

# คู่มือ Quartus II สำหรับการเริ่มต้น

## บทนำ

โปรแกรม QUARTUS II เป็นโปรแกรมของบริษัท ALTERA ([www.altera.com](http://www.altera.com)) ที่สร้างขึ้นมาเพื่อให้วิศวกรหรือนักพัฒนางานด้านออกแบบวงจรดิจิทัล ทำการสร้างแบบจำลองหรือวงจรทางด้านดิจิทัลที่มีความง่ายต่อการเขียนงานต้นแบบ (ใช้ได้ทั้งแบบกราฟิก logic diagram ,ภาษา VHDL, AHDL, Verilog, หรือเขียนเป็น state-diagram ) โดยผู้พัฒนางานสามารถทำการออกแบบ จำลองการทำงาน เพื่อทดสอบวงจร ในสถานการณ์ต่างๆ เสมือนจริงเพื่อค้นหาจุดบกพร่องของชิ้นงานต้นแบบก่อนที่จะสร้างชิ้นงานจริง เมื่อได้วงจรลอจิกต้นแบบที่พอใจแล้ว ก็จะนำไปสร้างชิ้นงานจริงด้วยการทำ configure โครงสร้างวงจรให้กับชิพ FPGA และทำการตรวจสอบชิ้นงานด้วยเครื่องมือ ออสซิลโลสโคปหรือเครื่องลอจิกอานาไลเซอร์ได้ ทำให้ Quartus II เป็นเครื่องมือสำหรับพัฒนาที่นิยมใช้อย่างแพร่หลายทั่วโลก

ปัจจุบันโปรแกรม QUARTUS II อยู่ในรุ่นเวอร์ชัน 15 (กลางปี 2015) แต่เวอร์ชันที่สามารถใช้งานได้และเหมาะสมกับระบบปฏิบัติการ Windows XP จะเป็นรุ่นที่ต่ำกว่าเวอร์ชัน 10 ทั้งนี้การจะติดตั้งใช้งานเวอร์ชันใดนั้นจะต้องขึ้นอยู่กับระบบปฏิบัติการที่เหมาะสม และบอร์ดทดลอง FPGA ที่มีอยู่ในห้องปฏิบัติการด้วย

## ขั้นตอนการใช้งาน

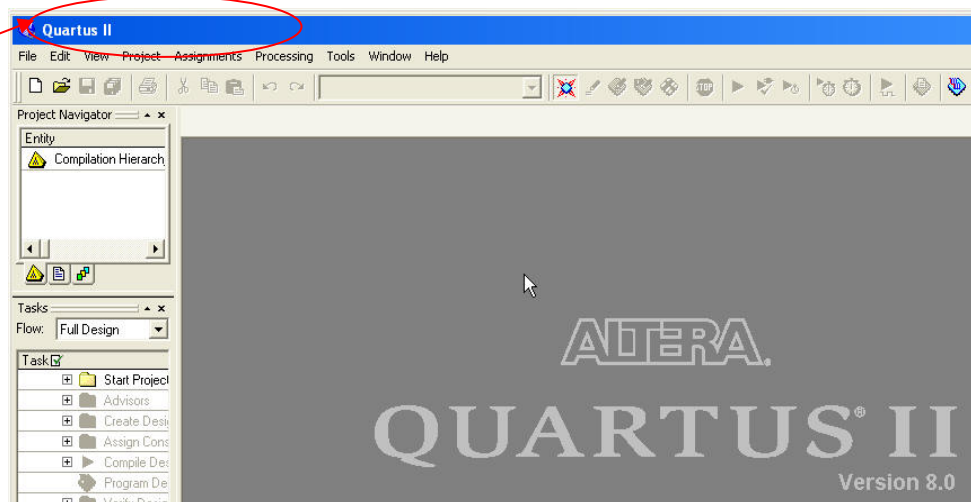
1. เรียกใช้งานโปรแกรม Quartus II โดยเลือกที่ไอคอน



รูปที่ 1

2. เมื่อ Quartus II ถูกเรียก ก็จะเปิดหน้าต่างขึ้นมาดังรูป

จุดสังเกต



รูปที่ 2

ขอให้สังเกตมุมด้านบนซ้ายของหน้าต่างหลัก จะต้องยังไม่มีข้อความใดๆ ต่อท้าย Quartus II แต่ถ้าหากไม่เป็นดังจุดสังเกตในรูปที่ 2 นี้ให้คลิกเข้าไปที่เมนู **File** ที่มุมบนด้านซ้ายของหน้าต่าง เพื่อปิดโปรเจกต์เก่าก่อนที่จะเริ่มงานใหม่ต่อไป

**File > Close Project**

**คำแนะนำ** สาเหตุที่จะต้องทำการปิดงานต้นแบบที่ค้างอยู่ก่อนหน้านี้ ทั้งนี้เนื่องจากอาจจะมีการ **โปรเจกต์ (ขอเรียกว่า Design)** เก่าของคนอื่นค้างอยู่ก็ได้ เพราะเครื่องในห้องแล็บจะมีผู้ใช้ร่วมกันหลายคนอาจจะทำให้ น.ศ. เกิดการสับสนได้ง่าย

(โดยปกติแล้วโปรแกรม Quartus II จะจำค่าต่างๆ ของงาน Design อันล่าสุดไว้เสมอ และจะเปิดขึ้นมาให้โดยอัตโนมัติ เมื่อมีการเรียกใช้งานในครั้งต่อไป ดังนั้นเมื่อทำงานเสร็จทุกครั้ง ผู้ใช้ควรสั่ง Close Project จะช่วยป้องกันปัญหานี้ได้)

### 3. การเตรียมสร้างชิ้นงานต้นแบบ ( Design ) ด้วยการสร้าง logic diagram

เมื่อเริ่มต้นสร้างชิ้นงานต้นแบบ (Design) จำเป็นจะต้องสร้างโปรเจกต์ไฟล์ (Project) ขึ้นก่อนทุกครั้ง ในขั้นตอนนี้จะต้องมีการเตรียมการที่เหมาะสมด้วยถ้าไม่เช่นนั้นแล้วจะเกิดปัญหาในขั้นตอนการคอมไพล์ที่ยุ่งยากซับซ้อนเพิ่มขึ้นมา

#### 3.1 เตรียมสร้างโปรเจกต์ (Project)

เตรียมพื้นที่บนฮาร์ดดิสก์สำหรับเก็บชิ้นงานต้นแบบ การใช้งานครั้งแรกนั้นผู้ใช้งานจำเป็นต้องสร้างโฟลเดอร์ของตนเองขึ้นมาก่อนบนฮาร์ดดิสก์ เพื่อจัดเก็บไฟล์ข้อมูลต่างๆ ที่จะเกิดขึ้นเมื่อมีการเขียน และคอมไพล์โปรแกรม

##### ข้อแนะนำในการสร้างโฟลเดอร์

ก) สร้างโฟลเดอร์เก็บข้อมูลไว้บนฮาร์ดดิสก์ไดรฟ์ C: หรือ D:

**ถ้าเก็บที่ C:** ให้เก็บ Design ไว้ที่ C:\ALTERA\81\QUARTUS\xxxxxxx

(โดยที่ xxxxxx ในที่นี้เป็นชื่อโฟลเดอร์ที่น.ศ.ตั้งขึ้นเอง ข้อควรระวังคือ เมื่อปิดเครื่อง ข้อมูลนี้จะหายเสมอ)

**ถ้าเก็บที่ D:**

ข) สร้างโฟลเดอร์ด้วยการใช้ Windows Explorer (ไม่ขออธิบาย)

ค) สร้างโฟลเดอร์ใหม่โดยใช้ Quartus II

#### 3.2 ทำการเปิด New Project ตามขั้นตอนดังนี้

a) เริ่มจากการเปิด New Project โดยเข้าไปที่เมนู

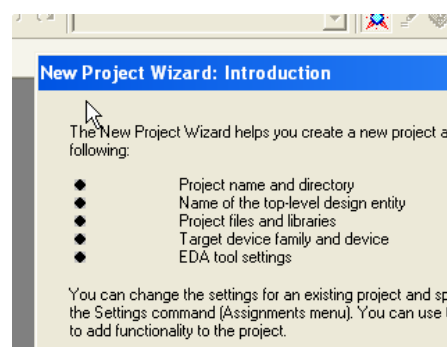
**File > New Project Wizard**

จะปรากฏหน้าต่าง ที่อธิบายและแนะนำรายละเอียดต่างๆ ที่จะดำเนินการในขั้นตอนการเปิด New Project เมื่ออ่านเข้าใจแล้วให้กด

**NEXT**

บนหน้าต่างเพื่อข้ามขั้นตอนนี้ไป

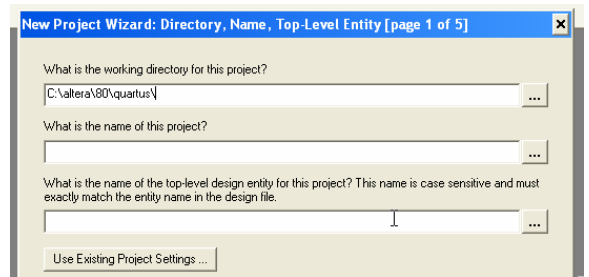
**หมายเหตุ:** คอมพิวเตอร์บางเครื่องอาจจะข้ามขั้นตอนนี้โดยอัตโนมัติ ให้น.ศ.ทำขั้นตอนถัดไป



รูปที่ 3

- b) ในขั้นตอนนี้จะปรากฏหน้าต่างขึ้นมาเพื่อให้ระบุโฟลเดอร์ที่ต้องการจะจัดเก็บโปรเจกต์ไฟล์ต้นแบบ Design

รูปที่ 4



ในตัวอย่างนี้ได้สร้างโฟลเดอร์ซ้อนลงไปในไดเรกทอรีเป็น C:\altera\80\quartus\LabDigit\521016LAB02

(การตั้งชื่อ ไม่ให้ใช้อักษรภาษาไทย และไม่เว้นวรรค ส่วนจะตั้งชื่ออย่างไรก็ได้ ขอให้ตนเองจำได้ก็พอ)

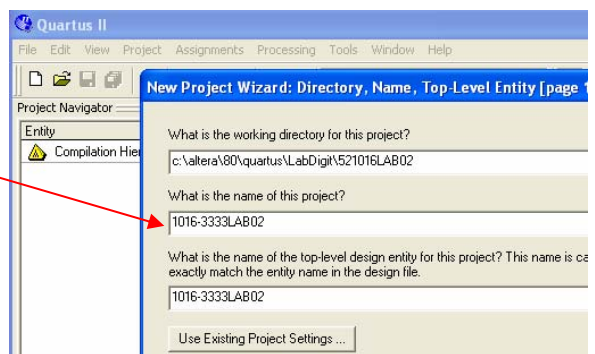
รูปที่ 5



- c) ตั้งชื่อให้กับโปรเจกต์ที่จะให้เป็นงาน Design ของนศ.

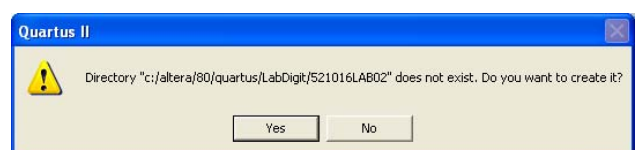
ในตัวอย่างนี้จะตั้งชื่อเป็น 1016-3333LAB02 จากนั้นก็กดปุ่ม **NEXT** ที่อยู่ด้านล่างของหน้าต่าง

รูปที่ 6



Quartus II ก็จะทำให้ยืนยันการสร้างโฟลเดอร์และโปรเจกต์ขึ้นมาใหม่ เมื่อกดปุ่ม **Yes** ก็จะเข้าสู่ขั้นตอนถัดไป

รูปที่ 7

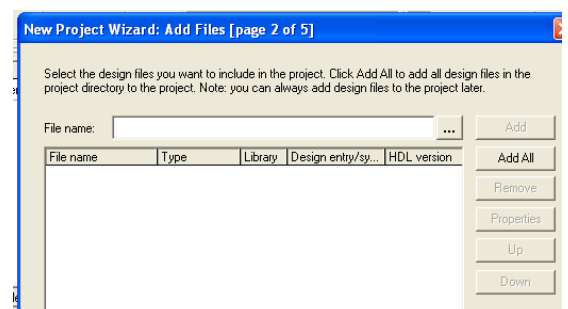


- d) ในขั้นนี้ Quartus II จะถามว่ามีไฟล์ข้อมูล (source files) อื่นๆ ที่ต้องการนำมาใช้งานในโปรเจกต์หรือไม่

ในขั้นนี้เป็นการเริ่มเขียนครั้งแรกจึงยังไม่มีไฟล์ต้นแบบเราก็จะข้ามขั้นตอนนี้ไปด้วยการกดปุ่ม

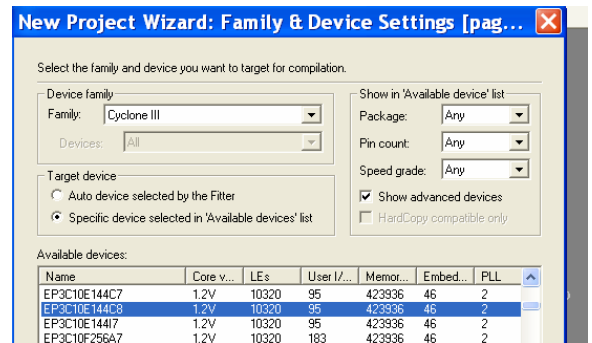
**NEXT**

รูปที่ 8



e) จากนั้น Quartus II จะถามถึงชนิดของชิพ (Chip) ที่จะเอา Design นี้ไปทำการสร้างจริง

(การสร้างชิ้นงานจริงนี้จะมีชื่อเรียกเฉพาะว่า การทำ “configuration” เนื่องจากมีวิธีการทำที่เป็นเอกลักษณ์เฉพาะตัว) เพื่อให้ได้ชิ้นงานต้นแบบขึ้นมา



รูปที่ 9

ในที่นี้ให้ใช้ชิพ Cyclone III เบอร์ EP3C10E144C8 ซึ่งมีใช้งานบนบอร์ดทดลองในห้องแล็บ

**คำเตือน** ให้ใช้ชิพ Cyclone III รุ่น EP3C10E144C8

f) โปรแกรม Quartus II จะถามเกี่ยวกับรายละเอียดของเครื่องมือหลักที่จะใช้ทำการออกแบบดังนี้

Design Entry/Synthesis:

Tool name: Design Compiler

Format: VHDL

Simulation:

Tool name: Custom

Format: VHDL

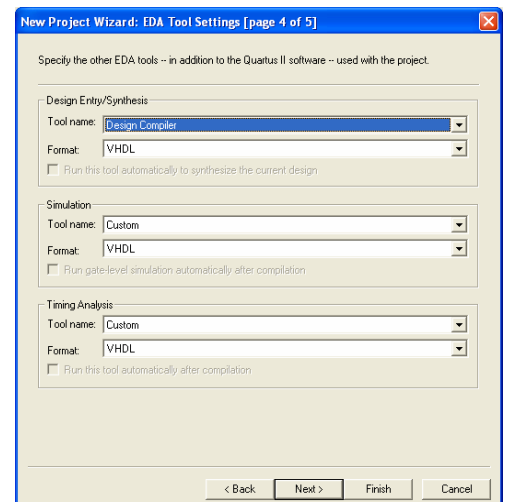
Timing Analysis:

Tool name: Custom

Format: VHDL

ในขั้นนี้ขอให้ตั้งค่าตามที่กำหนดข้างต้น

จากนั้นก็กดปุ่ม **NEXT**

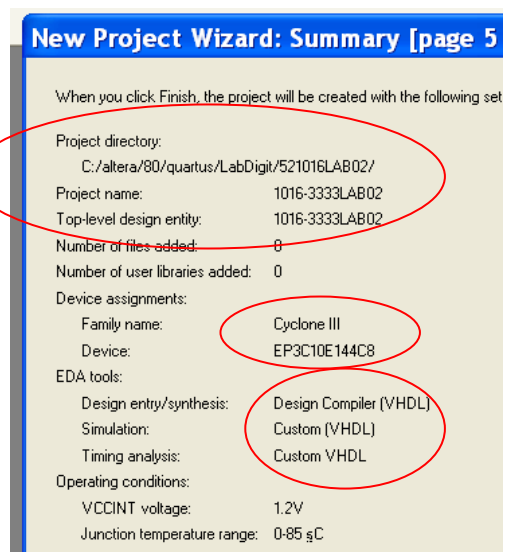


รูปที่ 10

**หน้าต่างสรุป** คุณลักษณะของโปรเจกจะปรากฏขึ้นโดยเป็นค่าตั้งต้น (Initial Setup /Project Property) ของชิ้นงานเรา

เมื่อตรวจสอบดูแล้วหากพบว่ายังไม่ตรงตามต้องการหรือมีสิ่งที่ต้องการแก้ไขก็สามารถทำการย้อนกลับไปแก้ไขได้ แต่ถ้าไม่มีการแก้ไขเปลี่ยนแปลงรายการใดๆ ให้กดปุ่ม

