



# การทดลองที่ 5 Code Converter

หน้า  
1 / 9

ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

ภาคการศึกษาที่.....1.....ปีการศึกษา.....2564

รหัสวิชา 010113026 ชื่อวิชา Digital Laboratory

ตอนเรียน .....4..... หมายเลขโต๊ะ.....

รหัสนักศึกษา.....6201011601188.....

ชื่อ-นามสกุล.....นายโศภณ สุขสมบูรณ์.....

อาจารย์ผู้สอน.....CSP.....

เวลาที่ทำการทดลอง TH: 13.00-16.00 วันที่.....26/08/64.....

## การทดลองที่ 5 Code Converter

### วัตถุประสงค์

1. เพื่อให้สามารถใช้โปรแกรมคอมพิวเตอร์เพื่อจำลองการทำงานของวงจรลอจิกเกตได้
2. เพื่อให้เข้าใจการทำงานของอุปกรณ์แปลงรหัส BCD to 7-Segment Converter
3. เพื่อให้เข้าใจการทำงานของอุปกรณ์แปลงรหัส Binary to Gray Code Converter

### เครื่องมือและอุปกรณ์

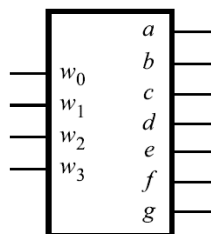
1. ระบบคอมพิวเตอร์ 1 เครื่อง พร้อมติดตั้งโปรแกรม Quartus II เวอร์ชัน 8.0 (Student Edition) ขึ้นไป

### การทดลองตอนที่ 1

วงจรแปลงรหัสแบบ BCD to 7-Segment Converter

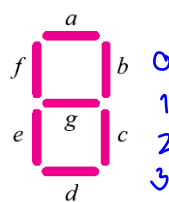
#### คำสั่งการทดลอง

1. ให้ออกแบบวงจรแปลงรหัส BCD ให้เป็นรหัสแบบ 7-Segment Decoder ในรูปที่ 1 โดยใช้ข้อมูลแสดงการทำงานจากตารางความจริงของแต่ละเซกเมนต์ a, b, c, d, e, f และ g ให้น.ศ. ออกแบบตามขั้นตอนดังนี้
  - ก) ให้แสดงวิธีการทำ K-Map จากตารางความจริง
  - ข) สร้างสมการลอจิกฟังก์ชัน จาก K-Map
  - ค) เขียนวงจรที่ออกแบบตามสมการลอจิกฟังก์ชัน โดยใช้เกตพื้นฐาน ( AND , OR , NOT etc. )



a )

a) BCD-7 Segment Code Converter block



b )

b) 7-Segment Display

รูปที่ 1

w <sub>3</sub>	w <sub>2</sub>	w <sub>1</sub>	w <sub>0</sub>	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1



# การทดลองที่ 5 Code Converter

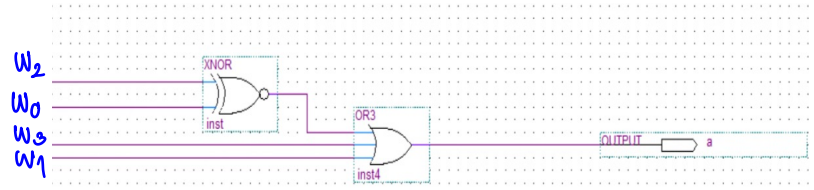
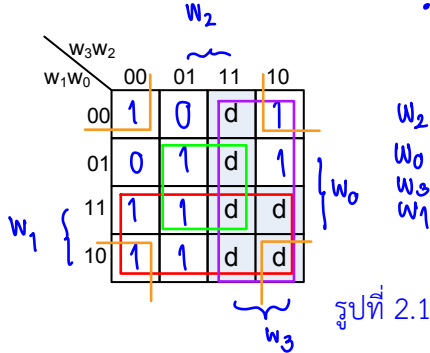
หน้า

2 / 9

2. บันทึกผลการออกแบบ (แสดงการลดรูปด้วยตาราง K-Map, สมการลอจิก, และ รูปวงจร), d=don't care

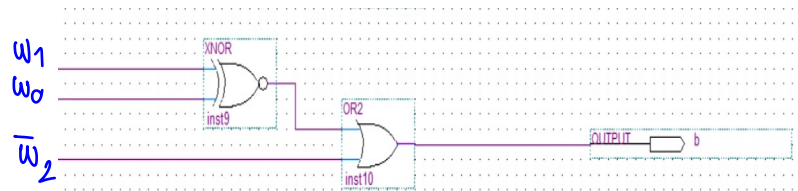
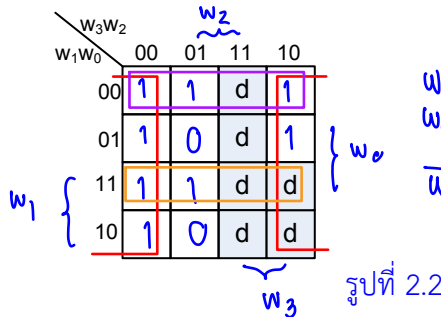
เช็ทเมนต์ a  $a(w_3, w_2, w_1, w_0) = w_2 w_0 + w_3 + w_1 + \overline{w_2} \overline{w_0}$

$$\therefore a(w_3, w_2, w_1, w_0) = \overline{w_2} \oplus w_0 + w_3 + w_1$$

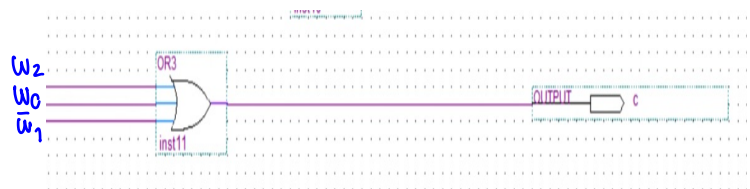
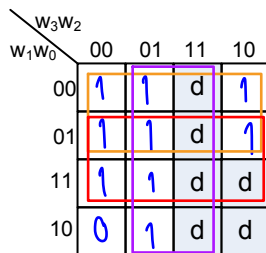


