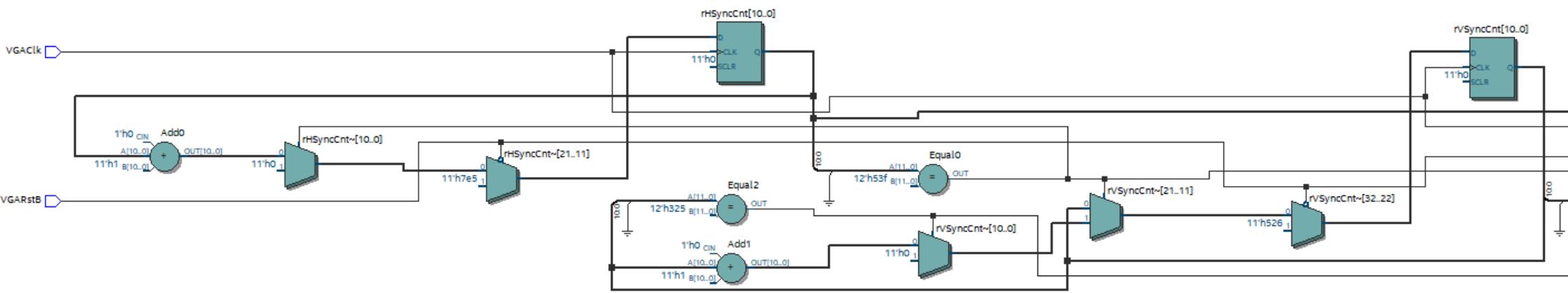
# LAB2 : RTL VIEWER RESULT OF VSYNC (1)



## LAB2 : RTL VIEWER RESULT OF VSYNC (2)



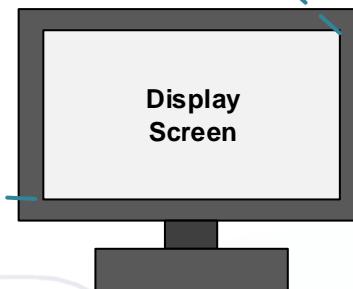
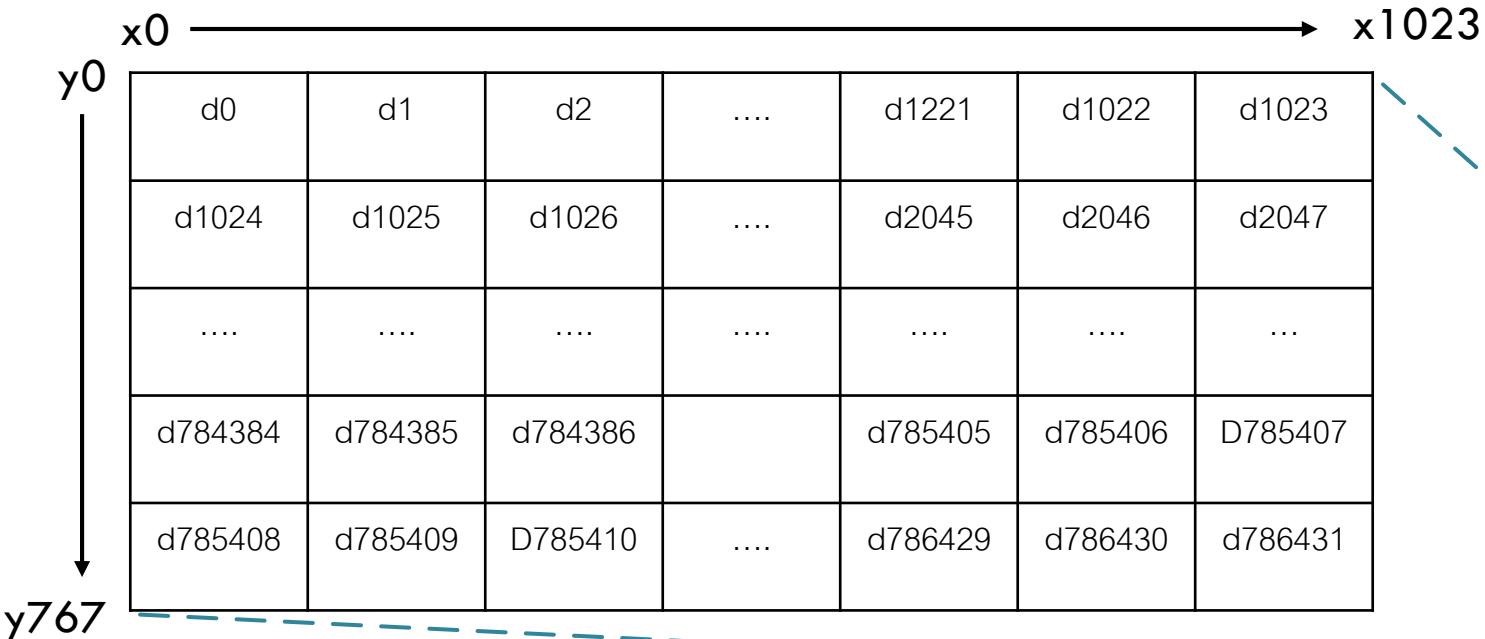
# LAB2 : RTL VIEWER RESULT OF VSYNC (3)



