



การทดลองที่ 7 Registers and Counters

หน้า

1 / 13

ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

ภาคการศึกษาที่ 1 ปีการศึกษา 2564

รหัสวิชา 010113026 ชื่อวิชา Digital Laboratory

ตอนเรียน 4 หมายเลขอั้ง 1

รหัสนักศึกษา 620101163 ๗๘๘

ชื่อ-นามสกุล วงศ์สวัสดิ์ สุขลักษณ์รุจัน

อาจารย์ผู้สอน CSP

เวลาที่ทำการทดลอง 13.00 - 16.00 วันที่ 16/09/64

การทดลองที่ 7

Registers and Counters

วัตถุประสงค์

- เพื่อให้สามารถใช้โปรแกรมคอมพิวเตอร์จำลองการทำงานของวงจรลอจิกเกทได้
- เพื่อให้เข้าใจพื้นฐานของอุปกรณ์ประเภท shift-register
- เพื่อให้เข้าใจพื้นฐานของอุปกรณ์ประเภท counter
- เพื่อให้สามารถใช้งานบอร์ดทดลอง Cyclone3-Lab01 ได้

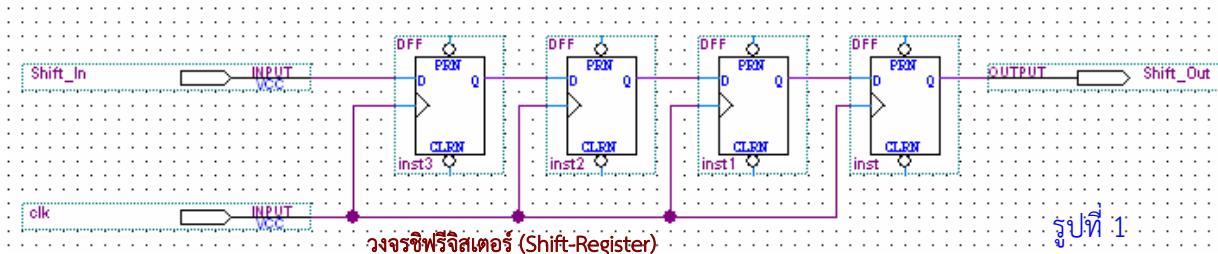
อุปกรณ์

- ระบบคอมพิวเตอร์ 1 เครื่อง พร้อมติดตั้งโปรแกรม Quartus II เวอร์ชัน 8.0 (Student Edition) ขึ้นไป
- บอร์ดทดลอง Cyclone3-Lab01 1 บอร์ดพร้อมคู่มือการใช้งาน
- สาย J-TAG 1 เส้น ใช้รุ่น USB-Blaster (สำหรับเครื่อง Notebook) หรือรุ่น Byte-Blaster (สำหรับเครื่อง PC)

การทดลองตอนที่ 1 วงจร Shift-Register

คำสั่งการทดลอง

- ให้สร้างไฟล์เดอร์สำหรับเก็บงานการทดลองนี้ชื่อ “Lab07_Counter”
ให้สร้างโปรเจคชื่อ “ShiftRegister” ใช้ชิป FPGA เบอร์ EP3C10E144C8

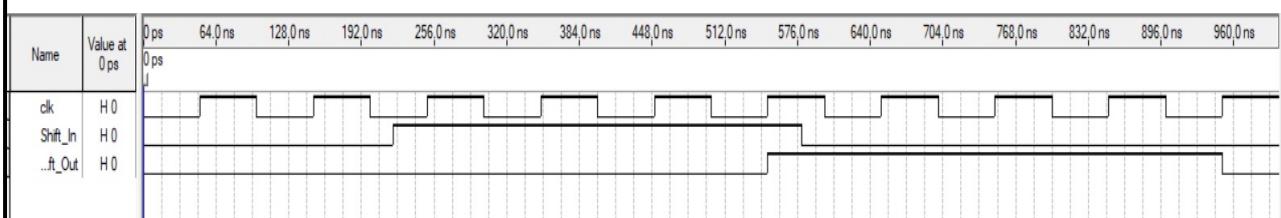


- เขียนวงจรชีฟรีจิสเตอร์ (Shift-Register) ในรูปที่ 1 ด้วยไฟล์พลอปชนิด D-FF คอมไมล์ และสร้าง symbol file ของวงจรขึ้นมาเตรียมไว้ใช้งานในขั้นถัดไป
- สร้างไฟล์แผนภาพทางเวลา ตั้งค่าจำลองการทำงานโดยให้ End Time = 1.0 us, Grid Size = 1 ns กำหนดอินพุท clk ให้เป็นแบบสัญญาณนาฬิกาเม็ด (period) = 100 ns Shift_In ให้เป็นโลจิก ‘1’ ในช่วง 220 – 580 ns นอกนั้นให้เป็น ‘0’ จำลองการทำงานใหม่ Functional mode บันทึกผลที่ได้ลงในรูปที่ 2



การทดลองที่ 7 Registers and Counters

หน้า
2 / 13



รูปที่ 2

ผลการทดลอง

ก) จากกราฟรูปที่ 2

อินพุทที่ขา Shift_In = '1' ที่ช่วง $t = 220 - 580 \text{ ns}$ มีความกว้างของพัลส์ = ns เอ้าท์พุทที่ขา Shift_Out = '1' ที่ช่วง $t = 580 - 950 \text{ ns}$ มีความกว้างของพัลส์ = ns

เพราะเหตุให้ความกว้างพัลส์ของสัญญาณเอ้าท์พุทกับอินพุทจึงมีขนาดไม่เท่ากัน

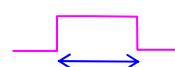
เนื่องจาก Input 'Shift_In' เป็นช่องแบ่งครึ่งที่ $t = 580 \text{ ns}$ แต่ใน logic D-flip-flop ทำงาน twice ทั้งขา ↑ ทำให้เกิดการเปลี่ยนแปลงเวลา เหลือ Pulse ของ Output จึงกว้างกว่า

ข) นับจาก Shift_In = '1' สัญญาณ clk ต้องเปลี่ยนขอบ '↑' (เปลี่ยนจาก 0 → 1) กี่ครั้ง สัญญาณเอ้าท์พุท จึงจะได้ค่าเท่ากันกับอินพุท 4 ครั้ง

เพราะเหตุใด Output จะเกิดการเปลี่ยนแปลง เนื่องจาก Input 当成 D-Flip-Flop 4 ครั้ง เนื่องจาก เป็นการแปลงแบบ Serial

หมายเหตุ :

ความกว้างของพัลส์



หมายถึงระยะเวลาเฉพาะส่วนที่สัญญาณมีค่า = '1'

ค) จากพฤติกรรมความสัมพันธ์กันระหว่างอินพุทกับเอ้าท์พุทในข้อ ก) และจำนวนของสัญญาณ clk ที่ต้องใช้ในการลำเลียงข้อมูลจากอินพุทไปยังเอ้าท์พุทในข้อ ข) ให้ น.ศ. สรุปความหมายของคำว่า “Shift” ว่าควรจะให้คำจำกัดความว่าอย่างไร การส่งงาน สั่งงานตาม Clock รุ่นเป็นการ串行แบบ SISI

ลายเซ็นอาจารย์ผู้ควบคุม..... / /

4. ให้ปิดโปรเจคที่สร้างมาในขั้นตอนที่ 1 – 2 ก่อนที่จะทำการทดลองต่อไป

a) ให้สร้างโปรเจคขึ้นใหม่ชื่อ “ParallelAccessShiftReg” และให้เก็บไว้ในโฟลเดอร์เดิม

b) เปิดไฟล์ขึ้นใหม่สำหรับเก็บงาน designed เพื่อเขียนวงจรทดลองในรูปที่ 3

c) ใช้ชิป EP3C10E144C8 และทำการคอมpileให้เรียบร้อย

5. สร้างไฟล์แสดงแผนภาพทางเวลา กำหนดค่าแสดงผลจำลองโดย End Time = 1.0 us, Grid Size = 1 ns

อินพุท clk ให้เป็นแบบสัญญาณนาฬิกา period 40 ns, offset 0

Serial_In ให้เป็นแบบสัญญาณนาฬิกา period 500 ns, offset 0,

nShift_Load = '1' ในช่วง $t = 90-110 \text{ ns}$ และช่วง $t = 450-470 \text{ ns}$

Parallel_In1 = Parallel_In2 = '1'

Parallel_In0 = Parallel_In3 = '0'