**FINISH** เมื่อตรวจสอบรายละเอียดถูกต้องครบแล้ว



รูปที่ 11

## 4. ตรวจสอบความพร้อมก่อนการเขียนวงจร

ลอจิกไดอะแกรม ( logic diagram )

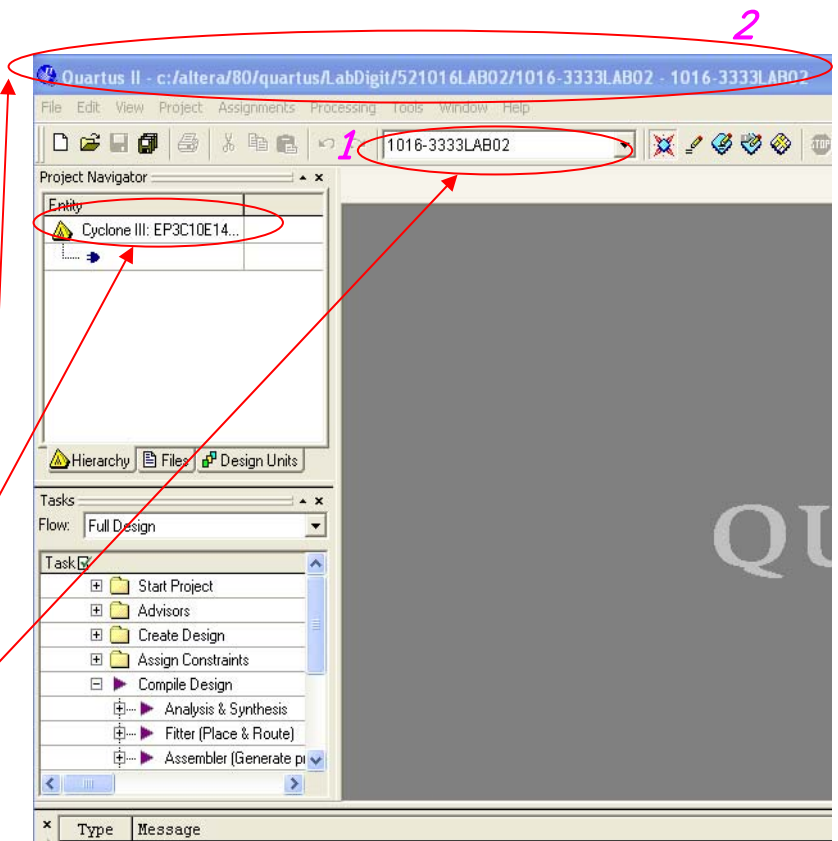
เมื่อสร้างโปรเจกใหม่(ขั้นตอนที่ 3) เสร็จ  
หน้าตาหลักของ Quartus II จะแสดงไฟล์  
เดอร์ของงาน Design ขึ้นที่ title bar  
ถ้าไม่ขึ้นตามนี้ แสดงว่าขั้นตอนที่ 3 ไม่ถูกต้อง

ชื่อโปรเจกและสถานที่เก็บงาน design  
(ขั้นตอนที่ 3c )

ชนิดของชิพ FPGA ที่ใช้ในการออกแบบ  
(ขั้นตอนที่ 3e )

ชื่อของงาน designที่กำลังถูกดำเนินการ  
(ขั้นตอนที่ 3c )

รูปที่ 12

**คำเตือน**

ขอให้ตรวจสอบด้วยว่า ชื่อของโปรเจกที่ปรากฏขึ้นในช่องแท็บหมายเลข 1 และ 2 มีชื่อ  
ตรงกันทุกครั้งเพื่อให้มั่นใจว่าในตอนคอมไพล์จะไม่เกิดปัญหาภายหลัง

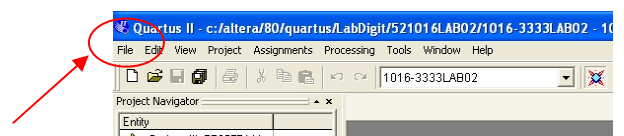
**ตอนที่ 1****การสร้างชิ้นงาน Project ด้วยการเขียนวงจรลอจิก**

## 5. เตรียมการสำหรับเขียนวงจร ลอจิกไดอะแกรม ( logic diagram )

ในขั้นตอนนี้จะเป็นการใช้เครื่องมือสำหรับสร้างชิ้นงาน (design Tools) ที่เรียกว่า **Block Diagram /Schematic File** มีลักษณะเป็นโปรแกรมแบบ Graphic Editor ของ Quartus II ช่วยให้มีความสะดวกมากในการสร้างโปรเจกแบบเป็น แผนภาพ block diagram หรือ รูปวงจร logic diagram โดยขั้นตอนการดำเนินงานมีดังนี้

## 5.1) การใช้ Top-Level Design Tools ของ Quartus II เพื่อสร้างวงจร logic diagram

- a) สร้างไฟล์ของงาน logic diagram เริ่มจากเมนูที่  
หน้าตาหลัก ให้เลือกเมนู

**FILE > NEW**

รูปที่ 13

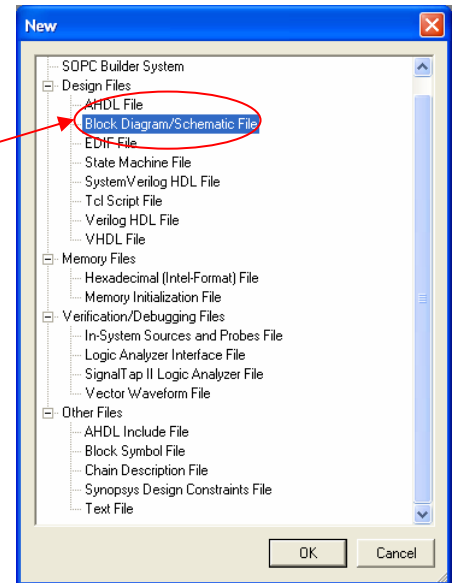
จะปรากฏหน้าต่างย่อยขึ้นดังรูปที่ 14

ให้เลือกหัวข้อ **Design files**

เป็นแบบ **Block Diagram/Schematic File**

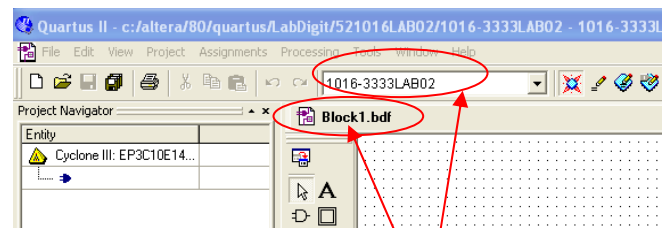
จากนั้นก็กดปุ่ม **OK**

หน้าต่างของ **graphic editor** จะแสดงขึ้นมาดังรูปที่ 15



รูปที่ 14

- b) บนหน้าต่างของ **graphic editor** จะสังเกตเห็นว่าไฟล์ที่ได้นั้นถูกกำหนดให้โดยอัตโนมัติเป็นชื่อ **block1.bdf** ซึ่งไม่ตรงกับชื่อที่กำหนดไว้ในขั้นตอนที่ 3 C) ดังนั้น เราจะต้องทำการเปลี่ยนชื่อไฟล์จาก **block1.bdf** ให้เป็นชื่อเดียวกันกับที่ได้กำหนดไว้ในขั้นตอน 3C ) โดยเลือกเมนู **FILE > Save As...**

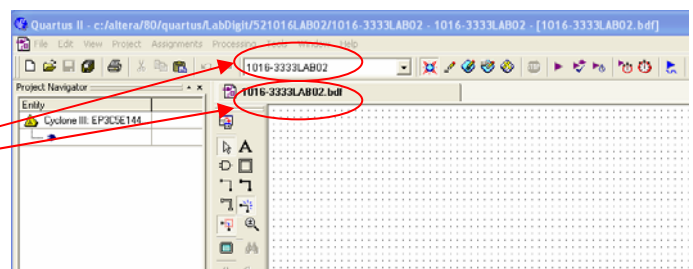


ชื่อต่างกัน จะทำให้คอมไพล์ไฟล์ไม่ได้

รูปที่ 15

จากนั้นก็ทำการเปลี่ยนชื่อไฟล์ให้ตรงกับชื่อของโปรเจกต์ซึ่งเราได้ตั้งชื่อไว้เป็น **521016LAB02.bdf** ก็ถือว่า Quartus II ของเราพร้อมใช้งานแล้ว

ชื่อตรงกัน จะทำให้สามารถคอมไพล์ไฟล์ได้



รูปที่ 16

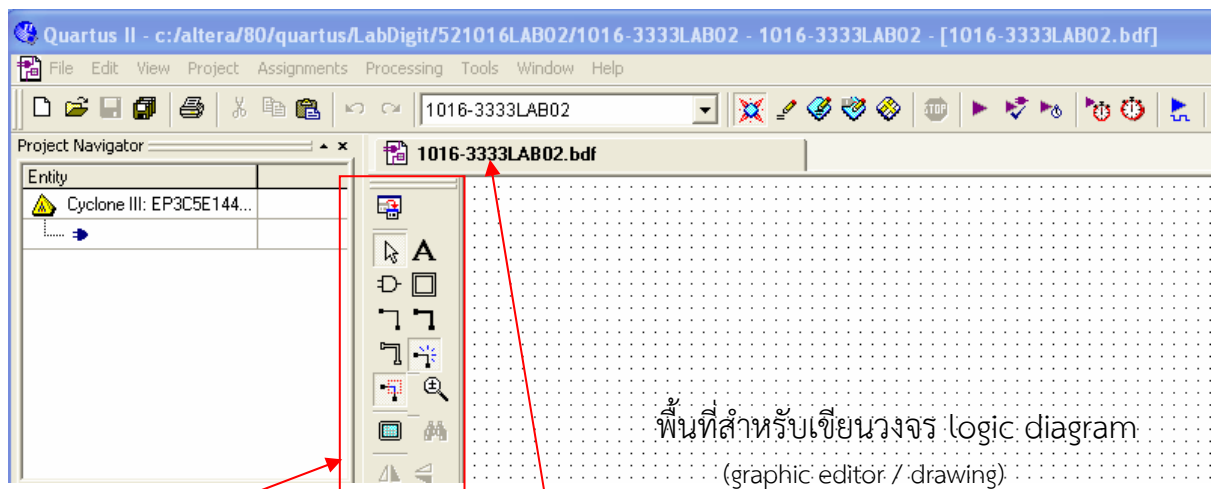


## c) เครื่องมือพื้นฐานสำหรับใช้งานบนหน้าต่าง graphic editor

การเริ่มต้นใช้ graphic editor เพื่อเขียนวงจร logic diagram เราจะใช้เครื่องมือพื้นฐานเพียงไม่กี่ชนิดก็สามารถที่จะทำงานง่ายๆ ได้

จากหน้าต่าง graphic editor จะแบ่งส่วนย่อยๆ ออกได้เป็น 3 ส่วน คือ

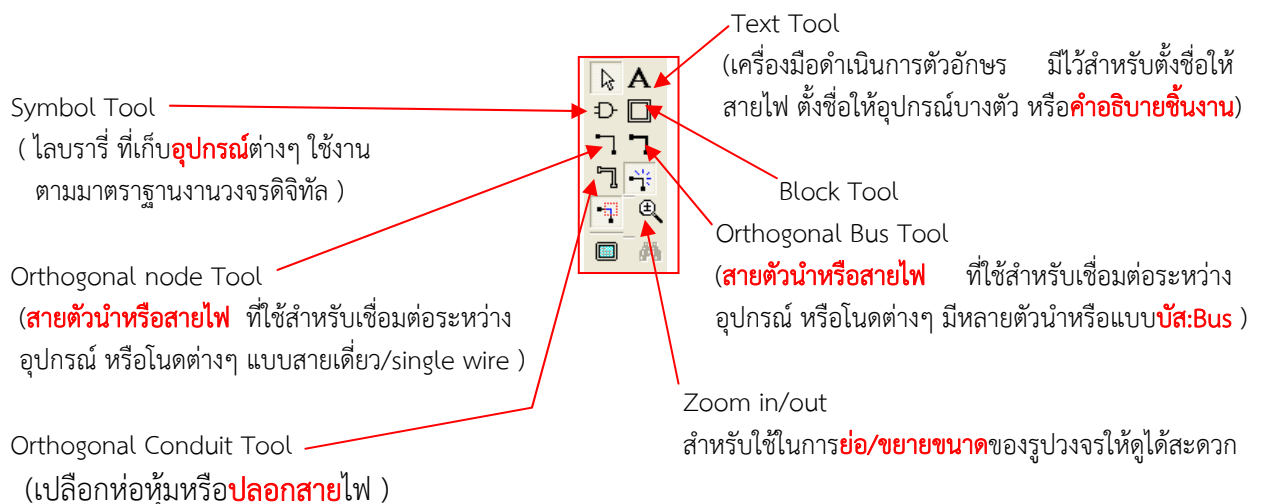
- พื้นที่สำหรับเขียนวงจร logic diagram เสมือนเป็นแผ่นกระดาษสำหรับใช้เขียนแบบชิ้นงาน เราสามารถเลือกขนาด size ได้ เช่น A4 A3 เป็นต้น
- ไอคอนสำหรับเลือกเครื่องมือในการเขียนแบบ หรือวงจร logic diagram เช่น อุปกรณ์ เกท ไอซี สายไฟ(สายตัวนำ) อักษรตัวอักษร เป็นต้น
- ชื่อของไฟล์ แผ่นชิ้นงานต้นแบบของเรา



เครื่องมือสำหรับการเขียนวงจร logic diagram หรือ schematic diagram

ชื่อไฟล์ที่เก็บวงจร logic diagram

รูปที่ 17



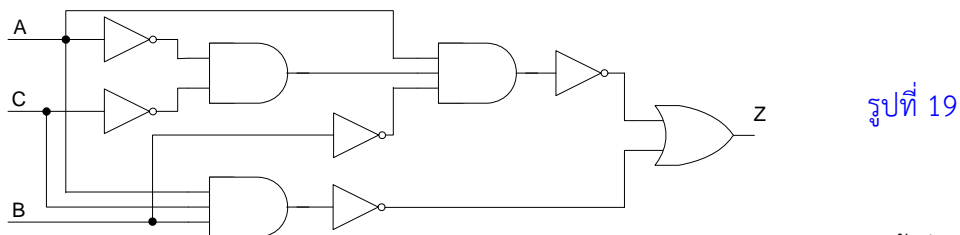
รูปที่ 18

## 5.2) สร้างวงจร logic Diagram

ในขั้นตอนนี้จะเป็นการสร้าง logic diagram โดยใช้ Graphic editor Tool ของ Quartus II

สมมุติเราได้วงจรลอจิกมาจากในหนังสือเรียน แต่เกิดข้อสงสัยว่าวงจรที่ได้มานั้นจะให้เอาต์พุต Z อย่างไร เพื่อตอบคำถามนี้เราอาจจะต้องใช้ความรู้ทางด้านดิจิทัลมาทำการสังเคราะห์ (synthesis) เพื่อหา logic function ของระบบนี้หรือในอีกทางหนึ่งเราอาจทำการจำลองการทำงานของวงจรนี้ขึ้นมา แล้วลองป้อนอินพุต (A,B,C) ให้กับวงจรนี้ในทุกๆ กรณีที่เป็นไปได้ (ตามค่าในตารางความจริง: [truth table](#))

5.2.1 เราจะเขียนวงจร และทำการจำลอง (กำหนด A,B,C) ดูการทำงาน (ผลลัพธ์ Z) ของวงจрдังรูปที่ 19

