การทดลองที่ 1
Logic Gate พื้นฐาน

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของลอจิกเกทพื้นฐาน (AND, OR, NOT, NAND, NOR และ XNOR) ได้
3. สามารถดาวน์โหลดวงจรลงไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

อุปกรณ์สำหรับการทดลอง

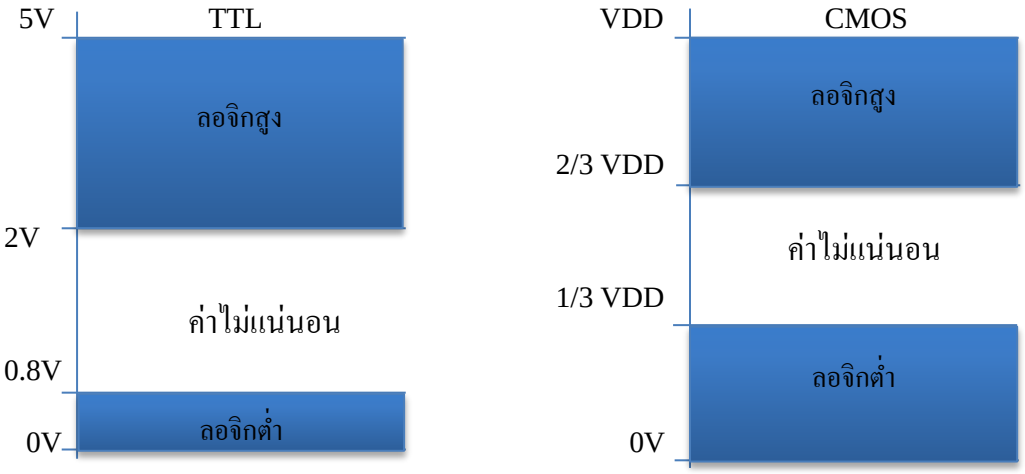
1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

ทฤษฎี

1. คำจำกัดความของระดับลอจิกในวงจรดิจิตอล

ในการศึกษาวงจรดิจิตอลนั้นจะต้องทำความเข้าใจกับเรื่องของระดับสัญญาณที่ใช้ในวงจรก่อน โดยระดับสัญญาณที่สำคัญในวงจรดิจิตอลได้แก่ ระดับลอจิกสูง (HIGH หรือบางครั้งแท้ด้วย 1) และลอจิกต่ำ (LOW หรือบางครั้งแทนด้วย 0) ซึ่งระดับลอจิกเหล่านี้ใช้แทนช่วงของระดับแรงดันไฟฟ้านั่นเอง

ระบบดิจิตอลมีการแบ่งอุปกรณ์ออกเป็น 2 ตระกูล ได้แก่ ทีทีแอล (TTL) และซีมอส (CMOS) ซึ่งช่วงระดับแรงดันที่ใช้แทนค่าลอจิกในวงจรดิจิตอลจะต่างกัน สำหรับไอซีทีทีแอลจะใช้แหล่งจ่ายแรงดันไม่เกิน 5 โวลต์ +/- 5% โดยระดับแรงดันที่มีค่าต่ำกว่า 0.8 โวลต์จะใช้แทนสถานะลอจิกต่ำ และระดับแรงดันที่สูงกว่า 2 โวลต์ขึ้นไปจะใช้แทนระดับลอจิกสูง ส่วนไอซีซีมอสจะเป็นไอซีที่มีช่วงการใช้แรงดันไฟเลี้ยงที่ยืดหยุ่นได้มากกว่าโดยสามารถใช้ได้ตั้งแต่ 3 โวลต์ ถึง 18 โวลต์ ซึ่งการกำหนดช่วงของระดับลอจิกจะสัมพันธ์กับระดับไฟเลี้ยงที่จ่ายให้แก่ไอซี โดยที่ระดับแรงดันตั้งแต่ 0 จนถึงประมาณ1/3 ของแรงดันไฟเลี้ยง จะใช้แทนระดับลอจิกต่ำ และ แรงดันตั้งแต่ช่วงประมาณ 2/3 ของแรงดันไฟเลี้ยงจนถึงระดับไฟเลี้ยงจะใช้แทนระดับลอจิกสูง แสดงดังรูปที่ 1.1



รูปที่ 1.1 ช่วงแรงดันไฟฟ้าและระดับลอจิก

2. แอนด์เกท (AND Gate)

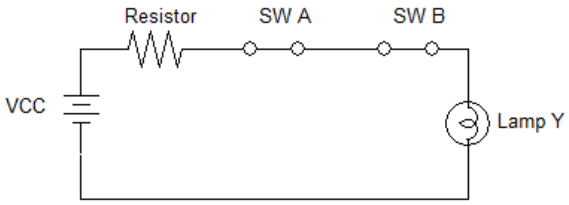
แอนด์เกทเป็นวงจรลอจิกเกทพื้นฐานที่ให้สัญญาณเอาต์พุทเป็น HIGH เมื่อสัญญาณอินพุททั้งหมดอยู่ในสภาวะ HIGH และให้เอาต์พุทเป็น LOW เมื่อสัญญาณขาใดขาหนึ่งเป็น LOW ซึ่งสามารถเขียนเป็นสมการบูลีนได้ดังนี้

Y = A_AND_B หรือบางครั้งสามารถเขียนได้ดังนี้ Y = A • B

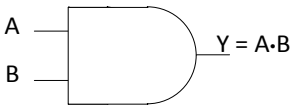
สมการดังกล่าวอ่านว่า Y เท่ากับ A แอนด์ B สำหรับสัญลักษณ์จุดในสมการเราอาจเขียนหรือไม่เขียนก็ได้ซึ่งสมการดังกล่าวจะเป็น

Y = A B

การทำงานของแอนด์เกทสามารถเปรียบเทียบเป็นวงจรเสมือนที่ประกอบขึ้นจากสวิตช์ที่ต่ออนุกรมกันดังรูปที่ 2 นั่นคือหลอดไฟจะติดได้ในกรณีที่สวิทช์ A และ B ปิด (ON) ทั้งคู่ และหลอดไฟจะดับลงในกรณีที่สวิตช์ตัวใดตัวหนึ่งเปิด (OFF) ซึ่งสามารถเขียนเป็นตารางความจริง (Truth Table) ได้ดังตารางที่ 1.1 ในทางปฏิบัติแอนด์เกทที่มีใช้งานโดยทั่วไปจะอยู่ในรูปของไอซี เช่น ไอซีเบอร์ 7408



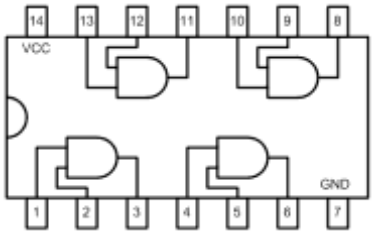
รูปที่ 1.2 วงจรเสมือนแอนด์เกท



รูปที่ 1.3 สัญลักษณ์ของแอนด์เกท

อินพุท		เอาต์พุท
A	**B**	**Y**
0	0	0
0	1	0
1	0	0
1	1	1

ตารางที่ 1.1 ตารางความจริงของแอนด์เกท



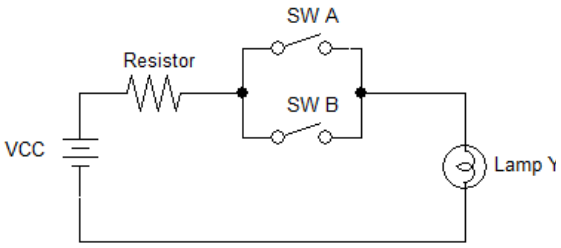
รูปที่ 1.4 โครงสร้างของไอซีเบอร์ 7408

3. ออร์เกท (OR Gate)

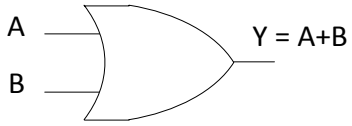
ออร์เกทเป็นวงจรลอจิกพื้นฐานที่ให้สัญญาณเอาต์พุทเป็น HIGH เมื่อสัญญาณอินพุทขาใดขาหนึ่งอยู่ในสภาวะ HIGH และให้สัญญาณเอาต์พุทเป็น LOW เมื่อสัญญาณอินพุททุกขาเป็น LOW ซึ่งสามารถเขียนเป็นสมการบูลีนได้ดังนี้

Y = A OR B หรือบางครั้งสามารถเขียนได้ดังนี้ Y = A + B

สมการดังกล่าวอ่านว่า Y เท่ากับ A แอนด์ B การทำงานของออร์เกทสามารถเปรียบเทียบเป็นวงจรเสมือนที่ประกอบขึ้นจากสวิตช์ที่ต่อขนานกันดังรูปที่ 1.5 นั่นคือหลอดไฟจะติดได้ในกรณีที่สวิทช์ A และ B ตัวใดตัวหนึ่งปิด (ON) และหลอดไฟจะดับลงในกรณีที่สวิตช์ทั้งสองตัวเปิด (OFF) ซึ่งสามารถเขียนเป็นตารางความจริง (Truth Table) ได้ดังตารางที่ 1.2 ในทางปฏิบัติออร์เกทที่มีใช้งานโดยทั่วไปจะอยู่ในรูปของไอซี เช่น ไอซีเบอร์ 7432



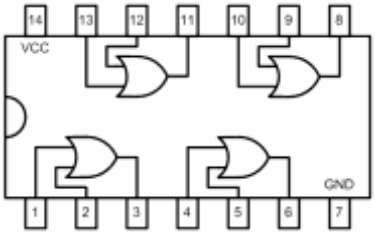
รูปที่ 1.5 วงจรสเหมือนออร์เกท



รูปที่ 1.6 สัญลักษณ์ของออร์เกท

อินพุท		เอาต์พุท
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

ตารางที่ 1.2 ตารางความจริงของออร์เกท



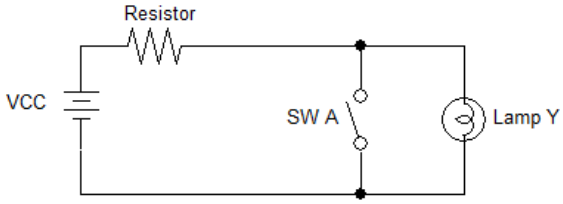
รูปที่ 1.7 โครงสร้างของไอซีเบอร์ 7432

4. นอตเกท (NOT Gate)

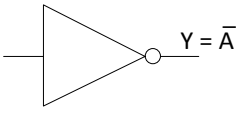
นอตเกทเป็นวงจรลอจิกพื้นฐานที่ให้สัญญาณเอาต์พุทตรงกันข้ามกับสัญญาณอินพุท ซึ่งสามารถเขียนเป็นสมการบูลีนได้ดังนี้

$Y = \overline{A}$ หรือบางครั้งสามารถเขียนได้ดังนี้ Y =A’

การทำงานของออร์เกทสามารถเปรียบเทียบเป็นวงจรเสมือนที่ประกอบขึ้นจากสวิตช์ที่ต่อขนานกันดังรูปที่ 1.8 นั่นคือหลอดไฟจะติดได้ในกรณีที่สวิทช์ A เปิด (OFF) และหลอดไฟจะดับลงในกรณีที่สวิตช์ A เปิด (ON) ซึ่งสามารถเขียนเป็นตารางความจริง (Truth Table) ได้ดังตารางที่ 1.3 ในทางปฏิบัตินอตเกทที่มีใช้งานโดยทั่วไปจะอยู่ในรูปของไอซี เช่น ไอซีเบอร์ 7404



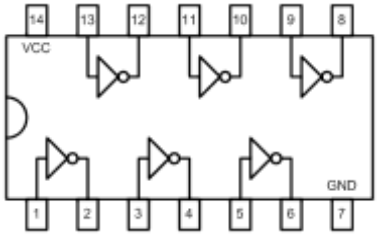
รูปที่ 1.8 วงจรสเหมือนนอตเกท



รูปที่ 1.9 สัญลักษณ์ของนอตเกท

อินพุท	เอาท์พุท
A	Y
0	1
1	0

ตารางที่ 1.3 ตารางความจริงของนอตเกท



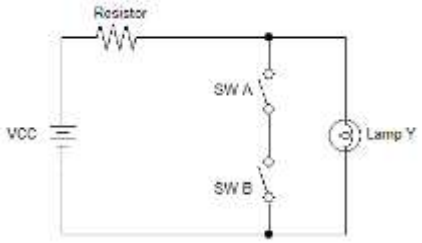
รูปที่ 1.10 โครงสร้างของไอซีเบอร์ 7404

5. แนนด์เกท (NAND Gate)

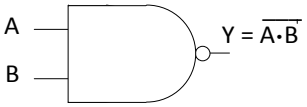
แนนด์เกทเป็นเกทพื้นฐานที่เกิดจากการนำแอนด์และนอตเกทมาผสมกัน โดยการนำนอตเกทต่อเข้าทางเอาต์พุทของแอนด์เกท ซึ่งสามารถเขียนเป็นสมการบูลีนได้ดังนี้

$$Y = \overline{A \bullet B}$$

การทำงานของออร์เกทสามารถเปรียบเทียบเป็นวงจรเสมือนที่ประกอบขึ้นจากสวิตช์ที่ต่อขอนุกรมกันดังรูปที่ 1.11 นั่นคือหลอดไฟจะติดได้ในกรณีที่สวิทช์ตัวใดตัวหนึ่ง หรือทั้งคู่เปิด (OFF) และหลอดไฟจะดับลงในกรณีที่สวิตช์ทั้งสองตัวเปิด (ON) ซึ่งสามารถเขียนเป็นตารางความจริง (Truth Table) ได้ดังตารางที่ 1.4 ในทางปฏิบัตินอตเกทที่มีใช้งานโดยทั่วไปจะอยู่ในรูปของไอซี เช่น ไอซีเบอร์ 7400



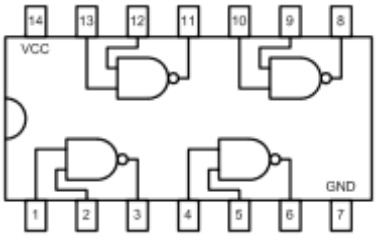
รูปที่ 1.11 วงจรเสมือนแนนด์เกท



รูปที่ 1.12 สัญลักษณ์ของแนนด์เกท

อินพุท		เอาต์พุท
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

ตารางที่ 1.4 ตารางความจริงของแนนด์เกท



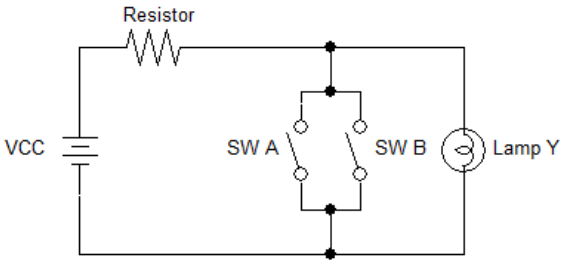
รูปที่ 1.13 โครงสร้างของไอซีเบอร์ 7400

6. นอร์เกท (NOR Gate)

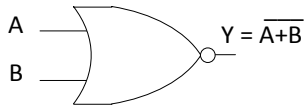
นอร์เกทเป็นเกทพื้นฐานที่เกิดจากการนำออร์เกทและนอตเกทมาผสมกัน โดยการนำนอตเกทต่อเข้าทางเอาต์พุทของออร์เกท ซึ่งสามารถเขียนเป็นสมการบูลีนได้ดังนี้

$$Y = \overline{A + B}$$

การทำงานของนอร์เกทสามารถเปรียบเทียบเป็นวงจรเป็นวงจรเสมือนที่ประกอบขึ้นจากสวิตช์ที่ต่อขนุกรมกันดังรูปที่ 1.14 นั่นคือหลอดไฟจะติดได้ในกรณีที่สวิทช์ทุกตัวเปิด (OFF) และหลอดไฟจะดับลงในกรณีที่สวิตช์ตัวใดตัวหนึ่งปิด (ON) ซึ่งสามารถเขียนเป็นตารางความจริง (Truth Table) ได้ดังตารางที่ 1.5 ในทางปฏิบัตินอตเกทที่มีใช้งานโดยทั่วไปจะอยู่ในรูปของไอซี เช่น ไอซีเบอร์ 7402



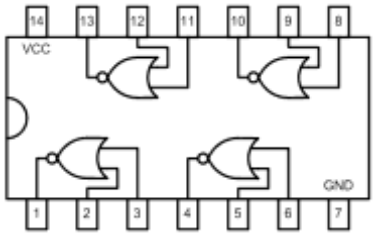
รูปที่ 1.14 วงจรเสมือนนอร์เกท



รูปที่ 1.15 สัญลักษณ์ของนอร์เกท

อินพุท		เอาต์พุท
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

ตารางที่ 1.5 ตารางความจริงของนอร์เกท



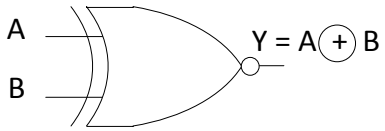
รูปที่ 1.16 โครงสร้างของไอซีเบอร์ 7402

7. เอ็กซ์คลูซีฟออร์เกท (Exclusive – OR Gate)

เอ็กซ์คลูซีฟออร์เกทเป็นเกทที่ให้เอาต์พุทเป็น 1 เมื่ออินพุทแตกต่างกันและจะให้เอาต์พุทเป็น 0 ในกรณีที่อินพุทมีค่าเหมือนกัน ซึ่งสามารถเขียนสมการบูลีนได้ดังนี้

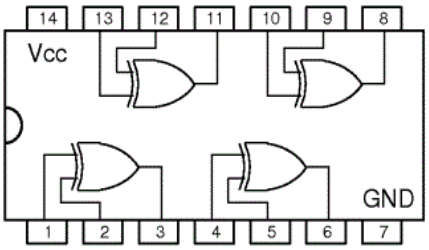
$$Y = A \oplus B$$

สำหรับลักษณ์และและตารางความจริงแสดงดังรูปที่ 1.17 และตารางที่ 1.6 ตามลำดับ ซึ่งไอซีทำหน้าที่เป็นเอ็กคลูซีฟออร์เกทที่มีอยู่ตามท้องตลาดได้แก่เบอร์ 7486



อินพุท		เอาต์พุท
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

รูปที่ 1.17 สัญลักษณ์ของเอ็กซ์คลูซีฟออร์เกท ตารางที่ 1.6 ตารางความจริงของเอ็กซ์คลูซีฟออร์เกท



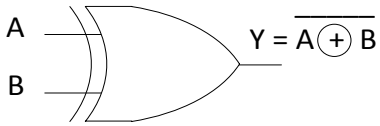
รูปที่ 1.18 ไอซีเอ็กคลูซีฟออร์เกทเบอร์ 7486

8. เอ็กซ์คลูซีฟนอร์เกท (Exclusive – NOR Gate)

เอ็กซ์คลูซีฟนอร์เกทเป็นเกทที่เกิดจากการนำนอตเกทไปต่อทางเอาต์พุทของเอ็กซ์คลูซีฟออร์เกทเพื่อกลับผลเอาต์พุทให้มีค่าตรงกันข้ามกับ เอ็กซ์คลูซีฟนอร์เกท ดังนั้นเอาต์พุทของมันจะมีค่าเป็น 0 เมื่ออินพุทแตกต่างกัน และจะให้เอาต์พุทเป็น 1 ในกรณีที่อินพุทมีค่าเหมือนกัน ซึ่งสามารถเขียนสมการบูลีนได้ดังนี้

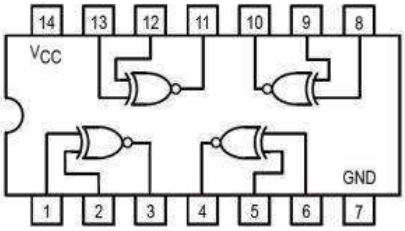
$$Y = \overline{A \oplus B}$$

สำหรับลักษณ์และและตารางความจริงแสดงดังรูปที่ 1.19 และตารางที่ 1.7 ตามลำดับ ซึ่งไอซีทำหน้าที่เป็นเอ็กคลูซีฟนอร์เกทที่มีอยู่ตามท้องตลาดได้แก่เบอร์ 74266



อินพุท		เอาต์พุท
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

รูปที่ 1.19 สัญลักษณ์ของเอ็กซ์คลูซีฟนอร์เกท ตารางที่ 1.7 ตารางความจริงของเอ็กซ์คลูซีฟนอร์เกท

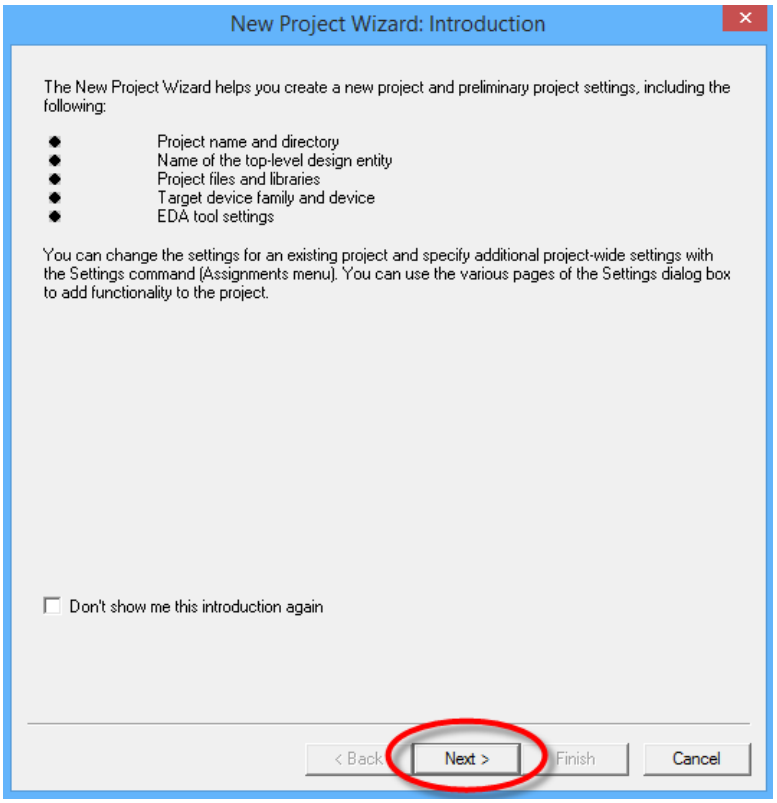


รูปที่ 1.20 ไอซีเอ็กคลูซีฟนอร์เกทเบอร์ 7486

ขั้นตอนการทดลอง ตอนที่ 1 แอนด์เกท

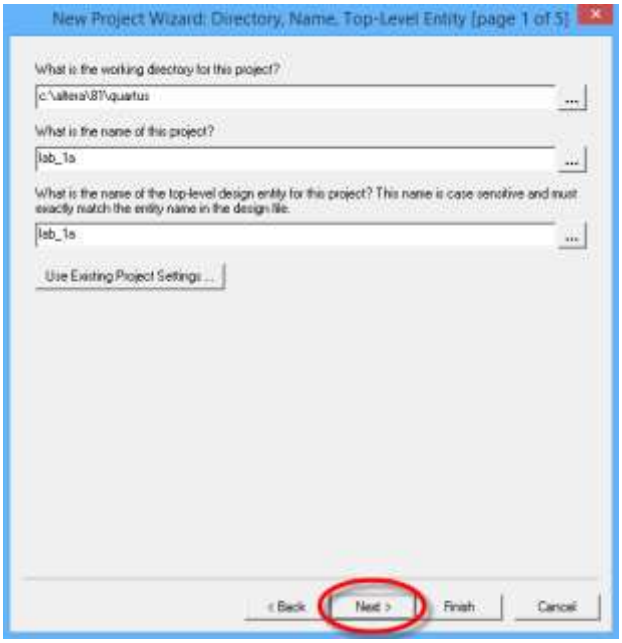
1. เปิดโปรแกรม Quartus II 8.1 Web Edition
2. สร้างโปรเจคโดยใช้เมนู File -> New Project Wizard ดังรูปที่ 1.21
3. จากนั้นหน้าต่าง New Project Wizard : Introduction จะปรากฏขึ้น ให้คลิกที่ Next ดังรูปที่ 1.22

รูปที่ 1.21 การเรียกเมนูสร้างโปรเจคใหม่



รูปที่ 1.22 การสร้างโปรเจคใหม่

4. จากนั้นหน้าต่าง New Project Wizard : Directory Name Top-Level Entity จะปรากฏขึ้น ให้ใส่ชื่อโปรเจคในชื่อ lab_1a ลงในช่อง What is the name of this project เสร็จแล้วคลิกที่ปุ่ม Next แสดงดังรูปที่ 1.23

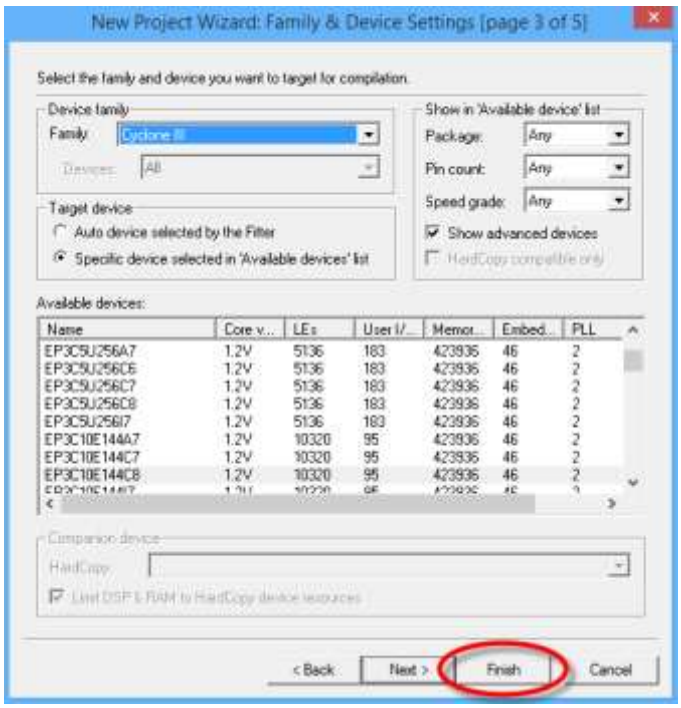


รูปที่ 1.23 การสร้างโปรเจคใหม่

5. จากนั้นหน้าต่าง New Project Wizard : Add Files จะปรากฏขึ้น คลิก Next ดังรูปที่ 24

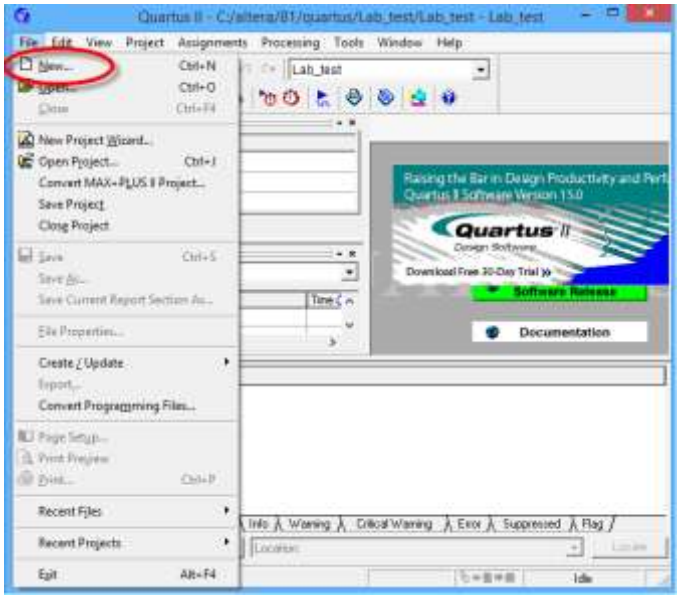
รูปที่ 1.24 การสร้างโปรเจคใหม่

6. จากนั้นหน้าต่าง New Project Wizard : Family & Device Settings จะปรากฏขึ้น ในช่อง Device family ให้เลือกรุ่นของไอซีเป็น Cyclone III และในช่อง Available devices ให้เลือกเบอร์ของไอซีเป็น EP3C10E144C8 เสร็จแล้วคลิกเลือกที่ปุ่ม Finish ดังรูปที่ 1.25



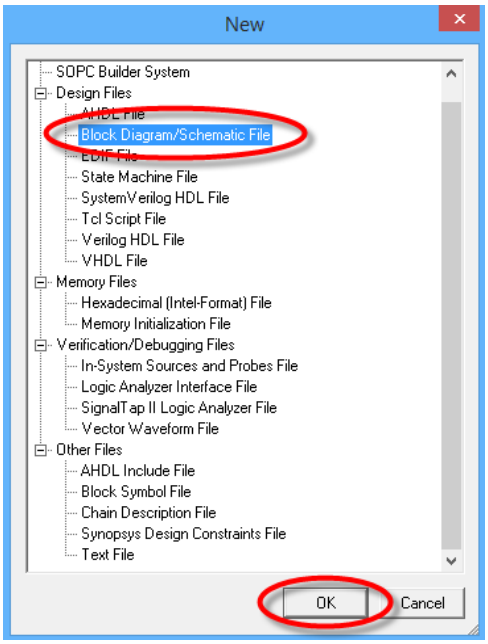
รูปที่ 1.25 การสร้างโปรเจคใหม่

7. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New ดังรูปที่ 1.26



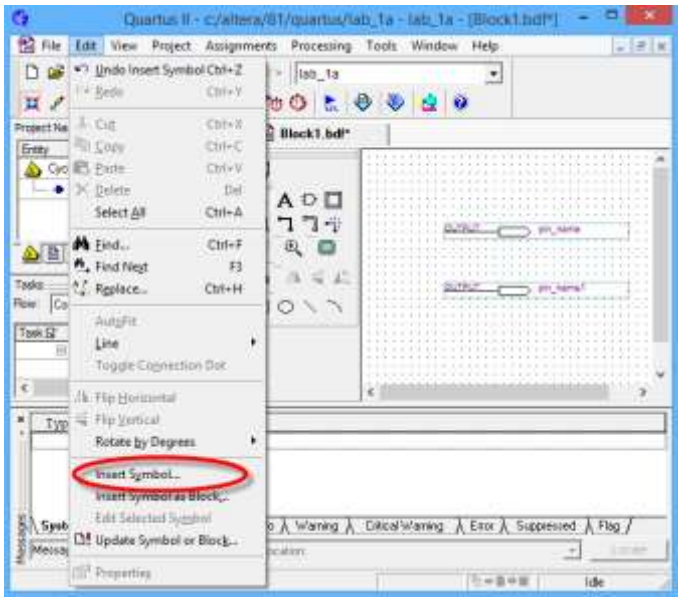
รูปที่ 1.26 การสร้างไฟล์ใหม่

8. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK ดังรูปที่ 1.27



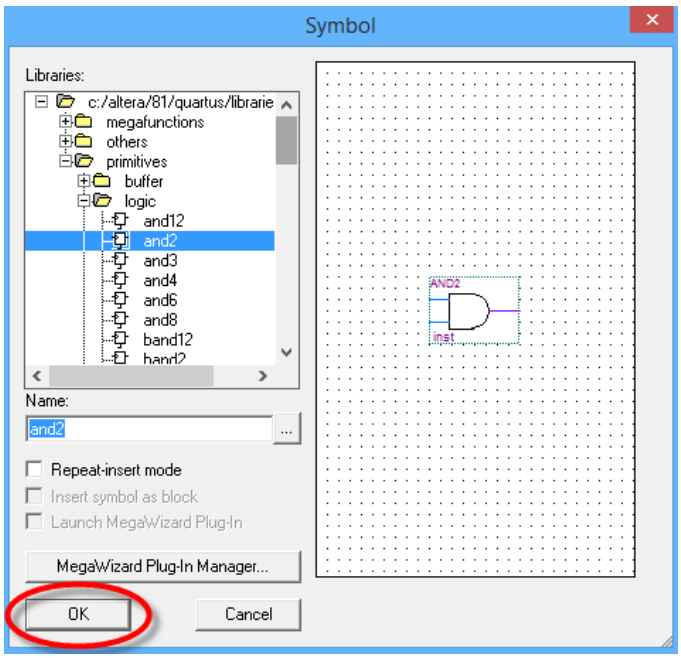
รูปที่ 1.27 เลือกสร้างไฟล์กราฟิกสำหรับวาดวงจร

9. ทำการวาดอุปกรณ์ลงบน Graphic Editor โดยเลือกที่ Editor -> Insert Symbol เพื่อเรียกหน้าต่าง Symbol ขึ้นมาดังรูปที่ 1.28



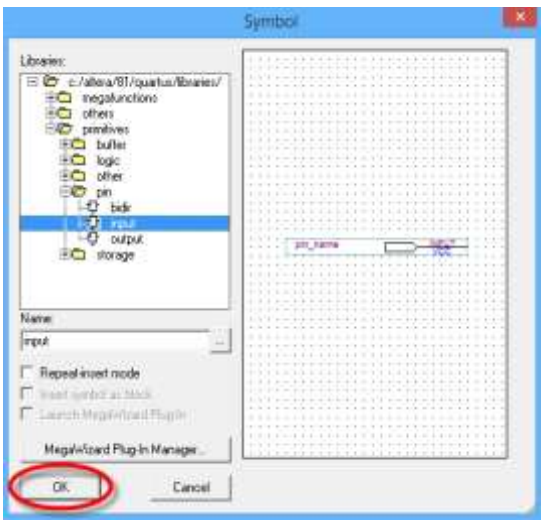
รูปที่ 1.28 เรียกหน้าต่าง Symbol

10. จากนั้นจะมีหน้าต่าง Symbol ปรากฏขึ้นมา ให้คลิกเลือกที่ C:/altera/81/quartus/libraries -> primitives -> logic -> and2 หรือให้ค้นหาโดยพิมพ์คำว่า and2 ในช่อง Name เสร็จแล้วคลิกที่ OK ดังรูปที่ 1.29

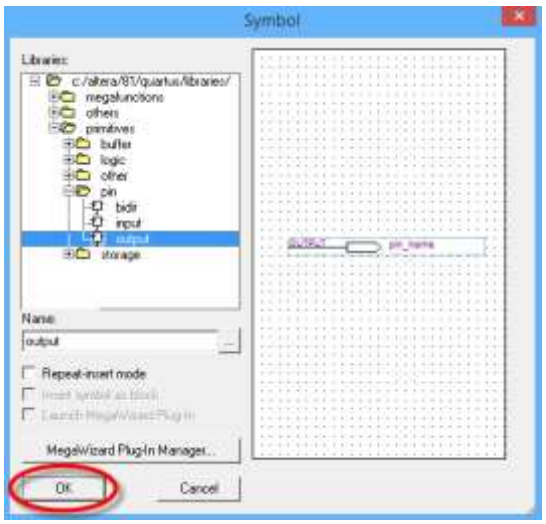


รูปที่ 1.29 การสร้างแอนด์เกท 2 อินพุท

11. เรียกหน้าต่าง Symbol ขึ้นมาอีกครั้งเพื่อสร้างพอร์ต input และ output ดังรูปที่ 1.30 และ ดังรูปที่ 1.31 ตามลำดับ เสร็จแล้วคลิกที่ OK

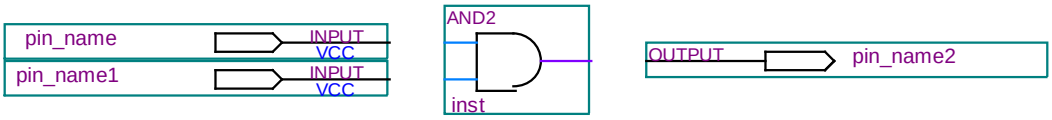


รูปที่ 1.30 การสร้างอินพุท



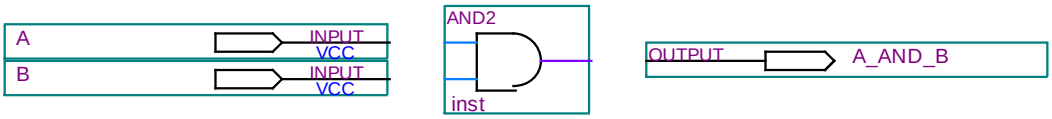
รูปที่ 31 การสร้างเอาต์พุท

12. จากนั้นทำการวางอุปกรณ์ ดังรูปที่ 1.32



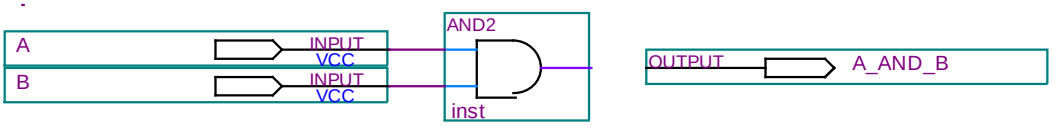
รูปที่ 1.32 การจัดวางอุปกรณ์

13. ทำการเปลี่ยนชื่อพอร์ตอินพุตและเอาท์พุตจาก PIN_NAME ให้เป็นดังรูปที่ 1.33 โดยดับเบิลคลิกที่ตัวหนังสือ PIN_NAME โปรแกรมจะขึ้นแถบสีดำตรงชื่อพอร์ต ซึ่งขณะนี้เราสามารถพิมพ์ชื่อพอร์ตทับลงไปได้ทันที



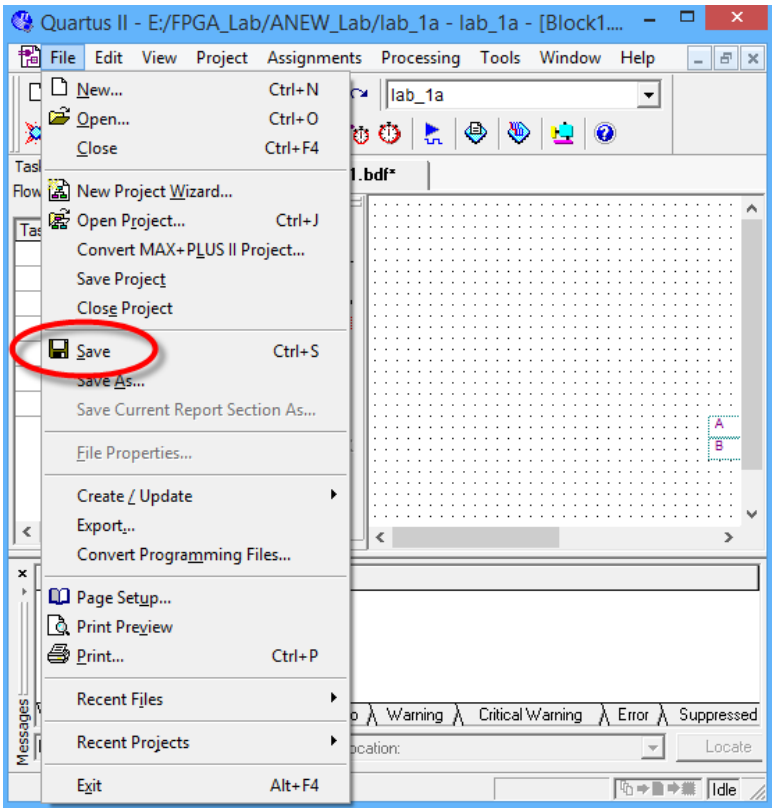
รูปที่ 1.33 การเปลี่ยนชื่อพอร์ตอินพุต และเอาท์พุต

14. เสร็จแล้วทำการลากเส้นเชื่อมต่อขาของอุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูปที่ 1.34 โดยเลื่อนเมาส์มาที่บริเวณขั้วของอุปกรณ์ ขณะนี้จะสังเกตได้ว่าเคอร์เซอร์เปลี่ยนจากลูกศรกลายเป็นเส้นครอสแฮร์ (Cross Hair : เครื่องหมายบวกขนาดใหญ่) ให้คลิกแล้วลากเพื่อเชื่อมต่อกับขั้วอุปกรณ์ โดยในการลากเส้น ให้ลากให้พอดีกับขั้วอุปกรณ์ห้ามลากเกินเข้าไปในตัวอุปกรณ์ ซึ่งอาจทำให้โปรแกรมตีความได้ว่าจุดนั้นไม่มีการเชื่อมต่อ



รูปที่ 1.34 การลากเส้นเชื่อมต่ออุปกรณ์เข้าด้วยกัน

15. ขณะนี้วงจรได้ถูกวาดเสร็จเรียบร้อยแล้ว ให้ทำการบันทึกโดยใช้เมนู File -> Save



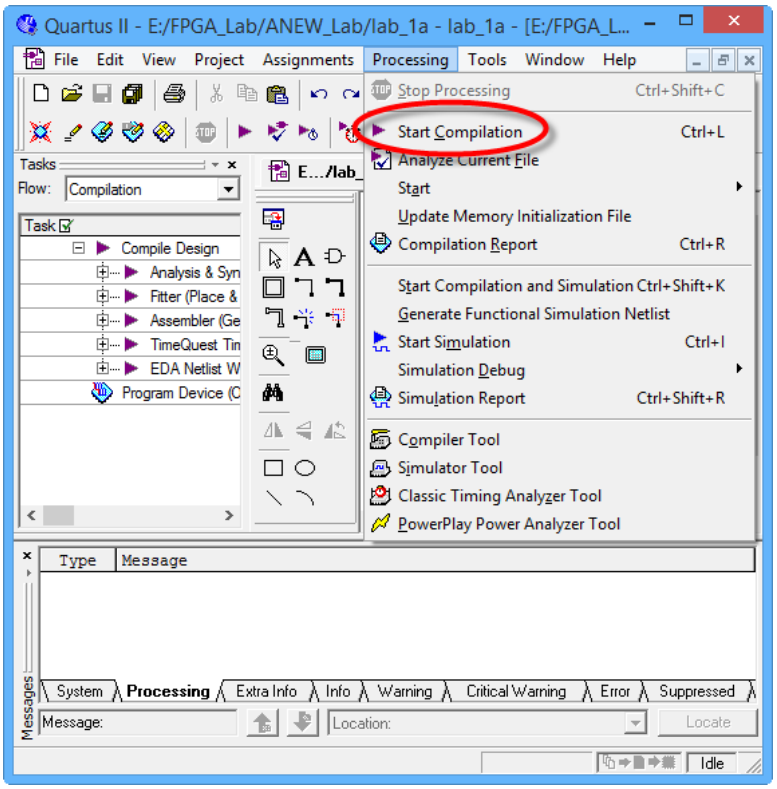
รูปที่ 1.35 การบันทึกไฟล์

16. จากนั้นหน้าต่าง Save As จะปรากฏขึ้น ให้คลิกที่ Save ดังรูปที่ 1.36

รูปที่ 1.36 การบันทึกไฟล์

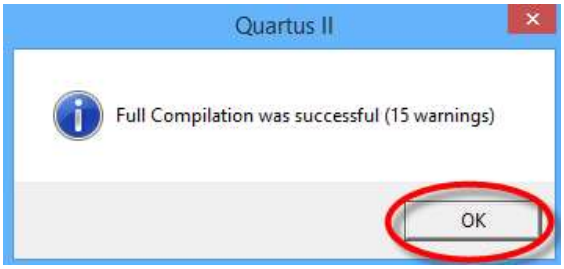
การคอมไพล์วงจร และการจำลองการทำงาน

17. ต่อไปเราจะทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ เริ่มจากคลิกเมนู Processing -> Start Compilation ดังรูปที่ 1.37



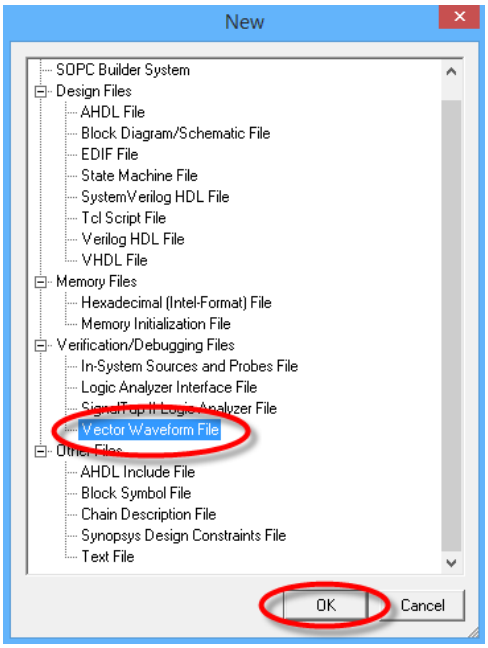
รูปที่ 1.37 เมนูเรียกหน้าต่างคอมไพล์เลอร์

18. เมื่อโปรแกรมทำการคอมไพล์เสร็จ จากนั้นหน้าต่าง Quartus II 8.1 Web Edition จะปรากฏขึ้น ให้คลิกที่ OK



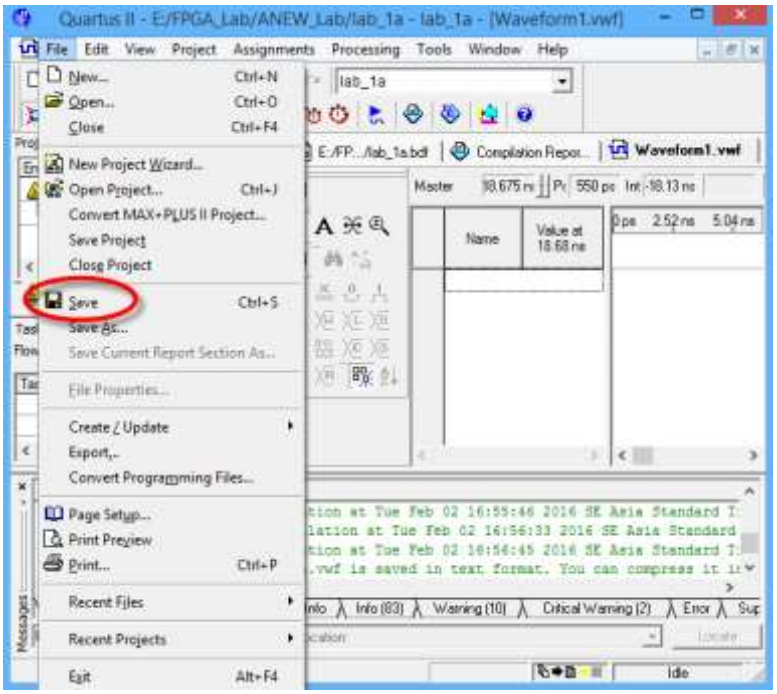
รูปที่ 1.38 หน้าต่างแสดงการคอมไพล์เลอร์เสร็จแล้ว

19. ต่อไปจะทำการกำหนดรูปคลื่นสัญญาณให้กับวงจรเพื่อจำลองการทำงาน โดยการเรียกที่เมนู File -> New จะปรากฏหน้าต่าง New ขึ้นมา ให้เลือกที่ Vector Waveform File จากนั้นคลิกที่ OK ดังรูปที่ 1.39



รูปที่ 1.39 สร้างไฟล์ Vector Waveform สำหรับบันทึกผลการจำลองการทำงาน

20. จากนั้นหน้าต่าง Vector Waveform File จะปรากฏขึ้นมา ให้บันทึกไฟล์ในชื่อ lab_1a.scf โดยใช้เมนู File -> Save ดังรูปที่ 1.40

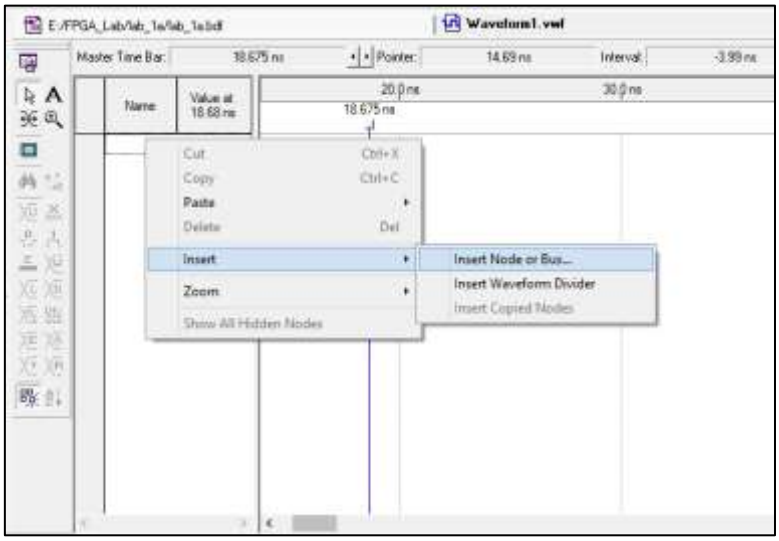


รูปที่ 1.40 การบันทึกไฟล์

21. จากนั้นหน้าต่าง Save As จะปรากฏขึ้น ให้คลิกที่ Save ดังรูปที่ 1.41

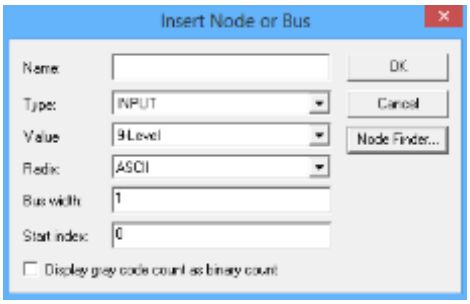
รูปที่ 1.41 การบันทึกไฟล์

22. ทำการเพิ่ม Node เพื่อวิเคราะห์สัญญาณโดยคลิกขวาบน Vector Waveform Editor จากนั้นจะมีเมนู Popup ขึ้นมา ให้ เลือกที่ Insert -> Insert Node Bus…



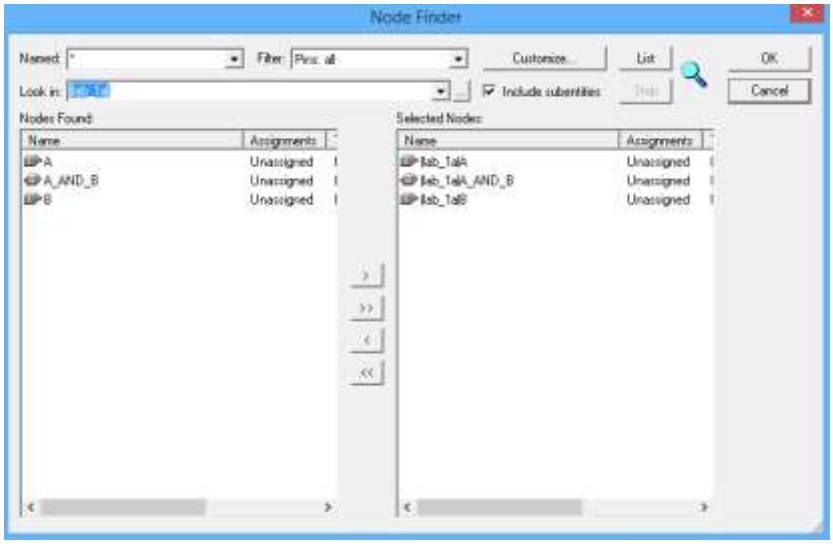
รูปที่ 1.42 การเพิ่ม Node เพื่อวิเคราะห์สัญญาณ

23. คลิกที่ Node Finder… เพื่อค้นหาโหนด



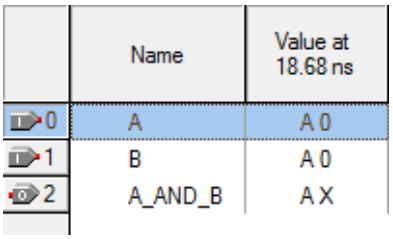
รูปที่ 1.43 การเพิ่ม Node เพื่อวิเคราะห์สัญญาณ

24. คลิกที่ปุ่ม List เพื่อแสดงรายการโหนดที่สามารถลำลองการทำงานได้ แล้วเลือกที่ปุ่ม ›› เสร็จแล้วคลิกที่ปุ่ม OK



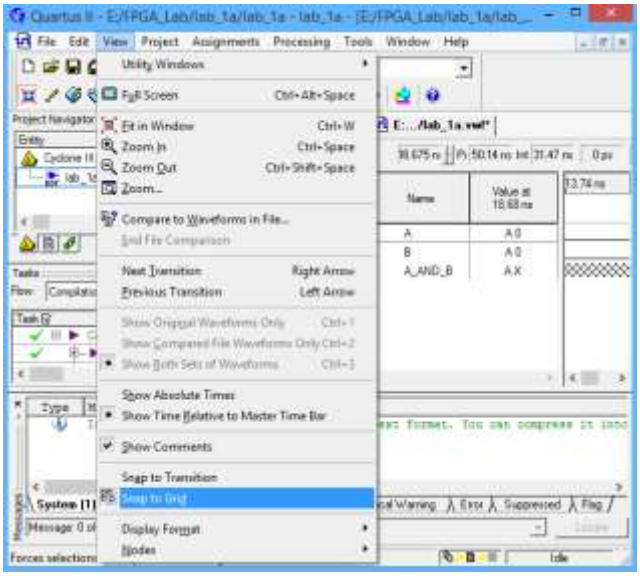
รูปที่ 44 การเพิ่ม Node เพื่อวิเคราะห์สัญญาณ

25. จากรูปที่ 1.44 จะเห็นได้ว่าการจัดเรียงลำดับสัญญาณที่โปรแกรมสร้างมาให้อาจทำให้สับสนในการพิจารณาซึ่งเราสามารถเปลี่ยนลำดับของสัญญาณได้โดยคลิกที่รูปพอร์ตแล้วลากไปวางยังตำแหน่งที่ต้องการดังรูปที่ 1.45



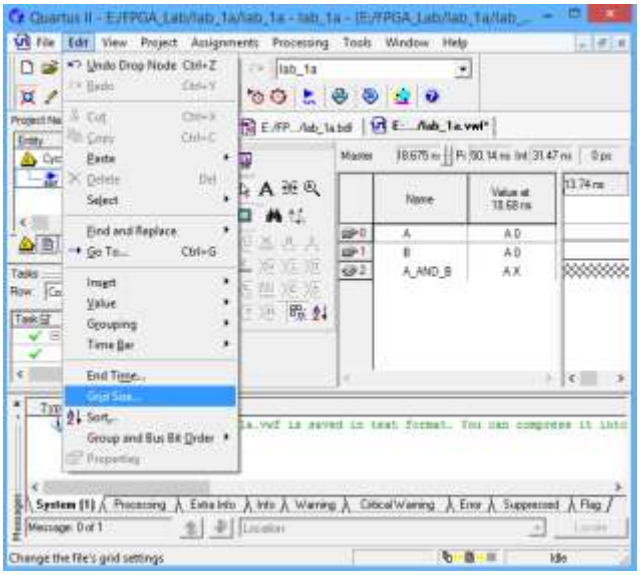
รูปที่ 1.45 ลำดับสัญญาณใน Waveform Editor

26. เพื่อให้ง่ายแก่การกำหนดค่าสัญญาณที่ช่วงเวลาต่าง ๆ ให้ทำการเปิด Option Snap to Grid ก่อน โดยเลือกจาก เมนู View -> Snap to Grid แต่ถ้ามีการเปิดการใช้งานอยู่แล้วให้ข้ามขั้นตอนนี้ไปได้เลย



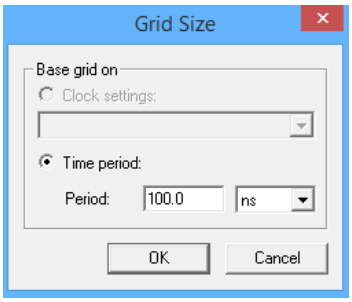
รูปที่ 1.46 การแสดงฟังก์ชัน Snap to Grid

27 กำหนดขนาดของ Grid โดยเลือกจากเมนู Edit -> Grid Size…



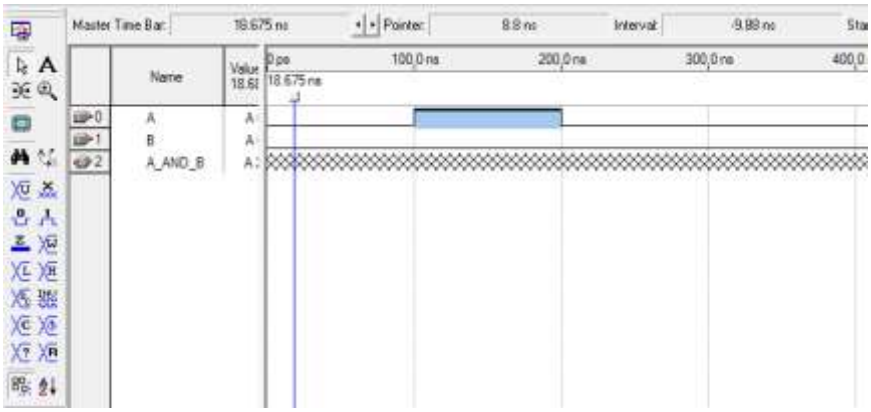
รูปที่ 1.47 กำหนดขนาดของ Grid

28. หลังจากนั้นจะมีหน้าต่าง Grid Size ขึ้นมา ให้ป้อนค่า 100.0ns แล้วคลิก OK จะสังเกตได้ว่าในขณะที่หน้าต่าง Vector Waveform Editor จะมีเส้นประแนวตั้งปรากฏขึ้นทุก ๆ 100ns



รูปที่ 1.48 กำหนดขนาดของ Grid เท่ากับ 100 ns

29. กำหนดสัญญาณอินพุท A และ B โดยเริ่มจากการกำหนดค่าสัญญาณลอจิกให้แก่สัญญาณลอจิกให้แก่สัญญาณ A ในช่วงเวลา 100 us จนถึง 200 us ให้เป็น “1” โดยคลิกที่ค่าเวลา 100 us บนสัญญาณ A แล้วลากแถบสีทึบไปจนถึงค่าเวลา 200 us จากนั้นคลิกที่ปุ่ม  บนแถบเครื่องมือบนปุ่มรูปพัลส์ “1”

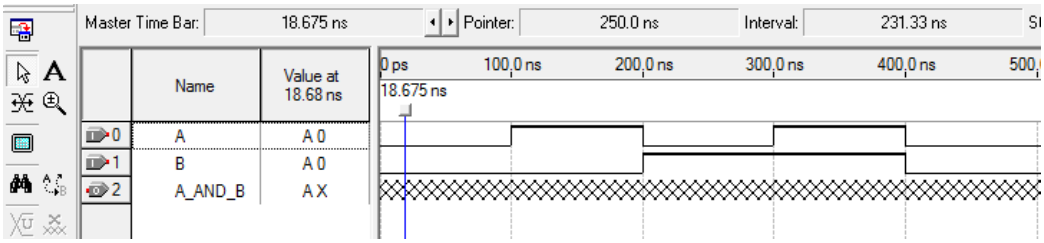


รูปที่ 1.49 คลิกเลือกช่วงรูปคลื่นที่ต้องการกำหนดค่าระดับสัญญาณลอจิก

30. กำหนดค่าสัญญาณอินพุทในช่วงเวลาที่เหลือดังนี้

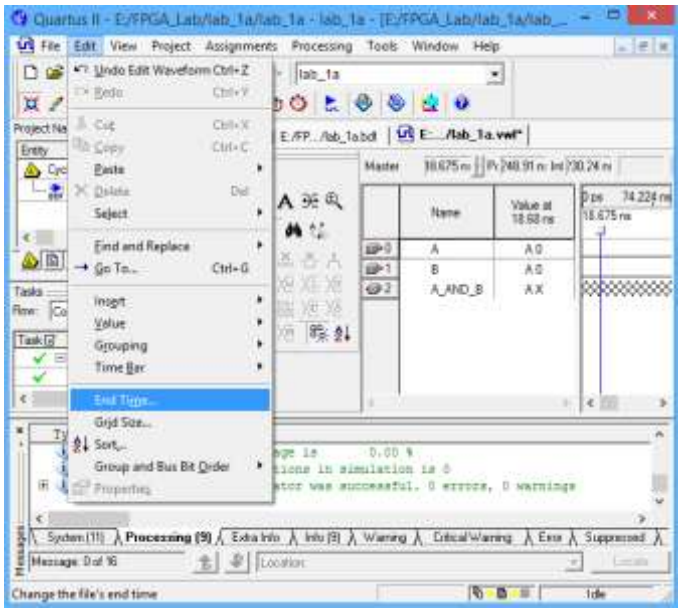
สัญญาณ A เป็น “1” ในช่วงเวลา 100us-200us และ 300us-400us

สัญญาณ B เป็น “1” ในช่วงเวลา 200us-400us



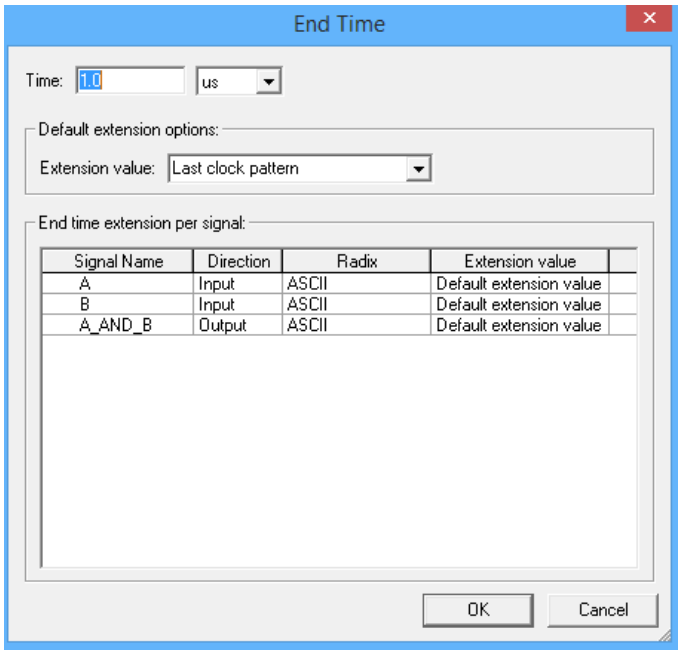
รูปที่ 1.50 รูปคลื่นอินพุท A และ B สำหรับจำลองการทำงาน

31. กำหนดช่วงเวลาสิ้นสุดของการจำลองการทำงาน (End Time) โดยใช้เมนู File -> End Time



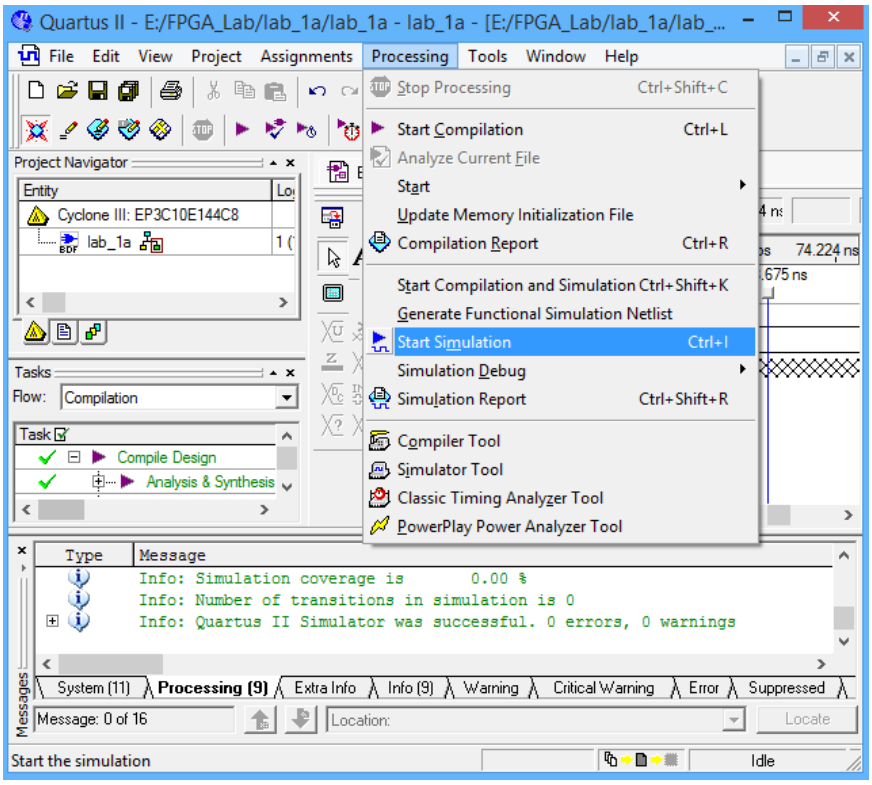
รูปที่ 1.51 กำหนดเวลาสิ้นสุดการจำลองการทำงาน

32. จากนั้นกำหนดให้ช่วงเวลาสิ้นสุดการจำลองการทำงานมีค่าเท่ากับ 1 us เสร็จแล้วคลิกที่ปุ่ม OK



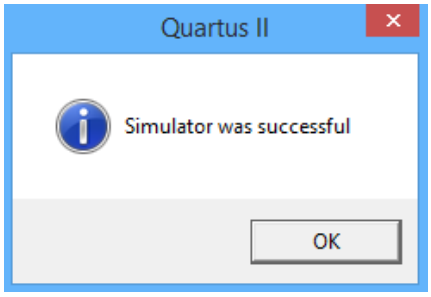
รูปที่ 1.52 กำหนดจุดสิ้นสุดการจำลองการทำงานที่ 1us

33. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

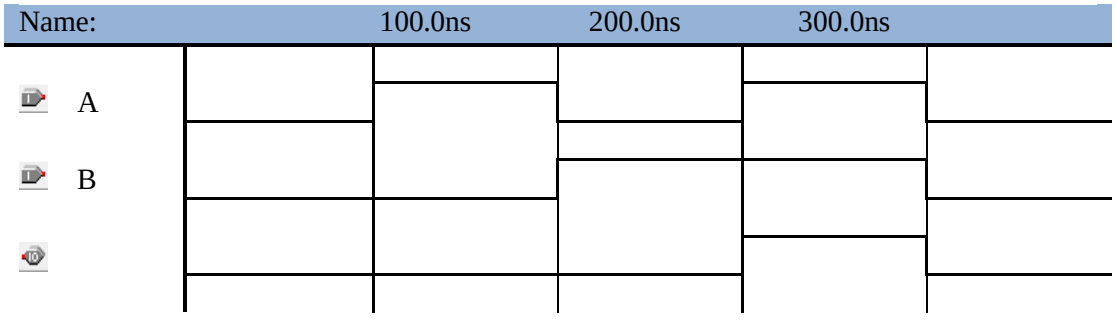


รูปที่ 1.53 การเรียกหน้าต่าง Simulator จากเมนู

34. จากนั้นคลิกที่ปุ่ม OK ดังรูปที่ เสร็จแล้วดูผลการจำลองการทำงานดังรูปที่ 1.54



รูปที่ 1.54 หน้าต่างแจ้งผลการจำลองการทำงาน



รูปที่ 1.55 ผลการจำลองการทำงาน

35. วิเคราะห์รูปคลื่นที่ค่าเวลาต่าง ๆ แล้วป้อนค่าลอจิกลงในตาราง

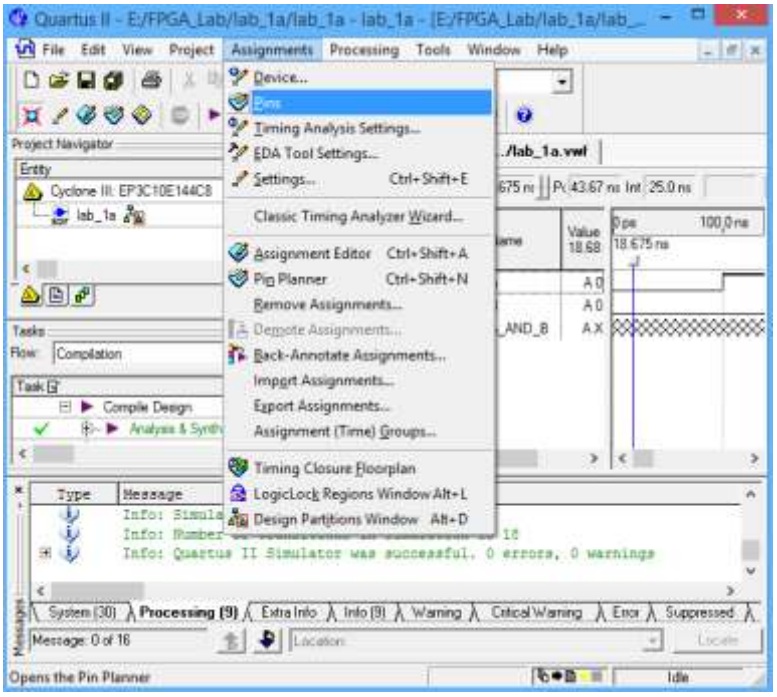
A	B	A_AND_B
0	0	
0	1	
1	0	
1	1	

ตารางที่ 1.8 ตารางความจริงของแอนด์เกท 2 อินพุท

36. จากขั้นตอนการคอมไพล์ที่ผ่านมา นอกจากคอมไพล์เลอร์จะคำนวณค่าความหน่วงเวลาเพื่อจำลองการทำงานแล้ว มันยังสร้างไฟล์สำหรับโปรแกรมลงไอซี FPGA ให้เราอีกด้วย

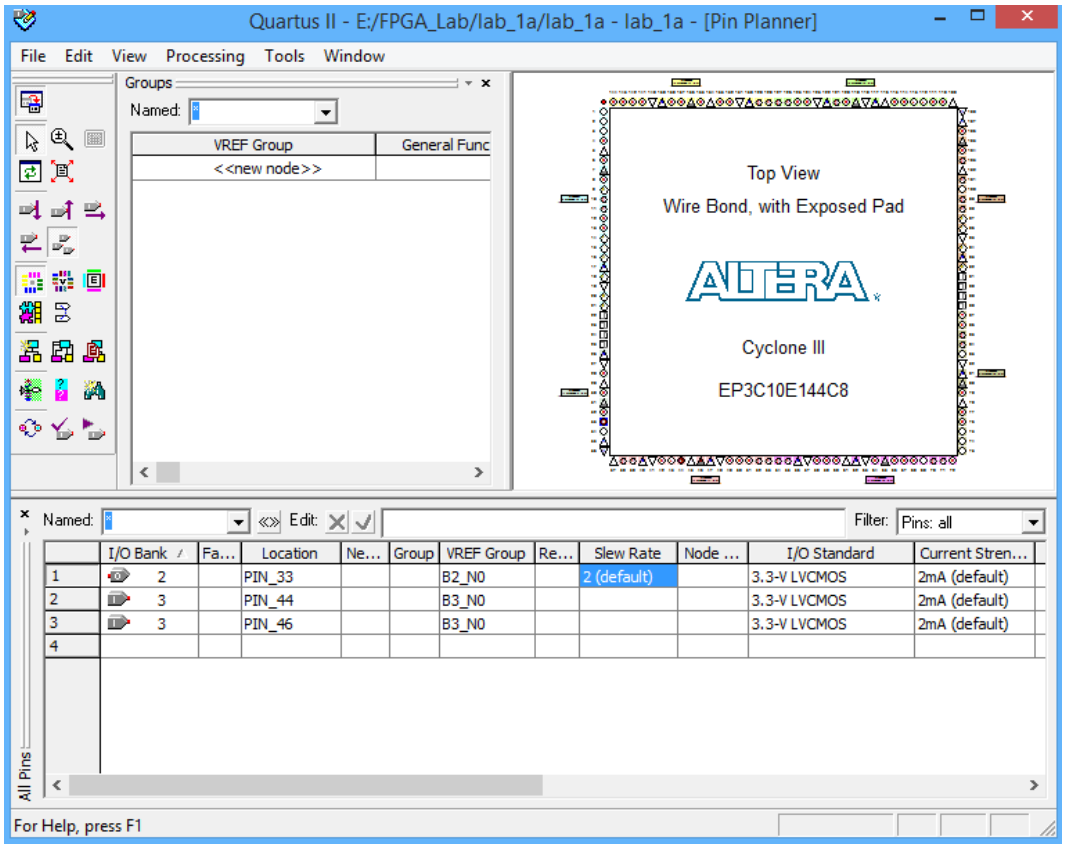
การเปลี่ยนตำแหน่งของขา I/O และการโปรแกรมลงไอซี

37. หลังจากที่เราทำการคอมไพล์วงจรแล้ว โปรแกรม Quartus II 8.1 Web Edition จะทำการกำหนดขา I/O ต่าง ๆ ในวงจรเองโดยอัตโนมัติ ซึ่งสามารถดูได้จากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins



รูปที่ 1.56 การเรียกหน้าต่าง Pin Planner

38. จากนั้นจะมีหน้าต่าง Pin Planner ปรากฏขึ้นมา แสดงดังรูปที่ 1.57



รูปที่ 1.57 หน้าต่าง Pin Planner

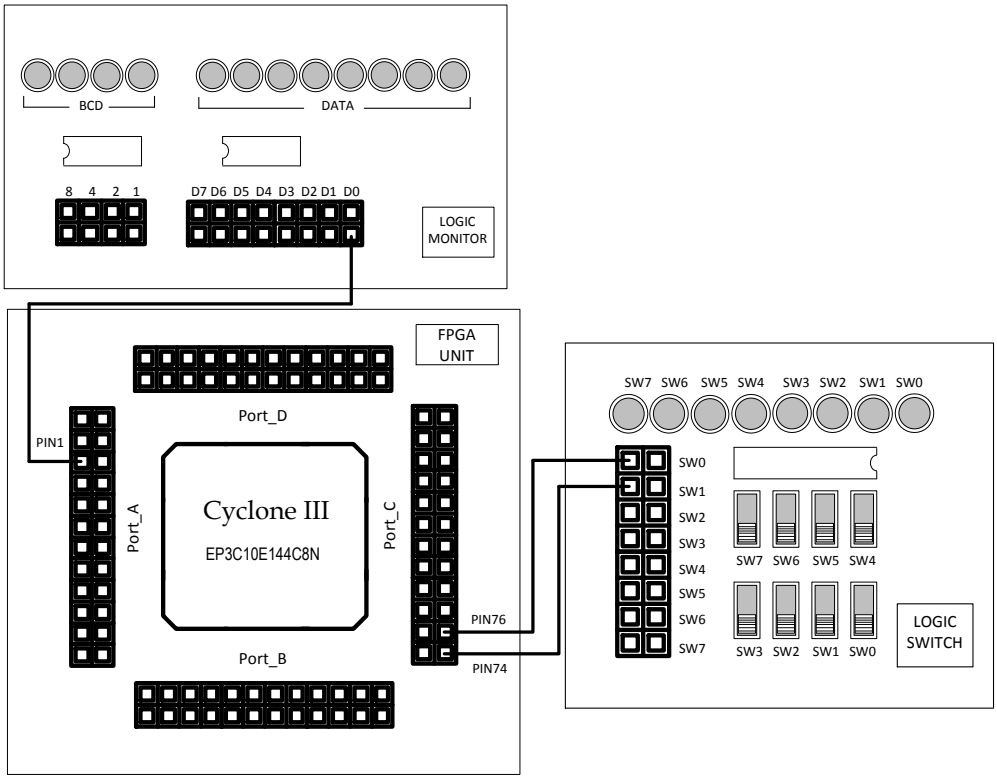
34. จากนั้นให้ทำการเปลี่ยนค่าอินพุท และเอาต์พุทในช่อง Location และช่อง I/O Standard ดังตารางที่ 1.9

I/O Bank	Location	I/O Standard
A	74	3.3V LVCMOS
B	76	3.3V LVCMOS
A_AND_B	1	3.3V LVCMOS

ตารางที่ 1.9 การตั้งค่าไอซีเพื่อใช้งานจริง

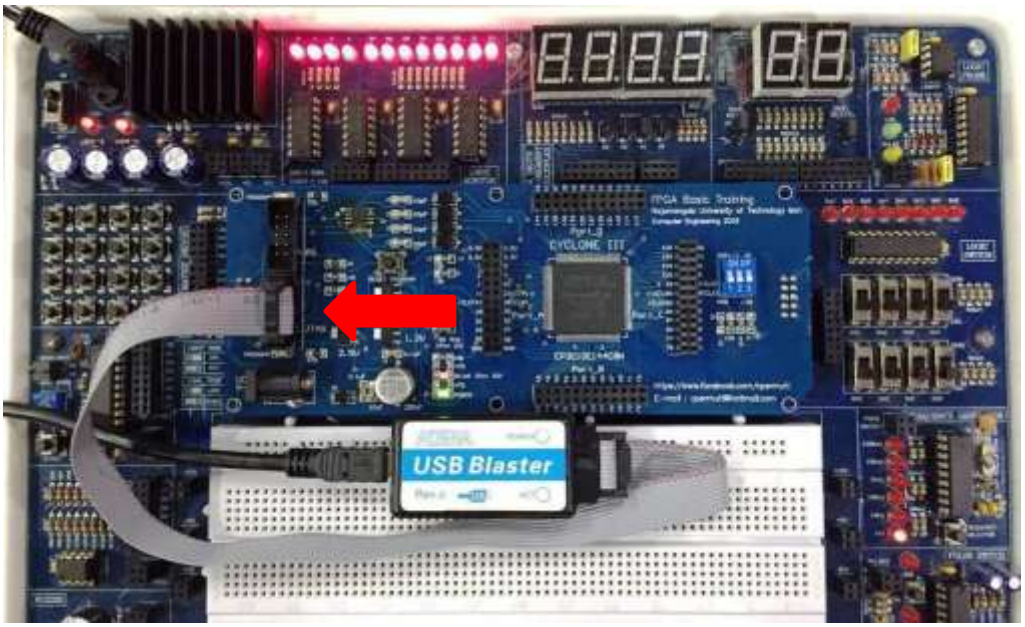
35. เมื่อเปลี่ยนตำแหน่งขา เรียบร้อยแล้วให้คอมไพล์ใหม่อีกครั้ง

36. ต่อวงจรในชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA ดังรูปที่ 1.58



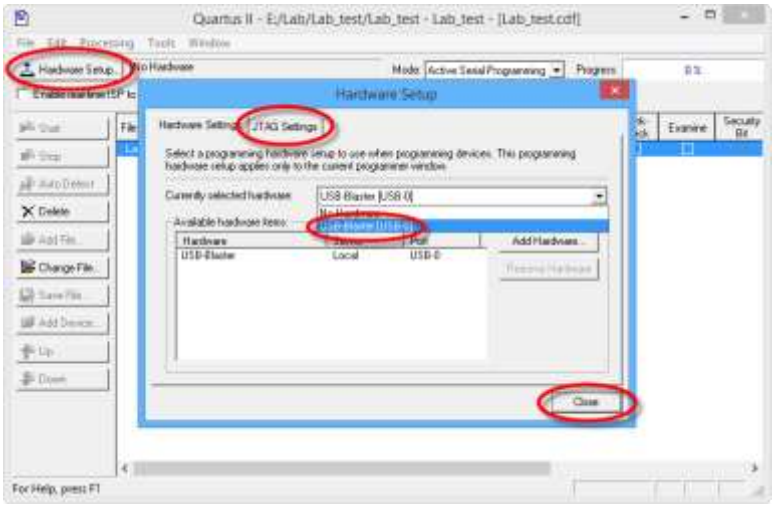
รูปที่ 1.58 การต่อวงจรบนชุดทดลอง

37. เชื่อมต่อสาย USB Blaster เข้าที่พอร์ต USB ของคอมพิวเตอร์ และปลายอีกด้านหนึ่งเชื่อมต่อเข้ากับซ็อกเกท JTAG ที่อยู่บนชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA ดังรูปที่ 1.59



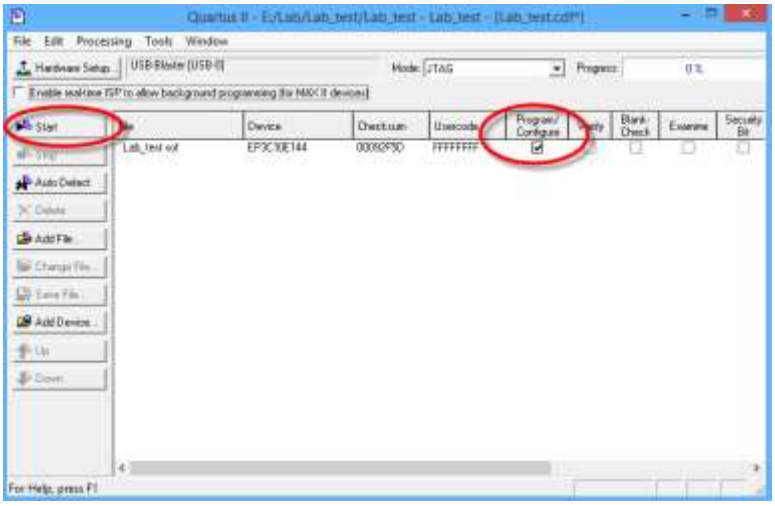
รูปที่ 1.59 ต่อสาย USB Blaster เข้ากับชุดทดลอง

38. จากนั้นเลือกที่เมนู Tools -> Programmer จะปรากฏหน้าต่าง Programmer ขึ้นมาดังรูปที่ 1.60 ให้ทำการเลือกที่ปุ่ม Hardware Setup จะมีหน้าต่าง Hardware Setup ในช่อง Currently Selected Hardware ให้เลือกที่ USB Blaster [USB-0] หลังจากนั่นคลิกเลือกที่ปุ่ม Close



รูปที่ 1.60 การเลือกสายดาวน์โหลดโปรแกรม

39. หลังจากเลือกสายดาวน์โหลดแล้ว ให้คลิก Enable ที่ช่อง Program/Configure ดังรูปที่ 1.61



รูปที่ 1.61 การโปรแกรมลงในชุดทดลอง

40. หลังจากนั้นเลือกที่ปุ่ม Start เพื่อโปรแกรมข้อมูลลงชุดทดลอง

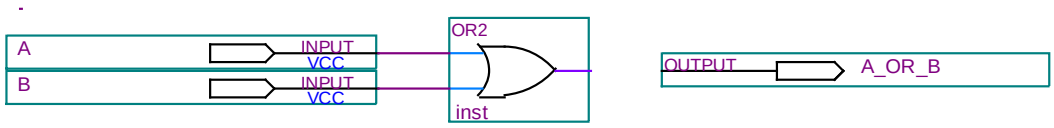
41. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิกที่ LED D0 จากนั้นบันทึกผลลงในตาราง

A (SW1)	B (SW0)	A_AND_B (LED DO)
LOW	LOW	
LOW	HIGH	
HIGH	LOW	
HIGH	HIGH	

ตารางที่ 10 ตารางความจริงของแอนด์เกท 2 อินพุท

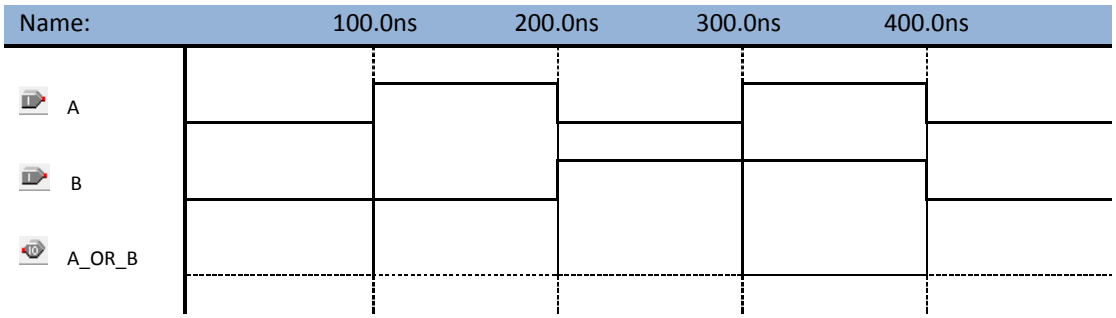
ขั้นตอนการทดลอง ตอนที่ 2 ออร์เกท

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำตามขั้นตอนเดิม สร้างโปรเจคใหม่ในชื่อ lab_1b โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิกที่ OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ or2, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป



รูปที่ 1.62 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_1b.gdf
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_1b.scf เพื่อจำลองการทำงานของวงจร แล้วบันทึกผลการจำลองการทำงาน



รูปที่ 1.63 ผลการจำลองการทำงานของวงจร

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

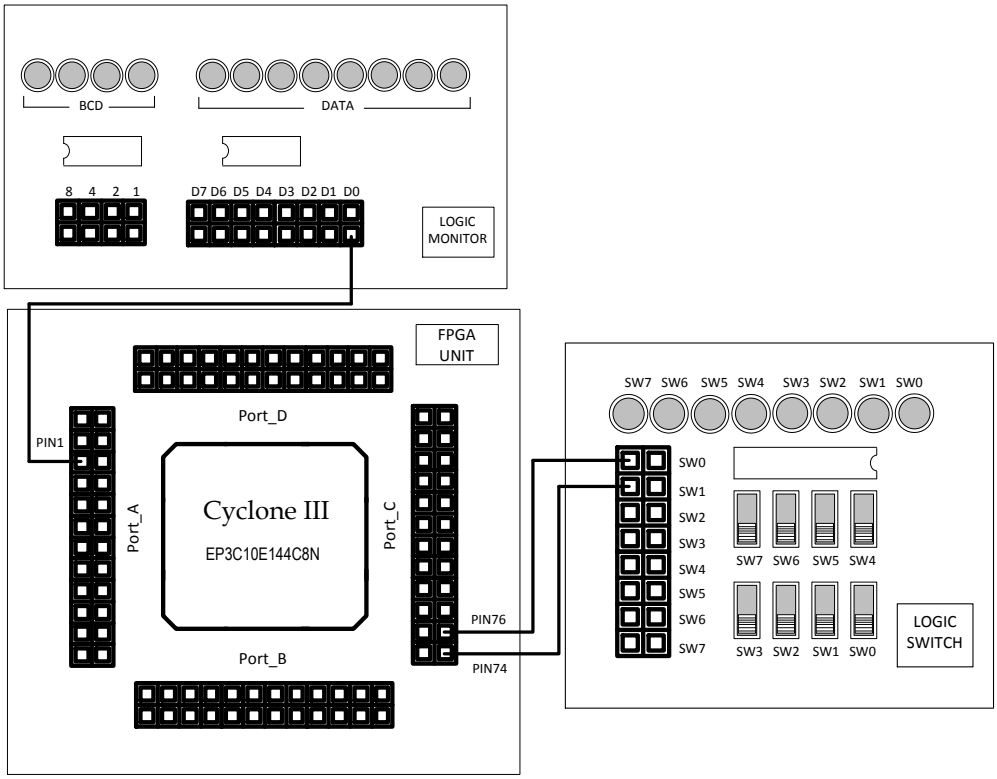
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	**ตำแหน่งขาบน FPGA**
A	67
B	74
A_OR_B	1

ตารางที่ 1.11 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 1.75 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 1.75 การต่อวงจรบนชุดทดลอง

13. ทดสอบเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิกที่ LED D0 จากนั้นบันทึกผลลงในตาราง

A (SW1)	**B (SW0)**	**A_OR_B (LED DO)**
LOW	LOW	
LOW	HIGH	
HIGH	LOW	
HIGH	HIGH	

ตารางที่ 1.12 ตารางความจริงของออร์เกท 2 อินพุท

ขั้นตอนการทดลอง ตอนที่ 3 นอตเกท

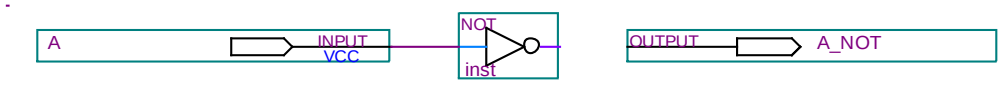
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำตามขั้นตอนเดิม สร้างโปรเจคใหม่ในชื่อ lab_1c โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ not, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

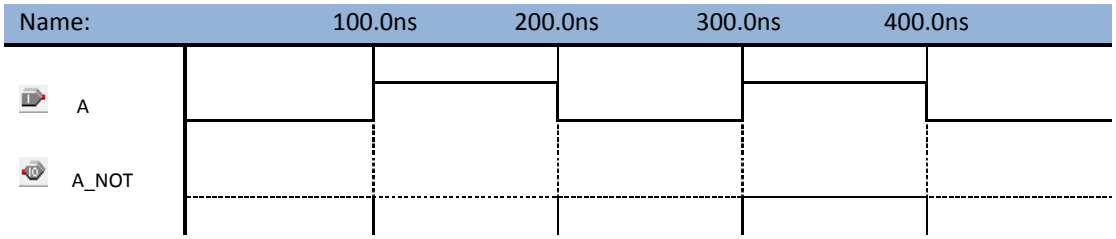


รูปที่ 1.76 วงจรสำหรับการทดลองตอนที่ 3

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_1c.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_1c.scf เพื่อจำลองการทำงานของวงจร แล้วบันทึกผลการจำลองการทำงาน



รูปที่ 1.77 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

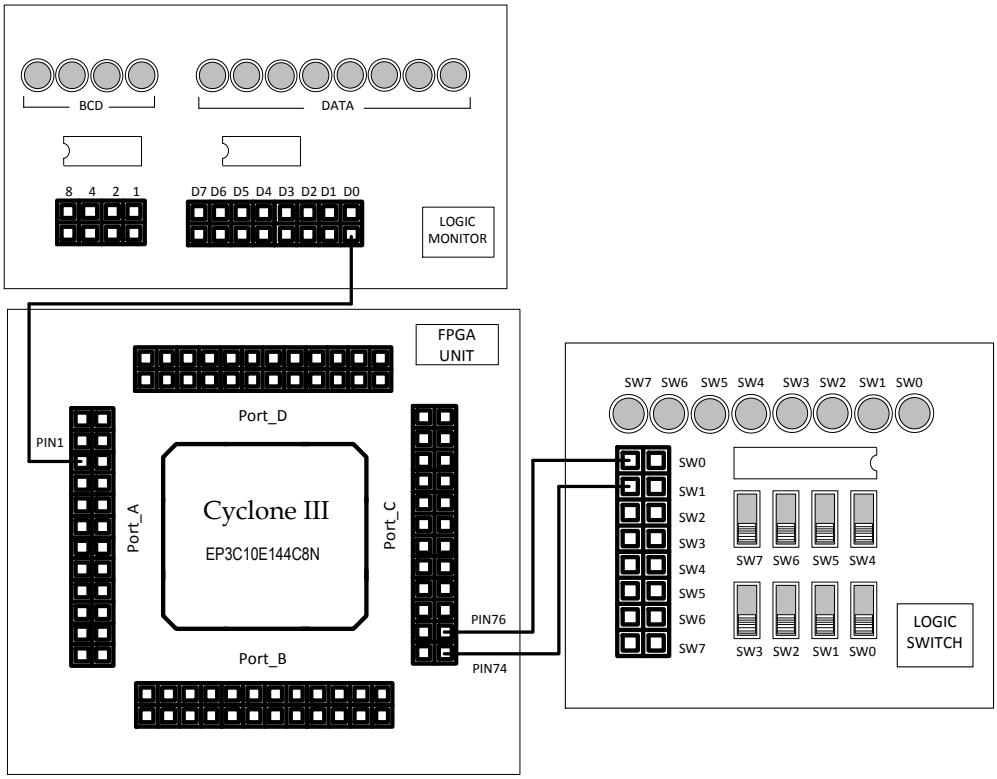
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A	76
A_NOT	1

ตารางที่ 1.13 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 1.78 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 1.78 การต่อวงจรบนชุดทดลอง

13. ทดสอบเปลี่ยนสถานะของสวิตช์ SW0 แล้วดูผลทางลอจิกที่ LED D0 จากนั้นบันทึกผลลงในตาราง

A (SW0)	**A_NOT (LED D0)**
LOW	
HIGH	

ตารางที่ 1.14 ตารางความจริงของนอตเกท

ขั้นตอนการทดลอง ตอนที่ 4 แนนด์เกท

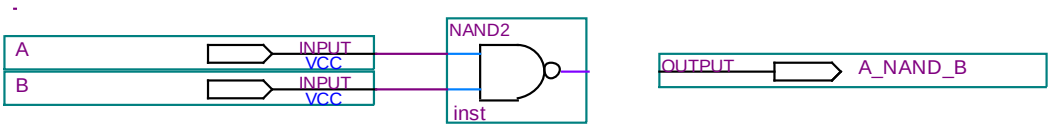
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำตามขั้นตอนเดิม สร้างโปรเจคใหม่ในชื่อ lab_1d โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ nand2, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

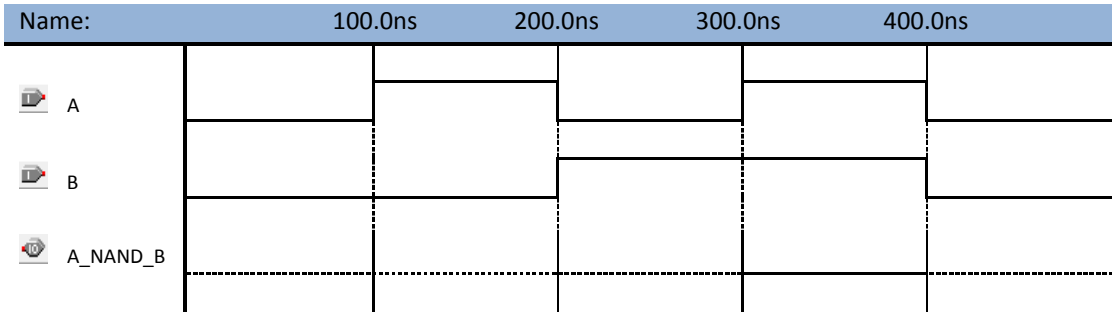


รูปที่ 1.79 วงจรสำหรับการทดลองตอนที่ 4

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_1d.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_1d.scf เพื่อจำลองการทำงานของวงจร แล้วบันทึกผลการจำลองการทำงาน



รูปที่ 1.80 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

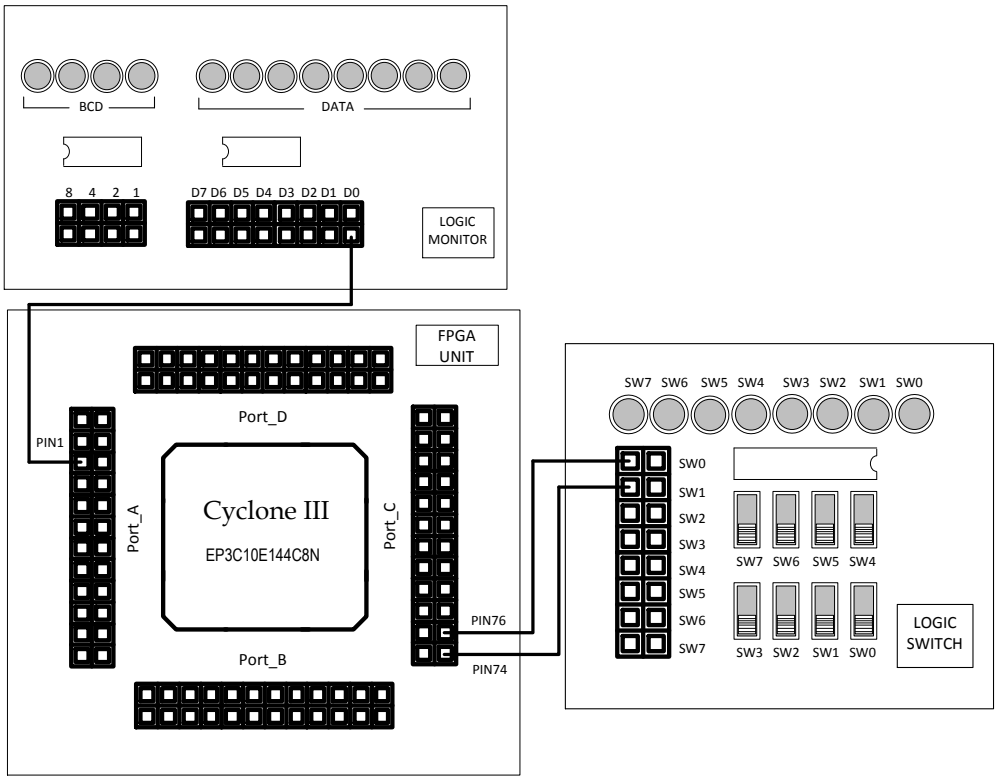
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A	76
B	74
A_NAND_B	1

ตารางที่ 1.15 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 1.81 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 1.81 การต่อวงจรบนชุดทดลอง

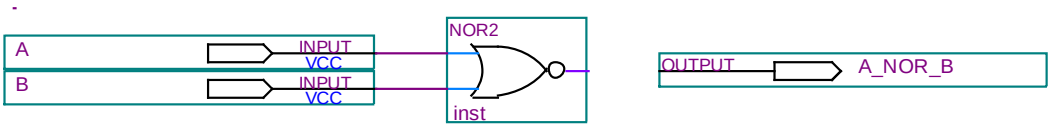
13. ทดสอบเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิกที่ LED D0 จากนั้นบันทึกผลลงในตาราง

A (SW1)	**B (SW0)**	**A_NAND_B (LED D0)**
LOW	LOW	
LOW	HIGH	
HIGH	LOW	
HIGH	HIGH	

ตารางที่ 1.16 ตารางความจริงของแนนด์เกท 2 อินพุต

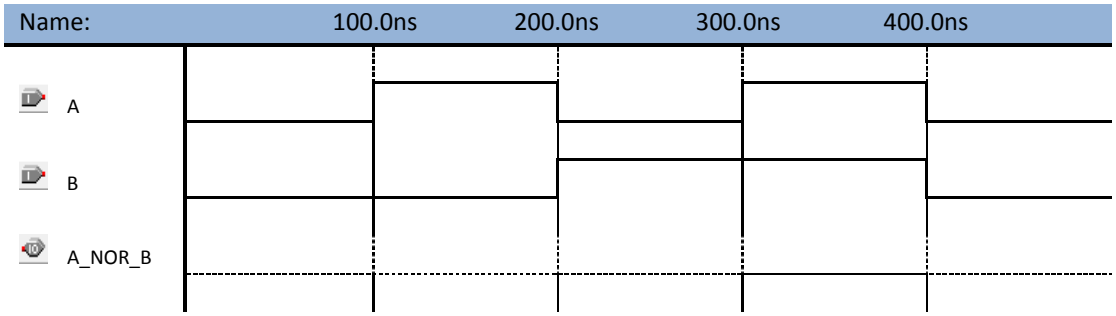
ขั้นตอนการทดลอง ตอนที่ 5 นอร์เกท

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำตามขั้นตอนเดิม สร้างโปรเจคใหม่ในชื่อ lab_1e โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ nor2, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป



รูปที่ 1.82 วงจรสำหรับการทดลองตอนที่ 5

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_1e.gdf
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_1e.scf เพื่อจำลองการทำงานของวงจร แล้วบันทึกผลการจำลองการทำงาน



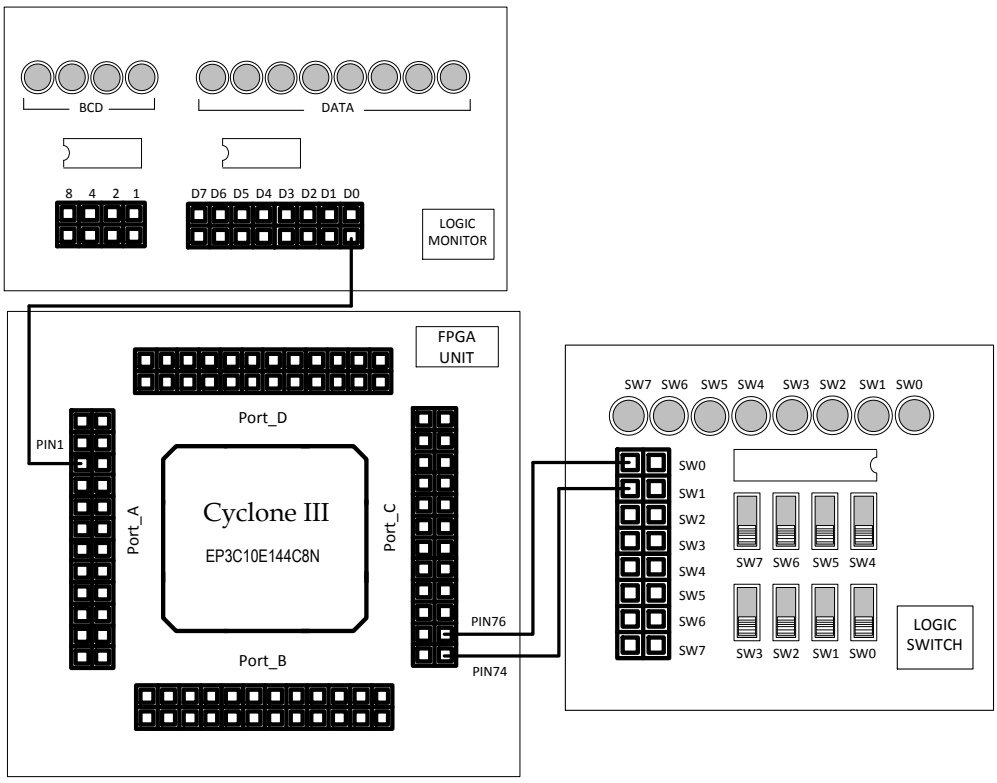
รูปที่ 1.83 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins
11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A	76
B	74
A_NOR_B	1

ตารางที่ 1.17 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 1.84 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 1.84 การต่อวงจรบนชุดทดลอง

13. ทดสอบเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิกที่ LED จากนั้นบันทึกผลลงในตาราง

A (SW1)	B (SW0)	A_NOR _B (LED D0)
LOW	LOW	
LOW	HIGH	
HIGH	LOW	
HIGH	HIGH	

ตารางที่ 1.18 ตารางความจริงของนอร์เกท 2 อินพุต

ขั้นตอนการทดลอง ตอนที่ 6 เอ็กซ์คลูซีฟออร์เกท

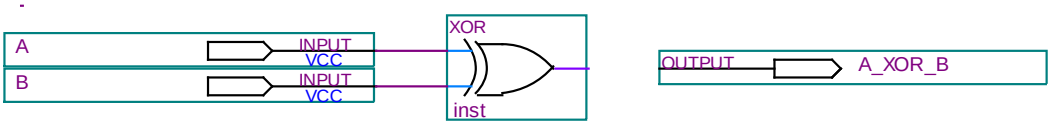
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำตามขั้นตอนเดิม สร้างโปรเจคใหม่ในชื่อ lab_1f โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ xor, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

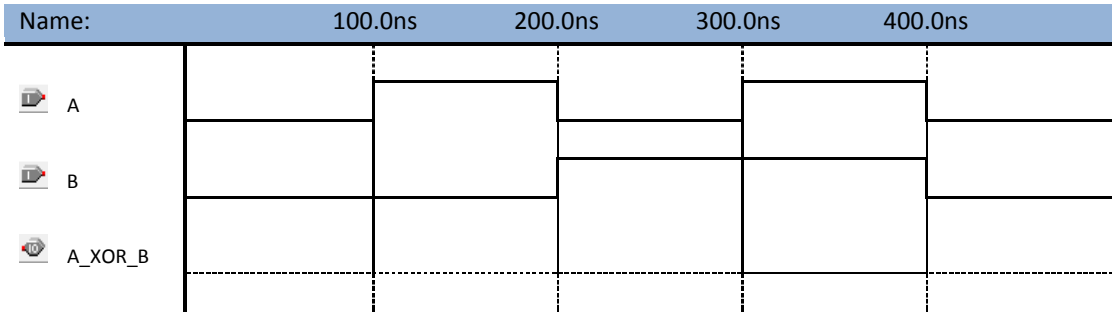


รูปที่ 1.85 วงจรสำหรับการทดลองตอนที่ 6

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_1f.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_1f.scf เพื่อจำลองการทำงานของวงจร แล้วบันทึกผลการจำลองการทำงาน



รูปที่ 1.86 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

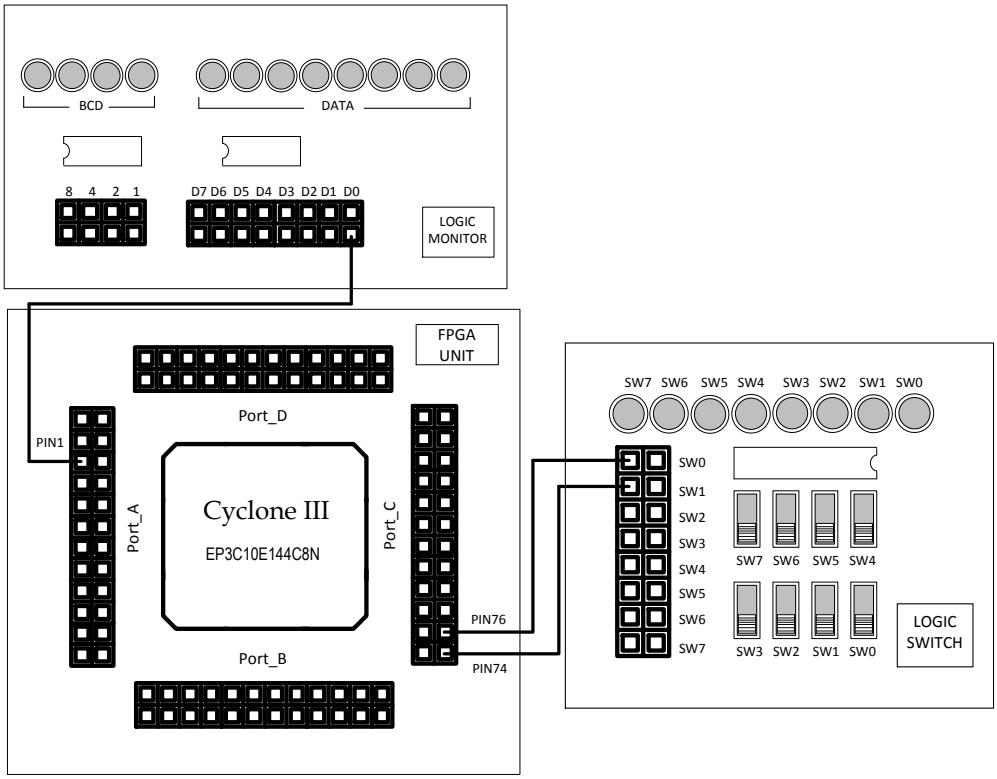
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	**ตำแหน่งขาบน FPGA**
A	76
B	74
A_XOR_B	1

ตารางที่ 1.19 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 1.87 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 1.87 การต่อวงจรบนชุดทดลอง

13. ทดสอบเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิกที่ LED จากนั้นบันทึกผลลงในตาราง

A (SW1)	**B (SW0)**	**A_XOR_B (LED D0)**
LOW	LOW	
LOW	HIGH	
HIGH	LOW	
HIGH	HIGH	

ตารางที่ 1.20 ตารางความจริงของเอ็กซ์คลูซีฟออร์เกท 2 อินพุต

ขั้นตอนการทดลอง ตอนที่ 7 เอ็กซ์คลูซีฟนอร์เกท

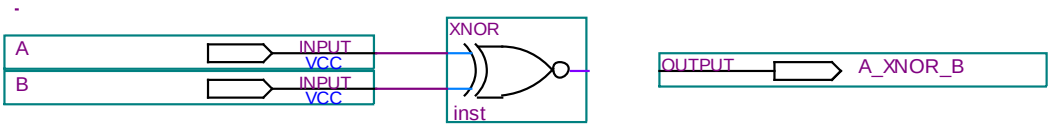
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำตามขั้นตอนเดิม สร้างโปรเจคใหม่ในชื่อ lab_1g โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ xnor, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

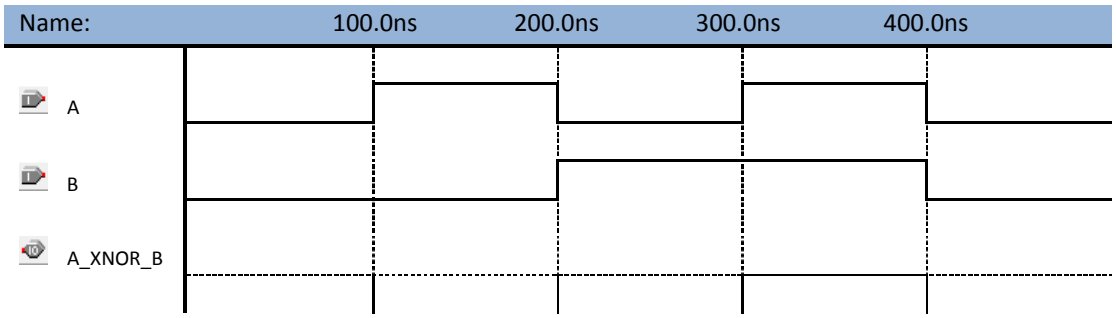


รูปที่ 1.88 วงจรสำหรับการทดลองตอนที่ 7

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_1g.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_1g.scf เพื่อจำลองการทำงานของวงจร แล้วบันทึกผลการจำลองการทำงาน



รูปที่ 1.89 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

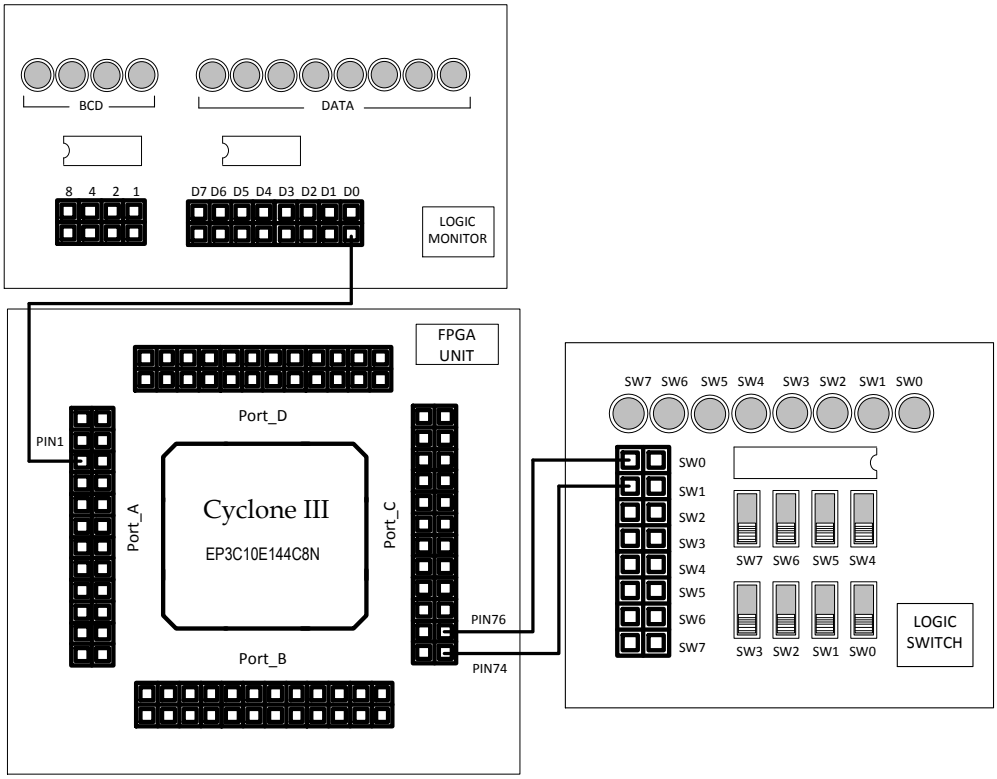
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A	76
B	74
A_XNOR_B	1

ตารางที่ 1.21 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 1.90 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 1.90 การต่อวงจรบนชุดทดลอง

13. ทดสอบเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิกที่ LED จากนั้นบันทึกผลลงในตาราง

A (SW1)	B (SW0)	A_XNOR_B (LED D0)
LOW	LOW	
LOW	HIGH	
HIGH	LOW	
HIGH	HIGH	

ตารางที่ 1.22 ตารางความจริงของเอ็กซ์คลูซีฟนอร์เกท 2 อินพุต

สรุปผลการทดลอง

การทดลองที่ 2

ลอจิกเกทที่มีเอาต์พุท 3 สถานะ(Tristate Logic)

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของลอจิกเกทแบบสามสถานะ
3. เรียนรู้การวาดสายเชื่อมต่อแบบบัส รวมถึงการแยก และการรวมสัญญาณบนสายเชื่อมต่อแบบบัสได้
4. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากลอจิกเกทที่มีเอาต์พุทแบบ 3 สถานะได้
5. สามารถดาวน์โหลดวงจรไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

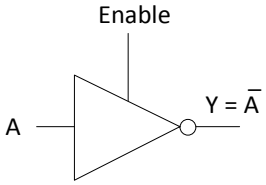
อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

ทฤษฎี

ลอจิกเกทที่เราได้เรียนรู้มาในการทดลองที่ 1 เป็นลอจิกเกทแบบพื้นฐานซึ่งสามารถให้ค่าเอาต์พุทในทางลอจิกเป็นระดับสูง และต่ำเท่านั้นแต่ยังมีลอจิกเกทอีกประเภทหนึ่งซึ่งให้เอาต์พุทได้มากกว่า 2 สถานะ โดยลอจิกเกทประเภทนี้เราเรียกว่า ลอจิกเกทที่มีเอาต์พุท 3 สถานะ (Tristate Logic) แต่ส่วนมากเราจะนิยมเรียกทับศัพท์ว่า ไทรสเตทลอจิกเกท

ไทรสเตทลอจิกเกท สามารถให้เอาต์พุททั้งหมด 3 สถานะ ได้แก่ ลอจิกสูง ลอจิกต่ำ และสถานะอิมพีแดนซ์สูง (High Impedance) ซึ่งสถานะอิมพีแดนซ์สูงเป็นสถานะที่เสมือนลอจิกเกทตัวนั้นถูกตัดขาดออกจากวงจร สำหรับลอจิกเกทประเภทนี้จะมีขาอินพุทเพิ่มมาอีก 1 ขา นั่นคือขา Enable ซึ่งเป็นขาที่ใช้ควบคุมให้เอาต์พุทของลอจิกเกทมีค่าเป็นลอจิก (สูง-ต่ำ) หรือสภาวะอิมพีแดนซ์สูง ดังตัวอย่างตารางความจริงของนอตเกทแบบไทรสเตท



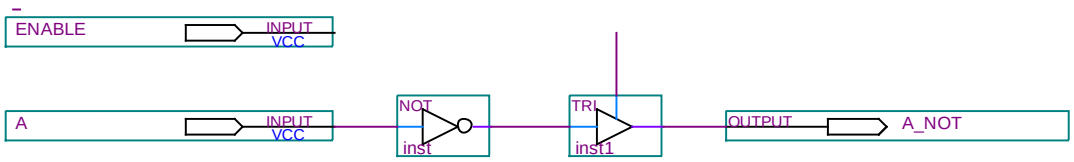
อินพุท		เอาต์พุท
EN	**A**	**Y**
0	0	Hi-Z
0	1	Hi-Z
1	0	1
1	1	0

รูปที่ 2.1 นอตเกทแบบไทรสเตท ตารางที่ 2.1 ตารางความจริงของนอตเกทแบบไทรสเตท

ไทรสเตทลอจิกถูกนำมาใช้ประโยชน์อย่างมากในบัสสื่อสารข้อมูลของระบบคอมพิวเตอร์ ซึ่งมรการแชร์บัสข้อมูลร่วมกันระหว่างอุปกรณ์ต่าง ๆ ภายในระบบ โดยที่ในขณะอุปกรณ์ตัวไหนต้องการสื่อสารข้อมูลตัวควบคุมระบบจะส่งสัญญาณ Enable เพื่อเปิดให้อุปกรณ์ตัวนั้นรับ – ส่งข้อมูลได้ ส่วนอุปกรณ์ตัวอื่น ๆ จะไม่ได้รับสัญญาณ Enable ทำให้มีสภาพเหมือนอุปกรณ์ตัวอื่น ๆ ไม่ได้ต่ออยู่บนระบบบัสนั้น ๆ

ขั้นตอนการทดลอง ตอนที่ 1 การทำงานของลอจิกเกทที่มี 3 สถานะ

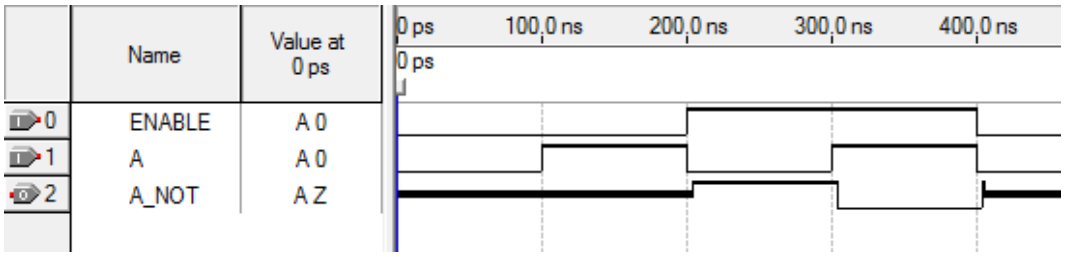
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_2a โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ not, tri, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป



รูปที่ 2.2 วงจรสำหรับทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_2a.gdf
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_2a.scf เพื่อจำลองการทำงานของวงจร แล้วสังเกตผลการจำลองการทำงานจะเห็นได้ว่าช่วงที่สัญญาณจากขา Enable มีค่าเป็น 0 ค่าสัญญาณทางเอาต์พุท

จะเป็นค่าอิมพีแดนซ์สูง โดยแสดงด้วยเส้นทึบหนาใน Waveform Editor และในช่วงสัญญาณ Enable เป็น 1 ค่าเอาต์พุทจะมีค่าตามการทำงานปกติของนอตเกท



รูปที่ 2.3 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

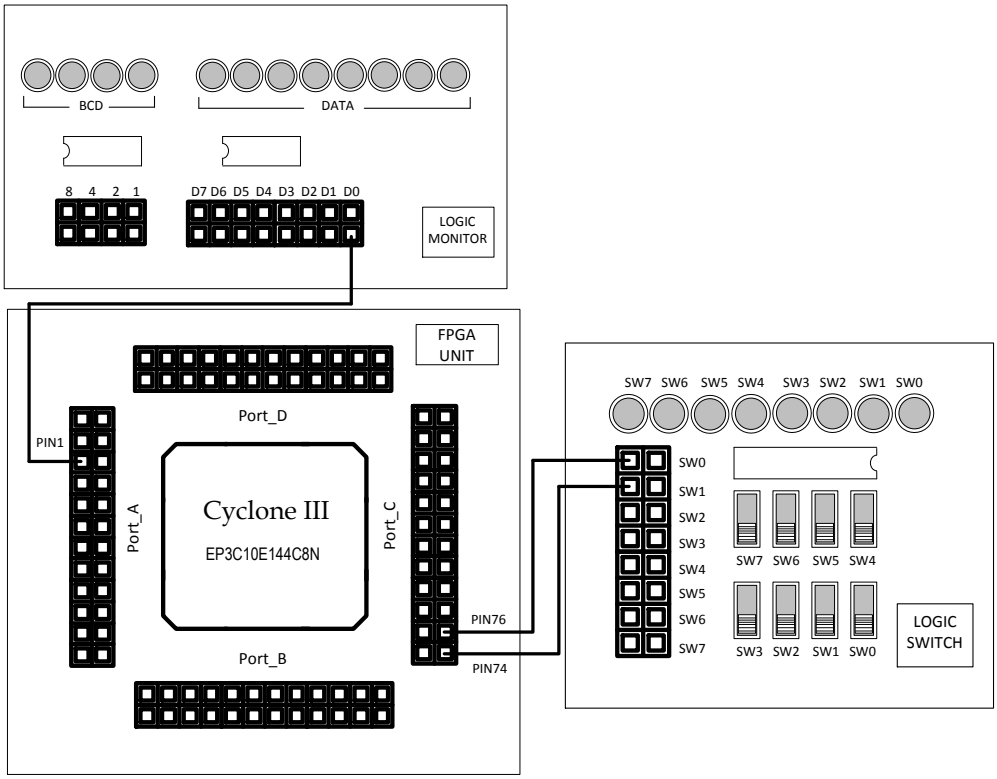
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	**ตำแหน่งขาบน FPGA**
A	76
ENABLE	74
A_NOT	1

ตารางที่ 2.2 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 2.4 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 2.4 การต่อวงจรบนชุดทดลอง

13. ทดสอบเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิกที่ LED D0 จากนั้นบันทึกผลลงในตาราง

ENABLE (SW1)	**A (SW0)**	**A NOT (LED D0)**
LOW	LOW	
LOW	HIGH	
HIGH	LOW	
HIGH	HIGH	

ตารางที่ 2.3 ตารางความจริงจากการทดลองตอนที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 การนำลอจิกเกทที่มี 3 สถานะมาประยุกต์ในการส่งข้อมูลบนบัสร่วมกัน

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_2b โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

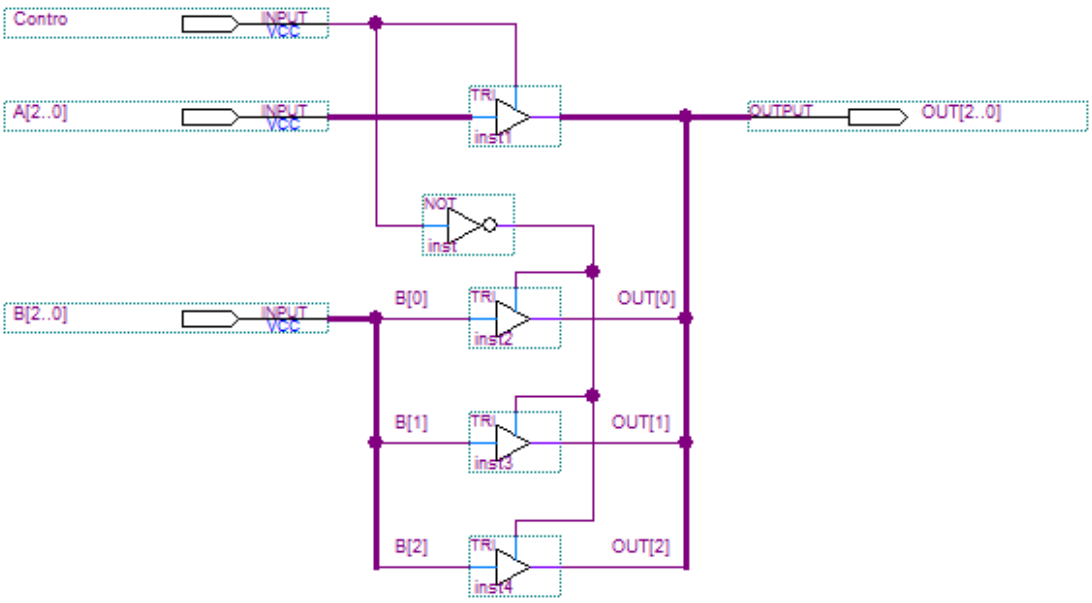
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ not, tri, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูปที่ 2.5 ซึ่งมีจุดที่ควรสังเกตดังนี้

   a. สายสัญญาณในวงจรมีทั้งเส้นหนา และเส้นบาง ซึ่งสายสัญญาณเส้นบางคือสายสัญญาณเดี่ยว และสายสัญญาณเส้นหนาจะใช้แทนบัสสัญญาณ

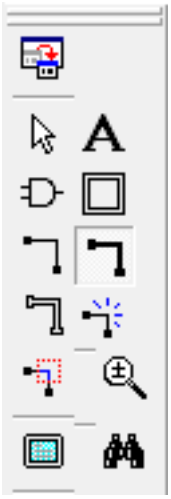
   b. ในการเชื่อมต่อสัญญาณเส้นเดี่ยวเขากับบัสจะต้องมีการกำหนดลาเบล (Label) ให้แก่มันด้วย

   c. อินพุทและเอาต์พุทที่เชื่อมต่อกับบัสจะต้องกำหนดเป็นแบบเวกเตอร์ นั่นคือมีการกำหนดขนาดให้แก่มันด้วย เช่น A[2..0] คือสายสัญญาณ A มีขนาด 3 บิต โดยการเรียงบิตจะเป็น (A2) (A1) (A0) หรือก็คือ MSB คือ A2 และ LSB คือ A0 นั่นเอง



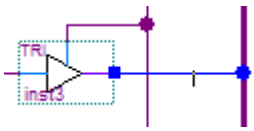
รูปที่ 2.5 วงจรสำหรับการทดลองตอนที่ 2

   d. การวาดสายสัญญาณแบบบัสแสดงดังรูปที่ 2.6 โดยเริ่มจากการคลิกที่ปุ่ม ⌐ ต่อจากนี้เราสามารถลากเส้นสัญญาณที่ต้องการให้เป็นบัสได้แล้ว

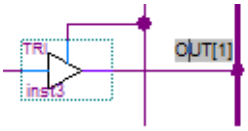


รูปที่ 2.6 ปุ่มที่อยู่ในโปรแกรม

e. การกำหนดลาเบลให้กับสายสัญญาณ และบัสดังรูปที่ 2.7a และ ดังรูปที่ 2.7b โดยเริ่มจากคลิกซ้ายเลือกสายสัญญาณหรือบัสที่ต้องการกำหนดชื่อลาเบล จากนั้นสายสัญญาณจะเปลี่ยนเป็นสีน้ำเงินพร้อมมีเคอร์เซอร์สี่เหลี่ยมขนาดเล็กกระพริบ ซึ่งขณะนี้เราสามารถพิมพ์ชื่อลาเบลที่ต้องการลงไปได้ทันที



รูปที่ 2.7a คลิกซ้ายบนสายสัญญาณที่ต้องการกำหนดลาเบล



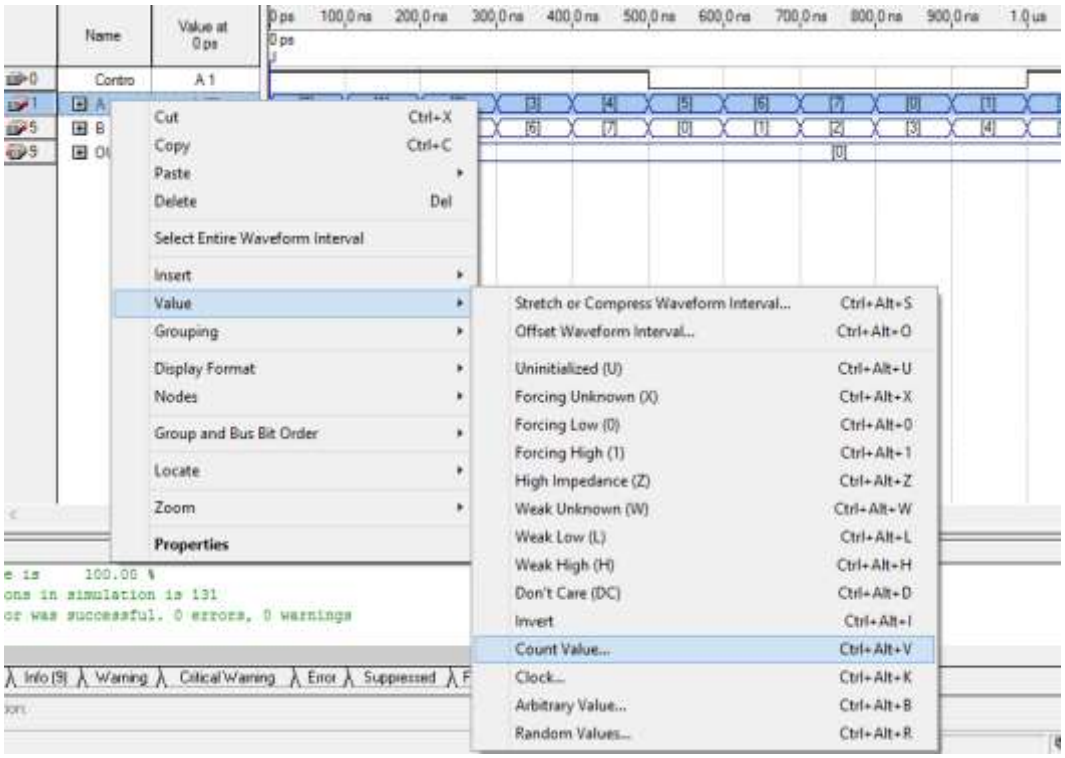
รูปที่ 2.7b พิมพ์ชื่อลาเบลที่ต้องการ

f. หากต้องการแก้ไขลาเบลที่เขียนไปแล้ว ให้ทำการดับเบิลคลิกที่ตัวหนังสือที่เป็นลาเบลแล้วให้พิมพ์ทับแก้ไขได้เลย

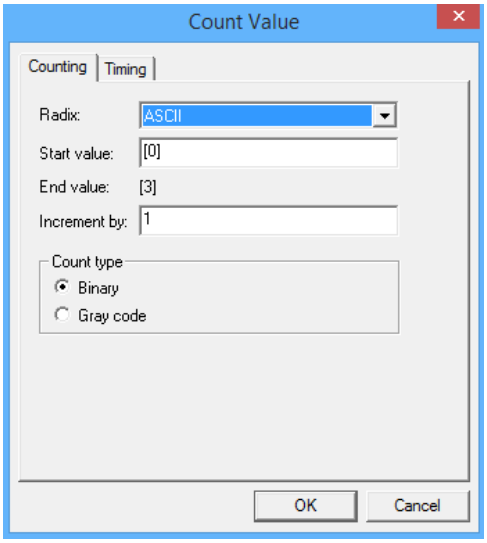
6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_2b.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

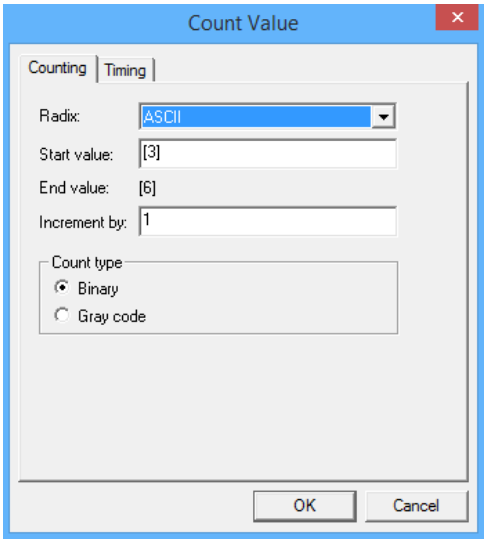
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_2a.scf เพื่อจำลองการทำงานของวงจร โดยสัญญาณที่จะนำมาทำการจำลองการทำงานได้แก่ Control (1), A[2..0] (1), B[2..0] และ OUT[2..0] (0) สำหรับการกำหนดสัญญาณเพื่อการจำลองการทำงานให้กำหนดสัญญาณ A และ B เป็นแบบ Count ซึ่งทำได้โดยการคลิกขวาที่ชื่อสัญญาณ จากนั้นจะมีเมนู Pop-up ปรากฏขึ้น ให้เลือกที่เมนู Value -> Count Value… แล้วกำหนดค่า Starting Value ของสัญญาณ A เป็น 0 และ Staring Value ของสัญญาณ B มีค่าเท่ากับ 3 ดังรูปที่ 2.9a และ รูปที่ 2.9b ตามลำดับ จากนั้นจำลองการทำงานและบันทึกผลการทดลอง



