สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

บอร์ดแสดงผลตัวอักษรแบบดอทเมตริกซ์ควบคุมด้วยเอฟพีจีเอ

Dot- matrix Display Board controlled by FPGA

T050340

โดย

นายพีรกิจ ชังเจริญสุข

นายภาสกร เจตกวิน

นางสาวภาสินี เจริญขวัญ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2545

เลขหมู่.....
เลขทะเบียน.....50340
วัน,เดือน,ปี.....29 เม.ย. 2547
.b.....
.i.....

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บอร์ดแสดงผลแบบดอทเมตริกซ์ควบคุมด้วยเอฟพีจีเอ

Dot- matrix Display Board controlled by FPGA

โดย

นายพีรกิจ ชังเจริญสุข 42010237

นายภาสกร เจตกวิน 42010257

นางสาวภาสินี เจริญขวัญ 42010259

อาจารย์ที่ปรึกษา

รองศาสตราจารย์ ดร.มนัส สังวรศิลป์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2545

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2545

ภาควิชาอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง บอร์ดแสดงผลแบบดอทเมตริกซ์ควบคุมด้วยเอฟพีจีเอ

ผู้จัดทำ

1. นายพีรกิจ ชังเจริญสุข
2. นายภาสกร เจตกวิน
3. นางสาวภาสินี เจริญขวัญ

..... อาจารย์ที่ปรึกษา

(ร.ศ. ดร.มนัส สังวรศิลป์)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โครงงานเรื่อง บอร์ดแสดงผลแบบดอทเมตริกซ์ควบคุมด้วยเอฟพีจีเอ

Dot- matrix Display Board controlled by FPGA

จัดทำโดย

1. นายพีรกิจ ชังเจริญสุข 42010237
2. นายภาสกร เจตกวิน 42010257
3. นางสาวภาสินี เจริญขวัญ 42010259

โครงงานนี้ได้รับการตรวจสอบแล้ว พร้อมที่จะทำการสอบได้

.....

(ร.ศ. ดร. มนัส สังวรศิลป์)

อาจารย์ที่ปรึกษา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บอร์ดแสดงผลแบบดอทเมตริกซ์ควบคุมด้วยเอฟพีจีเอ

พีรกิจ ชังเจริญสุข
ภาสกร เจตกวิน
ภาสินี เจริญขวัญ
ร.ศ.ดร. มนัส สังวรศิลป์ อาจารย์ที่ปรึกษา
ปีการศึกษา 2545

บทคัดย่อ

โครงงานนี้ นำเสนอการออกแบบและสร้างบอร์ดแสดงผลแบบดอทเมตริกซ์ที่ใช้เอฟพีจีเอ (FPGA : Field Programable Gate Array) ควบคุมการทำงาน โดยสัญญาณอินพุทจะมาจากการกดแป้นตัวอักษรจากคีย์บอร์ดแบบพีเอสทู (PS/2) และผ่านการประมวลผลจากเอฟพีจีเอ และจากนั้นสัญญาณเอาท์พุทจะถูกส่งไปแสดงผลเป็นตัวอักษรที่ดอทเมตริกซ์ขนาด 8x8 ที่ต่อเรียงกันอยู่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Dot- matrix Display Board controlled by FPGA

Peerakit Chang-charoensuk

Passakorn Jettakawin

Pasinee Charoenkwan

Assoc. Prof. Dr. Manus Sangworasilp Advisor

Education Year 2002

Abstract

This Project presents the designing and invention of the Dot-matrix Display Board controlled by the FPGA (Field Programmable Gate Array).The character input signal is sent from the PS/2 keyboard via the processing by FPGA and output will display at 8x8 dot-matrixs .

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำนำ

โครงงานนี้นำเสนอการออกแบบและสร้างบอร์ดแสดงผลตัวอักษรด้วยดอทเมตริกซ์(LED Dot Matrix) ควบคุมการแสดงผลด้วยเอฟพีจีเอ (FPGA : Field Programmable Gate Array) ซึ่งเป็นอุปกรณ์ที่สามารถโปรแกรมได้ โดยมีจุดมุ่งหมายเพื่อศึกษาหลักการทำงานและแนวทางในการนำเอฟพีจีเอไปประยุกต์ใช้งานด้านอิเล็กทรอนิกส์ที่เกี่ยวกับการควบคุม (Control Circuit) เช่นการแสดงผลตัวอักษรวิ่ง และเพื่อศึกษาหลักการและสามารถเขียนโปรแกรมควบคุมการทำงานของวงจรด้วยภาษาวีเอชดีแอลซึ่งเป็นภาษาที่ใช้อธิบายพฤติกรรมของฮาร์ดแวร์ได้

รายละเอียดที่เกี่ยวข้องกับโครงงานได้อธิบายไว้ในรายงานฉบับนี้แล้ว หากมีข้อผิดพลาดประการใด ทางผู้จัดทำก็ขออภัยมา ณ ที่นี้ด้วย

นายพีรกิจ ชังเจริญสุข
นายภาสกร เจตกวิน
นางสาวภาสินี เจริญขวัญ
ผู้จัดทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

เรื่อง	หน้า
บทคัดย่อภาษาไทย	iii
บทคัดย่อภาษาอังกฤษ	iv
สารบัญรูปภาพ	viii
สารบัญตาราง	x
บทที่ 1 บทนำ	1
1.1 บทนำ	1
1.2 จุดประสงค์ของโครงงาน	1
1.3 ขอบเขตของโครงงาน	1
1.4 ลักษณะโดยทั่วไปของบอร์ดแสดงผล	2
1.4.1 คุณสมบัติของบอร์ดแสดงผล	2
1.4.2 บล็อคไดอะแกรมของโครงงาน	3
บทที่ 2 หลักการและทฤษฎีที่เกี่ยวข้อง	5
2.1 บทนำ	5
2.2 มาสโปรแกรมเอเบิ้ล (Mask Programmable)	6
2.3 ฟิลด์โปรแกรมเอเบิ้ล (Field Programmable)	6
2.3.1 พีแอลดี (PLD : Programmable Logic Device)	7
2.3.2 พรอม (PROM : Programmable Read Only Memory)	7
2.3.3 พีเอแอล (PAL : Programmable Array Logic)	8
2.3.4 พีแอลเอ (PLA : Programmable Logic Array)	9
2.3.5 เอฟพีจีเอ (FPGA : Field Programmable Gate Array)	10
2.4 เทคโนโลยีของเอฟพีจีเอ	11
2.4.1 เอฟพีจีเอแบบต่างๆ	11
2.4.2 โครงสร้างภายใน	12
2.5 การออกแบบวงจรด้วยเอฟพีจีเอ	13
2.5.1 การออกแบบโดยใช้ภาษาอธิบายฮาร์ดแวร์(Hardware Description Language)	14
2.5.2 การสังเคราะห์วงจร (Logic Synthesis)	14

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5.3 การแบ่งวงจร (Partitioning) 15
2.5.4 การวางอุปกรณ์ (Placement) 15
2.5.5 การเชื่อมต่อสัญญาณ (Routing) 16
2.5.6 ความหน่วงทางด้านเวลา (Delay) 16
2.5.7 การโปรแกรมอุปกรณ์เอฟพีจีเอ (Configuration) 17
2.5.8 การจำลองการทำงานของวงจร (Simulation) 17
2.6 คีย์บอร์ด (Keyboard) 18
2.6.1 ลักษณะโดยทั่วไป 18
2.6.2 ลักษณะการเชื่อมต่อ 18
2.6.3 ลักษณะการส่งข้อมูลของคีย์บอร์ด 19
2.7 แอลอีดีดอทเมตริกซ์ 23
บทที่ 3 Master Flex-A01 มาสเตอร์เฟลก เอศูนย์หนึ่ง 24
3.1 บทนำ 24
3.2 โครงสร้างของบอร์ด Master Flex-A01 24
บทที่ 4 การออกแบบและการสร้างวงจร 32
4.1 ลักษณะโดยทั่วไป 32
4.2 หลักการสแกนแอลอีดีดอทเมตริกซ์ 34
4.3 รายการอุปกรณ์ 36
บทที่ 5 โปรแกรมควบคุมการทำงานและโฟล์วชาร์ต 37
5.1 โฟล์วชาร์ตการทำงานของวงจรทั้งหมด 37
5.2 ผลการจำลองการทำงานของโปรแกรม 56
5.3 วงจรที่ได้จากการสังเคราะห์จากโปรแกรม 57
บทที่ 6 บทสรุปและวิจารณ์ 63
6.1 ข้อจำกัดที่พบในการทำโครงงาน 63
6.2 บทสรุป 64
ภาคผนวก 65
กิตติกรรมประกาศ 120
บรรณานุกรม 121

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ

รูป	หน้า
รูปที่ 1.1 แสดงบล็อคไดอะแกรมของโครงงาน	3
รูปที่ 1.2 แสดง Characters Set	4
รูปที่ 2.1 แสดงแผนผังการแบ่งกลุ่มวงจรรวม ASIC	6
รูปที่ 2.2 แสดงวงจรพื้นฐานของอุปกรณ์พีแอลดีซึ่งอยู่ในรูปผลคูณร่วมบวก	7
รูปที่ 2.3 แสดงลักษณะของพรอมเมื่อเปรียบเทียบเป็นวงจรในรูปผลคูณร่วมบวก	8
รูปที่ 2.4 แสดงโครงสร้างภายในของพีเอแอล	9
รูปที่ 2.5 แสดงวงจรพื้นฐานภายในของพีแอลเอ	10
รูปที่ 2.6 แสดงโครงสร้างภายในของ FPGA ตระกูลต่าง ๆ	12
รูปที่ 2.7 แสดงขั้นตอนการออกแบบโดยใช้ FPGA	13
รูปที่ 2.8 แสดงคอนเนคเตอร์ที่ใช้กับคีย์บอร์ดแบบ PS/2	18
รูปที่ 2.9 แสดงลักษณะการส่งข้อมูลของคีย์บอร์ด	19
รูปที่ 2.10 แสดงโครงสร้างภายในของดอทเมตริกซ์	23
รูปที่ 4.1 แสดงบล็อกไดอะแกรมการทำงานภายในของไอซี 74LS154	32
รูปที่ 4.2 แสดงการสแกนแบบ คอลัมน์	34
รูปที่ 4.3 แสดงวงจรของบอร์ดแสดงผล	35
รูปที่ 5.1 แสดงโฟว์ชาร์ตของระบบ	37
รูปที่ 5.2 แสดงโฟว์ชาร์ตลำดับการทำงานของโปรแกรม	38
รูปที่ 5.3 แสดงโฟว์ชาร์ตของส่วน Input - Part	39
รูปที่ 5.4 แสดงโฟว์ชาร์ตของส่วน Memory – Part	40
รูปที่ 5.5 แสดงโฟว์ชาร์ตของส่วน Output – Part	41
รูปที่ 5.6 แสดงลำดับการทำงานของโปรแกรมทั้งหมด	42
รูปที่ 5.7 แสดงโฟว์ชาร์ตของโปรแกรม READ_PS2	43
รูปที่ 5.8 แสดงโฟว์ชาร์ตของโปรแกรม READ_KB	44
รูปที่ 5.9 แสดงโฟว์ชาร์ตของโปรแกรม SEPARATE	45
รูปที่ 5.10 แสดงโฟว์ชาร์ตของโปรแกรม DEL_DATA	46
รูปที่ 5.11 แสดงโฟว์ชาร์ตของโปรแกรม BLK_DATA	47

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 5.12 แสดงโฟว์ชาร์ตของโปรแกรม INS_DATA 48
รูปที่ 5.13 แสดงโฟว์ชาร์ตของโปรแกรม RAM 49
รูปที่ 5.14 แสดงโฟว์ชาร์ตของโปรแกรม MODE 50
รูปที่ 5.14.1 แสดงโฟว์ชาร์ตของโปรแกรม MODE (ต่อ) 51
รูปที่ 5.14.2 แสดงโฟว์ชาร์ตของโปรแกรม MODE (Sub Program B) 52
รูปที่ 5.15 แสดงโฟว์ชาร์ตของโปรแกรม RCV_DATA 53
รูปที่ 5.16 แสดงโฟว์ชาร์ตของโปรแกรม DECODE 54
รูปที่ 5.17 แสดงโฟว์ชาร์ตของโปรแกรม DISPLAY 55
รูปที่ 5.18 แสดงผลการจำลองการทำงานของส่วน Input – Part 56
รูปที่ 5.19 แสดงผลการจำลองการทำงานของส่วน Memory – Part 56
รูปที่ 5.20 แสดงผลการจำลองการทำงานของส่วน Output – Part 57
รูปที่ 5.21 แสดงวงจรที่ได้จากการสังเคราะห์ของส่วน Input – Part 58
รูปที่ 5.22 แสดงวงจรที่ได้จากการสังเคราะห์ของส่วน Memory – Part 59
รูปที่ 5.23 แสดงวงจรที่ได้จากการสังเคราะห์ของส่วน Output – Part 60
รูปที่ 5.24 แสดงวงจรที่ได้จากการสังเคราะห์ของส่วน Div 1000 61
รูปที่ 5.25 แสดงวงจรที่ได้จากการสังเคราะห์ของวงจรทั้งหมด 62