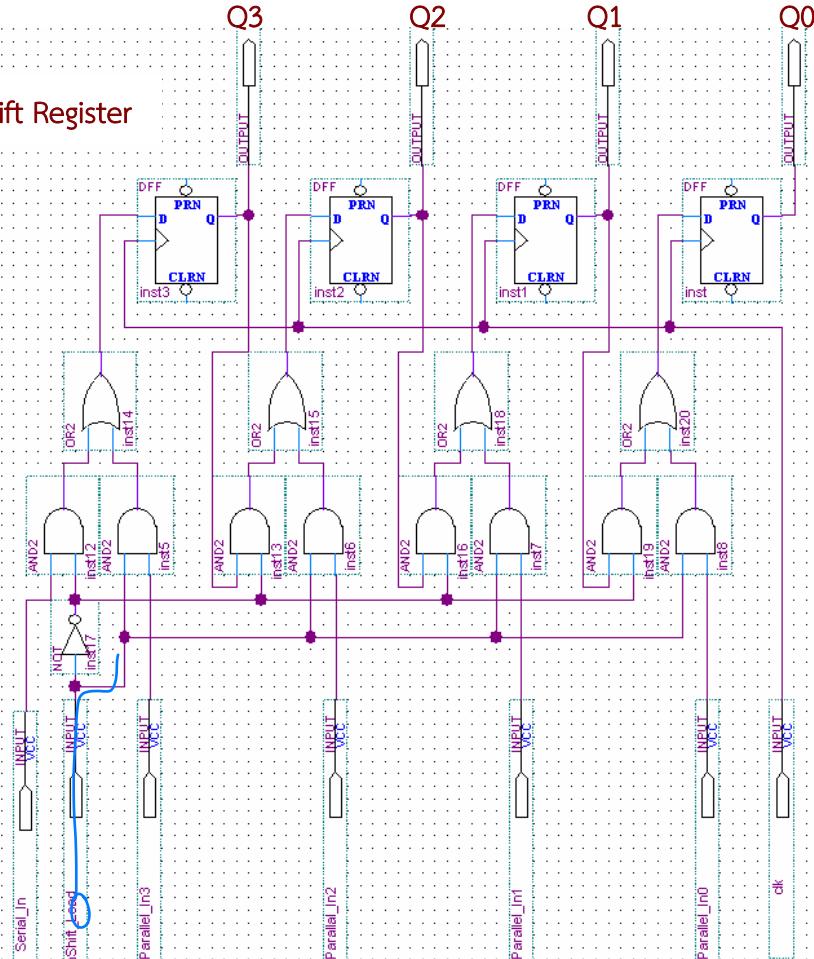


การทดลองที่ 7 Registers and Counters

หน้า
3 / 13

ทำการจำลองการทำงานโหมด Functional mode บันทึกผลที่ได้ลงในรูปที่ 4

ວຈຮ Parallel-Access Shift Register



รูปที่ 3



รูปที่ 4

ผลการทดลอง

- ก) เมื่อ clk เปลี่ยนขอบขาขึ้น ' \uparrow ' ครั้งที่ 1 ค่าของ Q3 = 0 Q2 = 0 Q1 = 0 Q0 = 0
 เมื่อ clk เปลี่ยนขอบขาขึ้น ' \uparrow ' ครั้งที่ 2 ค่าของ Q3 = 0 Q2 = 1 Q1 = 1 Q0 = 0

เมื่อ Clk เปิดยังชี้ปุ่มบน ครองที่ 3 ค่าของ Q3 = Q2 = Q1 = Q0 =

เพราเดตุไดค่าของ Q3,Q2,Q1,Q0 จึงเป็นเช่นนั้น

ค่า Q0 Input 'nshift-load' ลักษณะจะเป็น 1 ที่มาจากตัว CLK กด 3



การทดลองที่ 7 Registers and Counters

หน้า
4 / 13

ข) เมื่อ clk เปลี่ยนขอบขึ้น ‘↑’ ครั้งที่ 4 ค่าของ Q3= 0 Q2= 0 Q1= 1 Q0= 1

ค่าที่ปรากฏที่ Q3 เป็นค่าที่ได้มาจากการ Serial_in

ค่าที่ปรากฏที่ Q2 เป็นค่าที่ได้มาจากการ Q3

ค่าที่ปรากฏที่ Q1 เป็นค่าที่ได้มาจากการ Q2

ค่าที่ปรากฏที่ Q0 เป็นค่าที่ได้มาจากการ Q1

ค) เมื่อ clk เปลี่ยนขอบขึ้น ‘↑’ ครั้งที่ 5 ค่าของ Q3= 0 Q2= 0 Q1= 0 Q0= 1

ค่าที่ปรากฏที่ Q3 เป็นค่าที่ได้มาจากการ Serial_in

ค่าที่ปรากฏที่ Q2 เป็นค่าที่ได้มาจากการ Q3

ค่าที่ปรากฏที่ Q1 เป็นค่าที่ได้มาจากการ Q2

ค่าที่ปรากฏที่ Q0 เป็นค่าที่ได้มาจากการ Q1

ง) เมื่อ clk เปลี่ยนขอบขึ้น ‘↑’ ครั้งที่ 7 ค่าของ Q3= 1 Q2= 0 Q1= 0 Q0= 0

เพราะเหตุใดค่าของ Q3,Q2,Q1,Q0 จึงเป็นเช่นนี้ Serial_in สำหรับจังหวะที่ 1

จ) เมื่อ clk เปลี่ยนขอบขึ้น ‘↑’ ครั้งที่ 8 ค่าของ Q3= 1 Q2= 1 Q1= 0 Q0= 0

เพราะเหตุใดค่าของ Q3,Q2,Q1,Q0 จึงเป็นเช่นนี้ Q2 เก็บ住 Q3 shift ลง Serial_in

แล้ว Q3 ถูก serial_in เก็บ 1

ฉ) ความสัมพันธ์กันระหว่างอินพุต (ขา Serial_In, Parallel_In3,...,0) กับขาเอาท์พุต Q3,...,Q0 จะขึ้นอยู่กับการทำงานของขาควบคุมที่ชื่อ nShift_Load ดังนี้จากผลการทดลองในข้อ ก) ถึง ง) น.ศ. ควรจะสรุปหรือให้คำจำกัดความเกี่ยวกับหน้าที่ของขาควบคุมที่ชื่อ nShift_Load ว่าอย่างไร

nShift_load หน้าที่เดียวของการ Shift ลง Serial หรือ Parallel shift

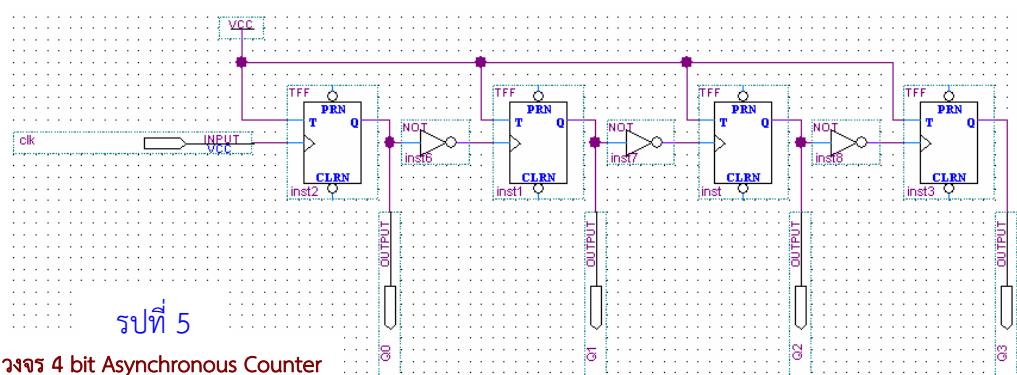
ลายเซ็นอาจารย์ผู้ควบคุม..... / /

การทดลองตอนที่ 2 วงจร Asynchronous Counter และวงจร Synchronous Counter

6. ให้ปิดโปรเจคที่สร้างมาในขั้นตอนที่ 4-5 ก่อนที่จะทำการทดลองต่อไป

a) ให้สร้างโปรเจคขึ้นใหม่ชื่อ “AsynchronousCounter” ให้เก็บไว้ในไฟล์เดอร์เดิม

b) ให้เปิดไฟล์ขึ้นใหม่สำหรับเขียนวงจรทดลองในรูปที่ 5 ใช้ชิป EP3C10E144C8 ทำการคอมไพล์และสร้าง symbol file





การทดลองที่ 7 Registers and Counters

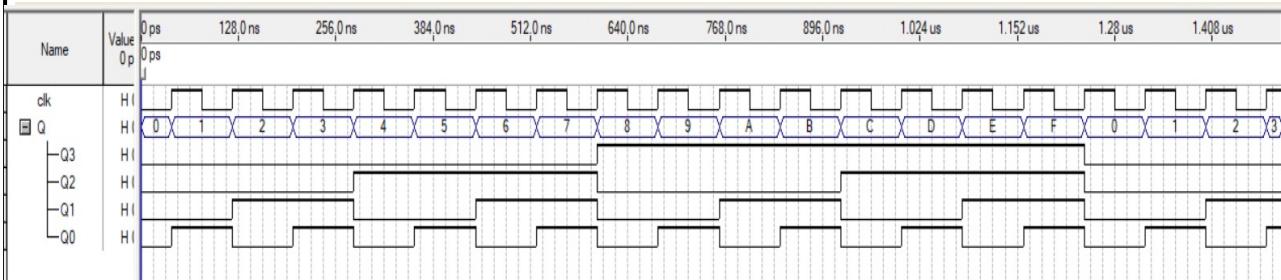
หน้า
5 / 13

7. สร้างไฟล์แสดงแผนภาพทางเวลา (Timing diagram) กำหนดค่าแสดงผลจำลองการทำงานโดยให้

End Time = 1.5 us Grid Size = 1 ns

อินพุท clk ให้เป็นแบบสัญญาณนาฬิกา period 80 ns, offset 0

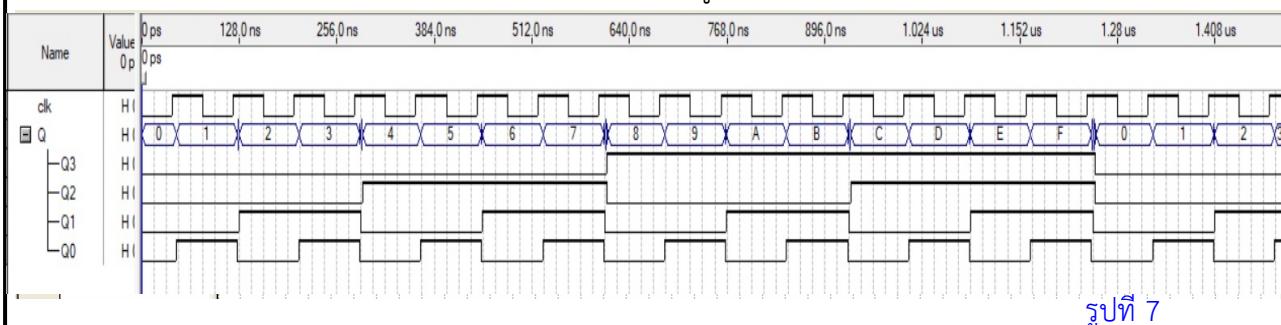
จำลองการทำงานโหมด Functional mode บันทึกผลที่ได้ลงในรูปที่ 6