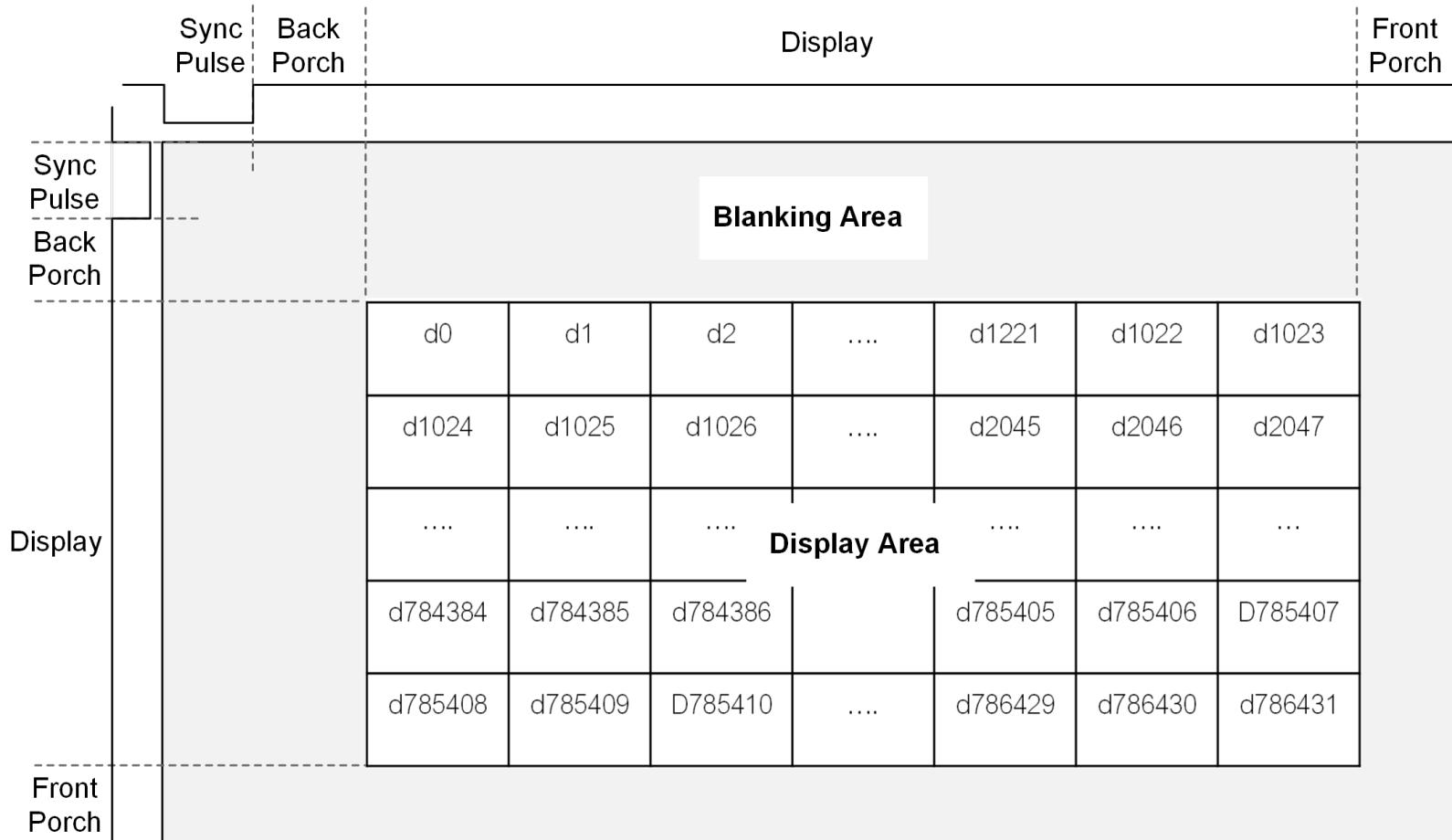


# CHALLENGE : VGA GENERATOR