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

ตาราง	หน้า
ตารางที่ 2.1 แสดงคอนเน็คเตอร์แบบ ps/2	19
ตารางที่ 2.2 แสดงข้อมูลที่ส่งมาจากคีย์บอร์ด	20
ตารางที่ 2.3 แสดงค่าสแกนโค้ด (Keyboard Scan codes)	20
ตารางที่ 3.1 แสดงความสัมพันธ์ระหว่างพอร์ตขยายช่องสัญญาณ Ext A และ Ext B กับตำแหน่งขาของ EPF10K20TC144	25
ตารางที่ 3.2 ความสัมพันธ์ระหว่างวงจรสื่อสารข้อมูลแบบอนุกรม กับตำแหน่งขาของ EPF10K20TC144	27
ตารางที่ 3.3 ความสัมพันธ์ระหว่างคอนเน็กเตอร์สำหรับเชื่อมต่อ กับจอ VGA กับตำแหน่งขาของ EPF10K20TC144	27
ตารางที่ 3.4 ความสัมพันธ์ระหว่างคอนเน็กเตอร์แบบ PS/2 กับตำแหน่งขาของ EPF10K20TC144	27
ตารางที่ 3.5 ความสัมพันธ์ระหว่างคอนเน็กเตอร์แบบ Centronics Port กับตำแหน่งขาของ EPF10K20TC144	27
ตารางที่ 3.6 ความสัมพันธ์ระหว่างวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอล กับตำแหน่งขาของ EPF10K20TC144	28
ตารางที่ 3.7 ความสัมพันธ์ระหว่างวงจรแปลงสัญญาณดิจิตอลเป็นอนาลอก กับตำแหน่งขาของ EPF10K20TC144	28
ตารางที่ 3.8 ความสัมพันธ์ระหว่างวงจรหน่วยความจำ กับตำแหน่งขาของ EPF10K20TC144	29
ตารางที่ 3.9 ความสัมพันธ์ระหว่างสวิตซ์กดติด-ปล่อยดับ กับตำแหน่งขาของ EPF10K20TC144	29
ตารางที่ 3.10 ความสัมพันธ์ระหว่างดิพสวิตซ์ 8 บิต กับตำแหน่งขาของ EPF10K20TC144	30
ตารางที่ 3.11 ความสัมพันธ์ระหว่างไดโอดเปล่งแสง กับตำแหน่งขาของ EPF10K20TC144	30
ตารางที่ 3.12 ความสัมพันธ์ระหว่างวงจรแสดงผล 7-Segment 4 หลัก กับตำแหน่งขาของ EPF10K20TC144	30

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 3.13 ความสัมพันธ์ระหว่างโมดูลออสซิเลเตอร์
กับตำแหน่งขาของ EPF10K20TC144 31
ตารางที่ 4.1 แสดงค่าความจริงของไอซี 74LS154 33

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 บทนำ

ปัจจุบันการโฆษณา ประชาสัมพันธ์และการแสดงผลข้อมูลมีการพัฒนารูปแบบมากขึ้น ในการนำเสนอข้อมูลที่มีรูปแบบที่แตกต่างออกไปและทันสมัยจะช่วยดึงดูดความสนใจได้ดี ไม่ว่าจะเป็นการประชาสัมพันธ์ตามจุดต่างๆ การแสดงหมายเลขลำดับที่ เป็นต้น

การแสดงผลด้วยตัวอักษรหรือข้อความก็เป็นอีกรูปแบบหนึ่งที่นำมาใช้ เนื่องจากมีข้อดีและจุดเด่นหลายประการ เช่น สามารถเปลี่ยนแปลงข้อความได้ และมีการเคลื่อนไหวของข้อความ ทำให้มีความน่าสนใจ ไม่น่าเบื่อ สามารถนำไปประยุกต์ใช้งานอย่างอื่นได้หลายแบบ ในโครงงานนี้นำเสนอการออกแบบและสร้างบอร์ดแสดงผลตัวอักษรด้วยดอทเมตริกซ์(LED Dot Matrix) ควบคุมการแสดงผลด้วยเอฟพีจีเอ (FPGA : Field Programmable Gate Array) เนื่องจาก FPGA มีข้อดีหลายประการ เช่น มีจำนวนลอจิกเกตภายในมากพอสำหรับการออกแบบวงจรและใช้งาน ทั้งยังสามารถแก้ไขวงจรหรือการทำงานภายในได้ซึ่งเป็นประโยชน์อย่างมาก

1.2 จุดประสงค์ของโครงงาน

1.) เพื่อศึกษาและนำเอา FPGA ไปประยุกต์ใช้งานทางด้านอิเล็กทรอนิกส์

2.) ศึกษาหลักการเขียนภาษาวีเอชดีแอลเพื่อนำไปใช้ในการเขียนโปรแกรมอธิบายพฤติกรรมของฮาร์ดแวร์

3.) เพื่อศึกษาเรียนรู้การใช้ซอฟท์แวร์ Max+plusII ในการออกแบบและจำลองการทำงานของวงจรดิจิตอล

4.) สามารถนำความรู้ที่ได้ไปใช้ในการออกแบบและสร้างวงจรแสดงผลตัวอักษรวิ่งโดยใช้ FPGA

1.3 ขอบเขตของโครงงาน

1.) ศึกษาหลักการของ FPGA และแนวทางการนำไปประยุกต์ใช้งาน

2.) ศึกษาวิธีและหลักการออกแบบวงจรด้วยภาษาวีเอชดีแอล

3.) ศึกษาหลักการทำงานของคีย์บอร์ดแบบ PS/2

4.) ออกแบบและเขียนโปรแกรมควบคุมการทำงานของบอร์ดแสดงผลด้วยภาษาวีเอชดีแอล (VHDL)

5.) ออกแบบและสร้างวงจรบอร์ดแสดงผลแบบดอทเมตริกซ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.) ทดสอบวงจรที่สร้างขึ้นและจำลองการทำงานของโปรแกรมควบคุมการทำงาน

7.) วิเคราะห์หาข้อผิดพลาดและทำการแก้ไขวงจรและโปรแกรมควบคุมการทำงาน

8.) วิเคราะห์ผลการทดลอง

9.) สรุปผลและทำการเขียนรายงาน

1.4 ลักษณะโดยทั่วไปของบอร์ดแสดงผล

ในการออกแบบวงจรจะแบ่งได้เป็นสองส่วนคือ ส่วนควบคุมและส่วนแสดงผล ซึ่งส่วนควบคุมจะใช้เอฟพีจีเอและเขียนโปรแกรมควบคุมด้วยภาษาวีเอชดีแอล เนื่องจากมีความยืดหยุ่นและสามารถแก้ไขวงจรได้ง่าย นอกจากนี้ยังใช้ได้กับเทคโนโลยีเอฟพีจีเอของทุกบริษัท สำหรับโครงงานนี้ใช้ซอฟท์แวร์ Max+plusII ของบริษัทอัลเทรา(Altera) ในการออกแบบวงจร (Design) และการจำลองการทำงาน (Simulation) และใช้ชิฟเอฟพีจีเอตระกูล FLEX10K เบอร์ EPF10K20TC144-4 ควบคุมการแสดงผลตัวอักษรที่ป้อนเข้าไป โดยใช้คีย์บอร์ดคอมพิวเตอร์แบบ PS/2 ในการป้อน ข้อมูลตัวอักษรที่ต้องการ เนื่องจากหาได้ง่ายและมีจำนวนตัวอักษรที่ต้องการเพียงพอ

1.4.1 คุณสมบัติของบอร์ดแสดงผล

- การแสดงผลตัวอักษรใช้ LED Dot Matrix ขนาด 8x8 จำนวน 10 ชุดมาเรียงกัน เพื่อให้สามารถแสดงตัวอักษรได้อย่างเพียงพอและชัดเจน
- ภาษาที่แสดงผลเป็นภาษาอังกฤษตัวพิมพ์ใหญ่ A-Z
- ในการป้อนข้อมูลที่จะแสดงผลใช้ คีย์บอร์ดคอมพิวเตอร์แบบ PS/2 เนื่องจากหาได้ง่าย
- ใช้ชิฟเอฟพีจีเอตระกูล flex10k เบอร์ EPF10K2OTC144-4
- วงจรที่ใช้ออกแบบด้วย ภาษาวีเอชดีแอล (VHDL)
- รูปแบบการแสดงตัวอักษรมี 2 แบบคือหยุดนิ่ง และ วิ่งจากขวาไปซ้าย
- การป้อนตัวอักษรจะป้อนจากขวาไปซ้ายโดยที่

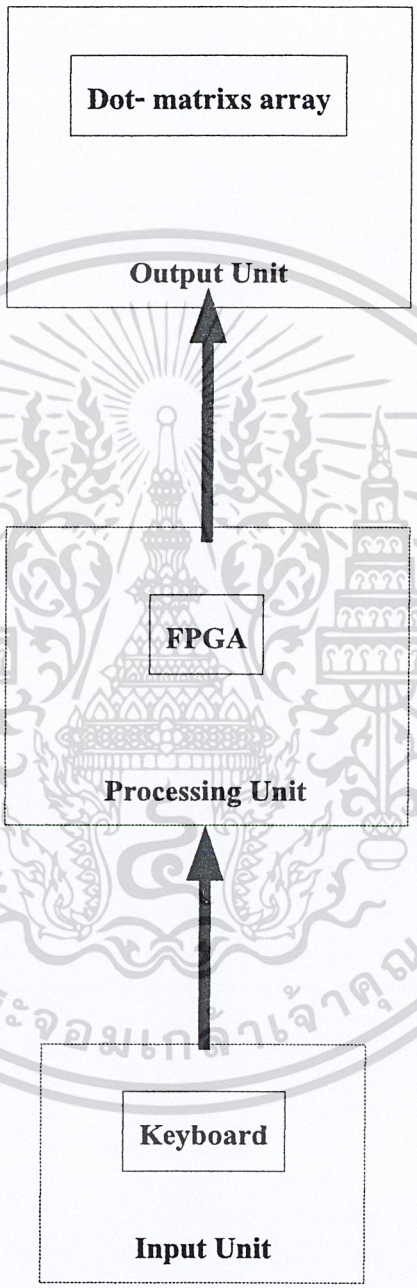
  -- กดแป้น Enter ตัวอักษรจะวิ่งจากขวาไปซ้าย

  --กดแป้น Esc ตัวอักษรหยุดวิ่งและกลับมาอยู่ในลักษณะเหมือนตอนก่อนวิ่ง

  --กดแป้น BackSpace จะเป็นการลบตัวอักษรตัวขวาสุดได้ 1 ตัวอักษรต่อการกด 1 ครั้ง
- จำนวนบรรทัด 1 บรรทัด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.4.2 บล็อคไดอะแกรมของโครงงาน แสดงได้ดังรูปที่ 1.1



รูปที่ 1.1 แสดง Block Diagram ของโครงงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 1.2 แสดง Characters Set

ส่วนรายละเอียดในวิธีการออกแบบและการสร้างวงจร ตลอดจนทฤษฎีพื้นฐานที่เกี่ยวข้องจะได้กล่าวถึงในบทต่อๆไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
หลักการและทฤษฎีที่เกี่ยวข้อง

2.1 บทนำ

ความก้าวหน้าของเทคโนโลยีอิเล็กทรอนิกส์ในปัจจุบัน ทำให้เกิดการพัฒนาความสามารถของอุปกรณ์อิเล็กทรอนิกส์ต่างๆมากมาย อุปกรณ์อิเล็กทรอนิกส์มีขนาดลดลง ใช้งานง่ายขึ้นและราคาถูก นอกจากนี้ก็ได้มีการเพิ่มประสิทธิภาพและระดับความน่าเชื่อถือได้ของวงจรรวมสูงขึ้น ซึ่งการพัฒนาวงจรรวมทางอิเล็กทรอนิกส์จะแบ่งได้เป็น 2 ส่วน คือ

- ส่วนที่ 1 การพัฒนาวงจรทางด้านอนาลอก เมื่อเราออกแบบวงจรแล้วจะต้องมีการนำไปผ่านกระบวนสร้าง มีค่าใช้จ่ายที่สูงและระยะเวลานาน
- ส่วนที่ 2 การพัฒนาวงจรทางด้านดิจิตอล เมื่อเราออกแบบวงจรแล้วสามารถนำไปโปรแกรมลงชิฟไอซีได้เลย วิธีนี้จะช่วยลดค่าใช้จ่ายและระยะเวลาสั้นลงได้

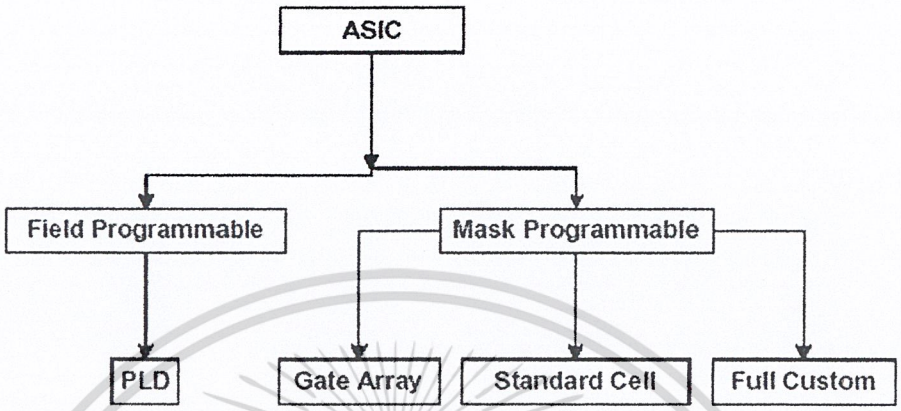
ในการพัฒนาวงจรดิจิตอลแบบเก่าโดยการใช้ดิจิตอลไอซีมาต่อวงจรมาต่อวงจร จะมีความยุ่งยากมากทั้งขนาดและการทดสอบความถูกต้องในการทำงาน ทำให้เกิดช่องว่างระหว่างไอซีมาตรฐานและวงจรรวมมากยิ่งขึ้น ในการพัฒนาวงจรรวมทางดิจิตอลได้เพิ่มความหนาแน่นและจำนวนฟังก์ชันลอจิก (Function Logic) ที่เหมาะสม นักออกแบบอุปกรณ์ทางด้านดิจิตอลได้พิจารณาถึงการผลิตให้มีปริมาณมากๆ และในการผลิตวงจรรวมเฉพาะงาน (ASIC : Application Specific Integrated Circuit) จะแบ่งตามการสร้างออกเป็น 2 กลุ่ม คือ