หมายเหตุ ให้จัดสัญญาณเอ้าท์พุท Q3 Q2 Q1 Q0 เข้าเป็นกลุ่ม แสดงผลเป็น 4 บิตจะเด่นการเปลี่ยนแปลงค่าที่เอ้าท์พุทได้ถ่องแท้

รูปที่ 6

จำลองการทำงานโหมด Timing mode บันทึกผลที่ได้ลงในรูปที่ 7



รูปที่ 7

สังเกตผลการทดลอง

ก) จากรูปที่ 6 ให้สังเกตการเปลี่ยนแปลงค่าของเอ้าท์พุท Q เปรียบเทียบกับ **ขอบขาขึ้น ↑** ของ clk

เมื่อ clk ‘↑’ ครั้งที่ 1 ค่า Q3= 0 Q2= 0 Q1= 0 Q0= 1 $Q = Q_3Q_2Q_1Q_0 = 1$ (Hex.)

เมื่อ clk ‘↑’ ครั้งที่ 2 ค่า Q3= 0 Q2= 0 Q1= 1 Q0= 0 $Q = Q_3Q_2Q_1Q_0 = 2$ (Hex.)

เมื่อ clk ‘↑’ ครั้งที่ 3 ค่า Q3= 0 Q2= 0 Q1= 1 Q0= 1 $Q = Q_3Q_2Q_1Q_0 = 3$ (Hex.)

เมื่อ clk ‘↑’ ครั้งที่ 8 ค่า Q3= 1 Q2= 0 Q1= 0 Q0= 0 $Q = Q_3Q_2Q_1Q_0 = 8$ (Hex.)

ข) จากรูปที่ 7 ค่าของเอ้าท์พุท (ผลลัพธ์ที่ถูกต้อง) โดย

ใช้เวลาน้อยที่สุดเมื่อ clk= ‘↑’ ครั้งที่ 1 โดย Q เปลี่ยนจากเลข 0 ไปเป็นเลข 1 ใช้เวลา 6.196 ns

ใช้เวลามากที่สุดเมื่อ clk= ‘↑’ ครั้งที่ 8 โดย Q เปลี่ยนจากเลข 7 ไปเป็นเลข 8 ใช้เวลา 10.878 ns

ให้ชุมชนขยายภาพดูในช่วงที่ Q เปลี่ยนระหว่าง 7⇒8 และ 0⇒F เอ้าท์พุทบิดได้ที่มี delay time มากที่สุด 7 → 8

เพราเหตุใด **ເນື້ອງການ ອັງວຽກໄປສະໜັບປະລຸງແປງງວອງ ສົມທູກການ clk ມາດຕະຖານີ flip-flop ກຳມະນຸຍາ**

ค) จากข้อ ข) ถ้าวงจรที่ใช้ในการทดลองในรูปที่ 6 เพิ่มขนาดเป็นการนับเลข 8 บิต ค่า delay time (เมื่อเทียบ

กับ clk= ‘↑’) มากที่สุดที่เกิดขึ้นในวงจร จะมีค่าเป็นอย่างไร อธิบาย

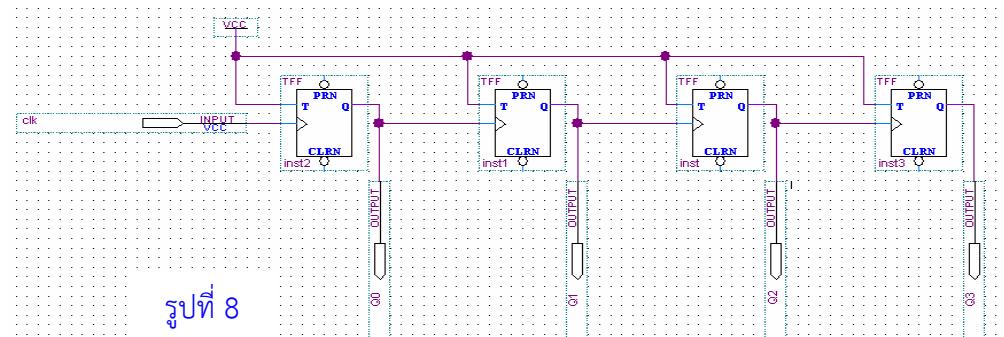
ເກີ່າມີຕາມໄປຕົວເທິງ flip-flop ລົງລະອົບ Delay time ຕໍ່ກຳມະນຸຍາ



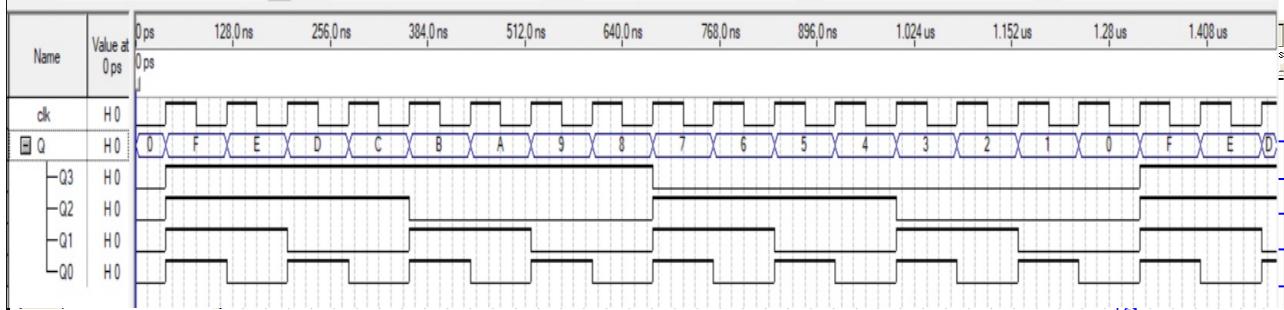
การทดลองที่ 7 Registers and Counters

หน้า
6 / 13

8. ทำการดัดแปลงวงจรในรูปที่ 5 ด้วยการเพล NOT gate ออกจากวงจรทั้งหมดจะได้ว่าจะดังรูปที่ 8



9. ทำการบันทึกไฟล์และคอมไพร์แล้วที่ดัดแปลงให้เรียบร้อย จำลองการทำงานโหมด Functional mode โดยไปอ่านไฟล์ที่ได้ตั้งไว้ เลือก parametrize ไปตามภาพทางเวลา ฯจังที่ก่อนที่ได้ลงไว้รุ่นที่ 9



ให้เปรียบเทียบลำดับค่าของอี๊พุท Q ระหว่างผลที่ได้จากรูปที่ 6 และรูปที่ 9 แตกต่างกันอย่างไร.....

Pic. 6 ห้องจาก 0- F

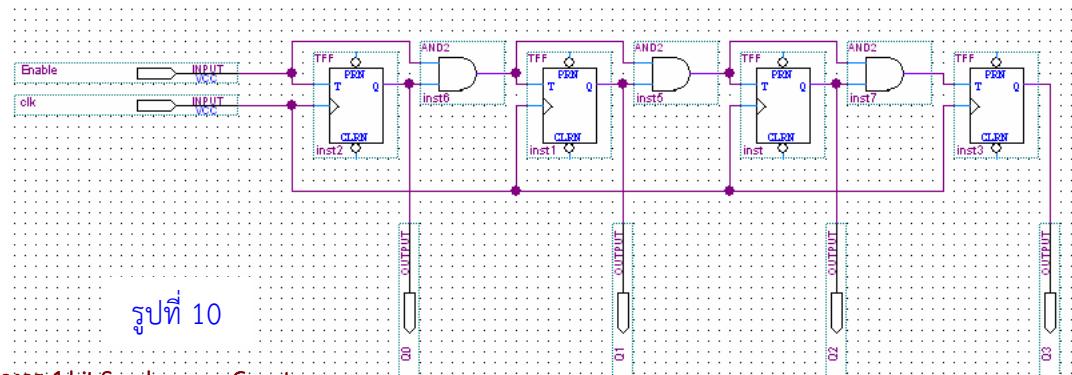
Pic. 9 ห้องมา F-0

ลายเซ็นอาจารย์ผู้ควบคุม..... //

10. ให้ปิดโปรเจกที่สร้างมาในขั้นตอนที่ 6-9 ก่อนที่จะทำการทดลองต่อไป

a) ให้สร้างโปรเจกขึ้นใหม่ชื่อ “**SynchronousCounter**” ให้เก็บไว้ในไฟล์เดอร์เดิม

b) ให้เปิดไฟล์ขึ้นใหม่สำหรับเขียนวงจรทดลองในรูปที่ 10 และทำการคอมไพล์ให้เรียบร้อย