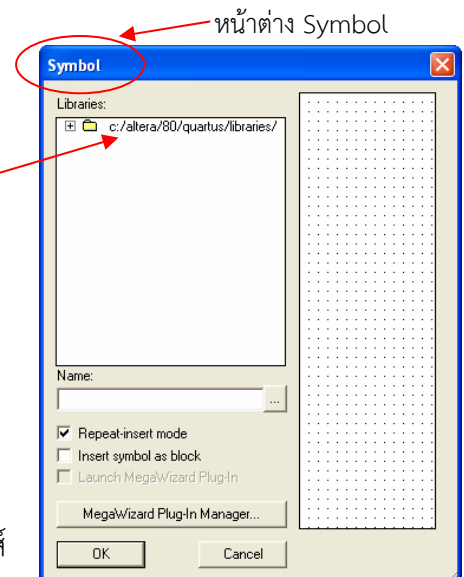


รูปที่ 19

5.2.2 เริ่มต้นด้วยการใช้เมาส์กดดับเบิลคลิกที่ symbol tool  จะปรากฏหน้าต่าง Symbol ขึ้นมาให้เลือกไลบรารีดังรูป

ไลบรารี : สถานที่สำหรับเก็บอุปกรณ์ทางดิจิทัลทั้งหมดที่มีติดตั้งมาให้พร้อมแล้วในโปรแกรม

รูปที่ 20



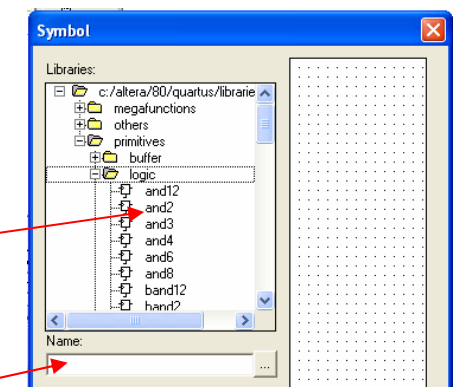
ถ้าอยากรู้ว่าในไลบรารีนั้นมีอุปกรณ์อะไรบ้าง ก็ทำได้โดยใช้เมาส์กดดับเบิลคลิกที่ปุ่มไอคอนเครื่องหมายบวก ก็จะพบว่ามามีมากมาย แต่อย่างไรก็ตามในขั้นนี้เราจะสนใจเฉพาะอุปกรณ์พื้นฐานของวงจรลอจิกตามที่เคยเรียนมา เช่น AND OR NAND NOR XOR Flip-Flop ก่อน

5.2.3 ในหน้าต่าง Symbol ให้คลิกเมาส์เพื่อทำการเลือกอุปกรณ์พื้นฐานได้จากไดเรกตอรีที่ชื่อ

**primitives / Logic**

และเราเลือกลอจิกเกตต่างๆ ตามต้องการได้จากรายชื่อของอุปกรณ์ที่มีให้ ดังรูป

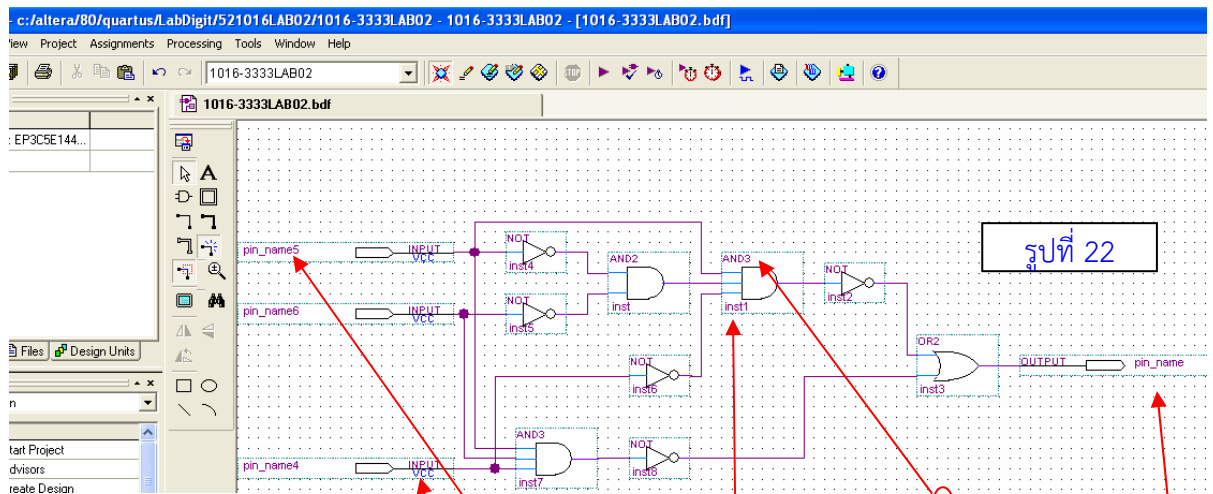
หรือเมื่อเราใช้โปรแกรมได้คล่อง จนจำชื่ออุปกรณ์ได้แล้ว ก็สามารถพิมพ์ชื่อของอุปกรณ์ลงในช่อง Name: เลยก็ได้



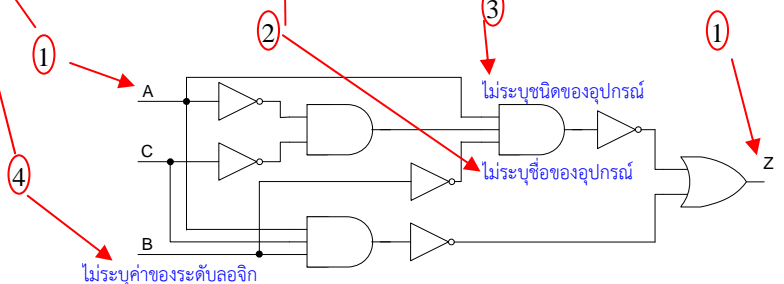
รูปที่ 21



5.2.4 การเขียนวงจร ก็จะทำโดยการนำลอจิกเกตที่เลือกมาได้ไปวางบนหน้าต่างของ graphic editor ได้ทันที และหากต้องการยกเลิก ก็สามารถกดปุ่มแป้นพิมพ์ **ESC** เพื่อทำการยกเลิกได้ เมื่อเขียนไปจนเสร็จจะได้ logic diagram ดังรูป



อย่างไรก็ตามเราจะพบว่ามีข้อแตกต่างระหว่างงานเขียนแบบ (ข้อ 5.2.1) กับงานเขียนวงจร logic diagram ในขั้นตอนนี้เพื่อจะจำลองการทำงานของเรายู่ 4 จุดคือ



- ① รูปวงจร logic diagram บน Quartus II จะต้องมียูปรณ์ **input port** เพื่อรองรับการป้อนสัญญาณเข้าสู่วงจรและมี **output port** รับส่งผลลัพธ์ที่ได้ออกมาจากวงจร ในที่นี้เราสามารถใส่เมาส์ชี้ที่ **pin\_name** แล้วกด **ดับเบิลคลิก**เพื่อทำการเปลี่ยนชื่อของแต่ละพอร์ตได้
- ② ชื่อของอุปกรณ์ที่มีอยู่ในวงจร logic diagram ซึ่งโปรแกรมจะตั้งชื่อเรียกให้โดยอัตโนมัติ
- ③ ชนิดหรือประเภทของอุปกรณ์ ถ้าเป็นอุปกรณ์มาตรฐาน ก็จะแสดงให้เห็นดังในรูปเช่น AND3 เป็นต้น
- ④ คำว่า **VCC (สีน้ำเงิน)** ที่ปรากฏอยู่ แจ้งให้ทราบว่า ขาอินพุตนี้ถูกต่อไว้ด้วยลอจิก “1” (ระดับเท่ากับไฟเลี้ยงวงจร) ไว้เสมอ (by default)

เปรียบเทียบจำนวน ชนิดของลอจิกเกต และอุปกรณ์ที่ใช้ในวงจร มีดังนี้

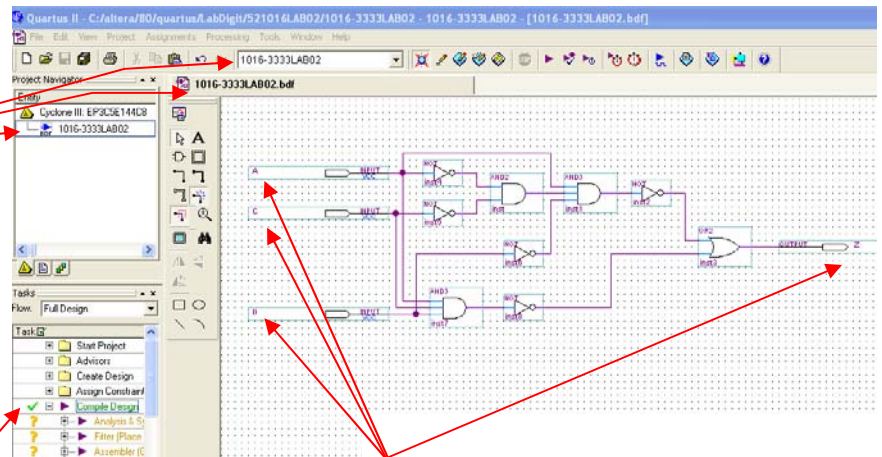
อุปกรณ์ logic ชนิด/ประเภท	ชนิด (ในงานเขียนแบบ)	ชนิด (ใน Quartus II)	จำนวนที่ใช้ในวงจร
เกตแบบ AND , 2 อินพุต	AND เกต 2 อินพุต	<b>AND2</b>	จำนวน 1 เกต
เกตแบบ AND , 3 อินพุต	AND เกต 3 อินพุต	<b>AND3</b>	จำนวน 2 เกต
เกตแบบ INVERTER	NOT เกต	<b>NOT</b>	จำนวน 5 เกต
เกตแบบ OR , 2 อินพุต	OR เกต 2 อินพุต	<b>OR2</b>	จำนวน 1 เกต
ขา (PIN) หรือพอร์ตทางเข้า	(ไม่มี)	<b>INPUT</b>	จำนวน 3 ตัว
ขา (PIN) หรือพอร์ตทางออก	(ไม่มี)	<b>OUTPUT</b>	จำนวน 1 ตัว

- หมายเหตุ**
1. การ**เปลี่ยนชื่อ** input pin และ output pin ให้มีชื่อเป็น A, B, C และ Z สามารถทำได้โดยการเลื่อนเมาส์ไปวางบนข้อความที่เป็นชื่อของ input pin เช่น **pin\_name5** แล้วกดดับเบิลคลิกก็จะปรากฏกล่องข้อความขึ้นมาให้แก้ไขชื่อ
  2. เราสามารถที่จะ**ตั้งชื่อให้กับสายตัวนำ** (wiring) ที่เชื่อมต่อระหว่างตัวอุปกรณ์ได้ด้วยการใช้เมาส์ชี้แล้วกดดับเบิลคลิก
  3. เมื่อเขียนวงจรเสร็จแล้วให้กดปุ่มบันทึกไฟล์ก่อนที่จะทำขั้นตอนต่อไป

## 6. เตรียมการสำหรับคอมไพล์วงจร ลอจิกไดอะแกรม

ก่อนคอมไพล์ขอให้ น.ศ. ตรวจสอบ  
ความถูกต้องของโปรเจก ก่อนดังนี้

- ① ชื่อจะต้องตรงกันทั้งในส่วนของ
  - ชื่อของ Entity
  - ชื่อของไฟล์ที่เก็บ logic diagram
  - ชื่อของโปรเจก
 จึงจะทำให้สามารถคอมไพล์ได้
- ② รายละเอียดขั้นตอนของการคอมไพล์จะ  
แสดงให้ดูก่อนที่จะเริ่มดำเนินการจริง  
ในหน้าต่าง Tasks Flow



รูปที่ 23

- ③ ชื่อของพอร์ทถูกแก้ไขให้ถูกต้องตามที่กำหนด  
ไว้เป็น A, B, C และ Z (ตามที่โจทย์กำหนด)

## 7. คอมไพล์ logic diagram

การคอมไพล์โปรเจกทำโดยกดปุ่ม hot Icon



ที่อยู่บนเมนูที่หน้าต่างหลักเพื่อทำการคอมไพล์โปรแกรม

หากไม่มีข้อผิดพลาดใดๆ เกิดขึ้น การคอมไพล์ก็จะสำเร็จ และหน้าต่างดังรูปที่ 24 จะปรากฏขึ้น

ผลของการคอมไพล์ของโปรแกรม จะสามารถพิจารณาได้เป็น 4 หน้าต่างหลักคือ

สถานะของการคอมไพล์  
“การคอมไพล์สำเร็จ”

รายละเอียดของการคอมไพล์  
ไฟล์ จะมีประโยชน์มากใน  
กรณีที่เกิดการคอมไพล์ไม่  
ผ่าน โดยจะบอกว่าขั้นตอน  
ใดบ้างที่เกิดข้อผิดพลาด

② รายละเอียดทั้งหมดของการออกแบบ

① ข้อมูลสำคัญของโปรเจกที่เราควรรับรู้

③ หน้าต่างสำหรับแสดงข้อความ ระบุข้อผิดพลาด(กรณีคอมไพล์ไม่สำเร็จ) หรือแสดงค่าเตือนระดับต่างๆ (แยกโดยสีของข้อความ)เตือนความไม่สมบูรณ์ของโปรเจก (ผู้ออกแบบอาจจะขาดความรอบคอบในการออกแบบ) ซึ่งอาจจะส่งผลกระทบได้ในงานจริง

รูปที่ 24

ทำความเข้าใจเบื้องต้น จากผลการคอมไพล์

ในหน้าต่างที่ 1 จะเป็นการแสดงสรุปรายการที่เกี่ยวข้องกับจำนวนของเกทที่มีอยู่และที่ถูกใช้งานไปเมื่อมีการนำไปสร้างชิ้นงานจริง (ในที่นี้จะแสดงสาระสำคัญด้าน **ทรัพยากรที่ใช้ไป**) หากต้องการดูรายละเอียดทางเทคนิค รวมถึงเทคโนโลยีที่นำมาใช้ หรือ **ข้อจำกัด/เงื่อนไข** ของการออกแบบ ที่โปรแกรมได้ดำเนินการให้เราในขั้นตอนของการคอมไพล์ (ค่อนข้างจะซับซ้อน) ทั้งหมด (ซึ่งมีเยอะมาก) ผู้ใช้ก็สามารถเปิดดูได้ใน **หน้าต่างที่ 2**

ในหน้าต่างที่ 3 จะแสดงรายการต่างๆว่า **การคอมไพล์นั้นสำเร็จจนครบทุกขั้นตอนหรือไม่** ถ้ามีปัญหาในขั้นตอนใด (กรณีเกิดการคอมไพล์ไม่ได้) การแสดงผลการคอมไพล์จะเป็นรูปของเครื่องหมายกากบาท ✗ แทนที่จะเป็นเครื่องหมายถูก ✓

ในหน้าต่างที่ 4 ใช้สำหรับแสดงข้อความต่างๆ ที่เป็นการ **แจ้ง error (ถ้ามี)** หากผู้ออกแบบไม่ปฏิบัติตามกฎของการออกแบบหรือ **คำเตือน** (warning message) หากมีการออกแบบโดยไม่ระบุเงื่อนไขสถานะของลอจิกในสถานะเริ่มต้น หรือรายละเอียดอื่นที่ควรมีแต่ถ้าหากไม่มี (อาจจะไม่ได้ตั้งใจ หรือไม่รู้) โปรแกรมจะกำหนดใส่ให้เอง (by default) ซึ่งจะไม่ส่งผลกระทบต่อการทำงานตามฟังก์ชันที่ออกแบบไว้แต่อย่างใด