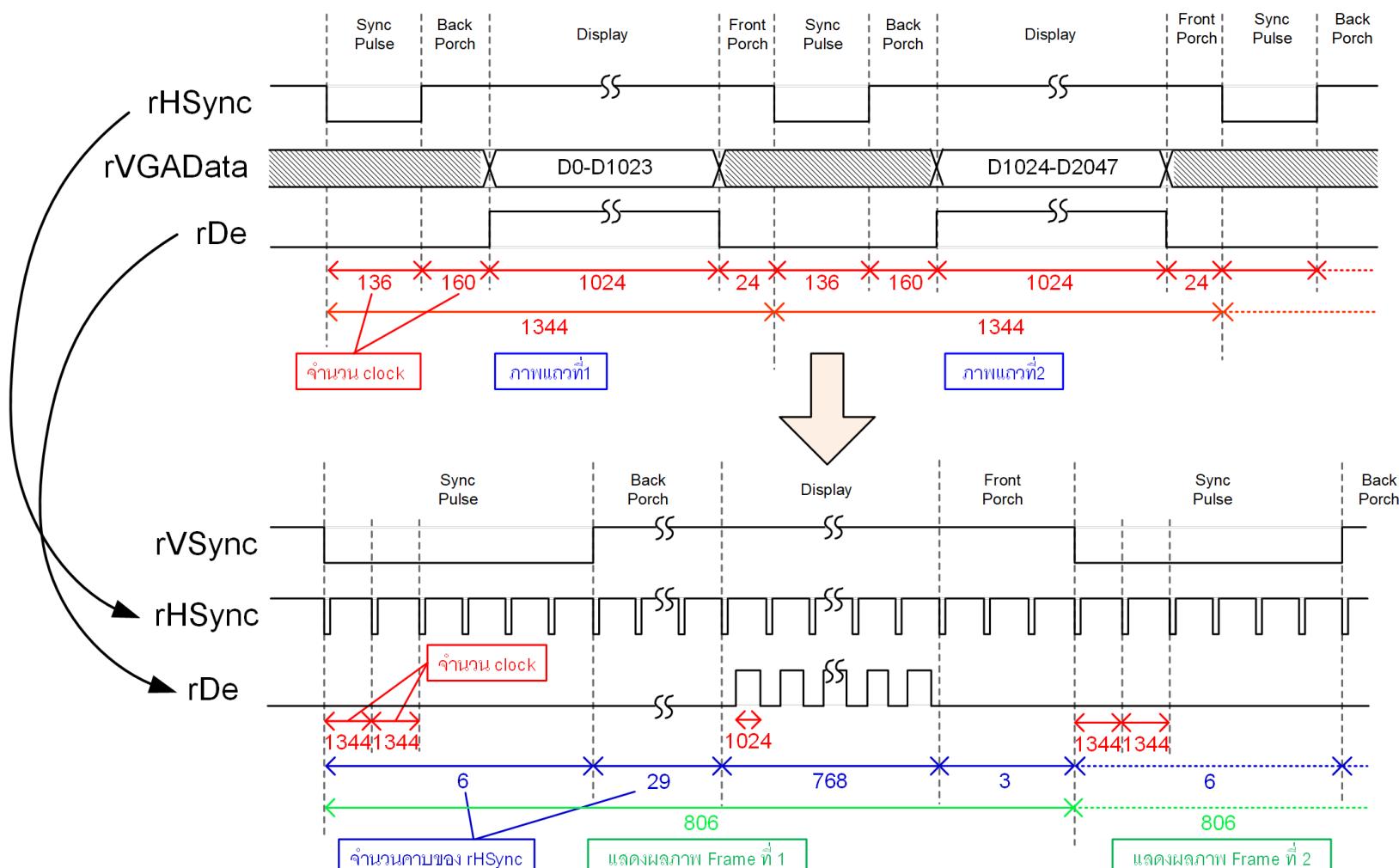
DISPLAY SEQUENCE SIZE = 1024X768