วงจร 4 bit Synchronous Counter

11. สร้างไฟล์แสดงแผนภาพทางเวลา (Timing diagram) ให้ค่าของพารามิเตอร์สำหรับแสดงผลจำลองการทำงาน



การทดลองที่ 7 Registers and Counters

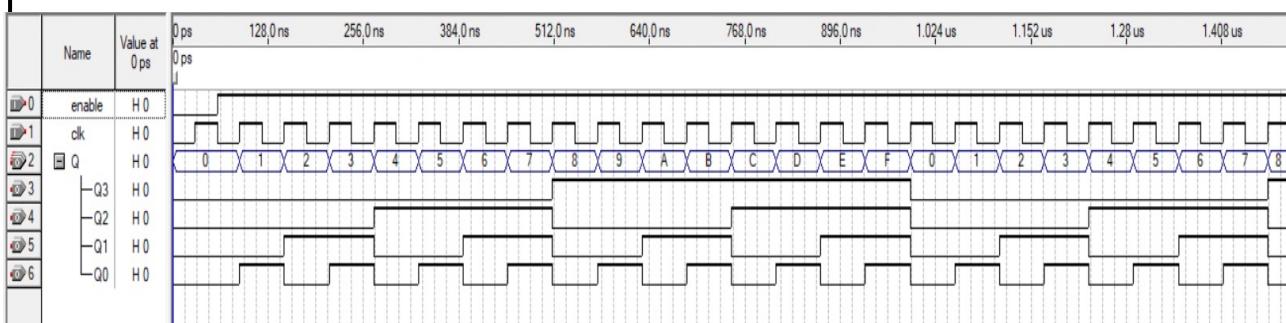
หน้า
7 / 13

มีค่า End Time = 1.5 us Grid Size = 1 ns

สัญญาณ clk ให้มีค่า period 60 ns, offset 0

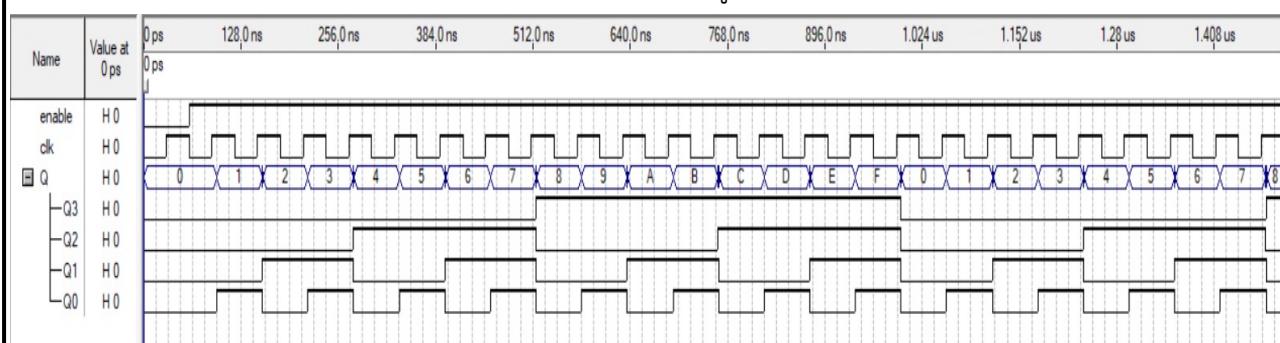
Enable = '1' ตั้งแต่เวลา 60 ns ไปจนสุดกราฟ

จำลองการทำงานโหมด Functional mode บันทึกผลที่ได้ลังในรูปที่ 11



จำลองการทำงานโหมด Timing mode บันทึกผลที่ได้ลังในรูปที่ 12

รูปที่ 11



สังเกตผลการทดลอง

ก) จากรูปที่ 11 ให้สังเกตการเปลี่ยนแปลงค่าของอั้วท์พุท Q เปรียบเทียบกับขอบขึ้น ' \uparrow ' ของ clk

เมื่อ clk ' \uparrow ' ครั้งที่ 1 ค่า Q3= 0 Q2= 0 Q1= 0 Q0= 0 $Q = Q_3Q_2Q_1Q_0 = \underline{\hspace{2cm}} 0 \underline{\hspace{2cm}}$ (Hex.)

เมื่อ clk ' \uparrow ' ครั้งที่ 2 ค่า Q3= 0 Q2= 0 Q1= 1 Q0= 1 $Q = Q_3Q_2Q_1Q_0 = \underline{\hspace{2cm}} 1 \underline{\hspace{2cm}}$ (Hex.)

เมื่อ clk ' \uparrow ' ครั้งที่ 3 ค่า Q3= 0 Q2= 0 Q1= 1 Q0= 0 $Q = Q_3Q_2Q_1Q_0 = \underline{\hspace{2cm}} 2 \underline{\hspace{2cm}}$ (Hex.)

เมื่อ clk ' \uparrow ' ครั้งที่ 8 ค่า Q3= 0 Q2= 1 Q1= 1 Q0= 1 $Q = Q_3Q_2Q_1Q_0 = \underline{\hspace{2cm}} 4 \underline{\hspace{2cm}}$ (Hex.)

ข) จากรูปที่ 12 ค่าของอั้วท์พุท (ผลลัพธ์ที่ถูกต้อง) โดย

ใช้เวลาน้อยที่สุดเมื่อ clk= ' \uparrow ' ครั้งที่ 3 โดย Q เปลี่ยนจากเลข 1 ไปเป็นเลข 2 ใช้เวลา 6.182 ns

ใช้เวลามากที่สุดเมื่อ clk= ' \uparrow ' ครั้งที่ 9 โดย Q เปลี่ยนจากเลข 4 ไปเป็นเลข 8 ใช้เวลา 6.432 ns

ให้ชุมชนขยายภาพดูในช่วงที่ Q เปลี่ยนระหว่าง 7 \Rightarrow 8 หรือ 0 \Rightarrow F อั้วท์พุทบิตใดเปลี่ยนแปลงชาที่สุด 7 \rightarrow 8

เหตุการณ์นี้เหมือนหรือแตกต่างจากการจrnับแบบ Asynchronous Counter ในรูปที่ 7 อย่างไร

Synchronous vs Delay time ห้องก่อ (6.432 : 10.876)

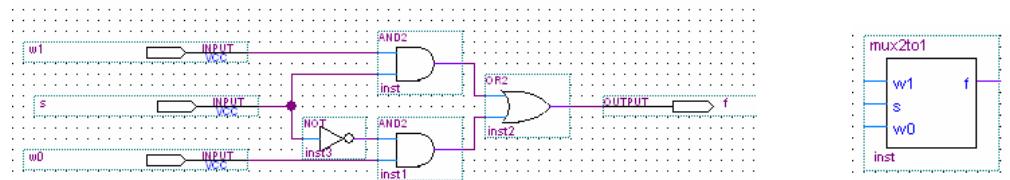
เพราะเหตุใด วงจรໄວ่ได้รับ clock นานกัน ทำให้ไม่ต้องรอ clk บน flip-flop ก่อไปเลย