1.Mask Programmable

2.Field Programmable

ดังแสดงได้ดังรูปที่ 2.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.1 แสดงแผนผังการแบ่งกลุ่มวงจรรวม ASIC

2.2 Mask Programmable

การใช้งานวงจรรวมเฉพาะงาน ASIC ในเชิงพาณิชย์ จำเป็นต้องใช้แบบ Mask programmable เนื่องจากต้นทุนต่อเนื่องต่อหนึ่งตัวจะต่ำกว่าวงจรแบบ field programmable ในกรณีที่ปริมาณการผลิตสูงนับพันตัวขึ้นไป วงจรรวมเฉพาะงานประเภทนี้ หลังจากผู้ใช้ออกแบบวงจรและตรวจสอบการทำงานจนเป็นที่น่าพอใจแล้ว ต้องส่งให้ผู้ผลิตทำการเจือสาร ไม่สามารถโปรแกรมได้ด้วยตนเอง จึงมีระยะเวลาในการผลิตยาวนานและค่าใช้จ่ายเบื้องต้นสูง วงจรรวมเฉพาะงานแบบ Mask programmable ในปัจจุบันได้แก่ เกตอาเรย์ (Gate Array) เซลล์มาตรฐาน และฟูลคัสตัม(Full Custom)

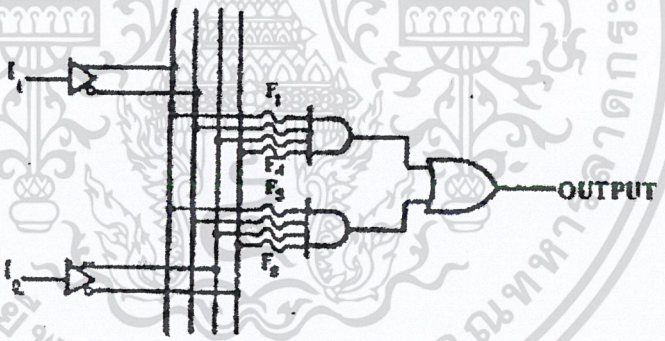
2.3 Field Programmable

อุปกรณ์วงจรรวมเฉพาะงานแบบ field programmable มีอยู่มากมายหลายชนิด แต่มีลักษณะการสร้างหรือการกำหนดการทำงานของวงจรที่เหมือนกัน กล่าวคือ ผู้ใช้สามารถออกแบบและสร้างวงจรที่ต้องการใช้ลงในตัวอุปกรณ์ได้เองโดยไม่ต้องไปที่โรงงานเพื่อผลิต ซึ่งในปัจจุบันนี้มีเครื่องมือที่ใช้ช่วยในการออกแบบและสร้างวงจรร่วมกับไมโครคอมพิวเตอร์ที่มีความสามารถสูงในการพัฒนาตั้งแต่ขั้นการออกแบบ การจำลองการทำงาน จนถึงการสร้างวงจรลงในอุปกรณ์ รวมทั้งอุปกรณ์ field programmable เหล่านี้สามารถหาซื้อได้ง่าย ทำให้การสร้างวงจรอิเล็กทรอนิกส์หันมาใช้อุปกรณ์จำพวกนี้ประกอบในวงจรมากขึ้นแทนการใช้อุปกรณ์ย่อยๆแยกชิ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.1 พีแอลดี (PLD : Programmable Logic Device)

ภายในอุปกรณ์พีแอลดีถูกเตรียมเป็นวงจรพื้นฐานทางด้านลอจิกต่อกันอยู่เป็นกลุ่ม มีทั้งวงจรคอมบิเนชัน (Combination) และซีเควนเชียล (Sequential) ซึ่งมีส่วนประกอบเป็นวงจรภายในเทคโนโลยีของวงจรที่ใช้สร้างพีแอลดีมีทั้งทีทีแอล (TTL) อีซีแอล (ECL) และซีมอส (CMOS) ตามความเหมาะสมของแต่ละระบบ อุปกรณ์พีแอลดีทุกชนิดมีหลักการพื้นฐานของวงจรภายในที่เหมือนกันโดยมีวงจรคอมบิเนชันที่ให้ผลเป็นผลคูณร่วมบวก (Sum of Product) ประกอบไปด้วยชุดของแอนด์เกตที่ต่อร่วมกับออร์เกต การโปรแกรมคือการเลือกว่าจะให้มีการต่ออินพุทภายในของแอนด์เกตกับสัญญาณอินพุทใดบ้าง ซึ่งมีทั้งจากภายนอกและสัญญาณป้อนกลับจากเอาท์พุทภายในเอง การติดต่ออินพุทของออร์เกตกับเอาท์พุทของแอนด์เกตตัวต่างๆ วิธีการเลือกหรือการโปรแกรมทางกายภาพ อินพุทต่างๆ ของอุปกรณ์ทุกตัวจะถูกต่อผ่านฟิวส์เข้ากับแหล่งสัญญาณ ซึ่งถ้าไม่ต้องการใช้สัญญาณใดจะตัดฟิวส์ ทำให้สามารถโปรแกรมได้ครั้งเดียว อุปกรณ์พีแอลดีบางชนิดใช้มอสทรานซิสเตอร์แทนฟิวส์ ทำให้สามารถโปรแกรมโดยใช้กระแสไฟฟ้า สามารถลบและโปรแกรมเข้าไปใหม่ได้อีก



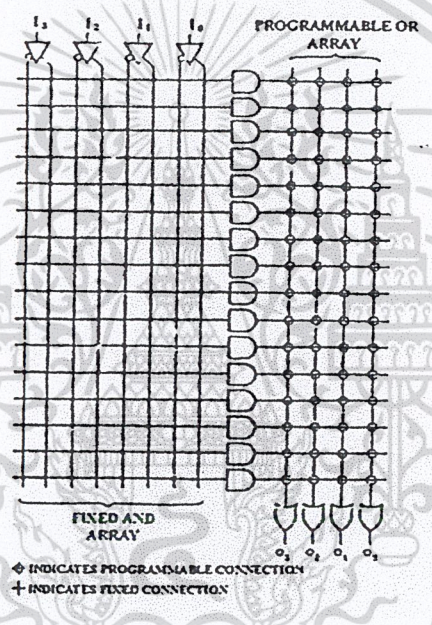
รูปที่ 2.2 แสดงวงจรพื้นฐานของอุปกรณ์พีแอลดีซึ่งอยู่ในรูปผลคูณร่วมบวก

2.3.2 พรอม (PROM : Programmable Read Only Memory)

พรอมคือหน่วยความจำรอม (ROM) โปรแกรมได้ นับว่าเป็นอุปกรณ์พีแอลดีชนิดหนึ่งซึ่งวงจรภายในของพรอมเสมือนกับประกอบด้วยแถวลำดับของแอนด์และออร์เกต (AND/Or Array) ผลเอาท์พุทที่ขาดาต้าเอาท์พุทสามารถแสดงในสมการของฟังก์ชันผลคูลร่วมบวกของสัญญาณอินพุทที่ขาแอดเดรส ในรูปที่ 2.3 แสดงถึงลักษณะการต่อเป็นแถวลำดับของแอนด์เกตและออร์เกตของพรอมขนาด 16 x 4 บิต วงจรทางด้านซ้ายบนสุดเป็นแอนด์เกตที่ให้ผลเป็นผลคูณ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(Product) ของกรณีอินพุทเป็น 0000 แอนด์เกตที่อยู่ถัดลงมาเป็นเป็นผลคูณของกรณีที่อินพุทเป็น 0001, 0010, ... จนถึงตัวล่างสุดคือผลคูณในกรณีที่อินพุทเป็น 1111 ที่เอาท์พุแต่ละบิตของหน่วยความจำ สามารถเลือกได้ว่าจะให้เป็น 1 ในกรณีที่อินพุทจากแอดเดรสเป็นอย่างไรบ้างเหมือนกัน เป็นการนำเอาท์พุทจากผลคูณที่ต้องการให้เอาท์พุทแต่ละบิตเป็น 1 ไปออร์กัน จึงเปรียบเหมือนกับว่าในพรอมมีจำนวนแอนด์เกตเท่ากับจำนวนตำแหน่งความจำและมีออร์เกตจำนวนเดท่ากับจำนวนบิตของสัญญาณข้อมูลออก (Data Output) อินพุทออร์เกตทุกตัวสามารถต่อเข้ากับแอนด์เกตตัวใดก็ได้ทุกตัว ซึ่งอาจเรียกได้ว่าพีแอลดีแบบ fixed AND / programmable OR

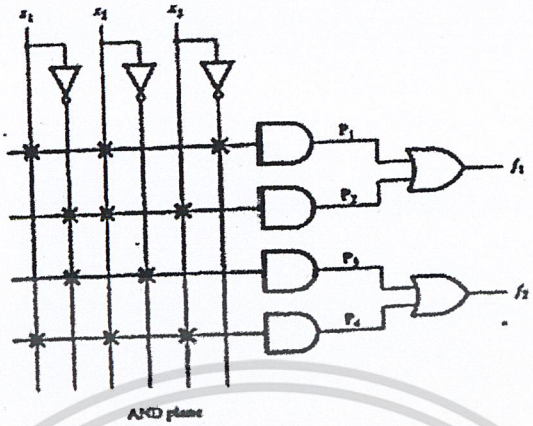


รูปที่ 2.3 แสดงลักษณะของพรอมเมื่อเปรียบเทียบเป็นวงจรในรูปผลคูณร่วมบวก

2.3.3 พีเอแอล (PAL : Programmable Array Logic)

ใน ค.ศ. 1970 บริษัทเอ็มเอ็มไอ (MMI : Monolithic Memory) ได้พัฒนาอุปกรณ์พีเอแอลเป็นพีเอแอลชนิดใหม่โดยใช้เทคโนโลยีแบบแอลเอสไอ (LSI : Large Scale Integration) สามารถโปรแกรมเลือกวงจรภายในโดยใช้ฟิวส์ที่เชื่อมต่ออยู่ระหว่างสัญญาณอินพุทภายนอก และการป้อนกลับจากภายในกับแอนด์เกตที่ต่อเป็นฟังก์ชันผลคูณอยู่ในตัววงจรรวม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

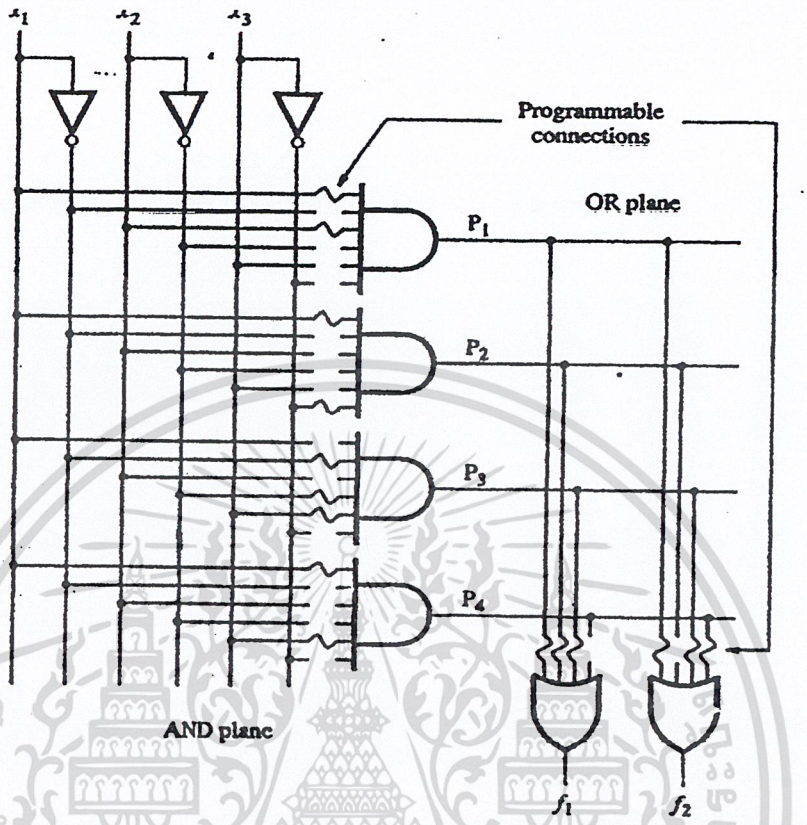


รูปที่ 2.4 แสดงโครงสร้างภายในของพีเอแอล

2.3.4 พีแอลเอ (PLA : Programmable Logic Array)

อุปกรณ์ที่สามารถโปรแกรมได้แบบพีแอลเอเกิดขึ้นเมื่อ ค.ศ. 1975 โดยบริษัทซิกเนทริกส์ (Signetics) ได้ผลิตและนำเสนออุปกรณ์ใช้ชื่อว่า เอฟพีแอลเอ (FPLA : Field Programmable Logic Array) สามารถโปรแกรมการต่อลอจิกทั้งทางด้านแอนด์เกตและออร์เกตได้ และยังเลือกเอาท์พุทเป็น Active High หรือ Active Low โดยต่อผ่านเอ็กคลูซีฟออร์เกต ให้ทำหน้าที่เป็นนอนอินเวอร์เตอร์หรือเป็นอินเวอร์เตอร์แล้วแต่ภายในของพีแอลเอ ต่อมาได้มีการสร้างเอฟพีแอลเอใหม่ที่มีรีจิสเตอร์ต่ออยู่ภายในวงจรเพิ่มขึ้น รวมทั้งสามารถเลือกสัญญาณอินพุทที่มาจากการป้อนกลับจากรีจิสเตอร์ได้ด้วย ทำให้สามารถใช้อุปกรณ์เอฟพีแอลเอแบบใหม่นี้สร้างวงจรสเตทแมชชีนได้ อุปกรณ์ใหม่ที่มีรีจิสเตอร์อยู่ด้วยนี้ถูกตั้งชื่อใหม่เป็นเอฟพีแอลเอส (FPLS : Field Programmable Logic Sepucncer) มีทั้งที่เป็น TTL และ CMOS ดังรูปที่ 2.5 ซึ่งแสดงวงจรพื้นฐานภายในของพีแอลเอ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.5 แสดงวงจรพื้นฐานภายในของพีแอลเอ