เช็ทเมนต์ b  $b(w_3, w_2, w_1, w_0) = \overline{w_1} \overline{w_0} + \overline{w_2} + w_1 w_0$

$$\therefore b(w_3, w_2, w_1, w_0) = \overline{w_1} \oplus w_0 + \overline{w_2}$$

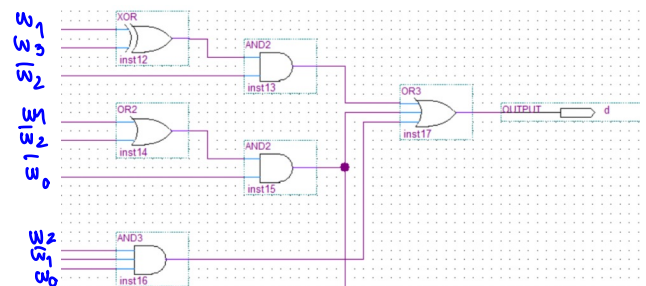
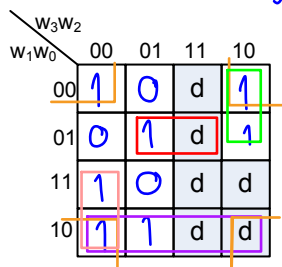


เช็ทเมนต์ c  $c(w_3, w_2, w_1, w_0) = w_2 + w_0 + \overline{w_1}$



เช็ทเมนต์ d  $d(w_3, w_2, w_1, w_0) = w_2 \overline{w_1} w_0 + w_3 \overline{w_2} \overline{w_1} + \overline{w_3} \overline{w_2} w_1 + w_1 \overline{w_0} + \overline{w_2} \overline{w_0}$

$$\therefore d(w_3, w_2, w_1, w_0) = \overline{w_2} (w_3 \oplus w_1) + \overline{w_0} (w_1 + \overline{w_2}) + w_2 \overline{w_1} w_0$$





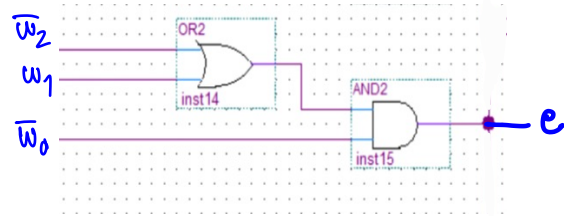
# การทดลองที่ 5 Code Converter

หน้า  
3 / 9

ใช้เกมนต์ e  $e(w_3, w_2, w_1, w_0) = \overline{w_2} \overline{w_0} + w_1 \overline{w_0}$   
 $\therefore e(w_3, w_2, w_1, w_0) = \overline{w_0} (\overline{w_2} + w_1)$

$w_3 w_2$	00	01	11	10
$w_1 w_0$				
00	1	0	d	1
01	0	0	d	0
11	0	0	d	d
10	1	1	d	d

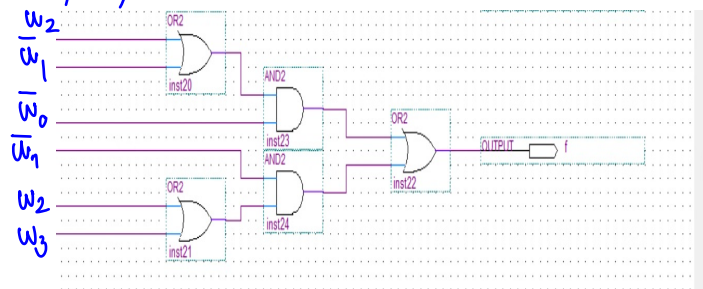
รูปที่ 2.5



ใช้เกมนต์ f  $f(w_3, w_2, w_1, w_0) = w_2 \overline{w_0} + w_2 \overline{w_1} + w_3 \overline{w_1} + \overline{w_1} \overline{w_0}$   
 $\therefore f(w_3, w_2, w_1, w_0) = \overline{w_0} (w_2 + \overline{w_1}) + \overline{w_1} (w_2 + w_3)$

$w_3 w_2$	00	01	11	10
$w_1 w_0$				
00	1	1	d	1
01	0	1	d	1
11	0	0	d	d
10	0	1	d	d

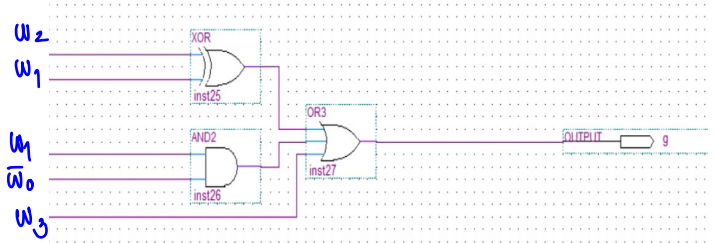
รูปที่ 2.6



ใช้เกมนต์ g  $g(w_3, w_2, w_1, w_0) = w_2 \overline{w_1} + w_1 \overline{w_0} + w_3 + \overline{w_2} w_1$   
 $\therefore g(w_3, w_2, w_1, w_0) = (w_2 \oplus w_1) + w_1 \overline{w_0} + w_3$

$w_3 w_2$	00	01	11	10
$w_1 w_0$				
00	0	1	d	1
01	0	1	d	1
11	1	0	d	d
10	1	1	d	d

รูปที่ 2.7



ลายเซ็นอาจารย์ผู้ควบคุม..... /...../.....