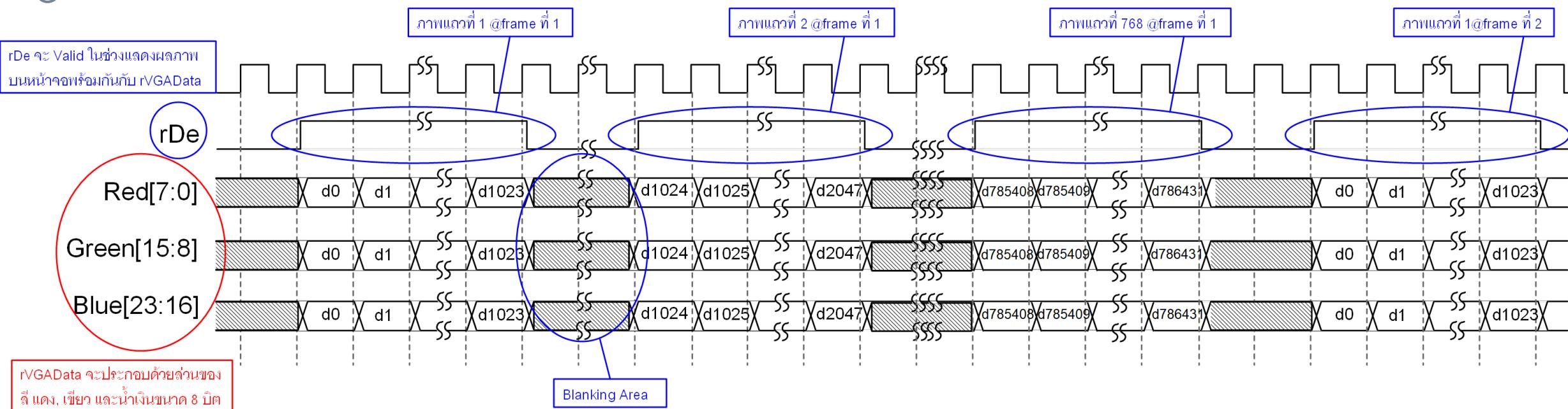
# VGA SPECIFICATION & TIMING DIAGRAM (1)