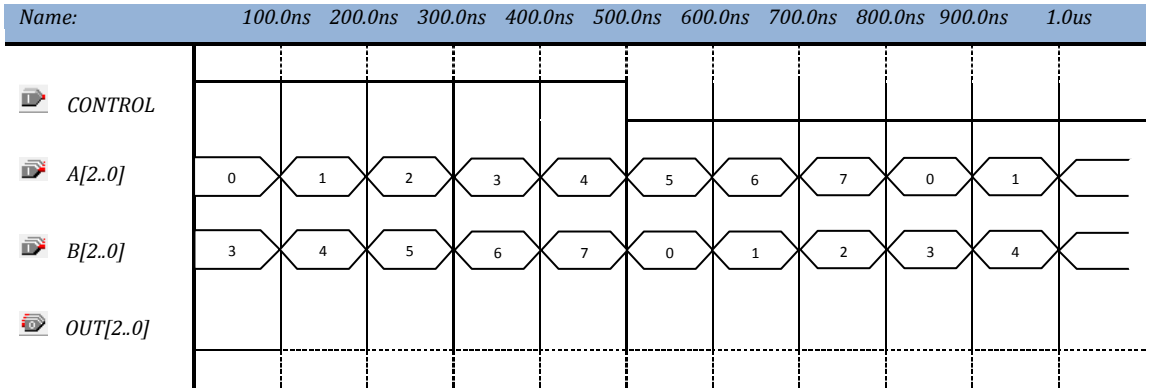
รูปที่ 2.8 คลิกขวาที่สัญญาณเพื่อเรียกเมนู Pop-up



รูปที่ 2.9a การกำหนดค่าเริ่มต้นให้แก่สัญญาณ A



รูปที่ 2.9b การกำหนดค่าเริ่มต้นให้แก่สัญญาณ B



รูปที่ 2.10 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

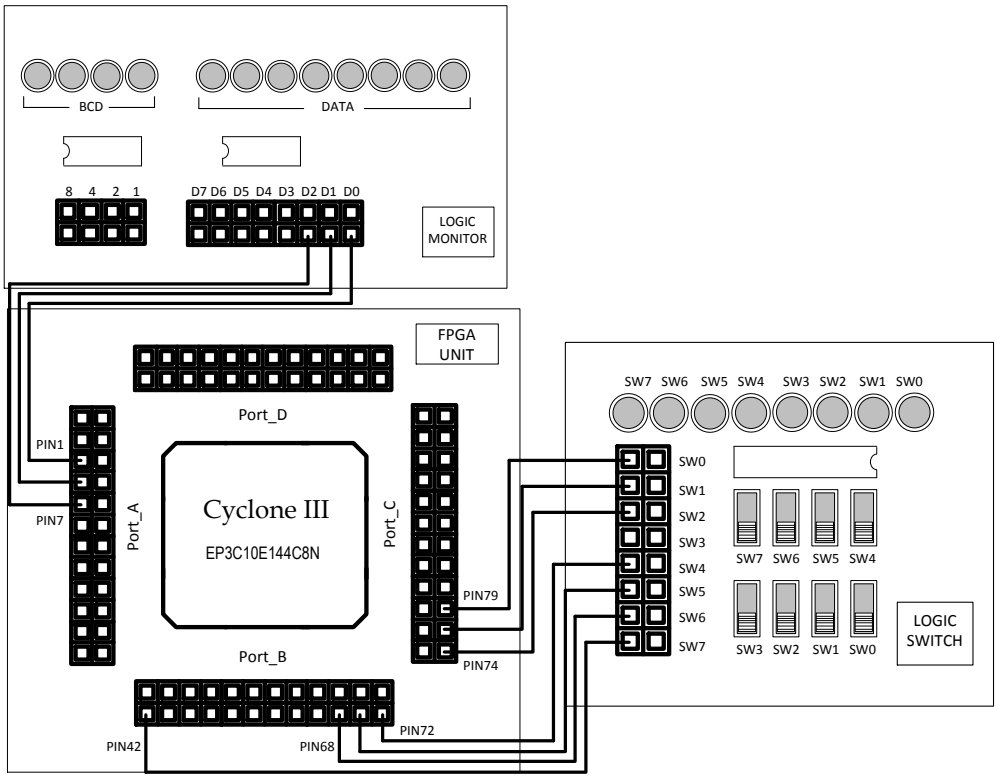
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	**ตำแหน่งขาบน FPGA**
CONTROL	42
A0	72
A1	70
A2	68
B0	79
B1	76
B2	74
OUT0	1
OUT1	3
OUT2	7

ตารางที่ 2.4 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 2.11 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 2.11 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1, SW2, SW3, SW4, SW5, SW6 และ SW7 แล้วดูผลทางลอจิก ที่ LED_D0, LED_D1 และ LED_D2 จากนั้นบันทึกผลลงในตาราง

SW7 (Control)	SW6 (A2)	SW5 (A1)	SW4 (A0)	SW2 (B2)	SW1 (B1)	SW0 (B0)	LED_D2 (OUT2)	LED_D1 (OUT1)	LED_D0 (OUT0)
0	0	0	0	0	1	1			
0	0	0	1	1	0	0			
0	0	1	0	1	0	1			
0	0	1	1	1	1	0			
0	1	0	0	1	1	1			
1	1	0	1	0	0	0			
1	1	1	0	0	0	1			
1	1	1	1	0	1	0			
1	0	0	0	1	0	0			
1	0	0	1	1	0	1			

ตารางที่ 2.5 ตารางความจริงจากการทดลองตอนที่ 2

สรุปผลการทดลอง

การทดลองที่ 3

วงจรเข้ารหัส (Encoder)

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรเข้ารหัส
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรเข้ารหัส
4. สามารถดาวน์โหลดวงจรลงไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

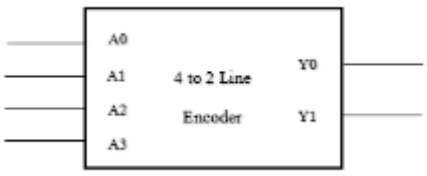
อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

ทฤษฎี

วงจรเข้ารหัส (Encoder) คือ วงจรที่ใช้สำหรับแปลงข้อมูลตัวเลขต่าง ๆ เช่น ข้อมูลปมที่ถูกกดบนคีย์บอร์ด ให้อยู่ในรูปแบบของรหัสเลขฐานสองซึ่งแทนด้วยระดับลอจิก 1 และ 0 ข้อสังเกตอย่างหนึ่งของวงจรเข้ารหัสคือ มันจะมีอินพุทที่แอคทีฟในเวลาหนึ่ง ๆ ได้เพียงอินพุทเดียวเท่านั้น จำนวนรูปแบบสูงสุดของอินพุทที่เป็นไปได้นั้นจะขึ้นอยู่กับจำนวนบิตทางด้านเอาต์พุท โดยจะมีค่าเป็น 2^n บิต เมื่อ n คือ จำนวนบิตของเอาต์พุท

สำหรับตัวอย่างและตารางความจริงของวงจรเข้ารหัสแบบ 4 สาย เป็น 2 สาย (4 to 2 line Encoder) แสดงดังรูปที่ 3.1 และตารางที่ 3.1 ตามลำดับ



รูปที่ 3.1 วงจรเข้ารหัสแบบ 4 สายเป็น 2 สาย

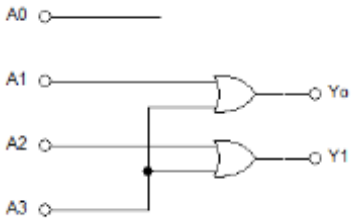
อินพุท				เอาต์พุท	
A3	A2	A1	A0	Y1	Y0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

ตารางที่ 3.1 ตารางความจริงของวงจรเข้ารหัสแบบ 4 สายเป็น 2 สาย

จากตารางความจริงของวงจรเข้ารหัสแบบ 4 สาย เป็น 2 สาย เราสามารถนำมาเขียนเป็นสมการบูลีน และวาดเป็นวงจรลอจิกได้ดังนี้

$$Y_0 = A_1 + A_3$$

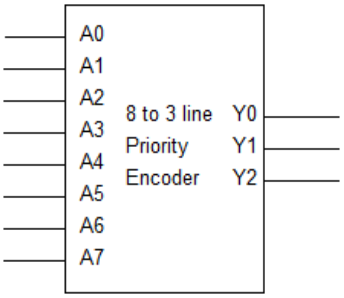
$$Y_1 = A_1 + A_3$$



รูปที่ 3.2 วงจรเข้ารหัสแบบ 4 สายเป็น 2 สาย

รูปที่ 3.3 วงจรเข้ารหัสแบบ 4 สายเป็น 2 สาย

ข้อเสียของวงจรเข้ารหัสแบบนี้คือ ในขนาดที่มีอินพุทแอคทีฟขึ้นมามากกว่า 1 อินพุท เช่น การกดสวิตช์ พร้อมกัน 2 ตัว จะทำให้การเข้ารหัสเกิดความผิดพลาดได้ ดังนั้นเราจึงต้องมีการให้ความสำคัญแก่ลำดับของอินพุทด้วย ซึ่งวงจรเข้ารหัสแบบนี้เราเรียกว่า วงจรเข้ารหัสแบบให้ความลำดับสำคัญ (Priority Encoder) ซึ่งจะมีการกำหนดลำดับความสำคัญให้แก่อินพุทแต่ละตัวมีค่าต่างกัน โดยอินพุทที่มีลำดับความสำคัญสูงกว่าจะถูกเข้ารหัสออกมาเพียงตัว ในกรณีที่มีอินพุทเข้ามา 2 ตัวพร้อมกันวงจรจะทำการเข้ารหัสเฉพาะอินพุทที่มีความสำคัญสูงกว่าเท่านั้น สำหรับตารางความจริงของวงจรเข้ารหัสแบบให้ความสำคัญสามารถแสดงดังตารางที่ 3.2 ซึ่งตัวอย่างของไอซีที่ทำหน้าที่เป็นวงจรเข้ารหัสแบบให้ลำดับความสำคัญได้แก่ ไอซีเบอร์ 74148

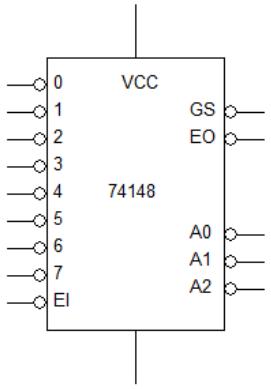


อินพุท								เอาต์พุท		
A7	A6	A5	A4	A3	A2	A1	A0	Y2	Y1	Y0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	x	0	0	1
0	0	0	0	0	1	x	x	0	1	0
0	0	0	0	1	x	x	x	0	1	1
0	0	0	1	x	x	x	x	1	0	0
0	0	1	x	x	x	x	x	1	0	1
0	1	x	x	x	x	x	x	1	1	0
1	x	x	x	x	x	x	x	1	1	1

รูปที่ 3.4 วงจรเข้ารหัสแบบ 8 สายเป็น 3 สาย

ตารางที่ 3.2 ตารางความจริงของวงจรเข้ารหัสแบบ 8 สายเป็น 3 สาย

ไอซี 74148 มีสัญลักษณ์และตารางความจริงดังรูปที่ 3.5 และ ตารางที่ 3.3 โดยด้านอินพุตและเอาต์พุทเป็นชนิดแอคทีฟโลว์ (Active Low) ซึ่งมีขาอินพุท 9 ขา โดยแบ่งเป็นขาอินพุทสำหรับเข้ารหัส 8 ขา (IO-I7) และขาสำหรับเปิดสัญญาณอินพุท (EI) ส่วนทางด้านเอาต์พุทจะมี 5 ขา โดยใช้เป็นขาหรับรหัส ไบนารีที่ผ่านการเข้ารหัสแล้ว 3 ขา (A0-A2) ขา GS สำหรับใช้บ่งบอกว่ามีอินพุทแอคทีฟ และขา EO สำหรับเปิดไอซี 74148 ตัวถัดไปในกรณีที่ใช้งานเป็นระบบเข้ารหัสที่มากกว่า 8 บิต

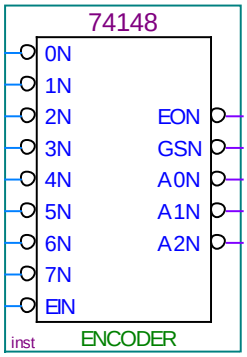


รูปที่ 3.5 ไอซี 74148

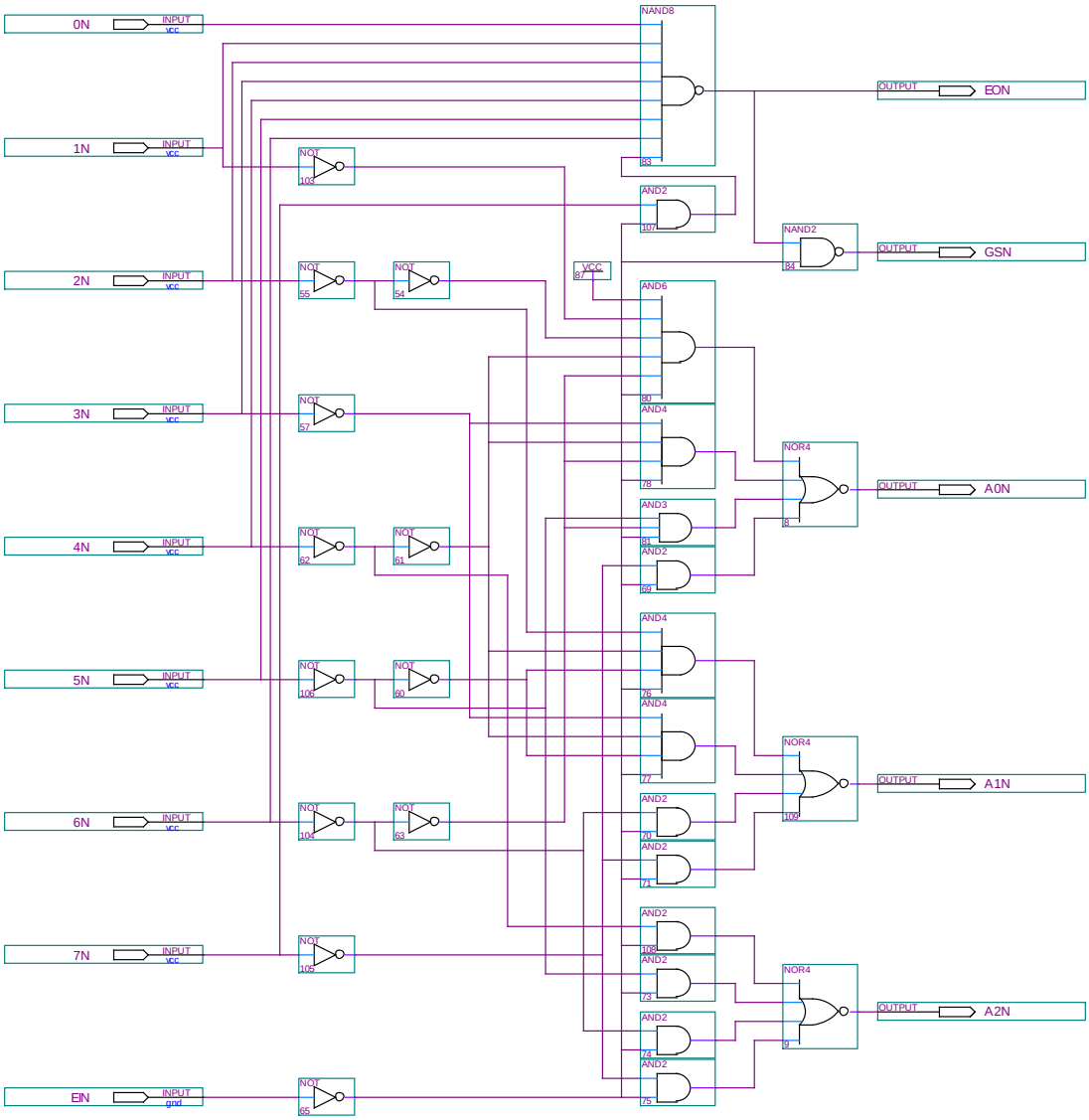
อินพุท									เอาต์พุท				
EI	0	1	2	3	4	5	6	7	A2	A1	A0	GS	EO
1	X	X	X	X	X	X	X	X	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	X	X	X	X	X	X	X	0	0	0	0	0	1
0	X	X	X	X	X	X	0	1	0	0	1	0	1
0	X	X	X	X	X	0	1	1	0	1	0	0	1
0	X	X	X	X	0	1	1	1	0	1	1	0	1
0	X	X	X	0	1	1	1	1	1	0	0	0	1
0	X	X	0	1	1	1	1	1	1	0	1	0	1
0	X	0	1	1	1	1	1	1	1	1	0	0	1
0	0	1	1	1	1	1	1	1	1	1	0	0	1

ตารางที่ 3.3 ตารางความจริงของไอซี 74148

สำหรับการทดลองนี้จะมีการนำไลบรารี่ของไอซี 74148 ซึ่งมีอยู่ในโปรแกรม Quartus II 8.1 Web Edition มาใช้งาน โดยไลบรารี่นี้จะเก็บอยู่ที่ C:/altera/81/quartus/libraries -> Others -> 74148 ซึ่งเมื่อวาด 74148 ลงในวงจรแล้วเราสามารถดับเบิ้ลคลิกที่สัญลักษณ์ของ 74148 จะมีสัญลักษณ์ วงกลมทั้งขาอินพุทและเอาต์พุทของ 74148 เป็นแบบ Active Low หรือก็คือ 74148 เป็นอุปกรณ์ที่ทำงานในสภาวะลอจิกต่ำนั่นเอง



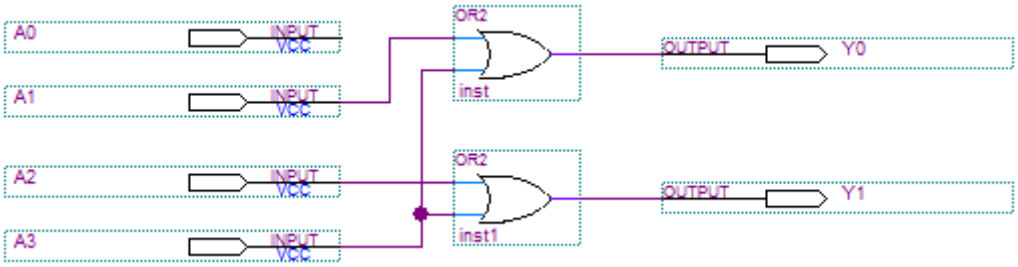
รูปที่ 3.6a สัญลักษณ์ของ 74148 ใน Quartus II 8.1 Web Edition



รูปที่ 3.6b วงจรภายใน 74148 เมื่อทำการดับเบิลคลิกบนตัวสัญลักษณ์ 74148

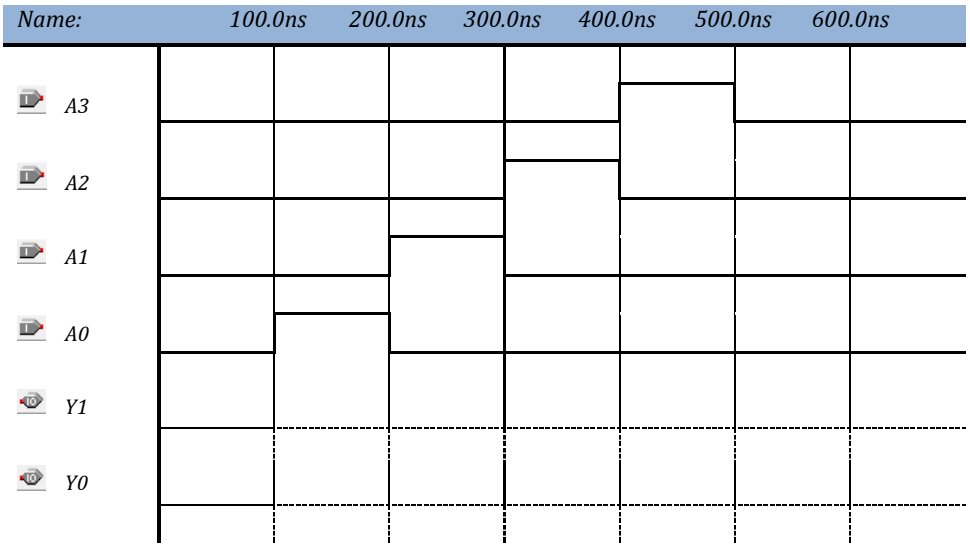
ขั้นตอนการทดลอง ตอนที่ 1 วงจรเข้ารหัส 4 สายเป็น 2 สาย

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_3a โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ or2, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป



รูปที่ 3.7 วงจรสำหรับการทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_3a.gdf
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_3a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 3.8 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

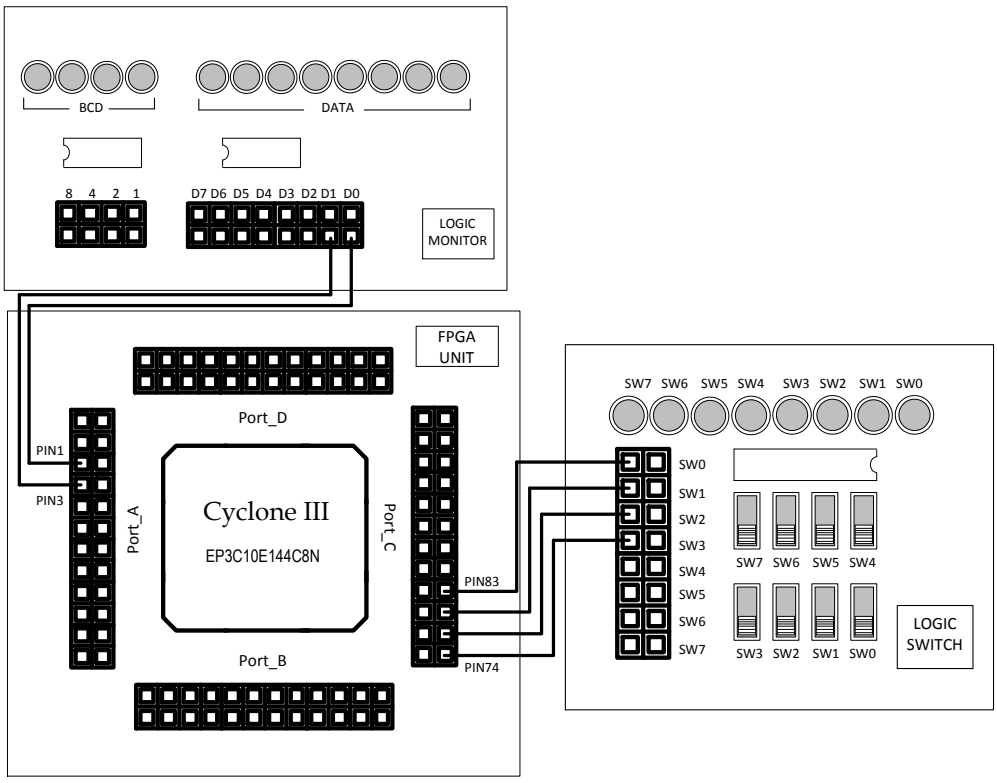
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A0	83
A1	79
A2	76
A3	74
Y0	1
Y1	3

ตารางที่ 3.4 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 3.9 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 3.9 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1, SW2 และ SW3 แล้วดูผลทางลอจิก ที่ LED_D0 และ LED_D1 จากนั้นบันทึกผลลงในตาราง

A3 (SW3)	A2 (SW2)	A1 (SW1)	A0 (SW0)	LED_D1 (Y1)	LED_D0 (Y0)
LOW	LOW	LOW	HIGH		
LOW	LOW	HIGH	LOW		
LOW	HIGH	LOW	LOW		
HIGH	LOW	LOW	LOW		

ตารางที่ 3.5 ตารางความจริงจากการทดลองที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 การใช้ไอซีวงจรเข้ารหัส 74148

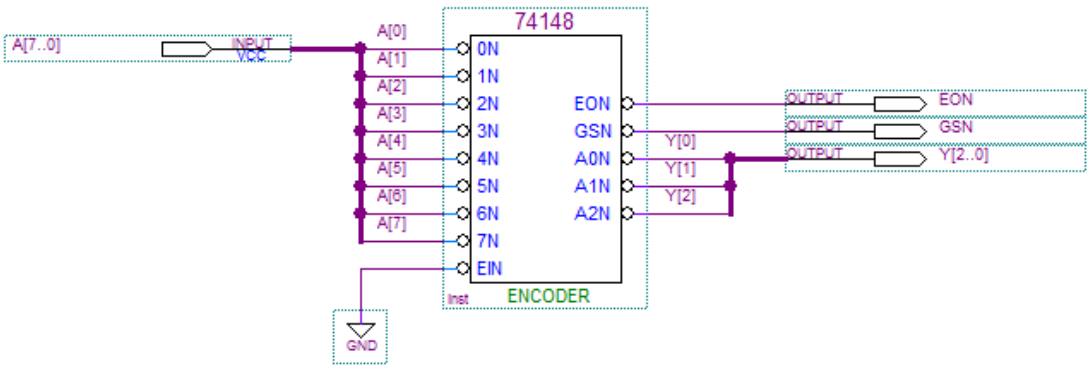
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_3b โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 74148, gnd, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

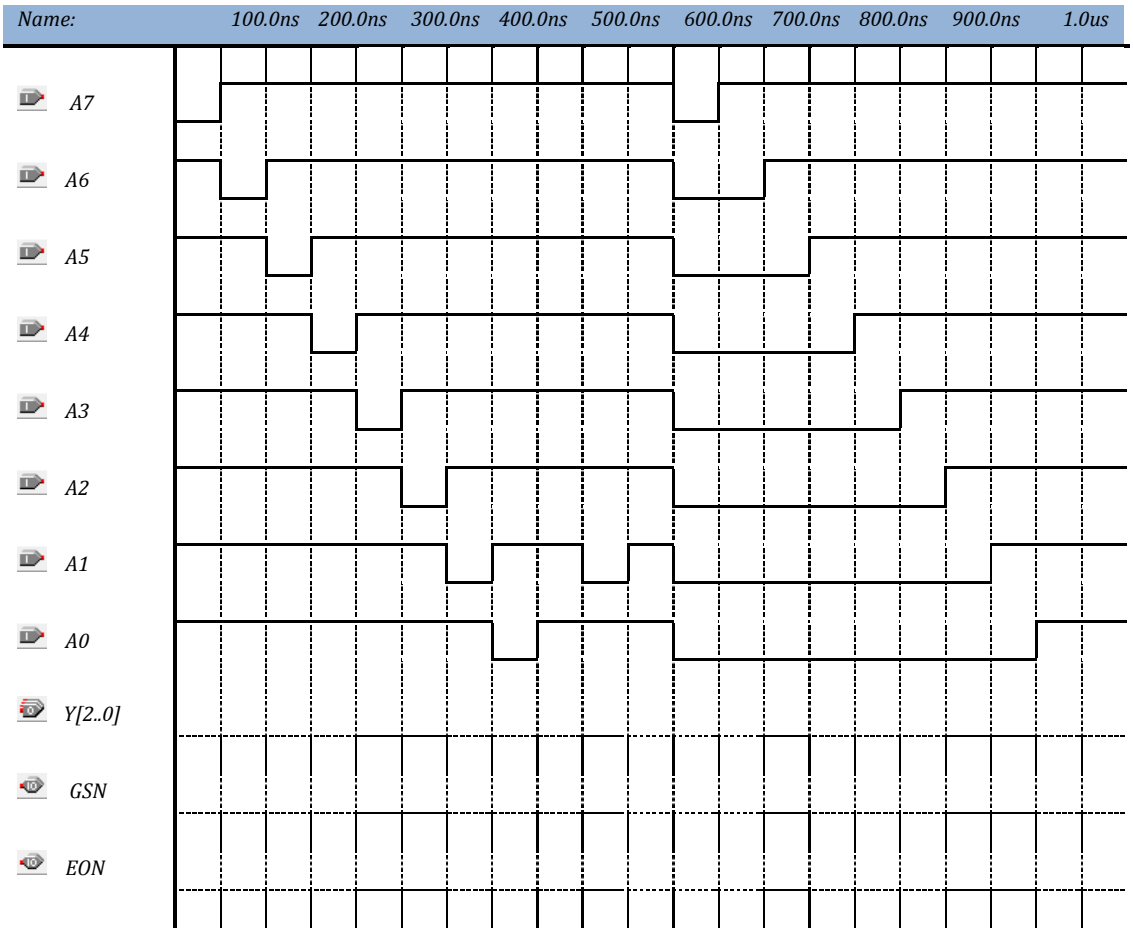


รูปที่ 3.10 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_3b.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_3b.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 3.11 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

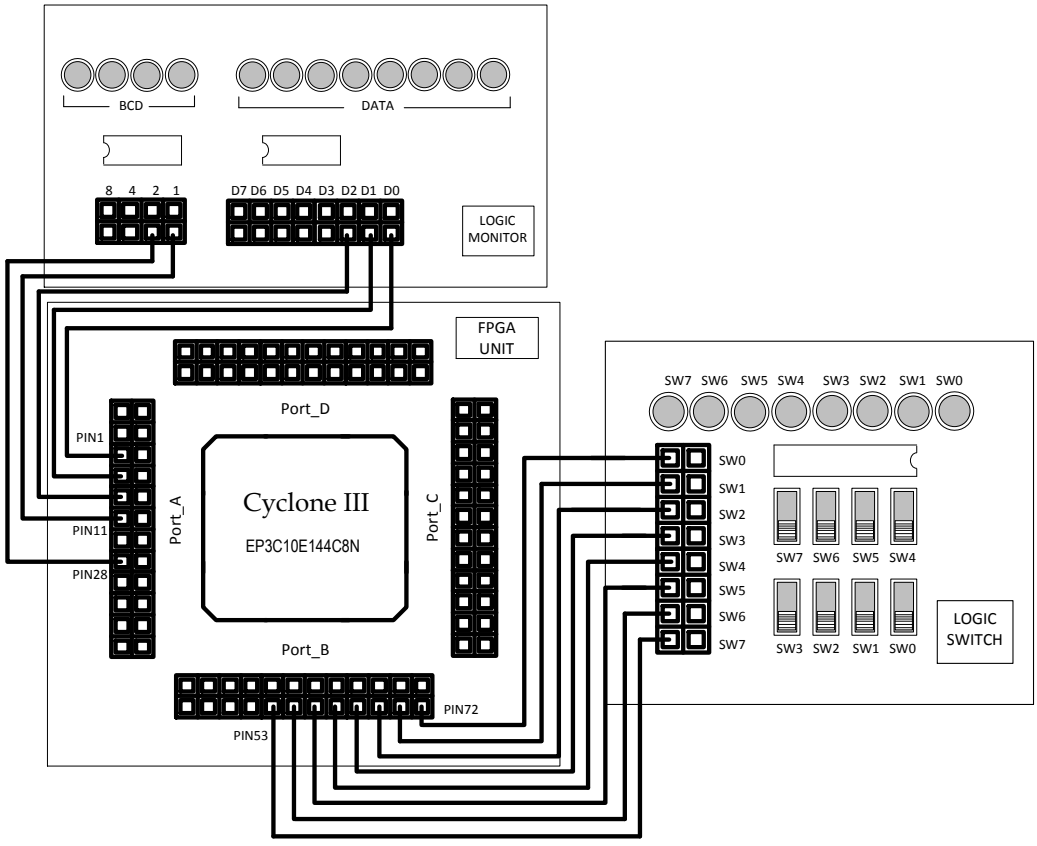
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
A0	53	A7	72
A1	55	Y0	1
A2	59	Y1	3
A3	64	Y2	7
A4	66	EON	11
A5	68	GSN	28
A6	70		

ตารางที่ 3.6 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 3.12 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 3.12 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1, SW2, SW3, SW4, SW5, SW6 และ SW7 และดูผลทางลอจิกที่ LED_D0, LED_D1, LED_D2, LED_1 และ LED_2

A0 (SW7)	A1 (SW6)	A2 (SW5)	A3 (SW4)	A4 (SW3)	A5 (SW2)	A6 (SW1)	A7 (SW0)	EON (LED_2)	GSN (LED_1)	Y2 (LED_D2)	Y1 (LED_D1)	Y0 (LED_D0)
1	1	1	1	1	1	1	0					
1	1	1	1	1	1	0	1					
1	1	1	1	1	0	1	1					
1	1	1	1	0	1	1	1					
1	1	1	0	1	1	1	1					
1	1	0	1	1	1	1	1					
1	0	1	1	1	1	1	1					
0	1	1	1	1	1	1	1					
1	0	1	0	1	1	1	1					
1	1	1	1	0	0	1	0					
1	1	1	1	1	1	1	1					

ตารางที่ 3.7 ตารางความจริงจากการทดลองที่ 2

สรุปผลการทดลอง

การทดลองที่ 4

วงจรถอดรหัส (Decoder)

วัตถุประสงค์

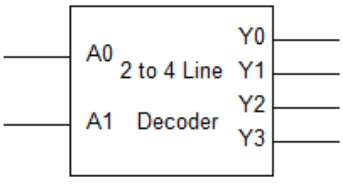
1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรถอดรหัส
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรถอดรหัสได้
4. สามารถดาวน์โหลดวงจรลงไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

ทฤษฎี

วงจรถอดรหัส (Decoder) คือ วงจรที่ใช้แปลงรหัสเลขฐานสองให้กลายเป็นรหัสตัวเลขต่าง ๆ เช่นตำแหน่งคีย์ของสวิทช์ ซึ่งเราอาจมองได้ว่าหน้าที่การทำงานของวงจรถอดรหัสจะมีลักษณะตรองกันข้ามกับวงจรเข้ารหัสนั่นเอง จำนวนรูปแบบสูงสุดของเอาต์พุทที่เป็นไปได้นั้นจะขึ้นอยู่กับจำนวนบิตทางด้านอินพุทโดยจะมีค่าเป็น 2^n บิต เมื่อ n คือจำนวนบิตของอินพุท



รูปที่ 4.1 วงจรถอดรหัสแบบ 2 สายเป็น 4 สาย

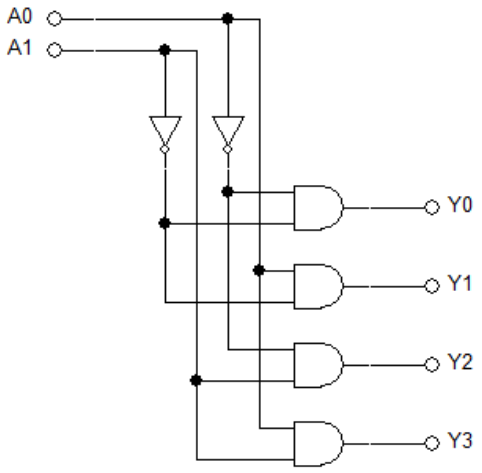
อินพุท			เอาต์พุท		
A1	A2	Y3	Y2	Y1	Y0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

ตารางที่ 4.1 ตารางความจริงของวงจรถอดรหัสแบบ 2 สายเป็น 4 สาย

จากตารางความจริงของวงจรถอดรหัสแบบ 2 สายเป็น 4 สาย เราสามารถนำมาเขียนเป็นสมการบูลีน และวาดเป็นวงจรลอจิกได้ดังนี้

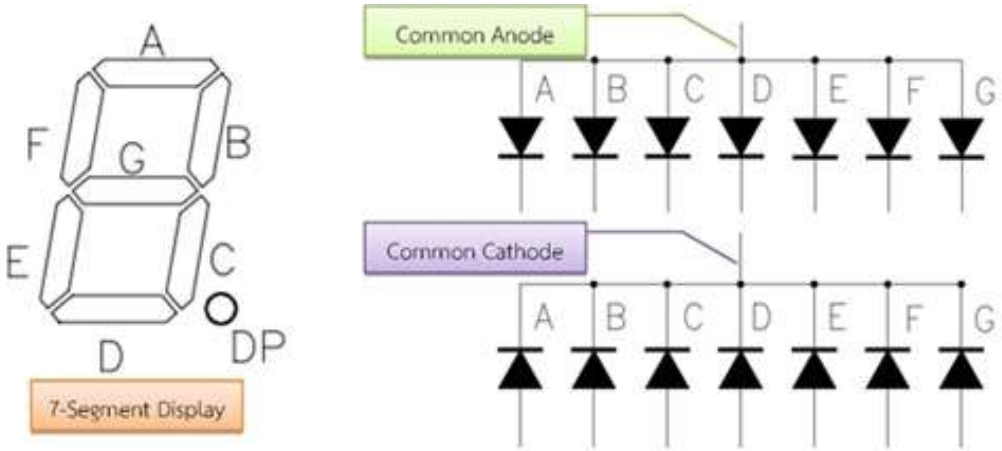
$$Y_0 = \overline{A_1}\,\overline{A_0}$$
$$Y_1 = \overline{A_1}\,A_0$$
$$Y_2 = A_1\,\overline{A_0}$$
$$Y_3 = A_1 A_0$$

รูปที่ 4.2 สมการบูลีนของวงจรถอดรหัสแบบ 2 สายเป็น 4 สาย



รูปที่ 4.3 วงจรถอดรหัสแบบ 2 สายเป็น 4 สาย

ตัวอย่างของวงจรถอดรหัสที่มีการใช้กันโดยทั่วไปได้แก่วงจรถอดรหัสเลขฐานสองให้เป็นตัวเลข 7 ส่วน (7-Segment) ซึ่งตัวเลข 7 ส่วนคืออุปกรที่มีการนำเอา LED 7 ตัวมาจัดเรียงเพื่อใช้แสดงเป็นตัวเลข หรือในบางครั้งใช้แทนตัวหนังสือบางตัว ในส่วนของ LED ที่นำมาต่อกันแต่ละส่วนหรือบางครั้งเรียกว่าเซกเมนต์ (Segment) จะมีชื่อเรียงตามลำดับตัวอักษร ตัวเลขเจ็ดส่วนที่มีใช้ในปัจจุบันแบ่งเป็น 2 ชนิดตามโครงสร้าง ได้แก่ คอมมอนแอโนด และคอมมอนแคโถด ซึ่งมีวงจรภายในดังรูปที่ 4.4

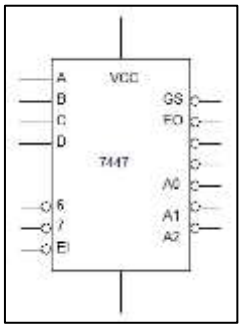


รูปที่ 4.4 ตัวอย่างการวางตัวเลข 7 ส่วน

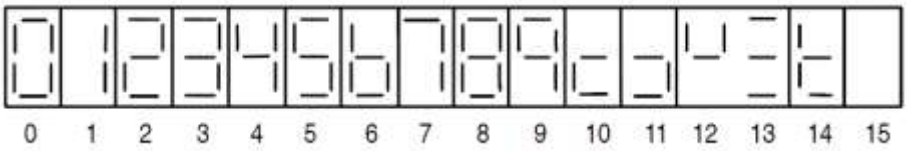
ตัวอย่างของไอซีที่ทำหน้าที่เป็นวงจรถอดรหัสได้แก่ไอซีเบอร์ 7447 ซึ่งทำหน้าที่เป็นตัวถอดรหัสเลขฐานสองให้เป็นตัวเลข 7 ส่วน (7-Segment) แบบคอมมอนแอโนด ในส่วนของการทำงานของไอซีเบอร์ 7447 นั้นจะเป็นดังตารางที่ ซึ่งท่านสามารถศึกษาการใช้งานโดยละเอียดได้จาก Datasheet ของผู้ผลิตอีกครั้ง

อินพุท								**เอาต์พุท**						
Decimal Or Function	$\overline{\textbf{LT}}$	$\overline{\textbf{RBI}}$	**D**	**C**	**B**	**A**	$\overline{\textbf{BI}}$ / $\overline{\textbf{RBO}}$	$\overline{\textbf{a}}$	$\overline{\textbf{b}}$	$\overline{\textbf{c}}$	$\overline{\textbf{d}}$	$\overline{\textbf{e}}$	$\overline{\textbf{f}}$	$\overline{\textbf{g}}$
0	H	H	L	L	L	L	H	L	L	L	L	L	L	H
1	H	X	L	L	L	H	H	H	L	L	H	H	H	H
2	H	X	L	L	H	L	H	L	L	H	L	L	H	L
3	H	X	L	L	H	H	H	L	L	L	L	H	H	L
4	H	X	L	H	L	L	H	H	L	L	H	H	L	L
5	H	X	L	H	L	H	H	L	H	L	L	H	L	L
6	H	X	L	H	H	L	H	H	H	L	L	L	L	L
7	H	X	L	H	H	H	H	L	L	L	H	H	H	H
8	H	X	H	L	L	L	H	L	L	L	L	L	L	L
9	H	X	H	L	L	H	H	L	L	L	H	H	L	L
10	H	X	H	L	H	L	H	H	H	H	L	L	H	L
11	H	X	H	L	H	H	H	H	H	L	L	H	H	L
12	H	X	H	H	L	L	H	H	L	H	H	H	L	L
13	H	X	H	H	L	H	H	L	H	H	L	H	L	L
14	H	X	H	H	H	L	H	H	H	H	L	L	L	L
15	H	X	H	H	H	H	H	H	H	H	H	H	H	H
$\overline{\text{BI}}$	X	X	X	X	X	X	L	H	H	H	H	H	H	H
$\overline{\text{RBI}}$	H	L	L	L	L	L	L	H	H	H	H	H	H	H
$\overline{\text{LT}}$	L	H	X	X	X	X	H	L	L	L	L	L	L	L

ตารางที่ 4.2 ตารางความจริงของไอซี 7447



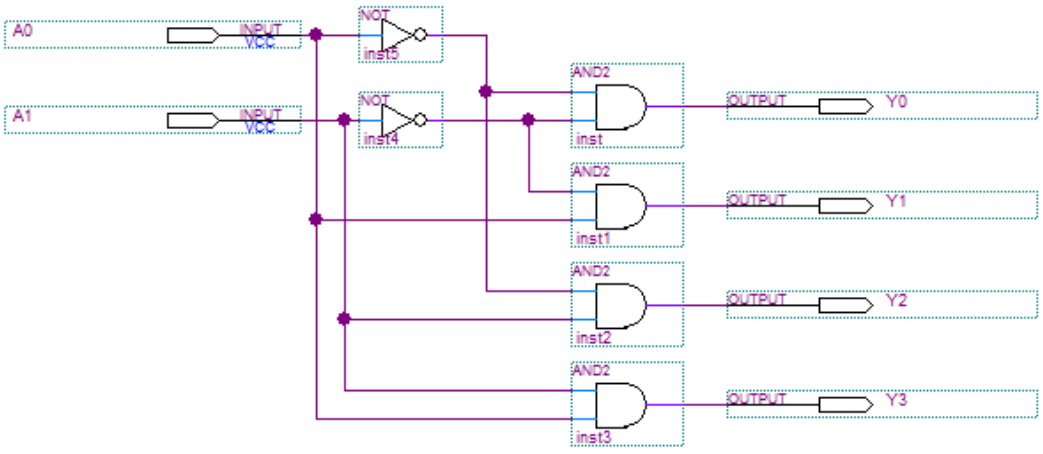
รูปที่ 4.5 ไอซี 7447



รูปที่ 4.6 รูปแบบการถอดรหัสของไอซี 7447

ขั้นตอนการทดลอง ตอนที่ 1 วงจรถอดรหัส 2 สายเป็น 4 สาย

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_4a โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ and2, not, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

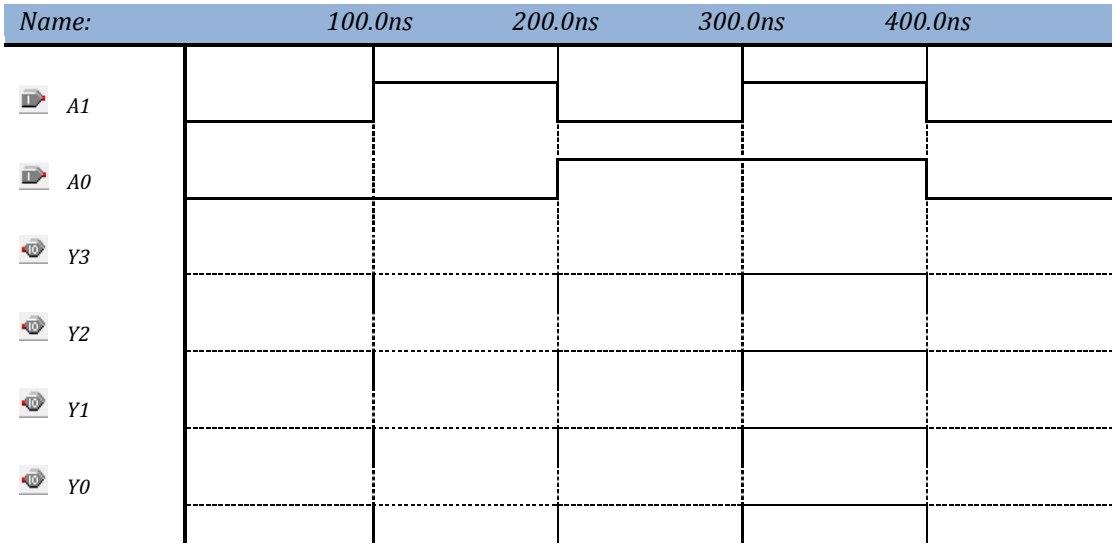


รูปที่ 4.7 วงจรสำหรับการทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_4a.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_4a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 4.8 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

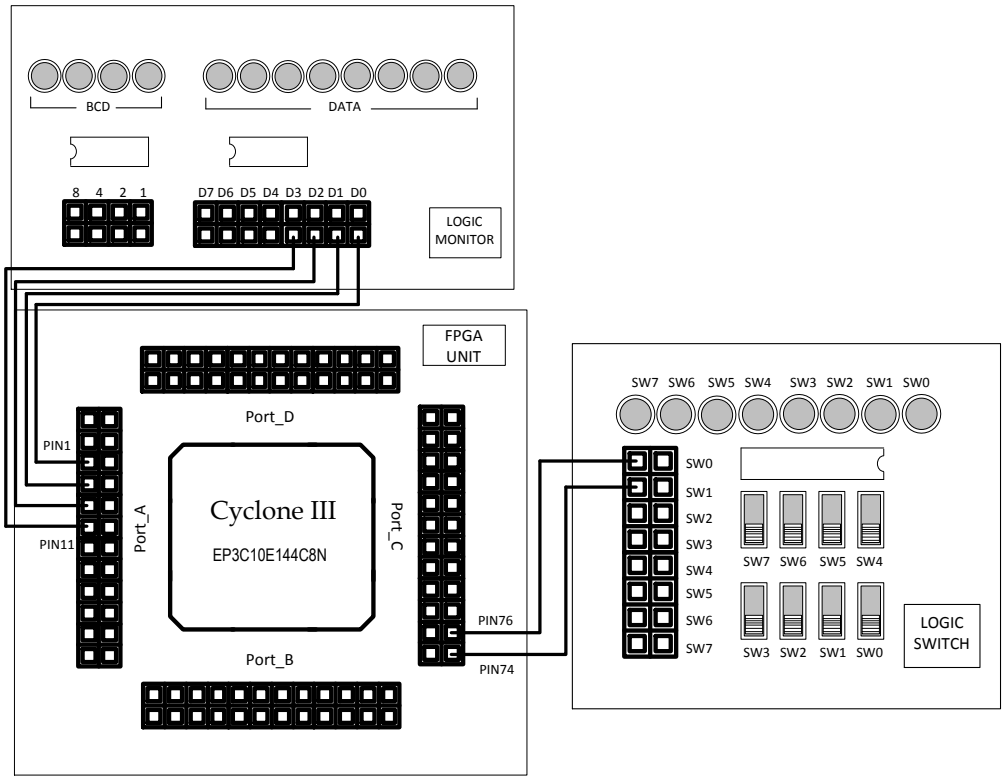
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A0	76
A1	74
Y0	1
Y1	3
Y2	7
Y3	11

ตารางที่ 4.3 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 4.9 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 4.9 การต่อวงจรบนชุดทดลอง

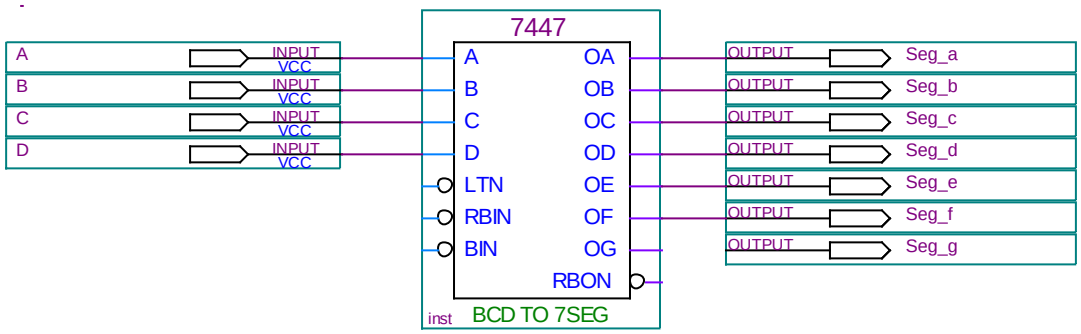
13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิกที่ LED_D0, LED_D1, LED_D2 และ LED_D3 จากนั้นบันทึกผลลงในตาราง

A1 (SW1)	**A0 (SW0)**	**Y3 (LED_D3)**	**Y2 (LED_D2)**	**Y1 (LED_D1)**	**Y0 (LED_D0)**
LOW	LOW				
LOW	HIGH				
HIGH	LOW				
HIGH	HIGH				

ตารางที่ 4.4 ตารางความจริงจากการทดลองที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 การใช้ไอซีวงจรถอดรหัส 7447

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_4b โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 7447, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป



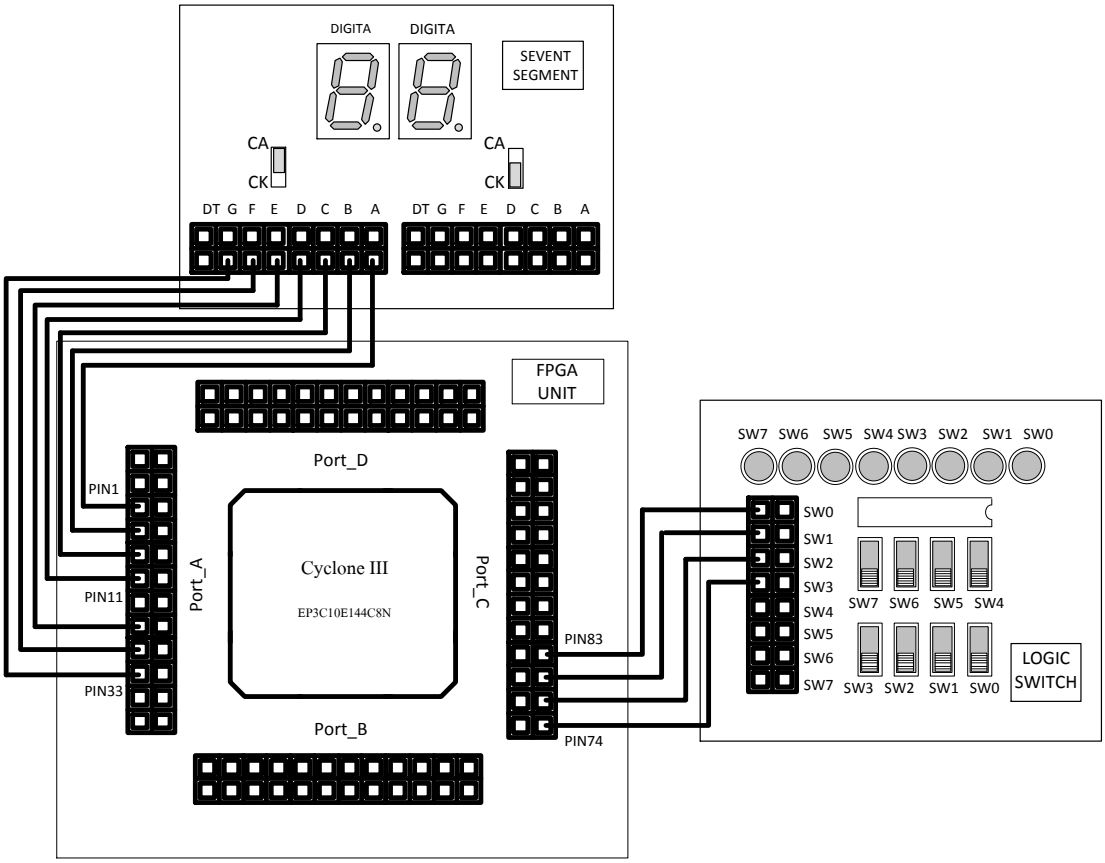
รูปที่ 4.10 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_4b.gdf
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation
8. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins
9. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
A	83	Seg_c	7
B	79	Seg_d	11
C	76	Seg_e	28
D	74	Seg_f	31
Seg_a	1	Seg_g	33
Seg_b	3		

ตารางที่ 4.5 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

10. ต่อวงจรตามรูปที่ 4.11 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 4.11 การต่อวงจรบนชุดทดลอง

11. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1, SW2 และ SW3 แล้วดูผลทางลอจิก ที่ SEVEN SEGMENT DIGIT จากนั้นบันทึกผลลงในตาราง

D (SW3)	C (SW2)	B (SW1)	A (SW0)	รูปแบบ LED บน (SEVEN SEGMENT)		D (SW3)	C (SW2)	B (SW1)	A (SW0)	รูปแบบ LED บน (SEVEN SEGMENT)
0	0	0	0			1	0	0	0	
0	0	0	1			1	0	0	1	
0	0	1	0			1	0	1	0	
0	0	1	1			1	0	1	1	
0	1	0	0			1	1	0	0	
0	1	0	1			1	1	0	1	
0	1	1	0			1	1	1	0	
0	1	1	1			1	1	1	1	

ตารางที่ 4.6 ตารางความจริงจากการทดลองที่ 2

สรุปผลการทดลอง

การทดลองที่ 5

วงจรมัลติเพล็กซ์ (Multiplexor)

วัตถุประสงค์

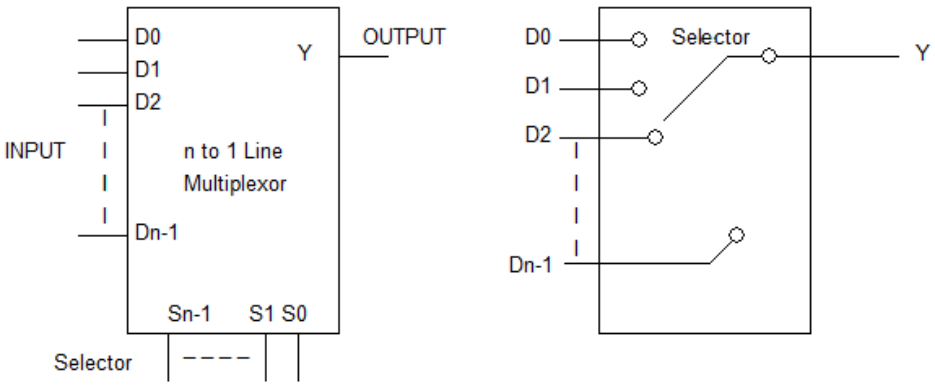
1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรมัลติเพล็ก
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรมัลติเพล็ก
4. สามารถดาวน์โหลดวงจรไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

ทฤษฎี

วงจรมัลติเพล็กซ์ (Multiplexor) เป็นวงจรที่ทำหน้าที่เป็นตัวเลือกข้อมูลดิจิตอล โดยลักษณะของวงจรจะมีชุดของอินพุทมากกว่า 1 ชุด แต่จะมีเอาต์พุทเพียงชุดเดียว ซึ่งการทำงานของวงจรมัลติเพล็กจะทำหน้าที่เป็นตัวเลือกชุดข้อมูลทางอินพุทให้ออกมาเพียงชุดเดียวเท่านั้น ซึ่งอาจเปรียบได้กับสวิทช์เลือก Selector Switch) นั่นเอง

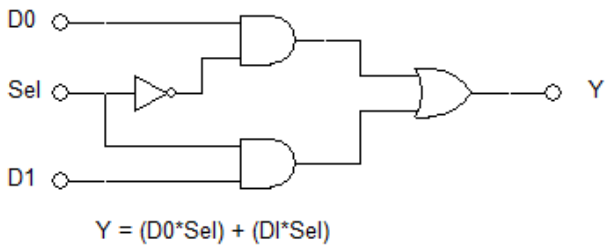


รูปที่ 5.1 วงจรมัลติเพล็กซ์

ในตารางที่ 5.1 และ รูปที่ 5.2 เป็นตารางความจริง รวมถึงสมการบูลีนและวงจรลอจิกของวงจรมัลติเพล็กซ์แบบ 2 สายเป็น 1 สาย

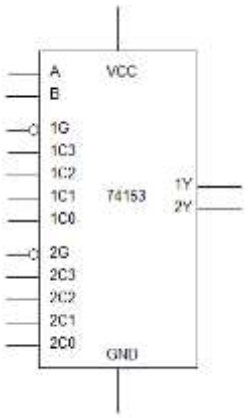
อินพุท			เอาต์พุท
Sel	D1	D0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

ตารางที่ 5.1 ตารางความจริงของวงจรมัลติเพล็กซ์แบบ 2 สายเป็น 1 สาย



รูปที่ 5.2 วงจรมัลติเพล็กซ์แบบ 2 สายเป็น 1 สาย

สำหรับตัวอย่างของไอซีที่ทำหน้าที่เป็นวงจรมัลติเพล็กซ์ได้แก่ ไอซีเบอร์ 74153 ซึ่งเป็นไอซีมัลติเพล็กซ์แบบ 4 สาย เป็น 1 สาย 2 ชุด โดยใช้ขาเลือกข้อมูลร่วมกัน และมีขาเปิดการทำงาน (Enable) เอาต์พุทแยกจากกันทั้ง 2 ชุด ซึ่งสามารถเลือกเปิดการทำงานเอาต์พุทชุดใดชุดหนึ่งโดยอิสระ



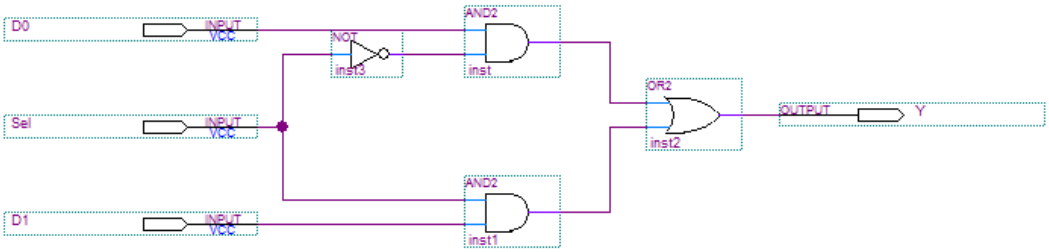
รูปที่ 5.3 ไอซี 74153

Select		Data					output
B	A	C0	C1	C2	C3	$\overline{G}$	
X	X	X	X	X	X	L	L
L	L	L	X	X	X	L	L
L	L	H	X	X	X	L	H
L	H	X	L	X	X	L	L
L	H	X	H	X	X	L	H
h	L	X	X	L	X	L	L
H	L	X	X	H	X	L	H
H	H	X	X	X	L	L	L
H	H	X	X	X	H	L	H

ตารางที่ 5.2 ตารางความจริงของไอซี 74153

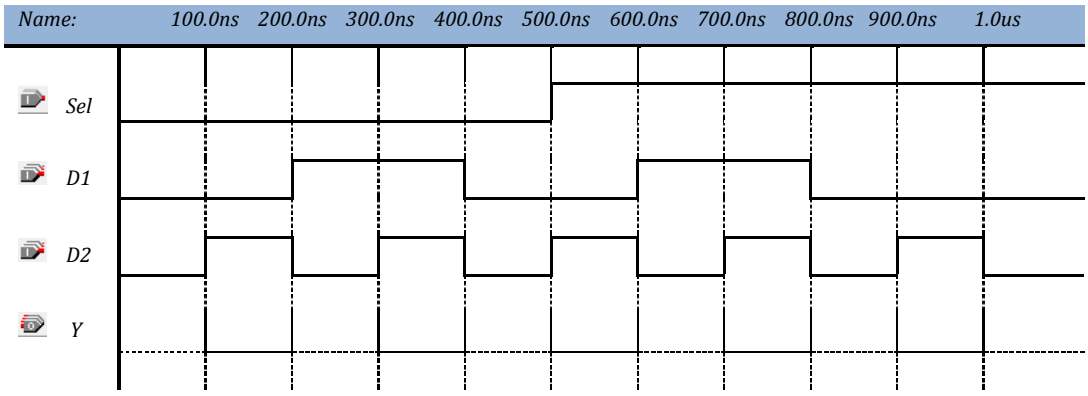
ขั้นตอนการทดลอง ตอนที่ 1 วงจรมัลติเพล็กซ์ 2 สายเป็น 1 สาย

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_5a โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor เลือกอุปกรณ์ and2, or2, not, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป



รูปที่ 5.4 วงจรสำหรับการทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_5a.gdf
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_5a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 5.5 ผลจำลองการทำงาน

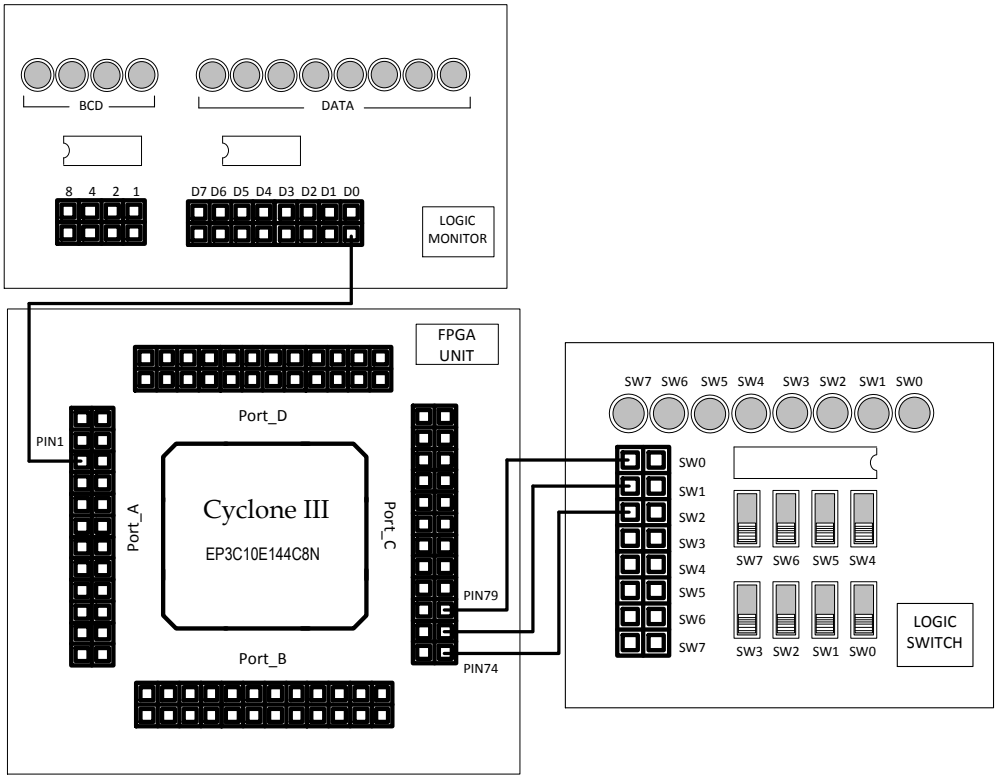
9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
D0	79
D1	76
Sel	74
Y	1

ตารางที่ 5.3 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 5.6 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 5.6 การต่อวงจรบนชุดทดลอง