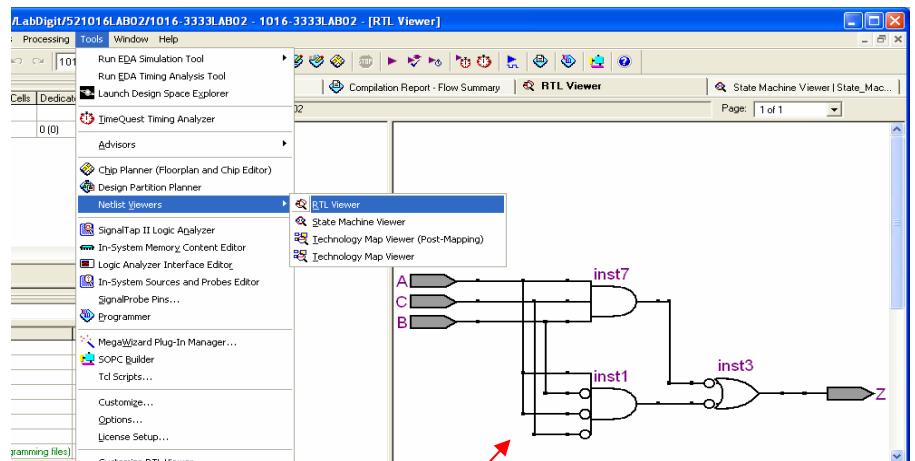
แนวทางพิจารณาผลงานโปรเจกต์ที่ออกแบบ

- 1 เนื่องจากการคอมไพล์เป็นการประมวลผลเพื่อทำ optimization ให้กับวงจรซึ่งมีสอง แนวทางคือ ต้องการได้ **เวลาเร็วที่สุด** หรือได้ **ขนาดวงจรเล็กที่สุด** โดยที่ยังให้ความสัมพันธ์กันระหว่าง input กับ output เหมือนเดิม ดังนั้นการตรวจสอบผลลัพธ์ของการคอมไพล์ (ดูประสิทธิภาพของการคอมไพล์) จึงเป็นสิ่งจำเป็น อย่างไรก็ตาม การแสดงผลนี้จะแสดงในระดับฟังก์ชัน **RTL (Register Transfer Level)** ไม่ได้แสดงในระดับฮาร์ดแวร์จริง
- 2 เราสามารถเข้าไปดูผลการคอมไพล์ในระดับ RTL ได้จากเมนู

Tools → Netlist Viewers

→ RTL Viewer

ตามลำดับก็จะได้ผลลัพธ์ดังรูป



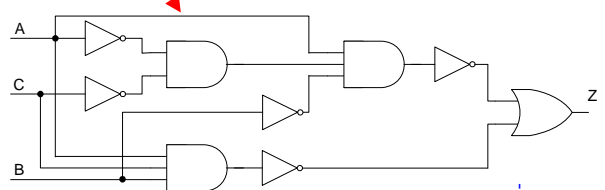
จากรูปแสดงให้เห็นผลลัพธ์ที่ได้จากการทำ simplified หรือลดรูปวงจร

แนะนำให้ น.ศ. ลองตรวจสอบดูกับการการคำนวณ ด้วยการใช้พีชคณิตบูลีน น่าจะช่วยให้เข้าใจการออกแบบวงจรดิจิทัลดีขึ้น

รูปที่ 25

สองวงจรนี้ให้ผลลัพธ์  $F(A,B,C) = Z$  เหมือนกัน แต่วงจรด้านบนจะประหยัดเกทได้มากกว่า

ข้อสังเกต จากรูปที่ 25 และ 26 จะทำให้เราพบความจริงที่ว่า ไม่ว่าเราจะเขียนวงจรให้มีจำนวนเกทมากเท่าใดแต่พอเมื่อนำมาสร้างบนโปรแกรม Quartus II แล้ว มันจะคำนวณ **cost** ของวงจรและปรับวงจรให้ใหม่ในตอนสร้างจริงเพื่อให้ได้ประสิทธิภาพที่สูงที่สุดเสมอ



รูปที่ 26

## 8. จำลองการทำงานของวงจร

การจำลองการทำงานในที่นี้จะเป็นการสมมุติสถานะลอจิกอินพุต (ค่าตามตารางความจริง) ให้กับวงจรแล้วดูผลลัพธ์ของเอาต์พุตที่เกิดขึ้นได้ในกรณีต่างๆ เพื่อให้เราสามารถตรวจสอบโปรเจกต์ว่าทำงานได้ตรงตามความต้องการหรือไม่ ส่วนเครื่องมือสำหรับการจำลองการทำงาน เราจะใช้ **แผนภาพทางเวลา (timing diagram)** มาแสดงความสัมพันธ์กันของ input และ output ที่เวลาต่างๆ เครื่องมือชนิดนี้มีข้อดีตรงที่ยอมให้เราซูม ย่อ/ขยาย การเปลี่ยนแปลงของสัญญาณได้ละเอียดมากถึง ระดับ  $10^{-12}$  วินาที

### การใช้ Simulation Tool ของ Quartus II

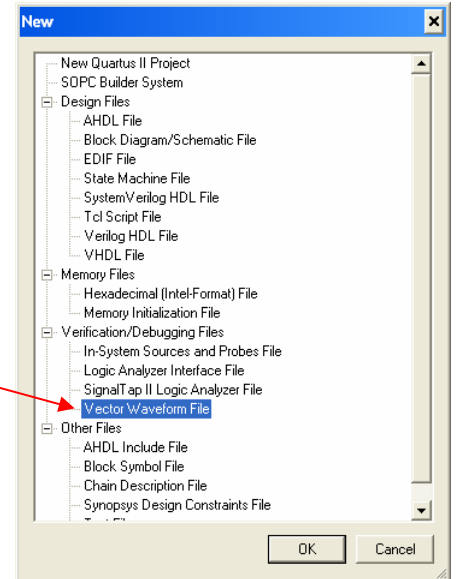
8.1) **เตรียมการจำลองการทำงาน** จำเป็นจะต้องสร้างแผนภาพทางเวลาขึ้นมาก่อน ในที่นี้จะเป็นการสร้างไฟล์สำหรับเก็บข้อมูลแผนภาพทางเวลาขึ้นมาหนึ่งไฟล์ ทำได้โดยไปที่หน้าต่างหลักและเลือกเมนู

FILE > NEW

เลือกสร้างไฟล์แบบ **Vector Waveform file**

กดปุ่ม **OK**

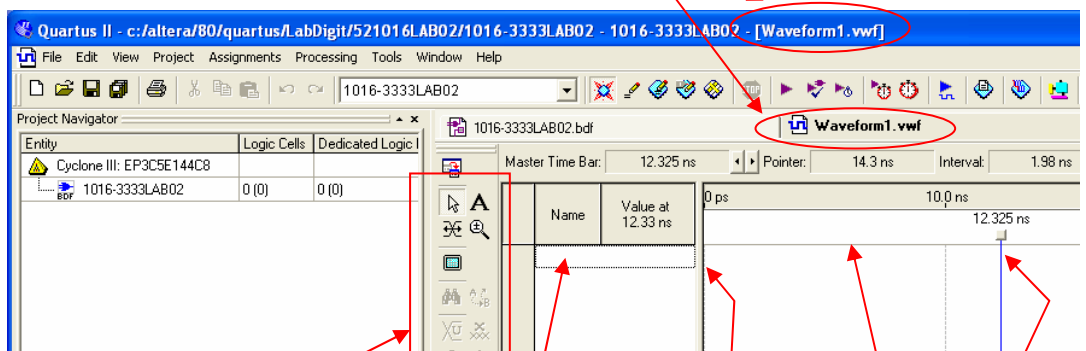
โปรแกรมก็จะสร้างไฟล์ที่มีนามสกุล **\*\*\*.vwf** ขึ้นมาให้



ไฟล์ที่บนหน้าต่างใหม่ดังรูปที่ 28 โปรแกรมตั้งชื่อไฟล์ให้เราเป็น **waveform.vwf** โดยอัตโนมัติ เราค่อยเปลี่ยนชื่อไฟล์ในภายหลัง เราจะต้องสร้างกราฟการเปลี่ยนแปลงค่าของลอจิกที่ A,B,C ในแบบ timing diagram เพื่อดูผลของ Z

รูปที่ 27

ไฟล์นามสกุล vwf และหน้าต่างสำหรับเขียนกราฟ



แถบเครื่องมือสำหรับ  
สร้างรูปสัญญาณลอจิก  
ให้กับขาสัญญาณอินพุต

พื้นที่สำหรับแสดง  
ชื่อของสัญญาณ

เส้นแนวตั้งบอก  
ระดับลอจิกมีค่า 0, 1

เส้นแนวนอนคือ  
แกนเวลาของกราฟ

เส้น เคอร์เซอร์ สามารถใช้เมาส์จับลากเลื่อนซ้าย-  
ขวาบนแกนเวลาเพื่อให้ดูเวลาได้ละเอียดยิ่งขึ้น

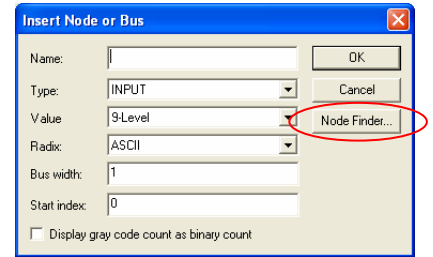
รูปที่ 28



8.2) เลือกสัญญาณอินพุต/เอาต์พุตที่ต้องการจะให้แสดงบนกราฟ Timing diagram ซึ่งในที่นี้เราจะเลือกเอาสัญญาณ A, B, C และ Z ของวงจรขึ้นมาแสดงรูปกราฟ โดยเลือกที่เมนู

Edit > Insert

จะปรากฏหน้าต่าง Insert node or bus ขึ้นมาดังรูปที่ 29 จากนั้น ให้กดปุ่ม Node Finder เพื่อค้นหา node หรือขาสัญญาณ (PIN หรือ PORT) ที่มีอยู่ในวงจร



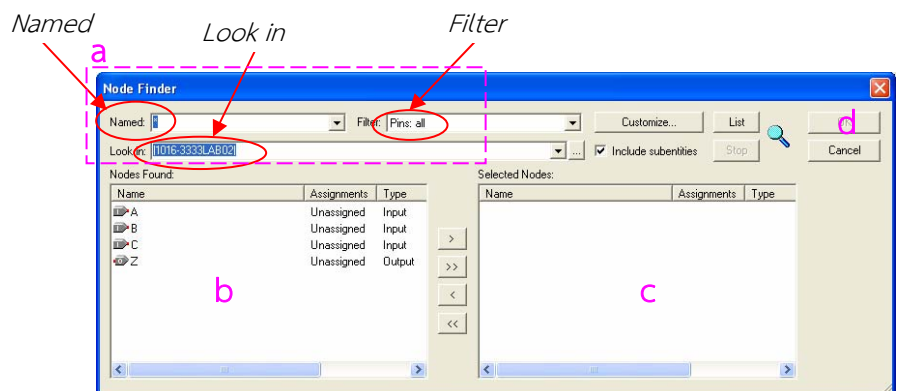
รูปที่ 29

a) ขั้นแรกเราจำเป็นต้องกำหนดเงื่อนไขของการค้นหา ในหน้าต่าง Node Finder โดย

Named : \* (ให้แสดงทุกชื่อของสัญญาณที่มีในวงจร)

Filter : all (ให้แสดงหรือเลือกดูทุกชนิดของสัญญาณ ทั้ง port, pin, wire, node, ...)

Look in : (ให้นำสัญญาณมาจากวงจรที่อยู่ในโปรเจกของเรา ซึ่งในที่นี้จะเป็นชื่อ 1016-3333LAB02)



รูปที่ 30

b) เมื่อกดปุ่ม List สัญญาณทั้งหมดก็จะปรากฏขึ้นที่ในบล็อกด้านซ้ายมือ (ในหน้าต่าง Nodes Found)

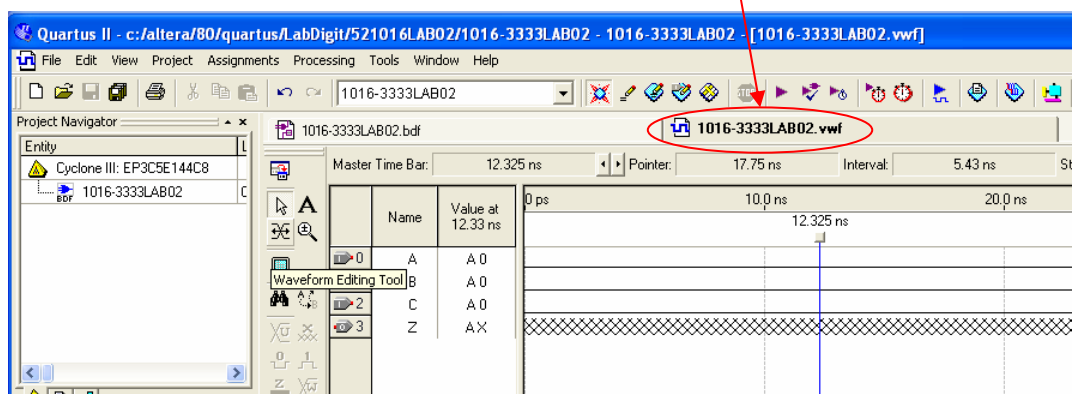
c) เมื่อกดปุ่ม >> เพื่อเลือกสัญญาณทั้งหมด และจะปรากฏขึ้นที่บล็อกด้านขวา (Selected Nodes)

d) กดปุ่ม OK ก็จะปรากฏรูปกราฟของ timing diagram ของสัญญาณทั้งหมด

e) ทำการ save ไฟล์รูปไว้ก่อนโดยการเลือกเมนู

File > Save As...

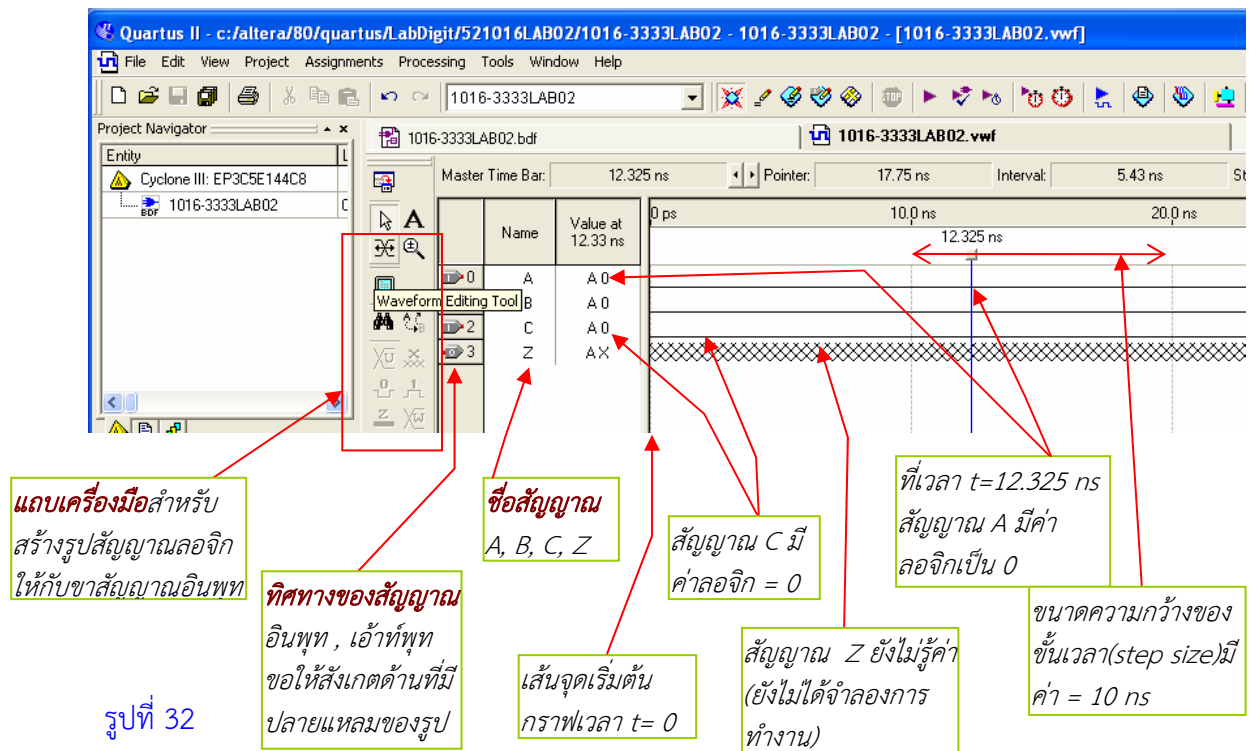
จะทำให้ไฟล์รูปของ timing diagram มีชื่อไฟล์เป็น 1016-3333LAB02.vwf โดยอัตโนมัติ