3. ให้ น.ศ. สร้างโฟลเดอร์สำหรับเก็บงานชิ้นใหม่เพื่อเก็บงานที่จะทดลองในการทดลองนี้ โดยให้ชื่อว่า

“Lab05CombiBuild2” จากนั้นให้สร้างโปรเจกต์

ชื่อ “BCD2\_7Seg” ขึ้นมา ดังรูปที่ 3 ให้ใช้ชิพ

FPGA เบอร์ EP3C10E144C8

รูปที่ 3

What is the working directory for this project?

C:\altera\80\quartus\LabDigit\Lab05CombiBuild2

What is the name of this project?

BCD2\_7Seg

What is the name of the top-level design entity for this project? exactly match the entity name in the design file.

BCD2\_7Seg



4. นำวงจรที่ได้จากการออกแบบในข้อ 2 (รูปที่ 2.1 – 2.7) มาเขียนวงจรบน Graphic Editor Tool และทำการคอมไพล์ให้เรียบร้อย จากนั้นให้ทำการสร้าง **symbol** ของวงจรขึ้นมา โดยไปที่เมนู

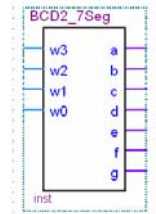
**File >> create/update**

และเลือก

**Create symbol file for current file**

ก็จะได้อุปกรณ์ชื่อ **BCD2\_7Seg** ดังรูปที่ 4

รูปที่ 4



5. สร้างไฟล์แสดงแผนภาพทางเวลา (Timing diagram) ด้วย Vector Waveform Editor Tool โดย

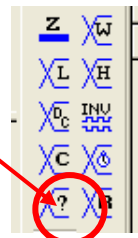
- กำหนดค่าของสำหรับแสดงผลจำลองการทำงานให้มีค่าดังนี้

End Time = 1.0 us

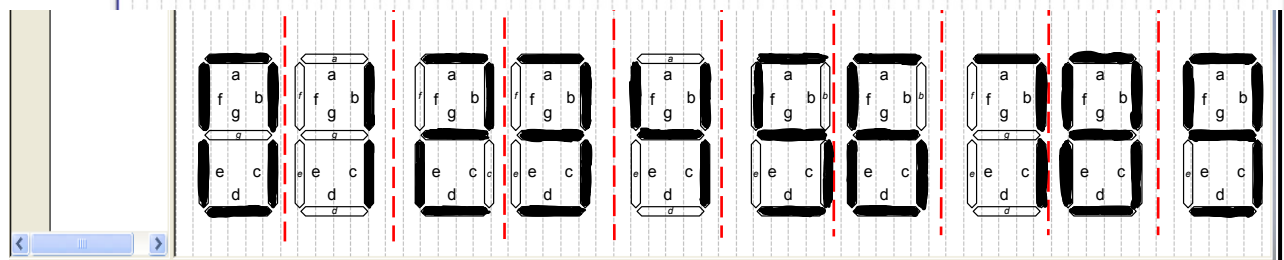
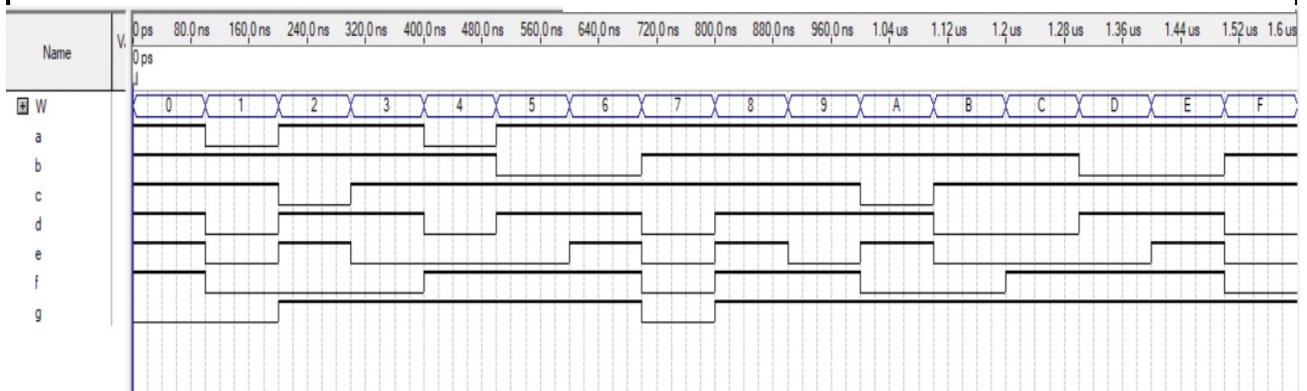
Grid Size = 1 ns

- จัดกลุ่มอินพุต  $w_0$ ,  $w_1$ ,  $w_2$  และ  $w_3$  ให้เป็นกลุ่มชื่อ “W” และกำหนดค่าเป็นแบบนับ 0 ถึง 9 (ดูรูปที่ 5) ปรับค่าช่วงละ 100 ns. ดังในรูปที่ 6

รูปที่ 5



6. จำลองการทำงานโหมด Functional Mode บันทึกผลที่ได้ลงในรูปที่ 6



รูปที่ 6

## บันทึกผลการทดลอง

ก) ในแต่ละช่วงค่าของ w ให้ น.ศ. ระบายสีดำ ลงบนเช็กแม้นต์ ( a ,b, c, d, e, f, g )

โดย เช็กแม้นต์ที่มีค่าลอจิกเป็น ‘1’ ให้ระบายระบายสีดำ (ใช้ดินสอสีเข้ม)

เช็กแม้นต์ที่มีค่าลอจิกเป็น ‘0’ ให้ปล่อยว่างไว้เช่นเดิม







สังเกตผลที่ได้จากการแสดงค่าบน 7 segment ตรงกันกับค่าของ w หรือไม่ ..... ตรงกัน



ข) อธิบายความสัมพันธ์กันของค่า w กับสถานะของลอจิกที่ปรากฏบนรูปของ 7-Segment

ในช่วง 0-9 ค่า Output แสดงผลค่าตรงกับ 7-segment ที่เกวลล์ เงินค่า 0-9 ต่อมาจึงแต่แล้ว Input อยู่ในช่วง A-F ค่า Output ที่แสดงออกมา เงินค่าที่ไม่สามารถอ่านได้ ไม่มีความหมายเชิงลอจิกหรือทศนิยมใด ๆ