ลายเซ็นอาจารย์ผู้ควบคุม..... //



การทดลองตอนที่ 3 วงจร Synchronous Counter ที่สามารถตั้งค่าเริ่มต้นการนับได้

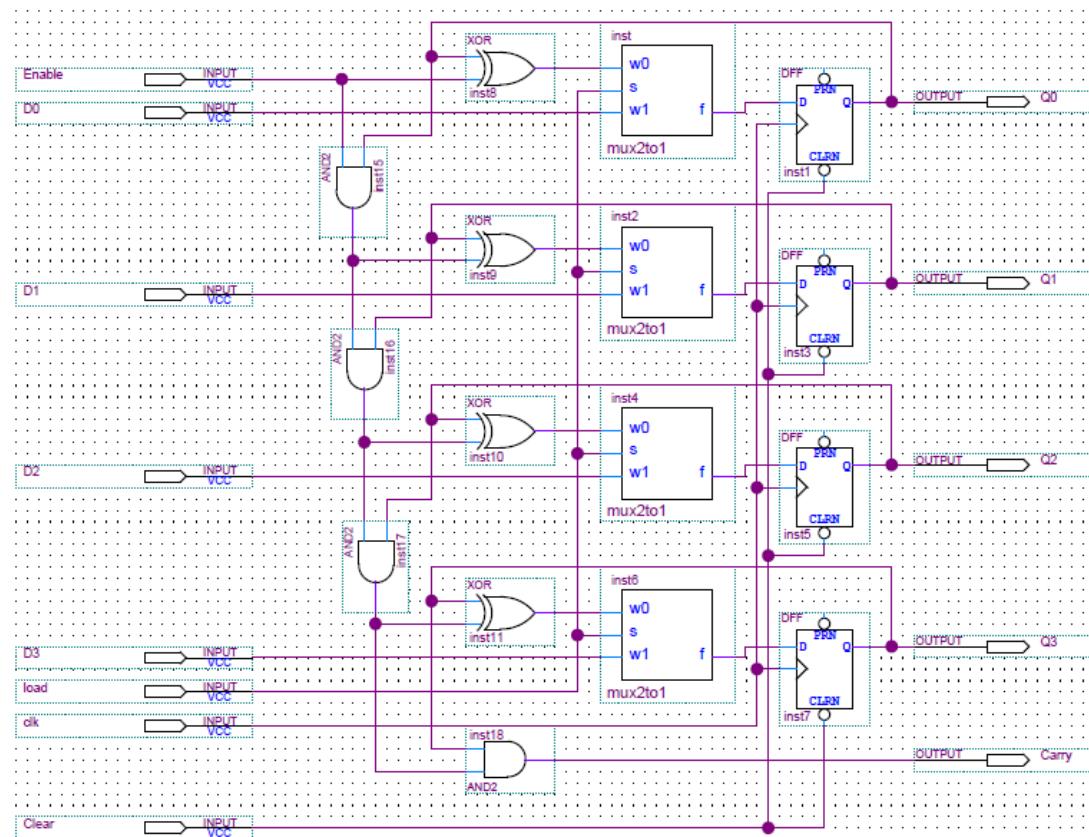
12. ให้ปิดโครงการที่สร้างมาในขั้นตอนที่ 10-11 ก่อนที่จะทำการทดลองต่อไป
13. ให้สร้างอุปกรณ์ 2-to-1 multiplexer เพื่อที่จะใช้ในการทดลองโดย
 - ให้สร้างโปรเจกขึ้นใหม่ชื่อ “**mux2to1**” และให้เก็บไว้ในไฟล์เดอร์เดิม
 - ให้เปิดไฟล์ใหม่สำหรับเขียนวงจรในรูปที่ 13 ใช้ชิป EP3C10E144C8 และคอมไฟล์ให้เรียบร้อย
 - สร้าง **symbol file** เพื่อเตรียมใช้งานในขั้นตอนที่ 14 จากนั้นให้ปิดโครงการ



วงจร multiplexer 2-to-1 (น.ศ.เคยทำมาแล้วในการทดลองที่ 4 แต่มีขนาดเล็กกว่า)

รูปที่ 13

14. ให้ปิดโครงการที่สร้างมาในขั้นตอนที่ 13 และให้ดำเนินการดังนี้
 - สร้างโปรเจกขึ้นใหม่ชื่อ “**Counter_Parallel_Load**” ให้เก็บไว้ในไฟล์เดอร์เดิม
 - ให้เปิดไฟล์ใหม่สำหรับเขียนวงจรทดลองในรูปที่ 14 ใช้ชิป EP3C10E144C8 ทำการคอมไฟล์ และให้สร้าง **symbol** ไว้เพื่อใช้ในข้อถัดไป



คำเตือน : โปรดระวังการต่อขาของอุปกรณ์ mux2to1 ในวงจรสลับขา กัน

รูปที่ 14



การทดลองที่ 7 Registers and Counters

หน้า
9 / 13

15. สร้างไฟล์แสดงแผนภาพทางเวลา (Timing diagram) ค่าของการจำลองแสดงผลดังนี้

End Time = 1.0 us **Grid Size** = 1 ns **clk** = period 50 ns, offset 0

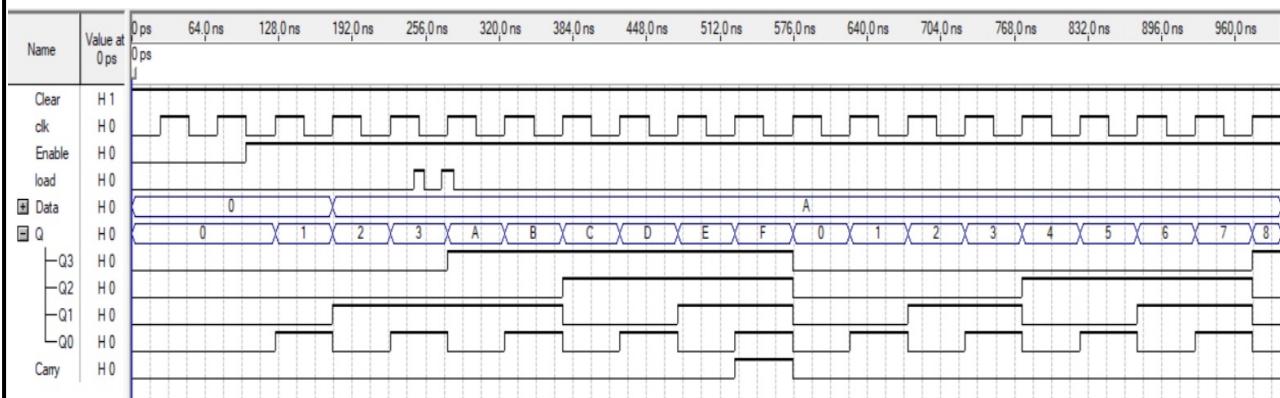
Enable = '1' ที่ช่วงเวลา $t= 100\text{ns} - 1.0 \text{ us}$

Clear = '1' ตลอดเวลา

Load = '1' ที่ช่วงเวลา $t= 246-255\text{ns}$ และ $t= 270-280 \text{ ns}$ นอกนั้นให้เป็น '0'

Data (กลุ่มของ D3,D2,D1,D0) = "0000" ช่วง $t= 0-175 \text{ ns}$ และ "1010" ช่วง $t= 175-1000 \text{ ns}$

จำลองการทำงานโดย mode Functional mode บันทึกผลที่ได้ลงในรูปที่ 15



รูปที่ 15

การทดลอง

ก) การเปลี่ยนแปลงของอัตราพุท Q ในช่วงที่ **Enable** = '0'

เมื่อ clk '↑' ครั้งที่ 1 ค่า Carry= 0 Q3= 0 Q2= 0 Q1= 0 Q0= 0 Q = 0 (Hex.)

เมื่อ clk '↑' ครั้งที่ 2 ค่า Carry= 0 Q3= 0 Q2= 0 Q1= 0 Q0= 0 Q = 0 (Hex.)

ข) การเปลี่ยนแปลงของอัตราพุท Q ในช่วงที่ **Enable** = '1'

เมื่อ clk '↑' ครั้งที่ 3 ค่า Carry= 0 Q3= 0 Q2= 0 Q1= 0 Q0= 1 Q = 1 (Hex.)

เมื่อ clk '↑' ครั้งที่ 4 ค่า Carry= 0 Q3= 0 Q2= 0 Q1= 1 Q0= 0 Q = 2 (Hex.)

เมื่อ clk '↑' ครั้งที่ 5 ค่า Carry= 0 Q3= 0 Q2= 0 Q1= 1 Q0= 1 Q = 3 (Hex.)

ค) การเปลี่ยนแปลงของอัตราพุท Q ในช่วงที่ **Enable** = '1' และค่าของ **Load** = '1'

เมื่อ clk '↑' ครั้งที่ 6 ค่า Carry= 0 Q3= 1 Q2= 0 Q1= 1 Q0= 0 Q = A (Hex.)

ง) ในช่วงเวลา $t = 246-255 \text{ ns}$ จะมีค่า Enable = '1' และค่าของ Load = '1'

เหตุการณ์นี้ส่งผลต่ออัตราพุทเหมือนกับในข้อ ก) หรือไม่ เพราะเหตุใด ในช่วงเวลา $t = 246-255 \text{ ns}$ จะส่งผลต่อ clk หากเป็นไปได้ การฟลัชจะหายไป

16. ให้เปลี่ยนค่าสัญญาณที่ขา **Clear** = '0' ในช่วง $t=760-780 \text{ ns}$

เกิดผลอย่างไรต่ออัตราพุท..... Output หลักไฟลัชเป็น 0 (Hex.) นาน

$t = 760-780 \text{ ns}$

สัญญาณ Clear เมื่อเปรียบเทียบกับขอบ '↑' ของ clk สัญญาณได้มีความสำคัญต่ออัตราพุทมากกว่ากัน เพราะเหตุใด หากต้องการให้ตัว Clear ทำงานได้ต้องต่อขา Clear ให้อยู่ในจังหวะต่อไป

เมื่อ Clear ว่างอยู่ด้วย 0 จังหวะต่อไปจะต้องต่อตัว Clear ให้ต่อต่อไป

เมื่อ clk ต้องเป็น 1 ก่อน ถ้าเช่นอาจารย์ผู้ควบคุม..... //

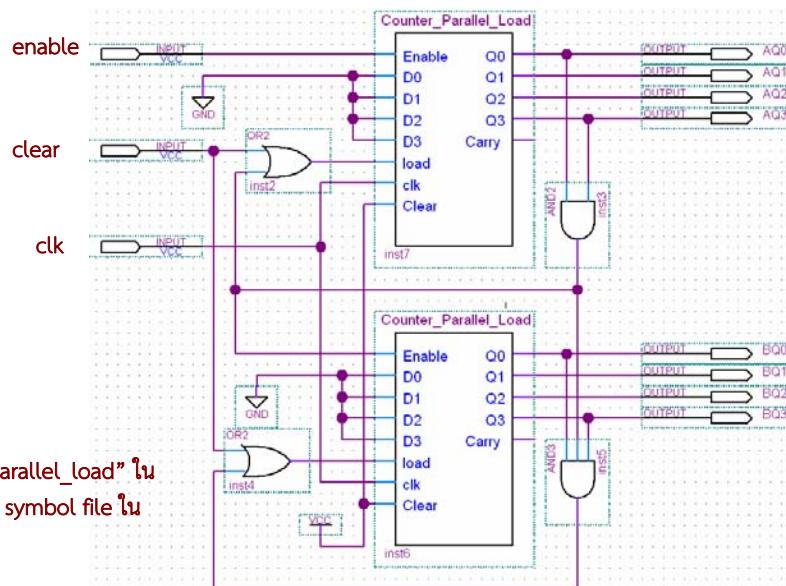


การทดลองตอนที่ 4 วงจรนับแบบ BCD Counter สำหรับนับเลขได้ตั้งแต่ 0 ถึง 99 (2 digits)