## VGA SPECIFICATION & TIMING DIAGRAM (2)



# DATA ENABLE & VGA DATA TIMING DIAGRAM

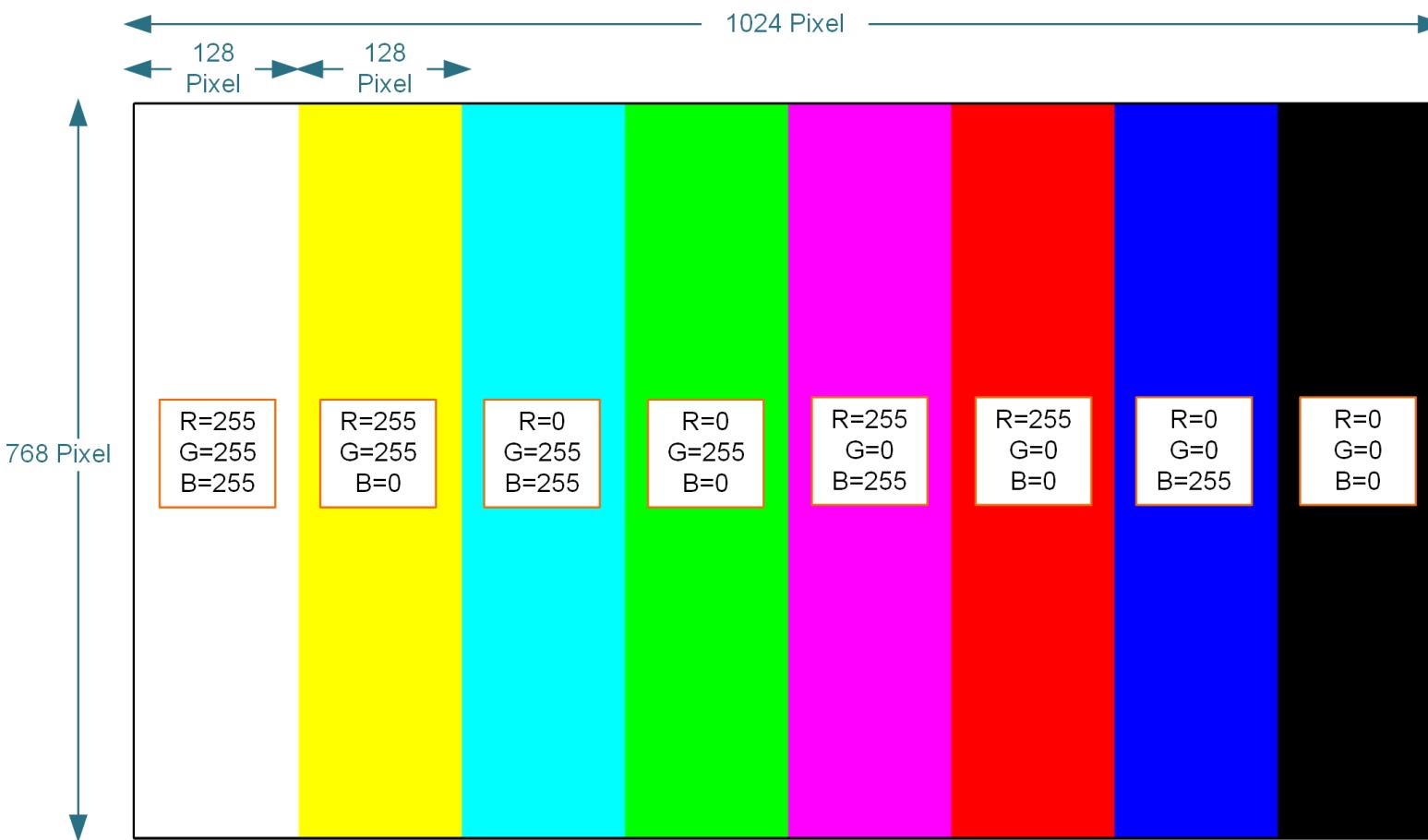




EXERCISE1

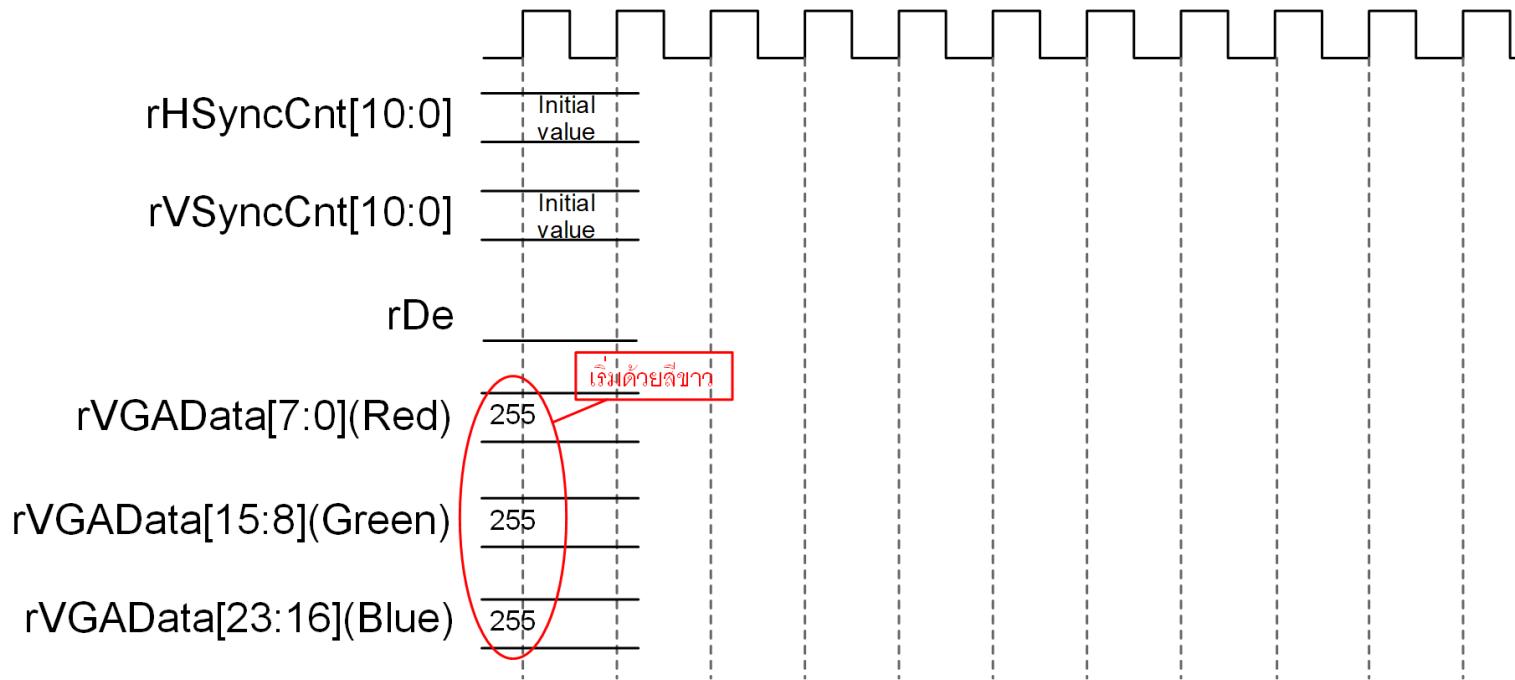
# VERTICAL COLORBAR PATTERN

# VERTICAL COLOR BAR TEST PATTERN



Ref: [https://commons.wikimedia.org/wiki/File:EBU\\_Colorbars\\_HD.svg](https://commons.wikimedia.org/wiki/File:EBU_Colorbars_HD.svg)