2.3.5 เอฟพีจีเอ (FPGA : Field Programable Gate Array)

FPGA เป็นชิพในลักษณะของ Programmable Device ที่เราสามารถโปรแกรมตัวมันให้สามารถเป็นวงจรดิจิตอลใดๆก็ได้ โครงสร้างภายในประกอบด้วย Array ของลอจิกเกตต่างๆ มากมาย ซึ่งในปัจจุบันความจุเกทภายในตัวชิพ FPGA ได้เพิ่มขึ้นในระดับไม่กี่พันตัวจนถึงล้านตัว ดังนั้นจึงรองรับวงจรดิจิตอลที่มีความสลับซับซ้อนเป็นอย่างมาก และรวมถึงความเร็วในการทำงานที่ได้เปรียบอย่างมากมายเมื่อเทียบกับไมโครโปรเซสเซอร์ นอกจากนี้ในด้านการออกแบบพัฒนาและการทดสอบก็ทำได้ง่าย เนื่องจากเราสามารถโปรแกรมได้เองและสามารถโปรแกรมหลายๆครั้ง จึงไม่มีความเสี่ยงใดๆทั้งสิ้น และในปัจจุบันการออกแบบที่เป็น FPGA Base IC Design นั้นก็เป็นที่นิยมทำกันและมีแนวโน้มที่จะนำมาใช้งานมากขึ้นเรื่อยๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4 เทคโนโลยีของเอฟพีจีเอ

2.4.1 เอฟพีจีเอแบบต่างๆ

เนื่องจาก FPGA เป็นลักษณะของชิพที่สามารถโปรแกรมได้ นั่นคือสามารถกำหนดจุดเชื่อมต่อต่างๆ ภายในได้ เพื่อประกอบเป็นลักษณะของวงจรตามที่เราต้องการ และสามารถแบ่งลักษณะของจุดเชื่อมต่อต่างๆ ได้ดังนี้

2.4.1.1 Physical Changing

1.) Fused สามารถโปรแกรมได้ครั้งเดียว หลังจากโปรแกรมจุดเชื่อมต่อขาดจากกัน

2.) Anty Fuse สามารถโปรแกรมได้ครั้งเดียว หลังจากโปรแกรมจุดเชื่อมต่อจะเชื่อมถึงกัน

2.4.1.2 Memory Base

1.) EEPROM – Base FPGA

มักเรียก FPGA ประเภทนี้ว่า CPLD จะใช้เทคโนโลยีเหมือนกับ EEPROM ในการโปรแกรม ซึ่งจะทำให้มีความจุของเกตต่ำ โดยทั่วๆไปจะน้อยกว่า 20,000 เกต แต่ข้อดีของ EEPROM – Base FPGA คือสามารถเก็บข้อมูลที่โปรแกรมลงไปได้โดยไม่จำเป็นต้องมีไฟเลี้ยง และในการโปรแกรม จะใช้ทรานซิสเตอร์ 1 ตัวต่อ 1 บิต สามารถโปรแกรมได้ประมาณ 10,000 ครั้ง มักจะมีการจัดสถาปัตยกรรมในรูปแบบของอาร์เรย์ ใช้ AND – OR Plane ในการทำลอจิกฟังก์ชัน

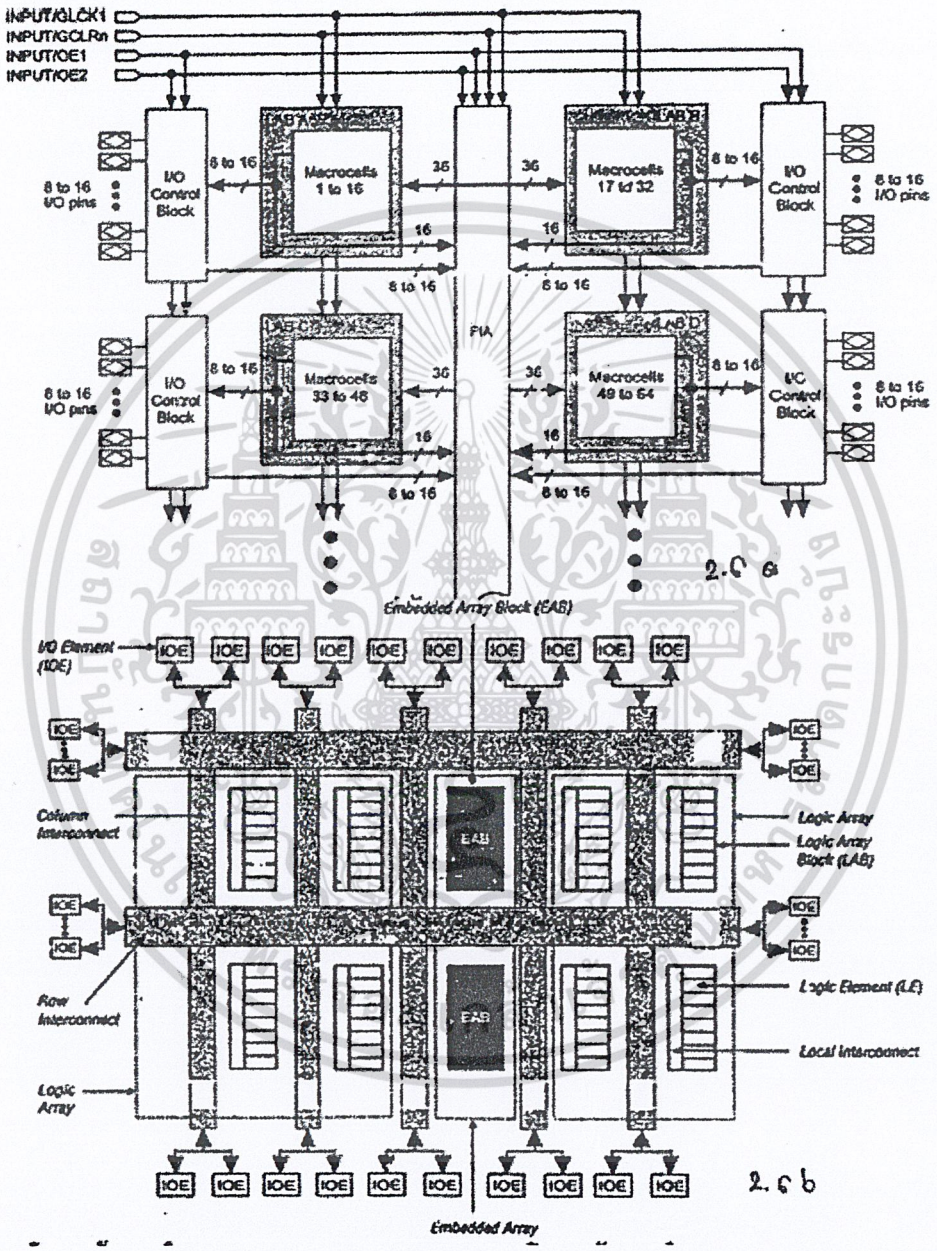
2.) SRAM – Base FPGA

จะใช้เทคโนโลยีเหมือน SRAM ในการโปรแกรม ซึ่งจะสามารถทำให้โปรแกรมซ้ำได้ไม่จำกัดจำนวนครั้ง มีความจุของเกตปานกลางถึงสูงมาก ประมาณ 10,000 – 1,000,000 เกต จะใช้ Look –Up Table ในการทำลอจิกฟังก์ชัน และจะมีการจัดทรัพยากรภายในโครงสร้างแบบอาร์เรย์ ข้อดีของ SRAM – Base FPGA คือจะใช้เวลาในการโปรแกรมน้อย ในระดับ 1/1000 วินาที การโปรแกรมจะทำได้ง่ายเทียบเท่ากับการเขียน SRAM ทั่วไป และไม่จำกัดจำนวนครั้ง ในกระบวนการผลิตจะทำได้ง่ายและเหมาะสมสำหรับการออกแบบวงจรที่มีความสลับซับซ้อน ข้อเสียก็คือไม่สามารถเก็บโปรแกรมในภาวะที่ไม่มีไฟเลี้ยงได้ มักจะใช้ FPGA ชนิดนี้ควบคู่กับ ROM เพื่อเก็บโปรแกรมและจะโหลดโปรแกรมเข้าในตัวชิพเมื่อเริ่มต้นใช้งาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.2 โครงสร้างภายใน

ลักษณะโครงสร้างภายในเป็นอาร์เรย์ลอจิกบ็อกซ์ที่สามารถทำการโปรแกรมได้ดังแสดงใน รูปที่ 2.6a และรูปที่ 2.6b



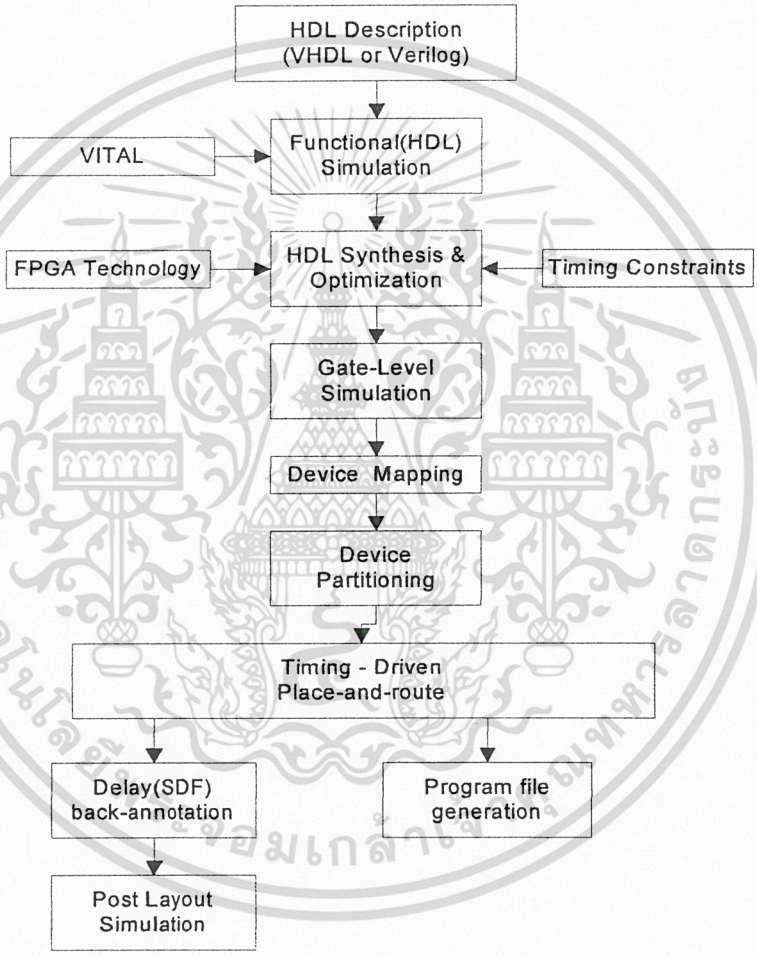
รูปที่ 2.6 แสดงโครงสร้างภายในของ FPGA ตระกูลต่างๆ

2.6a) โครงสร้างภายในของตระกูล MAX7000S 2.6b) โครงสร้างภายในของตระกูล FLEX10k

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5 การออกแบบวงจรด้วย FPGA

อุปกรณ์ FPGA สามารถนำไปประยุกต์ใช้งานได้หลากหลาย เช่น ทำการประมวลผลสัญญาณเชิงเลข (DSP : Digital Signal Processing) หรือไมโครคอนโทรลเลอร์ ในการออกแบบวงจรด้วย FPGA ก็จะคล้ายคลึงกับการออกแบบ ASIC ซึ่งมีขั้นตอนต่างๆดังรูปที่ 2.7



รูปที่ 2.7 แสดงขั้นตอนการออกแบบโดยใช้ FPGA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5.1 ออกแบบโดยใช้ภาษาอธิบายฮาร์ดแวร์ (Hardware Description Language)