17. ให้ปิดโปรเจคที่สร้างมาในขั้นตอนที่ 15 ก่อนที่จะทำการทดลองต่อไป

a) ให้สร้างโปรเจคขึ้นใหม่ชื่อ “BCD_Counter2Digit” เก็บไว้ในโฟลเดอร์เดิม

b) ให้เปิดไฟล์ใหม่สำหรับเขียนวงจรทดลองในรูปที่ 16 ใช้ชิป EP3C10E144C8 ทำการคอมไพล์ให้เรียบร้อย



18. สร้างไฟล์แสดงแผนภาพทางเวลา แสดงผลจำลองการทำงานดังนี้

End Time = 1.0 us Grid Size = 1 ns

clk = period 25 ns, offset 0

Enable = '1' ตั้งแต่ 50 ns จนสุดกราฟ

Clear = '1' เฉพาะที่เวลา t=705-720 ns

ให้จัดกลุ่มสัญญาณ $AQ = AQ_3AQ_2AQ_1AQ_0$ และ $BQ = BQ_3BQ_2BQ_1BQ_0$

จำลองการทำงานโดย Functional mode บันทึกผลที่ได้ลงในรูปที่ 17



สังเกตผลการทดลอง

ก) การเปลี่ยนแปลงของเอาท์พุท QB และ QA ในช่วงที่ Enable = '1'

เมื่อ clk '↑' ครั้งที่ 1 ค่า QB= 0 QA= 0เมื่อ clk '↑' ครั้งที่ 2 ค่า QB= 0 QA= 0

ข) การเปลี่ยนแปลงของเอาท์พุท QB และ QA ในช่วงที่ Enable = '1'

เมื่อ clk '↑' ครั้งที่ 3 ค่า QB= 0 QA= 1เมื่อ clk '↑' ครั้งที่ 4 ค่า QB= 0 QA= 2เมื่อ clk '↑' ครั้งที่ 5 ค่า QB= 0 QA= 3



จากการทดลองในข้อ ก) และ ข) ขาควบคุมชื่อ enable มีหน้าที่อะไรในวงจร

เป็นตัวกำหนดการทำงานของวงจร ตัว enable มีค่า 0 ให้ทำงาน & enable มีค่า 1 หายใจ

ก) การเปลี่ยนแปลงของอัตโนมัติ Q ในช่วงที่ Enable = '1' และค่าของ clear = '1'

เมื่อ clk ขอบขึ้น ' \uparrow ' ครั้งที่ 29 ค่า QB=..... 0 QA=..... 0

จากการทดลองในข้อ ค) ขาควบคุมชื่อ clear มีหน้าที่อะไรในวงจร **ทำให้ Clear ที่ Output ห้ามลง 0 (Hex.) ให้เริ่มทำงานต่อไปแล้วถ้าคราวนี้清 ลง clear เท่า '0'**

ลายเซ็นอาจารย์ผู้ควบคุม..... //

การทดลองตอนที่ 5 สร้างวงจรนับ Binary Counter 8 bits ลงบนบอร์ดทดลอง Cyclone3-Lab01

ขั้นเตรียมการ

เนื่องจากบอร์ดทดลอง Cyclone3-Lab01 มีสัญญาณ clock ให้มาบนบอร์ด **20 MHz** (มีการเปลี่ยนแปลงสลับกันระหว่าง '1' และ '0' 20 ล้านครั้งใน 1 วินาที) หากเรานำสัญญาณนี้มาป้อนให้กับวงจร Binary Counter 8 bits โดยตรงจะทำให้เกิดการนับที่เร็วมากเกินกว่าที่สายตาของมนุษย์จะอ่านค่าของตัวเลขได้ทัน ดังนั้นเพื่อให้สามารถสังเกตการณ์นับได้ จึงต้องลดค่าความถี่ของสัญญาณ clock นี้ให้ช้าลง **เหลือประมาณ 1-16 Hz**

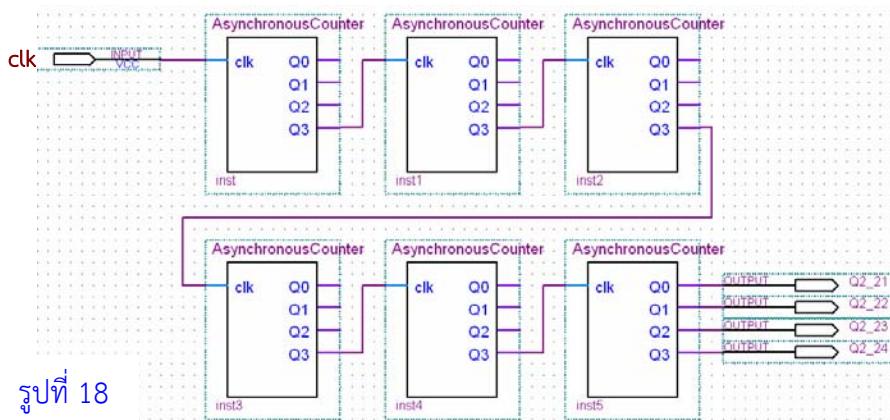
การลดค่าความถี่ของ clock ทำได้โดยการประยุกต์ใช้งาน asynchronous counter ในรูป ที่ 5 (ซึ่งเราได้สร้าง symbol file ไว้แล้วในขั้นตอนที่ 6) สาเหตุที่ใช้งาน asynchronous counter ก็เพราะหากเราสังเกตผลการทดลองในรูปที่ 6 จะพบว่าค่าของสัญญาณ Q3 มีค่าเป็น 2 เท่าของ Q2 และเป็น 16 เท่าของ clk ที่ป้อนเข้าไป เท่ากับว่าค่าความถี่ของ Q3 จะน้อยกว่า clk อยู่ 16 เท่า ดังนั้นหากเรา narrowing หรือทำการต่อ cascade กันในจำนวนที่มากพอ ก็จะสามารถลดความถี่ของ clock ให้มีค่าที่ต่ำลงได้ตามต้องการเช่นกัน

ขั้นที่ 1 สร้างวงจร clock divider ที่สามารถลดความถี่ clk ลงจาก 20MHz ให้เหลือ 1.2, 2.4, 4.8, 9.6 Hz

19. ให้ปิดโปรเจคที่สร้างมาในขั้นตอนที่ 17- 18 ก่อนที่จะทำต่อไป

a) ให้สร้างโปรเจคขึ้นใหม่ใช้ชื่อ “ClockDivider” เก็บไว้ในโฟลเดอร์เดิม

b) เปิดไฟล์ขึ้นมาสำหรับเขียนวงจร “หารสัญญาณนาฬิกา” (clock divider) ในรูปที่ 18 โดยอุปกรณ์ที่ใช้คือตัว Asynchronous Counter (จากขั้นตอนที่ 6) ให้คอมpile และสร้าง symbol file ไว้ในขั้นตอนถัดไป





การทดลองที่ 7 Registers and Counters

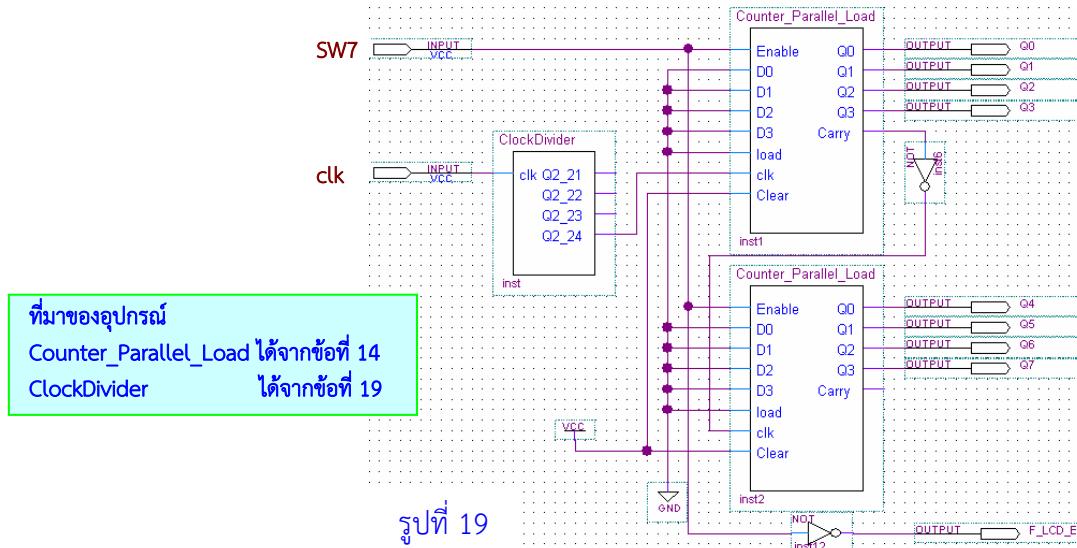
หน้า
12 / 13

ขั้นที่ 2 สร้างวงจร Binary Counter 8 bits มีขาควบคุมการนับด้วย เพื่อให้เหมาะสมกับบอร์ดทดลอง