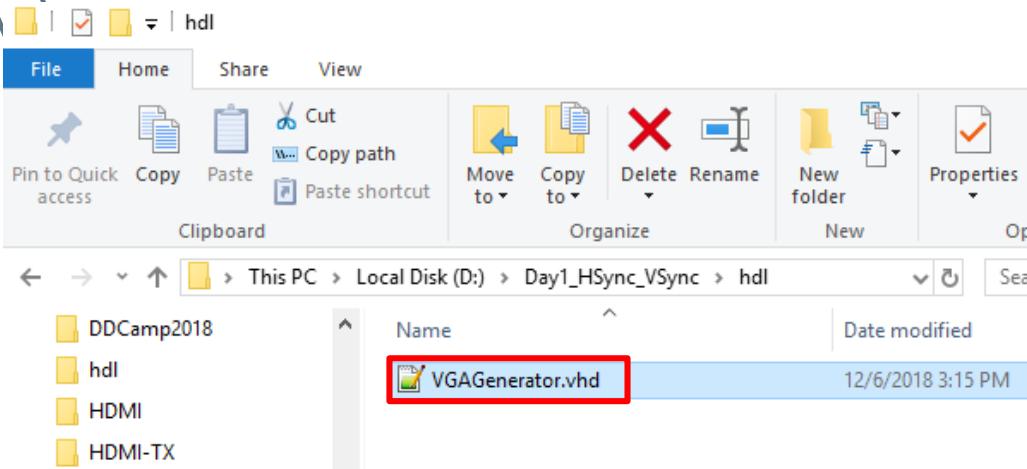
# EXERCISE 1 : DRAW TIMING DIAGRAM OF VERTICAL COLOR BAR



## คำแนะนำ

- ใช้ rHSyncCnt และ rVSyncCnt เป็น input ในการสร้าง rDe ให้ตรงจังหวะที่ต้องการ
- ใช้ **rHSyncCnt** มาใช้ในการสร้าง rVGAData ให้เปลี่ยนค่าตามจังหวะที่ต้องการ
- วาด timing diagram โดยการย่อ counter เมื่อไม่ใช่จังหวะที่สัญญาณ rVGAData และ rDe มีการเปลี่ยนแปลงค่า
- วาด timing diagram อย่างน้อย 1 Frame (แบบย่อ)