ในการออกแบบวงจรดิจิตอลนั้น ทำได้ทั้งโดยการวาดวงจร (Schematic Drawing) หรือการใช้ภาษาอธิบายฮาร์ดแวร์ ในขั้นตอนนี้เป็นขั้นตอนที่ไม่แตกต่างกันนักระหว่างการออกแบบด้วย FPGA และASIC ในกรณีที่ใช้ภาษาอธิบายฮาร์ดแวร์ แต่ในกรณีที่ออกแบบโดยวิธีการวาดวงจรจะต่างกัน ผู้ทำการออกแบบด้วยวิธีนี้ต้องคำนึงถึงเทคโนโลยีที่จะใช้ซึ่งแต่ละเทคโนโลยีก็มีความแตกต่างกันไป จะเห็นได้ว่าการออกแบบโดยใช้ภาษาอธิบายฮาร์ดแวร์ทำได้สะดวกกว่าเพราะการทำวิธีนี้ผู้ออกแบบไม่ต้องมาคำนึงถึงเทคโนโลยีที่จะใช้ และที่สำคัญสามารถแก้ไขโมเดลหรือเปลี่ยนเทคโนโลยีได้สะดวกกว่าเพราะไม่ต้องวาดวงจรใหม่ นั่นคือการออกแบบโดยใช้ภาษาอธิบายฮาร์ดแวร์จะทำให้โมเดลที่ได้ไม่ขึ้นกับเทคโนโลยี สำหรับภาษาที่ใช้ก็มี VHDL หรือ Verilog ในการเขียนโค้ด (code) สิ่งที่ต้องคำนึงถึงก็คือจะต้องเขียนอย่างไรจึงจะสามารถสังเคราะห์เป็นวงจรได้และให้โมเดลที่มีคุณสมบัติถูกต้องตามกำหนด เพราะลักษณะการเขียนโค้ดจะมีผลโดยตรงกับโมเดลที่ได้ เนื่องจากในการสังเคราะห์วงจรนั้น ซอฟท์แวร์สังเคราะห์วงจร (Synthesis Tools) จะสังเคราะห์วงจรตามโค้ด เช่น ในการเขียนโค้ด VHDL อธิบายการทำงานของโมเดลวงจรอันเดียวกัน แต่เขียนโค้ดในลักษณะที่ต่างกัน เมื่อสังเคราะห์จะได้วงจรต่างกันและจากวงจรที่ต่างกัน เมื่อนำไปทำต้นแบบด้วย FPGA หรือการทำ ASIC แล้วจะได้ไอซีที่มีคุณสมบัติต่างกันทั้งในด้านของขนาดหรือความเร็ว (area and time) ดังนั้นในการออกแบบขั้นตอนนี้ผู้ออกแบบต้องระมัดระวังเป็นพิเศษ ส่วนการที่จะเขียนโค้ดในลักษณะใดเพื่อให้ได้ผลลัพธ์ที่ดีที่สุดนั้นก็ขึ้นอยู่กับประสบการณ์ของผู้ออกแบบ

2.5.2 การสังเคราะห์วงจร (Logic Synthesis)

ในขั้นตอนนี้จะใช้ซอฟท์แวร์สังเคราะห์วงจร (Synthesis Tools) ทำการสังเคราะห์โค้ด VHDL หรือ Verilog เพื่อให้ได้เป็นวงจรขึ้นมา ซึ่งทำได้โดยใช้ซอฟท์แวร์อีกเช่นกัน แต่ต้องตรวจสอบด้วยว่าซอฟท์แวร์นั้นๆ สนับสนุนเทคโนโลยี FPGA (FPGA Library) ที่ต้องการใช้หรือไม่ ตัวอย่าง FPGA ที่มีการใช้งานแพร่หลายเช่น FPGA ของบริษัท Xilinx และบริษัท Altera ในการทำ FPGA นั้นมีซอฟท์แวร์หลายตัวที่สามารถใช้ได้ เช่น Maxplus2, Synopsys, Exemplar ฯลฯ ในขั้นตอนนี้ ซอฟท์แวร์สังเคราะห์วงจรจะแปลงโค้ด VHDL และทำออพติไมซ์ (Optimization) เพื่อให้ได้วงจรตามเทคโนโลยีที่เลือกใช้ ในการสังเคราะห์วงจรเพื่อทำ FPGA นั้น วงจรระดับเกต (Gate-Level) ไม่เหมาะสมกับโครงสร้างที่มีอยู่ในอุปกรณ์ FPGA ดังนั้นในการออพติไมซ์ ซอฟท์แวร์สังเคราะห์วงจรจะต้องทำการออพติไมซ์ให้ได้เป็นวงจรที่ประกอบด้วยกลุ่มของลอจิกที่เหมาะสมกับอุปกรณ์ FPGA นั้นๆ จึงจะทำให้ผลลัพธ์ที่ได้มีประสิทธิภาพ และในขั้นตอนการสังเคราะห์วงจรนี้ผู้ออกแบบสามารถกำหนดข้อบังคับสำหรับโมเดลได้ เช่น ข้อบังคับในเรื่องของ

เอกสารนี้มีลิขสิทธิ์ ห้ามนำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เวลา (Timing Constraints) หรือข้อบังคับในเรื่องของพื้นที่ (Area) หรือกำหนดชนิดและตำแหน่งของ I/O ซึ่งข้อบังคับเหล่านี้จะถูกนำไปใช้ในขั้นตอนออพติไมซ์เพื่อให้วงจรเป็นไปตามที่กำหนด ส่วนสำคัญในการออพติไมซ์คือการเทียบ (mapping) โมเดลให้เข้ากับเทคโนโลยีที่ใช้เพื่อให้ได้วงจรที่เหมาะสมกับโครงสร้างสถาปัตยกรรมภายในอุปกรณ์ FPGA ในกรณีของ Xilinx FPGA จะเทียบโดยใช้วิธี LUT (Look Up Table)

เมื่อทำการสังเคราะห์วงจรเสร็จแล้ว ซอฟท์แวร์สังเคราะห์วงจรก็จะมีการรายงานผลว่าโมเดลที่ออกแบบไปนั้นเป็นอย่างไร เช่น มีความหน่วง (Delay) เท่าไหร่ ใช้ทรัพยากรต่างๆใน FPGA อะไรบ้าง เมื่อมาถึงขั้นนี้ผู้ออกแบบก็จะทราบว่าโมเดลเป็นไปตามข้อบังคับหรือไม่ ถ้าไม่ก็ให้ทำการสังเคราะห์ใหม่จนกว่าจะเป็นไปตามที่กำหนด ในขั้นตอนการสังเคราะห์นี้มีเทคนิคต่างๆที่ผู้ออกแบบนำมาใช้เพื่อให้โมเดลเป็นไปตามข้อบังคับที่กำหนด

2.5.3 การแบ่งวงจร (Partitioning)

ขั้นตอนนี้เป็นการแบ่งวงจรที่ได้จากการสังเคราะห์ เป็นส่วนย่อยๆ สำหรับลงใน CLBs, IOBs หรือองค์ประกอบอื่นๆภายในอุปกรณ์ FPGA สำหรับเกณฑ์ที่ใช้ในการแบ่งคือ ให้แต่ละส่วนที่จะแยกออกจากกันมีจำนวนสัญญาณที่เชื่อมต่อกันระหว่างกันน้อยที่สุดเท่าที่จะทำได้ เพื่อช่วยลดความหนาแน่นในตอนทำการเชื่อมต่อสัญญาณ (Routing) ในขั้นตอนนี้จะใช้ซอฟท์แวร์ทำโดยซอฟท์แวร์จะเทียบส่วนประกอบของวงจร เช่น เกท, ฟลิบฟลอบ ลงในทรัพยากรต่างๆ ที่มีอยู่ภายในอุปกรณ์ FPGA (CLBs, IOBs, BUFT และ edge decoder)

หลังจากทำขั้นตอนนี้เสร็จแล้ว ผู้ออกแบบสามารถที่จะทราบว่าวงจรใช้จำนวนทรัพยากรภายในอุปกรณ์ FPGA ไปเท่าไหร่ ส่วนข้อมูลทางเวลานั้นผู้ออกแบบจะทราบเฉพาะความหน่วงภายในแต่ละส่วนเท่านั้น หรือที่เรียกว่าความหน่วงลอจิก (logic delay) ส่วนซอฟท์แวร์ที่ใช้ในขั้นตอนนี้ เช่น M1 ของ Xilinx ซึ่งจะรวมเอาซอฟท์แวร์ย่อยอื่นๆ อีก เพื่อให้การทำ PPR (Partition, Placement & Routing) เป็นไปอย่างต่อเนื่อง

2.5.4 การวางอุปกรณ์ (Placement)

ขั้นตอนนี้เป็นการเลือกทำเลที่ตั้งของแต่ละส่วนของวงจรที่ผ่านการแบ่งวงจรมาแล้วว่าควรจะอยู่ ณ ตำแหน่งไหนในอุปกรณ์ FPGA เพื่อให้ได้ผลลัพธ์ที่ดีที่สุด เช่น วงจรส่วนไหนควรอยู่ใกล้กันเพื่อจะได้ค้นหาเส้นทาง (route) ได้ง่าย หรือช่วยลดความหน่วง จะเห็นได้ว่าตำแหน่งภายในอุปกรณ์ FPGA นั้นมีความสำคัญ เพราะถ้าจัดวางวงจรลงในตำแหน่งที่ไม่เหมาะสมแล้วจะทำให้ความหน่วงเพิ่มขึ้น หรือ Router ทำการค้นหาเส้นทางสัญญาณได้ไม่หมด

การวางอุปกรณ์ที่ดี ควรวางส่วนต่างให้อยู่ใกล้กันโดยเฉพาะส่วนที่มีการเชื่อมต่อสัญญาณด้วยกัน นอกจากนั้นการกำหนดตำแหน่งขา I/O (I/O pin) ตามตำแหน่งขา I/O ของ FPGA บน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แผ่น PCB ก็จะมีผลโดยตรงเลยคือซอฟท์แวร์ จะวาง I/O ลงในตำแหน่งที่ผู้ออกแบบกำหนด ซึ่งบางครั้งตำแหน่งที่กำหนดไปไม่เหมาะสม ดังนั้นการกำหนดขา I/O ควรกำหนดตำแหน่งให้เหมาะสม หรือไม่ก็ให้ซอฟท์แวร์จัดการเอง อาจจะให้ซอฟท์แวร์ทำซ้ำหลายๆครั้งเพื่อหาครั้งที่ดีที่สุด สำหรับเกณฑ์ที่ใช้ในการตัดสินคือ ความหน่วงที่ได้หลังจากทำการค้นหาเส้นทางแล้ว หรือที่เรียกว่า routing delay

2.5.5 การเชื่อมต่อสัญญาณ (Routing)

ในขั้นตอนนี้เป็นการเชื่อมต่อสัญญาณระหว่างองค์ประกอบภายในอุปกรณ์ FPGA เช่น ระหว่าง CLBs หรือระหว่าง CLBs กับ IOBs ขั้นตอนนี้จะทำต่อเนื่องจากการวางอุปกรณ์ ในกรณีที่วางอุปกรณ์ไว้ไม่ดี ซอฟท์แวร์ก็จะทำการเชื่อมต่อสัญญาณได้ไม่หมด เนื่องจากจำนวนทรัพยากรสำหรับการเชื่อมต่อสัญญาณนั้นมีอยู่จำกัด หรือเกิดความหน่วงเกินค่าที่กำหนดในข้อบังคับ

ผู้ออกแแบบสามารถทำขั้นตอนนี้โดยการใช้ซอฟท์แวร์หรือจะทำการเชื่อมต่อสัญญาณด้วยตัวเองก็ได้ แต่วิธีที่ดีควรใช้ซอฟท์แวร์ทำซ้ำหลายๆครั้งเพื่อหาครั้งที่ดีที่สุด นอกจากนั้นการกำหนดข้อบังคับทางเวลาจะช่วยให้ผลที่ได้จากการเชื่อมต่อสัญญาณดีขึ้นได้

2.5.6 ความหน่วงทางด้านเวลา (Delay)

ในการทำ FPGA นั้นความหน่วงที่เกิดขึ้น เป็นความหน่วงที่เกิดจากการวางตำแหน่ง (layout) ของซึ่งไม่สามารถแก้ไขได้ แต่สามารถทำให้มีความหน่วงน้อยที่สุดได้ สำหรับความหน่วงที่เกิดขึ้นนั้นแยกได้เป็นสองประเภท คือ

- *ความหน่วงลอจิก (Logic Delay)* เป็นความหน่วงภายในองค์ประกอบของอุปกรณ์ FPGA เอง เช่น ความหน่วงภายใน CLBs
- *ความหน่วงที่เกิดจากการเชื่อมต่อสัญญาณ (Routing Delay)* เป็นความหน่วงที่เกิดจากการเชื่อมต่อสัญญาณระหว่างองค์ประกอบภายในอุปกรณ์ FPGA

โดยปกติแล้ว ความหน่วงลอจิกควรมีค่าไม่เกิน 50 % ของค่าความหน่วงที่ยอมรับได้ เพราะว่าความหน่วงที่เกิดจากการเชื่อมต่อสัญญาณ มักจะมีค่ามากกว่าค่าความหน่วงลอจิก ดังนั้นในการวางอุปกรณ์และเชื่อมต่อสัญญาณ ผู้ออกแบบควรกำหนดข้อบังคับทางเวลาเพื่อให้ซอฟท์แวร์ได้ทำงานอย่างมีประสิทธิภาพเพิ่มขึ้นและเพื่อผลลัพธ์ที่ดีขึ้น

ค่าความหน่วงที่ได้หลังจากการวางอุปกรณ์และเชื่อมต่อสัญญาณแล้ว เป็นค่าความหน่วงที่ค่อนข้างแน่นอน ซึ่งผู้ออกแบบสามารถทราบได้ว่าโมเดลที่ออกแบบนั้นเป็นไปตามข้อกำหนดหรือไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

2.5.7 การโปรแกรมอุปกรณ์ FPGA (Configuration)

หลังจากที่โมเดลผ่านขั้นตอนต่างๆ จนกระทั่งผ่านการทำ PPR (Partition, Placement & Routing) แล้วนั้น ถึงตอนนี้ก็สามารถที่จะดาวน์โหลดลงในอุปกรณ์ FPGA ได้แล้ว ในการดาวน์โหลดนี้ ก่อนอื่นต้องแปลงแบบวงจรรวมที่ได้เป็นข้อมูลวงจร (configuration data) ซึ่งอยู่ในรูปของบิตสตรีม (bit - stream) ก่อน แล้วจึงดาวน์โหลดลงไปเพื่อให้อุปกรณ์ FPGA มีฟังก์ชันการทำงานตามโมเดลที่ผู้ออกแบบต้องการ ในขั้นตอนนี้จะใช้วิธีที่แตกต่างกันออกไปสำหรับอุปกรณ์ FPGA ของแต่ละบริษัท ถ้าเป็นแบบที่ต้องโปรแกรมด้วยวิธี SRAM จะต้องมีการเก็บข้อมูลวงจร (configuration data) ไว้ในหน่วยความจำประเภท EPROM หรือ serial PROM ด้วยเพื่อที่จะได้ใช้งานได้สะดวกขึ้น ในการใช้งานโมเดลครั้งต่อไปไม่ต้องดาวน์โหลดข้อมูลวงจรอีก เพราะมีข้อมูลก็ไม่จำเป็นต้องมีหน่วยความจำไว้สำหรับเก็บข้อมูลวงจร เพราะอุปกรณ์ประเภทนี้ข้อมูลที่ดาวน์เก็บอยู่ในหน่วยความจำอยู่แล้ว แต่กรณีที่เป็นชนิดที่โปรแกรมโดยใช้วิธี EPROM หรือ Antifuse โหลดลงไปแล้ว จะยังคงอยู่ในอุปกรณ์ FPGA และครั้งต่อไปก็สามารถใช้งานโมเดลที่ออกแบบไว้ได้เลย