20. ให้ปิดโปรเจคที่สร้างมาในขั้นตอนที่ 19 ก่อนที่จะดำเนินการต่อไป

a) ให้สร้างโปรเจคชื่อ “BinaryCounter8bit” เก็บไว้ในโฟลเดอร์เดิม

b) เปิดไฟล์ใหม่สำหรับเขียนวงจรในรูปที่ 19 ใช้ชิป EP3C10E144C8 และทำการคอมไพล์วงจร



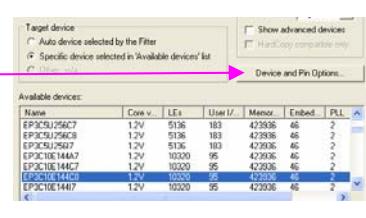
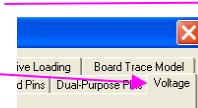
ขั้นที่ 3 นำชิ้นงาน (Designed) ที่ออกแบบมาทำ configuration ลงบอร์ดทดลอง

21. เตรียมนำชิ้นงานที่ได้จากขั้นตอนที่ 20 ลงบอร์ดทดลอง Cyclone3-Lab01 โดยทำขั้นตอนดังนี้

ก) ที่เมนู Assignment >> Device

เลือกปุ่ม Device and Pin Option

เลือกแทป Voltage



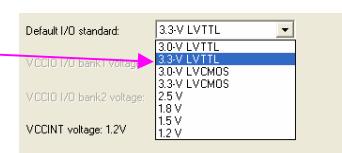
รูปที่ 20

กำหนดค่าระดับ logic level ของพอร์ท

(Default I/O standard) เป็น 3.3V-LVTTL

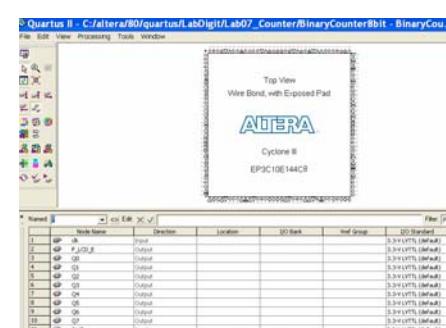
กดปุ่ม ok

(ดูรายละเอียดเพิ่มเติมได้จากเอกสารชุดที่ Doc.No.01 หน้า 24 ถึง 28)



ข) ที่เมนู Assignment >> Pins

เมื่อเลือก Pins จะปรากฏหน้าต่างรายละเอียดของขา input และ output ขนาดๆ ขึ้นมาดังรูปที่ 21



22. ใช้เม้าส์ดับเบิลคลิกที่ช่อง Location ของขาที่ต้องการ จะปรากฏข้อของชิป Cyclone III ขึ้นมาให้เลือก

ให้กำหนดขาอุปกรณ์ทั้งหมดตามรูปที่ 22 จากนั้นปิดหน้าต่าง Pins Assignment และทำการคอมไпал์วงจรซ้ำ



การทดลองที่ 7 Registers and Counters

หน้า
13 / 13

	Node Name	Direction	Location	I/O Bank	Vref Group	I/O Standard	Ref
clk	Input	PIN_22	1	B1_N0	3.3-V LVTTL (default)		
F_LCD_E	Output	PIN_54	4	B4_N0	3.3-V LVTTL (default)		
Q0	Output	PIN_38	3	B3_N0	3.3-V LVTTL (default)		
Q1	Output	PIN_39	3	B3_N0	3.3-V LVTTL (default)		
Q2	Output	PIN_42	3	B3_N0	3.3-V LVTTL (default)		
Q3	Output	PIN_43	3	B3_N0	3.3-V LVTTL (default)		
Q4	Output	PIN_44	3	B3_N0	3.3-V LVTTL (default)		
Q5	Output	PIN_46	3	B3_N0	3.3-V LVTTL (default)		
Q6	Output	PIN_49	3	B3_N0	3.3-V LVTTL (default)		
Q7	Output	PIN_50	3	B3_N0	3.3-V LVTTL (default)		
SW7	Input	PIN_33	2	B2_N0	3.3-V LVTTL (default)		

รูปที่ 22

หมายเหตุ การต่อพอร์ทอุปกรณ์ สามารถดูรายละเอียดของการต่อขาพอร์ทด้านๆได้จากคู่มือบอร์ดชื่อเอกสารคือ **4.WARRIOR CYCLONE3 Education Board User's Manual.pdf**

วงจรหลอด LED ในรูปที่ 12 (หน้า16), ตารางที่ 3 (หน้า17) วงจรสวิทช์เลื่อน F_SL_SW7 ในรูปที่ 16 (หน้า19). ตารางที่ 7 (หน้า20) วงจรสัญญาณนาฬิกา clk, clock ในหน้าที่ 31 ข้อที่ 9

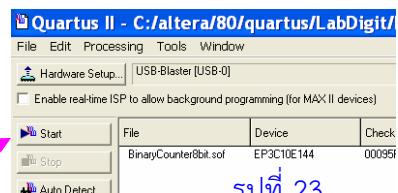
23. นำบอร์ดทดลอง Cyclone3-Lab01 มาต่อแหล่งจ่ายไฟ 12 โวลท์ เปิดสวิตช์ จากนั้นต่อสายสำหรับโหลดโปรแกรมจากคอมพิวเตอร์ลงในบอร์ด

ใช้สาย Byte Blaster สำหรับเครื่องคอมพิวเตอร์ PC

ใช้สาย USB Blaster สำหรับเครื่องคอมพิวเตอร์โน๊ตบุค

24. ดาวน์โหลดลงบอร์ดโดยไปที่เมนู **TOOLS >> Programmer**

จะปรากฏหน้าต่างดังรูปที่ 23 เมื่อกดปุ่ม **Start** ข้อมูลของวงจร ที่ออกแบบไว้จะถูกโหลดลงบอร์ดทันที



รูปที่ 23

ผลการทดลอง หลอด LED ทั้ง 8 ดวงที่ແນກມໍາດີລະບົບຂອງ Q₇...Q₀ ບັນບຼອດທົດລອງ (LED ຕັບ ແສດສຄານະເປັນລອຈິກ '0')

- ก) ปรับสวิตช์เลื่อน SW7 ไปที่ตำแหน่ง high สังเกตการณ์ติด/ดับของ LED (เสมีอนเป็นเข้าฐานสอง)
ค่าเริ่มนับจากค่าต่ำสุดคือ $Q_7 \dots Q_0 = \underline{\hspace{2cm}} 0000 \quad 0001$ (เขียนเป็นเลขในนารี)

ข) รอเวลาประมาณสิบวินาทีจึงปรับสวิตช์เลื่อน SW7 ไปที่ตำแหน่ง low ค่า $Q_7 \dots Q_0 = \underline{\hspace{2cm}} X$
เกิดอะไรขึ้นกับ LED ไฟ灭了

ปรับสวิตช์กลับไปที่ตำแหน่ง high อีกครั้ง เกิดอะไรขึ้นกับ LED X

การทำงาน low หรือ high ของ SW7 ส่งผลอย่างไรต่อการนับ X

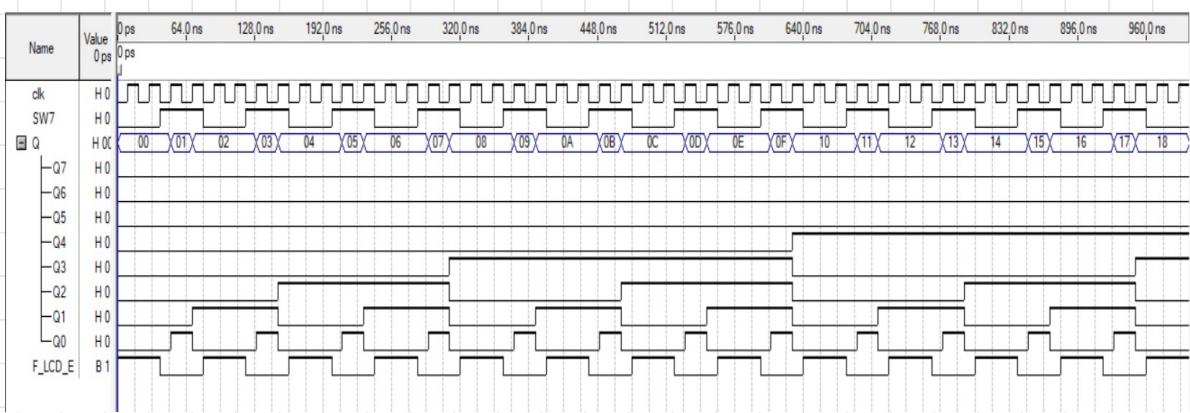
ค) ถ้าจะให้การปรับสวิตช์เลื่อน SW7 โดยที่ค่าของ $Q_7 \dots Q_0$ ยังคงค้างไว้เช่นเดิมแต่ LED ไม่ดับ จะต้องดัดแปลงวงจรอย่างไร (ทำให้ดู) X