# LAB1 : CODING VERTICAL COLOR BAR



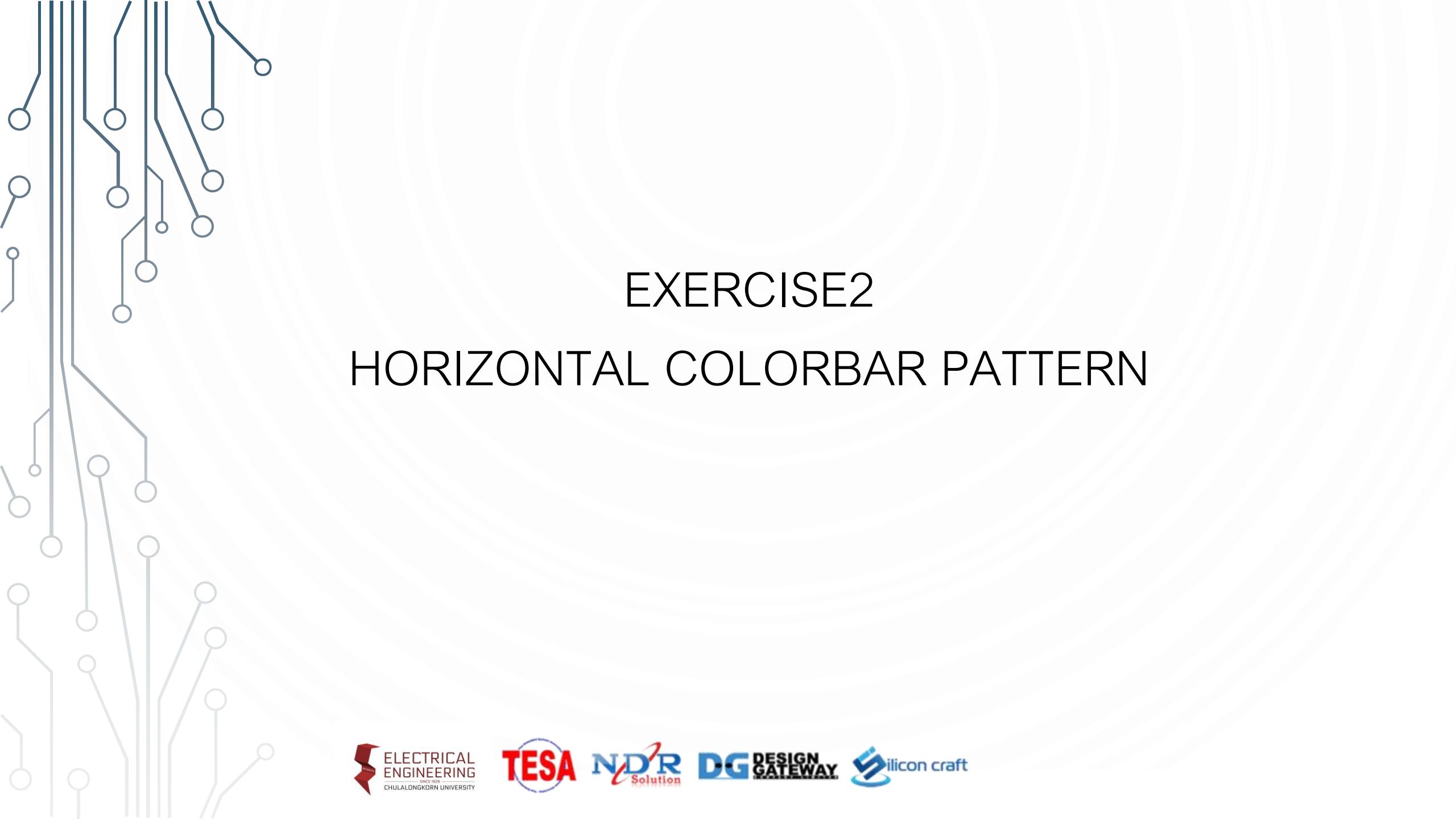
1) เปิดโปรเจ็ค Day1\_Hsync\_Vsync เข้า hdl folder แล้วแก้ไขไฟล์ VGAGenerator.vhd

```
129
130  -- [[ Challenge Exercise : CODING TEST PATTERN ]]
131
132  u_rDe : Process (VGAClk) Is
133  Begin
134      if ( rising_edge(VGAClk) ) then
135          if ( VGARstB='0' ) then
136              -- coding here
137              -- initial value
138          else
139              -- coding here
140              -- behaviour
141          end if;
142      end if;
143  End Process u_rDe;

145  u_rVGAData : Process (VGAClk) Is
146  Begin
147      if ( rising_edge(VGAClk) ) then
148          -- coding here
149          -- behaviour
150      end if;
151  End Process u_rVGAData;

153  End Architecture rtl;
154
```

2) ภายใน VGAGenerator.vhd เขียน code สำหรับสัญญาณ rDe และ rVGAData ใน Process u\_rDe และ u\_rVGAData ตามลำดับ