ค) หากค่าของ W มีค่ามากเกินไปกว่า 9 (ค่าระหว่าง 10 ถึง 15) ปรากฏผลบนรูปของ 7-segment เช่นไร

A → , B → , C = , D → , E → , F → 

ในทางปฏิบัติจริงค่าของ W มีค่ามากเกินไปกว่า 9 (ค่าระหว่าง 10 ถึง 15) ได้หรือไม่ เพราะเหตุใด

ไม่ได้ เนื่องจาก วงจรดังกล่าว เงินวงจร BCD ที่ต้องแสดงผลเฉพาะช่วง 0-9 เท่านั้น

ลายเซ็นอาจารย์ผู้ควบคุม..... /...../.....

- ให้ทำการ**ปิดโปรเจค**ที่สร้างมาในขั้นตอนที่ 1 – 5 ก่อนที่จะทำการทดลองต่อไป
- ให้เขียนภาษา VHDL เพื่อสร้างวงจรสำหรับแปลงรหัส **BCD (ระบบเลขฐานสิบที่เขียนแทนด้วยเลขไบนารีขนาด 4 บิต)** ให้เป็นรหัสแบบ **7-Segment Decoder** ดังในรูปที่ 7 โดยดำเนินการดังนี้

- ให้สร้างโปรเจคขึ้นใหม่ชื่อ **“VHD\_7SEGM”**
- เก็บงานนี้ไว้ในไฟล์ชื่อ **“VHD\_7SEGM.vhd”**
- เมื่อคอมไพล์เสร็จแล้วให้สร้าง Symbol ไว้เพื่อเตรียมใช้ในการทดลองต่อไป

รูปที่ 7

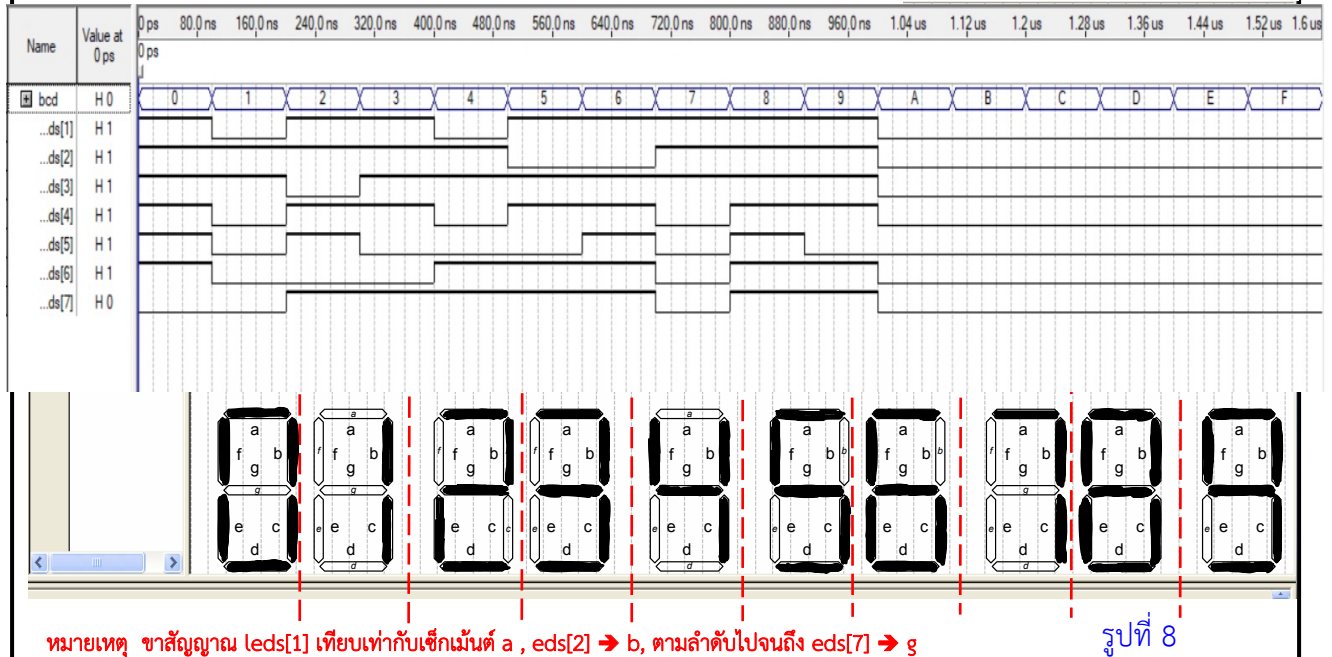
```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;

ENTITY VHD_7SEGM IS
    PORT ( bcd : IN      STD_LOGIC_VECTOR(3 DOWNTO 0) ;
          leds : OUT     STD_LOGIC_VECTOR(1 TO 7) ) ;
END VHD_7SEGM ;

ARCHITECTURE Behavior OF VHD_7SEGM IS
BEGIN
    PROCESS ( bcd )
    BEGIN
        CASE bcd IS
            -- abcdefg
            WHEN "0000" => leds <= "1111110" ;
            WHEN "0001" => leds <= "0110000" ;
            WHEN "0010" => leds <= "1101101" ;
            WHEN "0011" => leds <= "1111001" ;
            WHEN "0100" => leds <= "0110011" ;
            WHEN "0101" => leds <= "1011011" ;
            WHEN "0110" => leds <= "1011111" ;
            WHEN "0111" => leds <= "1110000" ;
            WHEN "1000" => leds <= "1111111" ;
            WHEN "1001" => leds <= "1111011" ;
            WHEN OTHERS => leds <= "0000000" ;
        END CASE ;
    END PROCESS ;
END Behavior ;
```