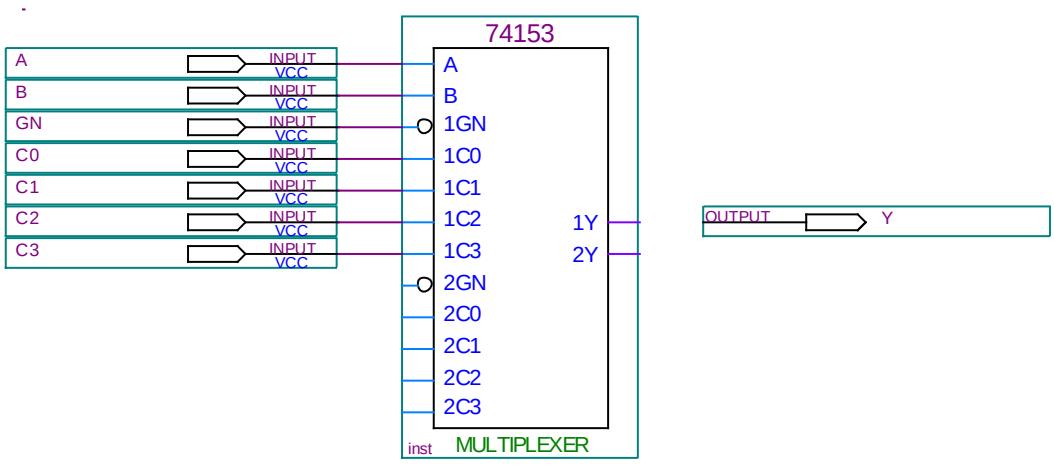
13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1 และ SW2 แล้วดูผลทางลอจิกที่ LED_D0 จากนั้นบันทึกผลลงในตาราง

Sel (SW2)	D1 (SW1)	D0 (SW0)	Y0 (LED_D0)
LOW	LOW	LOW	
LOW	LOW	HIGH	
LOW	HIGH	LOW	
LOW	HIGH	HIGH	
HIGH	LOW	LOW	
HIGH	LOW	HIGH	
HIGH	HIGH	LOW	
HIGH	HIGH	HIGH	

ตารางที่ 5.4 ตารางความจริงจากการทดลองที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 ไอซีวงจรมัลติเพล็กซ์ 74153

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_5b โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 74153, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

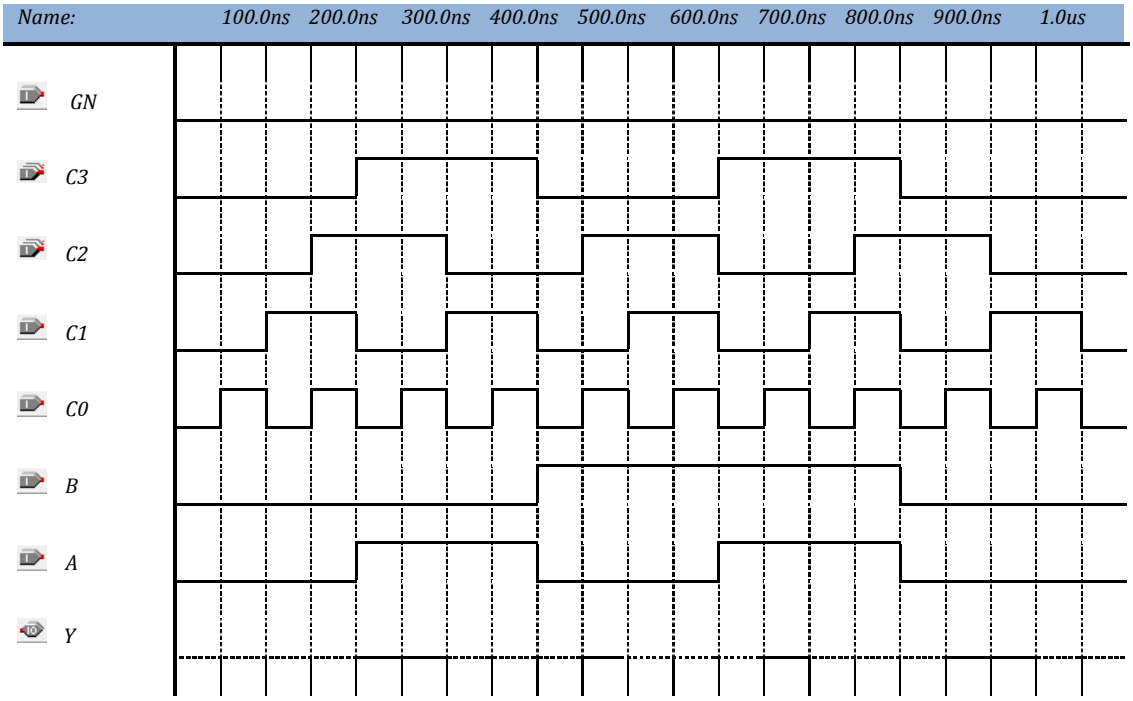


รูปที่ 5.7 วงจรสำหรับการทดลองที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_5b.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_5b.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 5.8 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

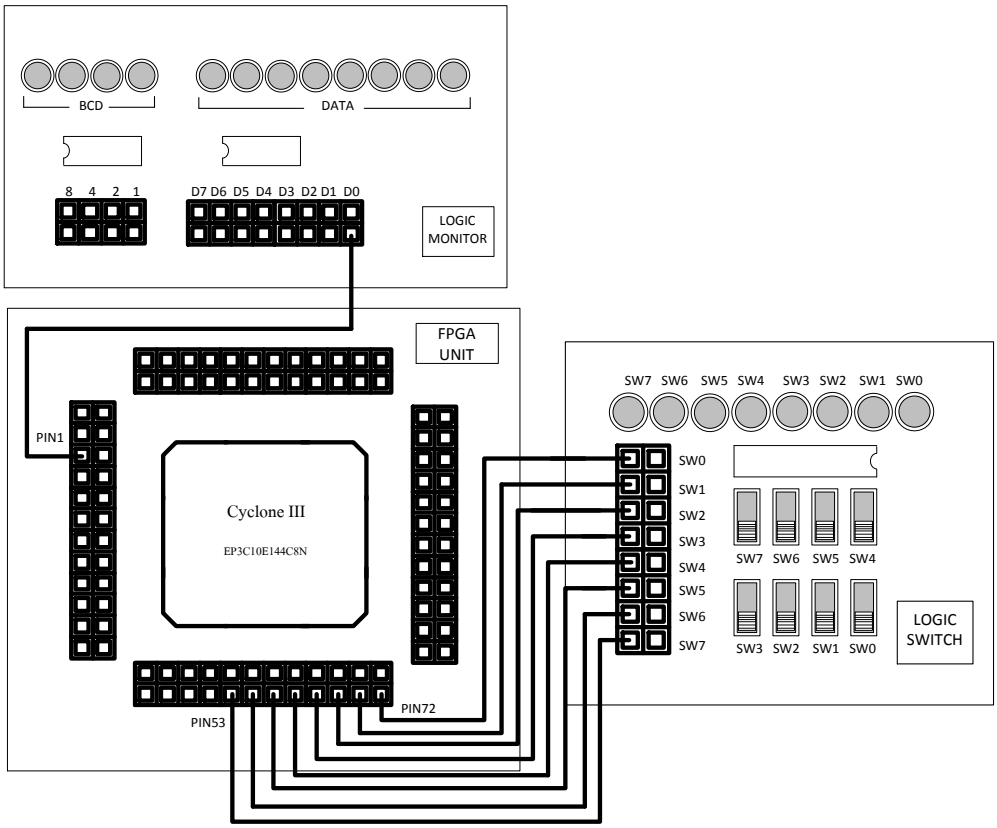
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
C0	72	A	64
C1	70	B	59
C2	68	GN	55
C3	66	Y	1

ตารางที่ 5.5 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 5.9 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 5.9 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1, SW2, SW3, SW4, SW5 และ SW6 แล้วดูผลทางลอจิกที่ LED_D0 จากนั้นบันทึกผลลงในตาราง

GN (SW6)	**B (SW5)**	**A (SW4)**	**C3 (SW3)**	**C2 (SW2)**	**C1 (SW1)**	**C0 (SW0)**	**Y0 (LED_D0)**
0	0	0	0	0	0	1	
0	0	0	0	0	1	0	
0	0	0	0	1	1	1	
0	0	0	1	1	0	0	
0	0	1	1	1	0	1	
0	0	1	1	0	1	0	
0	0	1	1	0	1	1	
0	0	1	0	0	0	0	
0	1	0	0	1	0	1	
0	1	0	0	1	1	0	
0	1	0	0	1	1	1	
0	1	0	1	0	0	0	
0	1	1	1	0	0	1	
0	1	1	1	0	1	0	
0	1	1	1	1	1	1	
0	1	1	0	1	0	0	

ตารางที่ 5.6 ตารางความจริงจากการทดลองที่ 2

สรุปผลการทดลอง

การทดลองที่ 6

วงจรดีมัลติเพล็กซ์ (Demultiplexor)

วัตถุประสงค์

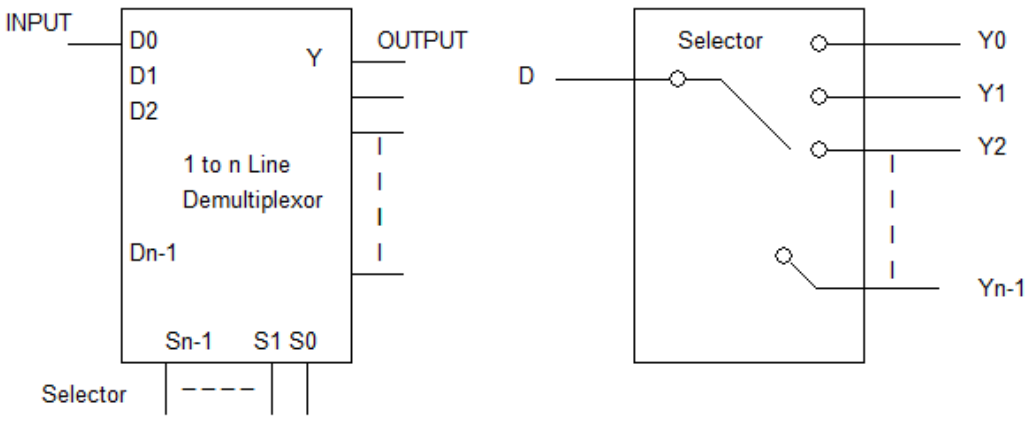
1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรดีมัลติเพล็ก
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรดีมัลติเพล็ก
4. สามารถดาวน์โหลดวงจรไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

ทฤษฎี

วงจรดีมัลติเพล็กซ์ (Demultiplexor) เป็นวงจรที่ทำหน้าที่กระจายข้อมูลจากอินพุทที่มีอยู่เพียง 1 ชุดให้ออกไปที่เอาต์พุทใดเอาต์พุทหนึ่งโดยอาศัยขาควบคุมในการกระจายข้อมูล ซึ่งการทำงานของดีมัลติเพล็กซ์จะมีลักษณะตรงกันข้ามกับวงจรมัลติเพล็กซ์นั่นเอง

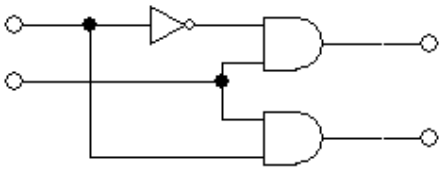


รูปที่ 6.1 วงจรดีมัลติเพล็กซ์

ในตารางที่ 6.1 และรูปที่ 6.2 เป็นตารางความจริง รวมถึงสมการบูลีนและวงจรลอจิกของวงจรดีมัลติเพล็กซ์แบบ 1 สายเป็น 2 สาย

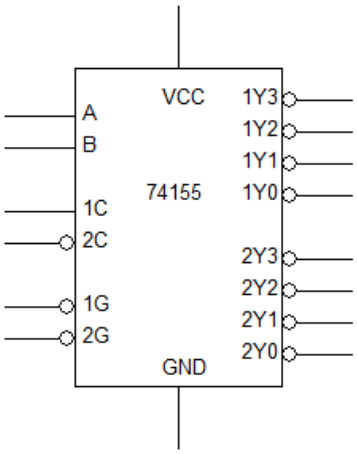
อินพุท		เอาต์พุท	
Sel	D	Y1	Y0
0	0	0	0
0	1	0	1
1	0	0	0
1	1	1	0

ตารางที่ 6.1 ตารางความจริงของวงจรดีมัลติเพล็กซ์แบบ 1 สายเป็น 2 สาย



รูปที่ 6.2 วงจรดีมัลติเพล็กซ์แบบ 1 สายเป็น 2 สาย

ตัวอย่างของไอซีที่ทำหน้าที่เป็นวงจรดีมัลติเพล็กซ์ได้แก่ ไอซีเบอร์ 74155 ซึ่งเป็นไอซีดีมัลติเพล็กซ์แบบ 1 สาย เป็น 4 สาย 2 ชุด โดยใช้ขาเลือกข้อมูลร่วมกัน และมีขาสโตรบ (Strobe) เปิดการทำงาน (Eanable) เอาต์พุทแยกจากกันทั้ง 2 สองชุด ซึ่งสามารถเลือกเปิดการทำงานเอาต์พุทชุดใดชุดหนึ่งโดยอิสระ ข้อมูลที่ถูกป้อนเข้ามาที่ขา 1C จะถูกกลับสัญญาณเอาต์พุท ส่วนสัญญาณที่เข้ามา 2C จะไม่มีการกลับสัญญาณ ซึ่งการกลับสัญญาณจากข้อมูล 1C จะทำให้เราสามารถนำไอซี 74155 ไปใช้เป็นตัวถอดรหัสแบบ 3 สายเป็น 8 สาย ได้ หรืออาจใช้เป็นวงจรมัลติเพล็กซ์แบบ 1 สายเป็น 8 สายได้ โดยไม่ต้องเพิ่มอุปกรณ์ภายนอก สำหรับการทดลองในบทนี้เราจะทดลองเฉพาะในส่วนของการทำหน้าที่เป็นวงจรดีมัลติเพล็กซ์แบบ 1 สาย เป็น 4 สายเท่านั้น



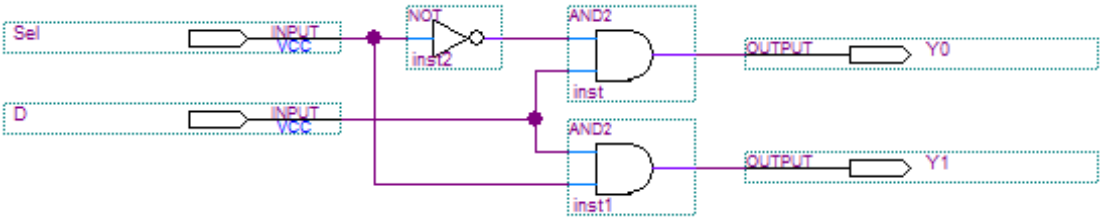
รูปที่ 6.3 ไอซี 74155

Select		Strobe	Data	Output			
B	A	1G	1C	1Y0	1Y1	1Y2	1Y3
X	X	H	X	H	H	H	H
L	L	L	H	L	H	H	H
L	H	L	H	H	L	H	H
H	L	L	H	H	H	L	H
H	H	L	H	H	H	H	L
X	X	X	L	H	H	H	H
Select		**Strobe**	**Data**	**Output**			
B	A	2G	2C	2Y0	2Y1	2Y2	2Y3
X	X	H	X	H	H	H	H
L	L	L	H	L	H	H	H
L	H	L	H	H	L	H	H
H	L	L	H	H	H	L	H
H	H	L	H	H	H	H	L
X	X	X	L	H	H	H	H

ตารางที่ 6.2 ตารางความจริงของไอซี 74155

ขั้นตอนการทดลอง ตอนที่ 1 วงจรดีมัลติเพล็กซ์ 1สายเป็น 2 สาย

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_6a โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor เลือกอุปกรณ์ and2, not, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

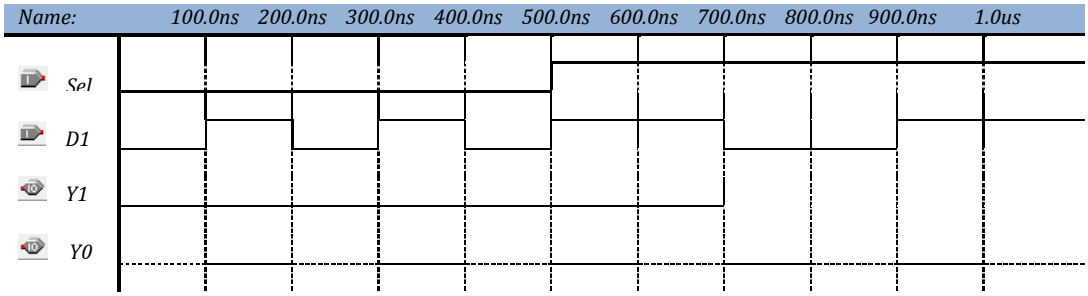


รูปที่ 6.4 วงจรสำหรับทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_6a.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_6a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 6.5 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

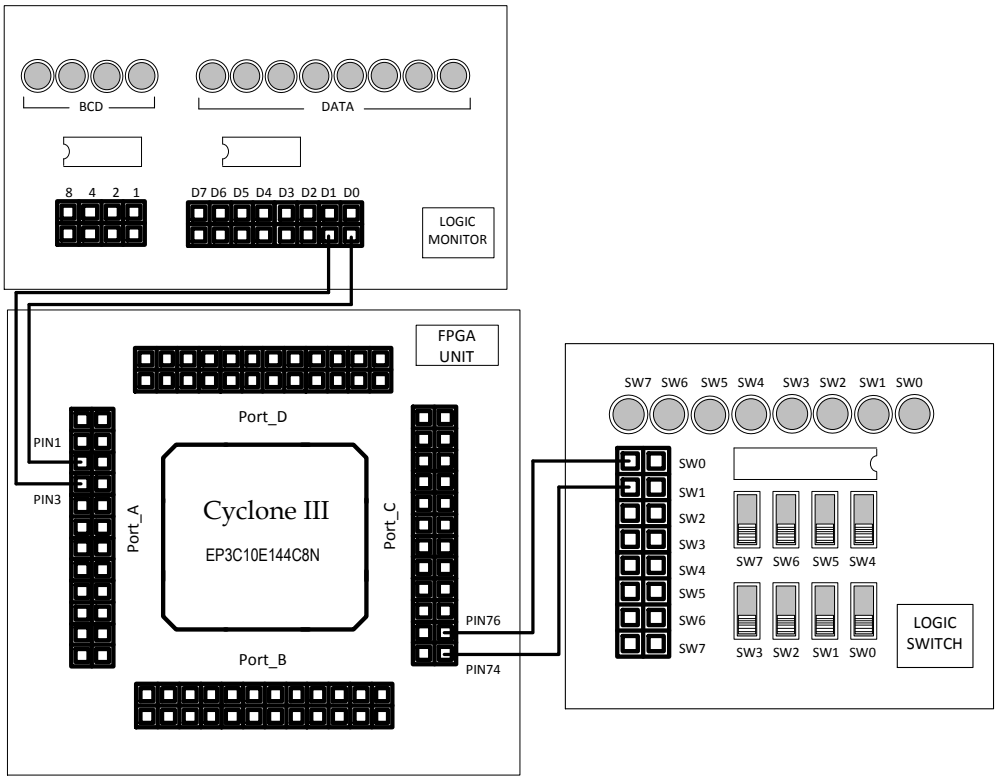
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
Sel	74
D	76
Y0	1
Y1	3

ตารางที่ 6.3 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 6.6 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 6.6 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1 และ SW2 แล้วดูผลทางลอจิกที่ LED_D0 และ LED_D1 จากนั้นบันทึกผลลงในตาราง

Sel (SW1)	**D (SW0)**	**Y1 (LED_D1)**	**Y0 (LED_D0)**
LOW	LOW		
LOW	HIGH		
HIGH	LOW		
HIGH	HIGH		

ตารางที่ 6.4 ตารางความจริงจากทดลองตอนที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 ไอซีวงจรมัลติเพล็กซ์ 74155

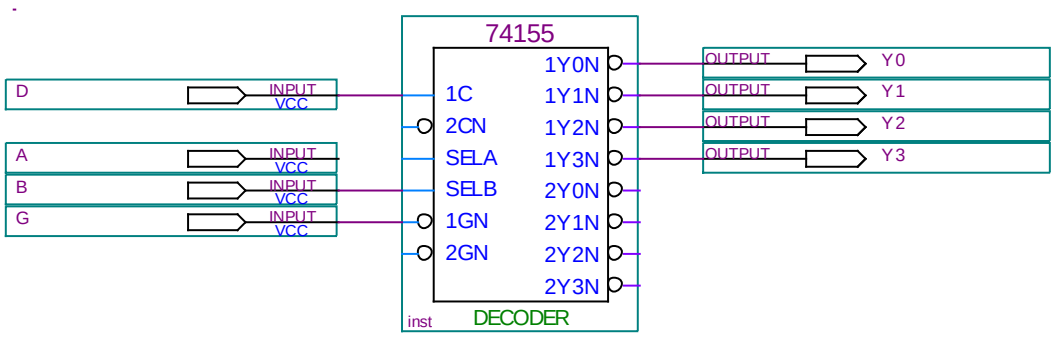
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_6b โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 74155, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

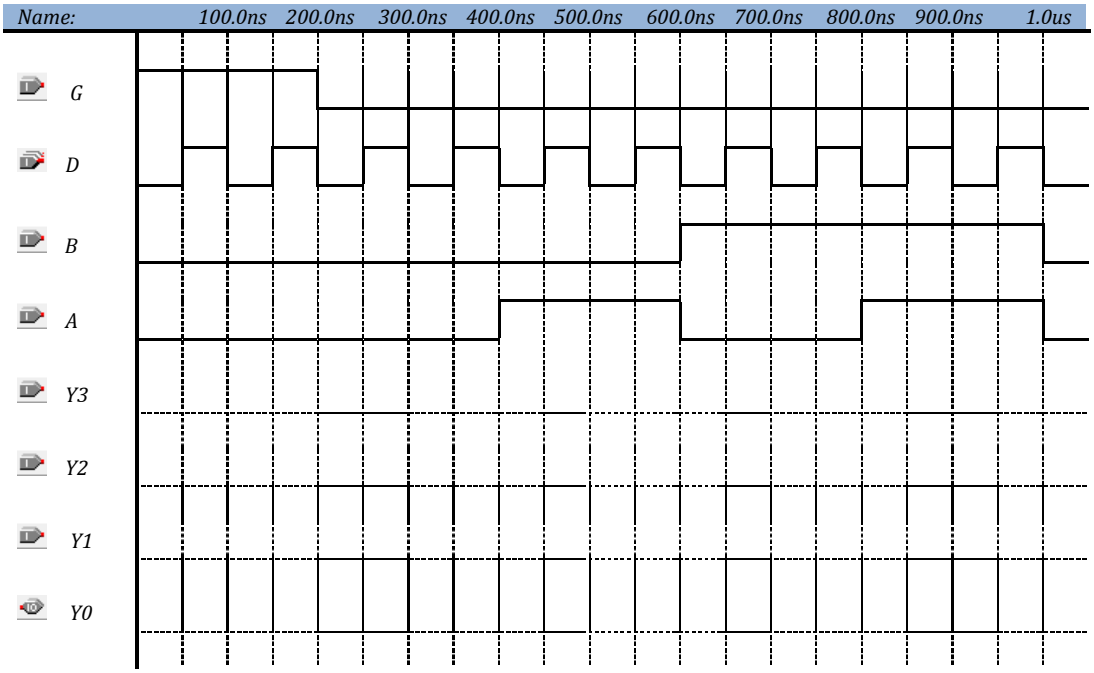


รูปที่ 6.7 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_6b.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_6b.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 6.8 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

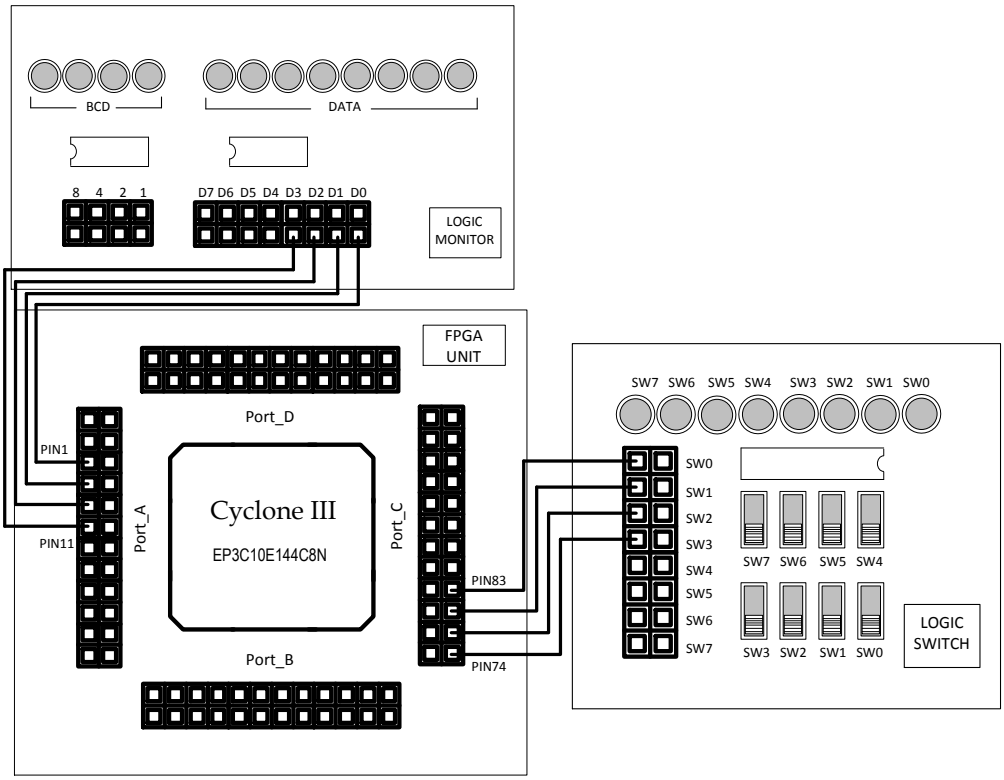
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
G	74	Y0	1
B	76	Y1	3
A	79	Y2	7
D	83	Y3	11

ตารางที่ 6.5 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 6.9 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 6.9 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1, SW2 และ SW3 แล้วดูผลทางลอจิก ที่ LED_D0, LED_D1, LED_D2 และ LED_D3 จากนั้นบันทึกผลลงในตาราง

GN (SW3)	**B (SW2)**	**A (SW1)**	**D (SW0)**	**Y3 (LED_D3)**	**Y2 (LED_D2)**	**Y1 (LED_D1)**	**Y0 (LED_D0)**
1	0	1	0				
1	0	1	1				
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				

ตารางที่ 6.6 ตารางความจริงจากการทดลองตอนที่ 2

สรุปผลการทดลอง

การทดลองที่ 7

วงจรเปรียบเทียบ (Comparator)

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรเปรียบเทียบ
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรเปรียบเทียบได้
4. สามารถดาวน์โหลดวงจรไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

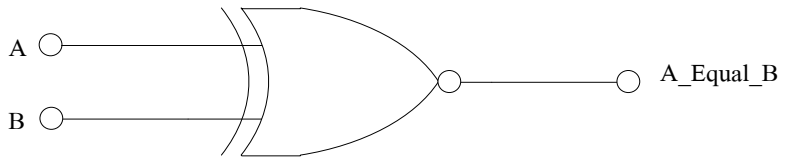
อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

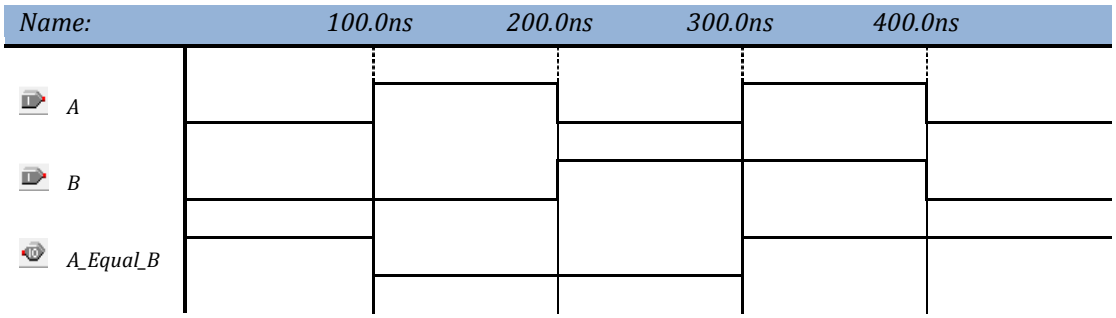
ทฤษฎี

วงจรเปรียบเทียบ (Comparator) คือวงจรที่ใช้สำหรับเปรียบเทียบค่าลอจิกทางอินพุทที่มากกว่า 1 ชุดขึ้นไปโดยอาจเป็นการเปรียบเทียบค่าอินพุทว่ามีลอจิกเท่ากัน มากกว่า หรือน้อยกว่า เป็นต้น ซึ่งวงจรเปรียบเทียบจะมีประโยชน์มากในการตรวจสอบเงื่อนไขเพื่อการตัดสินใจของโปรแกรมต่างๆ เช่นโปรแกรมในเครื่องคัดแยกขนาดของผลิตการเกษตร วงจรเปรียบเทียบอย่างง่ายสามารถสร้างได้จากเอ็กซ์คลูซีฟนอร์เกท นั่นคือ ในกรณีที่อินพุทมีค่าเท่ากัน มันจะให้ค่าเอาต์พุทเป็นลอจิก 1 และในกรณีที่เอาต์พุทมีค่าต่างกันมันจะให้เอาต์พุทเป็นลอจิก 0

นอกจากนี้เราสามารถนำเอ็กซ์คูซีฟนอร์เกทมาทำเป็นวงจรก็ได้เช่นกัน ซึ่งเกททั้งสองจะให้ลอจิกทางเอาต์พุทในลักษณะคล้ายกันเพียงแต่มีค่าตรงข้ามกันเท่านั่น



รูปที่ 7.1 การนำเอ็กซ์คูซีฟนอร์เกทมาทำวงจรเปรียบเทียบอย่างง่าย

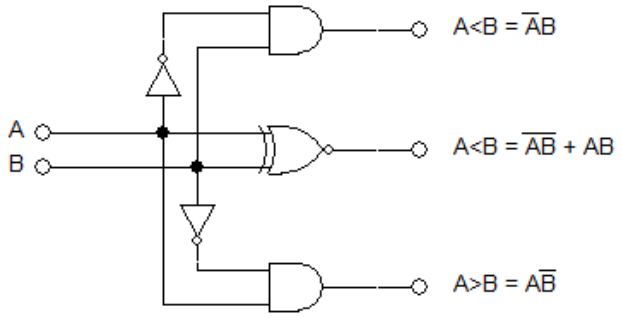


รูปที่ 7.2 ผลการจำลองการทำงานของวงจรการเปรียบเทียบอย่างง่าย

สำหรับวงจรจรเปรียบเทียบที่เพิ่มความสามารถในการตรวจสอบเงื่อนไขที่มากขึ้นแสดงดังรูปที่ 7.2 ซึ่งสร้างจากตารางความจริงในตารางที่ ซึ่งวงจรนี้สามารถตรวจสอบค่าอินพุทที่มากกว่า น้อยกว่า และเท่ากันได้

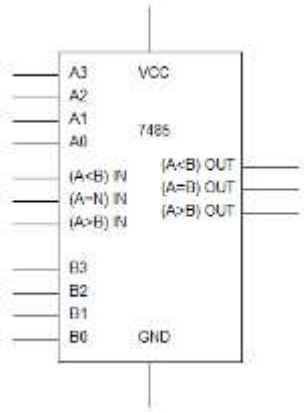
อินพุท		**เอาต์พุท**		
A	B	A<B	A=B	A>B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

ตารางที่ 7.1 ตารางความจริงของวงจรเปรียบเทียบ



รูปที่ 7.3 วงจรเปรียบเทียบ

ตัวอย่างของไอซีสำเร็จรูปที่ทำหน้าที่เป็นตัวเปรียบเทียบลอจิกได้แก่ไอซีเบอร์ 7485 ซึ่งทำหน้าที่เปรียบเทียบแม็กนีจูด (Magnitude) ของอินพุทขนาด 4 บิต 2 ชุด ได้แก่อินพุท A และ B โดยมันสามารถเปรียบเทียบได้ว่าอินพุท A และ B มากกว่า หรือน้อยกว่าได้ (A=B), (A>B), (A<B) ซึ่งการต่อใช้งานสามารถนำไปเปรียบเทียบอินพุทที่มากกว่า 4 บิตได้ ซึ่งสามารถศึกษาได้โดยละเอียดจาก Datasheet ของผู้ผลิต แต่สำหรับการทดลองนี้เราจะทดลองเฉพาะการเปรียบเทียบอินพุทขนาด 4 บิตเท่านั้น



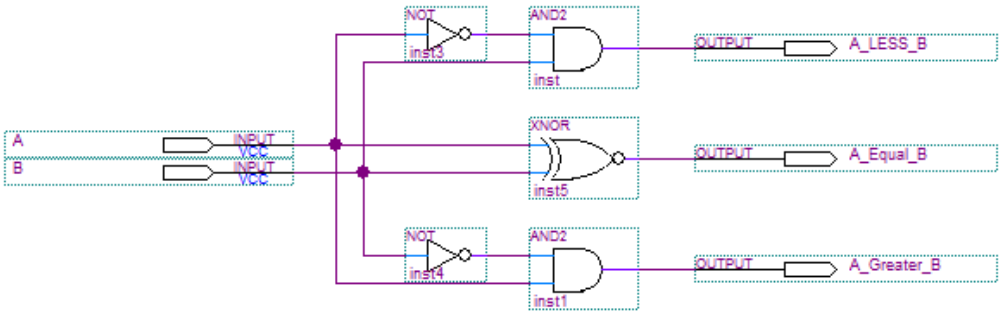
Comparing Input				Outputs		
A3, B3	**A2, B2**	**A1, B1**	**A0, B0**	**A>B**	**A<B**	**A=B**
A3>B3	X	X	X	H	L	L
A3<B3	X	X	X	L	H	L
A3=B3	A2>B2	X	X	H	L	L
A3>B3	A2<B2	X	X	L	H	L
A3<B3	A2=B2	A1>B1	X	H	L	L
A3=B3	A2=B2	A1<B1	X	L	H	L
A3>B3	A2=B2	A1=B1	A0>B0	H	L	L
A3<B3	A2=B2	A1=B1	A0<B0	L	H	L
A3=B3	A2=B2	A1=B1	A0=B0	L	L	H

รูปที่ 7.4 ไอซี 7485

ตารางที่ 7.2 ตารางความจริงของไอซี 7485

ขั้นตอนการทดลอง ตอนที่ 1 วงจรเปรียบเทียบพื้นฐาน

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_7a โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ and2, or2, not, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

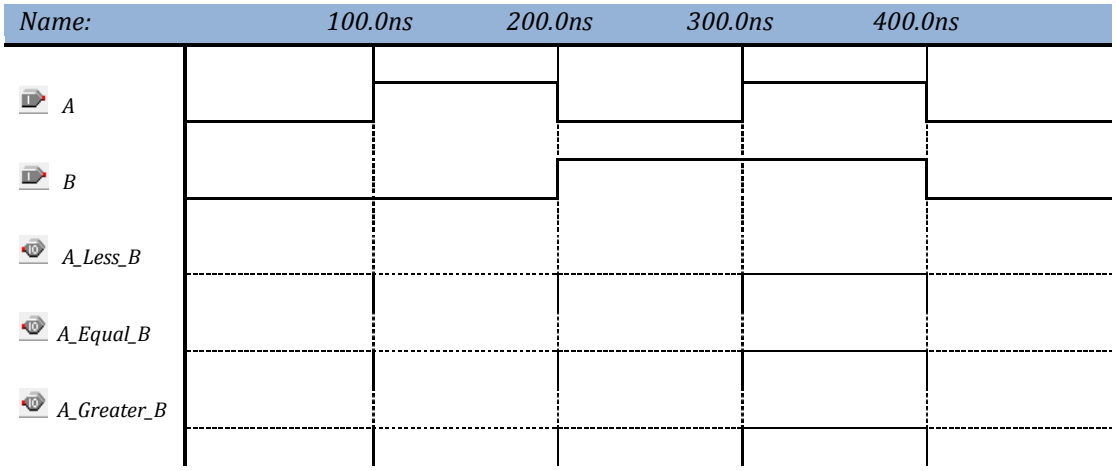


รูปที่ 7.5 ผลการจำลองการทำงาน

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_7a.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_7a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 7.6 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

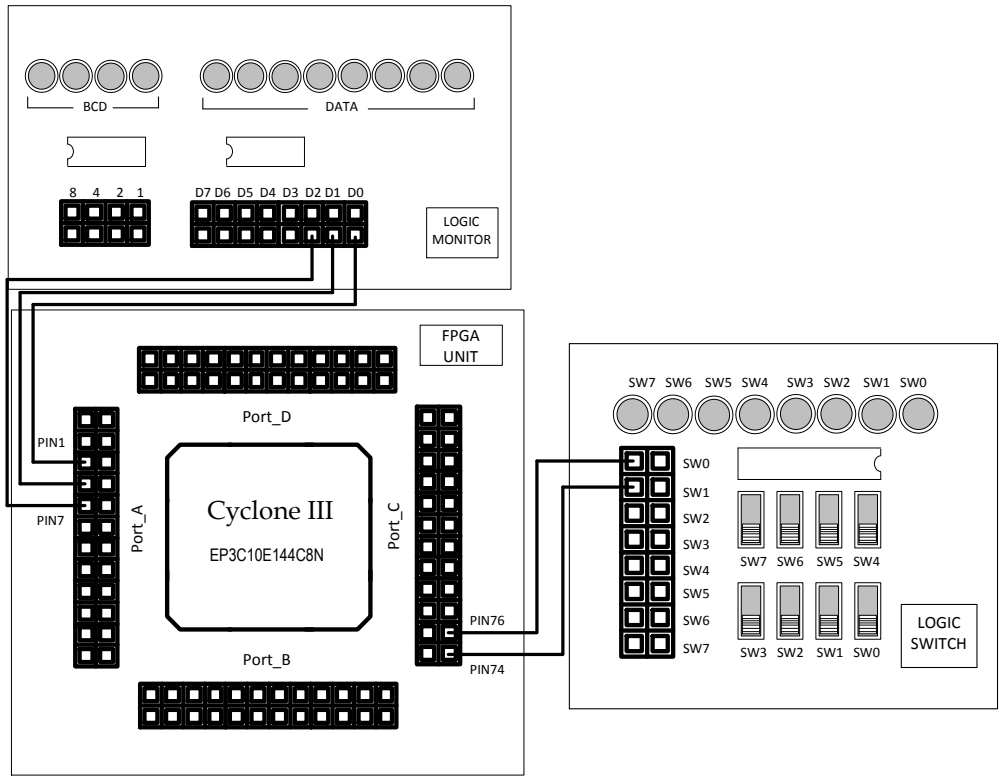
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A	76
B	74
A_Lass_B	7
A_Equal_B	3
A_Greater_B	1

ตารางที่ 7.3 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 7.7 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 7.7 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิก ที่ LED_D0, LED_D2 และ LED_D3 จากนั้นบันทึกผลลงในตาราง

B (SW1)	**A (SW0)**	**(A_Less_B) LED D2**	**(A_Equal_B) LED D1**	**(A_Greater) LED D0**
LOW	LOW			
LOW	HIGH			
HIGH	LOW			
HIGH	HIGH			

ตารางที่ 7.4 ตารางความจริงจากการทดลองตอนที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 การใช้ไอซีวงจรเปรียบเทียบ 7485

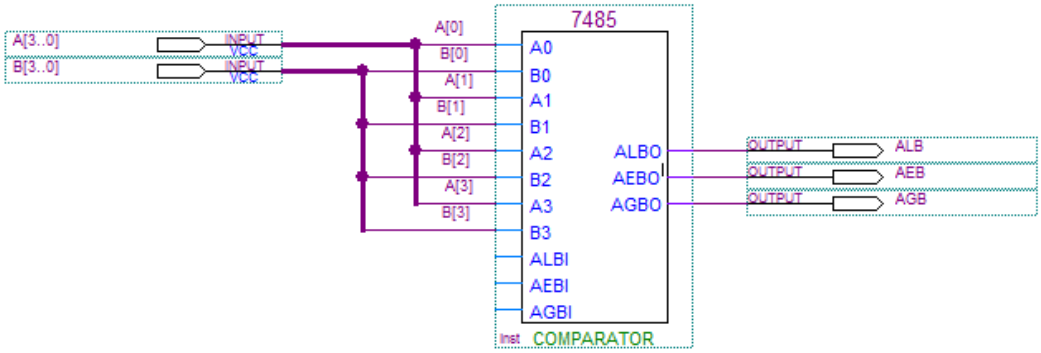
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_7b โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 7485, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป



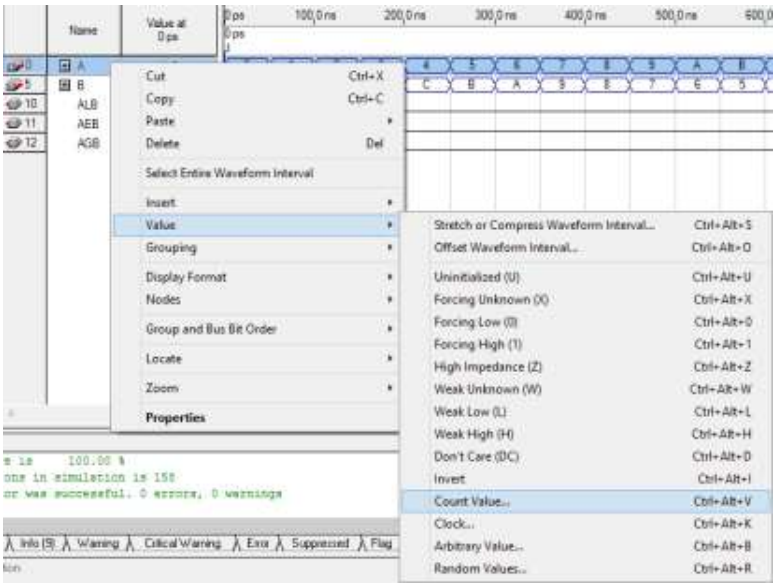
รูปที่ 7.8 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_7b.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

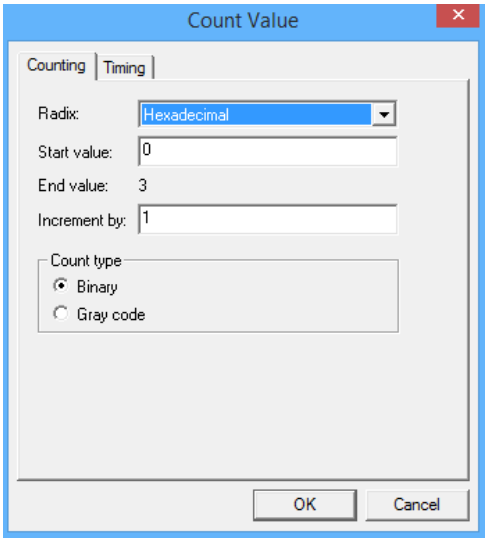
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_7b.scf เพื่อจำลองการทำงานของวงจร โดยกำหนดสัญญาณให้เป็นดังต่อไปนี้

a. กำหนดสัญญาณให้เป็นแบบ Count ซึ่งทำได้โดยการคลิกขวาที่ชื่อสัญญาณจากนั้นจะมีเมนู Value -> Count Value… ดังรูปที่ 7.8a



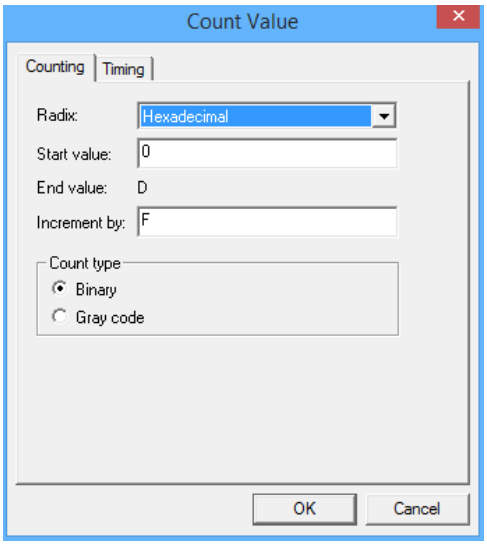
รูปที่ 7.8a กำหนดค่าในช่อง Radix

b. กำหนดค่าในช่อง Radix. ให้เป็น Hexadecimal ในสัญญาณ A ให้กำหนดค่าในช่อง Starting Value เป็น 0 และ Increment By มีค่าเท่ากับ 1 ดังรูปที่ 7.8b

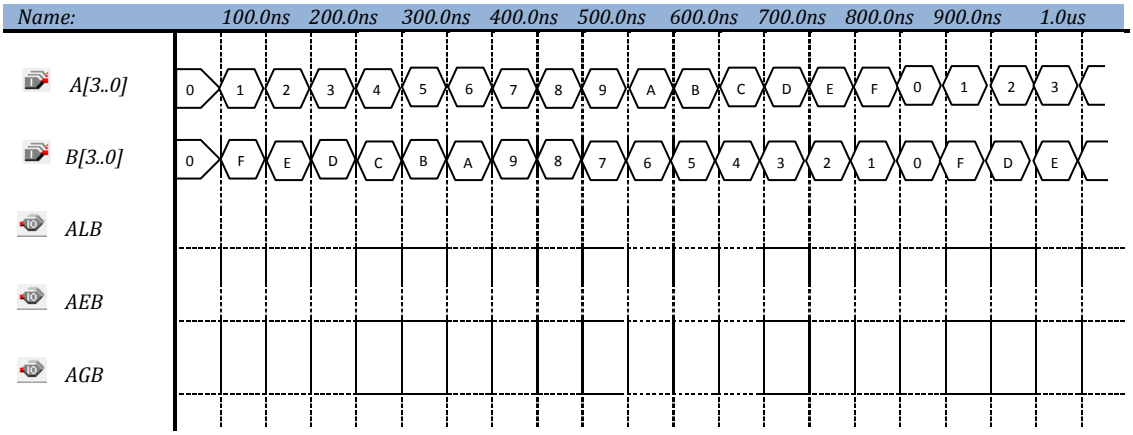


รูปที่ 7.8b กำหนดค่าเริ่มต้นให้กับสัญญาณ A

c. กำหนดค่าในช่อง Radix .ให้เป็น Hexadecimal ในสัญญาณ B ให้กำหนดค่าในช่อง Starting Value เป็น 0 และ Increment By มีค่าเท่ากับ F ดังรูปที่ 7.8c



รูปที่ 7.8c กำหนดค่าเริ่มต้นให้กับสัญญาณ B



รูปที่ 7.9 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

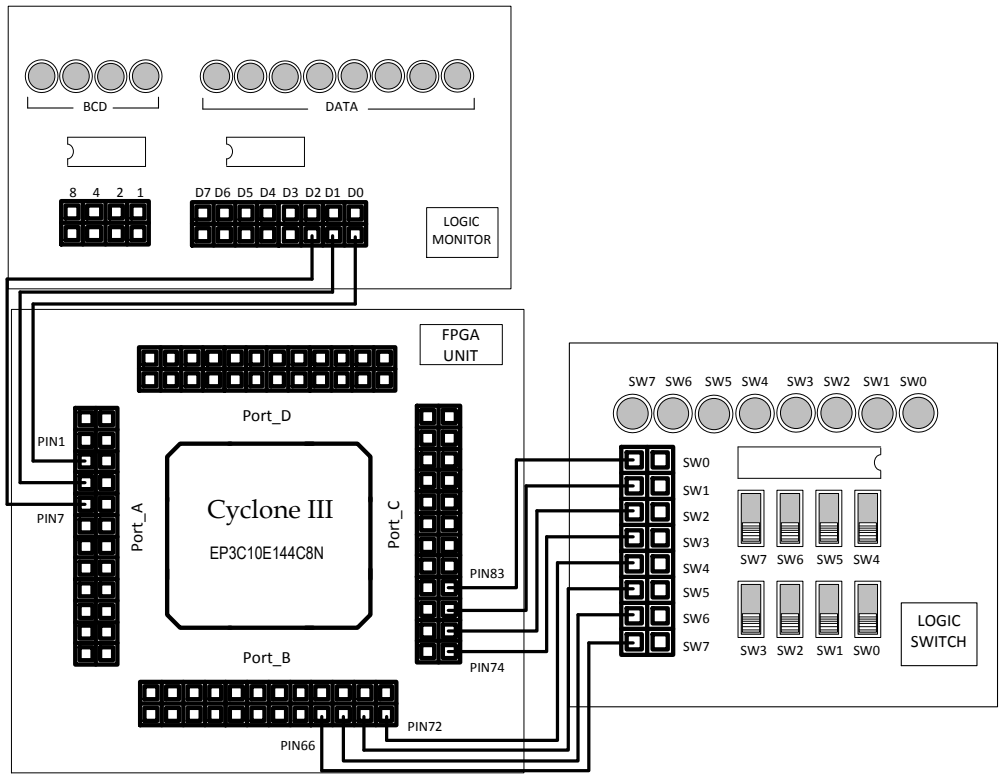
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
A0	83	B2	68
A1	79	B3	66
A2	76	ALB	7
A3	74	AEB	3
B0	72	AGB	1
B1	70		

ตารางที่ 7.5 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 7.10 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1, SW2, SW3, SW4, SW5, SW6 และ SW7 แล้วดูผลทางลอจิก ที่ LED_D0, LED_D1 และ LED_D2 จากนั้นบันทึกผลลงในตาราง

B3 (SW7)	B2 (SW6)	B1 (SW5)	B0 (SW4)	B3 (SW3)	B2 (SW2)	B1 (SW1)	B0 (SW0)	ALB (LED_D2)	AEB (LED_D1)	AGB (LED_D0)
0	0	0	0	0	0	0	0			
0	0	0	0	0	0	0	1			
0	0	0	1	0	0	0	0			
0	0	0	1	0	0	1	0			
0	0	1	0	0	0	0	1			
0	0	1	1	0	1	0	0			
0	1	0	0	0	0	1	1			
0	1	1	1	1	0	0	0			
1	0	0	0	0	1	1	1			
1	1	1	1	1	1	1	1			

ตารางที่ 7.6 ตารางความจริงจากทดลองตอนที่ 2

สรุปผลการทดลอง

การทดลองที่ 8

วงจรบวกเลขฐานสอง (Adder)

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรบวกเลขฐานสองแบบ Half Adder และ Full Adder
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรบวกเลขฐานสองได้แบบ Half Adder และ Full Adder ได้
4. สามารถดาวน์โหลดวงจรไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

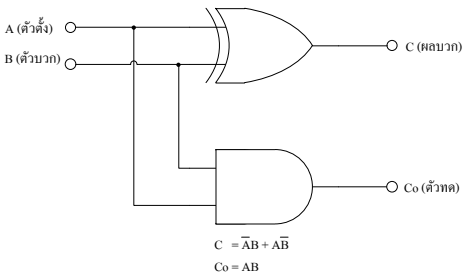
ทฤษฎี

วงจรบวกเลขฐานสองแบ่งเป็นสองชนิดได้แก่ วงจรบวกเลขฐานสองแบบ Half Adder และ Full Adder ซึ่งวงจรแบบ Half Adder จะเป็นวงจรบวกแบบไม่มีการนำตัวทดจากหลักต่ำกว่ามาร่วมคำนวณด้วย แต่สำหรับวงจรแบบ Full Adder จะเป็นวงจรบวกที่มีการนำเอาตัวทดของของหลักที่ต่ำกว่ามาร่วมคำนวณด้วย ซึ่งวงจรทั้งสองสามารถสร้างขึ้นจากการเขียนตารางความจริงโดยใช้หลักการบวกเลขฐานสองมาพิจารณา

สำหรับวงจรบวกแบบ Half Adder สามารถเขียนเป็นตารางความจริงได้ดังตารางที่ 8.1 และสามารถสร้างวงจรได้ดังรูปที่ 8.1

อินพุท		เอาต์พุท	
A (ตัวตั้ง)	B (ตัวบวก)	C (ผลบวก)	Co (ตัวทด)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

ตารางที่ 8.1 ตารางความจริงของวงวงจร Half Adder

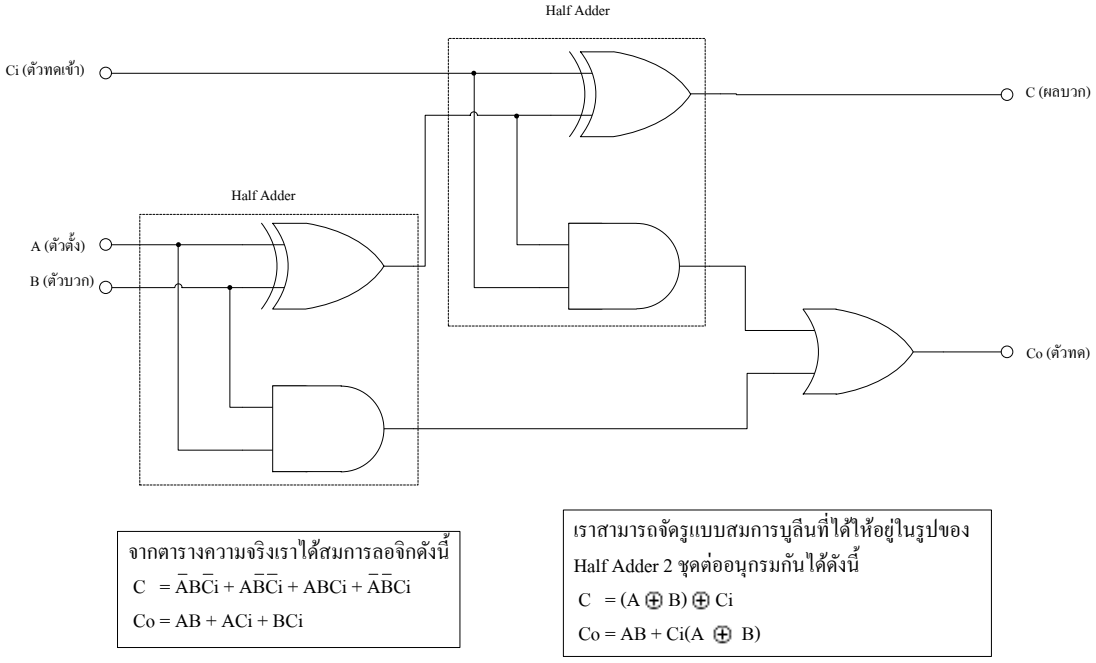


รูปที่ 8.1 วงจร Half Adder

ส่วนวงจรบวกแบบ Full Adder จะสามารถเขียนเป็นตารางความจริงและวงจรได้ดังตารางที่ 8.2 และรูปที่ 8.2 ตามลำดับ ซึ่งจะสังเกตได้ว่าอินพุทของวงจร Full Adder จะมีอินพุทของตัวทดบิตต่ำกว่ามาคำนวณด้วย

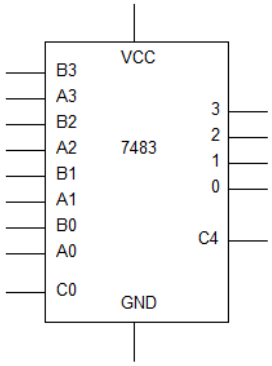
อินพุท			**เอาต์พุท**	
Ci (ตัวทดเข้า)	**A (ตัวติดตั้ง)**	**B (ตัวบวก)**	**C=A+B+Ci (ผลบวก)**	**Co (ตัวทดออก)**
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

ตารางที่ 8.2 ตารางความจริงของวง Full Adder



รูปที่ 8.2 วงจร Full Adder

ไอซีสำเร็จรูปที่ทำหน้าที่เป็นวงจรบวกแบบ Full Adder ได้แก่เบอร์ 7483 ซึ่งเป็นไอซี Full Adder ขนาด 4 บิต โดยภายในประกอบไปด้วยวงจร Full Adder 4 ชุด มีการต่อขาตัวทดจากแต่ละหลักไปเป็นอินพุทของหลักถัดไปอยู่ภายในตัวมัน และมีขาตัวทดจากหลักสุดท้ายเพื่อใช้สำหรับต่อขยายในการคำนวณระบบตัวเลขที่มีบิตมากขึ้น สำหรับสัญลักษณ์และตารางความจริงของ 7483 แสดงดังรูปที่ 8.3 และตารางที่ 8.3 ตามลำดับ

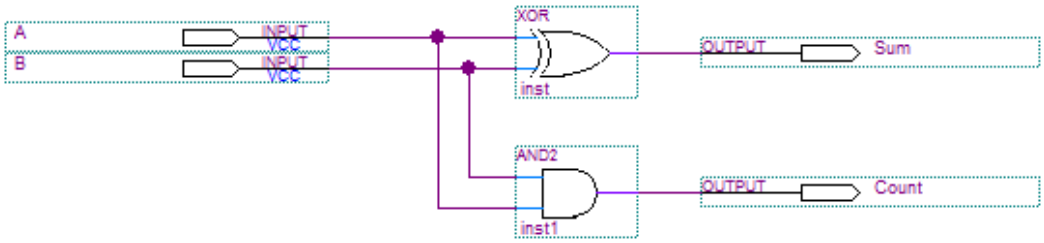


Comparing Input			Output	
C(n-1)	**An**	**Bn**	**En**	**Cn**
L	L	L	L	L
L	L	H	H	L
L	H	L	H	L
L	H	H	L	H
H	L	L	H	L
H	L	H	L	H
H	H	L	L	H
H	H	H	H	H

รูปที่ 8.3 ไอซี 7483 ตารางที่ 8.3 ตารางความจริงของไอซี 7483

ขั้นตอนการทดลอง ตอนที่ 1 วงจร Half Adder

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_8a โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ xor, and2, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

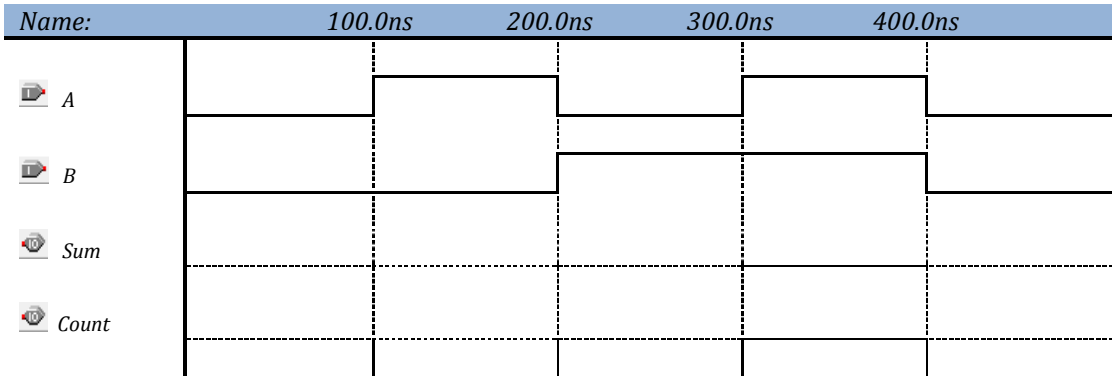


รูปที่ 8.4 วงจรสำหรับการทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_8a.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_8a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 8.5 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

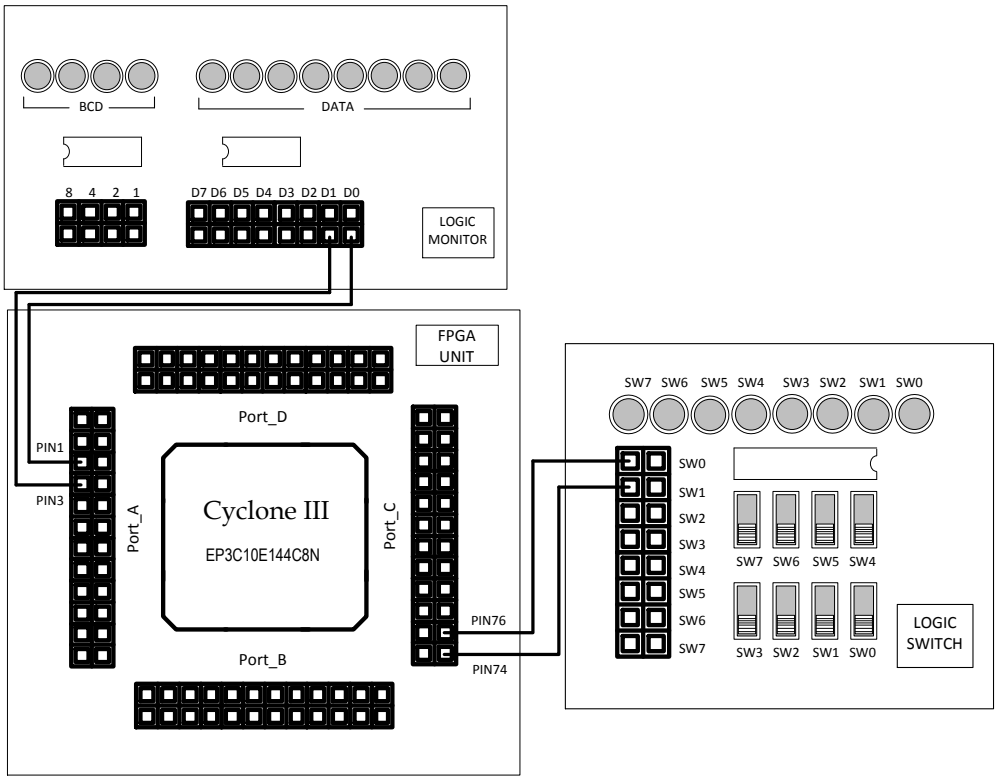
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A	74
B	76
Sum	3
Cout	1

ตารางที่ 8.4 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 8.6 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 8.6 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1 และ SW2 แล้วดูผลทางลอจิกที่ LED_D0, LED_D1 จากนั้นบันทึกผลลงในตาราง

A (SW1)	**B (SW0)**	**Sum (LED D1)**	**Count (LED D0)**
LOW	LOW		
LOW	HIGH		
HIGH	LOW		
HIGH	HIGH		

ตารางที่ 8.5 ตารางความจริงจากทดลองตอนที่ 2

ขั้นตอนการทดลอง ตอนที่ 2 วงจร Full Adder

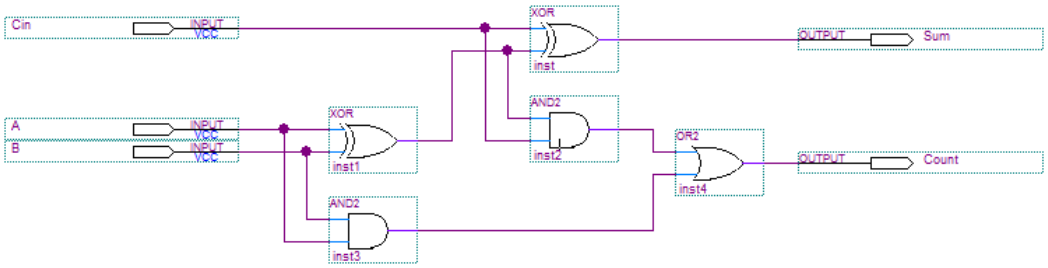
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_8b โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ xor, and2, or2, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

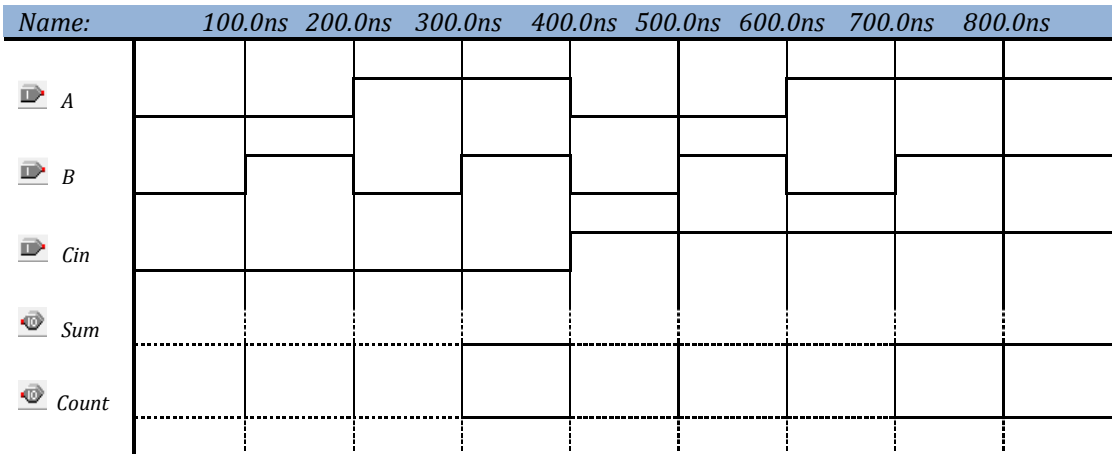


รูปที่ 8.7 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_8b.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_8b.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 8.8 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