รูปที่ 31

## 8.3) เตรียมการจำลองการทำงาน

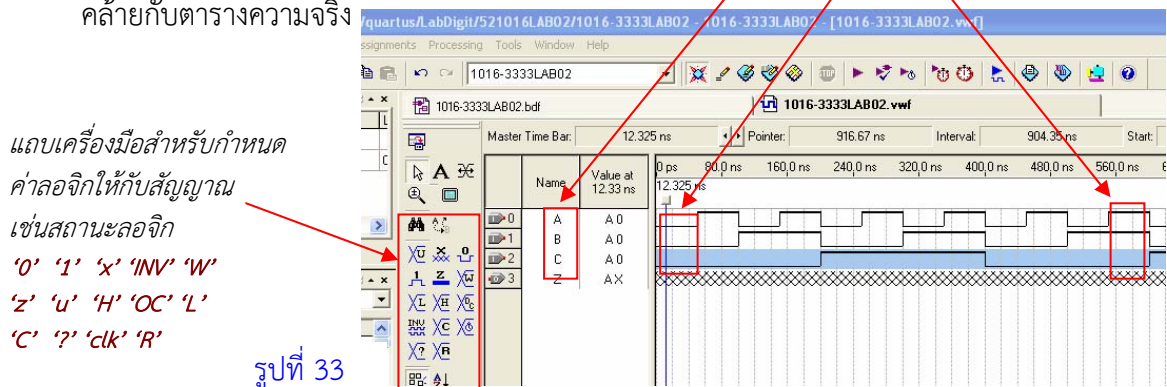
ก) ทำความเข้าใจ หน้าต่างแสดงกราฟแผนภาพทางเวลา (Timing diagram) ของ Quartus II ดังรูปที่ 32



- ข้อแนะนำ :
1. เลื่อนกราฟดู timing diagram ที่เวลาต่างๆได้โดยใช้เมาส์คลิกที่แท็บด้านล่างสุดของหน้าต่าง
  2. ย่อ/ขยายกราฟโดย เลือกเมนู View>>Zoom In... หรือ View>>Zoom Out...
  3. กำหนดค่าความละเอียดของแกนเวลาบนกราฟได้โดยกำหนดค่าของ grid size โดยไปที่จากเมนู Edit >> Grid size ... (ซึ่งค่า default ถ้าเราไม่ระบุ โปรแกรมจะกำหนดเป็น  $10 \text{ ns}$  โดยอัตโนมัติ)
  4. กำหนดค่าช่วงเวลาของการจำลองการทำงานได้ (โดยเริ่มเวลาที่  $t = 0$  ไปสิ้นสุดที่เวลา  $t = \text{End time}$ ) เราสามารถเปลี่ยนค่าของ End time ได้จากเมนู Edit>>End time ... (ซึ่งโดย default ถ้าเราไม่ระบุ โปรแกรมจะกำหนดเป็น  $1.0 \mu\text{s}$  โดยอัตโนมัติ)

## ข) กำหนดค่าลอจิกให้กับขาสัญญาณอินพุต

กำหนดค่าลอจิกให้กับ A B และ C (ไม่ต้องกำหนดให้ Z เพราะเป็นเอาท์พุต) ดังรูปที่ 33 โดยในที่นี้เราจะมองให้มันเป็นระบบเลขไบนารีขนาด 3 บิตโดยค่าตั้งแต่ CBA = 000 ถึง 111 ซึ่งมีทั้งหมด 8 สถานะ คล้ายกับตารางความจริง

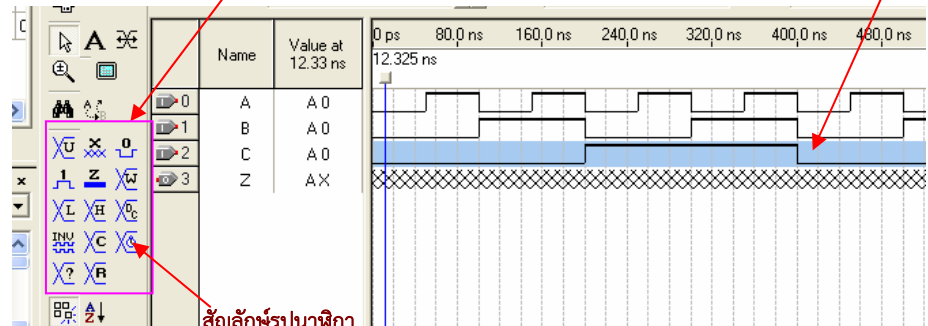




## วิธีการกำหนดค่าลอจิกให้กับสัญญาณ

1. **สำหรับสัญญาณที่ไม่เป็นรายคาบ** หากเราต้องการกำหนดค่าในช่วงใด ก็ใช้เมาส์ชี้ตำแหน่งนั้นแล้วปุ่มคลิกค้างไว้แล้วลากไปจนได้ระยะที่ต้องการ (ใช้เมาส์ทำ drag and drop) ก็จะเกิดแถบสีน้ำเงินขึ้นมา จากนั้นก็ใช้เมาส์เลือก **ค่าลอจิก** ที่ต้องการ

รูปที่ 34



2. **สำหรับสัญญาณแบบรายคาบ** กำหนดได้โดยการใช้เมาส์คลิกเลือกจุดต้องการตั้งค่าสัญญาณ เช่นเดียวกับข้างต้นคือจะได้แถบสีน้ำเงินขึ้นมา ก่อน จากนั้นก็เลือกเมนู

**Edit > Value > Clock...**

หรือใช้เมาส์ชี้เลือกที่ **สัญลักษณ์รูปนาฬิกา** ก็จะปรากฏหน้าต่างสำหรับตั้งค่าคาบเวลา (time period) มาให้ดังรูป

**คำแนะนำ**

หากเรากำหนดให้ C มี Period = 400.0 ns (แทน 50)

B มี Period = 200.0 ns,

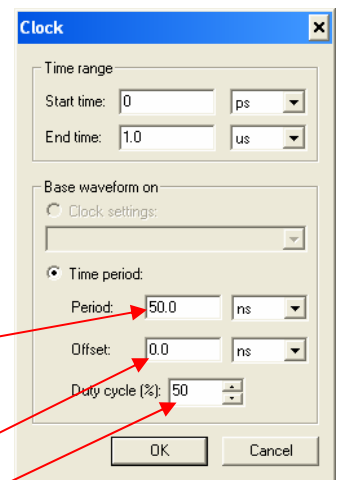
A มี Period = 100.0 ns

โดยที่ทั้ง A, B, C มีค่า time Offset = 0 ns

และ A, B, C มีค่า duty cycle = 50 %

จะได้แผนภาพทางเวลาสำหรับสัญญาณอินพุต A, B, C

ดัง timing diagram ในรูปที่ 34 เช่นเดียวกัน



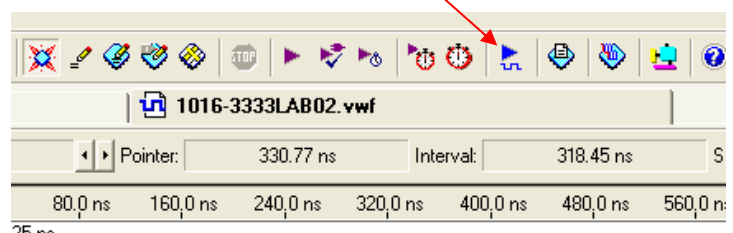
รูปที่ 35

3. **ทำการ save ไฟล์**ก่อนที่จะจำลองการทำงานในขั้นตอนถัดไป

## ค) จำลองการทำงาน

จำลองการทำงานของวงจรด้วยการคลิกเมาส์ที่ปุ่ม **Start Simulation** จากนั้นโปรแกรมก็จะทำการประมวลผลจนเสร็จและแสดงผลลัพธ์ที่ได้ในหน้าต่าง simulation waveform

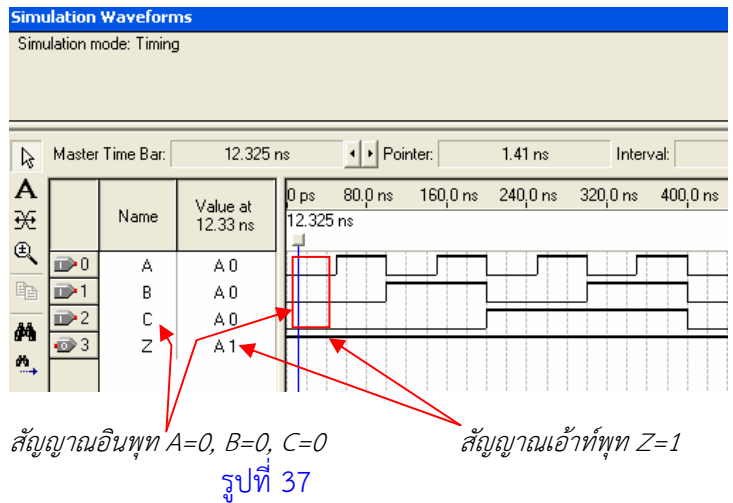
รูปที่ 36



## ง) ผลการจำลองการทำงานของวงจร

หน้าต่าง simulation waveform จะเห็นกราฟแสดงแผนภาพทางเวลา (timing diagram) ของสัญญาณอินพุต Z ที่ได้จากการจำลองการทำงานในขั้นตอนที่ผ่านมา

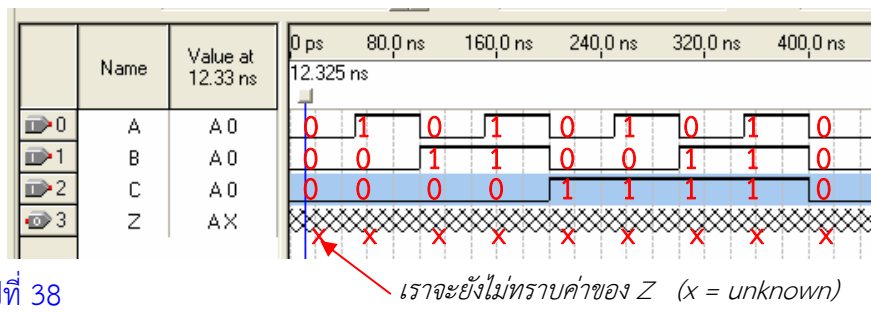
จากแผนภาพทางเวลา หากเรามองรูปกราฟในแนวดิ่ง ทั้ง A, B, C ที่เป็นอินพุต และ Z ที่เป็นเอาต์พุต จะเป็นการมองความสัมพันธ์กันของทั้ง 4 สัญญาณในช่วงเวลาเดียวกัน ดังนั้นรูปกราฟแบบนี้จึงถูกเรียกว่า **Timing diagram** ซึ่งมีประโยชน์มากคือทำให้เห็นค่าของ Z ได้ทุกๆ กรณีที่ค่าของอินพุตเปลี่ยนแปลง (ในช่วงเวลาที่แตกต่างกัน)



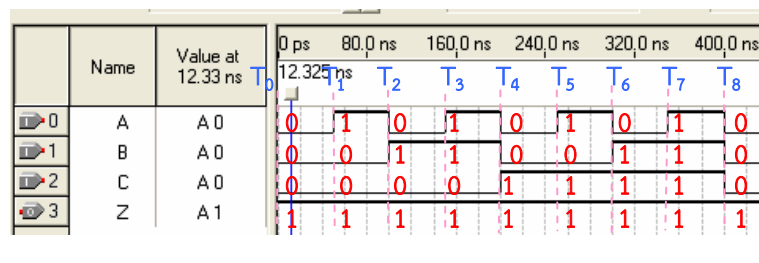
### ข้อแนะนำ การดูความสัมพันธ์กันของแผนภาพ Timing Diagram และตารางความจริง (Truth Table)

เราจะลองจินตนาการว่า ให้มีลอจิก '0' และ '1' ถูกแทนลงไปบนแผนภาพทางเวลา เพื่อแสดงค่าทางลอจิกของกราฟ Timing Diagram จากนั้นลองเอากราฟแยกออกไปให้เหลือแค่ตัวเลข Table ดังในรูปที่ 38-39

a) แผนภาพทางเวลาก่อนการจำลองการทำงาน (ยังไม่รู้ค่า Z จึงให้ค่าเป็น x หรือ unknown )



b) หลังจากผ่านการจำลองการทำงาน จะปรากฏค่าของ Z ด้วย เรากำหนดค่า  $T_0 \dots T_8$  ในแต่ละการเปลี่ยนแปลงขอบ



c) จะพบว่าค่าลอจิกทางฝั่งขวาของเส้นประของ  $T_n$  จัดเรียงตัวเหมือนในตารางความจริง แต่วางตัวในแนวตะแคงเท่านั้น

		$T_0$	$T_1$	$T_2$	$T_3$	$T_4$	$T_5$	$T_6$	$T_7$	$T_8$
Input	A	0	1	0	1	0	1	0	1	0
	B	0	0	1	1	0	0	1	1	0
	C	0	0	0	0	1	1	1	1	0
Output	Z	1	1	1	1	1	1	1	1	1

## ตอนที่ 2

### การสร้างชิ้นงาน Project ด้วยโปรแกรมภาษา VHDL

การสร้างชิ้นงานต้นแบบ ในงานดิจิทัลสามารถสร้างได้หลายแบบเช่น

- สร้างชิ้นงานต้นแบบโดยใช้ **graphic** ซึ่งจะทำให้ชิ้นงานอยู่ในรูปของ วงจรแบบ **logic diagram** (รายละเอียดได้กล่าวไว้ในขั้นตอนข้อที่ 5 – 8 ของเอกสารนี้)
- สร้างชิ้นงานต้นแบบโดยใช้โปรแกรมภาษา **VHDL** ซึ่งจะทำให้ชิ้นงานอยู่ในรูปของโค้ดโปรแกรม VHDL (จะกล่าวอย่างละเอียดในขั้นตอนจากนี้ไป)
- สร้างชิ้นงานต้นแบบโดยใช้แผนภาพทางสถานะ **Finite State Machine** (จะกล่าวอย่างละเอียดในเอกสารการทดลองที่ 9 และ 10 )

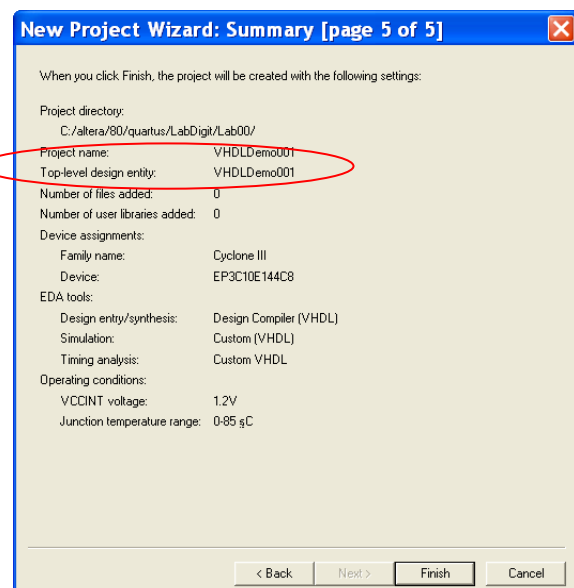
หมายเหตุ : มีเครื่องมืออื่นๆ อีกหลายชนิดเช่น ภาษา AHDL , Verilog ที่สามารถนำมาใช้ได้แต่ไม่กล่าวในที่นี้

#### 9. การเตรียมสร้างชิ้นงานต้นแบบ ( Design ) ด้วยการใช้ภาษา VHDL

- 9.1) **เตรียมการสร้างโปรเจค** เราจะเตรียมสร้างโปรเจคเช่นเดียวกันกับในขั้นที่ 1 ถึง 4 (หน้า 1- 5 ) หากยังไม่คล่องพอ ก็ขอให้กลับไปศึกษาให้จนสามารถดำเนินการได้อย่างถูกต้องเสียก่อน

**ในขั้นนี้เราสมมติว่า** เราสร้างโปรเจค(ขั้นที่ 1 ถึง 4) เสร็จแล้ว โดยใช้ชื่อเป็น **VHDLdemo001** รายละเอียดดังรูปที่ 40

รูปที่ 40



- 9.2) **เตรียมสร้าง ไฟล์ VHDL** ด้วย text editor ( น.ศ. สามารถใช้โปรแกรม Notepad บนวินโดว์เขียนก็ได้เช่นกัน)  
ในขั้นตอนนี้จะเป็นการสร้างไฟล์ชิ้นงานต้นแบบ โดยใช้ **Text editor Tool** ของ Quartus II