2.5.8 การจำลองการทำงานของวงจร (Simulation)

เป็นขั้นตอนที่สำคัญอีกขั้นตอน เพราะใช้ในการตรวจสอบฟังก์ชันการทำงานของโมเดลว่าถูกต้องหรือไม่ มีข้อผิดพลาดตรงไหนเพื่อจะได้ทำการแก้ไขให้ถูกต้อง ในขั้นตอนนี้จะมีซอฟท์แวร์สำหรับการจำลองการทำงานของวงจร ซึ่งควรทำทุกครั้งหลังจากการออกแบบในแต่ละขั้นตอนเสร็จแล้ว เพื่อจะได้ทราบว่าข้อผิดพลาดของโมเดลเกิดจากขั้นตอนไหนจะได้แก้ไขให้ถูกต้องตรงขั้นตอนนั้นๆได้เลย ไม่ต้องมาคอยตรวจหาว่าเกิดจากขั้นตอนใด นั่นคือ การทำการจำลองการทำงานของวงจรต้องทำทั้งหลังจากการเขียนโค้ด การสังเคราะห์วงจร และการทำ PPR ในการจำลองการทำงานของวงจรหลังจากการเขียนโค้ดแล้วนั้น จะทำให้ทราบได้แค่โมเดลทำงานถูกต้องหรือไม่(function test) แต่ไม่สามารถตรวจสอบการทำงานในเชิงเวลาได้ถูกต้อง การจำลองการทำงานของวงจรหลังจากการสังเคราะห์เป็นวงจรแล้วก็เป็นการตรวจสอบว่าฟังก์ชันการทำงานยังคงถูกต้องอยู่หรือไม่ และค่าความหน่วงที่เกิดขึ้นเป็นไปตามข้อบังคับหรือมีข้อผิดพลาดเกิดขึ้นหรือไม่ ถ้ามีก็จะได้แก้ไข ส่วนการจำลองการทำงานของวงจรหลังจากทำ PPR ก็มีความสำคัญเพราะผลที่ได้จากการ simulate ตอนนี้จะเป็นผลลัพธ์ของโมเดลเลย ซึ่งผู้ออกแบบนอกจากจะตรวจสอบฟังก์ชันการทำงานแล้วยังต้องตรวจสอบคุณสมบัติอื่นๆอีก เช่นความหน่วงที่ได้จากการทำ PPR ในรูปแบบค่าความหน่วงมาตรฐาน (Standard Delay Format : SDF) หรือการตรวจสอบว่าแบบวงจรรวมนี้สามารถใช้งานได้ที่ค่าความถี่สูงสุดเท่าใด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

50340

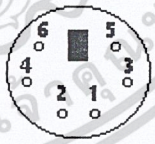
2.6 คีย์บอร์ด (Keyboard)

2.6.1 ลักษณะโดยทั่วไป

คีย์บอร์ดเป็นอุปกรณ์มาตรฐานที่ใช้ในการป้อนข้อมูลสำหรับเทอร์มินัลและคอมพิวเตอร์ โดยที่คีย์บอร์ดแต่ละตัวจะมีไมโครคอนโทรลเลอร์อยู่ภายในทำหน้าที่สร้างสัญญาณหรือรหัสที่ได้จากการกดแป้นพิมพ์และทำหน้าที่ส่งรหัสดังกล่าวผ่านสายส่งสัญญาณข้อมูลไปยังอุปกรณ์ปลายทางเพื่อแปลความหมายและแสดงผล ซึ่งการทำงานของคีย์บอร์ดนอกจากจะมีสายส่งสัญญาณข้อมูลแล้ว ยังมีสายส่งสัญญาณนาฬิกาเพื่อใช้ควบคุมจังหวะการรับส่งข้อมูลเมื่อมีการกดแป้นพิมพ์คีย์บอร์ดผ่านอุปกรณ์เชื่อมต่อระหว่างคีย์บอร์ดกับอุปกรณ์ปลายทาง ซึ่งเราจะแบ่งประเภทคีย์บอร์ดตามลักษณะการเชื่อมต่อได้ 3 แบบ คือคีย์บอร์ดแบบ AT Keyboard, คีย์บอร์ดแบบ XT keyboard และคีย์บอร์ดแบบ PS/2 keyboard ซึ่งในโครงงานนี้จะกล่าวถึงคีย์บอร์ดแบบ PS/2 Keyboard ที่ใช้ เท่านั้น ซึ่งมีการอินเตอร์เฟสแบบ PS/2 ที่มีการรับ-ส่งสัญญาณข้อมูลแบบอนุกรมทั้งสัญญาณนาฬิกาและสัญญาณข้อมูล โดยการรับส่งทั้ง 2 สัญญาณนี้จะเป็นแบบ สองทิศทาง(bi-directional) และแทนสถานะไฟเลี้ยงเท่ากับลอจิก 1 แทนสถานะกราวนด์ด้วยลอจิก 0 เมื่อสายสัญญาณข้อมูลและสายสัญญาณนาฬิกาไม่มีการใช้งานก็จะถูกปล่อยลอยไว้ทำให้ค่าเอาท์พุทมีค่าอิมพีแดนซ์สูงและแสดงลอจิกเป็น 1

2.6.2 ลักษณะการเชื่อมต่อ (The Connector)

การเชื่อมต่อของคีย์บอร์ดแบบ PS/2 จะใช้คอนเนคเตอร์แบบ 6-pin mini-DIN ซึ่งมีลักษณะดังรูปที่ 2.8



รูปที่ 2.8a คอนเนคเตอร์ตัวผู้ **รูปที่ 2.8b คอนเนคเตอร์ตัวเมีย**

รูปที่ 2.8 แสดง คอนเนคเตอร์ที่ใช้กับคีย์บอร์ดแบบ PS/2

จากรูป หน้าที่ของขาต่างๆของคอนเนคเตอร์มีดังนี้คือ

- ขาที่ 1 ขาสัญญาณข้อมูล (Data)
- ขาที่ 2 ไม่มีการใช้งาน (No connect)
- ขาที่ 3 ขากราวนด์ (GND)
- ขาที่ 4 ขาไฟเลี้ยง ซึ่งจะรับมาจากอุปกรณ์ที่คีย์บอร์ดต่ออยู่ +5 โวลต์ (+5VCC) ใช้ กระแสสูงสุดประมาณ 200 มิลลิแอมป์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ขาที่ 5 ขาสัญญาณนาฬิกา (CLK) เป็นตัวกำหนดอัตราเร็วในการส่งสัญญาณนาฬิกา
- ขาที่ 6 ไม่มีการใช้งาน (No connect)

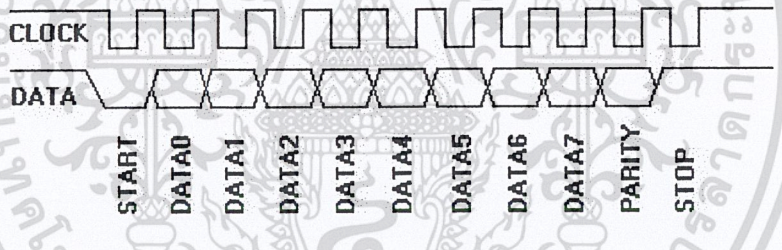
ซึ่งสรุปเป็นตารางดังนี้

ตารางที่ 2.1 แสดงคอนเนคเตอร์แบบ PS/2

ตำแหน่งขา (pin)	1	2	3	4	5	6
การส่งสัญญาณ (signal)	DATA	NC	GND	+VCC	CLK	NC

2.6.3 ลักษณะการส่งข้อมูลของคีย์บอร์ด

ในการกดแป้นพิมพ์ของคีย์บอร์ดแต่ละครั้ง คีย์บอร์ดจะส่งข้อมูลออกมาทั้งหมด 11 บิต พร้อมๆ กับการส่งสัญญาณนาฬิกา ซึ่งการส่งข้อมูลจะเป็นแบบอนุกรม แสดงได้ดังรูปที่ 2.9



รูปที่ 2.9 แสดงลักษณะการส่งข้อมูลของคีย์บอร์ด

จากรูปที่ 2.9 ข้อมูลจะถูกส่งออกมาเมื่อสัญญาณนาฬิกาของคีย์บอร์ดมีการเปลี่ยนแปลงจากสถานะ 1 เป็น 0 ข้อมูลที่ส่งออกมาแต่ละบิตแสดงได้ดังตารางที่ 2.2 โดยบิตแรกจะมีสถานะลอจิกเป็นศูนย์เสมอและเป็นบิตเริ่มต้น(start bit) อีกแปดบิตต่อมาเป็นบิตข้อมูล (data bit) ของ แป้นพิมพ์ที่ถูกกด บิตต่อมาเป็นบิตที่ใช้ตรวจสอบความถูกต้องของสัญญาณหรือพาริตี้บิต (parity bit) โดยจะมีสถานะเป็นหนึ่ง เมื่อจำนวนบิตข้อมูลที่ส่งออกมาทั้งหมดที่มีสถานะเป็นหนึ่งรวมกันได้เป็นจำนวนเลขคู่ และพาริตี้บิตจะมีสถานะเป็นศูนย์เมื่อจำนวนบิตข้อมูลรวมกันได้เป็นจำนวนเลขคี่ และบิตสุดท้ายจะมีสถานะเป็นหนึ่งเสมอ (stop bit) ซึ่งจะแสดงให้รู้ว่าสิ้นสุดการส่งข้อมูลแล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2.2 แสดงข้อมูลที่ส่งมาจากคีย์บอร์ด

บิต (Bit)	สัญญาณ (Signal)
1	Start Bit (สถานะ 0)
2	Data0
3	Data1
4	Data2
5	Data3
6	Data4
7	Data5
8	Data6
9	Data7
10	Parity Bit
11	Stop Bit (สถานะ 1)

ในการกดแป้นพิมพ์แต่ละครั้ง ข้อมูลที่คีย์บอร์ดส่งออกมาจะเรียกว่า สแกนโค้ด (Scan Codes) เพื่อแสดงว่าแป้นพิมพ์ถูกกด สแกนโค้ดที่แสดงว่าแป้นพิมพ์ถูกกดอยู่คือ เมคโค้ด (Make Code) และคีย์บอร์ดจะส่ง เบรคโค้ด (Break Code) ออกมาเมื่อมีการปล่อยแป้นพิมพ์นั้นๆ โดยแป้นพิมพ์แต่ละแป้นจะมีค่าสแกนโค้ดแตกต่างกัน

ค่าสแกนโค้ดที่ใช้กันอยู่ในปัจจุบันมีหลายแบบ แต่แบบที่ใช้กับคีย์บอร์ด PS/2 ในโครงงานนี้แสดงได้ตามตารางที่ 2.3

ตารางที่ 2.3 แสดงค่าสแกนโค้ด (Keyboard Scan Codes) แสดงในรหัสฐานสิบหก

ใช้กับคีย์บอร์ดจำนวน 101, 102, และ 104 คีย์

KEY	MAKE	BREAK	KEY	MAKE	BREAK	KEY	MAKE	BREAK
A	1C	F0,1C	9	46	F0,46	[	54	FO,54
B	32	F0,32	`	0E	F0,0E	INSERT	E0,70	E0,F0,70
C	21	F0,21	-	4E	F0,4E	HOME	E0,6C	E0,F0,6C
D	23	F0,23	=	55	FO,55	PG UP	E0,7D	E0,F0,7D
E	24	F0,24	\	5D	F0,5D	DELETE	E0,71	E0,F0,71
F	2B	F0,2B	BKSP	66	F0,66	END	E0,69	E0,F0,69

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

KEY	MAKE	BREAK	KEY	MAKE	BREAK	KEY	MAKE	BREAK
G	34	F0,34	SPACE	29	F0,29	PG DN	E0,7A	E0,F0,7A
H	33	F0,33	TAB	0D	F0,0D	U ARROW	E0,75	E0,F0,75
I	43	F0,43	CAPS	58	F0,58	L ARROW	E0,6B	E0,F0,6B
J	3B	F0,3B	L SHFT	12	FO,12	D ARROW	E0,72	E0,F0,72
K	42	F0,42	L CTRL	14	FO,14	R ARROW	E0,74	E0,F0,74
L	4B	F0,4B	L GUI	E0,1F	E0,F0,1F	NUM	77	F0,77
M	3A	F0,3A	L ALT	11	F0,11	KP /	E0,4A	E0,F0,4A
N	31	F0,31	R SHFT	59	F0,59	KP *	7C	F0,7C
O	44	F0,44	R CTRL	E0,14	E0,F0,14	KP -	7B	F0,7B
P	4D	F0,4D	R GUI	E0,27	E0,F0,27	KP +	79	F0,79
Q	15	F0,15	R ALT	E0,11	E0,F0,11	KP EN	E0,5A	E0,F0,5A
R	2D	F0,2D	APPS	E0,2F	E0,F0,2F	KP .	71	F0,71
S	1B	F0,1B	ENTER	5A	F0,5A	KP 0	70	F0,70
T	2C	F0,2C	ESC	76	F0,76	KP 1	69	F0,69
U	3C	F0,3C	F1	05	F0,05	KP 2	72	F0,72
V	2A	F0,2A	F2	06	F0,06	KP 3	7A	F0,7A
W	1D	F0,1D	F3	04	F0,04	KP 4	6B	F0,6B
X	22	F0,22	F4	0C	F0,0C	KP 5	73	F0,73
Y	35	F0,35	F5	03	F0,03	KP 6	74	F0,74
Z	1A	F0,1A	F6	0B	F0,0B	KP 7	6C	F0,6C
0	45	F0,45	F7	83	F0,83	KP 8	75	F0,75
1	16	F0,16	F8	0A	F0,0A	KP 9	7D	F0,7D
2	1E	F0,1E	F9	01	F0,01	]	5B	F0,5B
3	26	F0,26	F10	09	F0,09	;	4C	F0,4C
4	25	F0,25	F11	78	F0,78	'	52	F0,52
5	2E	F0,2E	F12	07	F0,07	,	41	F0,41