ง) ถ้าต้องการให้วงจรนับนี้ เพิ่มขาควบคุมที่สามารถเคลียร์ค่าให้ $Q_7 \dots Q_0$ เป็นศูนย์ได้ ต้องดัดแปลงวงจรอย่างไร
บ้าง (ทำให้ดู) เพิ่งนำ VCC ที่อยู่ในวงจร 1 ไปเข้าสัญญาณ Input 'clear'
ซึ่งจะ clear ทันทีที่ทาง input มาจากตัวกรองที่ต้องห้ามกันไฟฟ้าชาร์จ! ที่ไม่ gate ทาง Input 'clear'

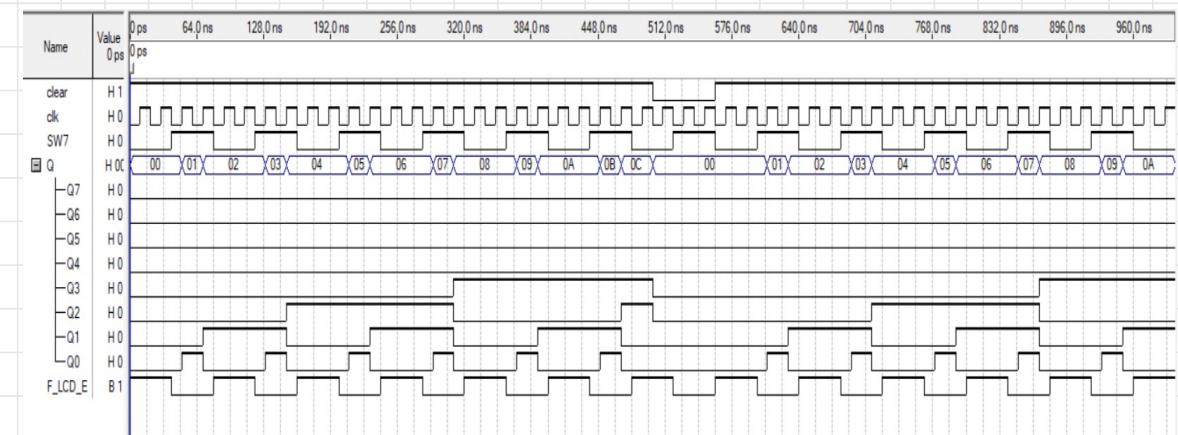
งานนักตรวจสอบภายในที่มีอำนาจหน้าที่ตรวจสอบ (ให้เชิงลวงเบรดดาน A4 ที่มีสิทธิ์ตรวจสอบและรายงานให้หัวหน้าอธิการบดี)

1. ให้อักษรแบบวงจรนับเลขฐานสิบบนภาค 4 หลัก (Digit) ที่เริ่มนับจาก 0 ไปจนถึงเลข ที่ตรงกับเลขท้ายสี่ตัวของรหัสนักศึกษา แล้วให้หยุดนับค้างไว้ เช่นรหัส 560101163101-1 จะเริ่มนับจาก 0 ไปหยุดค้างที่ 1011 โดย น.ศ.ต้องส่งให้ตรวจสอบเข้าเรียนครั้งต่อไป (การทดลองครั้งต่อไปจะเริ่มจากงานที่แต่ละคนออกแนวโน้มเท่านั้น หากไม่มีสิ่งจะอภิวัติได้เข้าเรียนในคราวหน้าด้วย)

Vector Waveforms (များများပေါ်)



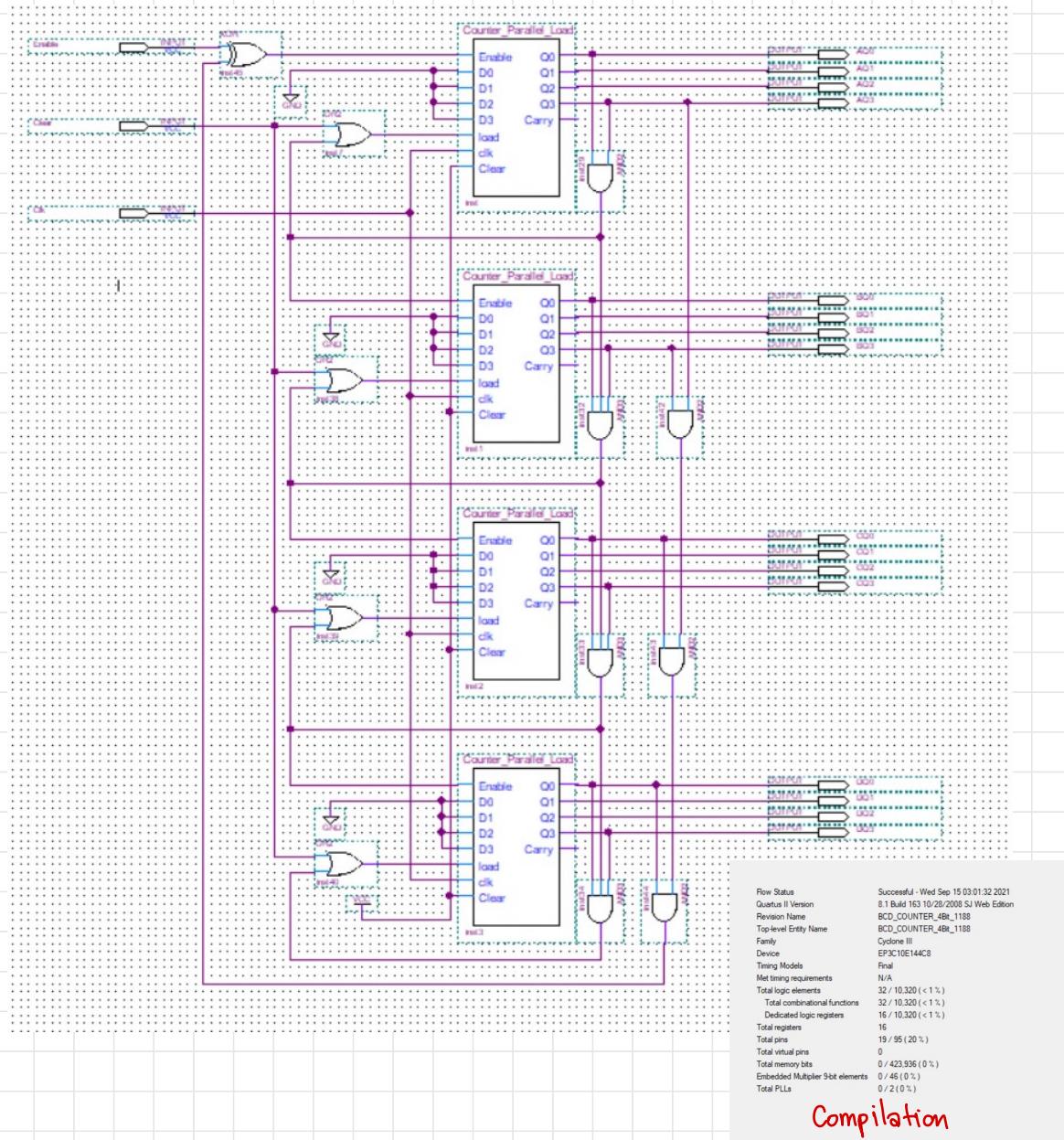
Vector Waveforms (မြန်မြည်)



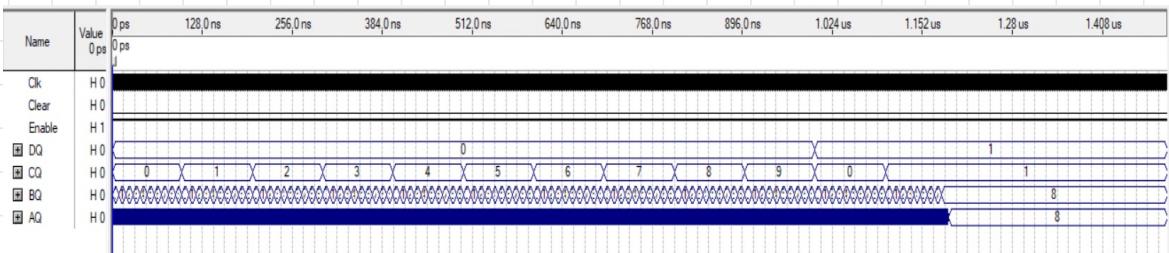
งานมอบหมายท้ายการทดลอง (ให้เขียนลงบนกระดาษ A4 ที่มีเส้นบรรทัดและรวมไว้ท้ายเอกสารการทดลอง)

- ให้อ่านแบบวงจรรับส่งฐานสิบ进位加法器 4 หลัก (Digit) ที่เริ่มนับจาก 0 ไปจนถึงเลข ที่ปรับเปลี่ยนได้ตัวของรหัสนักศึกษา และให้หัดนับด้วย เทคนิค 560101163101-1 จะเริ่มนับจาก 0 ไปหยุดที่ 1011 โดย น.ศ.ต้องสังเกตตรวจก่อนขั้นเรียน ครั้งต่อไป (การทดลองครั้งต่อไปจะเริ่มจากการที่แต่ละคนออกแบบมาเท่านั้น หากไม่มีสิ่งจะถือว่าไม่ได้เข้าเรียนในคืนนั้นด้วย)

๖๖ วงจรเลขสี่หลัก 4 Digit (ตัวเลข ๐ → ๑๙๘)



Simulation Waveforms



(frequen clk ဆဲ period 1 ns , clear ဆဲ '0' , Enable ဆဲ '1')