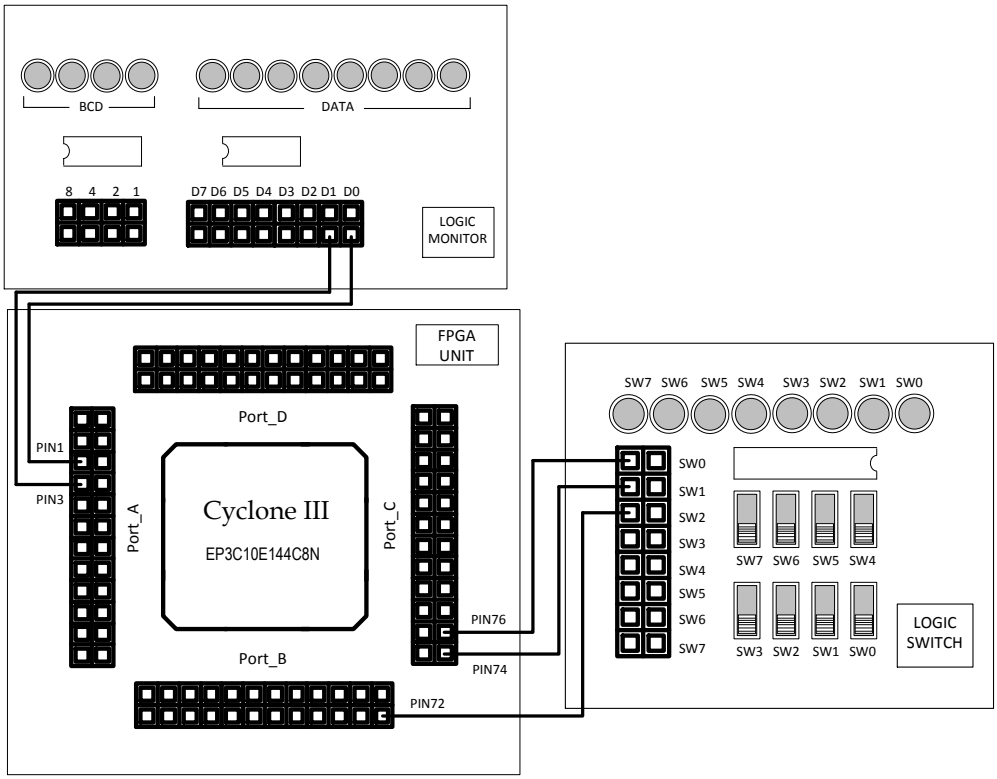
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A	72
B	74
Cin	76
Sum	3
Count	1

ตารางที่ 8.6 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 8.9 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 8.9 การต่อวงจรบนชุดทดลอง

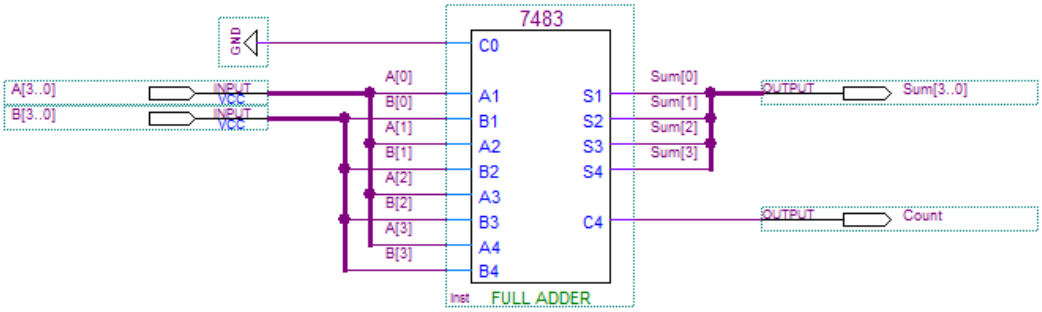
13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1 และ SW2 แล้วดูผลทางลอจิกที่ LED_D0 และ LED_D1 จากนั้นบันทึกผลลงในตาราง

Cin (SW2)	A (SW1)	B (SW0)	Sum (LED D1)	Count (LED D0)
LOW	LOW	LOW		
LOW	LOW	HIGH		
LOW	HIGH	LOW		
LOW	HIGH	HIGH		
HIGH	LOW	LOW		
HIGH	LOW	HIGH		
HIGH	HIGH	LOW		
HIGH	HIGH	HIGH		

ตารางที่ 8.7 ตารางความจริงจากทดลองตอนที่ 2

ขั้นตอนการทดลอง ตอนที่ 3 วงจร Full Adder 7483

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_8b โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 7483, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

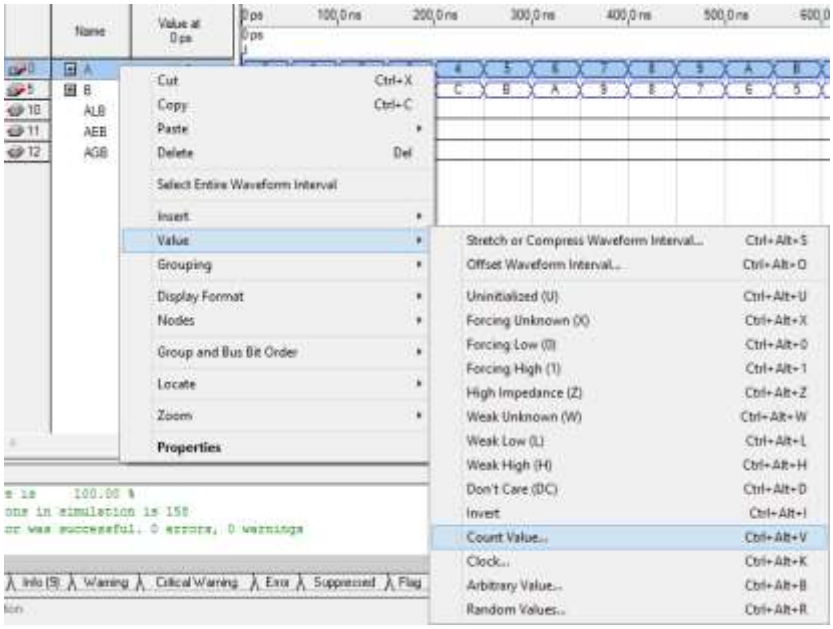


รูปที่ 8.10 วงจรสำหรับการทดลองตอนที่ 3

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_8c.gdf
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

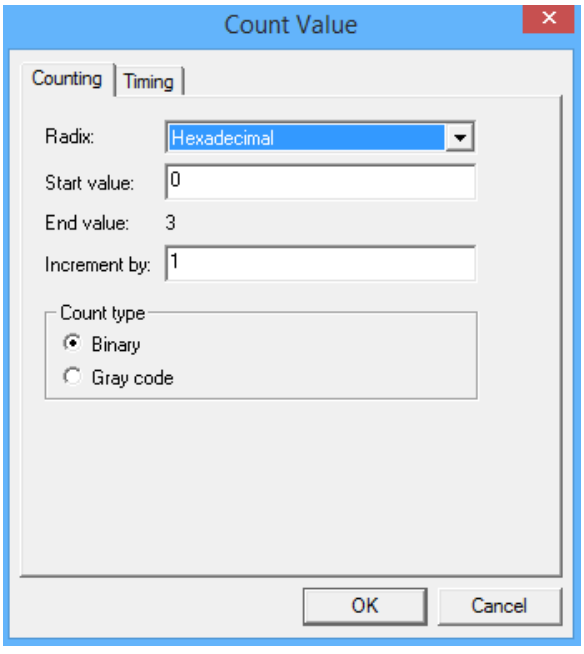
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_8c.scf เพื่อจำลองการทำงานของวงจร

a. กำหนดสัญญาณให้เป็นแบบ Count ซึ่งทำได้โดยการคลิกขวาที่ชื่อสัญญาณจากนั้นจะมีเมนู Value -> Count Value… ดังรูปที่ 8.11a



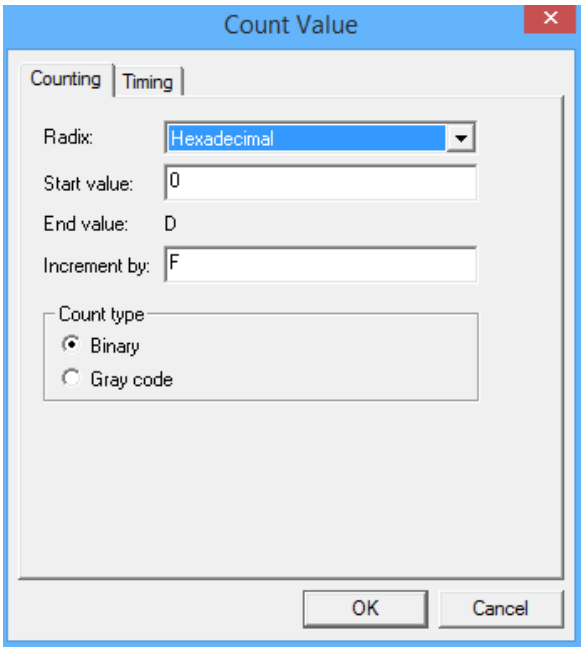
รูปที่ 8.11a กำหนดค่าในช่อง Radix

b. กำหนดค่าในช่อง Radix. ให้เป็น Hexadecimal ในสัญญาณ A ให้กำหนดค่าในช่อง Starting Value เป็น 0 และ Increment By มีค่าเท่ากับ 1 ดังรูปที่ 8.11b

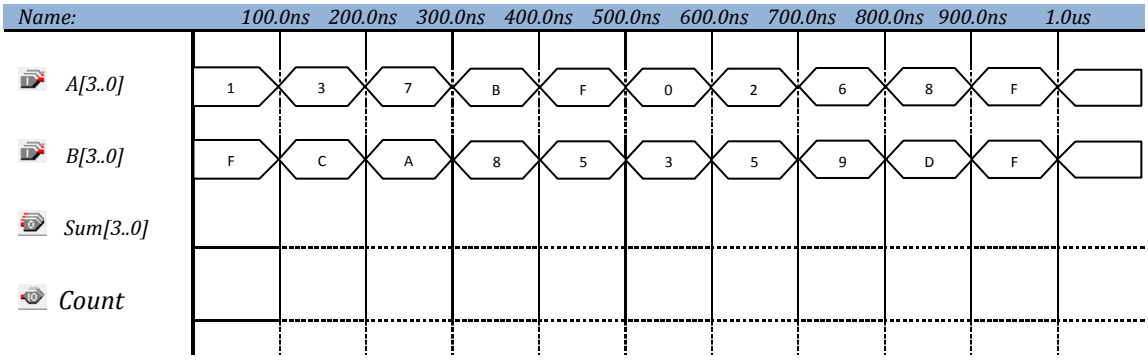


รูปที่ 8.11b กำหนดค่าเริ่มต้นให้กับสัญญาณ A

c. กำหนดค่าในช่อง Radix ให้เป็น Hexadecimal ในสัญญาณ B ให้กำหนดค่าในช่อง Starting Value เป็น 0 และ Increment By มีค่าเท่ากับ F ดังรูปที่ 8.11c



รูปที่ 8.11c กำหนดค่าเริ่มต้นให้กับสัญญาณ B



รูปที่ 8.12 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

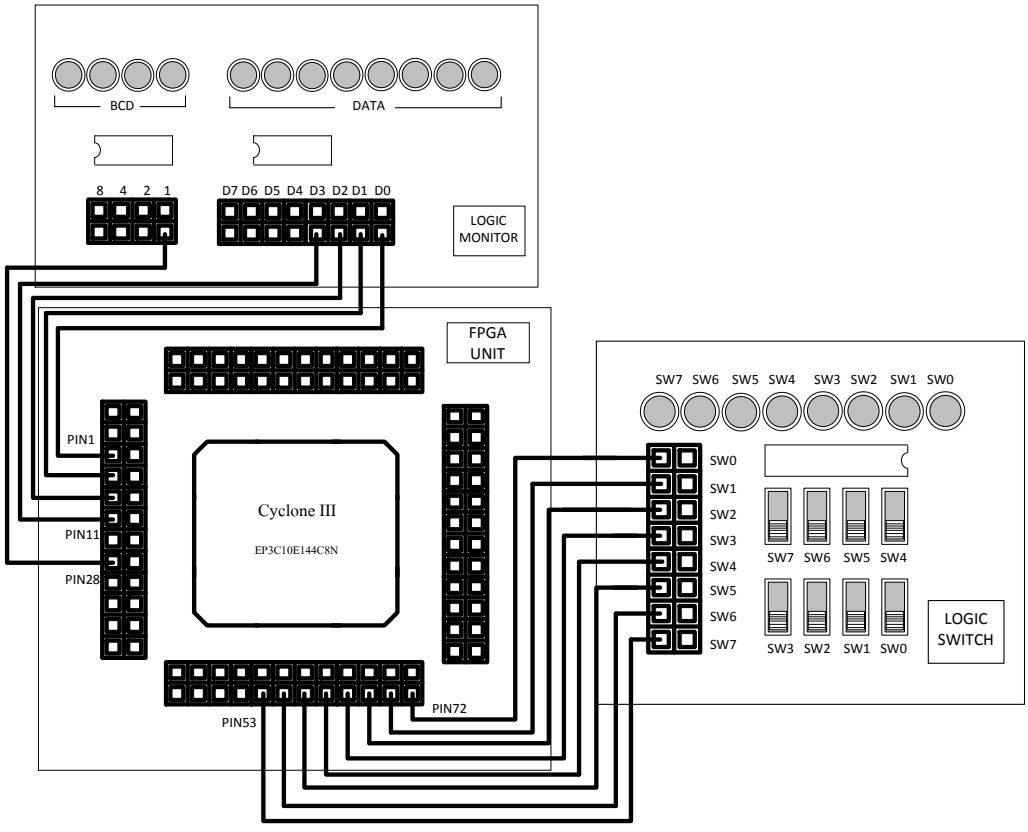
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
A0	72	B3	72
A1	70	Sum0	1
A2	68	Sum1	3
S3	66	Sum2	7
B0	64	Sum3	11
B1	59	Count	28
B2	55		

ตารางที่ 8.8 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 8.13 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 8.13 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1, SW2, SW3, SW4, SW5, Sw6 และ SW7 แล้วดูผลทางลอจิก ที่ LED D0, LED D1, LED D2, LED D3 และ LED 1 จากนั้นบันทึกผลลงในตาราง

B3 (SW7)	B2 (SW6)	B1 (SW5)	B0 (SW4)	A3 (SW3)	A2 (SW2)	A1 (SW1)	A0 (SW0)	Count (LED 1)	Sum3 (LED D3)	Sum2 (LED D2)	Sum1 (LED D1)	Sum0 (LED D0)
0	0	0	0	0	0	0	0					
0	0	0	1	0	0	0	1					
0	0	1	0	0	0	1	0					
0	0	1	1	0	0	1	1					
0	1	0	0	0	1	0	0					
0	1	0	1	0	1	0	1					
0	1	1	0	0	1	1	0					
0	1	1	1	0	1	1	1					
1	0	0	0	1	0	0	0					
1	0	0	1	1	0	0	1					
1	0	1	0	1	0	1	0					
1	0	1	1	1	0	1	1					
1	1	0	0	1	1	0	0					
1	1	0	1	1	1	0	1					
1	1	1	0	1	1	1	0					
1	1	1	1	1	1	1	1					

ตารางที่ 8.9 ตารางความจริงจากทดลองตอนที่ 3

สรุปผลการทดลอง

การทดลองที่ 9

วงจรลบเลขฐานสอง (Subtractor)

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรลบเลขฐานสองแบบ Half Subtractor และ Full Subtractor
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรลบเลขฐานสองแบบ Half Subtractor และ Full Subtractorได้
4. สามารถดาวน์โหลดวงจรไอซี เพื่อทดสอบการทำงานจริงของวงจรได้

อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

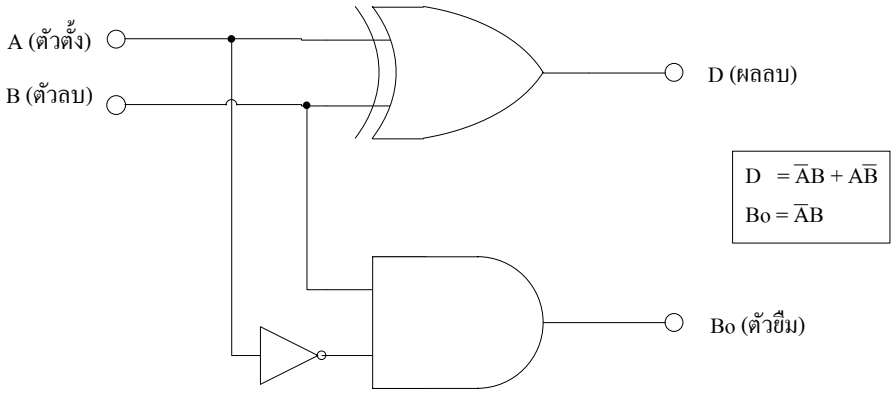
ทฤษฎี

หลักการลบในระบบจำนวนใด ๆ หากตัวลบมีค่ามากกว่าตัวตั้ง ตัวตั้งจะต้องมีการยืมจากหลักที่สูงกว่าจะทำให้หลักที่ถูกยืมค่าลดลง ซึ่งในการสร้างวงจรเลขฐานสองจึงต้องมีความเกี่ยวข้องกับตัวยืมด้วยเช่นกัน

วงจรลบเลขฐานสองแบ่งออกเป็นสองชนิดเช่นเดียวกับวงจรบวกซึ่งได้แก่ วงจรลบเลขฐานสองแบบ Half Subtractor และแบบ Full Subtractor ซึ่งวงจรแบบ Half Subtractor จะเป็นวงจรลบแบบไม่มีการนำตัวยืมที่ถูกยืมจากหลักต่ำกว่ามาร่วมคำนวณด้วย และสำหรับวงจรแบบ Full Sub tractor จะเป็นวงจรลบที่มีการนำตัวยืมที่ถูกยืมไปของหลักที่ต่ำกว่ามาเป็นอินพุทของวงจรเพื่อใช้ร่วมคำนวณด้วย ซึ่งวงจรทั้งสองสามารถสร้างขึ้นจากการเขียนตารางความจริงโดยใช้หลักการลบเลขฐานสองมาพิจารณา สำหรับวงจรลบแบบ Half Subtractor สามารถเขียนเป็นตารางความจริงและสร้างเป็นวงจรได้ดังตารางที่ 9.1 และรูปที่ 9.1

อินพุท		เอาต์พุท		
A (ตัวตั้ง)	**B (ตัวลบ)**	**D = A - B (ผลลบ)**	**Bo (ตัวยืม)**	**หมายเหตุ**
0	0	0	0	0-0 = ยืม 0
0	1	1	1	0-1 = ยืม 1
1	0	1	0	1-0 = ยืม 0
1	1	0	0	1-1 = ยืม 0

ตารางที่ 9.1 ตารางความจริงของวงจร Half Subtractor

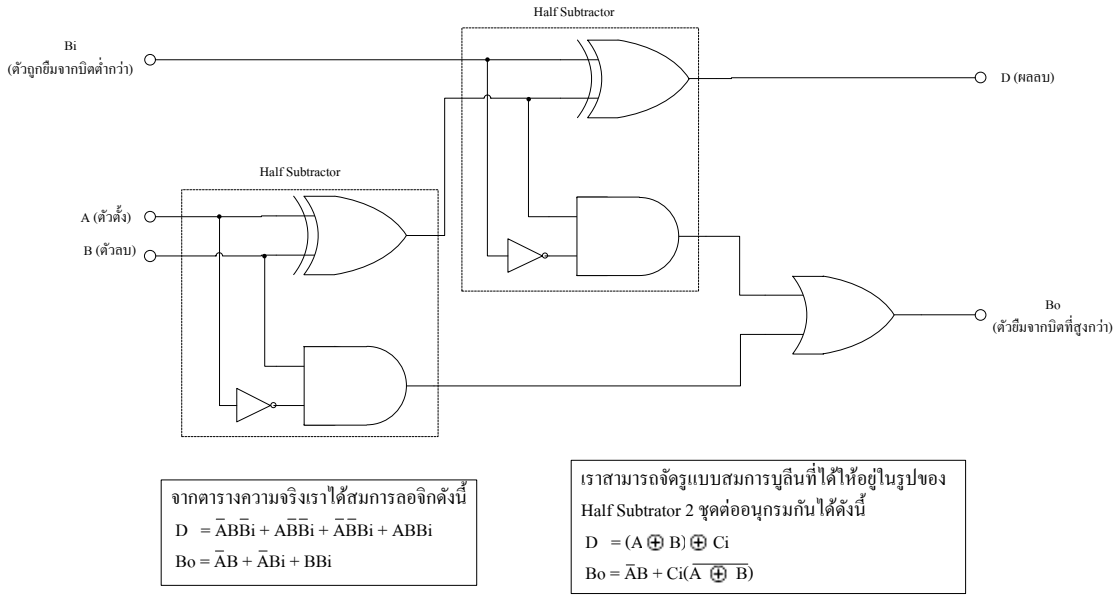


รูปที่ 9.1 วงจร Half subtract

ส่วนวงจรลบแบบ Full Sub tractor จะสามารถเขียนเป็นตารางความจริงและวงจรได้ตาราง ที่ และรูปที่ ตามลำดับ จะสังเกตได้อินพุทของวงจร Full Sub tractor จะมีอินพุทของยืมที่ถูกยืมไปของบิตต่ำกว่ามาคำนวณด้วย

อินพุท			เอาต์พุท		
A (ตัวตั้ง)	**B (ตัวลบ)**	**Bi (ตัวถูกยืมจากบิตต่ำกว่า)**	**D = A - B - Bi (ผลลบ)**	**Bo (ตัวยืมจากบิตสูงกว่า)**	**หมายเหตุ**
0	0	0	0	0	0-0-0= 0 ยืม 0
0	0	1	1	1	0-0-1= 1 ยืม 1
0	1	0	1	1	0-1-0= 0 ยืม 1
0	1	1	0	1	0-1-1= 0 ยืม 1
1	0	0	1	0	0-1-1= 0 ยืม 0
1	0	1	0	0	1-0-1= 0 ยืม 0
1	1	0	0	0	1-1-0= 0 ยืม 0
1	1	1	1	1	1-1-1= 1 ยืม 1

ตารางที่ 9.2 ตารางความจริงของวงจรจร Full Subtractor



รูปที่ 9.2 วงจร Full subtract

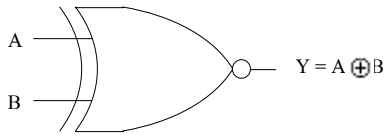
วิธีการลบเลขฐานสองอีกแบบที่นิยมใช้คือการใช้ 2's Complement ซึ่งมีวิธีการดังนี้

1. เปลี่ยนตัวลบให้เป็น 2's Complement (กลับบิตเป็นตรงกันข้ามแล้วบวกด้วย 1 เข้าที่บิต LSB)
2. นำตัวเลข 2's Complement ที่ได้จากข้อ 1 ไปบวกกับตัวตั้ง แล้วพิจารณาผลลัพธ์ที่ได้ดังนี้

   a. ถ้าเกิดตัวทดแสดงว่า ผลลัพธ์ที่ได้มีค่าเป็นบวก และให้ตัดตัวทดทิ้ง

b. ถ้าไม่เกิดตัวทดแสดงว่า ผลลัพธ์ที่ได้เป็นเลขลบ และให้นำผลไปแปลงเป็นตัวเลข 2’s Complement อีกครั้ง

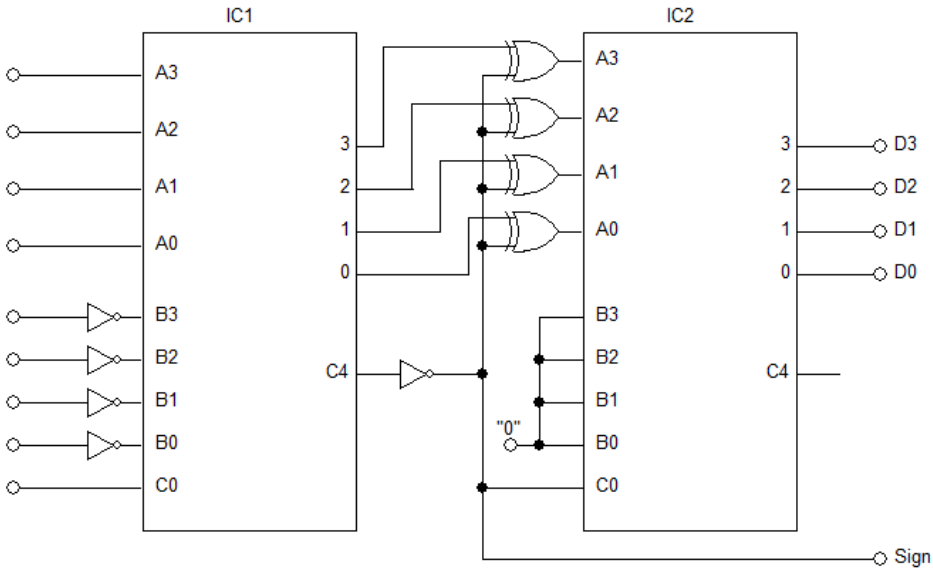
จากวิธีดังกล่าวจะเห็นได้ว่าเป็นการลบที่อาศัยการบวก แต่ใช้การแปลงตัวลบและพิจารณาผลลัพธ์แทน ซึ่งหลักการเหล่านี้เราสามารถนำไอซี 7483 ซึ่งเป็นไอซี Full Adder ขนาด 4 บิต มาสร้างเป็นวงจรลบเลขฐานสองได้ โดยใช้ นอตเกทมาทำให้ตัวลบเป็น 2’s Complement ร่วมกับขา C0 และอาศัย เอ็กซ์คลูซีฟออร์ เกทที่เราจะนำมาใช้คือ ในกรณีที่ให้อินพทขาใดขาหนึ่งเป็นลอจิกต่ำ มันจะให้เอาต์พุทเหมือนกับอินพุทขาที่เหลือ ซึ่งจากหลักการเราสามารถนำมาใช้ควบคุมให้เกิดการเปลี่ยนแปลงผลลัพธ์เป็นคอมพรีเมนต์หรือไม่แปลงก็ได้ โดยวงลบแบบใช้ 2’s Complement แสดงดังรูปที่ 9.4



รูปที่ 9.3 เอ็กซ์คลูซีฟออร์เกท

อินพุท		เอาต์พุท
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

ตารางที่ 9.3 ตารางความจริงของเอ็กซ์คลูซีฟออร์เกท



รูปที่ 9.4 วงจรลบที่สร้างจากหลักการ 2’s Complement

ขั้นตอนการทดลอง ตอนที่ 1 วงจร Half Subtrator

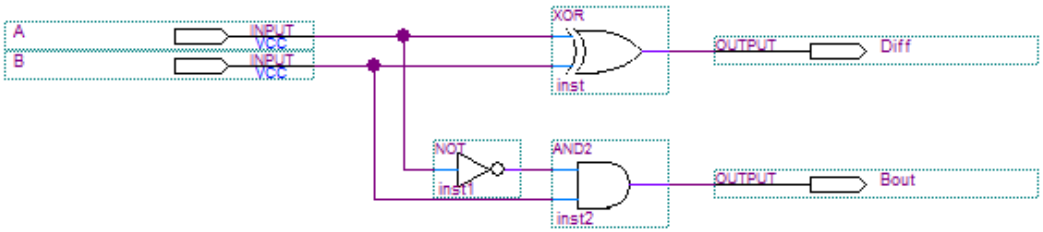
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_9a โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ xor, and2, not, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

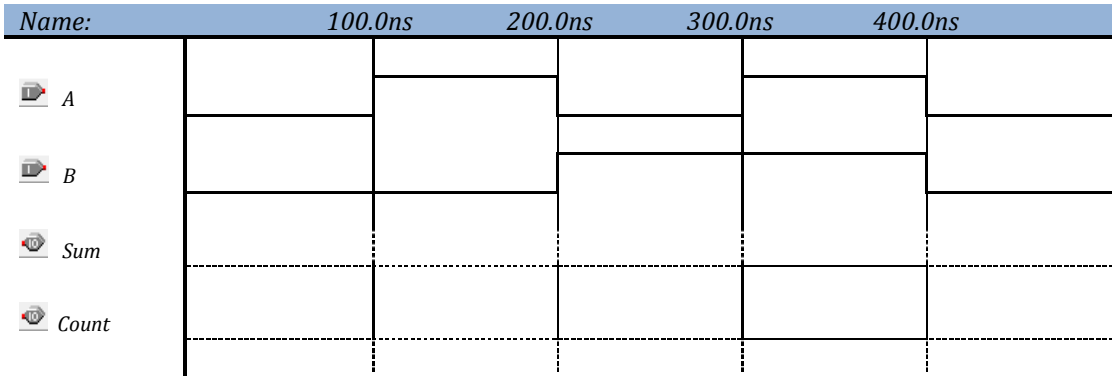


รูปที่ 9.5 วงจรสำหรับการทดลองที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_9a.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_9a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 9.6 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

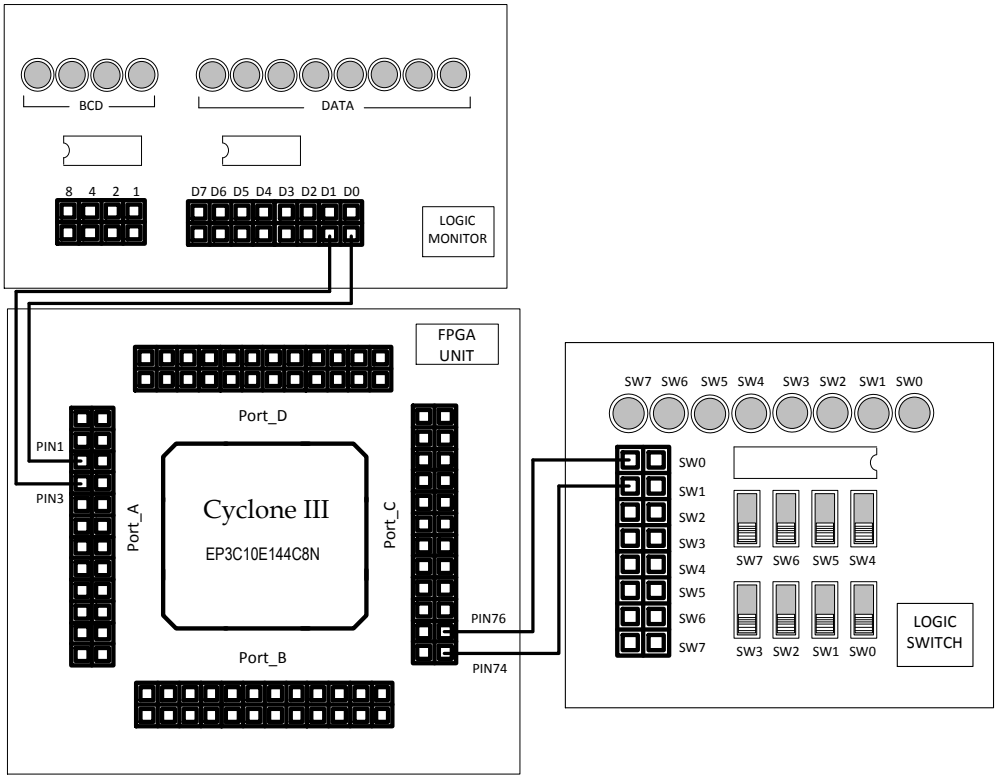
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A	74
B	76
Diff	3
Bout	1

ตารางที่ 9.4 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 9.7 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 9.7 การต่อวงจรบนชุดทดลอง

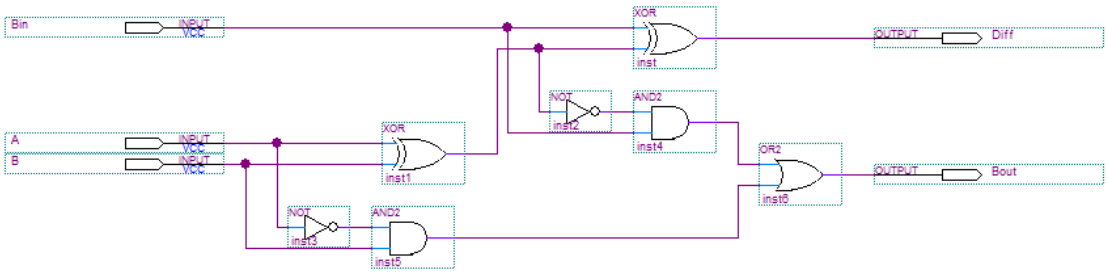
13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิก ที่ LED D0 และ LED D1 จากนั้นบันทึกผลลงในตาราง

A (SW1)	B (SW0)	Diff (LED D1)	Bout (LED D0)
LOW	LOW		
LOW	LOW		
HIGH	HIGH		
HIGH	HIGH		

ตารางที่ 9.5 ตารางความจริงจากการทดลองตอนที่ 1

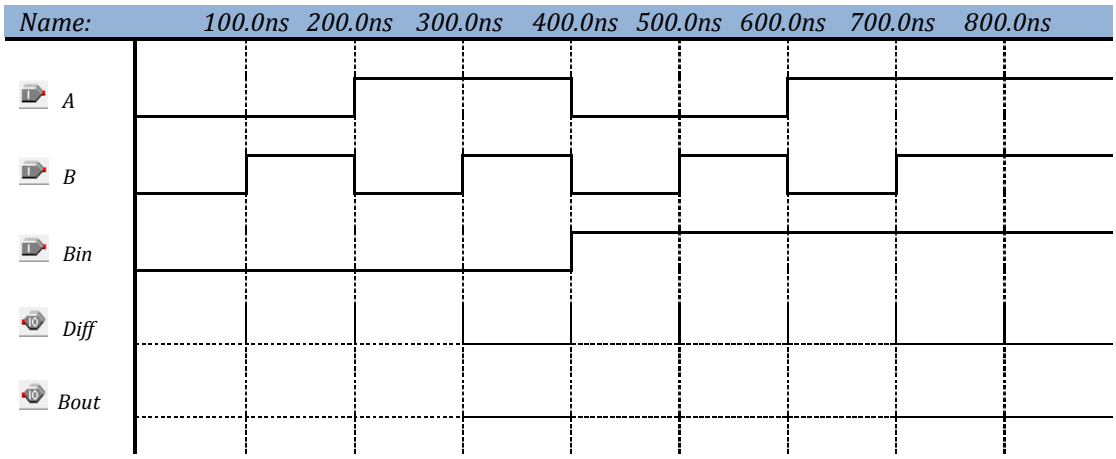
ขั้นตอนการทดลอง ตอนที่ 2 วงจร Full Subtrator

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_9b โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ xor, and2, not, or2, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป



รูปที่ 9.8 วงจรสำหรับการทดลองที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_9b.gdf
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_9b.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 9.9 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

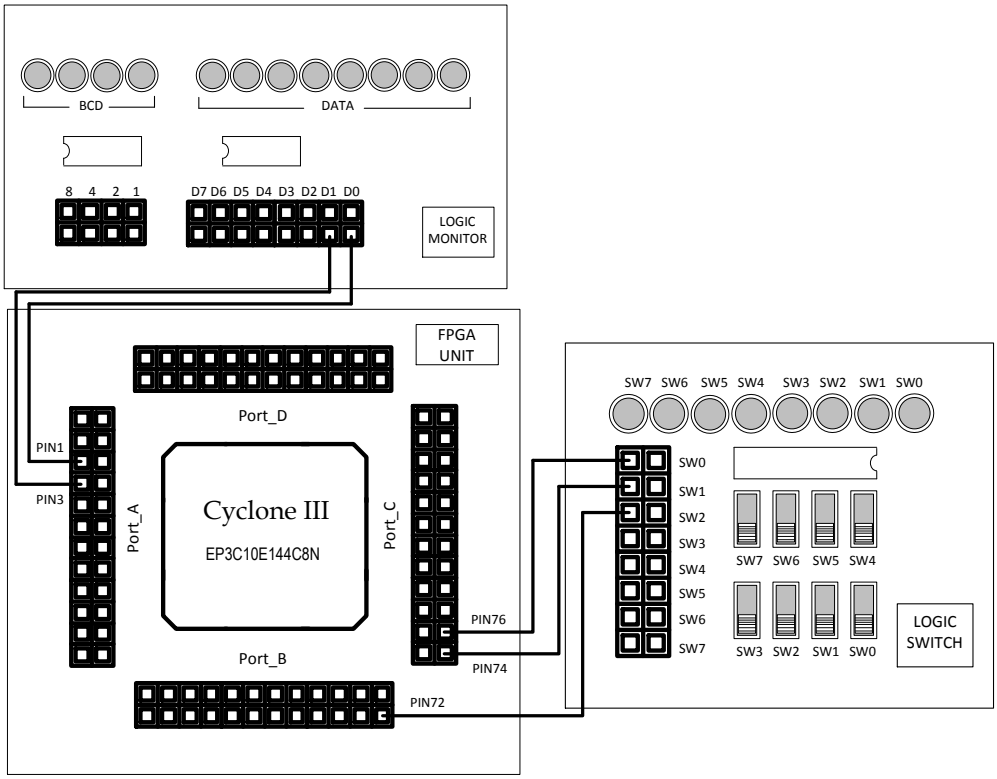
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
A	72
B	74
Bin	76
Diff	3
Bout	1

ตารางที่ 9.6 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 9.10 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 9.10 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1 และ SW2 แล้วดูผลทางลอจิกที่ LED_D0 และ LED_D1 จากนั้นบันทึกผลลงในตาราง

A (SW2)	**B (SW1)**	**Bin (SW0)**	**Diff (LED D1)**	**Bout (LED D0)**
LOW	LOW	LOW		
LOW	LOW	HIGH		
LOW	HIGH	LOW		
LOW	HIGH	HIGH		
HIGH	LOW	LOW		
HIGH	LOW	HIGH		
HIGH	HIGH	LOW		
HIGH	HIGH	HIGH		

ตารางที่ 9.7 ตารางความจริงจากทดลองตอนที่ 2

ขั้นตอนการทดลอง ตอนที่ 3 การใช้งานไอซี 7483 เพื่อสร้างเป็นวงจรลบ

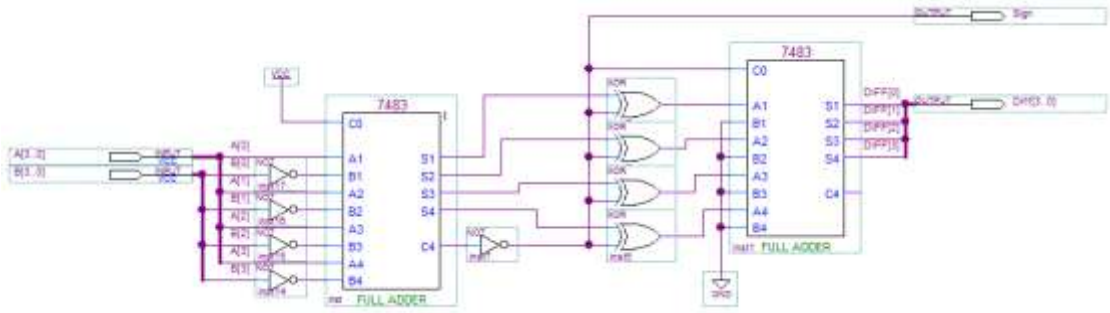
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_9c โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 7483, xor, not, vcc, gnd, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

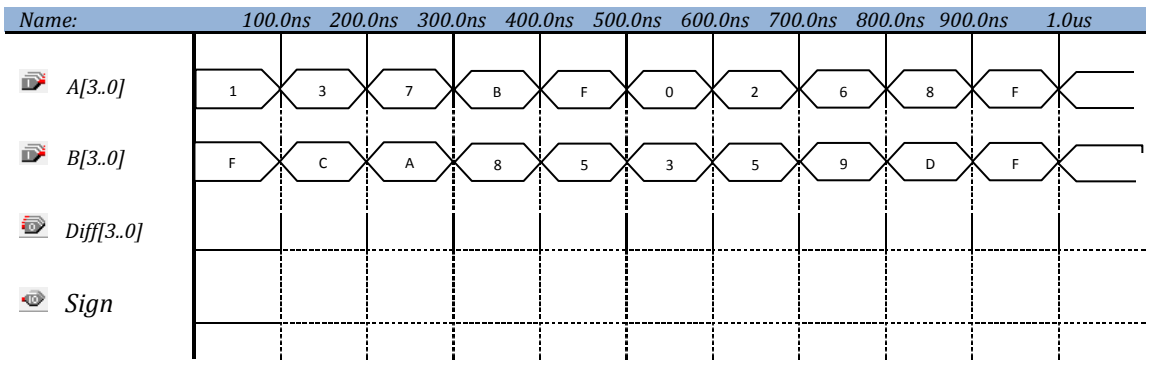


รูปที่ 9.11 วงจรสำหรับการทดลองที่ 3

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_9c.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_9c.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 9.12 ผลการจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

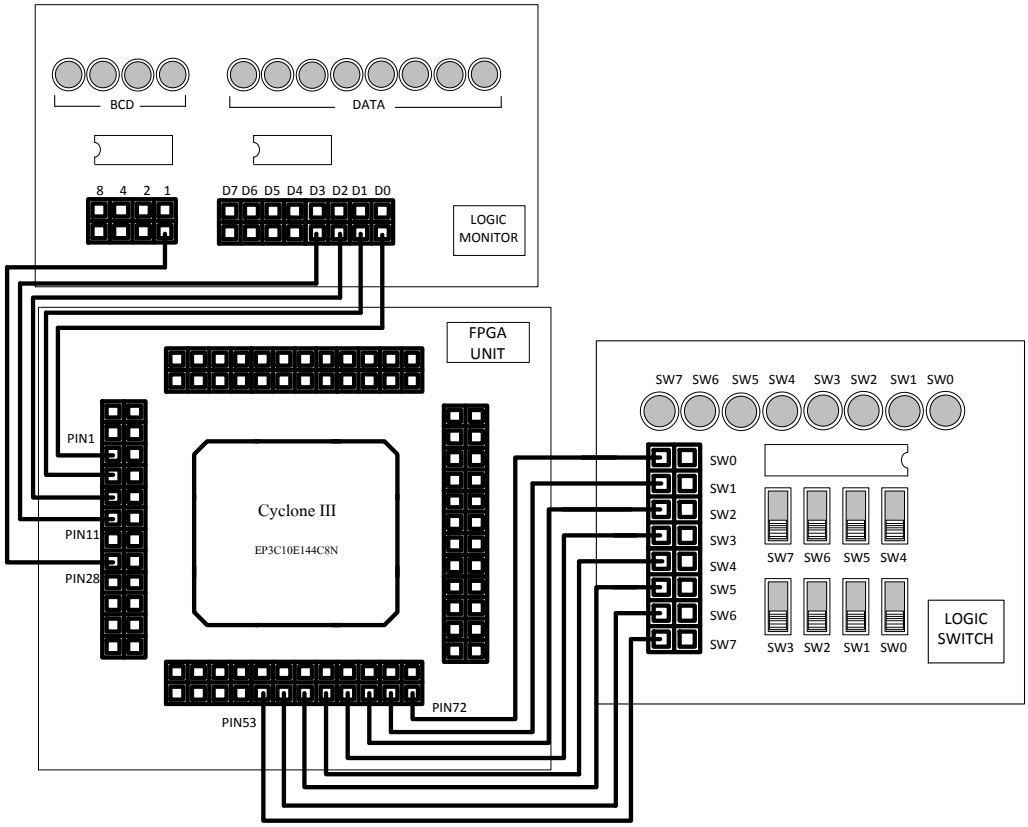
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
A0	72	B3	53
A1	70	Diff0	1
A2	68	Diff1	3
A3	66	Diff2	7
B0	64	Diff3	11
B1	59	Sign	28
B2	55		

ตารางที่ 9.8 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 9.13 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 9.13 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1, SW2, SW3, SW4, SW5, SW6 และ SW7 แล้วดูผลทางลอจิก ที่ LED D0, LED D1, LED D2, LED D3 และ LED 1 จากนั้นบันทึกผลลงในตาราง

B3 (SW7)	B2 (SW6)	B1 (SW5)	B0 (SW4)	A3 (SW3)	A2 (SW2)	A1 (SW1)	A0 (SW0)	Count (LED 1)	Sum3 (LED D3)	Sum2 (LED D2)	Sum1 (LED D1)	Sum0 (LED D0)
1	1	1	1	0	0	0	0					
1	1	0	1	0	0	1	0					
1	0	1	1	0	1	0	0					
1	0	0	1	0	1	1	0					
1	0	0	0	1	0	0	0					
0	1	1	0	1	0	0	1					
0	1	0	0	1	0	1	1					
0	0	1	0	1	1	0	1					
0	0	0	0	1	1	1	1					

ตารางที่ 9.9 ตารางความจริงจากทดลองตอนที่ 3

สรุปผลการทดลอง

การทดลองที่ 10

แลตช์ และ ฟลิปฟลอป (Latch and Flip-flop)

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรแลตช์และฟลิปฟลอป
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรแลตช์และฟลิปฟลอปได้
4. สามารถดาวน์โหลดวงจรไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

ทฤษฎี

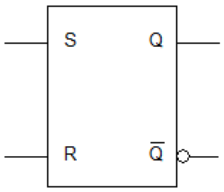
วงจรลอจิกแบ่งออกเป็น 2 ประเภทใหญ่ ๆ ได้แก่ วงจรคอมบิเนชั่น (Combination) และวงจรซีเควนเชียล (Sequential) โดยที่วงจรคอมบิเนชั่นคือวงจรที่สร้างขึ้นมาจากเกทชนิดพื้นฐานต่าง ๆ โดยการประยุกต์ใช้สมการบรูลีน และค่าของระดับลอจิกเอาต์พุทจะขึ้นอยู่กับค่าปัจจุบันของอินพุทเท่านั้น เช่น วงจรบวกเลขฐานสอง ในขนาดที่วงจรซีเคนเชียลจะมีค่าเอาต์พุทขึ้นอยู่กับค่าปัจจุบันของอินพุทและสถานะเดิมของวงจร เช่น วงจรควบคุมลิฟต์ซึ่งในการเลื่อนลิฟต์แต่ละครั้งวงจรจะต้องน้ำสถานะหรือชั้นที่ลิฟต์จอดอยู่มาพิจารณาด้วยว่าจะต้องให้ลิฟต์ขึ้นหรือลงกี่ชั้น

อุปกรณ์พื้นฐานที่นำมาเป็นวงจรซีเควนเชียลคืออุปกรณ์ประเภทหน่วยเก็บข้อมูล ได้แก่ แลตช์ (Latch) และฟลิปฟลอป (Flip-Flop) ซึ่งทั้งสองมีความแตกต่างกันที่แลตช์ใช้ระดับลอจิกในการทำงาน (Level Sensitive) ส่วนฟลิปฟลอปใช้ขอบของสัญญาณในการทำงาน (Edge Sensitive)

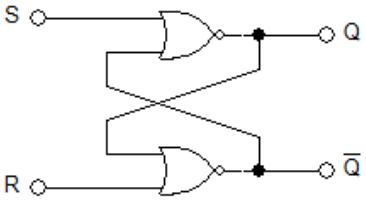
แลตช์ (Latch)

แลตช์ เป็นอุปกรณ์ประเภทไบสเตเบิลมัลติไวเบรเตอร์ (Bistable Multivibrator) ซึ่งหมายถึงวงจรที่มาค่าเอาต์พุทที่เป็นไปได้เพียงสองสถานะเท่านั้น นั่นคือสถานะเซท (SET) และรีเซท(RESET) ซึ่งสภาวะที่เอาต์พุท Q มีค่าเป็น 1 และ Q^ มีค่าเป็น 0 ส่วนสภาวะรีเซท คือสภาวะที่เอาต์พุท Q มีค่าเป็น 0 และ Q มีค่าเป็น 1

ตัวอย่างวงจรแลตช์พื้นฐานได้แก่ แลตช์แบบ S-R (Set-Reset) ซึ่งมีสัญลักษณ์ดังรูปที่ 10.1a และรูปที่ 10.1b เป็นวงจรแลตช์แบบ S-R ที่สร้างจากนอร์เกต การทำงานของมันจะให้ค่าเอาต์พุทขึ้นกับอินพุทขา R และ S ดังตารางที่ 10.1



รูปที่ 10.1a สัญลักษณ์ของแลตช์แบบ S-R ชนิดแอคทีฟไฮ

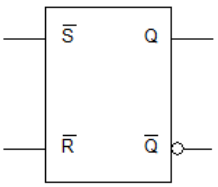


รูปที่ 10.1b วงจรของแลตช์แบบ S-R ชนิดแอคทีฟไฮ

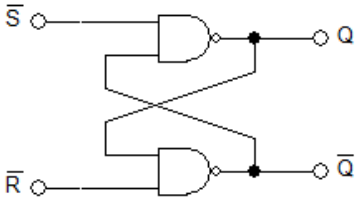
อินพุท		เอาต์พุท		โหมดการทำงาน
S	R	Q_{t+1}	$\overline{Q}_{t+1}$	
0	0	Q_t	$\overline{Q}_t$	ไม่เปลี่ยนแปลง
0	1	0	1	รีเซท
1	0	1	0	เซท
1	1	1	1	ห้ามใช้งาน

ตารางที่ 10.1 ตารางความจริงของแลตช์แบบ S-R ชนิดแอคทีฟไฮชนิดแอคทีฟไฮ

วงจรและสัญลักษณ์ในรูปที่ 10.1a และรูปที่ 10.1b เป็นแลตช์เซท – รีเซทแบบแอคทีฟไฮ (Active High) ส่วนแลตช์แบบ เซท-รีเซทชนิดแอคทีฟโลว์จะสร้างขึ้นจากแนนด์เกทดังรูปที่ 10.2a, 10.2b และตารางที่ 10.2



รูปที่ 10.2a สัญลักษณ์ของแลตช์แบบ S-R ชนิดแอคทีฟโลว์



รูปที่ 10.2b วงจรของแลตช์แบบ S-R ชนิดแอคทีฟโลว์

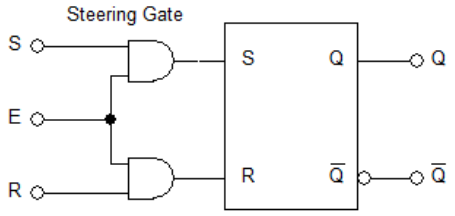
อินพุท		เอาต์พุท		โหมดการทำงาน
$\overline{S}$	$\overline{R}$	Q_{t+1}	$\overline{Q}_{t+1}$	
0	0	1	1	ห้ามใช้
0	1	0	1	เซท
1	0	1	0	รีเซท
1	1	Q_t	$\overline{Q}_t$	ไม่เปลี่ยนแปลง

ตารางที่ 10.2 ตารางความจริงของแลตช์แบบ S-R ชนิดแอคทีฟไฮชนิดแอคทีฟโลว์

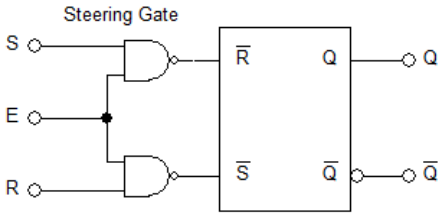
แลตช์ชนิด Gated S-R

แลตช์ชนิด Gated S-R เป็นแลต์ที่การเพิ่มชุดลอจิกเกทซึ่งเรียกว่า Steering Gate เข้าไปในวงจรแลตช์ S-R เพื่อใช้สำหรับเปิดปิด การทำงานของแลตช์ S-R ซึ่งตามปกติเมื่ออินพุทของแลตช์ S-R มีการเปลี่ยนแปลงจะส่งผลให้เอาต์พุทมีการเปลี่ยนแปลงทันที แต่สำหรับทำงานของแลตช์ชนิด R-S นั่นจะแตกต่างกัน โดยที่ถ้าหากอินพุทมีการเปลี่ยนแปลง สถานของแลตช์จะยังไม่เปลี่ยนแปลงทันที แต่จะต้องรอสัญญาณควบคุมอีกสัญญาณหนึ่งก่อนเพื่ออนุญาติให้แลตช์เปลี่ยนสภาวะได้ ซึ่งสัญญาณควบคุมนี้จะเป็นตัวที่ไปเปิดการทำงานของลอจิกเกทที่เพิ่มเข้าไปนั่นเอง

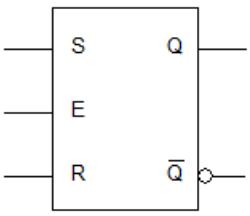
รูปที่ 10.3a เป็นแลตช์ชนิด Gated S-R ที่สร้างจากแลตช์ S-R แบบแอคทีฟไฮ และรูปที่ 10.3b เป็นแลตช์ชนิด Gated S-R ที่สร้างจากแลตช์ S-R แบบแอคทีฟโลว์ ซึ่งทั้งสองวงจรให้ผลลัพธ์ที่เหมือนกัน และจะทำงานต่อเมื่อ สัญญาณที่ขา E มีค่าเป็นลอจิก 1 เท่านั้น



รูปที่ 10.3a การสร้างแลตช์ Gated S-R จากแลตช์ชนิดแอคทีฟไฮ



รูปที่ 10.3b การสร้างแลตช์ Gated S-R จากแลตช์ชนิดแอคทีฟโลว์



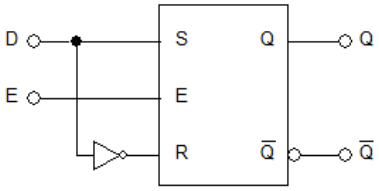
รูปที่ 10.4 สัญลักษณ์ของแลตช์แบบ Gated S-R

อินพุท			เอาต์พุท		โหมดการทำงาน
E	$\overline{\mathbf{S}}$	$\overline{\mathbf{R}}$	$\mathbf{Q_{t+1}}$	$\overline{\mathbf{Q}}_{\mathbf{t+1}}$	
1	0	0	Q_t	$\overline{Q}_t$	ไม่เปลี่ยนแปลง
1	0	1	0	1	รีเซท
1	1	0	1	0	เซท
1	1	1	1	1	ห้ามใช้
0	X	X	Q_t	$\overline{Q}_t$	หยุดการทำงาน

ตารางที่ 10.3 ตารางความจริงของแลตช์แบบ Gated S-R

แลตช์ชนิด Gated D

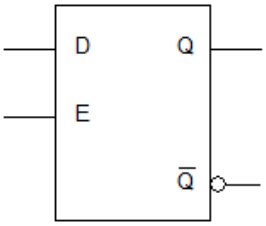
แลตช์ชนิด Gated D หรือมีชื่ออีกชื่อหนึ่งว่า Transparent Latch เป็นวงจรที่ถูกพัฒนามาจาดแลตช์ชนิด Gated S-R เพื่อใช้เป็นอุปกรณ์พื้นฐานสำหรับเก็บข้อมูล การทำงานของมันจะมี 2 โหมด นั่นคือเมื่อสัญญาณ Enable เป็นลอจิก 1 แลตช์จะมีสภาพเหมือนไม่มีตัวตน (Transparent) หรือก็คือสภาพที่เอาต์พุทมีค่าลอจิกเหมือนกับทางอินพุทและเมื่อสัญญาณ Enable มีค่าลอจิกเป็น 0 แลตช์จะอยู่ในสภาวะเก็บข้อมูลที่ปรากฏที่ขา D ในช่วงที่ขา Enable มีลอจิกเป็น 1 ครั้งล่าสุด ซึ่งโครงสร้างสัญลักษณ์และ ตารางความจริงของแลตช์ชนิด Gated D แสดงดังรูป 10.5a, 10.5b และตารางที่ 10.4



รูปที่ 10.5a การสร้างแลตช์ Gated D จากแลตช์ Gated S-R

อินพุท		เอาต์พุท		โหมดการทำงาน	หมายเหตุ
E	D	Q_{t+1}	$\overline{Q}_{t+1}$		
0	X	Q_t	$\overline{Q}_t$	ไม่เปลี่ยนแปลง	เก็บข้อมูล
0	1	0	1	รีเซท	ไม่มีตัวตน (Transparent)
1	0	1	0	เซท	

ตารางที่ 10.4 ตารางความจริงของแลตช์ Gated D



รูปที่ 10.5b สัญลักษณ์ของแลตช์ Gated D

ฟลิปฟลอป

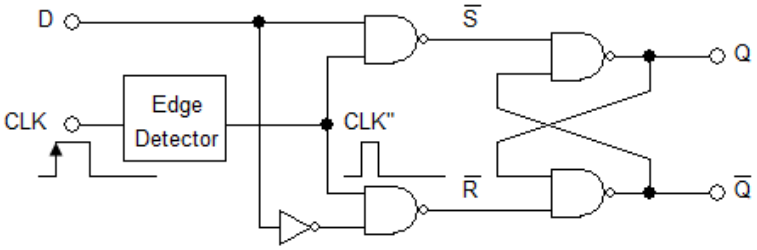
ฟลิปฟลอปเป็นอุปกรณ์ที่ทำการเปลี่ยนแปลงเอ้าท์พุทที่ขอบของสัญญาณอินพุทพิเศษที่เรียกว่า สัญญาณนาฬิกา ซึ่งอาจทำงานที่ขอบขาขึ้นและขอบขาลงของสัญญาณนาฬิกาก็ได้ โดยสังเกตที่สัญลักษณ์หากมีวงกลมที่ขาสัญญาณนาฬิกาแสดงว่าฟลิบฟลอบตัวนั้นทำงานที่ขอบขาลง และถ้าสัญลักษณ์ของสัญญาณนาฬิกาไม่มีวงกลมแสดงว่าฟลิปฟลอปตัวนั้นทำงานที่ขอบขาขึ้นของสัญญาณนาฬิกา

เราสามารถกล่าวได้ว่าแลตช์ที่มีสัญญาณนาฬิกาด้านอินพุทก็คือ ฟลิปฟลอปนั้นเอง แต่มันจะต่างกันที่แลตช์ใช้สัญญาณ Enable ในการทำงาน ซึ่งมันจะทำงานที่สภาวะต่าง ๆ ตลอดเวลาที่

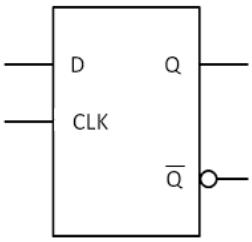
สัญญาณ Enable คงค่าอยู่ใน ลอจิกระดับใดระดับหนึ่ง แต่สำหรับฟลิปฟลอปแล้ว มันจะทำงานเฉพาะในกรณีที่สัญญาณนาฬิกามีการเปลี่ยนแปลงเท่านั้น

ฟลิปฟลอปแบบ D

ฟลิปฟลอปแบบ D มีลักษณะการทำงานคล้ายกับแลตช์แบบ Gated D แต่เอาพุตจะมีการเปลี่ยนแปลงในกรณีสัญญาณนาฬิกาเปลี่ยนแปลงเท่านั้นอาจเป็นขาขึ้นหรือขาลงขึ้นอยู่กับชนิดของการกระตุ้น ฟลิปฟลอป



รูปที่ 10.6 วงจรฟลิปฟลอป D



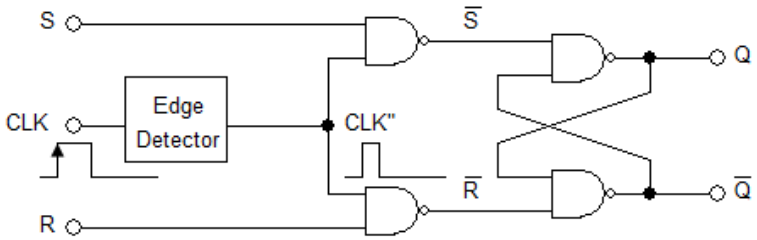
รูปที่ 10.7 สัญลักษณ์ของฟลิปฟลอป D

อินพุท		เอาต์พุท		โหมดการทำงาน
CLK	D	Q_{t+1}	$\overline{Q}_{t+1}$	
↑	0	0	1	รีเซท
↑	1	1	0	เซท
0	X	Q_t	$\overline{Q}_t$	หยุดการทำงาน
1	X	Q_t	$\overline{Q}_t$	หยุดการทำงาน
↓	X	Q_t	$\overline{Q}_t$	หยุดการทำงาน

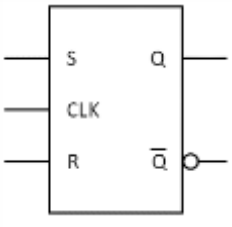
ตารางที่ 10.5 ตารางความจริงของฟลิปฟลอป D

ฟลิปฟลอปแบบ S-R

แลตช์ S-R ที่ใช้แนนด์เกทและแลตช์ S-R ที่ใช้นอร์เกทบ้างครั้งถูกเรียกว่า ฟลิปฟลอป S-R ซึ่งถือว่าเป็นสิ่งที่ผิดเมื่ออ้างอิงจากนิยามของฟลิปฟลอปที่กล่าวในที่นี้ ทั้งนี้เนื่องจากทั้ง 2 วงจร ไม่มีขาสัญญาณนาฬิกานั้นเอง ในความเป็นจริงแล้วฟลิปฟลอป S-R จะคล้ายกับแลตช์ Gated S-R ที่ใช้สัญญาณนาฬิกาและวงจรตรวจจับขอบสัญญาณนาฬิกาเพิ่มเข้ามาแทนสัญญาณ Enable



รูปที่ 10.8 วงจรฟลิปฟลอป



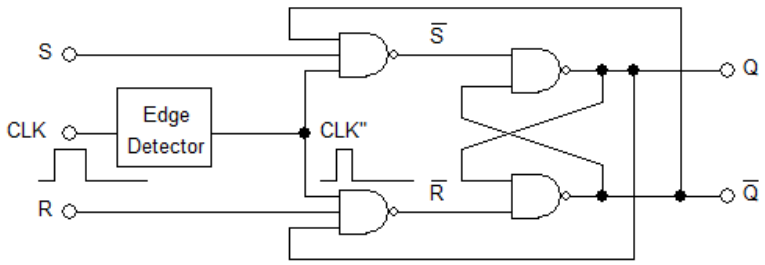
รูปที่ 10.9 สัญลักษณ์ของฟลิปฟลอป S-R

อินพุท			เอาต์พุท		โหมดการทำงาน
CLK	S	R	Q_{t+1}	$\overline{Q_{t+1}}$	
↑	0	0	Q_t	$\overline{Q_t}$	ไม่เปลี่ยนแปลง
↑	0	1	0	1	รีเซท
↑	1	0	1	0	เซท
↑	1	1	1	1	ห้ามใช้งาน

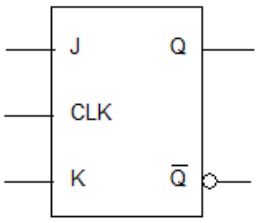
ตารางที่ 10.6 ตารางความจริงของฟลิปฟลอป S-R

ฟลิปฟลอป J-K

ฟลิปฟลอป J-K เป็นฟลิปฟลอปที่พัฒนาขึ้นจากฟลิปฟลอป S-R แต่จะมีการแก้ไขในกรณีที่ S และ R เป็นลอจิก1 ทั้งคู่ ซึ่งเป็นเงื่อนไขที่ห้ามใช้ของฟลิปฟลอป S-R โดยในฟลิปฟลอป J-K จะให้ผลลอจิกเป็นการกลับลอจิกเอาต์พุทให้เป็นตรงกันข้ามซึ่งเราเรียกว่า ท็อกเกิล (Toggle)



รูปที่ 10.10 วงจรฟลิปฟลอป J-K



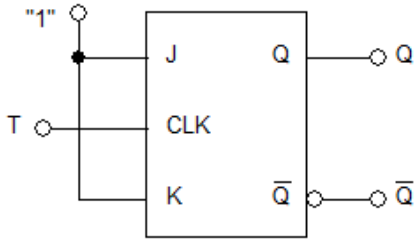
รูปที่ 10.11 สัญลักษณ์ของฟลิปฟลอป J-K

อินพุท			เอาต์พุท		โหมดการทำงาน
CLK	J	K	Q_{t+1}	$\overline{Q_{t+1}}$	
↑	0	0	Q_t	$\overline{Q_t}$	ไม่เปลี่ยนแปลง
↑	0	1	0	1	รีเซท
↑	1	0	1	0	เซท
↑	1	1	$\overline{Q_t}$	Q_t	ท็อกเกิล

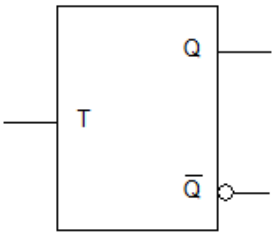
ตารางที่ 10.7 ตารางของของฟลิปฟลอป J-K

ฟลิปฟลอป T

ฟลิปฟลอป T ซึ่ง T ย่อมาจาก Toggle เป็นฟลิปฟลอปที่สร้างจากฟลิปฟลอป J-K โดยใช้คุณสมบัติท็อกเกิลที่เกิดขึ้นในฟลิปฟลอป J-K เท่านั่น ฟลิปฟลอป T จะมีอินพุทเพียงขาเดียวเท่านั่น โดยเมื่อขา T ถูกกระตุ่นด้วยสญญาณนาฬิกา มันจะทำการกลับค่าเอาต์พุทให้มีลอจิกเป็นตรงกันข้าม



รูปที่ 10.12a การสร้างฟลิปฟลอป T จากฟลิปฟลอป J-K



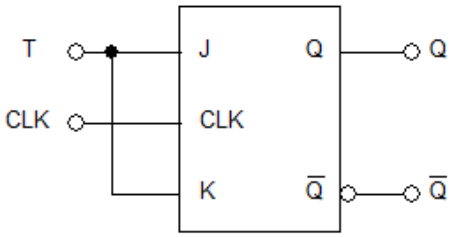
รูปที่ 10.12b สัญลักษณ์ของฟลิปฟลอป T

อินพุท	เอาต์พุท		โหมดการทำงาน
T	Q_{t+1}	$\overline{Q_{t+1}}$	
↑	$\overline{Q_t}$	Q_t	ท็อกเกิล
↑	Q_t	$\overline{Q_t}$	ท็อกเกิล