- สร้างไฟล์แสดงแผนภาพทางเวลา (Timing diagram) และกำหนดค่าของพารามิเตอร์สำหรับแสดงผลจำลอง โดยให้มามีค่าดังนี้ End Time = 1.0 us Grid Size = 1 ns
  - สัญญาณอินพุต “bcd” กำหนดค่าเป็นแบบตัวเลขนับจาก 0 ถึง 9 ช่วงละ 100 ns.
  - จำลองการทำงานโหมด Functional mode บันทึกผลที่ได้ลงในรูปที่ 8





## บันทึกผลการทดลอง

ก) ในแต่ละช่วงค่าของ w ให้ น.ศ. ระบายสีดำ (ใช้ดินสอสีเข้ม) ลงบนเช็กเมนต์ ( a , b, c, d, e, f, g )  
โดย เช็กเมนต์ที่มีค่าลอจิกเป็น '1' ให้ระบายระบายสีดำ (ใช้ดินสอสีเข้ม)  
เช็กเมนต์ที่มีค่าลอจิกเป็น '0' ให้ปล่อยว่างไว้เช่นเดิม  
สังเกตผลที่ได้จากการแสดงค่าบน 7 segment ตรงกันกับค่าของ w หรือไม่ ..... ตรงกัน

ข) หากค่าของ bcd มีค่ามากเกินไปกว่า 9 ( ช่วงระหว่าง 10 ถึง 15 ) จะปรากฏผลบนรูปของ 7segment เช่นไร  
ไม่เกิดการแสดงผล

ค) จากการออกแบบวงจรสำหรับแปลงรหัส BCD ให้เป็นรหัสแบบ 7-Segment Decoder ที่ออกแบบโดยใช้  
เกทพื้นฐาน และออกแบบโดยใช้ภาษา VHDL ให้อธิบายและเปรียบเทียบสิ่งที่แตกต่างกันของผลลัพธ์ที่ได้จาก  
การจำลอง การออกแบบด้วย Basic Gate จะมีการที่ Input แสดงผลเป็น A-F  
(ซึ่งเกิดค่า BCD) แต่ยังสามารถแสดงผล Output ได้ แต่ค่าที่ได้ยังไม่สามารถระบุได้ชัดเจนคืออะไร  
ในทำนองเดียวกัน การออกแบบด้วย VHDL แม้เราจะกำหนด Input เป็น BCD  
แต่ค่า Output ที่แสดงออกมาก็อยู่ในช่วง 0-9 เท่านั้น วงจรอื่นๆ จะเห็น 0 ทั้งหมด.

ลายเซ็นอาจารย์ผู้ควบคุม..... /...../.....



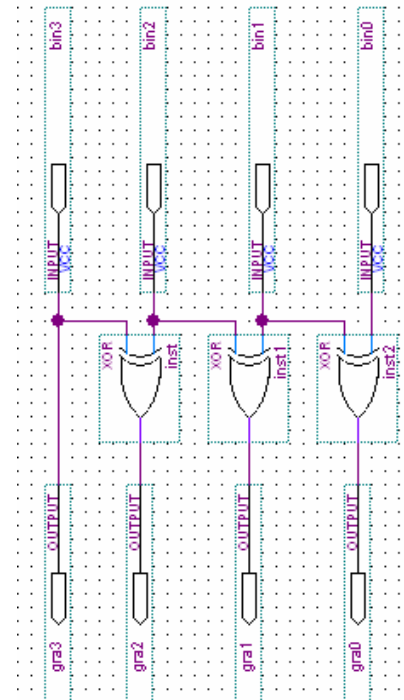
## การทดลองตอนที่ 2 หลักการทำงานของ Gray Code Converter

10. **ปิดโปรเจกต์**ที่สร้างมาในขั้นตอนก่อนหน้าและให้ทำดังนี้

- ให้สร้างโปรเจกต์ชื่อ “**Binary2Gray**” ขึ้นมาใหม่
- สร้างไฟล์ชิ้นใหม่ให้มีชื่อตรงกับชื่อโปรเจกต์สำหรับเก็บงานวงจรแปลงรหัสเลขไบนารีไปเป็นรหัสเลขแบบเกรย์ ในรูปที่ 9
- กำหนดให้ใช้ชิพ FPGA เบอร์ **EP3C10E144C8**
- เขียนวงจรดังรูปที่ 9 ทำการคอมไพล์และสร้าง symbol ของวงจรไว้สำหรับการทดลองขั้นต่อไป

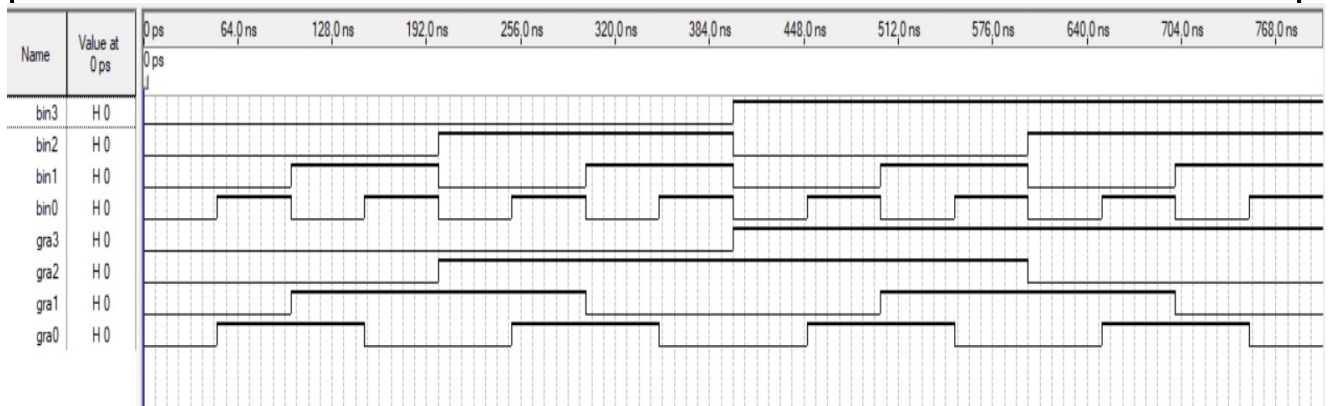
วงจรแปลงเลขไบนารีขนาด 4 บิตให้เป็น  
รหัสแบบ Gray code ขนาด 4 บิต