9.2.1) สร้างไฟล์นามสกุล \*.vhd ทำได้โดยไปที่หน้าต่างหลัก เรียกใช้ Text editor โดยเลือกที่เมนู

**FILE > NEW**

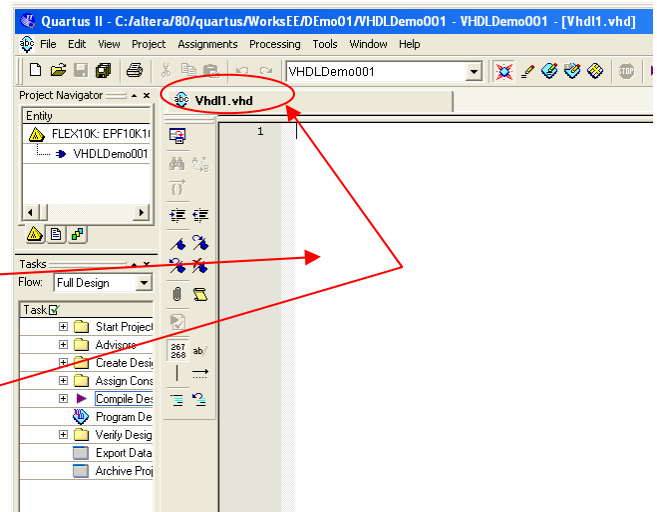
จะปรากฏหน้าต่างย่อยขึ้นดังรูปที่ 41

ให้เลือก Design files เป็นแบบ **VHDL file**

จากนั้นก็กดปุ่ม **OK**

สังเกตเห็นว่าโปรแกรมจะเปิดหน้าต่างว่างๆ ไว้ให้เราเขียนคำสั่งภาษา VHDL และได้ตั้งชื่อไฟล์ให้เราอย่างอัตโนมัติเป็น **Vhdl1.vhd** ด้วย (เราจะต้องเปลี่ยนชื่อให้ตรงกับกับชื่อของโปรเจค)

รูปที่ 41



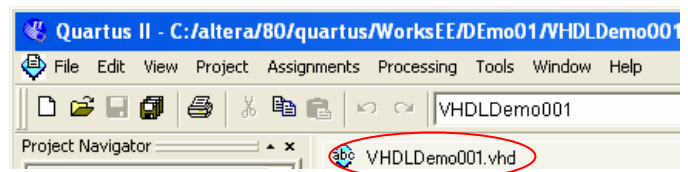
เราจะต้องทำการเปลี่ยนชื่อจาก ไฟล์เดิมคือ **Vhdl1.vhd** ให้เป็นชื่อเดียวกันกับชื่อของโปรเจคซึ่งในที่นี้เราใช้ชื่อว่า

**VHLDemo001.vhd**

โดยเลือกเมนู

**FILE > Save As ...**

จากนั้นเปลี่ยนชื่อไฟล์ให้ตรงกับชื่อของโปรเจคซึ่งได้ตั้งชื่อไว้เป็น **VHLDemo001.vhd** ตามข้อ 9.1



รูปที่ 42

9.2.2) ทำการพิมพ์โค้ดภาษา VHDL ใน

รูปที่ 43 ลงไปบน หน้าต่าง Text

editor ของ Quartus II ให้ถูกต้อง

รูปที่ 43

```
library ieee;
use ieee.std_logic_1164.all;

entity VHLDemo001 is
    port
    (
        clk,reset,d      : in std_logic;
        q                 : out std_logic );
end entity;

architecture rtl of VHLDemo001 is
    signal cnt          : std_logic;
begin
    process (clk)
    begin
        if (rising_edge(clk)) then
            if reset = '0' then
                cnt <= d;
            end if;
        end if;
    end process;
    q <= cnt;
end rtl;
```

เมื่อพิมพ์เสร็จ และ**ตรวจสอบความถูกต้อง** โดยขอให้ตรวจสอบส่วนต่างๆ ดังนี้

- โค้ดของ VHDL ข้างต้น ตรงตามหลัก Syntax
- จบคำสั่งด้วย**เครื่องหมาย ;** (semicolon)
- **ความตรงกันของชื่อไฟล์ ชื่อโปรเจกต์** เช่นเดียวกันกับขั้นตอนที่ 6

(ในขั้นนี้ หากเกิดการคอมไพล์ไม่สำเร็จ

จะมีเพียงสาเหตุเดียวคือ เกิดจากการพิมพ์ผิด !)

```

1  library ieee;
2  use ieee.std_logic_1164.all;
3
4  =entity VHDLDemo001 is
5  port
6  = ( clk,reset,d : in std_logic;
7    q      : out std_logic ); end entity;
8
9  =architecture rtl of VHDLDemo001 is
10 signal cnt : std_logic;
11 begin
12 = process (clk)
13 begin
14 = if (rising_edge(clk)) then
15 = if reset = '0' then
16 = cnt <= d;
17 = end if;
18 = end if;
19 = end process;
20 q <= cnt;
21 end rtl;
22

```

หมายเลขบรรทัดของคำสั่งในโปรแกรม

คำสั่ง คำสว่นของ VHDL จะแสดงด้วยอักษรสีน้ำเงิน

ตัวแปรต่างๆ ที่เราดังขึ้นมาเองจะแสดงด้วยอักษรสีดำ

เครื่องหมายต่างๆ และโอเปอเรเตอร์จะแสดงด้วยอักษรสีดำ

รูปที่ 44

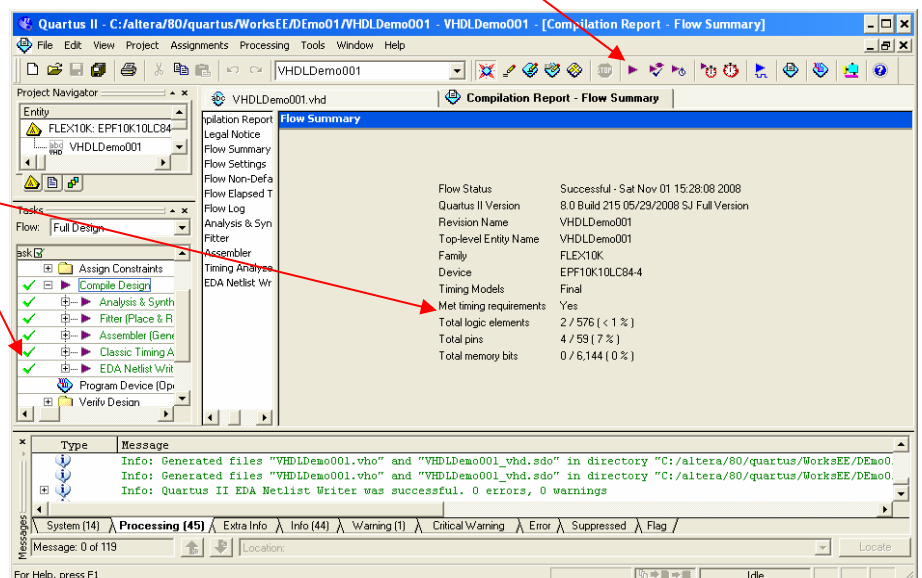
หมายเหตุ การกำหนดสีของ text เราสามารถกำหนดค่าเองได้ในเมนู preference seting

9.2.3) เมื่อตรวจสอบเสร็จแล้วให้ทำการคอมไพล์โดยกดปุ่ม hot Icon



ที่อยู่บนหน้าต่างหลักเพื่อทำการคอมไพล์

หากไม่มีข้อผิดพลาดใดเกิดขึ้น การคอมไพล์ก็จะสำเร็จ และแสดงหน้าจอดังรูปที่ 45 (แสดงผลการคอมไพล์ และมีเครื่องหมายถูกในแต่ละขั้นตอนของการคอมไพล์)



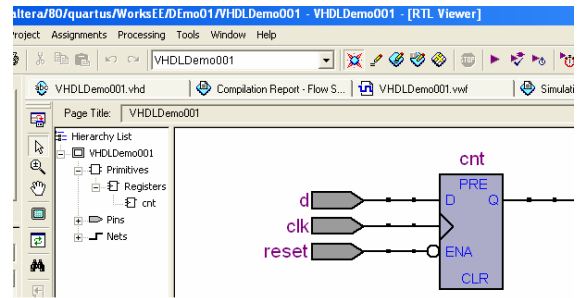
รูปที่ 45

หากการคอมไพล์สำเร็จ ก็สามารถจำลองการทำงานของชิ้นงานต้นแบบตามขั้นตอนที่ 8 ได้ หรือหากสนใจจะดูว่าชิ้นงานนี้ถูกคอมไพล์แล้วมันมีรูปร่างของวงจรลอจิกหน้าตาเป็นอย่างไรก็สามารถเข้าไปดูในแบบ RTL ได้เช่นกัน

ดูผลลัพธ์จากการคอมไพล์จากภาษา VHDL มาให้อยู่ในรูปแบบของวงจรลอจิก (Logic diagram) ได้โดยเลือกที่เมนู

Tools > Netlist Viewers > RTL Viewer

ก็จะแสดง Design ของ Entity ที่ชื่อว่า VHLDemo001 ในรูปแบบของ Logic Diagram ดังรูปที่ 46



รูปที่ 46

#### 10) การจำลองการทำงาน แบบเลือกโหมดการจำลองการทำงาน

โหมดจำลองการทำงานของ Quartus II มี 2 โหมดคือ

- โหมด Functional Simulation เป็นการจำลองการทำงานตามสมการลอจิกโดยตรงซึ่งเสมือนเป็นการจำลองแบบในอุดมคติ
- โหมด Timing Simulation เป็นการจำลองการทำงานแบบคำนึงถึงความเป็นจริงที่ว่าสัญญาณจะต้องใช้เวลาเดินทางในสายตัวนำทำให้เกิดเวลาหน่วง (delay time)

##### 10.1) สร้างไฟล์สำหรับเก็บผลการจำลอง

โดยไปที่หน้าต่างหลักและเลือกเมนู

FILE > NEW

เลือกสร้างไฟล์แบบ

Vector Waveform file

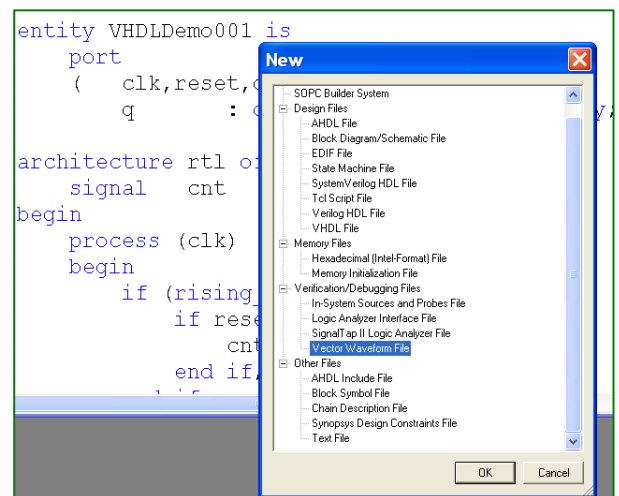
ดังรูปที่ 47

กดปุ่ม OK

โปรแกรมก็จะสร้างไฟล์ที่มีนามสกุล

\*\*\*.vwf ขึ้นมาให้

(ขั้นตอนเช่นเดียวกันกับในข้อที่ 8.1)



รูปที่ 47

##### 10.2) เลือกสัญญาณอินพุต/เอาต์พุต

ที่ต้องการจะให้เห็นบนกราฟ

Timing Diagram ซึ่งในที่นี้จะ

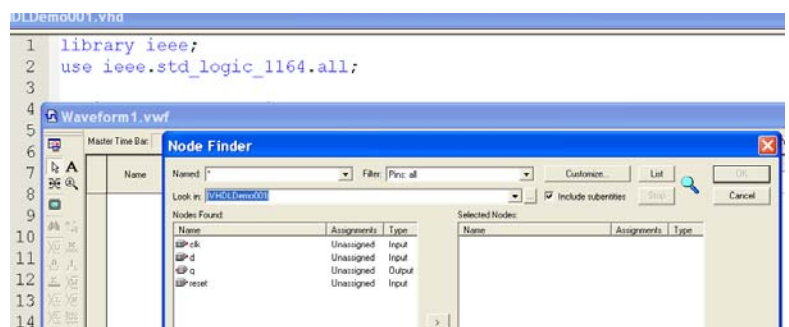
เลือกทั้งหมดคือ clk, d, q และ

Reset โดยไปที่เมนู

Edit > Insert

จะปรากฏหน้าต่างดังรูปที่ 48

(ขั้นตอนเช่นเดียวกันกับในข้อที่ 8.2)



รูปที่ 48



## 10.3) สร้างสัญญาณนาฬิกา clk

ด้วยการใช้เมาส์คลิกเลือกที่ชื่อสัญญาณ clk  
จะปรากฏแถบสีน้ำเงินบนรูปภาพ

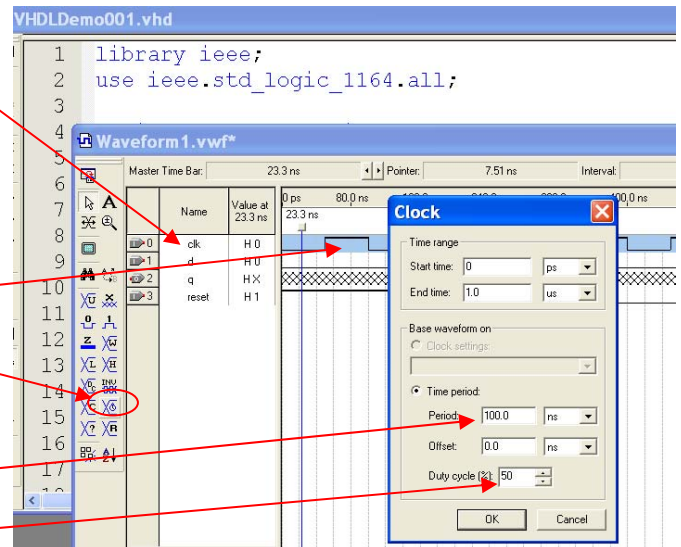
เลือกเครื่องมือสร้างกราฟแบบ clock

กำหนดคาบของสัญญาณเป็นแบบ

Period 100.0 ns

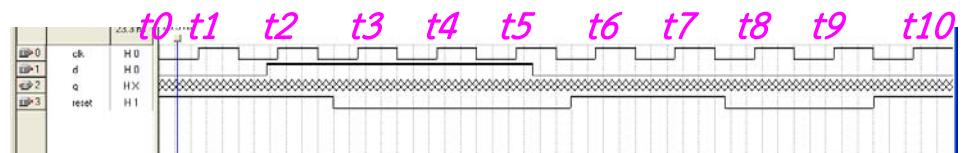
มีค่าลอจิก '1' และ '0' อย่างละครึ่ง

(ขั้นตอนเช่นเดียวกันกับในข้อที่ 8.3)



รูปที่ 49

## 10.4) สร้างสัญญาณอินพุต d และ reset



รูปที่ 50

**ข้อตกลง** ก่อนจะสร้างสัญญาณอินพุตขอให้เรามาทำความเข้าใจกันก่อนดังนี้

- จุดเริ่มต้นกราฟซ้ายสุด จะตั้งชื่อเป็น  $t_0$  เสมอ
- จุดที่มีการเปลี่ยนขอบของสัญญาณ clk จาก '0' ไปเป็น '1' จะตั้งชื่อเป็น  $t_1, t_2, t_3, \dots$  ไปจนถึงจำนวนตามเท่าที่ต้องการจะสื่อ (ชื่อเหล่านี้เราสมมุติขึ้นมาเพื่อให้สื่อสารกันได้เข้าใจตรงกันเท่านั้น น.ศ.จะไม่สามารถเขียนลงไปในการ์ดของ Quartus II ได้)

**สร้างสัญญาณอินพุต** (อ้างอิงจากตารางความจริงของ D Flip-Flop)

Reset = '0'	D = '0'	เกิดขึ้นที่ตำแหน่งตรงกับ	$t_8, t_9$
Reset = '1'	D = '0'	เกิดขึ้นที่ตำแหน่งตรงกับ	$t_1, t_6, t_7, t_{10}$
Reset = '0'	D = '1'	เกิดขึ้นที่ตำแหน่งตรงกับ	$t_3, t_4, t_5$
Reset = '1'	D = '1'	เกิดขึ้นที่ตำแหน่งตรงกับ	$t_2$