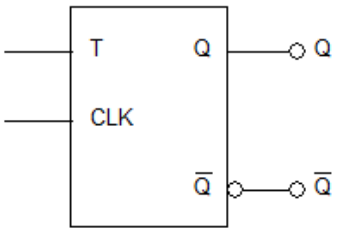
ตารางที่ 10.8 ตารางความจริงของฟลิปฟลอป T

ฟลิปฟลอป T แบบมีสัญญาณนาฬิกา

ฟลิปฟลอป T ชนิดนี้เป็นฟลิปฟลอป T ที่ถูกควบคุมด้วยสัญญาณนาฬิกา โดยฟลิปฟลอปจะทำการท็อกเกิลเอาต์พุทในกรณีที่อินพุท T มีลอจิกเป็น 1 และมีสัญญาณนาฬิกาเข้ามากระตุ้นที่ขา CLK แต่ในกรณีที่ค่าลอจิกที่ขา T มีค่าลอจิก 0 นั่นแม้ว่าจะมีสัญญาณนาฬิกาเข้ามากระตุ้นก็ตาม ค่าระดับลอจิกทางเอาต์พุทก็ยังมีค่าไม่เปลี่ยนแปลง



รูปที่ 10.13a การสร้างฟลิปฟลอป T แบบมีสัญญาณนาฬิกาจากฟลิปฟลอป J-K



รูปที่ 10.13b สัญลักษณ์ของ T แบบมีสัญญาณนาฬิกา

อินพุท		เอาต์พุท		โหมดการทำงาน
CLK	T	Q_{t+1}	$\overline{Q}_{t+1}$	
↑	0	Q_t	$\overline{Q}_t$	ไม่เปลี่ยนแปลง
↑	0	Q_t	$\overline{Q}_t$	ไม่เปลี่ยนแปลง
↑	1	$\overline{Q}_t$	Q_t	ท็อกเกิล
↑	1	$\overline{Q}_t$	Q_t	ท็อกเกิล

ตารางที่ 10.9 ตารางความจริงของฟลิปฟลอป

อินพุทแบบอะซิงโครนัส (Asynchronous Input)

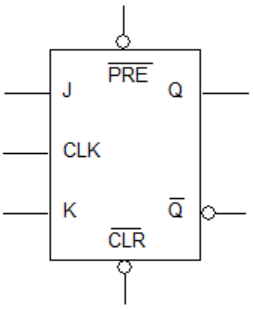
อินพุทของฟลิปฟลอปแบ่งเป็น 2 ชนิด ได้แก่ อินพุทแบบซิงโครนัส (Synchronous Input) และอินพุทแบบอะซิงโครนัส (Asynchronous Input) ซึ่งอินพุทแบบซิงโครนัสคืออินพุทที่ไม่สามารถเปลี่ยนแปลงค่าลอจิกทางเอาต์พุทได้โดยตรง แต่จะต้องรอให้มีสัญญาณนาฬิกาป้อนเข้ามาก่อน เช่น ขาอินพุท D, J และ K เป็นต้น

อินพุทแบบอะซิงโครนัส คืออินพุทของพลิปฟลอปที่สามารถเปลี่ยนแปลงค่าลอจิกทางเอาต์พุทได้ทันทีโดยไม่จำเป็นต้องรอสัญญาณนาฬิกาซึ่งขาเหล่านี่ได้แก่ ขา Preset และขา Clear ซึ่งจะได้กล่าวถึงต่อไป

ขา Preset คือขาอินพุทแบบอะซิงโครนัสที่มีหน้าที่ในการเซทเอาต์พุทของฟลิปฟลอปให้

อยู่ในภาวะเซต (SET) แบบทันทีทันใดโดยไม่ต้องรอการป้อนสัญญาณนาฬิกา ส่วนขา Clear จะมีหน้าที่ในการรีเซท (RESET) สภาวะเอาต์พุทของฟลิปฟลอบแบบทันทีทันใดโดยไม่ต้องรอสัญญาณนาฬิกาเช่นกัน

ขา Clear มีทั้งแบบแอคทีฟไฮและแอคทีฟโลว์ ซึ่งจะทำให้ผลการทำงานที่เหมือนกัน แต่แตกต่างกันเพียงระดับลอจิกที่ป้อนให้โดยถ้าเป็นขาแบบแอคทีฟไฮ เราจะต้องป้อนลอจิก 1 เพื่อให้มันทำงาน ส่วนถ้าเป็นขาแบบแอคทีฟโลว์ เราจะต้องป้อนลอจิก 0 เพื่อให้มันทำงาน



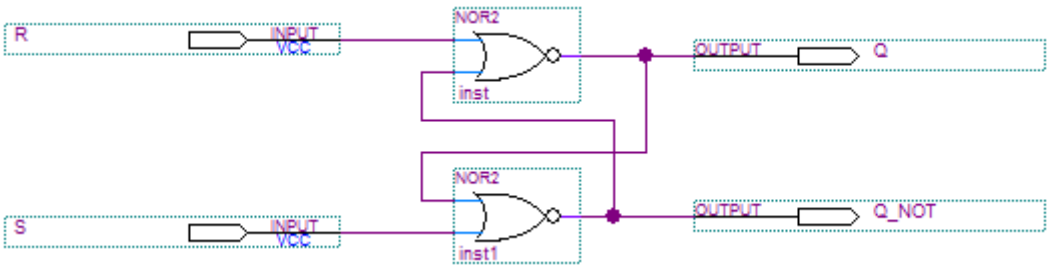
รูปที่ 10.14 สัญลักษณ์ของฟลิปฟลอป

อินพุท		โหมดการทำงาน
$\overline{PRE}$	$\overline{CLR}$	
1	1	ทำงานตามโหมดสัญญาณนาฬิกาปกติ
0	1	$Q_t = 1$ (SET โดยไม่สนใจสัญญาณ CLK)
1	0	$Q_t = 0$ (RESET โดยไม่สนใจสัญญาณ CLK)
0	0	ไม่มีการใช้งานในกรณีนี้

ตารางที่ 10.10 ตารางความจริงของฟลิปฟลอป J-K ที่มีขาอินพุทแบบอะซิงโครนัสในโหมดแอคทีฟโลว์

ขั้นตอนการทดลอง ตอนที่ 1 แลตช์ S-R แบบแอคทีฟไฮ

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_10a โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ nor2, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

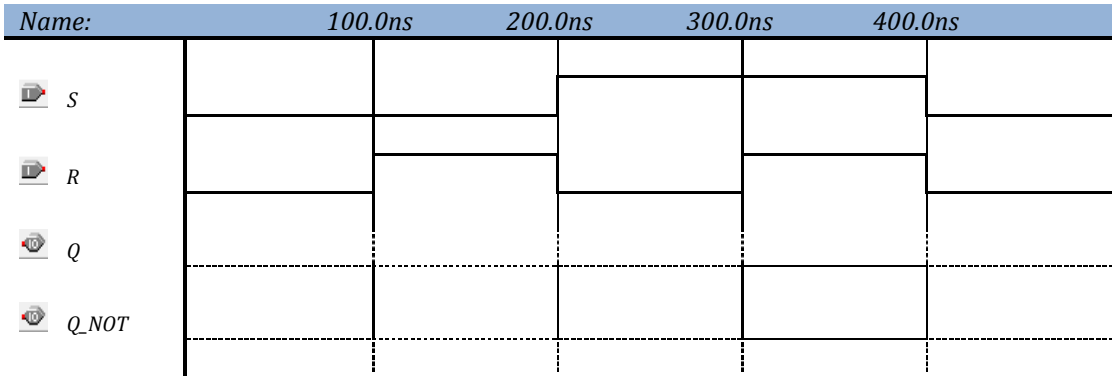


รูปที่ 10.15 วงจรสำหรับการทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_10a.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_10a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 10.16 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

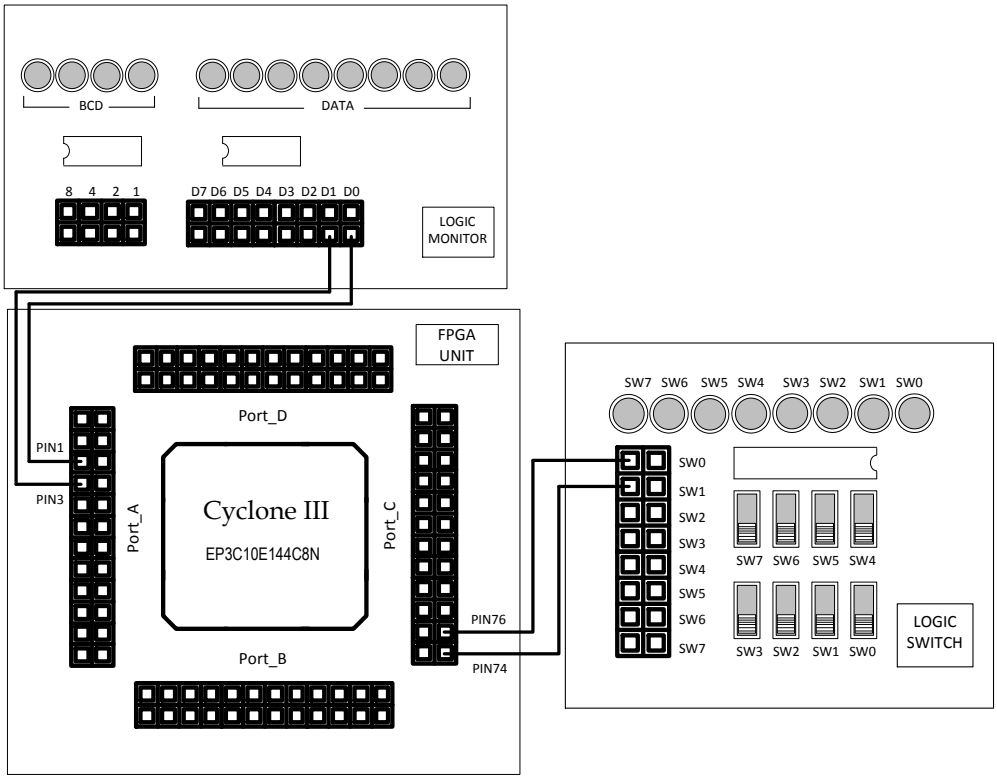
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
S	74
R	76
Q	3
Q_NOT	1

ตารางที่ 10.11 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 10.17 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 10.17 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0 และ SW1แล้วดูผลทางลอจิก ที่ LED D0 และ LED D1 จากนั้นบันทึกผลลงในตาราง

S (SW1)	**R (SW0)**	**Q (LED D1)**	**Q_NOT (LED D0)**
LOW	LOW		
LOW	HIGH		
HIGH	LOW		
HIGH	HIGH		

ตารางที่ 10.12 ตารางความจริงจากการทดลองตอนที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 แลตช์ S-R แบบแอคทีฟโลว์

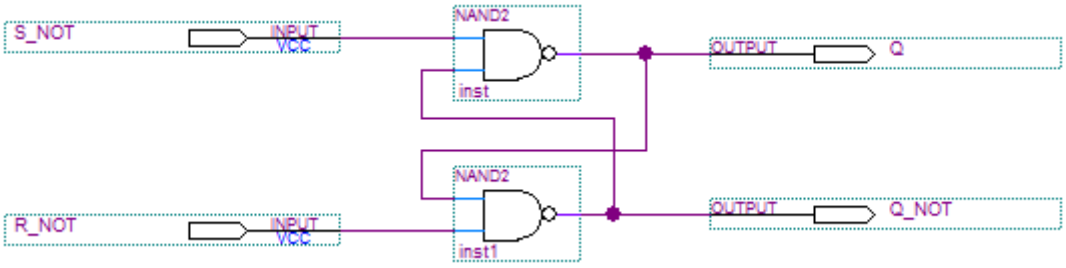
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_10b โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ nand2, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

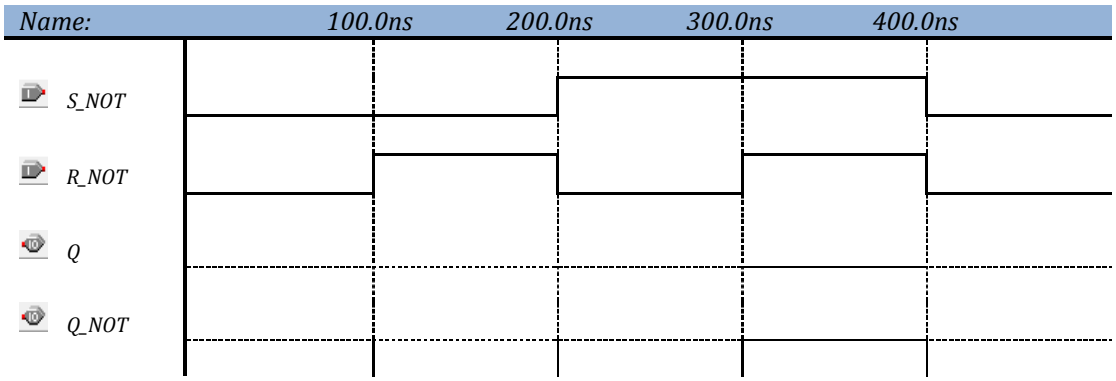


รูปที่ 10.18 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_10b.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_10b.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 10.19 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

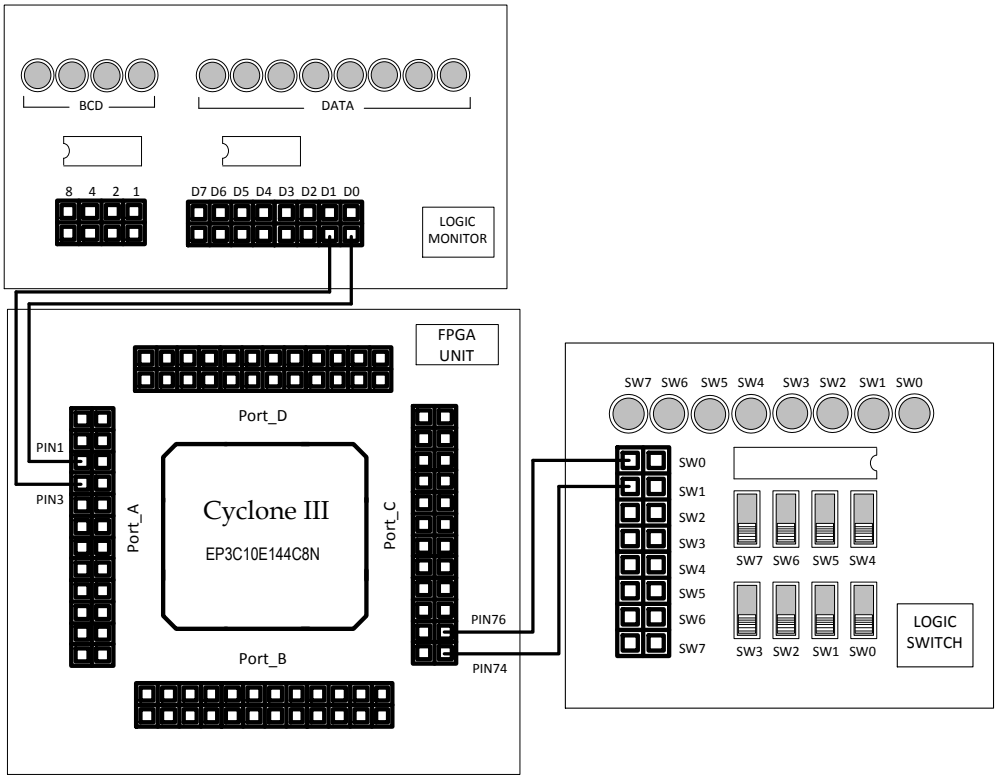
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
S_NOT	74
R_NOT	76
Q	3
Q_NOT	1

ตารางที่ 10.13 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 10.20 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 10.20 การต่อวงจรบนชุดทดลอง

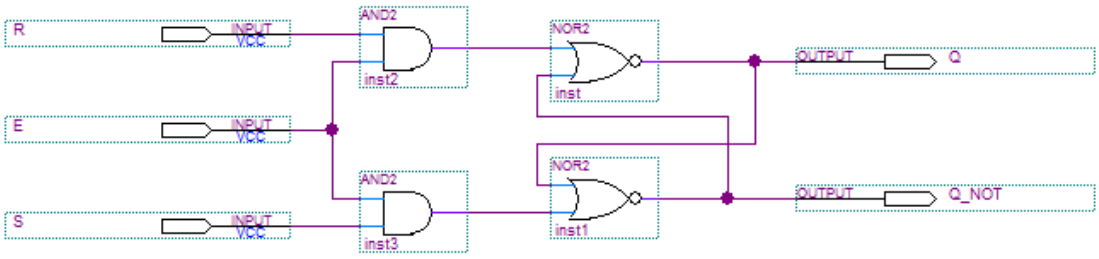
13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0 และ SW1แล้วดูผลทางลอจิก ที่ LED D0 และ LED D1 จากนั้นบันทึกผลลงในตาราง

S_NOT (SW1)	R_NOT (SW0)	Q (LED D1)	Q_NOT (LED D0)
LOW	LOW		
LOW	HIGH		
HIGH	LOW		
HIGH	HIGH		

ตารางที่ 10.14 ตารางความจริงจากการทดลองตอนที่ 2

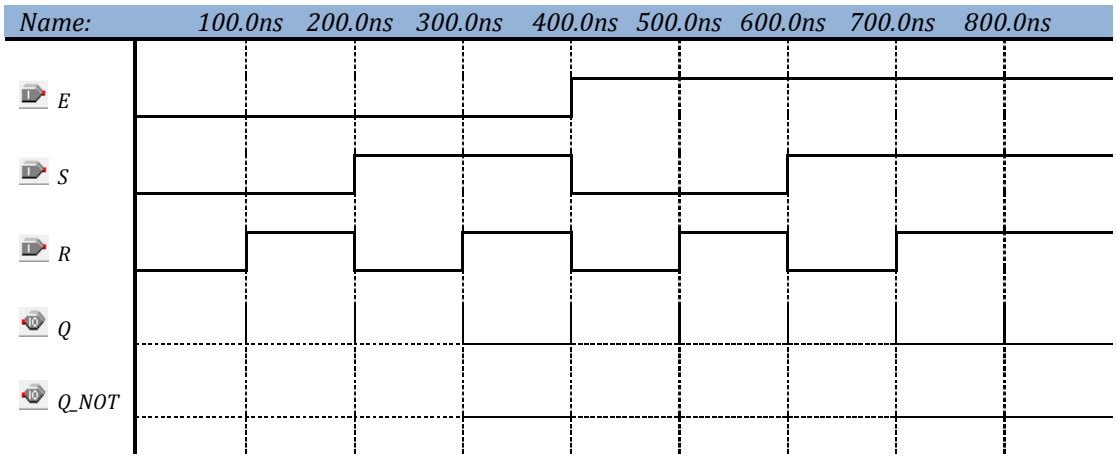
ขั้นตอนการทดลอง ตอนที่ 3 แลตช์ S-R แบบแอคทีฟโลว์

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_10b โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ nor2, and2, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป



รูปที่ 10.21 วงจรสำหรับการทดลองตอนที่ 3

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_10c.gdf
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_10c.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 10.22 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

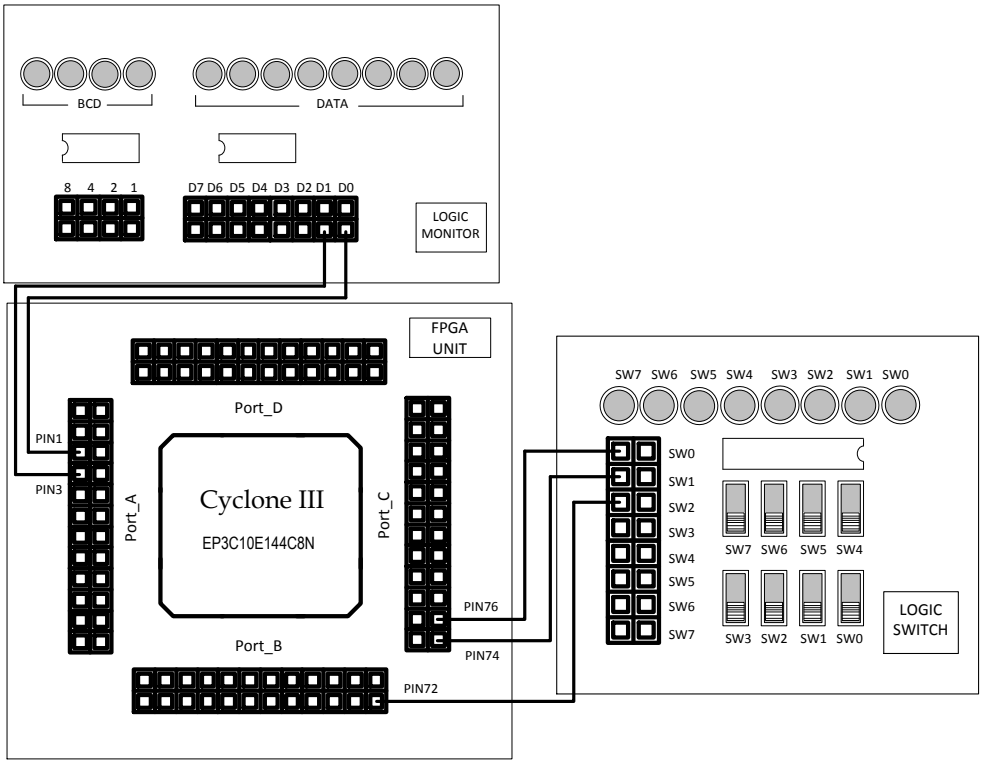
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
S	74
R	76
E	72
Q	3
Q_NOT	1

ตารางที่ 10.15 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 10.23 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 10.23 การต่อวงจรบนชุดทดลอง

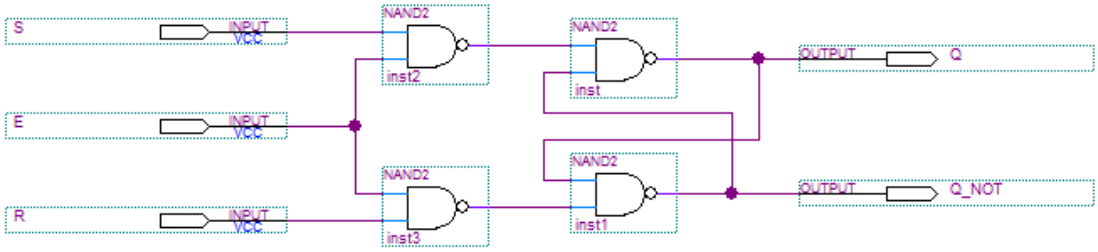
13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1 และ SW2 แล้วดูผลทางลอจิก ที่ LED D0 และ LED D1 จากนั้นบันทึกผลลงในตาราง

E (SW2)	**S (SW1)**	**R (SW0)**	**Q (LED D1)**	**Q_NOT (LED D0)**
LOW	LOW	LOW		
LOW	LOW	HIGH		
LOW	HIGH	LOW		
LOW	HIGH	HIGH		
HIGH	LOW	LOW		
HIGH	LOW	HIGH		
HIGH	HIGH	LOW		
HIGH	HIGH	HIGH		

ตารางที่ 10.16 ตารางความจริงจากการทดลองตอนที่ 3

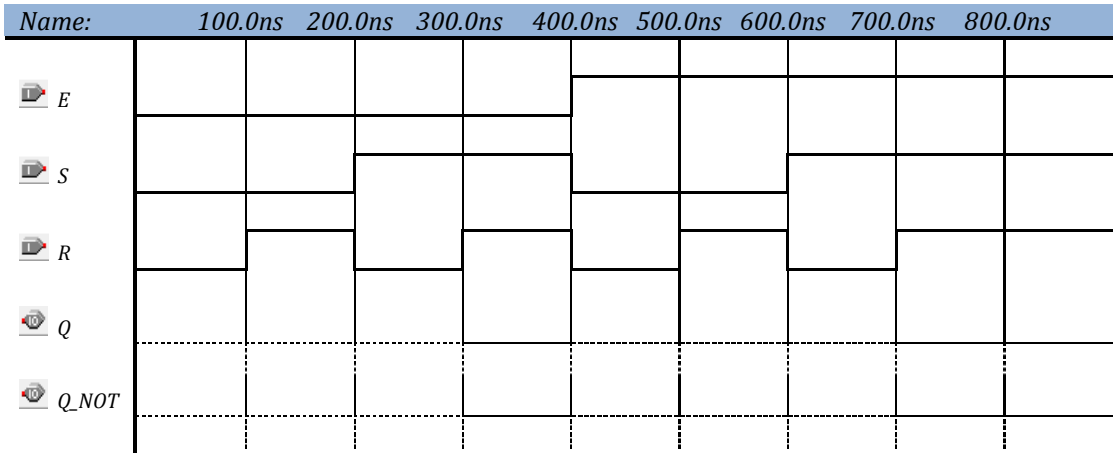
ขั้นตอนการทดลอง ตอนที่ 4 แลตช์แบบ Gated S-R ที่สร้างจากแลตช์ S-R ชนิดแอคทีฟโลว์

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_10d โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ nand2, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป



รูปที่ 10.24 วงจรสำหรับการทดลองตอนที่ 4

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_10d.gdf
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_10d.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 10.25 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

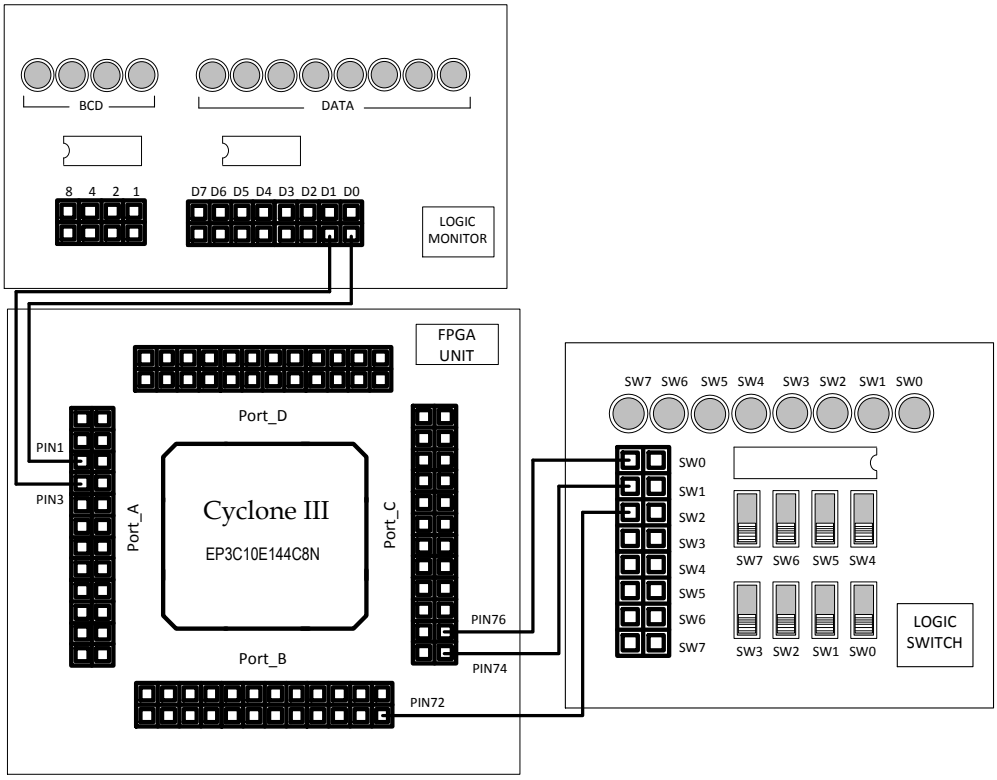
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
S	74
R	76
E	72
Q	3
Q_NOT	1

ตารางที่ 10.17 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 10.26 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 10.26 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1 และ SW2 แล้วดูผลทางลอจิกที่ LED D0 และ LED D1 จากนั้นบันทึกผลลงในตาราง

E (SW2)	S (SW1)	R (SW0)	Q (LED D1)	Q_NOT (LED D0)
LOW	LOW	LOW		
LOW	LOW	HIGH		
LOW	HIGH	LOW		
LOW	HIGH	HIGH		
HIGH	LOW	LOW		
HIGH	LOW	HIGH		
HIGH	HIGH	LOW		
HIGH	HIGH	HIGH		

ตารางที่ 10.18 ตารางความจริงจากการทดลองตอนที่ 4

ขั้นตอนการทดลอง ตอนที่ 5 แลตช์แบบ Gated D

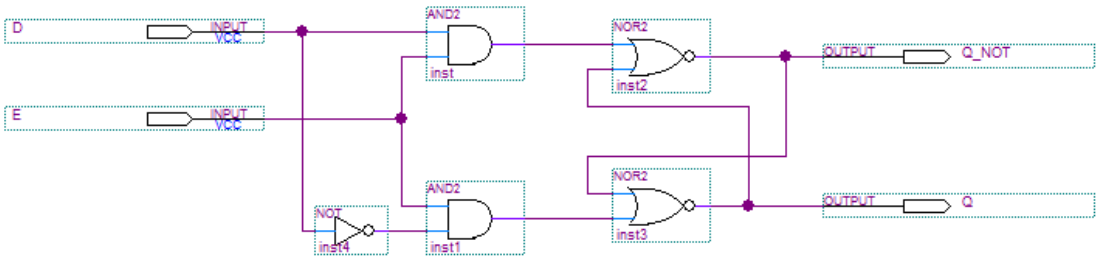
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_10e โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ nor2, and2, not, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

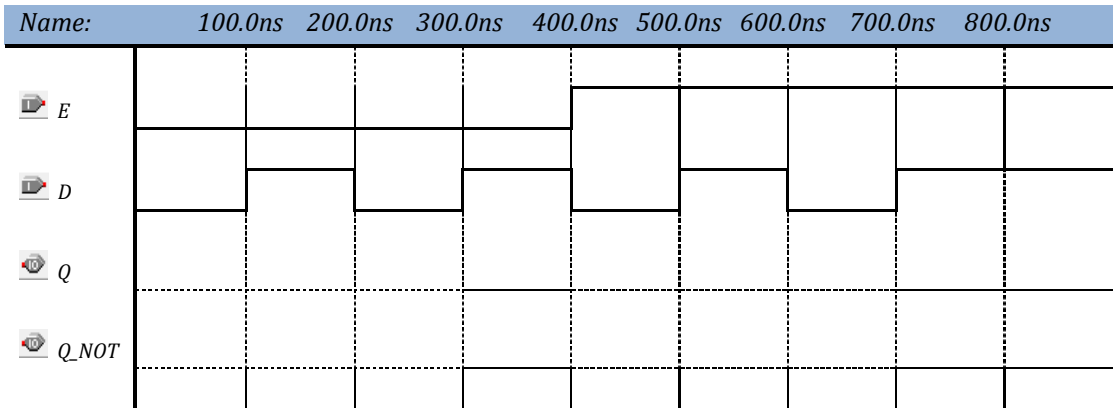


รูปที่ 10.27 วงจรสำหรับการทดลองตอนที่ 5

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_10e.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_10e.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 10.28 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

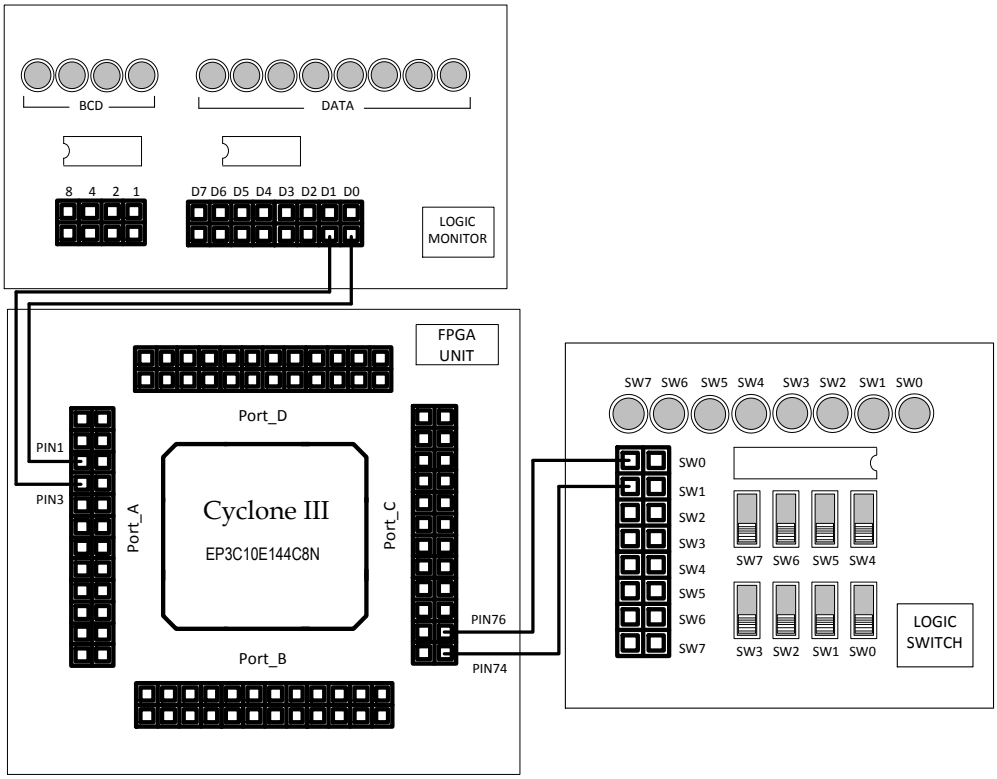
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
D	76
E	74
Q	3
Q_NOT	1

ตารางที่ 10.19 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 10.29 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 10.29 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิก ที่ LED D0 และ LED D1 จากนั้นบันทึกผลลงในตาราง

E (SW1)	**D (SW0)**	**Q (LED D1)**	**Q_NOT (LED D0)**
LOW	LOW		
LOW	HIGH		
HIGH	LOW		
HIGH	HIGH		

ตารางที่ 10.20 ตารางความจริงจากการทดลองตอนที่ 5

ขั้นตอนการทดลอง ตอนที่ 6 แลตช์แบบ Gated D และฟลิปฟลอป D

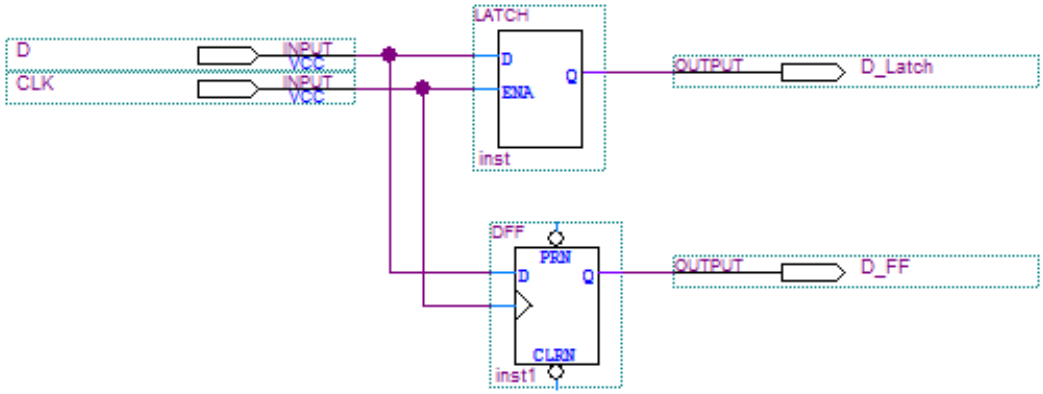
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_10f โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ latch, dff, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

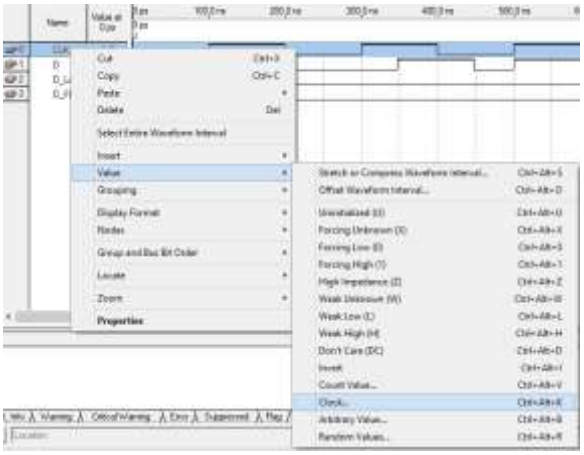


รูปที่ 10.30 วงจรสำหรับการทดลองตอนที่ 6

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_10f.gdf

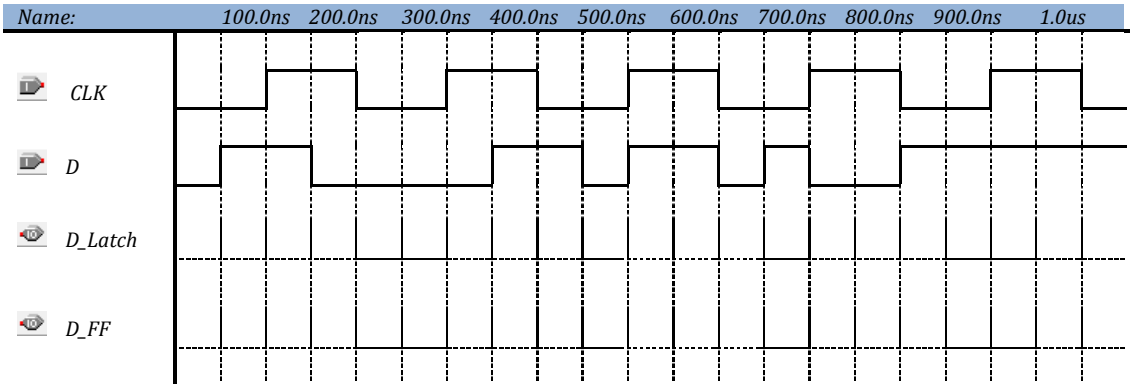
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_10f.scf เพื่อจำลองการทำงานของวงจร โดยกำหนด Grid Size เป็น 50 ns และในส่วนของสัญญาณ CLK กำหนดให้เป็นแบบ Clock ซึ่งทำได้โดยการคลิกขวาที่ชื่อสัญญาณ จากนั้นจะมีเมนู Pop-up ปรากฏขึ้น ให้เลือกที่เมนู Value -> Clock… แล้วกำหนดค่า Period เป็น 200 ns ดังรูปที่ 10.31b จากนั้นจำลองการทำงานและบันทึกผลการทดลอง



รูปที่ 10.31a คลิกขวาที่สัญญาณเพื่อเรียกเมนู Pop-up

รูปที่ 10.31b กำหนดคาบของสัญญาณนาฬิกา



รูปที่ 10.32 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

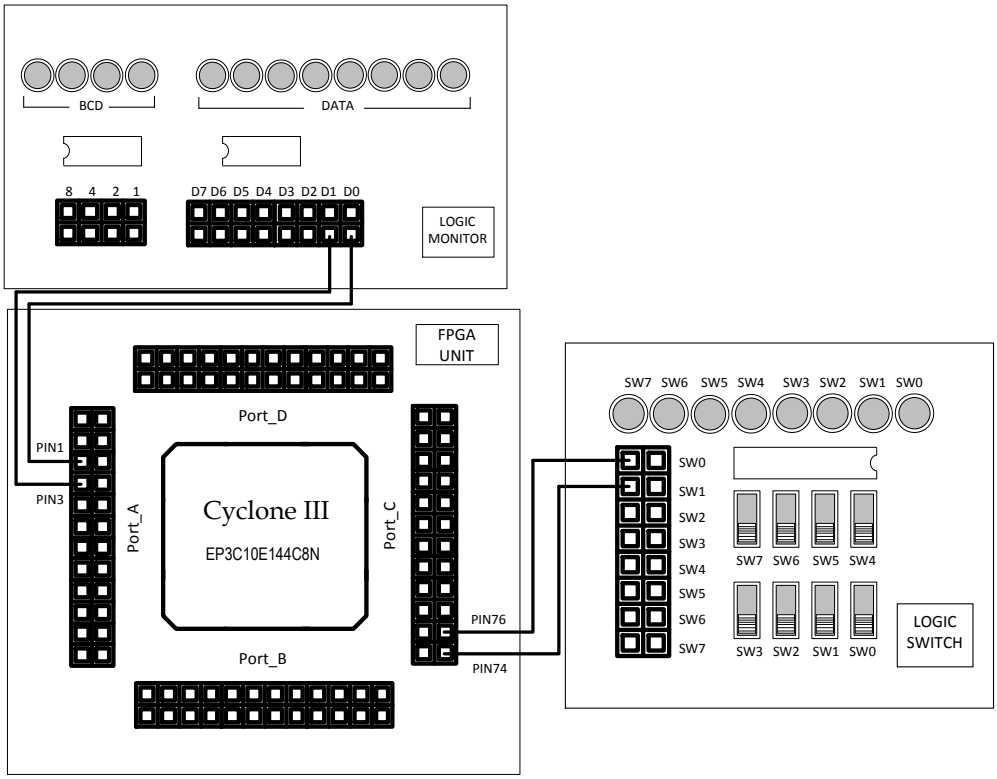
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
D	76
CLK	74
D_Latch	3
D_FF	1

ตารางที่ 10.21 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 10.33 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 10.33 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0 และ SW1 แล้วดูผลทางลอจิกที่ LED D0 และ LED D1 จากนั้นบันทึกผลลงในตาราง

CLK (SW1)	**D (SW0)**	**D_Latch (LED D1)**	**D_FF (LED D0)**
LOW	LOW		
LOW	HIGH		
HIGH	HIGH		
HIGH	LOW		
LOW	LOW		
LOW	LOW		
HIGH	LOW		
HIGH	HIGH		

ตารางที่ 10.22 ตารางความจริงจากการทดลองตอนที่ 6

ขั้นตอนการทดลอง ตอนที่ 7 ฟลิปฟลอปแบบ J-K

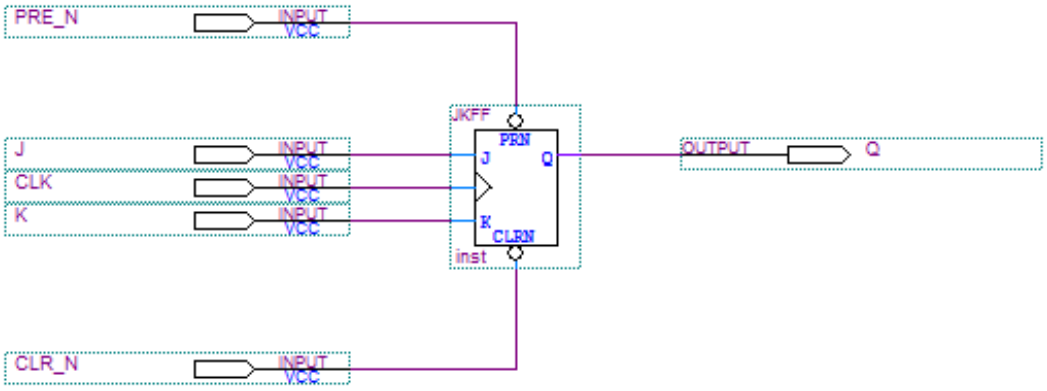
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_10g โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ jkff, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

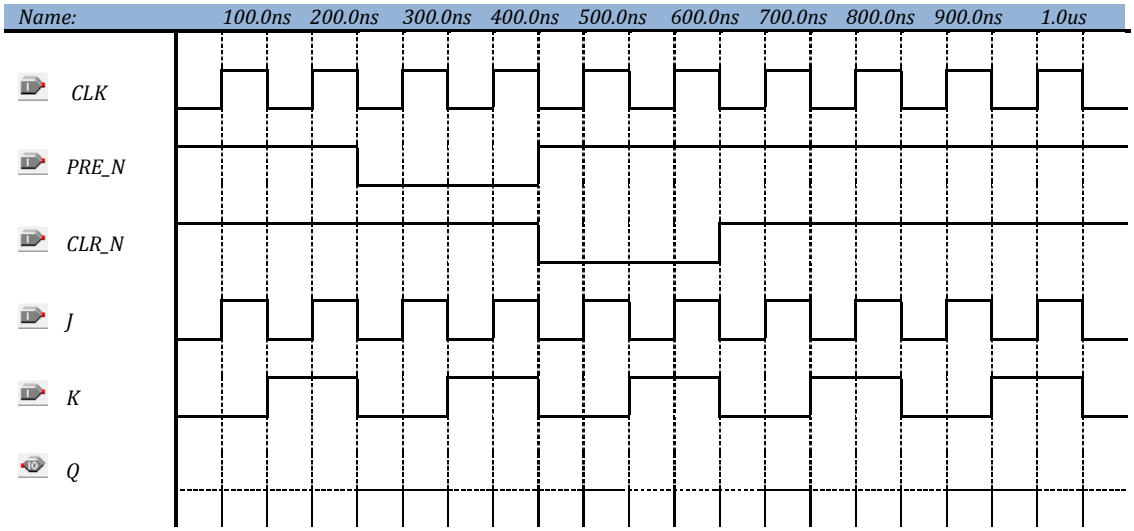


รูปที่ 10.34 วงจรสำหรับการทดลองตอนที่ 7

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_10g.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_10g.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 10.35 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

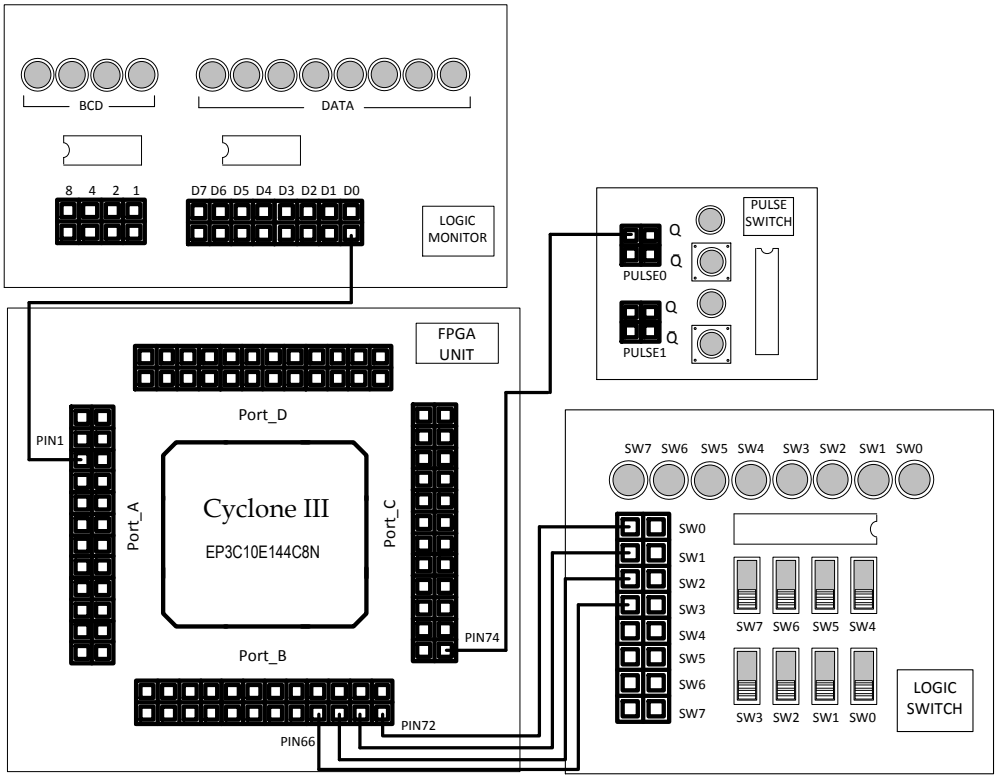
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	**ตำแหน่งขาบน FPGA**
CLK	74
PRE_N	66
CLR_N	68
J	70
K	72
Q	1

ตารางที่ 10.23 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 10.36 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 10.36 การต่อวงจรบนชุดทดลอง

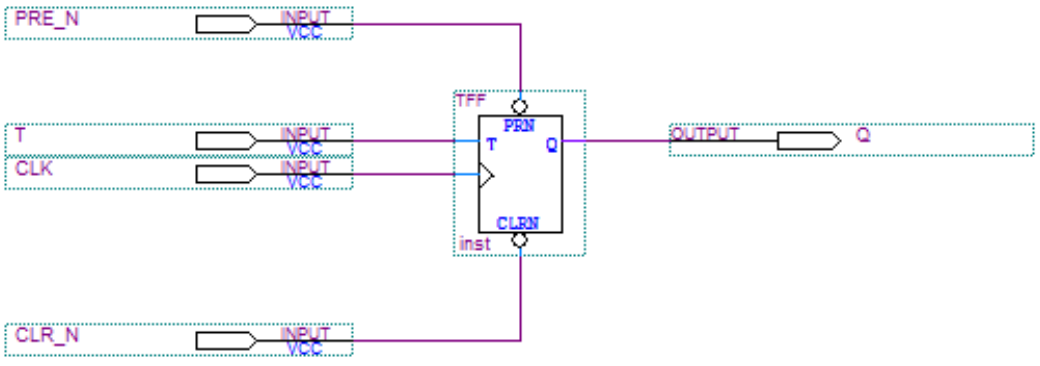
13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1, SW2 และ SW3 ทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0 จากนั้นบันทึกผลลงในตาราง

PRE_N (SW3)	CLR_N (SW2)	J (SW1)	K (SW0)	CLK (PULSEO_Q)	Q (LED D0)
HIGH	HIGH	LOW	LOW	↑	
HIGH	HIGH	HIGH	LOW	↑	
HIGH	HIGH	LOW	HIGH	↑	
HIGH	HIGH	HIGH	HIGH	↑	
LOW	HIGH	LOW	LOW	↑	
LOW	HIGH	HIGH	LOW	↑	
LOW	HIGH	LOW	HIGH	↑	
LOW	HIGH	HIGH	HIGH	↑	
HIGH	LOW	LOW	LOW	↑	
HIGH	LOW	HIGH	LOW	↑	
HIGH	LOW	LOW	HIGH	↑	
HIGH	LOW	HIGH	HIGH	↑	

ตารางที่ 10.24 ตารางความจริงจากการทดลองตอนที่ 7

ขั้นตอนการทดลอง ตอนที่ 8 ฟลิปฟลอปแบบ T

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_10h โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ tff, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

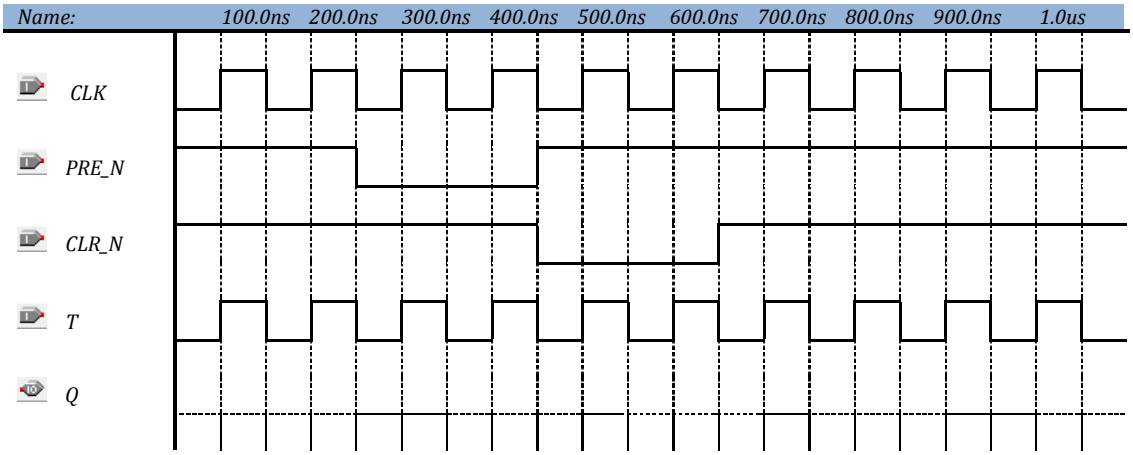


รูปที่ 10.37 วงจรสำหรับการทดลองตอนที่ 8

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_10h.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_10h.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 10.38 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

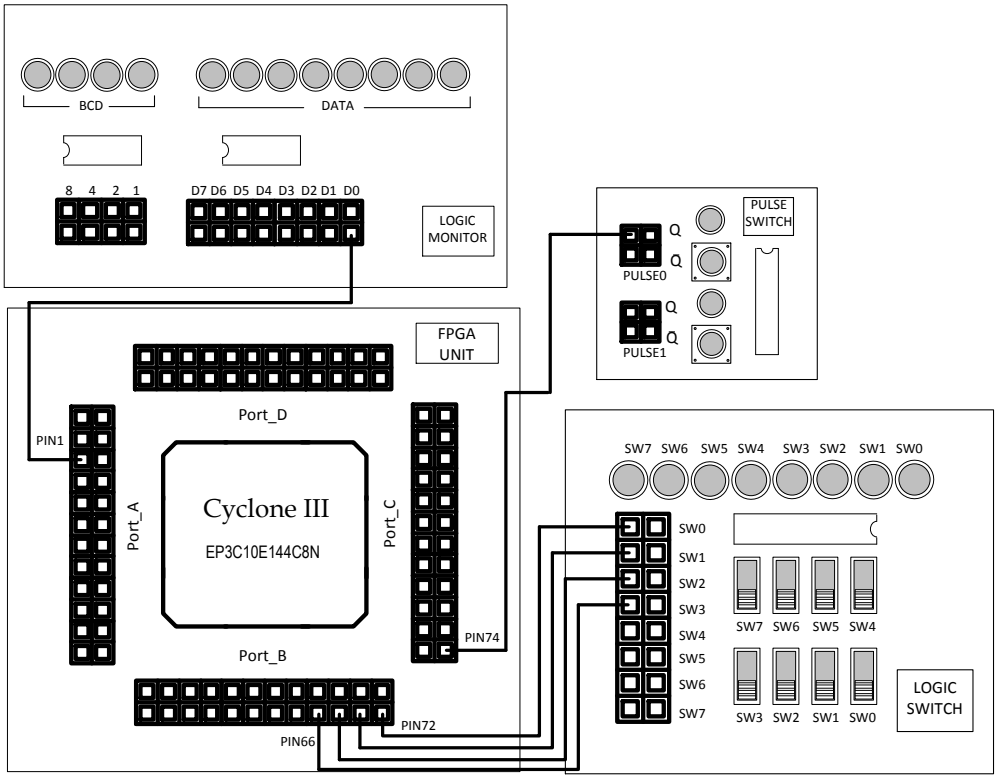
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
CLK	74
PRE_N	68
CLR_N	70
T	72
Q	1

ตารางที่ 10.25 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 10.39 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, SW1 และ SW2 ทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0 จากนั้นบันทึกผลลงในตาราง

PRE_N (SW2)	CLR_N (SW1)	T (SW0)	CLK (PULSEO_Q)	Q (LED D0)
HIGH	HIGH	LOW	↑	
HIGH	HIGH	HIGH	↑	
HIGH	HIGH	LOW	↑	
HIGH	HIGH	HIGH	↑	
LOW	HIGH	LOW	↑	
LOW	HIGH	HIGH	↑	
LOW	HIGH	LOW	↑	
LOW	HIGH	HIGH	↑	
HIGH	LOW	LOW	↑	
HIGH	LOW	HIGH	↑	
HIGH	LOW	LOW	↑	
HIGH	LOW	HIGH	↑	

ตารางที่ 10.26 ตารางความจริงจากการทดลองตอนที่ 8

สรุปผลการทดลอง

การทดลองที่ 11

วงจรนับแบบอะซิงโครนัส(Asynchrononous Counter)

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรนับแบบอะซิงโครนัส
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรนับแบบอะซิงโครนัสได้

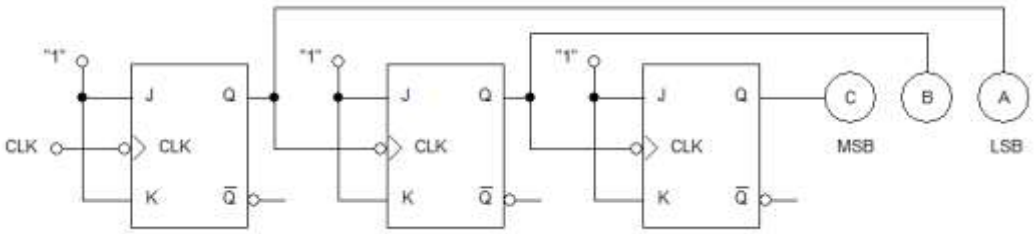
อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

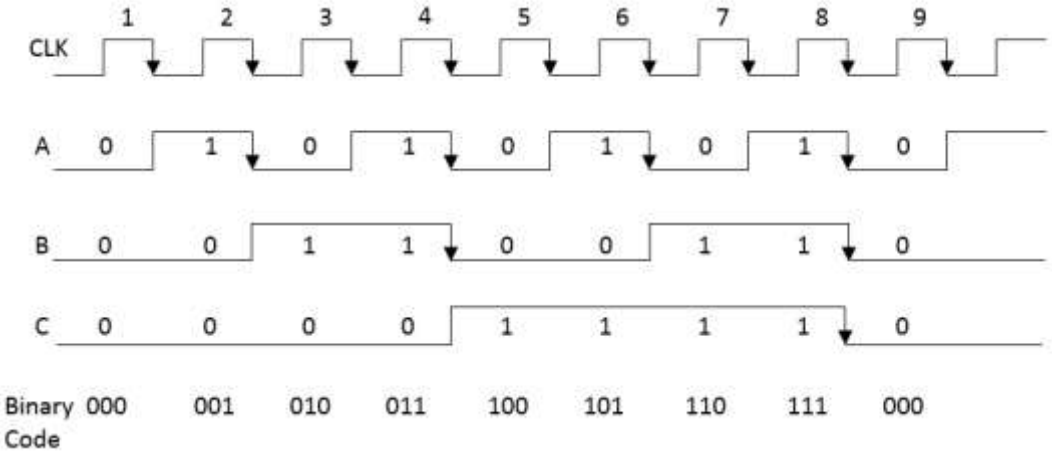
ทฤษฎี

วงจรนับในระบบดิจิตอลแบ่งเป็น 2 ชนิด ได้แก่วงจรนับแบบอะซิงโครนัส (Asynchronous Counter) ซึ่งในการทดลองนี้เราจะเป็นการศึกษาเฉพาะในส่วนของวงจรนับแบบอะซิงโครนัส เท่านั้น ส่วนวงจรนับแบบซิงโครนัสจะได้ศึกษาในการทดลองถัดไป

วงจรนับแบบแบบอะซิงโครนัสเป็นวงจรที่ประกอยขึ้นจากการนำฟลิปฟลอบมาต่ออนุกรมกันโดยสัญาณนาฬิกาจะถูกป้อนเข้าไปที่ฟลิปฟลอปตัวแรกเพียงตัวเดียวเท่านั่น และเอาต์พุทของฟลิปฟลอปแต่ละตัวจะถูกป้อนเป็นสัญญาณนาฬิกาให้ฟลิปฟลอปตัวถัดไป ฟลิปฟลอปแต่ละตัวจะถูกกระตุ้นไม่พร้อมกันโดยจะทำงานไล่มาจากตัวแรกและส่งต่อการกระตุ้นไปยังฟลิแฟลอปตัวถัดไป สำหรับจำนวนฟลิปฟลอปที่ต้องใช้ในวงจรจะมีจำนวนบิตของเลขที่ต้องการนับ เช่น ถ้าต้องการนับเลขฐาน 2 ขานด 3 บิต นับได้ตั้งแต่ 0 (000) ถึง 7 (111) เราต้องใช้ฟลิปฟลอบจำนวน 3 ตัวมาต่ออนุกรมกัน

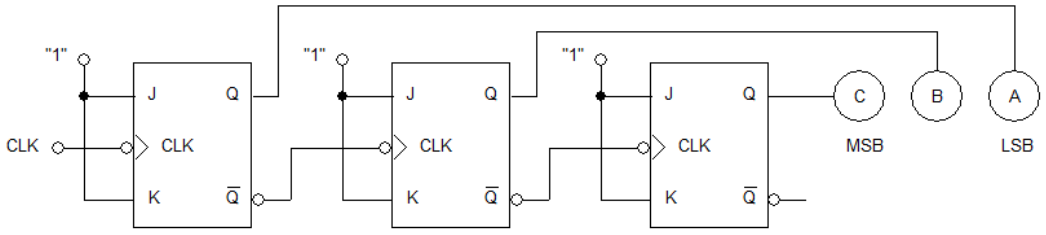


รูปที่ 11.1 วงจรนับอะซิงโครนัสขนาด 3 บิต แบบนับขึ้น

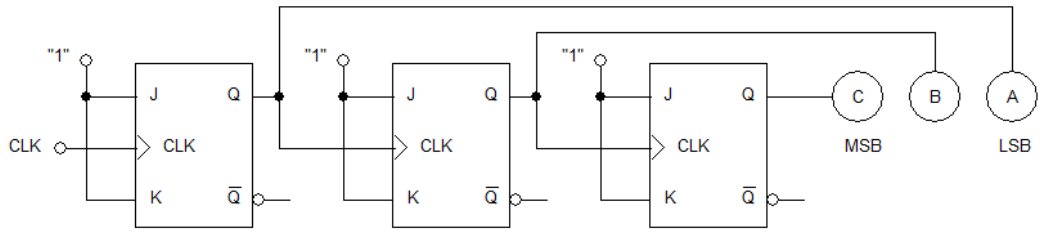


รูปที่ 11.2 รูปคลื่นสัญญาณของวงจรนับอะซิงโครนัส 3 บิต แบบนับขึ้น

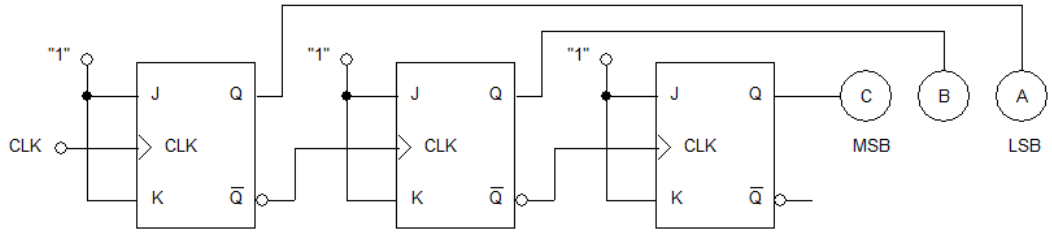
จากรูปที่ เป็นวงจรนับขึ้นโดยใช้ฟลิปฟลอป J-K ชนิดแอคทีฟโลว์ ซึ่งเราสามารถดัดแปลงเป็นวงจรนับลงได้ดังรูปที่ 11.3 ส่วนในรูปที่ 11.4 และ 11.5 เป็นวงจรนับอะซิงโครนัสแบบใช้ฟลิปฟลอป J-K ชนิดแอคทีฟไฮ ซึ่งจะสังเกตได้ว่ามีลักษณะการต่อตรงกันข้ามกับการใช้ฟลิปฟลอป J-K ชนิดแอคทีฟโลว์



รูปที่ 11.3 วงจรนับอะซิงโครนัสขนาด 3 บิต แบบนับลง

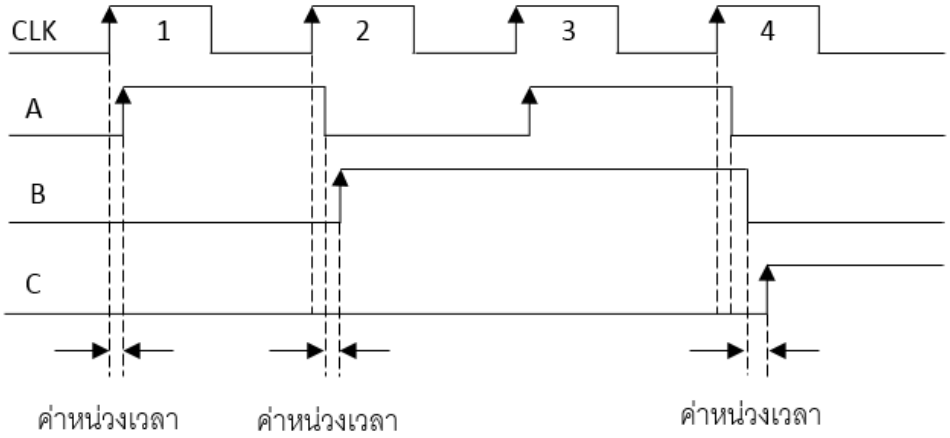


รูปที่ 11.4 วงจรนับอะซิงโครนัสขนาด 3 บิต แบบนับขึ้น โดยใช้ฟลิปฟลอบ J-K แบบแอคทีฟไฮ