# EXERCISE2

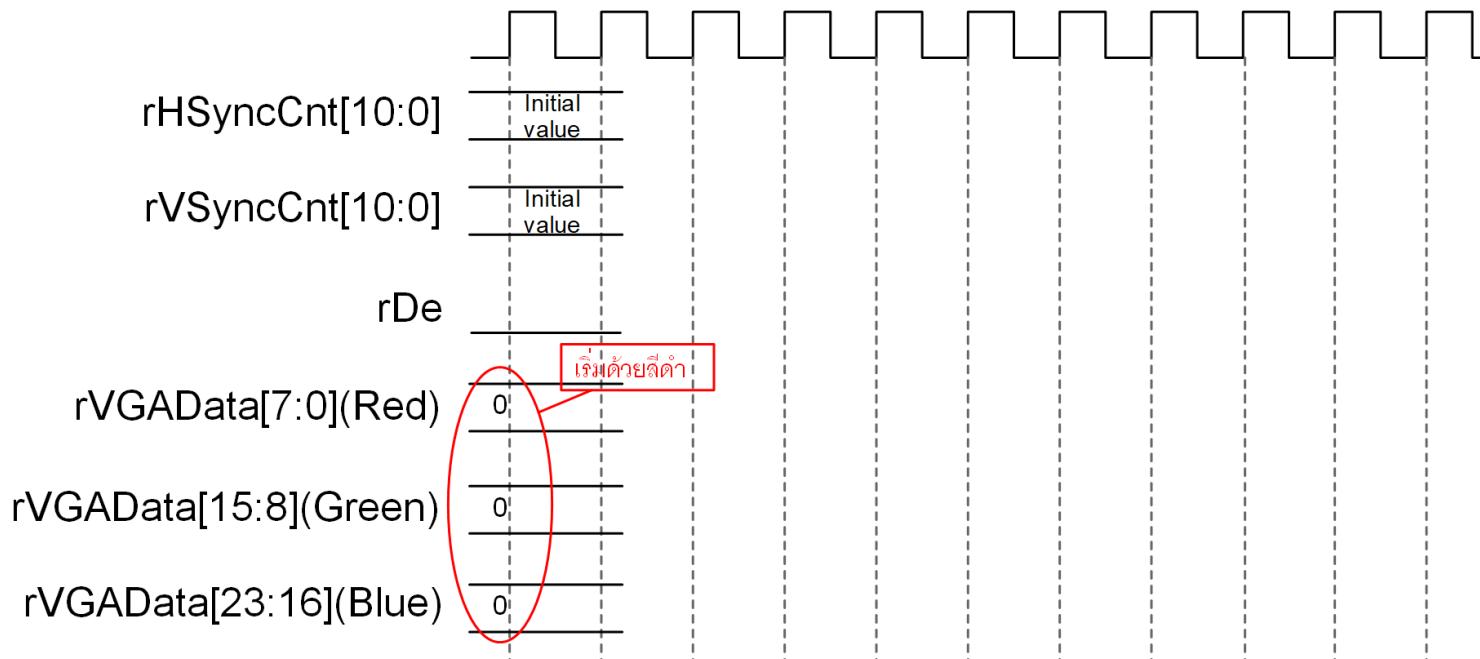
## HORIZONTAL COLORBAR PATTERN

# HORIZONTAL COLOR BAR TEST PATTERN



Ref: [https://commons.wikimedia.org/wiki/File:EBU\\_Colorbars\\_HD.svg](https://commons.wikimedia.org/wiki/File:EBU_Colorbars_HD.svg)

## EXERCISE 2 : DRAW TIMING DIAGRAM OF HORIZONTAL COLOR BAR



### คำแนะนำ

- ใช้ rHSyncCnt และ rVSyncCnt เป็น input ในการสร้าง rDe ให้ตรงจังหวะที่ต้องการ
- ใช้ rVSyncCnt มาใช้ในการสร้าง rVGAData ให้เปลี่ยนค่าตามจังหวะที่ต้องการ
- วาด timing diagram โดยการย่อ counter เมื่อไม่ใช่จังหวะที่สัญญาณ rVGAData และ rDe มีการเปลี่ยนแปลงค่า
- วาด timing diagram อย่างน้อย 1 Frame (แบบย่อ)

## Q & A

<https://www.facebook.com/DigitalDesignThailand/>



<https://forfpgadesign.wordpress.com/>