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

KEY	MAKE	BREAK	KEY	MAKE	BREAK	KEY	MAKE	BREAK
6	36	F0,36	PRNT SCRN	E0,12, E0,7C	E0,F0, 7C,E0, F0,12	.	49	F0,49
7	3D	F0,3D	SCROL L	7E	F0,7E	/	4A	F0,4A
8	3E	F0,3E	PAUSE	E1,14,77, E1,F0,14, F0,77	-NONE-			

ACPI Scan Codes:

Key	Make Code	Break Code
Power	E0, 37	E0, F0, 37
Sleep	E0, 3F	E0, F0, 3F
Wake	E0, 5E	E0, F0, 5E

Windows Multimedia Scan Codes:

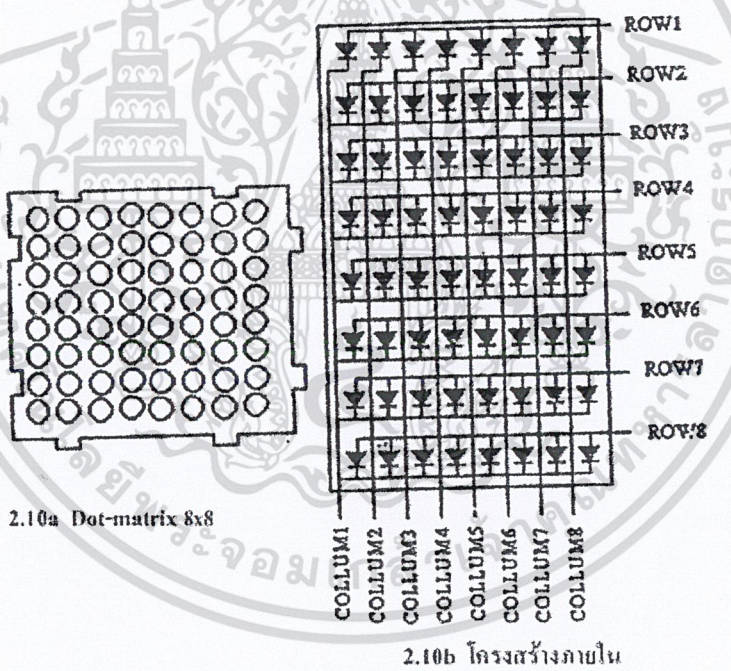
Key	Make Code	Break Code
Next Track	E0, 4D	E0, F0, 4D
Previous Track	E0, 15	E0, F0, 15
Stop	E0, 3B	E0, F0, 3B
Play/Pause	E0, 34	E0, F0, 34
Mute	E0, 23	E0, F0, 23
Volume Up	E0, 32	E0, F0, 32
Volume Down	E0, 21	E0, F0, 21
Media Select	E0, 50	E0, F0, 50
E-Mail	E0, 48	E0, F0, 48
Calculator	E0, 2B	E0, F0, 2B
My Computer	E0, 40	E0, F0, 40
WWW Search	E0, 10	E0, F0, 10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

WWW Home	E0, 3A	E0, F0, 3A
WWW Back	E0, 38	E0, F0, 38
WWW Forward	E0, 30	E0, F0, 30
WWW Stop	E0, 28	E0, F0, 28
WWW Refresh	E0, 20	E0, F0, 20
WWW Favorites	E0, 18	E0, F0, 18

2.7 แอลอีดี ดอทเมตริกซ์ (LED- Dot matrix)

สำหรับส่วนแสดงผลที่ใช้ในโครงงานนี้ เป็นดอทเมตริกซ์สีแดง ขนาด 8 x 8 จำนวน 10 ชุด ซึ่งมีโครงสร้างภายในดังรูปที่ 2.10



รูปที่ 2.10 แสดงโครงสร้างภายในของดอทเมตริกซ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

Master Flex-A01

3.1 บทนำ

สำหรับโครงงานนี้ ใช้บอร์ดทดลองเอฟพีจีเอ Master Flex-A01 ที่ออกแบบมาสำหรับอุปกรณ์เอฟพีจีเอในตระกูล flex 10k เบอร์ EPF10K20TC144 ซึ่งมีความจุเกตประมาณ 20,000 เกต ทำให้สามารถออกแบบวงจรหรือระบบดิจิตอลที่มีขนาดใหญ่หรือซับซ้อนได้ดี อีกทั้งภายในบอร์ด Master Flex-A01 ยังประกอบด้วยชุดอินเตอร์เฟสที่สำคัญๆที่จำเป็นกับโครงงานนี้ เช่น พอร์ตสำหรับการเชื่อมต่อแบบ PS/2 เป็นต้น ทำให้การออกแบบและใช้งานชิฟเอฟพีจีเอกับอุปกรณ์ภายนอกสามารถทำได้สะดวกมากยิ่งขึ้น

3.2 โครงสร้างของบอร์ด Master Flex-A01

ภายในบอร์ดจะประกอบไปด้วย

- วงจรรวมตระกูล FLEX10K ในอนุกรม EPF10K20TC144
- JTAG CONNECTOE
- พอร์ตขยายช่องสัญญาณขนาด 22x2 2 ชุด
- คอนเน็กเตอร์สำหรับส่งข้อมูลอนุกรมแบบ DB9 ตัวผู้ 1 ชุดพร้อมไอซีตัวแปลงระดับสัญญาณดิจิตอลเป็นระดับของ RS-232
- คอนเน็กเตอร์สำหรับเชื่อมต่อกับจอ VGA
- คอนเน็กเตอร์แบบ Centronics Port สำหรับการส่งข้อมูลแบบขนาน
- Analog to Digital Converter ขนาด 8 บิต
- Digital to Analog Converter ขนาด 8 บิต
- สวิตซ์แบบกดติด - ปล่อยดับ 2 ตัว
- ดิพสวิตซ์ 8 บิต 1ชุด
- ไดโอดเปล่งแสง 8 ดวง
- ชุดแสดงผล 7-Segment 4 หลัก ชนิด Common Anode แบบ Multiplex
- โมดูลออสซิลเลเตอร์ความถี่ 25.175 MHz 1 ชุด
- คอนเน็กเตอร์แหล่งจ่ายไฟให้กับบอร์ดสำหรับไฟกระแสตรงหรือกระแสสลับแรงดัน 7 ถึง 12 โวลต์
- คอนเน็กเตอร์แหล่งจ่ายไฟให้กับบอร์ดสำหรับไฟกระแสตรง -+15 โวลต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 3.1 ความสัมพันธ์ระหว่างพอร์ตขยายช่องสัญญาณ Ext A และ Ext B กับตำแหน่งขาของ EPF10K20TC144

Ext A Pin Number	EPF10K20 Pin Number	Ext B Pin Number	EPF10K20 Pin Number
1	144(I/O,NCS)	1	72(I/O)
2	143(I/O,CS)	2	70(I/O)
3	142(I/O,NWS)	3	69(I/O)
4	141(I/O,NRS)	4	68(I/O)
5	140(I/O)	5	67(I/O)
6	138(I/O)	6	65(I/O)
7	137(I/O)	7	64(I/O)
8	136(I/O)	8	63(I/O)
9	135(I/O)	9	62(I/O)
10	133(I/O)	10	60(I/O)
11	132(I/O)	11	59(I/O)
12	131(I/O)	12	56(Ded.Input)
13	130(I/O)	13	55(GlobalCLK)
14	128(I/O,DEV_OE)	14	54(Ded.Input)
15	126(Ded.Input)	15	51(I/O)
16	125(Global CLK)	16	49(I/O)
17	124(Ded.Input)	17	48(I/O)
18	122 (I/O,DEVCLR)	18	47(I/O)
19	121(I/O)	19	46(I/O)
20	120(I/O)	20	44(I/O)
21	119(I/O)	21	43(I/O)
22	118(I/O)	22	42(I/O)
23	117(I/O)	23	41(I/O)
24	116(I/O,DATA7)	24	39(I/O)
25	114(I/O,DATA6)	25	38(I/O)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Ext A Pin Number	EPF10K20 Pin Number	Ext B Pin Number	EPF10K20 Pin Number
26	113(I/O,DATA5)	26	37(I/O)
27	112(I/O,DATA4)	27	36(I/O)
28	111(I/O,DATA3)	28	33(I/O)
29	110(I/O,DATA2)	29	32(I/O)
30	109(I/O,DATA1)	30	31(I/O)
31	102(I/O)	31	30(I/O)
32	101(I/O)	32	29(I/O)
33	100(I/O)	33	28(I/O)
34	99(I/O)	34	27(I/O)
35	98(I/O)	35	26(I/O)
36	97(I/O)	36	23(I/O)
37	96(I/O)	37	22(I/O)
38	95(I/O)	38	21(I/O)
39	92(I/O)	39	20(I/O)
40	91(I/O)	40	19(I/O)
41	90(I/O)	41	18(I/O)
42	89(I/O)	42	17(I/O)
43	88(I/O)	43	14(I/O,INTDONE)
44	87(I/O)	44	13(I/O)
45	86(I/O)	45	12(I/O)
46	83(I/O)	46	11(I/O,RDBUSY)
47	82(I/O)	47	10(I/O)
48	81(I/O)	48	9(I/O)
49	80(I/O)	49	8(I/O)
50	79(I/O)	50	7(I/O,CLKUSR)
51	78(I/O)	51	GND
52	73(I/O)	52	VCC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 3.2 ความสัมพันธ์ระหว่างวงจรสื่อสารข้อมูลแบบอนุกรมกับตำแหน่งขาของ EPF10K20TC144

Serial Port Pin Name	EPF10K20 Pin Number	Serial Port Pin Name	EPF10K20 Pin Number
Rx	144(I/O,NCS)	Tx	143(I/O,CS)

ตารางที่ 3.3 ความสัมพันธ์ระหว่างคอนเน็กเตอร์สำหรับเชื่อมต่อกับจอ VGA กับตำแหน่งขาของ EPF10K20TC144

VGA Port Pin Name	EPF10K20 Pin Number	VGA Port Pin Name	EPF10K20 Pin Number
R	135(I/O)	V-SYNC	140(I/O)
G	136(I/O)	H-SYNC	138(I/O)
B	137(I/O)	-	-

ตารางที่ 3.4 ความสัมพันธ์ระหว่างคอนเน็กเตอร์แบบ PS/2 กับตำแหน่งขาของ EPF10K20TC144

PS/2 Port Pin Name	EPF10K20 Pin Number	PS/2 Port Pin Name	EPF10K20 Pin Number
DATA	142(I/O,NWS)	CLK_PS2	141(I/O,NSR)

ตารางที่ 3.5 ความสัมพันธ์ระหว่างคอนเน็กเตอร์แบบ Centronics Port กับตำแหน่งขาของ EPF10K20TC144

Centronics Port Pin Name	EPF10K20 Pin Number	Centronics Port Pin Name	EPF10K20 Pin Number
DB0	7(I/O,CLKUSR)	/LE/CR	18(I/O)
DB1	8(I/O)	/INITIALIZE	19(I/O)
DB2	9(I/O)	/SLIN	20(I/O)
DB3	10(I/O)	/ERROR	21(I/O)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Centronics Port Pin Name	EPF10K20 Pin Number	Centronics Port Pin Name	EPF10K20 Pin Number
DB4	11(I/O,RDBUSY)	SLCT	22(I/O)
DB5	12(I/O)	PE	23(I/O)
DB6	13(I/O)	/ACK	26(I/O)
DB7	14(I/O,INTDONE)	BUSY	27(I/O)
/STROBE	17(I/O)	-	-

ตารางที่ 3.6 ความสัมพันธ์ระหว่างวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอล กับตำแหน่งขาของ EPF10K20TC144

Circuit Pin Name	EPF10K20 Pin Number	Circuit Pin Name	EPF10K20 Pin Number
Clock200k	102(I/O)	ADC4	97(I/O)
ADC0	101(I/O)	ADC5	96(I/O)
ADC1	100(I/O)	ADC6	95(I/O)
ADC2	99(I/O)	ADC7	92(I/O)
ADC3	98(I/O)	-	-

ตารางที่ 3.7 ความสัมพันธ์ระหว่างวงจรแปลงสัญญาณดิจิตอลเป็นอนาลอก กับตำแหน่งขาของ EPF10K20TC144

Circuit Pin Name	EPF10K20 Pin Number	Circuit Pin Name	EPF10K20 Pin Number
DAC0	91(I/O)	DAC4	87(I/O)
DAC1	90(I/O)	DAC5	86(I/O)
DAC2	89(I/O)	DAC6	83(I/O)
DAC3	88(I/O)	DAC7	82(I/O)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 3.8 ความสัมพันธ์ระหว่างวงจรหน่วยความจำ กับตำแหน่งขาของ EPF10K20TC144