รูปที่ 11.5 วงจรนับอะซิงโครนัสขนาด 3 บิต แบบนับลง โดยใช้ฟลิปฟลอบ J-K แบบแอคทีฟไฮ

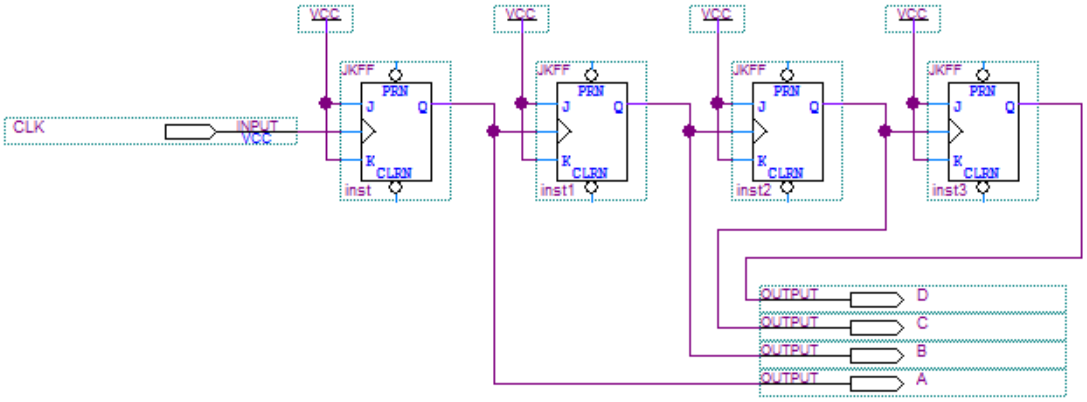
เนื่องจากการทำงานฟลิปฟลอปแต่ละตัวจะเป็นการกระตุ้นจากฟลิปฟลอปตัวที่อยู่ก่อนหน้ามัน ดังนั้นในการใช้งานจริงจะเกิดปัญหาจากค่าความหน่วงเวลาภายในของฟลิปฟลอป (Propagation Delay) ยิ่งในวงจรนับมีการใช้ฟลิปฟลอปมากเท่าใด ค่าความหน่างเวลานี้จะถูกสะสมมากขึ้นเท่านั้น ซึ่งค่าความหน่วงสะสมนี้จะต้องไม่มากเกินคาบเวลาของสัญญาณนาฬิกา จากสาเหตุนี้เองจึงทำให้วงจรนันอะซิงโครนัสไม่สามารถนำมาใช้กับการนับที่มีจำนวนบิตมาก ๆ ที่ความถี่สูงได้



รูปที่ 11.6 ค่าความเวลาในฟลิปฟลอปที่มีผลต่อการทำงานของวงจรอะซิงโครนัส

ขั้นตอนการทดลอง ตอนที่ 1 วงจรนับอะซิงโครนัส แบบนับลง

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_11a โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ jkff, vcc, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

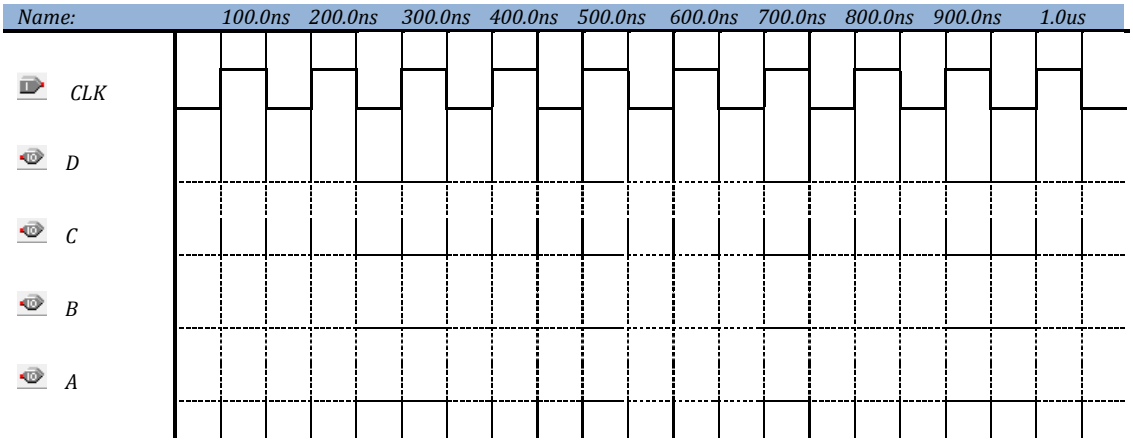


รูปที่ 11.7 วงจรสำหรับการทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_11a.gdf

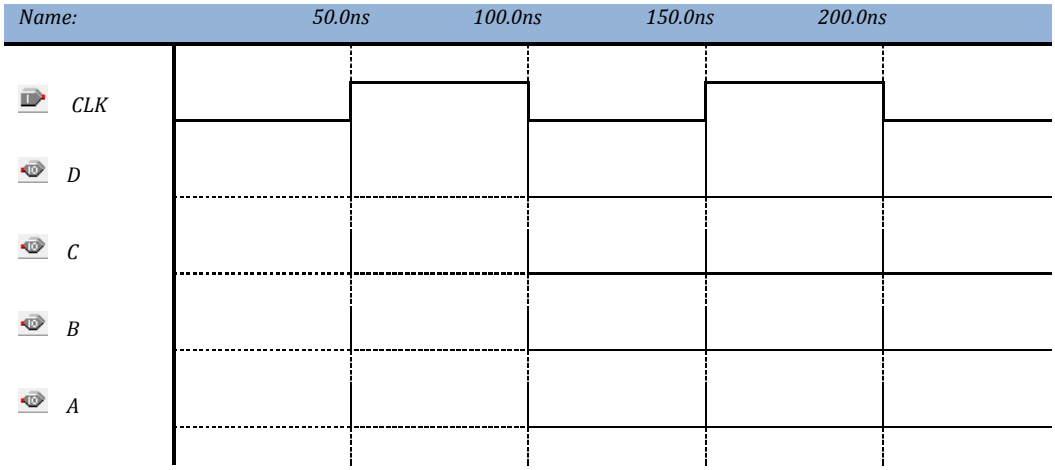
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_11a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 11.8 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation เสร็จแล้วให้ ซูมเพื่อดูค่าความหน่วงเวลาของฟลิปฟลอปในหน้าต่าง Vector Waveform โดยใช้เมนู Viwe -> Zoom In เสร็จแล้วบันทึกผลการทดลอง



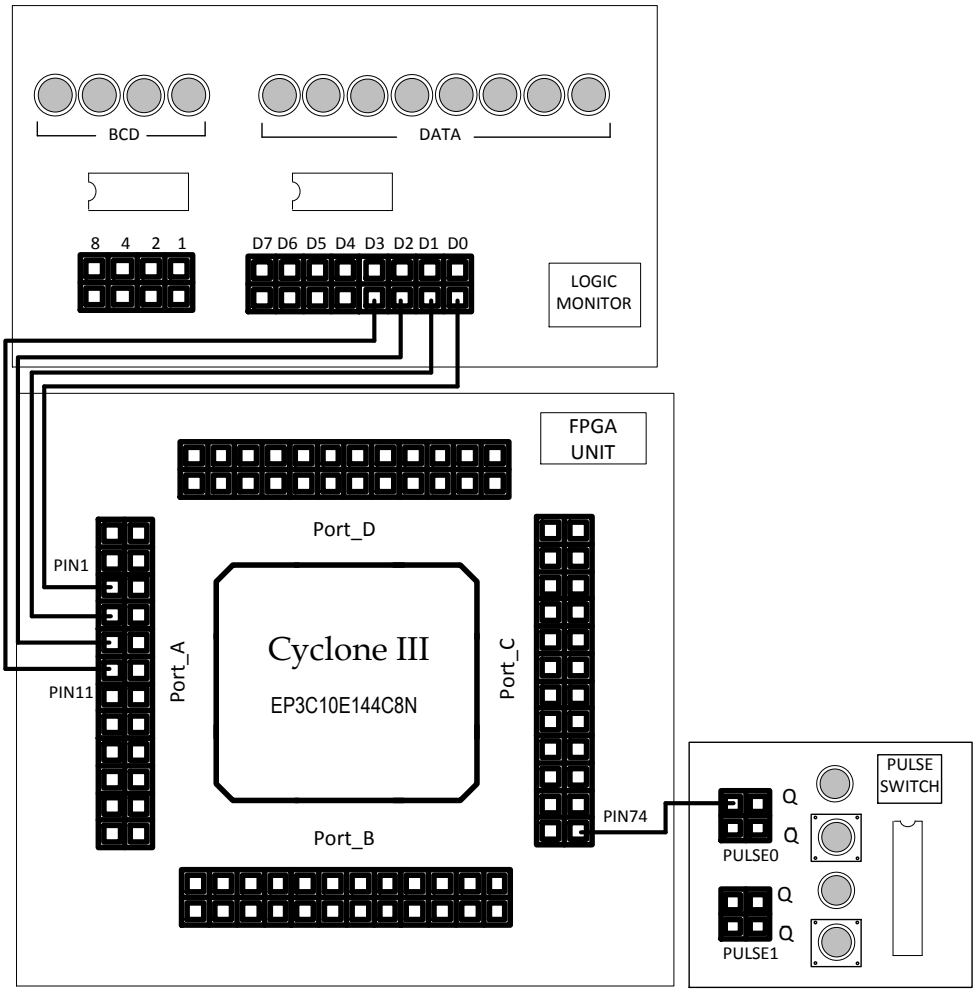
รูปที่ 11.9 ผลการจำลองการทำงาน

10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins
11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	**ตำแหน่งขาบน FPGA**
CLK	74
A	1
B	3
C	7
D	11

ตารางที่ 11.1 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 11.10 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 11.10 การต่อวงจรบนชุดทดลอง

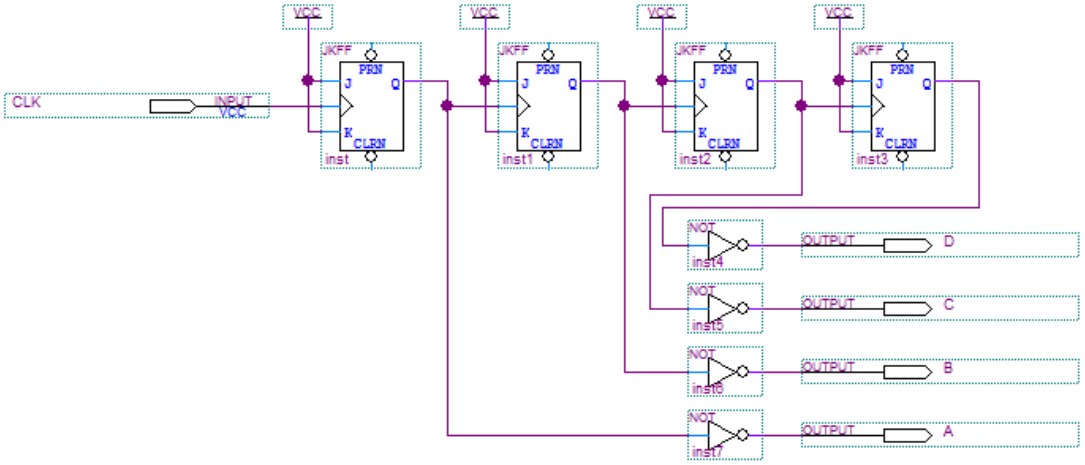
13. ทดสอบวงจรโดยทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0, LED D1, LED D2 และ LED D3 จากนั้นบันทึกผลลงในตาราง

CLK (PULSEO_Q)	D (LED D3)	C (LED D2)	B (LED D1)	A (LED D0)
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				

ตารางที่ 11.2 ตารางความจริงจากการทดลองตอนที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 วงจรนับอะซิงโครนัส แบบนับขึ้น

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_11b โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ jkff, not, vcc, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

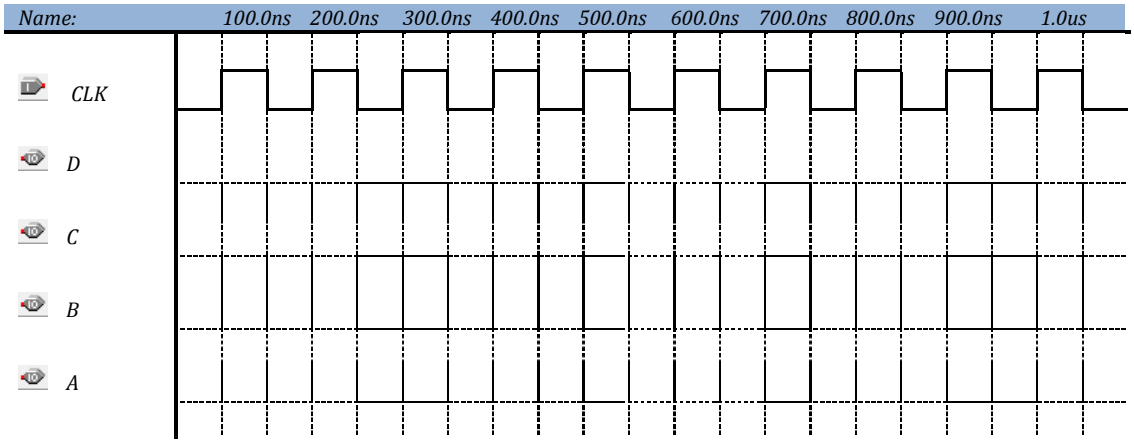


รูปที่ 11.11 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_11b.gdf

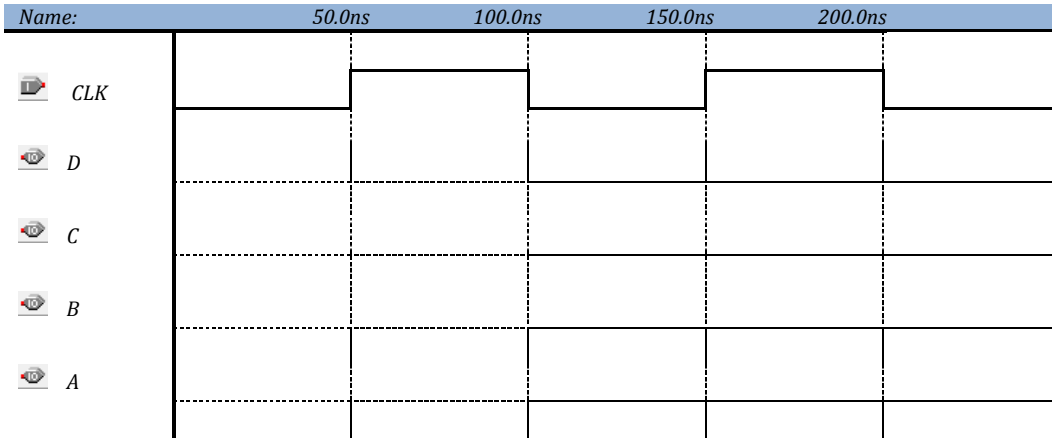
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_11b.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 11.12 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation เสร็จแล้วให้ ซูมเพื่อดูค่าความหน่วงเวลาของฟลิปฟลอปในหน้าต่าง Vector Waveform โดยใช้เมนู Viwe -> Zoom In เสร็จแล้วบันทึกผลการทดลอง



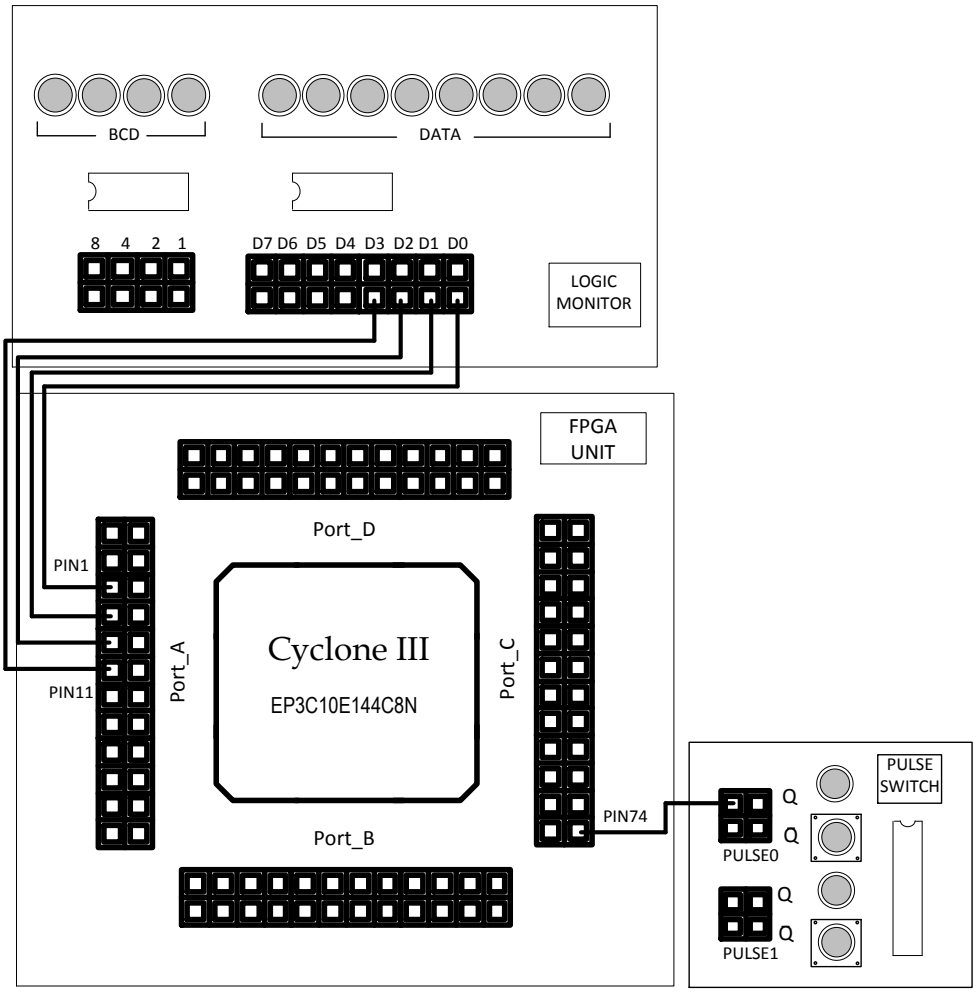
รูปที่ 11.13 ผลการจำลองการทำงาน

10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins
11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	**ตำแหน่งขาบน FPGA**
CLK	74
A	1
B	3
C	7
D	11

ตารางที่ 11.3 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 11.14 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 11.14 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0, LED D1, LED D2 และ LED D3 จากนั้นบันทึกผลลงในตาราง

CLK (PULSEO_Q)	D (LED D3)	C (LED D2)	B (LED D1)	A (LED D0)
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				

ตารางที่ 11.4 ตารางความจริงจากการทดลองตอนที่ 2

สรุปผลการทดลอง

การทดลองที่ 12

วงจรนับแบบซิงโครนัส (Synchrononous Counter)

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรนับแบบซิงโครนัส
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรนับแบบซิงโครนัสได้

อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

ทฤษฎี

วงจรนับแบบซิงโครนัส (Syncronous Counter) สร้างขึ้นจากฟลิปฟลอปเช่นเดียวกันกับวงจรอะซิงโครนัส แต่สัญญาณนาฬิกาจะถูกป้อนเข้าที่ฟลิปฟลอปทุกตัวโดยตรง ซึ่งฟลิปฟลอปทุกตัวจะจะทำงานได้พร้อมกันโดยไม่เกิดปัญหาค่าหน่วงเวลาสะสมแบบวงจรอะซิงโครนัส ดังนั้นวงจรนับชนิดนี้จึงสามารถนำไปใช้งานที่ความถี่สูง ๆ ได้ จากลักษณะการต่อวงจรที่มีสัญญาณนาฬิกาต่อเข้าที่ฟลิปฟลอปทุกตัวจึงทำให้บางครั้งเราเรียกวงจรจรนับชนิดนี้ว่าวงจรนับแบบขนาน (Parallel Counter)

ขั้นตอนการออกแบบวงจรนับซิงโครนัสเริ่มจากการเขียนสเตรทไดอะแกรม (State Diagram) แล้วใช้ตารางเอ็กซ์ไซท์ชั่น (Excitaion Table) สร้างตารางความจริง จากนั้นนำตารางความจริงไปทำการลดรูปและสร้างเป็นลอจิก

สำหรับตารางเอ็กซ์ไซด์เทชั่นของ J-K ฟลิปฟลิปโดยที่ X คือค่า Don’ Care หมายถึงลอจิกเป็น 0 หรือ 1 ก็ได้

อินพุท		เอาต์พุท	โหมดการทำงาน
J	**K**	$\mathbf{Q_{t+1}}$	
0	0	Q_t	ไม่เปลี่ยนแปลง
0	1	0	รีเซท
1	0	1	เซท
1	1	$\overline{Q}_t$	ท็อกเกิล

ตารางที่ 12.1 ตารางความจริงของ JK ฟลิปฟลอป

เอาต์พุท		อินพุท	
$\mathbf{Q_t}$	$\mathbf{Q_{t+1}}$	**J**	**K**
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

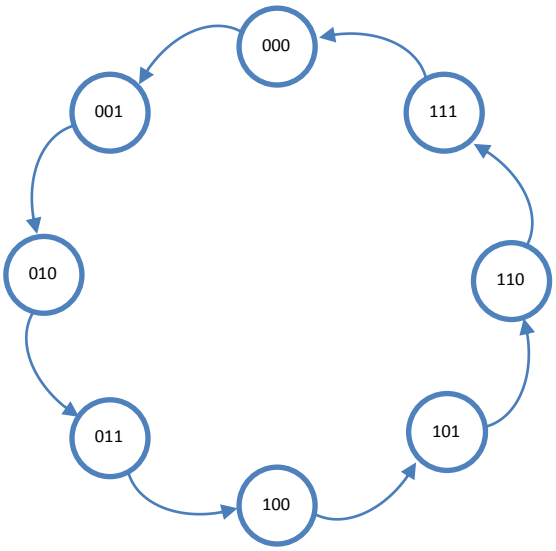
ตารางที่ 12.2 ตารางเอ็กไซท์เทชั่นของ JK ฟลิปฟลอป

ตารางเอ็กซ์ไซท์เทชั่นสร้างมาจากการสังเกตตารางความจริงดังนี้

1. ถ้าเอาต์พุท Q เปลี่ยนค่าจาก 0 เป็น 0 เหตุการณ์นี้จะเกิดขึ้นจาก 2 กรณีคือ
    1.1 กรณีที่ J เป็น 0 และ K เป็น 0 (คงสถานะไม่เปลี่ยนแปลง)
    1.2 กรณีที่ J เป็น 0 และ K เป็น 1
2. ถ้าเอาต์พุท Q เปลี่ยนค่าจาก 0 เป็น 1 เหตุการณ์นี้จะเกิดขึ้นจาก 2 กรณีคือ
    2.1 กรณีที่ J เป็น 1 และ K เป็น 0
    2.2 กรณีที่ J เป็น 1 และ K เป็น 1 (ท็อกเกิลกลับสถานะ)
3. ถ้าเอาต์พุท Q เปลี่ยนค่าจาก 1 เป็น 0 เหตุการณ์นี้จะเกิดขึ้นจาก 2 กรณีคือ
    3.1 กรณีที่ J เป็น 0 และ K เป็น 1
    3.2 กรณีที่ J เป็น 1 และ K เป็น 1 (ท็อกเกิลกลับสถานะ)
4. ถ้าเอาต์พุท Q เปลี่ยนค่าจาก 1 เป็น 1 เหตุการณ์นี้จะเกิดขึ้นจาก 2 กรณีคือ
    2.1 กรณีที่ J เป็น 1 และ K เป็น 0 (คงสถานะไม่เปลี่ยนแปลง)
    2.2 กรณีที่ J เป็น 1 และ K เป็น 0

ตัวอย่างขั้นตอนการออกแบบวงจรนับซิงโครนัส 3 บิตโดยใช้ J-K ฟลิปฟลอปสามารถทำได้ดังนี้

1. เขียนสเตทไดอะแกรมดังรูปที่ 12.1 เพื่อดูขั้นตอนการเปลี่ยนแปลงสภาวะเอาต์พุท



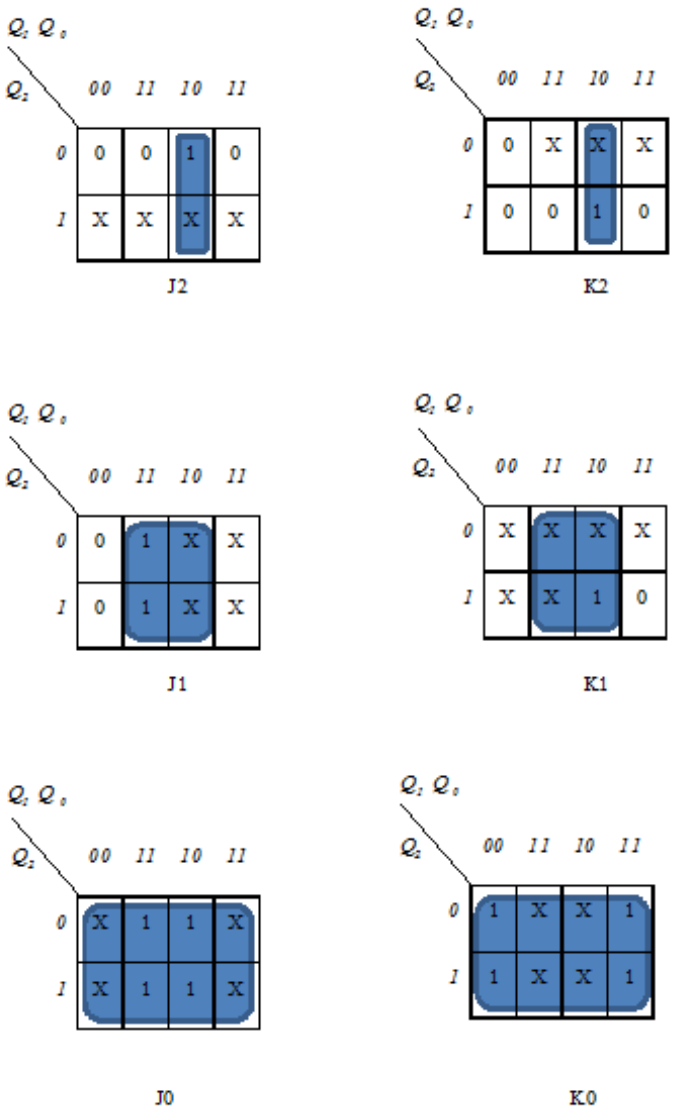
รูปที่ 12.1 สเตทไดอะแกรมของวงจรนับซิงโครนัส

2. เขียนตารางความจริงโดยใช้ตารางเอ็กซ์ไซท์เทชั่น

สถานะปัจจุบันของเอาต์พุท			สถานะต่อไปของเอาต์พุท			อินพุทของฟลิปฟลอป		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	J_2 K_2	J_1 K_1	J_0 K_0
0	0	0	0	0	1	0 X	0 X	1 X
0	0	1	0	1	0	0 X	1 X	X 1
0	1	0	0	1	1	0 X	X 0	1 X
0	1	1	1	0	0	1 X	X 1	X 1
1	0	0	1	0	1	X 0	0 X	1 X
1	0	1	1	1	0	X 0	1 X	X 1
1	1	0	1	1	1	X 0	X 0	1 X
1	1	1	0	0	0	X 1	X 1	X 1

ตารางที่ 12.3 ตารางความจริงของวงจรนับซิงโครนัส 3 บิต แบบใช้ฟลิปฟลอบ JK

3. ลดรูปวงจรลอจิกจากตารางความจริง

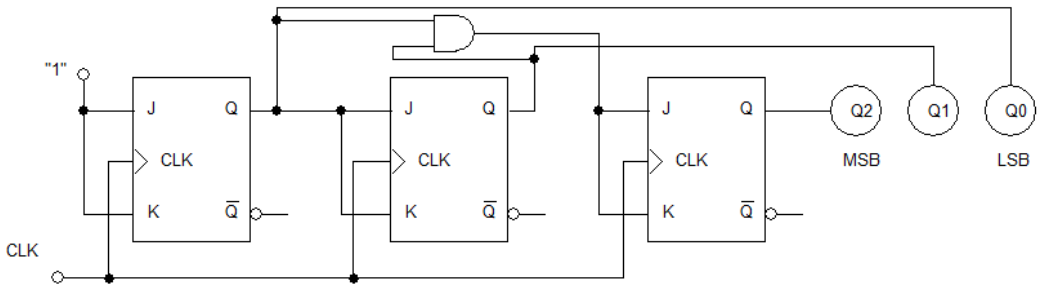


รูปที่ 12.2 K – Map ของวงจรนับซิงโครนัส 3 บิต แบบใช้ฟลิปฟลอป JK

จาก K-Map เราสามารถเขียนสมการลอจิกได้ดังนี้

$J2 = Q_1Q_0$ $K2 = Q_1Q_0$
$J1 = Q_0$ $K1 = Q_0$
$J0 = 1$ $K0 = 1$

4. สร้างวงจรโดยเชื่อมต่อขา CLK ของฟลิปฟลอปทุกตัวเข้าด้วยกัน จากนั้นนำสมการที่ได้มาสร้างเป็นวงจรลอจิกทางด้านอินพุทของฟลิปฟลอปแต่ละตัว



รูปที่ 12.3 วงจรนับอะซิงโครนัสขนาด 3 บิต

นอกจากการใช้ J-K ฟลิปฟลอป มาสร้างเป็นวงจรนับซิงโครนัสแล้วยังสามารถนำฟลิปฟลอป D มาใช้เช่นกัน ซึ่งขั้นตอนการออกแบบจะเป็นเช่นเดียวกับการใช้ J-K ฟลิปฟลอปแต่จะใช้ตารางเอ็กซ์ไซท์เทชั่นของ D ฟลิปฟลอปดังตารางที่ 12.5

อินพุท	เอาต์พุท	โหมดการทำงาน
D	Q_{t+1}	
0	Q_t	รีเซท
1	0	เซท

เอาต์พุท		อินพุท
Q_t	Q_{t+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

ตารางที่ 12.4 ตารางความจริงของ J-K

ตารางที่ 12.5 ตารางเอ็กซ์ไซท์เทชั่นของ J-K ฟลิปฟลอป

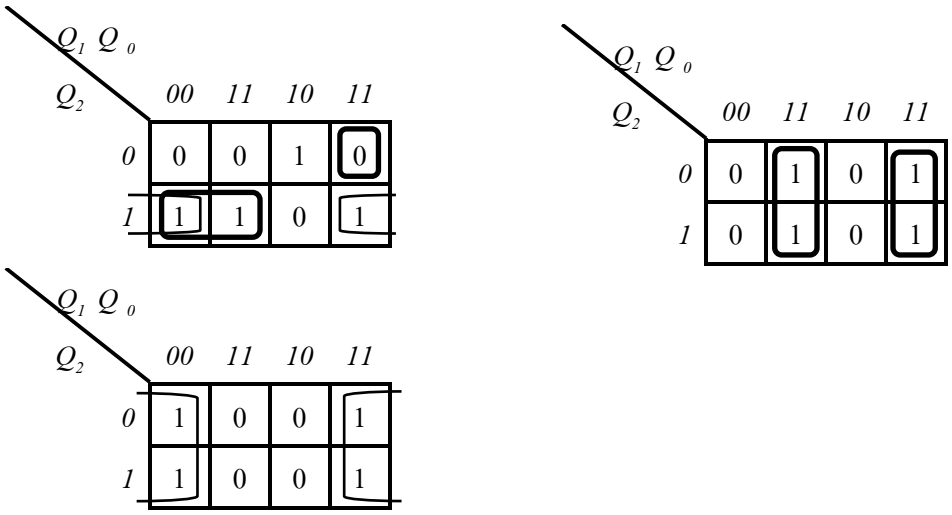
ตัวอย่างขั้นตอนการออกแบบวงจนนับซิงโครนัส 3 บิตโดยใช้ D ฟลิปฟลอปสามารถทำได้ดังนี้

1. เขียนสเตทไดอะแกรมดังรูปที่ 12.1 เพื่อดูขั้นตอนการเปลี่ยนแปลงสภาวะเอาต์พุท
2. เขียนตารางความจริงโดยใช้ตารางเอ็กซ์ไซท์เทชั่น

สถานะปัจจุบันของเอาต์พุท			สถานะต่อไปของเอาต์พุท			อินพุทของฟลิปฟลอป		
Q_2	Q_1	Q_0	Q_2	Q_1	Q_0	D2	D1	D0
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	1	1	0	1	1	0
1	1	0	1	1	1	1	1	1
1	1	1	0	0	0	0	0	0

ตารางที่ 12.6 ตารางความจริง

3. ลดรูปวงจรลอจิกจากตารางความจริง

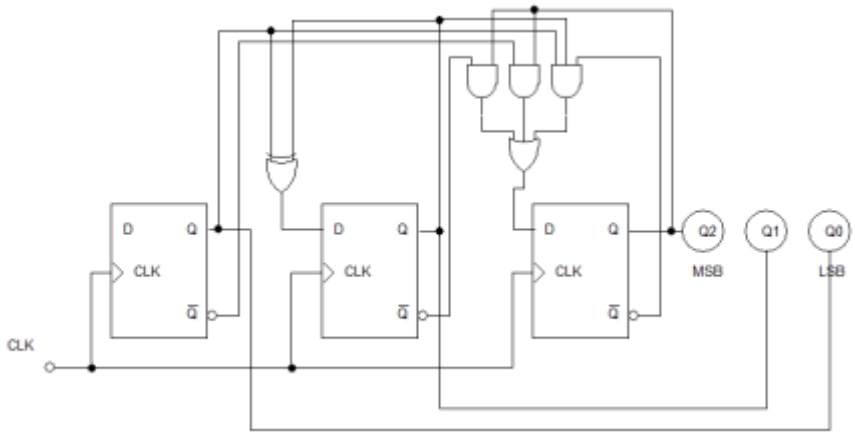


รูปที่ 12.4 K – Map ของวงจรนับซิงโครนัส 3 บิต แบบใช้ฟลิปฟลอป D

จาก K-Map เราสามารถเขียนสมการลอจิกได้ดังนี้

$$D2 = Q_2Q_1Q_0 + Q_2Q_0 + Q_2Q_1$$
$$D1 = Q_1Q_0 + Q_1Q_0 = Q_1 \oplus Q_0$$
$$D0 = Q_0$$

4. สร้างวงจรโดยเชื่อมต่อขา CLK ของฟลิปฟลอปทุกตัวเข้าด้วยกัน จากนั้นนำสมการที่ได้มาสร้างเป็นวงจรลอจิกทางด้านอินพุทของฟลิปฟลอปแต่ละตัว



รูปที่ 12.5 วงจรนับอะซิงโครนัสขนาด 3 บิต แบบนับขึ้น

ขั้นตอนการทดลอง ตอนที่ 1 **วงจรนับซิงโครนัส** 3 **บิต แบบใช้** JK **ฟลิปฟลอป**

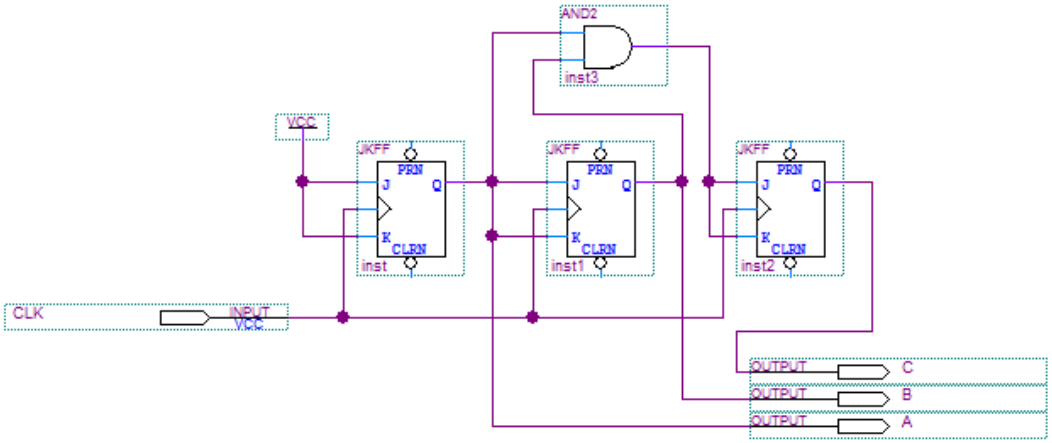
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_12a โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ jkff, and2, vcc, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

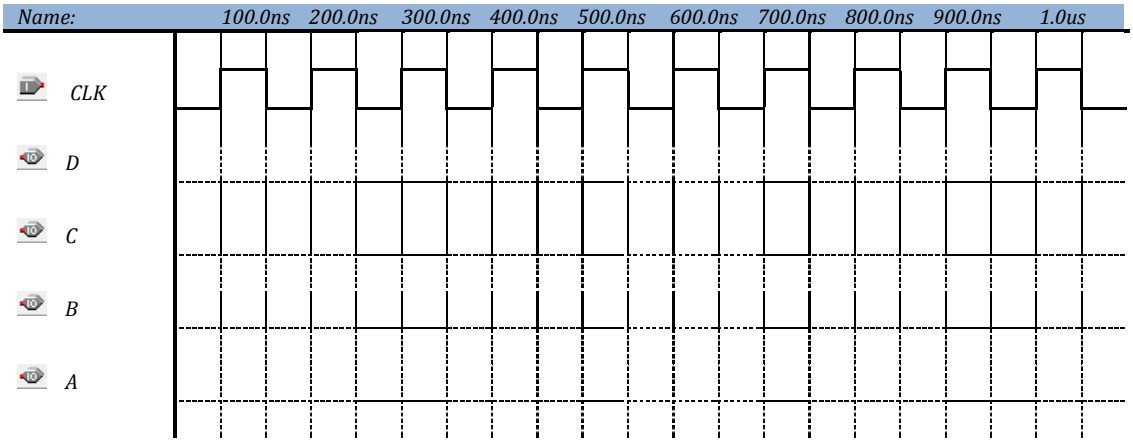


รูปที่ 12.6 วงจรสำหรับการทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_12a.gdf

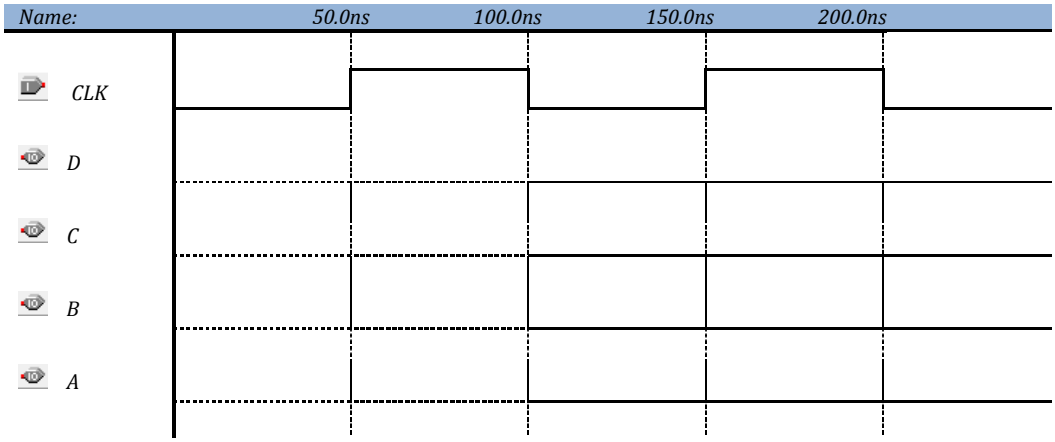
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_12a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 12.7 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation เสร็จแล้วให้ ซูมเพื่อดูค่าความหน่วงเวลาของฟลิปฟลอปในหน้าต่าง Vector Waveform โดยใช้เมนู Viwe -> Zoom In เสร็จแล้วบันทึกผลการทดลอง



รูปที่ 12.8 ผลการจำลองการทำงาน

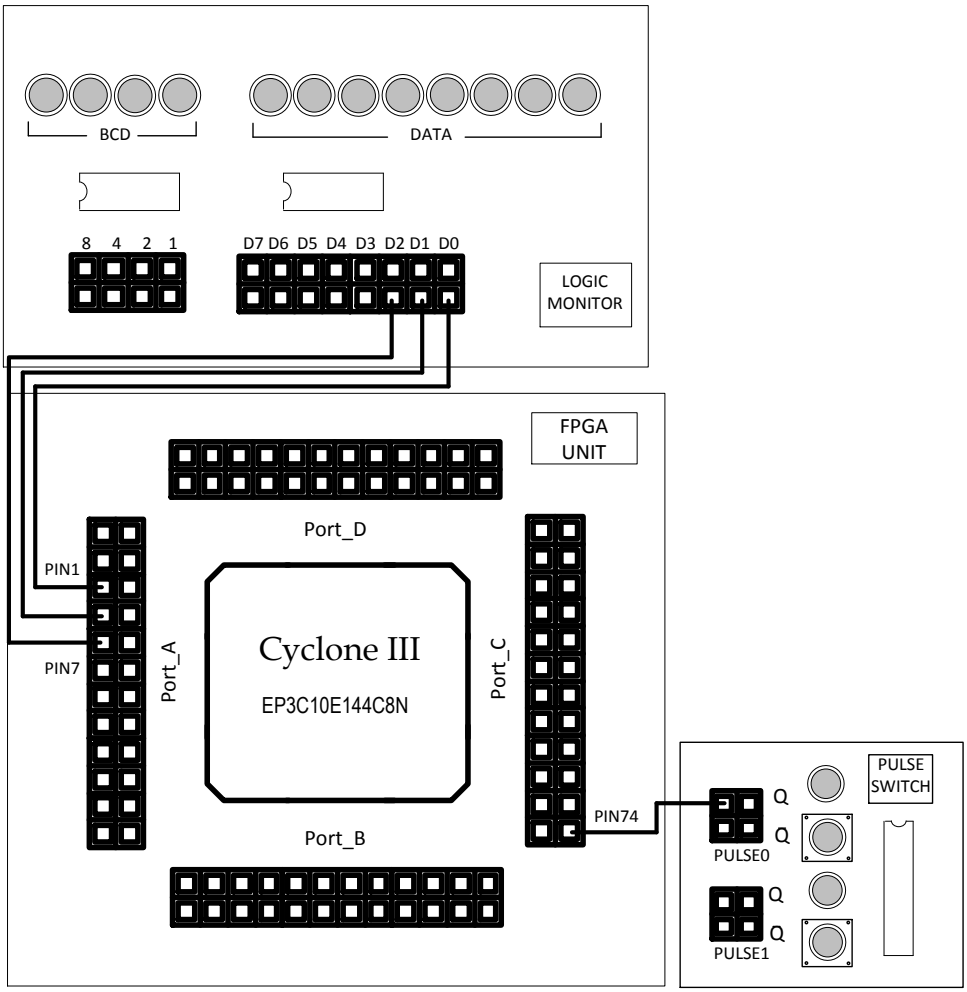
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA
CLK	74
A	1
B	3
C	7

ตารางที่ 12.7 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 12.9 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 12.9 การต่อวงจรบนชุดทดลอง

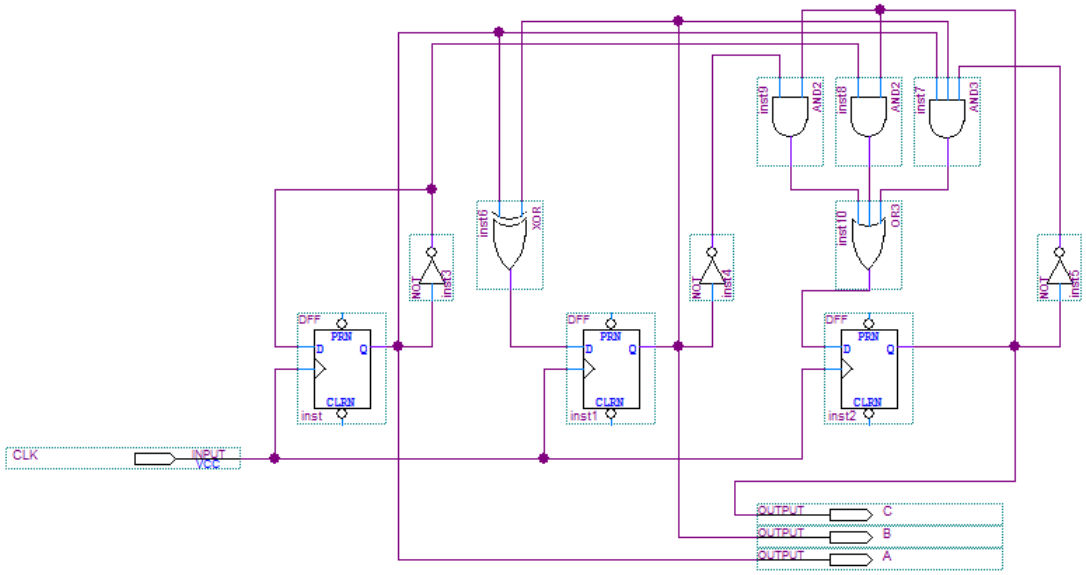
13. ทดสอบวงจรโดยทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0, LED D1 และ LED D2 จากนั้นบันทึกผลลงในตาราง

CLK (PULSEO_Q)	C (LED D2)	B (LED D1)	A (LED D0)
↑			
↑			
↑			
↑			
↑			
↑			
↑			
↑			
↑			
↑			
↑			
↑			

ตารางที่ 12.8 ตารางความจริงจากการทดลองตอนที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 วงจรนับซิงโครนัส 3 บิต แบบใช้ D ฟลิปฟลอป

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_12b โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ dff, and2, and3, xor, or3, not, vcc, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

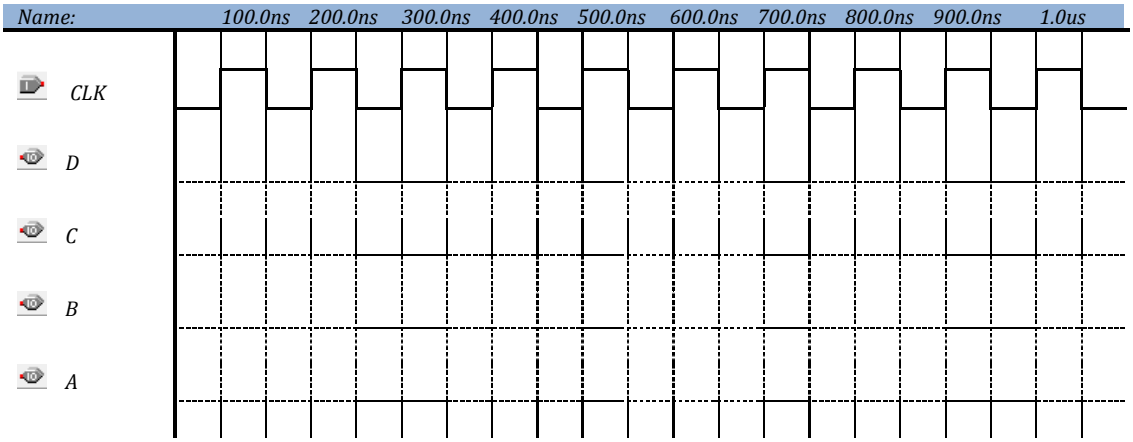


รูปที่ 12.10 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_12b.gdf

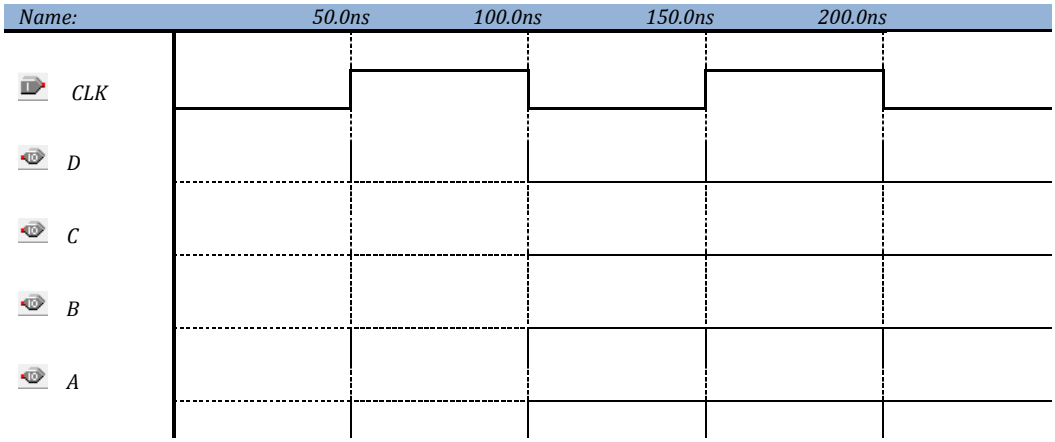
7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_12b.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 12.11 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation เสร็จแล้วให้ ซูมเพื่อดูค่าความหน่วงเวลาของฟลิปฟลอปในหน้าต่าง Vector Waveform โดยใช้เมนู Viwe -> Zoom In เสร็จแล้วบันทึกผลการทดลอง



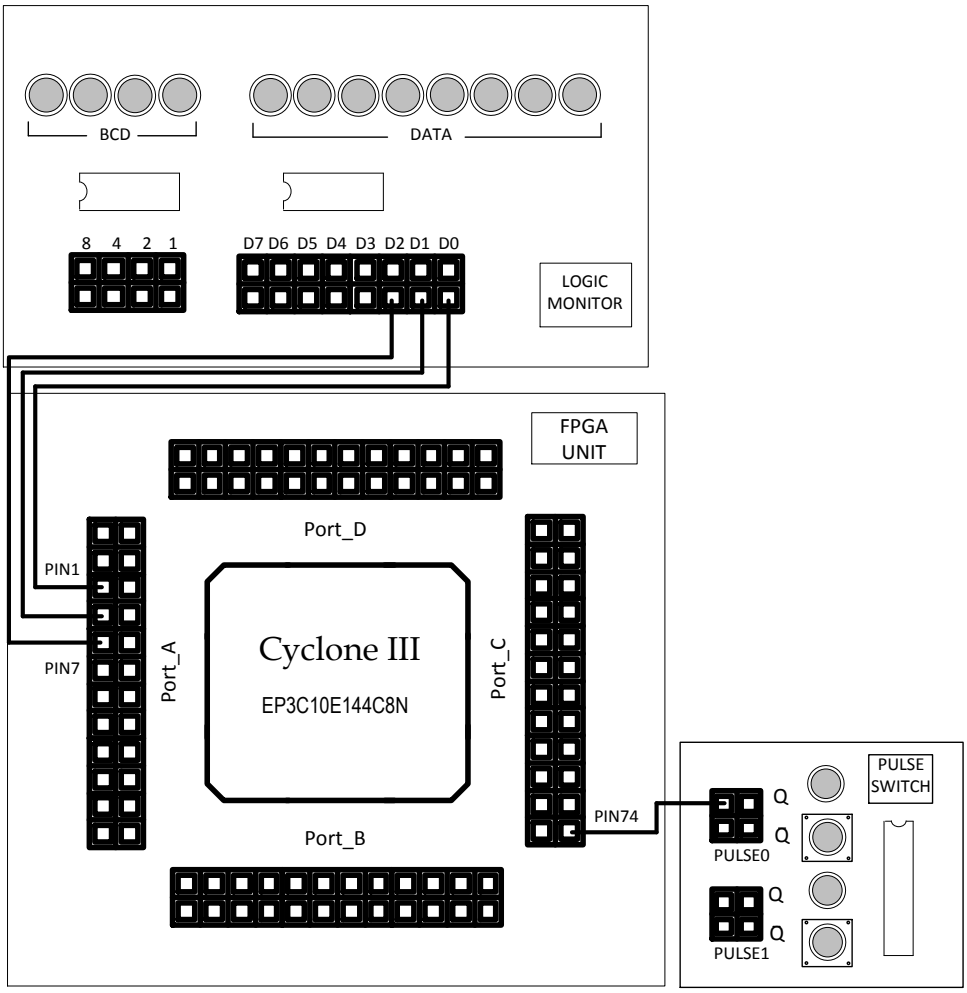
รูปที่ 12.12 ผลการจำลองการทำงาน

10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins
11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	**ตำแหน่งขาบน FPGA**
CLK	74
A	1
B	3
C	7

ตารางที่ 12.9 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 12.13 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 12.13 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0, LED D1 และ LED D2 จากนั้นบันทึกผลลงในตาราง

CLK (PULSEO_Q)	C (LED D2)	B (LED D1)	A (LED D0)
↑			
↑			
↑			
↑			
↑			
↑			
↑			
↑			
↑			
↑			
↑			
↑			

ตารางที่ 12.10 ตารางความจริงจากการทดลองตอนที่ 2

สรุปผลการทดลอง

การทดลองที่ 13

วงจรนับรูปแบบไอซีสำเร็จรูป

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรนับที่อยู่ในไอซีสำเร็จรูป
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรนับที่อยู่ในไอซีสำเร็จรูปได้
4. สามารถดาวน์โหลดวงจรลงไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

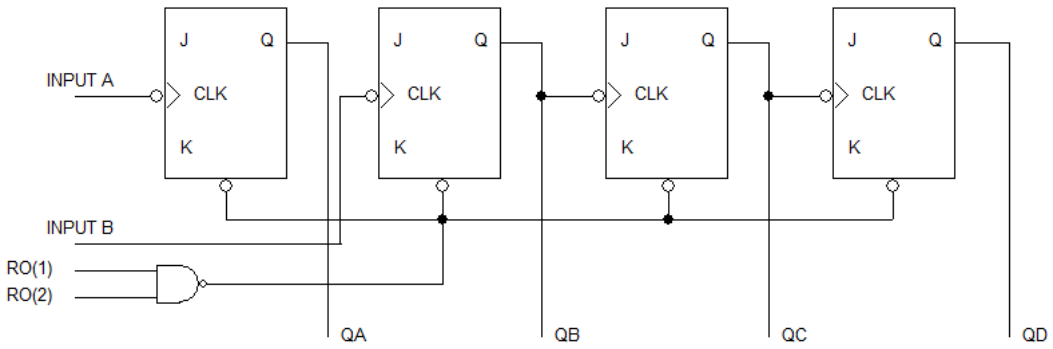
อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

ทฤษฎี

จากการทดลองที่ผ่านมาเป็นการศึกษาถึงหลังการทำงานพื้นฐานของวงจรนับอะซิงโครนัสและวงจรนับซิงโครนัส สำหรับการทดลองในบทนี้จะเป็นการศึกษาทดลองไอซีที่ทำหน้าที่เป็นวงจรนับแบบอะซิงโครนัสซึ่งได้แก่ เบอร์ 7493 และวงจรนับแบบซิงโครนัสเบอร์ 74190

ไอซีเบอร์ 7493 เป็นไอซีนับเลขฐานสองขนาด 4 บิตแบบอะซิงโครนัส มีโครงสร้างดังรูปที่ 13.1 ซึ่งภายในประกอบไปด้วยฟลิฟฟลอป 4 ตัว และมีการแบ่งวงจรภายในออกเป็น 2 ส่วน ได้แก่วงจรนับแบบโมดูโล 2 (Modulo 2 : MOD-2)ซึ่งมีเอาท์พุตออกที่ขา QA และวงจรนับโมดูโล 8 (MOD-8) ซึ่งมีเอาท์พุตออกที่ขา QB, QC และ QD ในใช้เราสามารถเลือกใช้เพียงวงจรใดวงจรหนึ่งก็ได้ แต่หากต้องการใช้งานเป็นวงจรนับ 4 บิตก็สามารถทำได้โดยการต่อขา QA เข้ากับ INPUT B เพื่อเชื่อมต่อสัญญาณนาฬิกาจากวงจรนับ MOD-2 เข้ากับวงจรนับ MOD-8



รูปที่13.1 วงจรภายในไอซี 7493

นับ	เอาต์พุท			
	QD	QC	QB	QA
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	L	H	L	H
6	L	H	H	L
7	L	H	H	H

นับ	เอาต์พุท			
	QD	QC	QB	QA
8	H	L	L	L
9	H	L	L	H
10	H	L	H	L
11	H	L	H	H
12	H	H	L	L
13	H	H	L	H
14	H	H	H	L
15	H	H	H	H

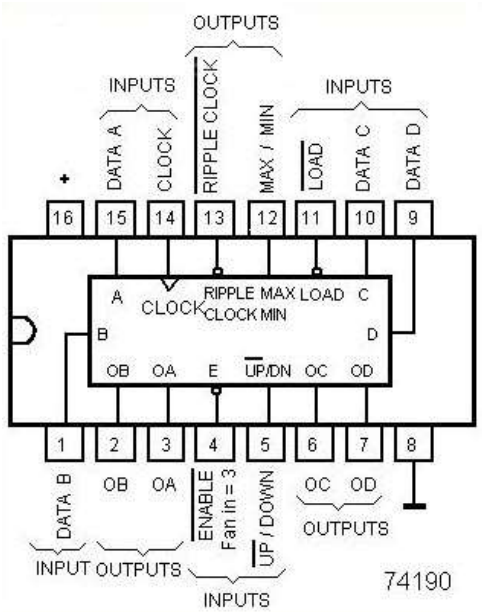
ตารางที่ 13.1 ตารางความจริงการนับของไอซี 7493

เอาต์พุท		อินพุท			
RO(1)	RO(2)	QD	QC	QB	QA
H	H	L	L	L	L
L	X	COUNT			
X	L	COUNT			

ตารางที่ 13.2 ตารางความจริงของฟังก์ชั่นการรีเซทวงจรนับของไอซี 7493

ไอซีเบอร์ 7493 มีแนนด์เกทอยู่ภายในสำหรับสร้างวงจรรีเซทเพื่อใช้สร้างวงจรนับโมดูโลแบบอื่นๆ เช่น วงจรนับแบบ MOD-10 ซึ่งสามารถแสดงดังรูปที่ 13.2 การนับของวงจรนับ MOD-10 นั้นจะเริ่มต้นนับจาก 0 (0000) ไปจนถึง 9(1001) และเมื่อวงจรนับมาถึงค่าถัดไปคือ 10(1010) เราจะต้องนำค่าลอจิกของเอาต์พุทในขณะนี้ไปรีเซทวงจรนับเพื่อให้มันกลับไปเริ่มต้นเป็นค่า 0000 ใหม่อีกครั้ง จากตารางที่ 13.2 จะเห็นได้ว่าเราจะต้องเลือกบิตที่เป็น 1 มาทำเป็นสัญญาณรีเซท ซึ่งในกรณีนี้ต้องใช้บิต 3 และบิต 1 (จาก 1010) มาต่อเข้าที่ขา R0(1) และ R0(2) เพื่อสร้างเป็นสัญญาณรีเซท

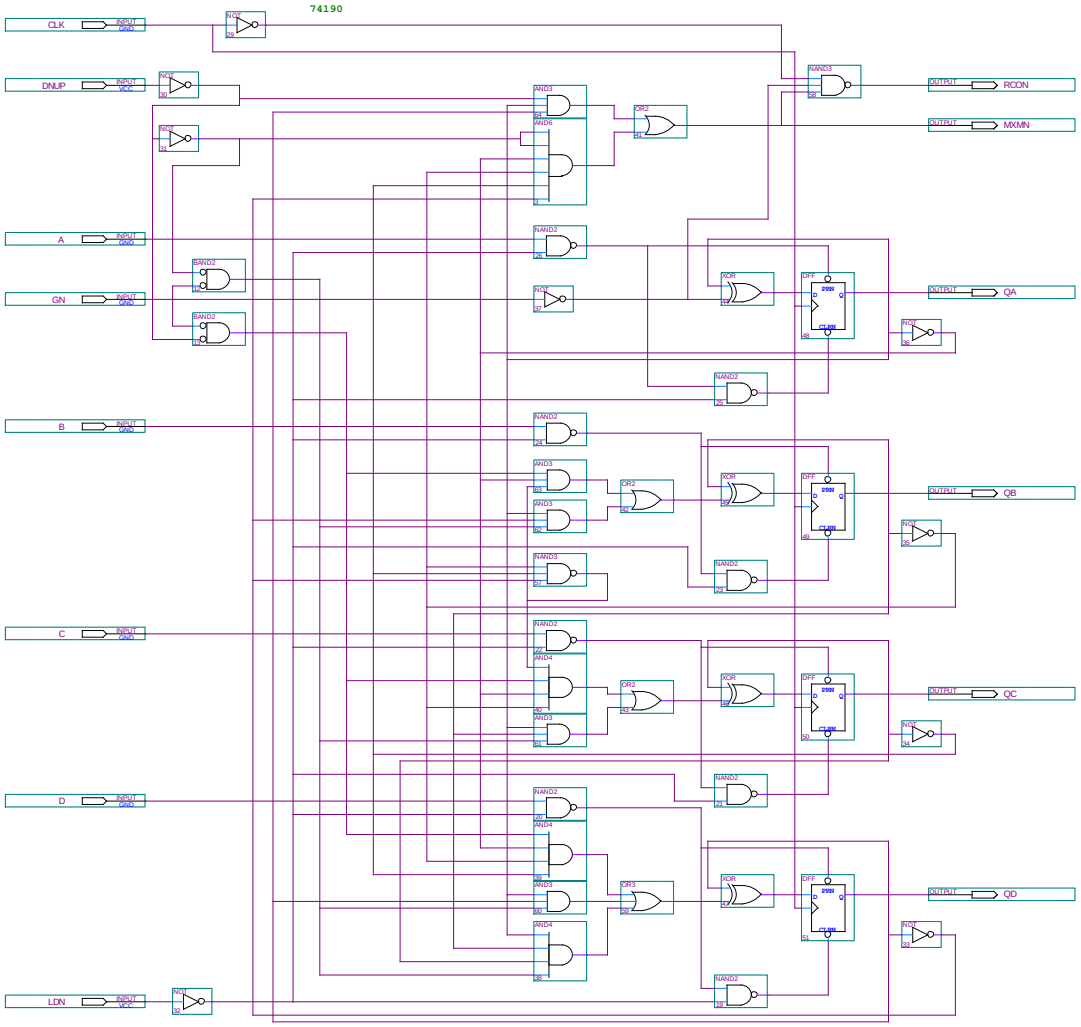
ไอซีนี้ทำหน้าที่เป็นวงจรนับอีกตัวหนึ่งที่จะศึกษาในการทดลองนี้คือ ไอซีเบอร์ 74190 ซึ่งเป็นไอซีวงจรนับ BCD (BCD Counter) โดยมีวงจรภายในเป็นวงจรนับแบบซิงโครนัส สามารถเซทให้นับขึ้น-ลงได้โดยการเซทค่าลอจิกที่ขาอินพุท down/up โดยเมื่อป้อนลอจิก 0 มันจะทำงานเป็นวงจรนับขึ้น และถ้าป้อน 1 มันจะทำงานเป็นวงจรนับลง



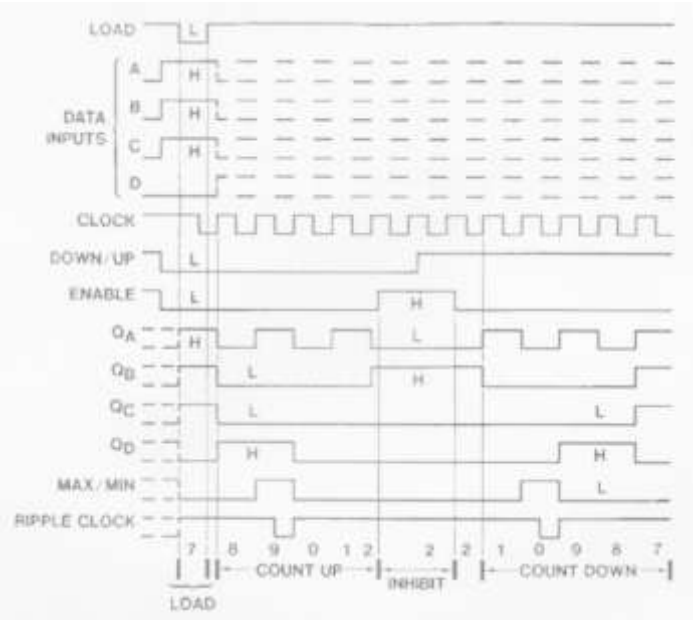
รูปที่ 13.3 ไอซี 74190

ไอซีตัวนี้สามารถเลือกจัดการทำงานได้ทุกอย่าง นั่นคือ เราสามารถเซทให้เอาต์พุทได้โดยการป้อนลอจิก 0 เข้าไปที่ขา LOAD และใส่ค่าเอาต์พุทที่ต้องการเข้าทางขาอินพุท DATA (A,B,C และ D) ซึ่งจะทำให้ค่าทางเอาต์พุทเปลี่ยนแปลงทันทีโดยไม่ต้องรอสัญญาณนาฬิกา

นอกจากนี้ยังมีขาเอาต์พุท 2 ขา สำหรับใช้เชื่อมต่อเพื่อเพิ่มจำนวนหลักในการนับ (Cascade) ได้แก่ขา Ripple Clock และ MAX/MIN Count โดยที่ขา MAX/MIN จะผลิตสัญญาณพัลส์ที่จะมีค่าเป็นลอจิก 1 มีช่วงเวลาเท่ากับสัญญาณนาฬิกา 1 ลูก เมื่อการนับมาถึงค่าสูงสุดในกรณีนับขึ้น และเมื่อการนับมาถึงค่าต่ำสุดในกรณีนับลง ส่วนขา Ripple Clock จะผลิตสัญญาณพัลส์ที่มีลอจิก 0 ในช่วงเลาเท่ากับส่วนของสัญญาณนาฬิกาที่เป็นลอจิก 0 เมื่อการนับมาถึงค่าสูงสุดในกรณีนับขึ้น และเมื่อมีการนับมาถึงค่าต่ำสุดในกรณีนับลง ในการต่อเพิ่มไอซีเพื่อให้ในการนับได้จำนวนหลักที่มากขึ้น เราสามารถนำขาเอาต์พุท Ripple Clock นี้ไปเชื่อมต่อกับขา Enable ของไอซีตัวถัดไปในกรณีที่ต่อขาสัญญาณนาฬิการ่วมกัน หรืออาจต่อขา Ripple Clock ไปที่ขา Clock ของไอซีตัวถัดไปในกรณีที่ไอซีแต่ละตัวต่อขา Enable ร่วมกันก็ได้