รูปที่ 9



11. สร้างไฟล์แสดงแผนภาพทางเวลา (Timing diagram) ดังรูปที่ 10 โดยกำหนดค่าสำหรับแสดงผลจำลองการทำงานโดยให้มีค่า End Time = 800 ns , Grid Size = 1 ns

- สัญญาณ “bin3 - bin0” กำหนดค่าเป็นแบบนาฬิกา ช่วงละ 50 ns. (bin0 มี period = 100ns)
- จำลองการทำงานโหมด Functional mode บันทึกผลลงในรูปที่ 10 และเขียนค่าลอจิกบนตารางความจริง



รูปที่ 10

### บันทึกผลการทดลอง

ก) อธิบายความสัมพันธ์กันของค่าไบนารี bin3-bin0 กับสถานะของลอจิกของ gra3-gra0 (อธิบายโดยใช้แผนภาพหรือสมการ)

$$gra3 = bin3$$

$$gra2 = bin3 \oplus bin2$$

$$gra1 = bin2 \oplus bin1$$

$$gra0 = bin1 \oplus bin0$$



ข) จากตารางความจริง **สังเกตการเปลี่ยนค่าของเลขไบนารี**

(เช่นจาก 0000 ไป 0001 เป็นต้น)

จุดที่มีการเปลี่ยนแปลงพร้อมกัน 2 บิต มี.....**4**.....จุดคือ

**0001 → 0010 , 0101 → 0110 ,**

**1001 → 1010 , 1101 → 1110**

จุดที่มีการเปลี่ยนแปลงพร้อมกัน 3 บิต มี.....**2**.....จุดคือ

**0011 → 0100 , 1011 → 1100**

จุดที่มีการเปลี่ยนแปลงพร้อมกัน 4 บิต มี.....**1**.....จุดคือ

**0111 → 1000**

ค) จากตารางความจริง **สังเกตการณ์เปลี่ยนค่าของรหัสเกรย์**

(gray code) มีการเปลี่ยนแปลงพร้อมกันมากกว่า 1 บิต

มี **0** จุดคือ **ไม่มีการเปลี่ยนแปลงพร้อมกัน > 1 บิต**

คุณสมบัติของ Gray Code ที่ดีกว่า Binary คือ

**↓ Delay time น้อยกว่า Binary มีโอกาสผิดพลาดน้อยกว่า Binary ที่มีการเปลี่ยนแปลงหลาย bits.**

Decimal	Binary				Gray			
	bin3	bin2	bin1	bin0	gra3	gra2	gra1	gra0
00	0	0	0	0	0	0	0	0
01	0	0	0	1	0	0	0	1
02	0	0	1	0	0	0	1	1
03	0	0	1	1	0	0	1	0
04	0	1	0	0	0	1	1	0
05	0	1	0	1	0	1	1	1
06	0	1	1	0	0	1	0	1
07	0	1	1	1	0	1	0	0
08	1	0	0	0	1	1	0	0
09	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

ลายเซ็นอาจารย์ผู้ควบคุม..... /...../.....

12. **ปิดโปรเจกต์**ที่สร้างมาในขั้นตอนที่ 10 – 12 ก่อนจะทำการทดลองต่อไป

13. a) ให้สร้างโปรเจกต์ชื่อ **“Gray2Bin”** และสร้างไฟล์ขึ้นมาเก็บงาน

ออกแบบวงจรในรูปที่ 11

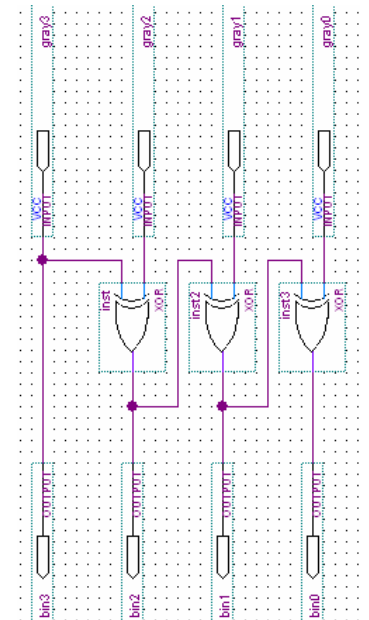
b) ให้บันทึกไฟล์ไว้ในโฟลเดอร์เดียวกันกับงานทดลองก่อนหน้านี้

c) ให้ใช้ชิพเบอร์ EP3C10E144C8

d) ทำการคอมไพล์และสร้าง symbol ของวงจรขึ้นมาไว้สำหรับการทดลองต่อไป

วงจรแปลงรหัสแบบ Gray code ขนาด 4 บิต  
ให้เป็นเลขไบนารีขนาด 4 บิต

รูปที่ 11



14. สร้างไฟล์แสดงแผนภาพทางเวลา กำหนดค่าของพารามิเตอร์สำหรับแสดงผลจำลองการทำงานโดยให้มีค่า