**คำสั่ง**

ให้ น.ศ. ทดลองสร้างสัญญาณ d และ reset เพื่อจำลองการทำงานให้ครอบคลุมตามตารางความจริงโดย

ก) ให้ตรงตามที่กำหนดข้างต้น (ขอให้มึรูปร่างคล้ายกับในรูปที่ 50 ก็พอ)

หรือ ข) น.ศ. สามารถสร้างรูปสัญญาณ ไม่จำเป็นต้องเหมือนกับในรูปที่ 50 แต่ค่าของ d และ reset ต้องมีค่าครบทุกสถานะ (state) ตามตารางความจริง

จากนั้นก็ให้บันทึกไฟล์ (ชื่อไฟล์ VHLDemo001.vwf) และทำการจำลองการทำงานในขั้นตอนถัดไป

## 10.5) เลือกโหมดจำลองการทำงาน

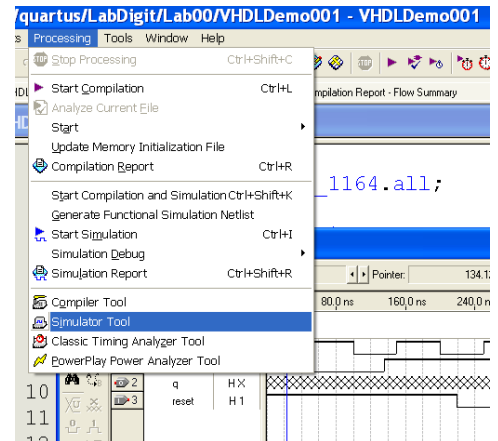
โดยไปที่เมนู

Processing &gt; Simulator Tool

ดังรูปที่ 51

จากนั้นจะปรากฏหน้าต่างของ  
Simulator Tool ขึ้นมาดังรูปที่ 52

รูปที่ 51



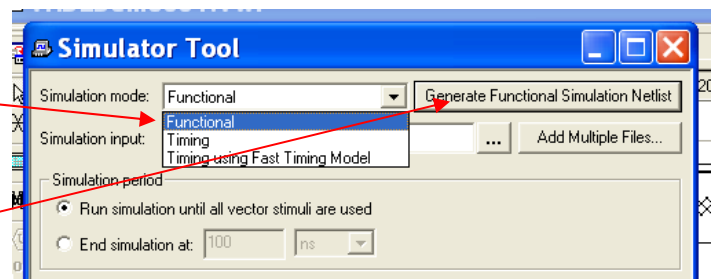
เลือกโหมดการจำลองเป็นแบบ

Functional (หรือแบบอุดมคติ)

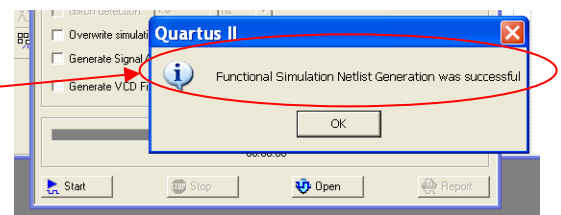
และสั่งให้ทำการประมวลผลวงจร

โดยกดปุ่ม Generate Function  
Simulation Netlist

รูปที่ 52

เมื่อประมวลผลเสร็จจะปรากฏหน้าต่าง  
แจ้งสถานะดังรูปที่ 53

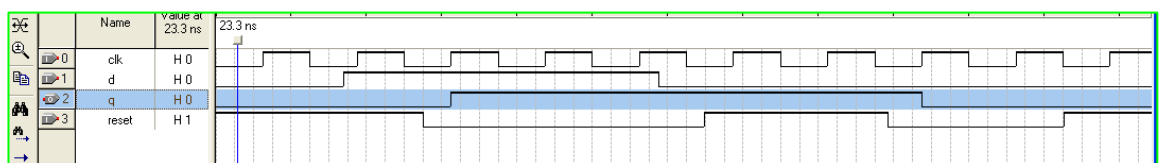
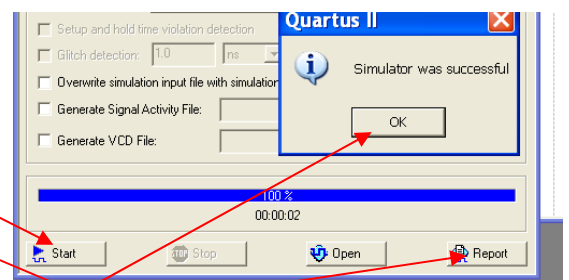
รูปที่ 53



จากนั้นก็กดปุ่ม **Start**  
เพื่อเริ่มจำลองการทำงาน  
เมื่อการจำลองการทำงานเสร็จสิ้น  
จะปรากฏหน้าต่างแสดงสถานะดังรูปที่ 54  
ให้กดปุ่ม OK และ Report  
ตามลำดับ ก็จะได้กราฟผลการจำลอง  
การทำงาน (ค่าของ q) ดังรูปที่ 55

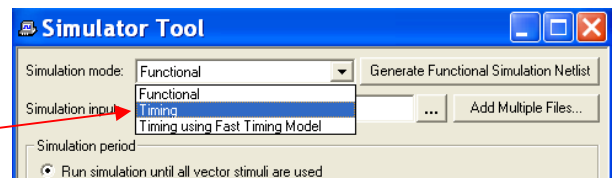
(น.ศ.ควรจะลองเปรียบเทียบค่าของ q นี้กับตารางความจริงของ D Flip-Flop ดูด้วยเพื่อให้มีความเข้าใจมากขึ้น)

รูปที่ 54

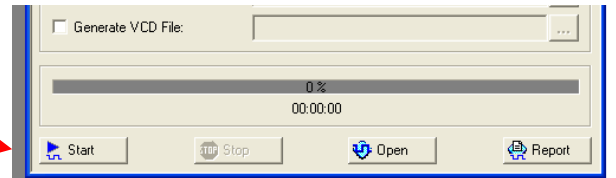


รูปที่ 55

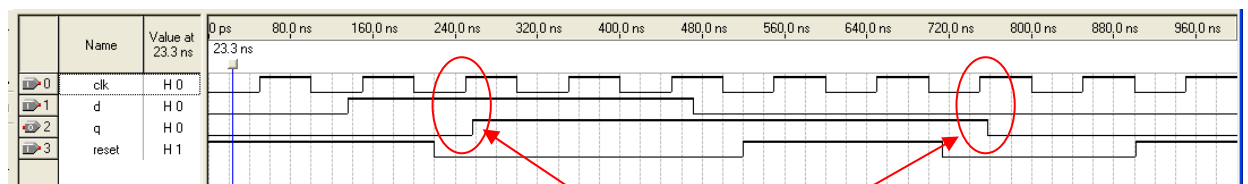
ให้ลองเปลี่ยนโหมดจำลองการทำงานเป็นแบบ Timing (แบบเวลาจริง ไม่ใช่ในอุดมคติ) ดูบ้าง ด้วยการเลือก timing ในหน้าต่าง Simulation Tool ดังรูปที่ 56



เมื่อกดปุ่ม Start ก็จะได้ผลการจำลองการทำงานดังรูปที่ 57



รูปที่ 56



รูปที่ 57

การจำลองการทำงาน แบบ timing diagram  
ช่วยให้เห็น ปรากฏการณ์ของเกิดเวลาหน่วง time delay

### คำแนะนำ

1. เปรียบเทียบผลการจำลองการทำงานโหมด Functional (รูปที่ 55) และ Timing (รูปที่ 57) จะพบว่ามีความแตกต่างกันอย่างเห็นได้ชัดที่ค่าหน่วงเวลา (time delay) ของสัญญาณเข้าที่พอร์ท q (น.ศ. สามารถหาความรู้เพิ่มเติมได้จากในตำราเรียน "Fundamental of Digital Logic with VHDL Design 3<sup>rd</sup> edition" หน้า 398)
2. หากสังเกตกราฟแสดง timing diagram ที่ได้จากการจำลองการทำงานดังรูปที่ 55 และ 57 จะพบว่า การเปลี่ยนแปลงของ q เป็นไปตามคำสั่งในภาษา VHDL ที่บรรยายพฤติกรรมของ D ฟลิปฟล็อป
3. ให้ลอง zoom ขยายดูการเกิด delay time ของสัญญาณ q เมื่อเทียบกับ d และ clk จะทำให้น.ศ. เข้าใจถึงข้อจำกัดหรือธรรมชาติของระบบหรืออุปกรณ์ดิจิทัลได้ดียิ่งขึ้น

## ตอนที่ 3

### การสร้างชิ้นงานจริงด้วยการทำ Chip Configuration ลงในชิพ FPGA

#### 11. การสร้างชิ้นงานจริงบนชิพ FPGA

การสร้างชิ้นงานจริง (Implementation) เป็นการนำเอา Design ของเราที่ทำการออกแบบ (รวมทั้งผ่านการดูกลไกการทำงานด้วยการจำลองสัญญาณ) มาทำการโปรแกรม (โดยปกติจะเรียกว่าเป็นการทำ configuration ซึ่งเป็นคำเฉพาะของงานด้าน FPGA development ไม่ใช่คำว่า ดาวน์โหลด เหมือนในคอนโทรลเลอร์)

##### 11.1) ออกแบบชิ้นงาน

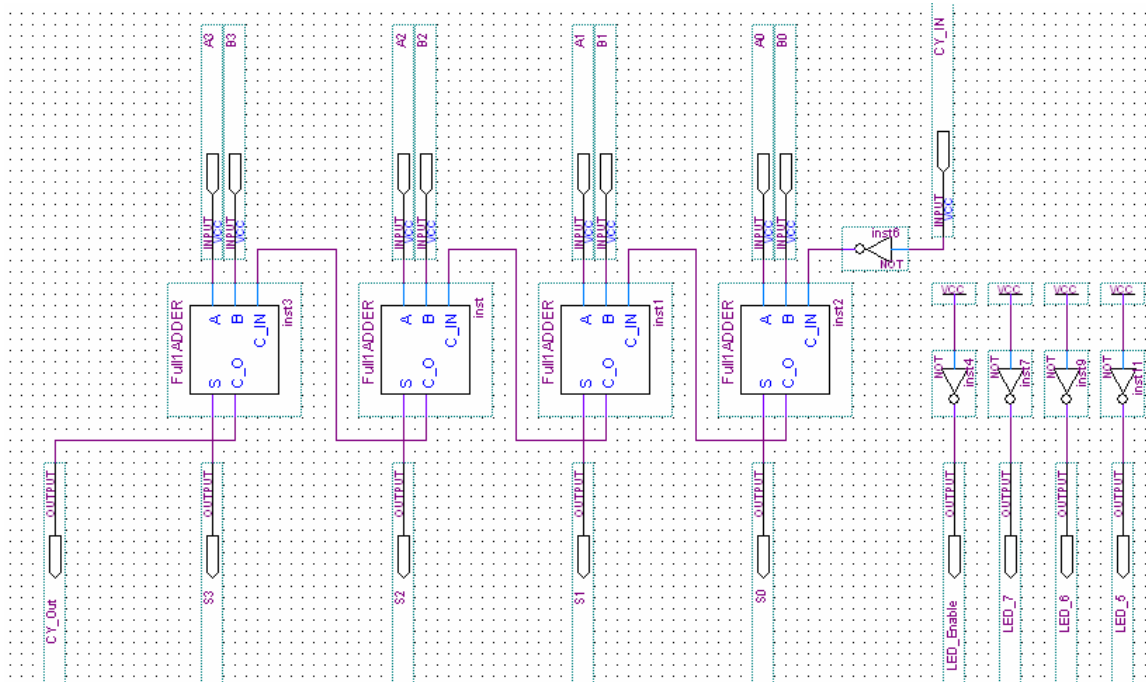
เพื่อให้เห็นภาพของการสร้างชิ้นงานได้ชัดเจน ในการทดลองตอนที่ 3 นี้จะใช้วงจรบวกเลขแบบไบนารี Full Adder ขนาด 4 บิตสองจำนวนที่ได้จากการกดสวิทช์ ค่าผลบวกที่จะแสดงเป็นไบนารีขนาด 4 บิตพร้อมกับบิตตัวทดอีก 1 บิต ส่งออกแสดงผลด้วยหลอด LED (เราจะไม่ใช้วงจรที่เหมือนกันกับในตอนที่ 1 และ 2 เพราะจะทำให้เราสังเกตด้วยสายตาได้ยาก)

#### คำเตือน

ในการทดลองที่ 2 เนื่องจาก น.ศ. ต้องดำเนินการบนบอร์ดทดลอง ซึ่งมีลักษณะเป็นบอร์ดทดลองแบบเอนกประสงค์ มีอุปกรณ์ที่ทำหน้าที่เป็น ตัวป้อนอินพุต หลายชนิด เช่น สวิทช์ และพอร์ตเชื่อมต่อ I/O ตัวแสดงผลแบบ LED แบบ LCD และ Buzzer เป็นต้น ที่เชื่อมต่อวงจรโดยตรงกับขาของชิพ FPGA อยู่แล้ว (ไม่ว่าเราจะอยากใช้งานมันหรือไม่ก็ตาม) **ดังนั้นการจะใช้งานอุปกรณ์ต่างๆ บนบอร์ดทดลอง อาจจะต้องมีการกำหนดสถานะลอจิกพิเศษเพื่อให้เกิดการ enable ขาบางขาของชิพ FPGA ก่อนจึงจะสามารถใช้งานอุปกรณ์ที่ต้องการได้** จึงขอให้ น.ศ. ทำการศึกษาคู่มือของบอร์ดทดลองให้เข้าใจ ก่อนที่จะทำการทดลอง (ดูรายละเอียดใน Lab sheet และคู่มือของบอร์ด) หากมีข้อสงสัยใด ๆ โปรดแจ้ง และปรึกษาอาจารย์ผู้สอนทันที เพื่อป้องกันความเสียหายที่จะเกิดขึ้นกับบอร์ดทดลอง

**ชิ้นงานต้นแบบ** ในขั้นนี้สมมุติว่า เราสร้างชิ้นงานต้นแบบซึ่งเป็นวงจรบวกเลขไบนารีขนาด 4 บิต ดังรูปที่ 58 ด้วยขั้นตอนที่ 1 ถึง 8 จนเสร็จสมบูรณ์เรียบร้อยแล้ว

วงจรต้นแบบ Full Adder ขนาด 4 bit ดังรูป จะถูกนำไปสร้างจริงบนชิพ และมีการป้อนไฟเข้าระบบเพื่อทดสอบการทำงานของมันว่าสามารถทำงานได้ตามทฤษฎีหรือไม่



รูปที่ 58

**การดำเนินการ** (รายละเอียดตามขั้นตอนที่ 1 – 8 ในหน้าที่ 1 ถึง 16)

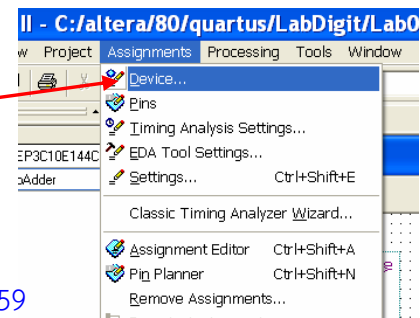
1. ต้องสร้างโปรเจกใหม่ (ให้ ตั้งชื่อเป็น Add4bit หรือ น.ศ.จะตั้งชื่ออื่นก็ได้)
2. ออกแบบชิ้นงานโดยใช้ Graphic design tool
3. คอมไพล์ชิ้นงานโปรเจก
4. ลองทำการจำลองการทำงานของวงจรด้วย timing diagram ดูก่อน เพื่อจะให้เห็นใจได้ว่าเมื่อเราทดสอบด้วยบอร์ดทดลอง FPGA แล้วจะเปรียบเทียบผลได้
5. เข้าสู่ขั้นตอนของการทำ configuration ที่จะได้กล่าวถึงต่อไป

#### 11.2) กำหนดรายละเอียดของชิพ FPGA บนบอร์ดทดลอง

ก่อนที่จะสร้างชิ้นงานจริงบนบอร์ดทดลอง เราจำเป็นจะต้องทำการกำหนดระดับแรงดันลอจิกที่จะใช้ในระบบก่อนว่า ลอจิก '0' และ '1' จะมีศักย์ไฟฟ้ากี่โวลต์ ทั้งนี้โดยปกติจะมีค่าตารางมาตรฐานอยู่แล้วเราเพียงแต่เลือกให้ถูกต้องแค่นั้น ในห้องแล็บของเราบอร์ดทดลอง มีค่าแรงดันลอจิกเป็นไปตามมาตรฐาน LVTTTL (Low voltage TTL) หากมีข้อสงสัยขอให้ดูตารางท้ายเอกสารประกอบ

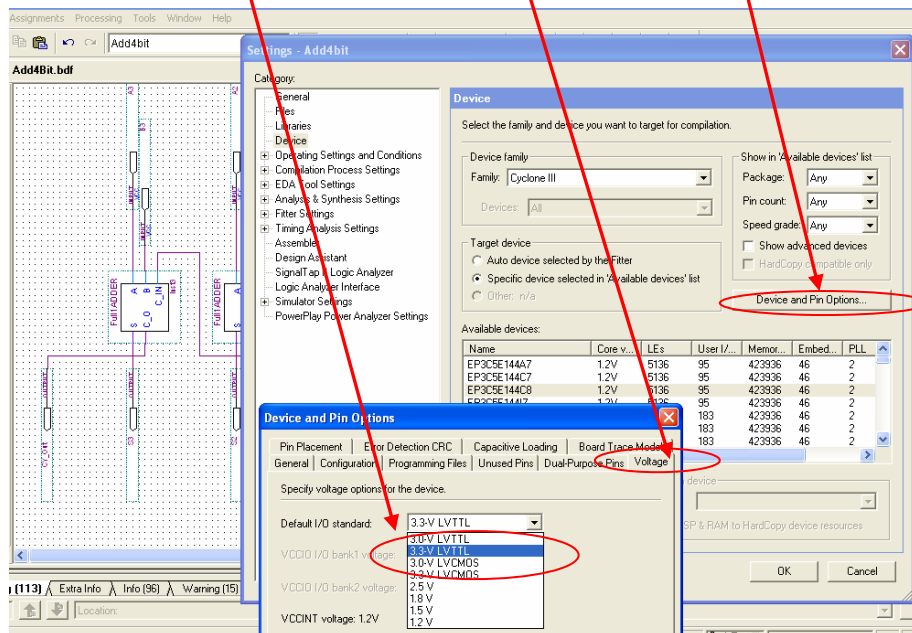
ก) กำหนดระดับแรงดันลอจิก ด้วยการเปิดเมนู

Assignments → Device...