รูปที่ 13.4 วงจรภายในของไอซี 74190



รูปที่ 13.5 รูปแบบการทำงานของไอซี 74190

ขั้นตอนการทดลอง ตอนที่ 1 การใช้งานไอซี 7493 เป็นวงจรนับ MOD-10

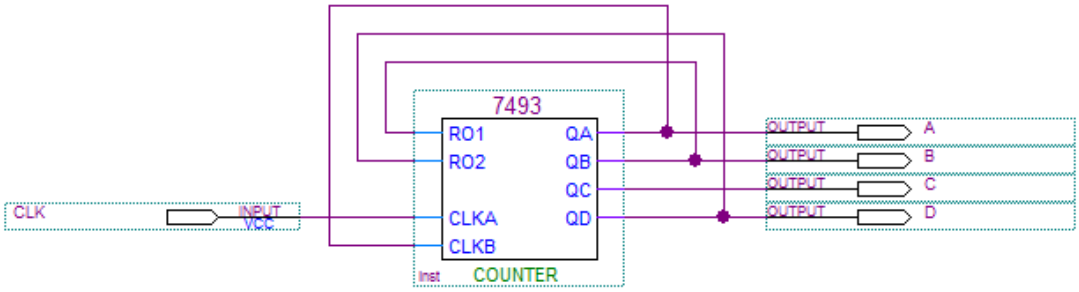
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_13a โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 7493, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

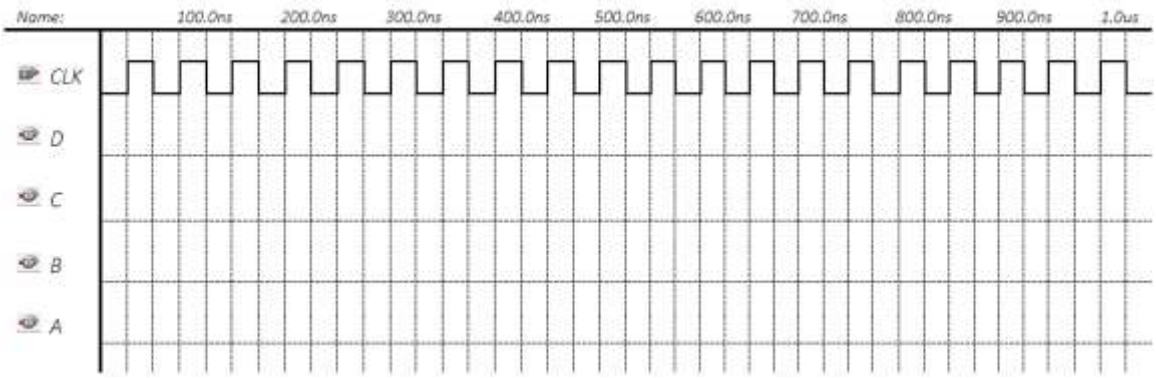


รูปที่ 13.6 วงจรสำหรับการทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_13a.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_13a.scf โดยตั้งค่า Grid Size เท่ากับ 50ns เพื่อจำลองการทำงานของวงจร



รูปที่ 13.7 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

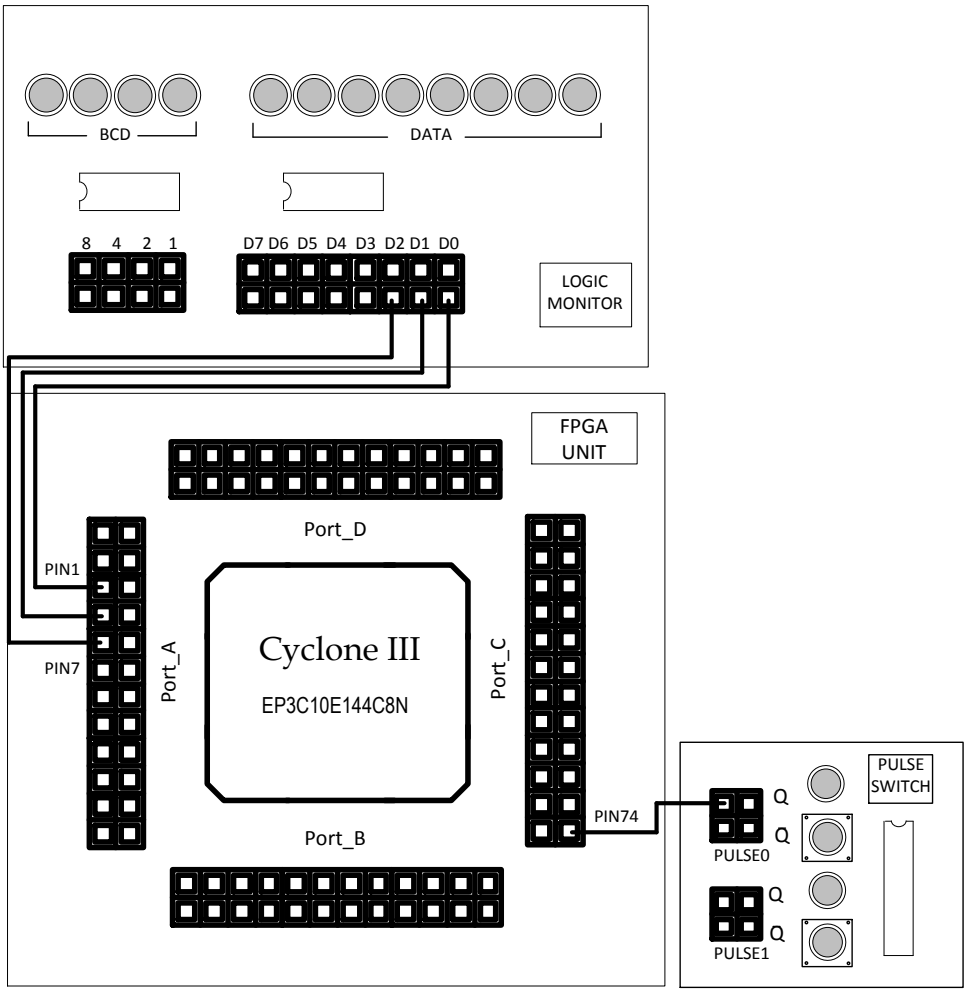
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	**ตำแหน่งขาบน FPGA**
CLK	74
A	1
B	3
C	7
D	11

ตารางที่ 13.3 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 13.8 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 13.8 การต่อวงจรบนชุดทดลอง

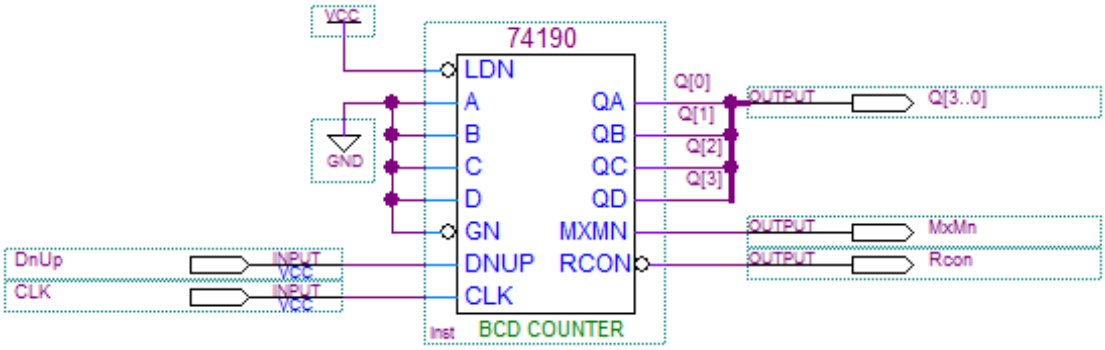
13. ทดสอบวงจรโดยทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0, LED D1 และ LED D2 จากนั้นบันทึกผลลงในตาราง

CLK (PULSEO_Q)	D (LED D3)	C (LED D2)	B (LED D1)	A (LED D0)
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				
↑				

ตารางที่ 13.4 ตารางความจริงจากการทดลองตอนที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 การใช้งานไอซี 74190

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_13b โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 74190, vcc, gnd, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

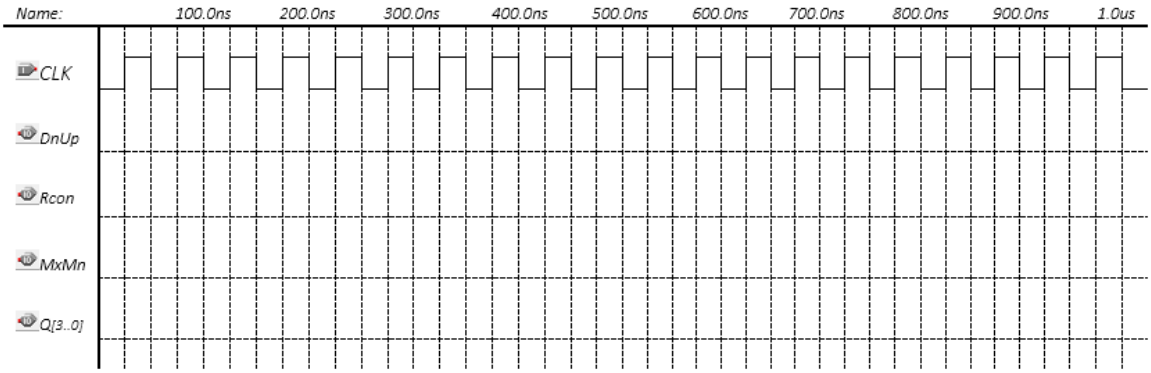


รูปที่ 13.9 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_13b.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

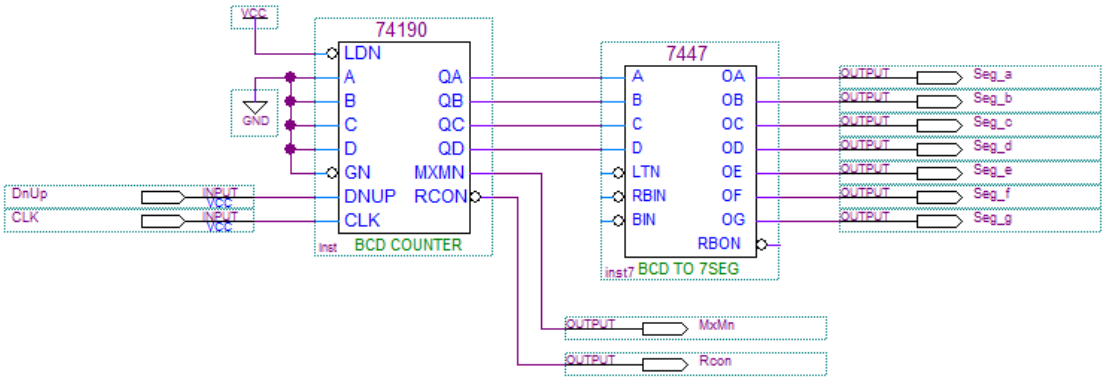
8. สร้างไฟล์ Vector Waveform ในชื่อ lab_13b.scf โดยตั้งค่า Grid Size เท่ากับ 50ns เพื่อจำลองการทำงานของวงจร



รูปที่ 13.10 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

10. เพิ่มวงจรถอดรหัส 7447 เพื่อแปลงรหัส BCD ให้กลายเป็นตัวเลขดังรูป



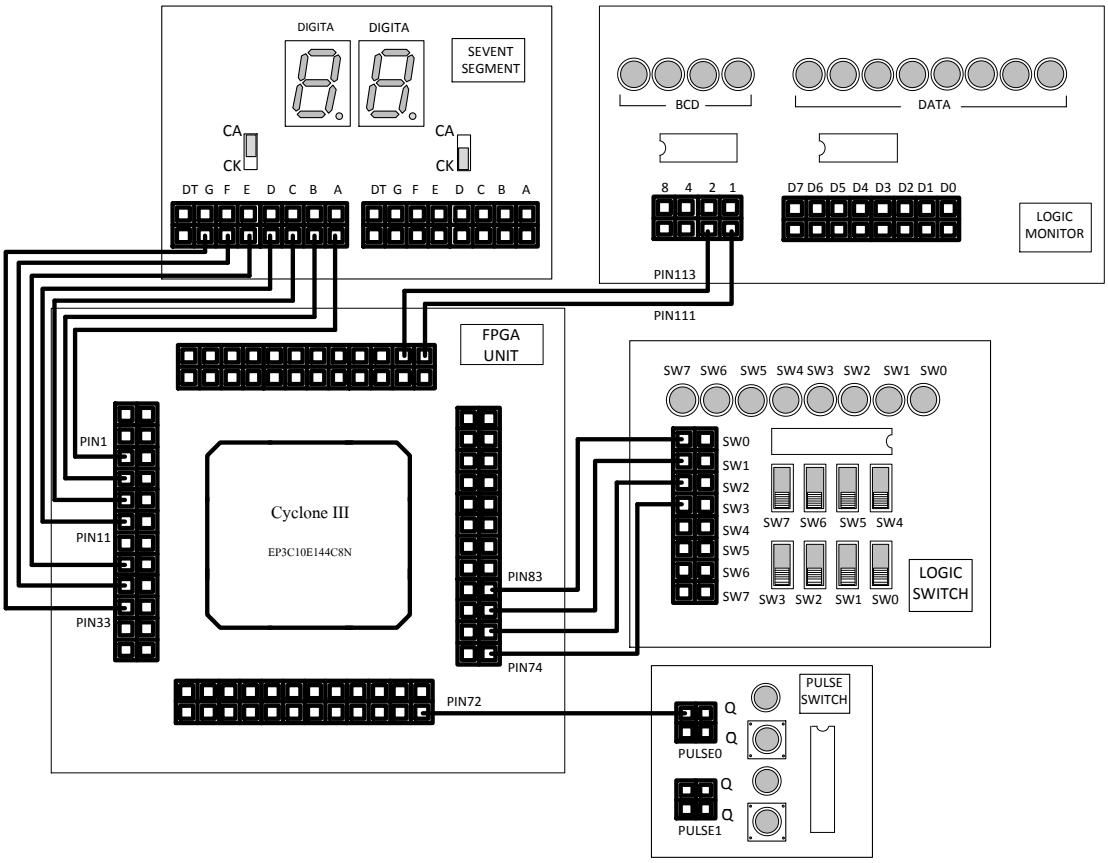
รูปที่ 13.11 วงจรสำหรับการทดลองตอนที่ 2

10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins
11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
CLK	72	Seg_c	7
DnUp	74	Seg_d	11
MxMn	111	Seg_e	28
Rcon	113	Seg_f	31
Seg_a	1	Seg_g	33
Seg_b	3		

ตารางที่ 3.5 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 13.12 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0 และ ทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0, LED D1 และ SEVENT SEGMENT DIGIT A จากนั้นบันทึกผลลงในตาราง

DnUp (SW0)	CLK (PULSEO_Q)	C (LED D2)	B (LED D1)	รูปแบบ LED บน SEVENT SEGMENT
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
0	↑			
1	↑			
1	↑			
1	↑			
1	↑			
1	↑			
1	↑			
1	↑			
1	↑			
1	↑			
1	↑			
1	↑			
1	↑			

DnUp (SW0)	CLK (PULSEO_Q)	C (LED D2)	B (LED D1)	รูปแบบ LED บน SEVENT SEGMENT
1	↑			
1	↑			
1	↑			

ตารางที่ 3.6 ตารางความจริงจากการทดลองตอนที่ 2

สรุปผลการทดลอง

การทดลองที่ 14

วงจรเลื่อนข้อมูล (Shift Register)

วัตถุประสงค์

1. เรียนรู้การใช้งานโปรแกรม Quartus II 8.1 Web Edition ในการสร้างแผนผังวงจรดิจิตอล สร้างรูปคลื่น คอมไพล์ และจำลอง การทงานของวงจร
2. เพื่อเรียนรู้การทำงานของวงจรวงจรเลื่อนข้อมูล
3. วิเคราะห์รูปคลื่น สร้างตารางความจริงจากวงจรวงจรเลื่อนข้อมูลได้
4. สามารถดาวน์โหลดวงจรลงไอซี FPGA เพื่อทดสอบการทำงานจริงของวงจรได้

อุปกรณ์สำหรับการทดลอง

1. ชุดทดลองการเรียนรู้เทคโนโลยีไอซี FPGA
2. สาย USB Blaster
3. สายไฟสำหรับต่อวงจร
4. โปรแกรม Quartus II 8.1 Web Edition

ทฤษฎี

วงจรเลื่อนข้อมูล (Shift Register) เป็นวงจรที่มีบทบาทสำคัญในการสื่อสารข้อมูลในระบบดิจิตอล รวมถึงการคำนวณทางคณิตศาสตร์ อุปกรณ์หลักที่ใช้สร้างวงจรเลื่อนข้อมูลคือ รีจิสเตอร์ (Register) ซึ่งเป็นตัวเก็บข้อมูลที่สร้างจากฟลิปฟลอบ ซึ่งอาจเป็น D ฟลิปฟลอปหรือ JK ฟลิปฟลอปก็ได้ ฟลิปฟลอปแต่ละตัวจะเก็บข้อมูลได้ตัวละ 1 บิต ดังนั้นขนาดของรีจิสเตอร์จึงขึ้นอยู่กับจำนวนฟลิปฟลอปที่ใช้

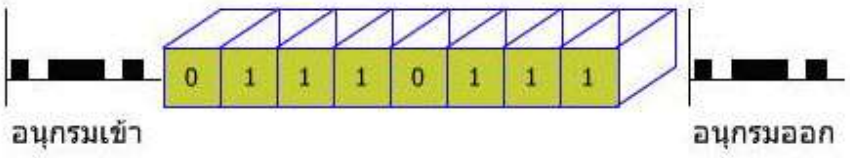
วงจรเลื่อนข้อมูลแบ่งออกเป็น 4 ชนิดได้แก่

1.วงจรเลื่อนข้อมูลแบบอนุกรมเข้า-อนุกรมออก (Serial ln/Serial Out : SISO)

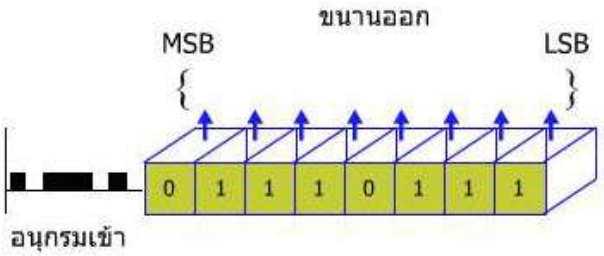
2.วงจรเลื่อนข้อมูลแบบอนุกรมเข้า-ขนานออก (Serial ln/Parallel Out : SIPO)

3.วงจรเลื่อนข้อมูลแบบขนานเข้า-อนุกรมออก (Parallel ln/Serial Out : PISO)

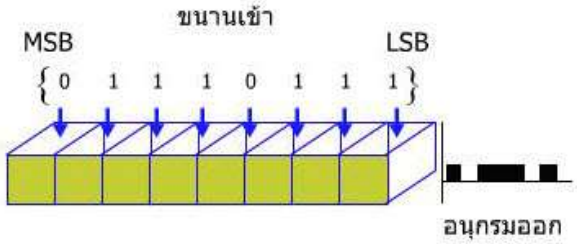
4. 3.วงจรเลื่อนข้อมูลแบบขนานเข้า-ขนานออก (Parallel ln/ Parallel Out : PIPO)



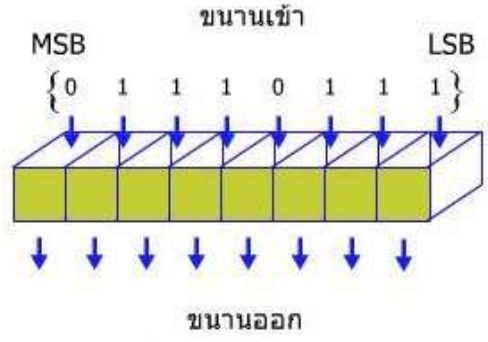
รูปที่ 14.1a วงจรเลื่อนข้อมูลแบบอนุกรมเข้า-อนุกรมออก



รูปที่ 14.1b วงจรเลื่อนข้อมูลแบบอนุกรมเข้า-ขนานออก



รูปที่ 14.1c วงจรเลื่อนข้อมูลแบบขนานเข้า-อนุกรมออก



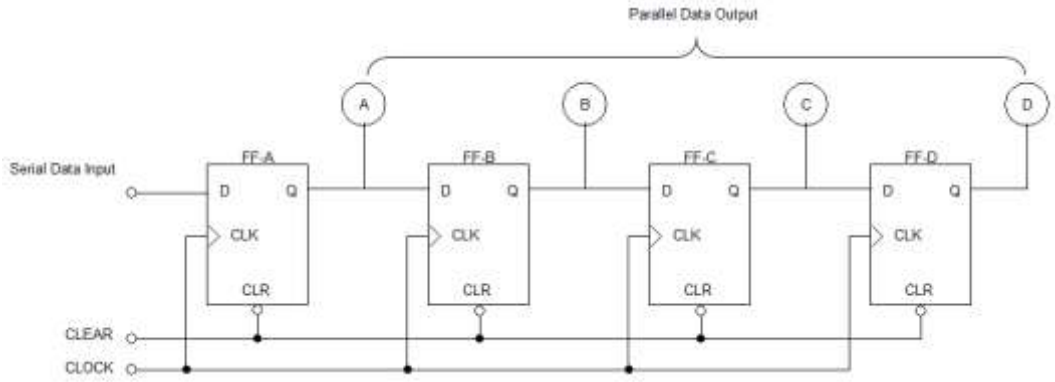
รูปที่ 14.1d วงจรเลื่อนข้อมูลแบบขนานเข้า-ขนานออก

ตัวอย่างวงจรเลื่อนข้อมูลแบบพื้นฐานแสดงดังรูปที่ 14.2 ซึ่งเป็นวงจรเลื่อนข้อมูลแบบอนุกรมเข้า-ขนานออก ขนาด 4 บิต การทำงานเริ่มจากส่งสัญญาณ Clear เพื่อกำหนดให้ฟลิปฟลอปทุกตัวมีค่าข้อมูลเป็น 0 ที่เวลา t0 และให้ขาData มีค่าเป็น 1 เมื่อขอบขาขึ้นของสัญญาณนาฬิกามาถึงที่เวลา t1 ข้อมูล 1 จะถูกป้อนเข้าไปเก็บไว้ในฟลิปฟลอป A และข้อมูลเดิมที่มีอยู่ในฟลิปฟลอป A จะถูกส่งต่อไปให้ฟลิปฟลอป B ส่วนข้อมูลเดิมของฟลิปฟลอป B ก็จะถูกเลื่อนไปเก็บไว้ในฟลิปฟลอป C และฟลิปฟลอป C จะส่งข้อมูลเดิมของมันไปเก็บที่ฟลิปฟลอป D เช่นกัน ดังนั้นขณะนี้ค่าเอาต์พุทของวงจรจะเป็น 0001

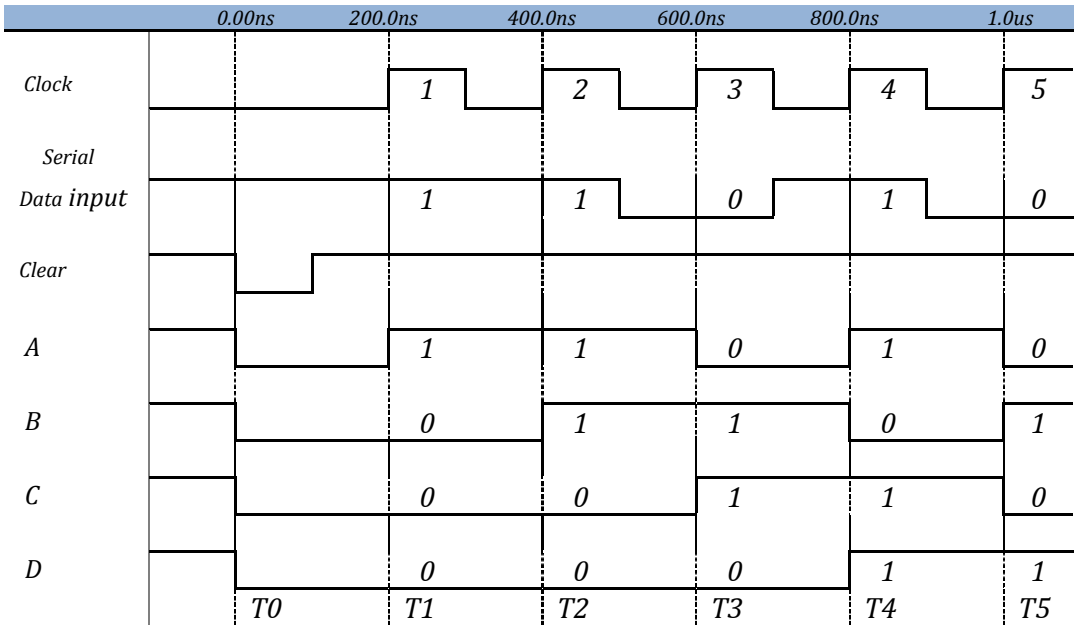
เมื่อถึงเวลา t2 ขา Data ยังมีค่าเป็น 1 ซึ่งจะถูกเลื่อนเข้าไปเก็บไว้ใน ฟลิปฟลอป A และข้อมูลเดิมของฟลิปฟลอปแต่ละตัวจะถูกเลื่อนไปเก็บยังตัวถัดไปเช่นเดิม ซึ่งค่าเอาต์พุทในขณะนี้จะมีค่าเป็น 0011

เมื่อถึงเวลา t3 ขณะนี้ขา Data เปลี่ยนลอจิก 1 เป็น 0 ข้อมูลจะเลื่อนเข้าไปและเอาต์พุทของวงจรขณะนี้จะมีค่าเป็น 0110

เมื่อถึงเวลา t4 ขา Data เปลี่ยนค่าลอจิกเป็น 1 ข้อมูลจะเลื่อนเข้าไปและเอาต์พุทของวงจรขณะนี้จะมีค่าเป็น 1101

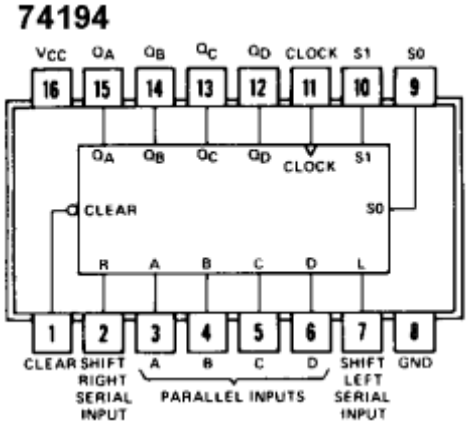


รูปที่ 14.2 วงจรเลื่อนข้อมูล



รูปที่ 14.3 รูปคลื่นการทำงานของวงจรเลื่อนข้อมูล

ไอซีที่ทำหน้าที่เป็นวงจรเลื่อนข้อมูลที่นิยมใช้กันมากได้แก่ เบอร์ 74194 (4 Bit-Universal Shift Register) เนื่องจากสามารถสร้างเป็นวงจรเลื่อนข้อมูลได้หลายแบบ เช่น วงจรเลื่อนข้อมูลแบบขนานเข้า-ขนานออก (PIPO), อนุกรมเข้า-ขนานออก (SIPO) และ อนุกรมเข้า-อนุกรมออก (SISO) เป็นต้น นอกจากนี้ยังสามารถต่อขยายให้เป็นวงจรเลื่อนข้อมูลมากกว่า 4 บิตได้



รูปที่ 14.4 ไอซี 74194

ไอซี 74194 เป็นไอซีที่ทำหน้าที่ขอบขาขึ้นของสัญญาณนาฬิกา โดยที่โหมดการทำงานของ 74194 แสดงดังรูป 14.1 ซึ่งมีทั้งหมด 5 โหมด ได้แก่ รีเซท, โฮลด์ (Hold) หรือคงสภาวะเดิม, เลื่อนซ้าย, เลื่อนขวา และ โหมดข้อมูลขนาน (Parallel Load)

โหมดรีเซทจะเกิดขึ้นเมื่อมีการป้อนลอจิก 0 เข้าที่ขา CLEAR ซึ่งจะทำให้ค่าเอาต์พุทที่ถูกเคลียร์ค่าเป็น 0000 (QA,QB,QC,QD) ทั้งหมดโดยไม่สนว่าค่าลอจิกทางขาข้อมูลอินพุทจะมีค่าเป็นอะไร

สำหรับโหมดที่เหลืออีก 4 โหมดจะถูกควบคุมด้วยการป้อนลอจิกเข้าที่ขา S0 และ S1 ดังนี้

-เมื่อขา S0 = 0 และ S1 = 0 วงจรเลื่อนข้อมูลจะเข้าสู่โหมดโฮลด์หรือก็คือการคงสภาวะข้อมูลทางเอาต์พุท

-โหมดเลื่อนซ้ายจะเกิดขึ้นเมื่อขา S0 = 0 และ S1 = 1 ข้อมูลจะถูกป้อนเข้าที่ขา Serial Input (Shift Left) หรือ DSL โดยข้อมูลจะถูกโหลดเข้าไปที่ QD และข้อมูลเดิมจะถูกเลื่อนซ้ายไปยัง QA เมื่อมีการกระตุ้นของสัญญาณนาฬิกาที่ขอบขาขึ้น

-โหมดเลื่อนขวาจะเกิดขึ้นเมื่อขา S0 = 1 และ S1 = 0 ข้อมูลจะถูกป้อนเข้าที่ขา Serial Input (Shift Right) หรือ DSR และเมื่อ การกระตุ้นของสัญญาณนาฬิกาที่ขอบขาขึ้น ข้อมูลจะถูกโหลดเข้าไปที่ QA และข้อมูลเดิมจะถูกเลื่อนไปทาง QD

-โหมดสุดท้ายคือโหมดโหลดขนาน หรือบางครั้งเรียกว่า Broadside Load จะเกิดขึ้นเมื่อขา S0 = 1 และ S1 = 1 ข้อมูลจากขาอินพุทแบบขนานจะถูกโหลดเข้าไปยังเอาต์พุทเมื่อถูกกระตุ้นด้วยสัญญาณนาฬิกาขาขึ้น

ในการใช้งาน 74194 นั้นสามารถประยุกต์ได้หลากหลาย ไม่ว่าจะเป็นการโหลดข้อมูลอินพุทแบบขนาน หรือ อนุกรมก็ได้ และการอ่านข้อมูลเอาต์พุทก็สามารถอ่านได้ทั้งแบบขนานและอนุกรมเช่นกัน (อ่านจากขาข้อมูลเดียว เช่น QD เป็นต้น)

อินพุท											*เอาต์พุท*			
MODE	*CLEAR*	*MODE*		*CLK*	*SERIAL*		*PARALLFL*				*QA*	*QB*	*QC*	*QD*
		S1	*S0*		*LEFT*	*RIGHT*	*A*	*B*	*C*	*D*				
Hold	*H*	*X*	*X*	*L*	*X*	*X*	*X*	*X*	*X*	*X*	*QA0*	*QB0*	*QC0*	*QD0*
Reset	*L*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*L*	*L*	*L*	*L*
Hold	*H*	*L*	*L*	*X*	*X*	*X*	*X*	*X*	*X*	*X*	*QA0*	*QB0*	*QC0*	*QD0*
Shift Left	*H*	*H*	*L*	↑	*H*	*X*	*X*	*X*	*X*	*X*	*QBn*	*QCn*	*QDn*	*H*
	H	*H*	*L*	↑	*L*	*X*	*X*	*X*	*X*	*X*	*QBn*	*QCn*	*QDn*	*L*
Shift Right	*H*	*L*	*H*	↑	*X*	*H*	*X*	*X*	*X*	*X*	*H*	*QAn*	*QBn*	*QCn*
	H	*L*	*H*	↑	*X*	*L*	*X*	*X*	*X*	*X*	*L*	*QAn*	*QBn*	*QCn*
Parallel Load	*H*	*H*	*H*	↑	*X*	*X*	*a*	*b*	*c*	*d*	*a*	*b*	*c*	*d*

ตารางที่ 14.1 ตารางความจริงของ 74149

X : Don’t care

a, b, c, d : ระดับลอจิกที่คงที่ที่ป้อนเข้าที่ขา A, B, C และ D ตามลำดับ

QA0, QB0, QC0, QD0 : เอาต์พุทไม่มีการเปลี่ยนแปลง

QAn, QBn, QCn, QDn : ระดับลอจิกของ QA, QB, QC และ QD หลังจากถูกกระตุ้นด้วยสัญญาณนาฬิกา

ขั้นตอนการทดลอง ตอนที่ 1 วงจรเลื่อนข้อมูลแบบใช้ฟลิปฟลอป

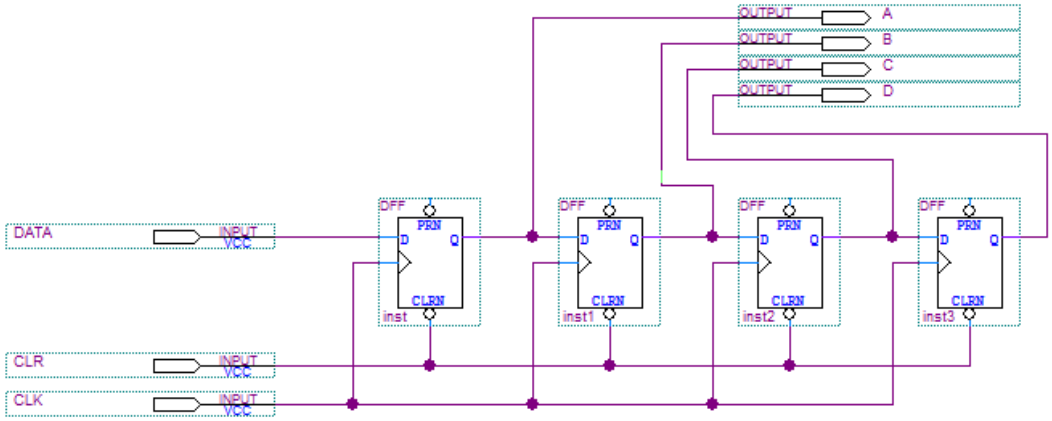
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_14a โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ dff, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

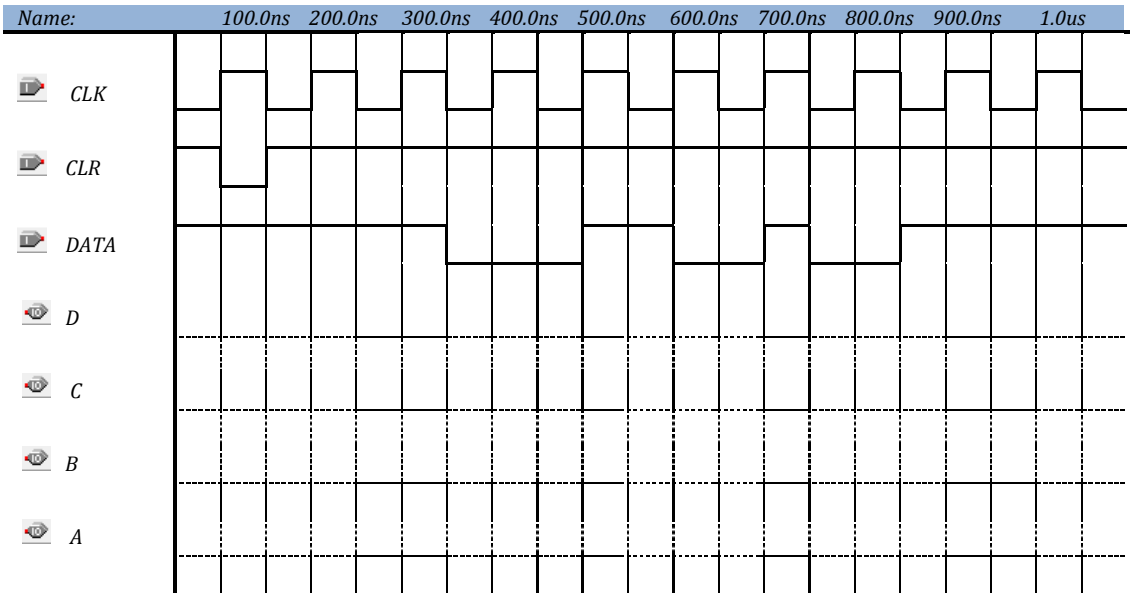


รูปที่ 14.5 วงจรสำหรับการทดลองตอนที่ 1

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_14a.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_14a.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 14.6 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

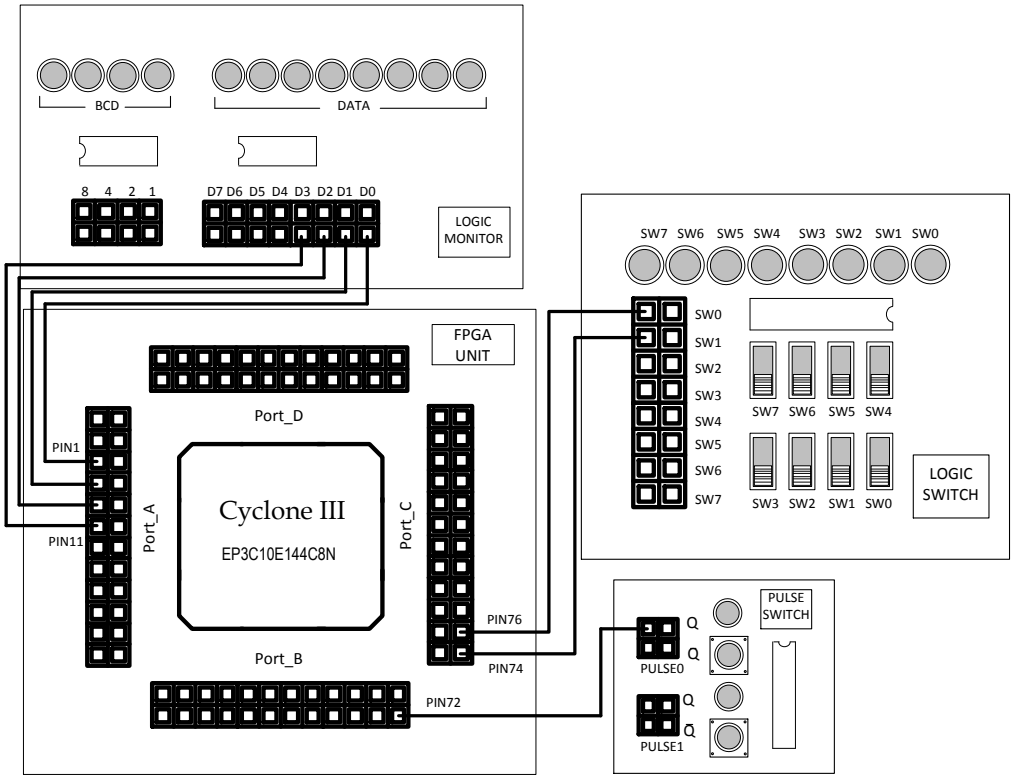
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
A	1	CLK	72
B	3	CLR	76
C	7	DATA	74
D	11		

ตารางที่ 14.2 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 14.7 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 14.7 การต่อวงจรบนชุดทดลอง

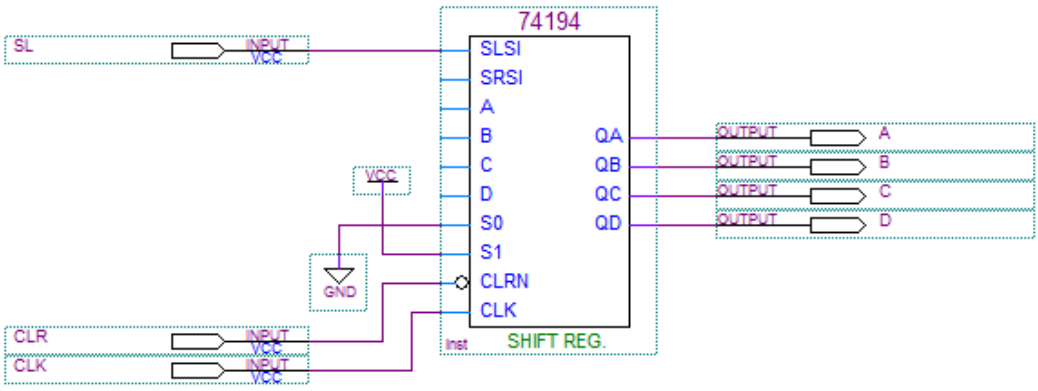
13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, และ SW1 ทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0, LED D1, LED D2, และ LED D3 จากนั้นบันทึกผลลงในตาราง

PRE_N (SW0)	CLR_N (SW1)	CLK (PULSEO Q)	D (LED D3)	C (LED D2)	B (LED D1)	A (LED D0)
HIGH	HIGH	↑				
HIGH	HIGH	↑				
HIGH	HIGH	↑				
HIGH	HIGH	↑				
LOW	HIGH	↑				
LOW	HIGH	↑				
LOW	HIGH	↑				
LOW	HIGH	↑				

ตารางที่ 14.3 ตารางความจริงจากการทดลองตอนที่ 1

ขั้นตอนการทดลอง ตอนที่ 2 การใช้ไอซี 74149 เป็นวงจรเลื่อนซ้ายแบบ SISO และ SIPO

1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_14b โดยใช้เมนู File -> New Project Wizard
2. สร้างไฟล์สำหรับวาดแผนวงจรโดยใช้เมนู File -> New
3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK
4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 74194, vcc, gnd, input และ output
5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

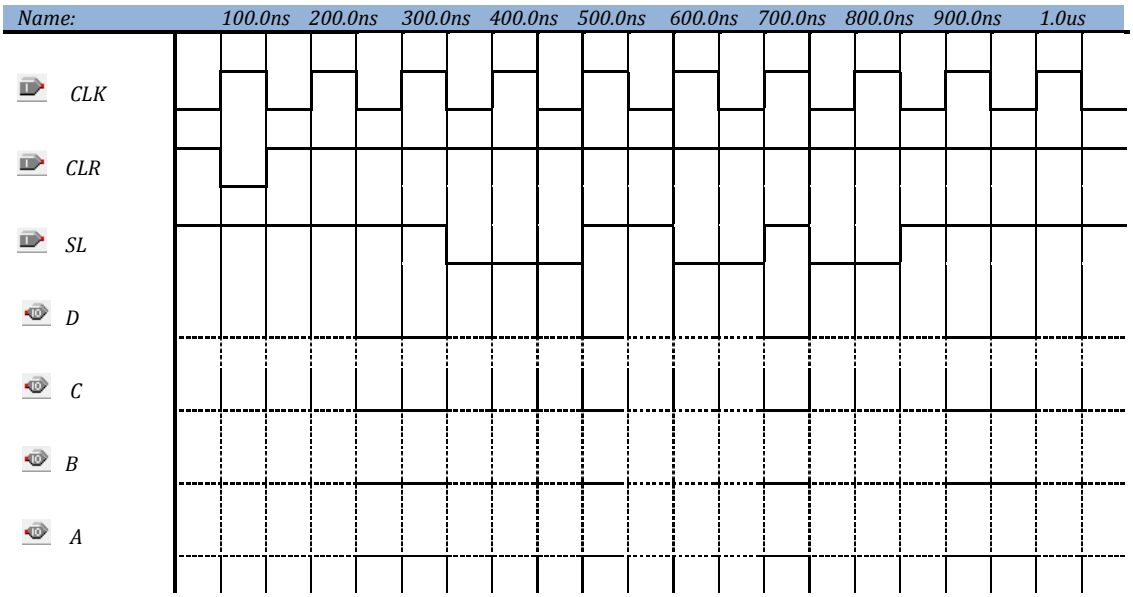


รูปที่ 14.8 วงจรสำหรับการทดลองตอนที่ 2

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_14b.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_14b.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 14.9 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจร โดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

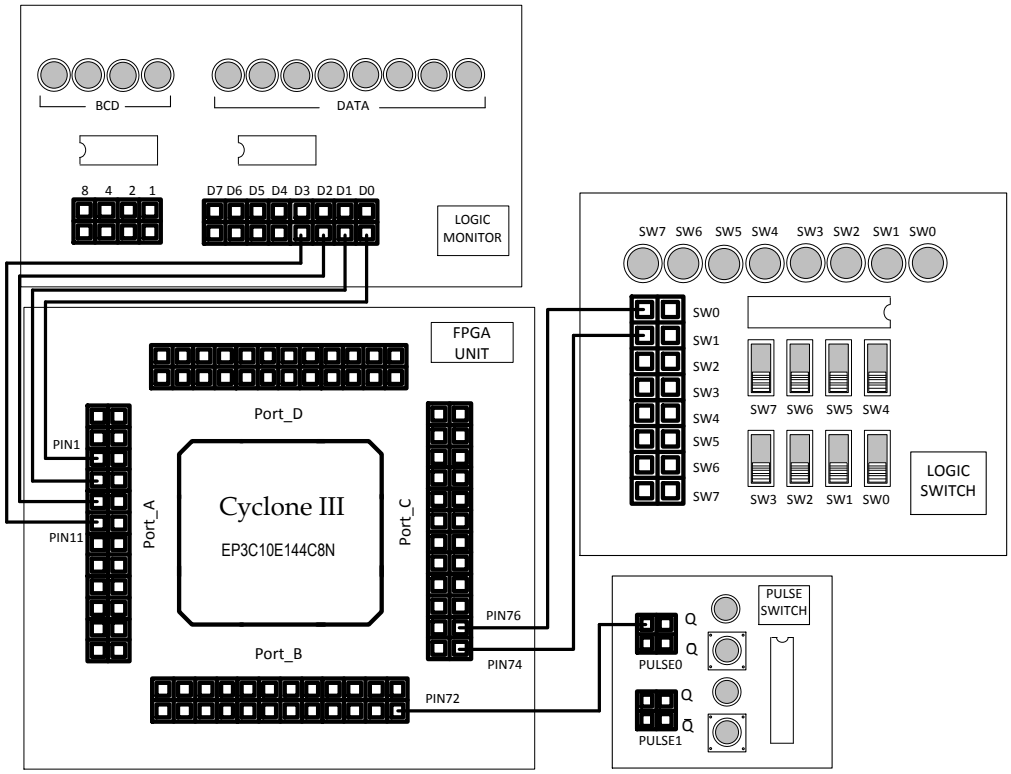
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
A	1	CLK	72
B	3	CLR	76
C	7	SL	74
D	11		

ตารางที่ 14.4 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 14.10 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 14.10 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, และ SW1 ทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0, LED D1, LED D2 และ LED D3 จากนั้นบันทึกผลลงในตาราง

CLR (SW0)	**SL (SW1)**	**CLK (PULSEO Q)**	**D (LED D3)**	**C (LED D2)**	**B (LED D1)**	**A (LED D0)**
LOW	HIGH	↑				
HIGH	HIGH	↑				
HIGH	HIGH	↑				
HIGH	LOW	↑				
HIGH	LOW	↑				
HIGH	LOW	↑				
HIGH	HIGH	↑				
HIGH	LOW	↑				
HIGH	LOW	↑				

ตารางที่ 14.5 ตารางความจริงจากการทดลองตอนที่ 2

ขั้นตอนการทดลอง ตอนที่ 3 การใช้ไอซี 74149 เป็นวงจรเลื่อนขวาแบบ SISO และ SIPO

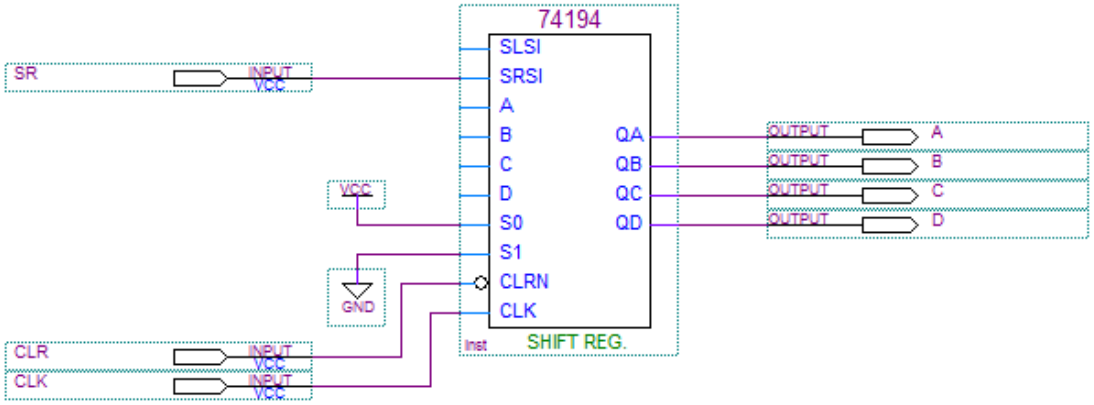
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_14c โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจร โดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 74194, vcc, gnd, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

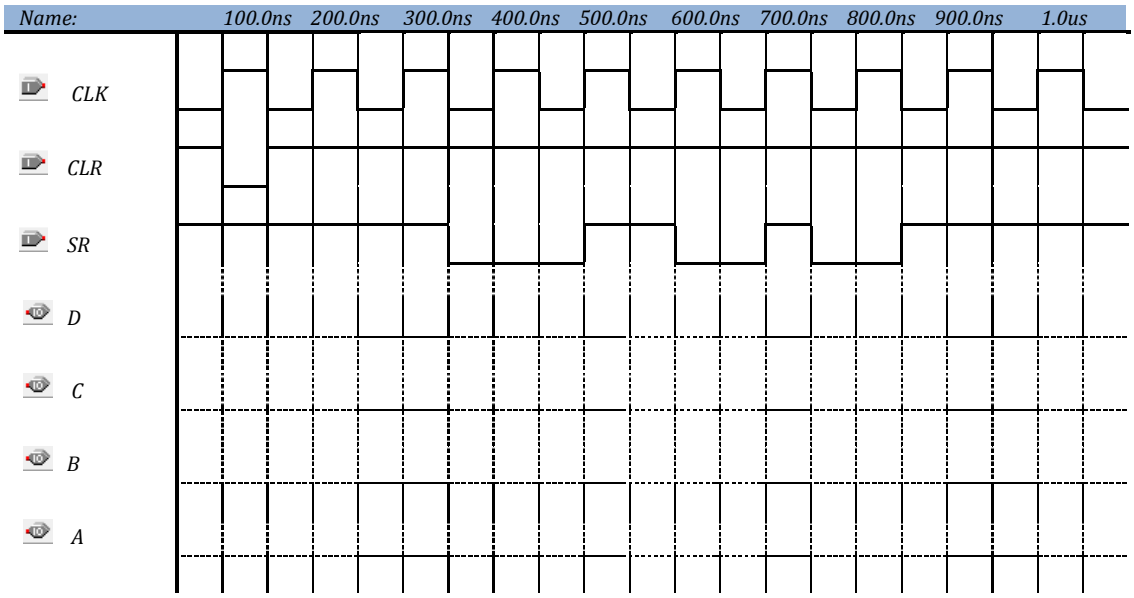


รูปที่ 14.11 วงจรสำหรับการทดลองตอนที่ 3

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_14c.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_14c.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 14.12 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

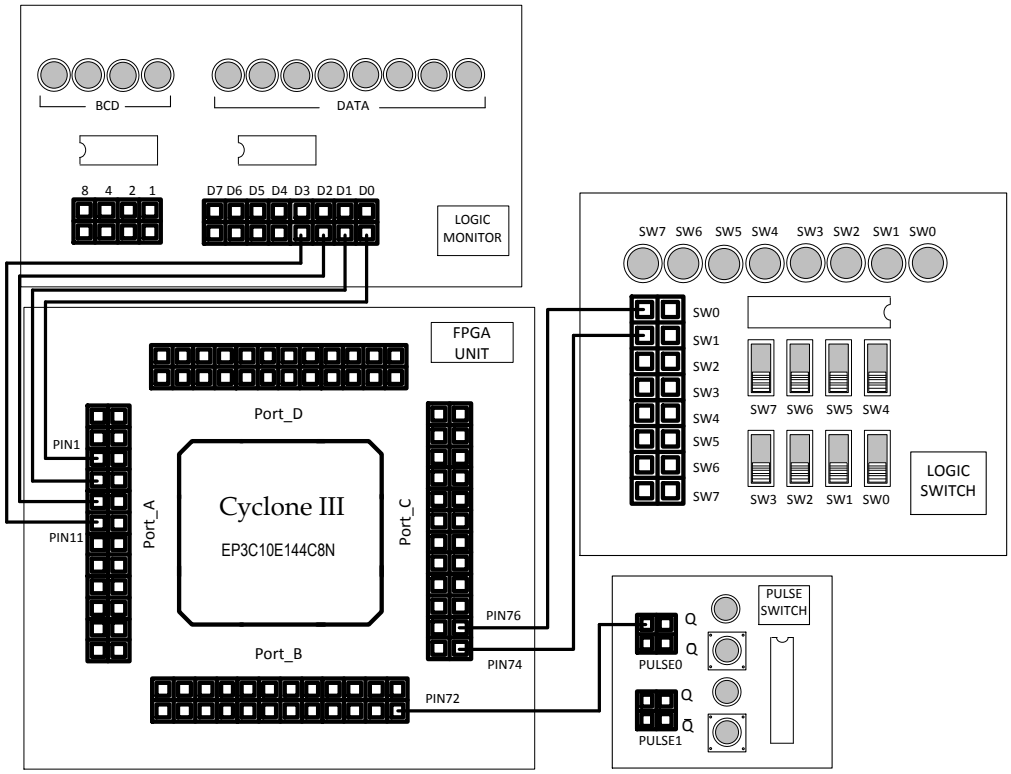
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
A	1	CLK	72
B	3	CLR	76
C	7	SR	74
D	11		

ตารางที่ 14.6 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 14.13 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 14.13 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, และ SW1 ทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0, LED D1, LED D2, และ LED D3 จากนั้นบันทึกผลลงในตาราง

CLR (SW0)	**SR (SW1)**	**CLK (PULSEO Q)**	**D (LED D3)**	**C (LED D2)**	**B (LED D1)**	**A (LED D0)**
LOW	HIGH	↑				
HIGH	HIGH	↑				
HIGH	HIGH	↑				
HIGH	LOW	↑				
HIGH	LOW	↑				
HIGH	LOW	↑				
HIGH	HIGH	↑				
HIGH	LOW	↑				
HIGH	LOW	↑				

ตารางที่ 4.7 ตารางความจริงจากการทดลองตอนที่ 3

ขั้นตอนการทดลอง ตอนที่ 4 การใช้ไอซี 74149 เป็นวงจรโหลดข้อมูลแบบขนาน

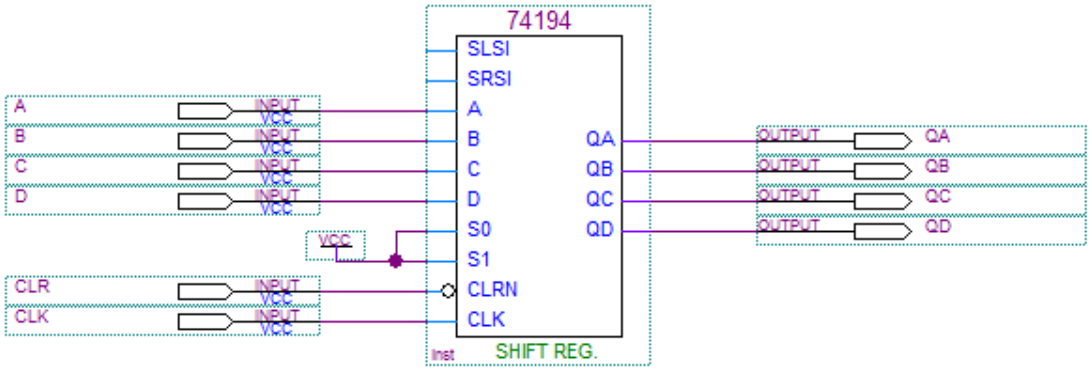
1. เปิดโปรแกรม Quartus II 8.1 Web Edition แล้วทำการสร้างโปรเจคใหม่ในชื่อ lab_14d โดยใช้เมนู File -> New Project Wizard

2. สร้างไฟล์สำหรับวาดแผนวงจร โดยใช้เมนู File -> New

3. จากนั้นหน้าต่าง New จะปรากฏขึ้น ให้เลือกที่ Block Diagram/Schematic File เสร็จแล้วคลิก OK

4. ทำการวาดอุปกรณ์ โดยดับเบิลคลิกบน Graphic Editor แล้วเลือกอุปกรณ์ 74194, vcc, input และ output

5. ลากเส้นเชื่อมต่ออุปกรณ์ต่าง ๆ เข้าด้วยกันดังรูป

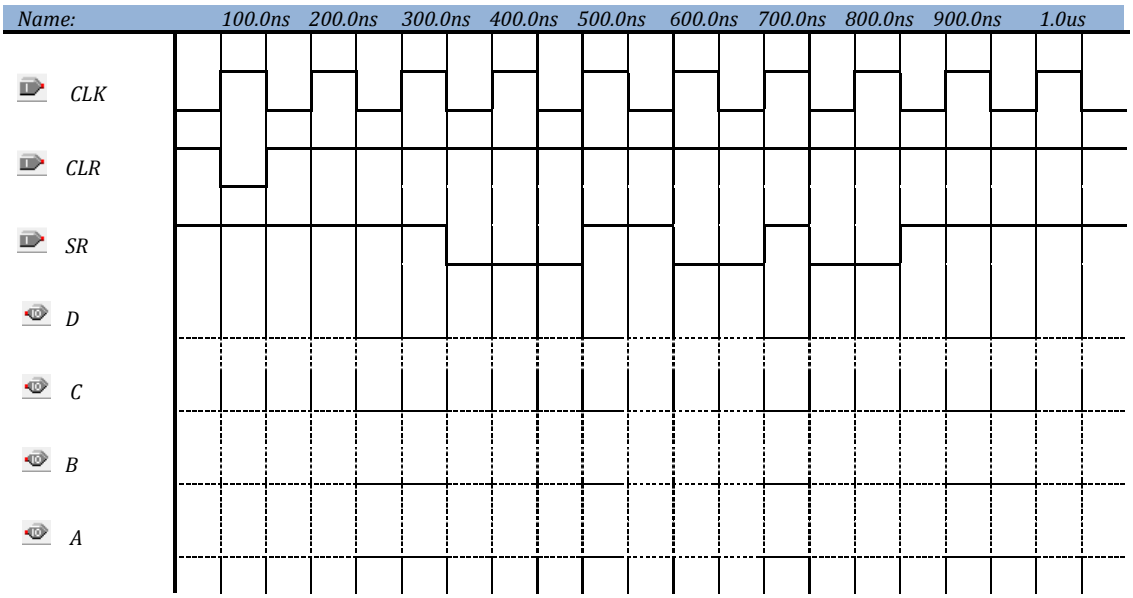


รูปที่ 14.14 วงจรสำหรับการทดลองตอนที่ 4

6. ให้ทำการบันทึกโดยบันทึกในชื่อ lab_14d.gdf

7. ทำการคอมไพล์เพื่อตรวจสอบวงจรที่วาดไว้ โดยใช้เมนู Processing -> Start compilation

8. สร้างไฟล์ Vector Waveform ในชื่อ lab_14d.scf เพื่อจำลองการทำงานของวงจร



รูปที่ 14.15 ผลจำลองการทำงาน

9. เรียกหน้าต่าง Simulator เพื่อคำนวณการจำลองการทำงานของวงจรโดยเลือกจากแถบเครื่องมือ หรือใช้การเลือกจากเมนู Processing -> Start Simulation

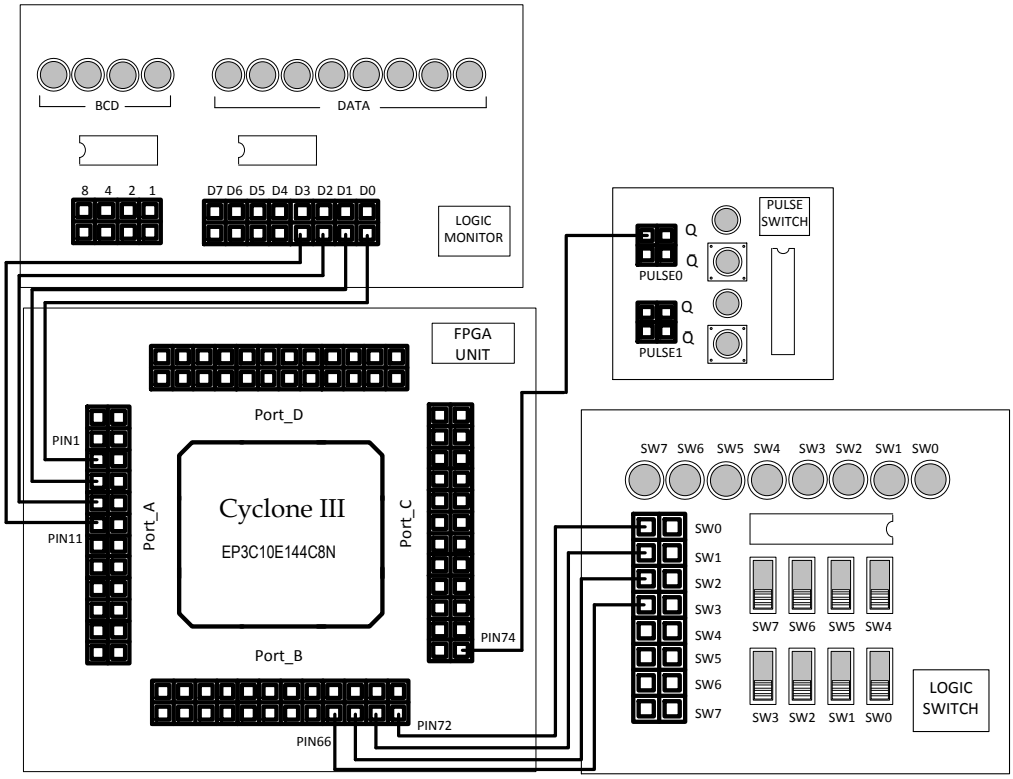
10. กำหนดขาจากหน้าต่าง Pin Planner โดยใช้เมนู Assignments -> Pins

11. เปลี่ยนตำแหน่งขาอินพุท และเอาต์พุทให้เป็นดังนี้

ขาในวงจร	ตำแหน่งขาบน FPGA	ขาในวงจร	ตำแหน่งขาบน FPGA
CLK	74	CLR	66
A	72	QA	1
B	70	QB	3
C	68	QC	7
D	64	QD	11

ตารางที่ 14.8 ตำแหน่งสัญญาณในวงจรและตำแหน่งขาบน FPGA

12. ต่อวงจรตามรูปที่ 14.16 แล้วดาวน์โหลดโปรแกรมลงไอซี FPGA



รูปที่ 14.16 การต่อวงจรบนชุดทดลอง

13. ทดสอบวงจรโดยการเปลี่ยนสถานะของสวิตช์ SW0, และ SW1 ทำการกดสวิทช์ PULSEO_Q แล้วดูผลทางลอจิกที่ LED D0, LED D1, LED D2, และ LED D3 จากนั้นบันทึกผลลงในตาราง

CLR (SW4)	A (SW0)	B (SW1)	C (SW2)	D (SW3)	CLK (PULSEO Q)	D (LED D3)	C (LED D2)	B (LED D1)	A (LED D0)
LOW	HIGH	HIGH	HIGH	HIGH	↑				
HIGH	LOW	LOW	LOW	LOW	↑				
HIGH	LOW	LOW	LOW	HIGH	↑				
HIGH	LOW	LOW	HIGH	LOW	↑				
HIGH	LOW	LOW	HIGH	HIGH	↑				
HIGH	LOW	HIGH	LOW	LOW	↑				
HIGH	LOW	HIGH	LOW	HIGH	↑				
HIGH	LOW	HIGH	HIGH	LOW	↑				
HIGH	LOW	HIGH	HIGH	HIGH	↑				
HIGH	HIGH	LOW	LOW	LOW	↑				
HIGH	HIGH	LOW	LOW	HIGH	↑				
HIGH	HIGH	LOW	HIGH	LOW	↑				
HIGH	HIGH	LOW	HIGH	HIGH	↑				
HIGH	HIGH	HIGH	LOW	LOW	↑				
HIGH	HIGH	HIGH	LOW	HIGH	↑				
HIGH	HIGH	HIGH	HIGH	LOW	↑				
HIGH	HIGH	HIGH	HIGH	HIGH	↑				

ตารางที่ 14.9 ตารางความจริงจากการทดลองตอนที่ 4

สรุปผลการทดลอง