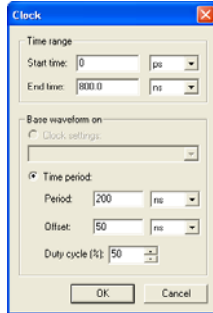
End Time = 800 ns      Grid Size = 1 ns

สร้างรูปคลื่นของสัญญาณ gra0 gra1 gra2 gra3 ให้เป็นแบบนาฬิกาตั้งในรูปที่ 12 และ**ทำการกลับเฟสของสัญญาณ**ด้วยเครื่องมือบนเมนูดังในรูปที่ 13



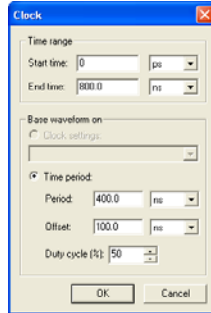


สัญญาณ gra0



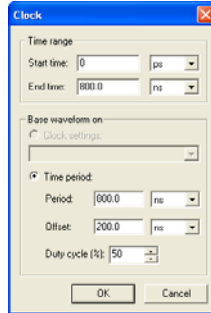
a

สัญญาณ gra1



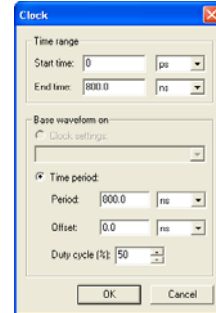
b

สัญญาณ gra2

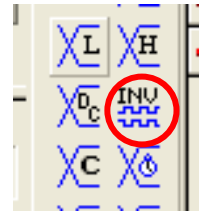


c

สัญญาณ gra3



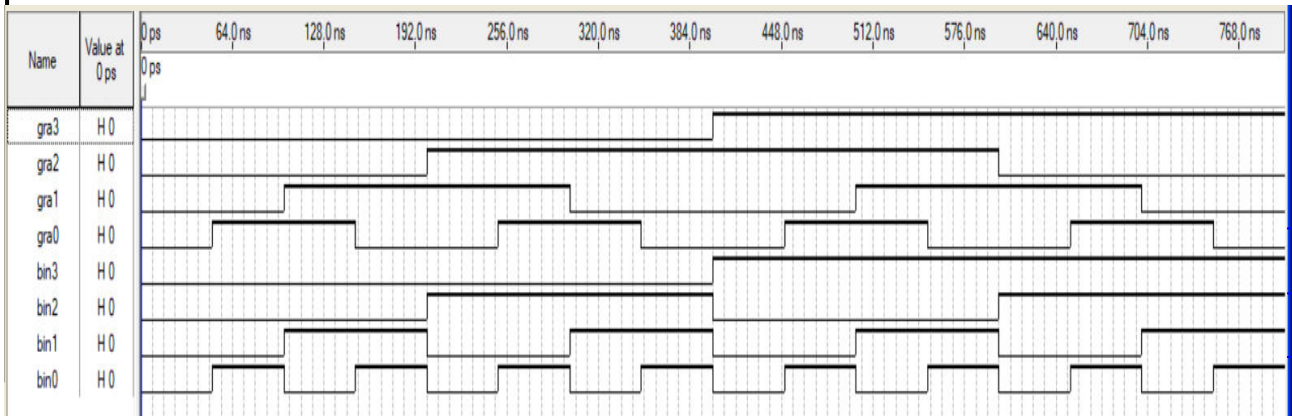
d



รูปที่ 13

รูปที่ 12

15. จำลองการทำงานด้วยโหมด “Functional mode” บันทึกผลที่ได้ลงในรูปที่ 14 และตารางความจริง



รูปที่ 14

## บันทึกผลการทดลอง

อธิบายความสัมพันธ์กันของค่าไบนารี bin3-bin0 กับสถานะของลอจิกของ gra3-gra0 (อธิบายโดยใช้แผนภาพหรือสมการ)

$$\begin{aligned} \text{bin3} &= \text{gra3} \\ \text{bin2} &= \text{gra3} \oplus \text{gra2} \\ \text{bin1} &= \text{gra3} \oplus \text{gra2} \oplus \text{gra1} \\ \text{bin0} &= \text{gra3} \oplus \text{gra2} \oplus \text{gra1} \oplus \text{gra0} \end{aligned}$$

ลายเซ็นอาจารย์ผู้ควบคุม ..... /.../.....

## งานมอบหมายท้ายการทดลอง

(ให้เขียนลงบนกระดาษ A4 ที่มีเส้นบรรทัดและรวมใส่ท้ายเอกสารการทดลอง ส่งอาจารย์ผู้สอนในคราวถัดไป)

1. ให้เขียนภาษา VHDL สำหรับแปลงรหัสระหว่าง

Gray Code  $\Rightarrow$  Binary

และ Binary  $\Rightarrow$  Gray Code

Decimal	Gray				Binary			
	gra3	gra2	gra1	gra0	bin3	bin2	bin1	bin0
00	0	0	0	0	0	0	0	0
01	0	0	0	1	0	0	0	1
02	0	0	1	1	0	0	1	0
03	0	0	1	0	0	0	1	1
04	0	1	1	0	0	1	0	0
05	0	1	1	1	0	1	0	1
06	0	1	0	1	0	1	1	0
07	0	1	0	0	0	1	1	1
08	1	1	0	0	1	0	0	0
09	1	1	0	1	1	0	0	1
10	1	1	1	1	1	0	1	0
11	1	1	1	0	1	0	1	1
12	1	0	1	0	1	1	0	0
13	1	0	1	1	1	1	0	1
14	1	0	0	1	1	1	1	0
15	1	0	0	0	1	1	1	1