รูปที่ 59

จะปรากฏหน้าต่าง Settings ขึ้นมาดังรูปที่ 60 ให้เลือกปุ่ม Device and Pin Options...  
จากนั้นเลือก แท็บ Voltage  
เลือกให้ระดับแรงดันเป็น 3.3V-LVTTL ดังรูป



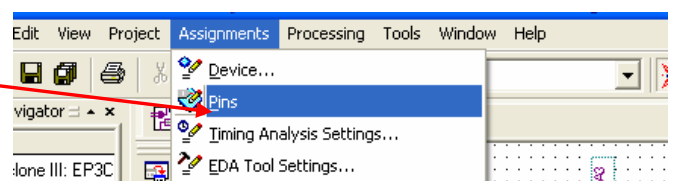
รูปที่ 60

ข) กำหนดขา (PIN) ของชิพ

เพื่อเลือกใช้ขาของชิพที่ต่อวงจรไว้กับสวิตช์ที่อยู่บนบอร์ดทดลอง และ LED (ในขั้นตอนนี้น.ศ. จะต้องเปิดคู่มือการใช้งานของบอร์ดเพื่อดูรายละเอียดให้ถูกต้องด้วย) เพื่อที่จะกำหนด ขาอินพุต ( $A_n, B_n, CY\_IN$ ) และเอาต์พุต ( $S_n, CY\_OUT$ ) ของวงจร 4 Bit Full Adder ให้สามารถทำการสร้าง (Synthesis) หรือจัดวางตำแหน่งได้อย่างถูกต้องบนชิพ FPGA

ดำเนินการด้วยการเปิดเมนู

Assignment → Pins



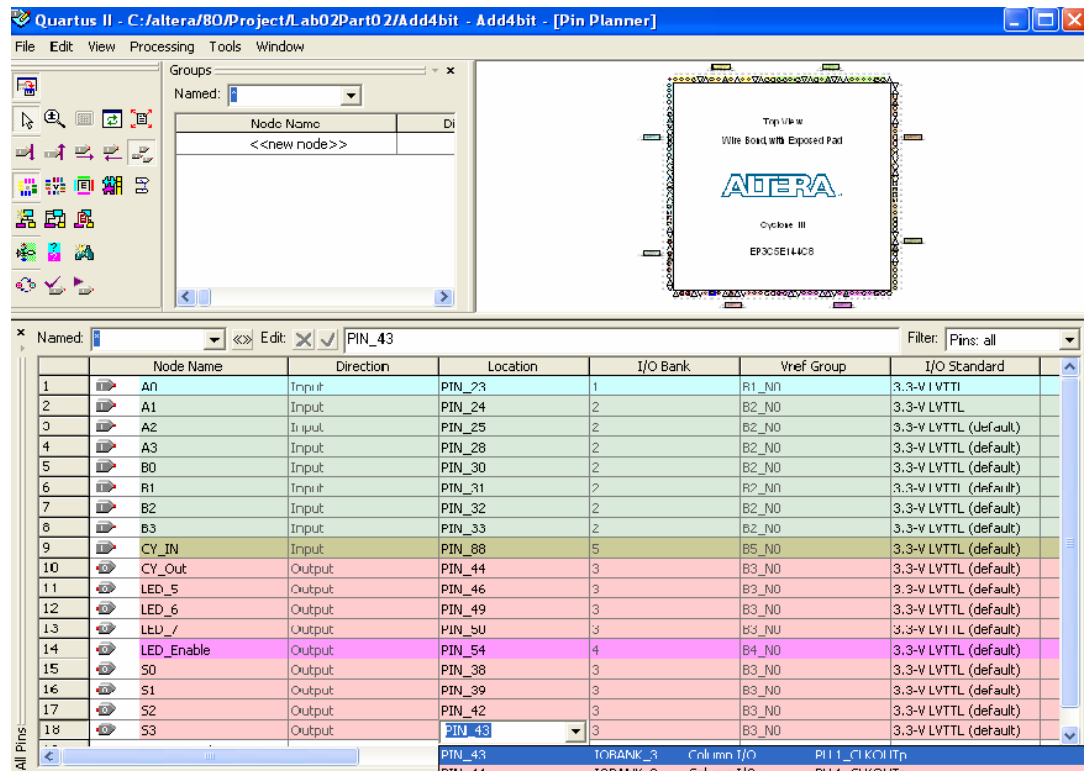
จะปรากฏหน้าต่าง Pin Planner ดังรูปที่ 61-62

รูปที่ 61



ค) หน้าต่างการกำหนดค่าต่างๆ ของ PIN ใน FPGA จะปรากฏดังรูป

รูปที่ 62



จะสังเกตเห็นว่าโปรแกรมได้กำหนดค่าของวงจร FULL Adder ไว้ให้แล้ว แต่หากดูดีๆเทียบกับคู่มือของบอร์ดทดลองของเรา ขา pin จะยังไม่ได้ตำแหน่งตามต้องการเช่น ขาที่ควรต่อกับสวิตช์กลับไม่ได้ต่อไว้กับสวิตช์เป็นต้น ดังนั้นเราจำเป็นต้องทำการจัดและย้ายขาของ วงจร FULL Adder เสียใหม่ให้ตรงกับ PIN ของ FPGA ที่มีอุปกรณ์จำพวก สวิตช์ และ LED ต่อไว้ ดังตำแหน่งที่ให้ไว้ในตารางด้านล่าง  
(ตารางนี้คัดมาจากคู่มือของบอร์ด CYCLONE III)

ตารางแสดง ขาของอุปกรณ์ 4 Bit full adder และ หมายเลข PIN หรือขาของชิพ FPGA ที่สอดคล้องกัน

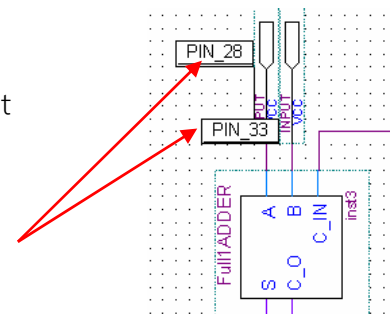
ขาของวงจร 4Bit Adder	A0	A1	A2	A3	B0	B1	B2	B3
หมายเลข PIN ของ FPGA	PIN_23	PIN_24	PIN_25	PIN_28	PIN_30	PIN_31	PIN_32	PIN_33

ขาของวงจร 4Bit Adder	CY_IN	CY_Out	LED_Enable	LED_5	LED_6	LED_7	S0	S1	S2	S3
หมายเลข PIN ของ FPGA	PIN_88	PIN_44	PIN_54	PIN_46	PIN_49	PIN_50	PIN_38	PIN_39	PIN_42	PIN_43

ง) กลับไปคอมไพล์ชิ้นงานต้นแบบอีกครั้ง เพื่อให้เกิดการย้าย PIN ตามต้องการ  
เมื่อกำหนดค่าของอุปกรณ์เสร็จแล้ว ให้ปิดหน้าต่าง Pin Assignment  
ทำการ save บันทึกไฟล์ แล้วจึง ทำการคอมไพล์ซ้ำอีกครั้ง

เมื่อคอมไพล์เสร็จจะสังเกตเห็นว่า รูปวงจร Block diagram จะมีรายละเอียดหมายเลข  
ต่างๆ ของขา PIN เพิ่มขึ้น (แสดงตำแหน่ง PIN ของชิพ) ดังในรูปที่ 63

รูปที่ 63




## 11.3) โปรแกรมชิป FPGA (การทำ Chip Configuration)

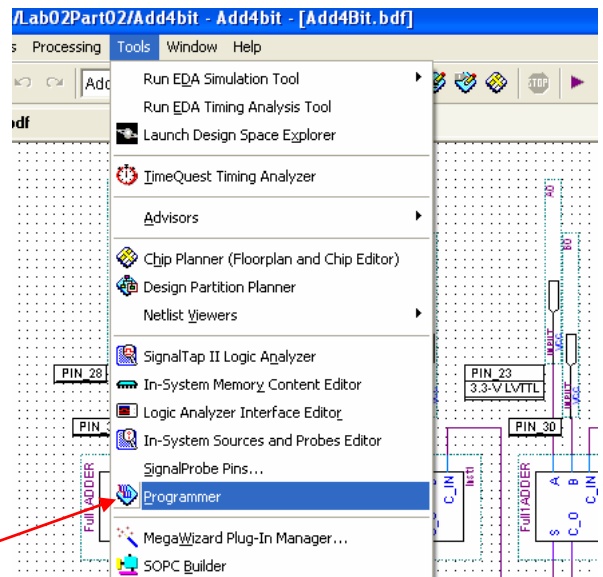
การโปรแกรมชิป FPGA เป็นการนำเอาชิ้นงานที่ออกแบบไว้บนคอมพิวเตอร์ใส่ลง ไปบนชิป FPGA ที่อยู่บนบอร์ดทดลองของเรา

- ก) เสียบสายดาวน์โหลด **JTAG** ระหว่างคอมพิวเตอร์กับบอร์ดทดลอง CYCLONE III (จ่ายไฟให้บอร์ดด้วย)
- จะใช้สาย **Byte Blaster** หากเป็น PC
  - จะใช้สาย **USB Blaster** หากเป็น Notebook

เริ่มด้วยการเปิดไปที่เมนู

**Tools > Programmer** ดังรูป

หรืออีกวิธีการหนึ่งก็สามารถกดปุ่ม hot icon  ที่อยู่บนแถบเครื่องมือของหน้าต่างหลักก็ได้



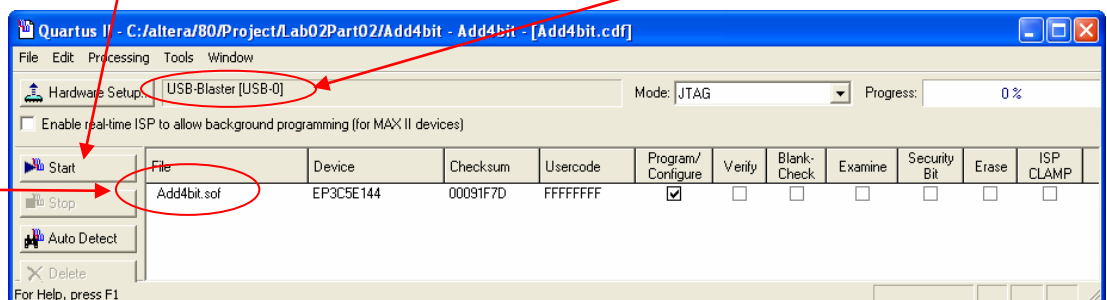
รูปที่ 64

**หมายเหตุ** ในกรณีที่เครื่องคอมพิวเตอร์ใช้พอร์ต USB (เครื่อง Notebook) จำจะต้องทำการติดตั้งโปรแกรมไดรเวอร์สำหรับใช้งาน **Logic Blaster** ด้วย (ขอให้ศึกษาขั้นตอนการติดตั้งจากคู่มือของโปรแกรมให้เข้าใจอย่างละเอียดด้วยเพื่อไม่ให้เกิดข้อผิดพลาดขึ้นได้ในภายหลัง)

- ข) ในกรณีที่เครื่องคอมพิวเตอร์ใช้พอร์ต USB หน้าต่าง Programmer จะปรากฏขึ้นมาดังรูปที่ 65 สังเกตเห็นว่าได้มีการแจ้งชื่อไฟล์ของชิ้นงานต้นแบบของเราที่ได้ออกแบบไว้ ในที่นี้คือไฟล์ Add4bit.sof และเครื่องมือที่ใช้ดาวน์โหลดโปรแกรม

ในที่นี้เราใช้ Note Book จึงแสดงเป็น **USB-Blaster [USB0]**

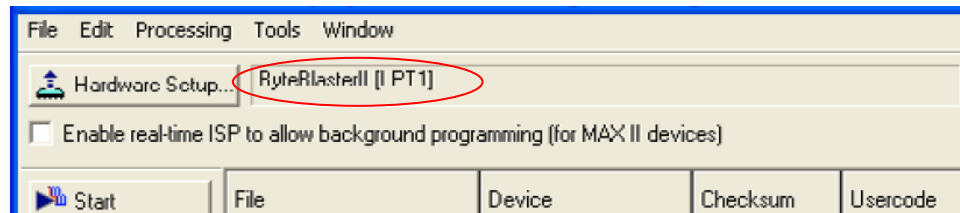
เมื่อกดปุ่ม start ระบบจะทำ chip configure ชิป FPGA ทันที



รูปที่ 65

ค) ในกรณีที่เครื่องคอมพิวเตอร์ ใช้พอร์ต LPT หรือ Parallel Port จะต้องติดตั้งโปรแกรมไดรเวอร์สำหรับใช้งาน Byte Blaster (ขอให้ศึกษาขั้นตอนการติดตั้งจากคู่มือของโปรแกรมให้เข้าใจอย่างละเอียดด้วย เพื่อไม่ให้เกิดข้อผิดพลาดขึ้นในภายหลัง)

หน้าต่าง Programmer จะปรากฏขึ้นมาดังรูปที่ 66



รูปที่ 66

สังเกตเห็นได้ว่าการแจ้งชื่อไฟล์ของงานที่ได้ออกแบบไว้ ในที่นี้คือ Add4bit.sof เช่นเดียวกันแต่ เครื่องมือที่ใช้โหลดขึ้นงานจะเป็น Byte-Blaster [LPT1] เมื่อกดปุ่ม start ระบบจะทำ chip configure ชิป FPGA ทันที

## 12) ทดสอบชิ้นงานจริง

เมื่อทำการโปรแกรมลงบอร์ดเสร็จแล้ว ก็สามารถทดสอบการทำงานได้จากการปรับสวิตช์แบบเลื่อน SW0 –SW7 พร้อมทั้งสังเกตการติด / ดับ ของหลอด LED ที่อยู่บนบอร์ดทดลอง WARRIOR CYCLONE III ว่าทำได้งานตรงกับความต้องการของเราที่ได้ออกแบบไว้หรือไม่

## จบ

ตารางแสดงมาตรฐานต่างๆ ของสถานะลอจิก '0' และ '1' เมื่อแสดงระดับแรงดันไฟฟ้า (หน่วยเป็นโวลต์)