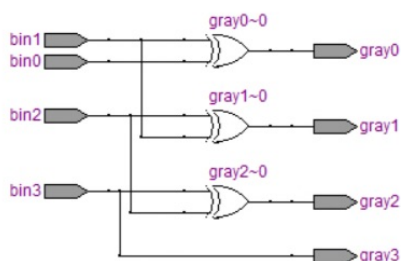
งานมอบหมายทำบทรหัส

# ① VHDL สำหรับแปลงรหัสจาก Binary → Gray Code

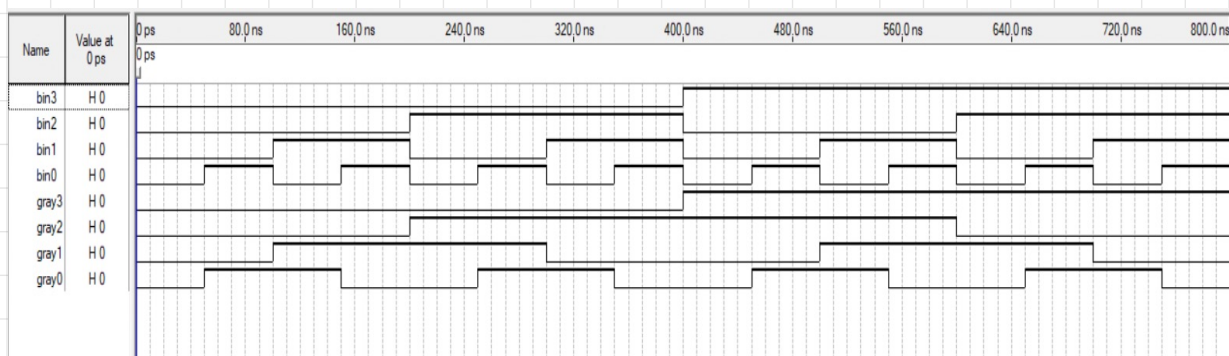
```

1  LIBRARY ieee ;
2  USE ieee.std_logic_1164.all ;
3  ENTITY VHDL_bin2gray IS
4  PORT ( bin3 , bin2 , bin1 , bin0      : IN STD_LOGIC ;
5        gray3 , gray2 , gray1 , gray0 : OUT STD_LOGIC ) ;
6
7  END VHDL_bin2gray ;
8
9  ARCHITECTURE Behavior OF VHDL_bin2gray IS
10 BEGIN
11     gray3 <= bin3 ;
12     gray2 <= bin3 xor bin2 ;
13     gray1 <= bin2 xor bin1 ;
14     gray0 <= bin1 xor bin0 ;
15
16 END behavior ;
    
```

(Name : VHDL\_bin2gray)



(RTL viewer for Bin2 Gray)



(Timing Diagram functional mode)

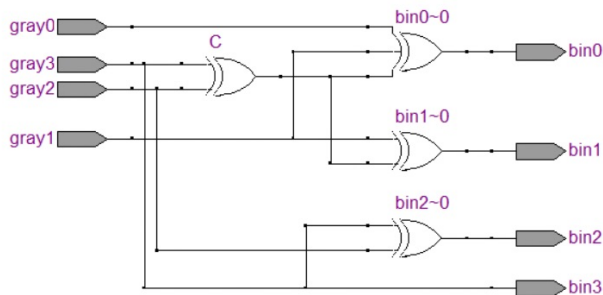
## ② VHDL สับรวมการแปลงรหัส Gray Code → Binary

```

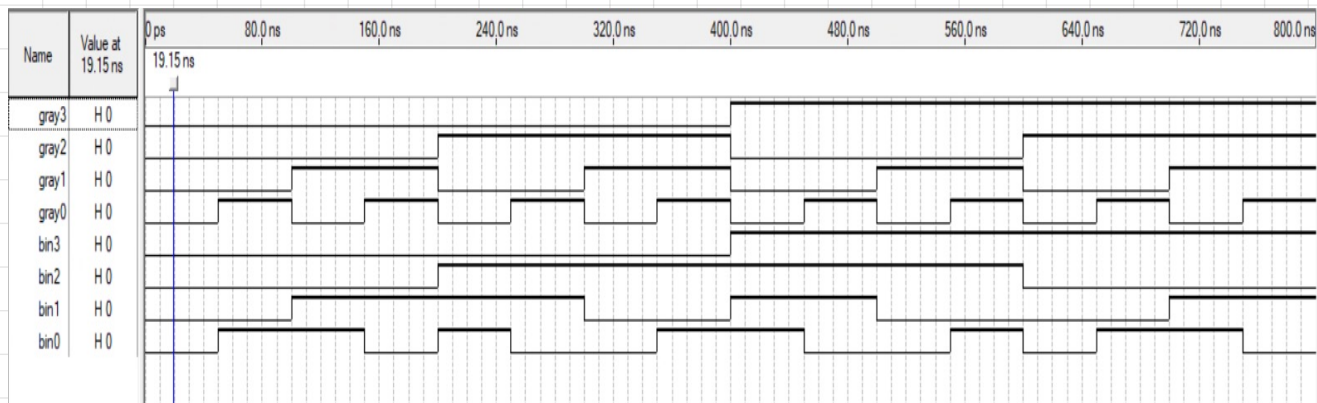
1  LIBRARY ieee ;
2  USE ieee.std_logic_1164.all ;
3
4  ENTITY VHDL_GRAY2BIN IS
5  PORT (gray3 , gray2 , gray1 , gray0 : IN STD_LOGIC ;
6        bin3 , bin2 , bin1 , bin0    : OUT STD_LOGIC);
7
8
9  END VHDL_GRAY2BIN ;
10
11 ARCHITECTURE behavior OF VHDL_GRAY2BIN IS
12     SIGNAL C,X : STD_LOGIC ;
13 BEGIN
14
15     C  <= gray3 xor gray2 ;
16     X  <= C xor gray1;
17     bin3 <= gray3 ;
18     bin2 <= gray3 xor gray2 ;
19     bin1 <= C xor gray1 ;
20     bin0 <= X xor gray0 ;
21
22 END behavior ; |

```

(Name : VHDL\_GRAY2BIN)



(RTL viewer for gray2bin)



(Timing Diagram Functional mode)