Circuit Pin Name	EPF10K20 Pin Number	Circuit Pin Name	EPF10K20 Pin Number
/OE	41(I/O)	AD11	64(I/O)
/WE	42(I/O)	AD10	65(I/O)
DM7	43(I/O)	AD9	67(I/O)
DM6	44(I/O)	AD8	68(I/O)
DM5	46(I/O)	AD7	69(I/O)
DM4	47(I/O)	AD6	70(I/O)
DM3	48(I/O)	AD5	72(I/O)
DM2	49(I/O)	AD4	73(I/O)
DM1	51(I/O)	AD3	78(I/O)
DM0	59(I/O)	AD2	79(I/O)
AD14	60(I/O)	AD1	80(I/O)
AD13	62(I/O)	AD0	81(I/O)
AD12	63(I/O)	-	-

ตารางที่ 3.9 ความสัมพันธ์ระหว่างสวิตซ์กดติด-ปล่อยดับ กับตำแหน่งขาของ EPF10K20TC144

Circuit Pin Name	EPF10K20 Pin Number	Circuit Pin Name	EPF10K20 Pin Number
SW1	56(Ded.Input)	SW2	54(Ded.Input)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 3.10 ความสัมพันธ์ระหว่างดิพสวิตซ์ 8 บิต กับตำแหน่งขาของ

EPF10K20TC144

Circuit Pin Name	EPF10K20 Pin Number	Circuit Pin Name	EPF10K20 Pin Number
DSW0	28(I/O)	DSW4	32(I/O)
DSW1	29(I/O)	DSW5	33(I/O)
DSW2	30(I/O)	DSW6	36(I/O)
DSW3	31(I/O)	DSW7	37(I/O)

ตารางที่ 3.11 ความสัมพันธ์ระหว่างไดโอดเปล่งแสง กับตำแหน่งขาของ

EPF10K20TC144

Circuit Pin Name	EPF10K20 Pin Number	Circuit Pin Name	EPF10K20 Pin Number
LED0	102(I/O)	LED4	130(I/O)
LED1	121(I/O)	LED5	131(I/O)
LED2	122(I/O,DEVCLR)	LED6	132(I/O)
LED3	128(I/O,DEV_OE)	LED7	133(I/O)

ตารางที่ 3.12 ความสัมพันธ์ระหว่างวงจรแสดงผล 7-Segment 4 หลัก กับตำแหน่งขาของ

EPF10K20TC144

Circuit Pin Name	EPF10K20 Pin Number	Circuit Pin Name	EPF10K20 Pin Number
SegA	109(I/O,DATA1)	SegG	116(I/O,DATA7)
SegB	110(I/O,DATA2)	SegDOT	117(I/O)
SegC	111(I/O,DATA3)	CM1	118(I/O)
SegD	112(I/O,DATA4)	CM2	119(I/O)
SegE	113(I/O,DATA5)	CM3	39(I/O)
SegF	114(I/O,DATA6)	CM4	38(I/O)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 3.13 ความสัมพันธ์ระหว่างโมดูลออสซิเลเตอร์ กับตำแหน่งขาของ EPF10K20TC144

Circuit Pin Name	EPF10K20 Pin Number
Osc	55(Global CLK)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
การออกแบบและการสร้างวงจร

4.1 ลักษณะโดยทั่วไป

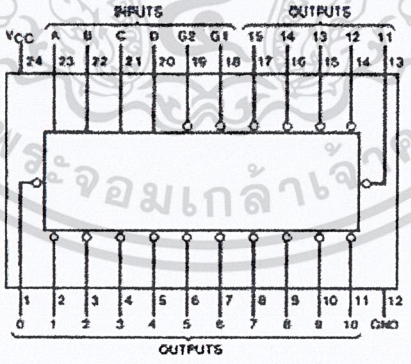
ในการออกแบบวงจรบอร์ดแสดงผลจะแบ่งเป็น 2 ส่วน คือ ส่วนวงจรควบคุมการแสดงผล ซึ่งใช้ชิฟเอฟพีจีเอตระกูล Flek10k เบอร์ EPF20K10TC144-4 มาควบคุมการทำงานและการประมวลผล และส่วนวงจรแสดงผลตัวอักษรซึ่งใช้ดอทเมตริกซ์ขนาด 8x8 จำนวน 10 ชุดต่อเรียงกัน

- ส่วนควบคุม

เมื่อมีการกดแป้นพิมพ์ คีย์บอร์ดจะส่งสัญญาณไปที่เอฟพีจีเอให้ทำการรับค่าและตรวจสอบทั้ง 11 บิตที่ส่งมาโดยจะตรวจสอบค่าพาริตี้บิตก่อนว่ามีความถูกต้องหรือไม่ จากนั้นก็ทำการตรวจสอบบิตข้อมูลที่ส่งมาว่ามีความถูกต้องหรือไม่ แล้วนำไปเปรียบเทียบกับข้อมูลที่มีอยู่ว่าเป็นข้อมูลของตัวอักษรอะไร เพื่อที่จะส่งข้อมูลออกไปที่ส่วนแสดงผลต่อไป

- ส่วนแสดงผล

ใช้ดอทเมตริกซ์ขนาด 8x80 จุด ในการแสดงตัวอักษรภาษาอังกฤษตัวพิมพ์ใหญ่ โดยใช้ไอซี เบอร์ 74LS154 ซึ่งแสดงการทำงานได้ดังรูปที่ 4.1 และตารางที่ 4.1 มาควบคุมดอทเมตริกซ์ให้ติดในหลัก(Column) และแถว(Row)ที่ต้องการ



รูปที่ 4.1 แสดงบล็อกไดอะแกรมการทำงานภายในของไอซี 74LS154

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.1 แสดงตารางค่าความจริงของไอซี 74LS154

รหัสเลขฐานสอง				เอาท์พุทที่ถูกเลือก
D	C	B	A	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

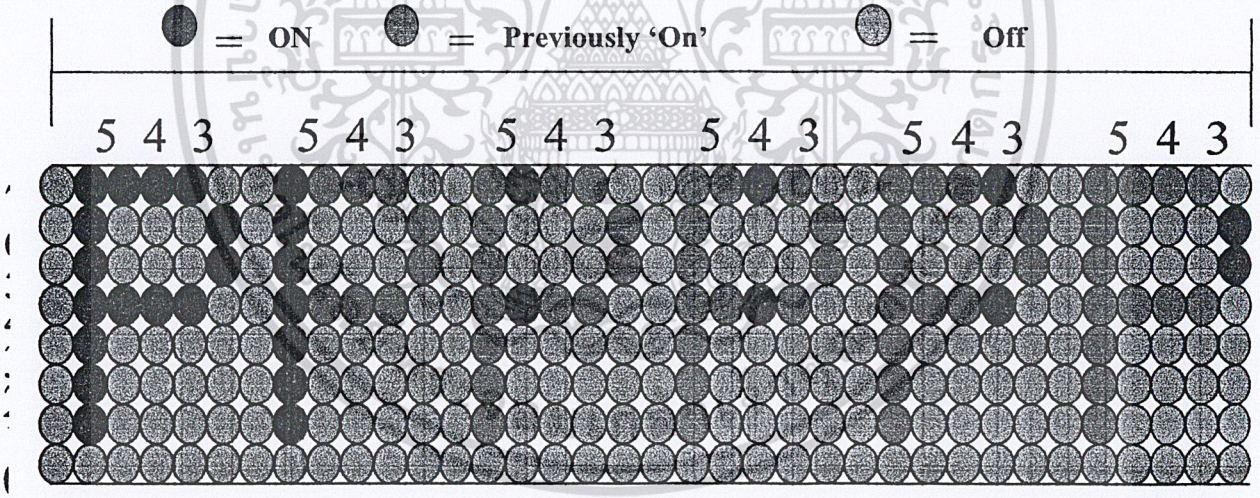
จากรูปที่ 4.1 ภายในจะมีวงจรสวิตซ์เลือก 16 ทางจำนวน4 ชุด และเอาท์พุทเป็นแบบมีบัฟเฟอร์แอคทีฟลอจิกโลว์ สามารถทำการดิสเอเบิ้ลได้โดยการไบอัสที่ขา INH ขาใดขาหนึ่งด้วยลอจิกไฮ การเลือกตำแหน่งสวิตซ์ภายในของไอซีควบคุมได้จากการป้อนรหัสเลขฐานสองขนาด 4 บิต (บิต A-D)ดังแสดงในตารางที่ 4.1 สำหรับการเชื่อมต่อเพื่อให้ไอซีทำหน้าที่เป็นวงจรถอดรหัสจาก 4 อินพุตเป็น 16 เอาท์พุตทำได้โดยการป้อนไฟเลี้ยง +5VCC ที่ขา 24 ขา 12 ต่อลงกราวนด์ ขา 18และ 19 ต่อเชื่อมกันและลงกราวนด์(ทำหน้าที่เป็นขา INH) สัญญาณเอาท์พุตที่ถอดรหัสได้จะขึ้นอยู่กับรหัสควบคุมสวิตซ์ที่ป้อนให้ ซึ่งแต่ละเอาท์พุตจะมีภาวะปกติเป็นสถานะลอจิกไฮ และจะถูกเลือกให้แอคทีฟเปลี่ยนเป็นลอจิกโลว์ได้ก็ต่อเมื่อที่เอาท์พุตนั้นถูกเลือกโดยการกำหนดรหัสควบคุม 4 บิตให้กับไอซี สำหรับในกรณีที่ต้องการนำไอซีมาใช้เป็นดีมัลติเพล็กเซอร์ สามารถ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำได้โดยการนำสัญญาณข้อมูลอินพุตป้อนเข้าที่ขา INH ขาใดขาหนึ่ง ส่วนขา INH ที่เหลืออีก 1 ขาให้ไบอัสด้วยลอจิกโลว์ไว้

4.2 หลักการสแกน แอลอีดี ดอทเมตริกซ์

ในการสแกนแอลอีดี ดอทเมตริกซ์ สามารถทำได้ 2 แบบทั่วๆไป คือการสแกนในแนว Row (แถว) และการสแกนในแนว Column (หลัก) โดยขึ้นอยู่กับความเหมาะสมของโครงสร้างทางฮาร์ดแวร์ และความยากง่ายของการเขียนโปรแกรมหรือซอฟท์แวร์ที่ใช้ควบคุม ในโครงงานนี้จะใช้การสแกนในแนว Column เพราะจะทำให้เขียนโปรแกรมควบคุมได้ง่ายกว่าการสแกนในแนว Row ขั้นตอนในการสแกนอธิบายได้ดังนี้คือ ในตอนแรกจะเริ่มสแกนที่ Row 1 ก่อนแล้วไปสิ้นสุดการสแกนที่ Row 8 แล้วกลับมาเริ่มการสแกนที่ Row 1 อีกครั้ง โดยให้มีความเร็วในการติดของดอทเมตริกซ์อยู่ที่ความเร็วค่าหนึ่ง ข้อมูลจะถูกเปลี่ยนไปตามตัวอักษรที่ต้องการ โดยที่ความเร็วในการสแกนที่เหมาะสมจะทำให้เห็นการติดของดอทเมตริกซ์เป็นตัวอักษรอย่างชัดเจน



รูปที่ 4.2 แสดงการสแกนแบบ คอลัมน์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 4.3 แสดงวงจรของบอร์ดแสดงผล

จากรูปที่ 4.3 ในภาคจ่ายไฟจะใช้ชิฟเอฟพีจีเอควบคุมการทำงานของไอซี 74HC541 ที่ทำหน้าที่เป็นตัวควบคุมการจ่ายกระแสให้กับวงจร โดยขา enable จะต่อลงกราวนด์ ส่วนขาอินพุทจะต่อเข้ากับขาเอาท์พุตของเอฟพีจีเอ ขาเอาท์พุตจะต่อเข้ากับขั้วบวกของ แอลอีดีดอทเมตริกซ์ที่ใช้ โดยที่อินพุตต้องมีสถานะลอจิกไฮ จึงจะจ่ายกระแสให้วงจรได้ สำหรับขั้วบวกของดอทเมตริกซ์แต่ละตัวจะต้องต่อเชื่อมกันและต่อเข้ากับขาเอาท์พุทของ ไอซี 74HC541ทั้ง 8 ขา ต่อขั้วลบเข้ากับขาเอาท์พุทของไอซี 74LS154

4.3 รายการอุปกรณ์ที่ใช้

- แอลอีดีดอทเมตริกซ์ขนาด 8x8 สีแดง	จำนวน	10	ตัว
- ไอซี 74HC541	จำนวน	1	ตัว
- ไอซี 74LS154	จำนวน	5	ตัว
- บอร์ดเอฟพีจีเอ Master Flex-A01	จำนวน	1	ชุด
- ชิฟเอฟพีจีเอเบอร์ EPF10K20TC144-4	จำนวน	1	ตัว
- แผ่นอะครีลิกใสขนาด 20" X 7"	จำนวน	1	แผ่น
- แผ่นอะครีลิกสีแดงใสขนาด 20" X 7"	จำนวน	1	แผ่น
- แผ่นปริ๊นไข่ปลาขนาดใหญ่(ปริ๊นอเนกประสงค์)	จำนวน	3	แผ่น
- อะแดปเตอร์ขนาด 12 โวลต์ 1.5 แอมป์	จำนวน	1	ตัว
- สายดาวน์โหลดโปรแกรม	จำนวน	1	เส้น
- คีย์บอร์ดคอมพิวเตอร์แบบ PS/2	จำนวน	1	ตัว
- สายแพขนาด 30 เส้นความยาว 50 ซ.ม.	จำนวน	1	เส้น
- ตัวล็อกสายแพ	จำนวน	1	ชุด
- คอนเน็กเตอร์ตัวเมียแถวเดี่ยวขนาด 40 PINs	จำนวน	12	ตัว
- ซ็อกเก็ตไอซี 20 ขา	จำนวน	1	ตัว
- ซ็อกเก็ตไอซี 24 ขา	จำนวน	5	ตัว

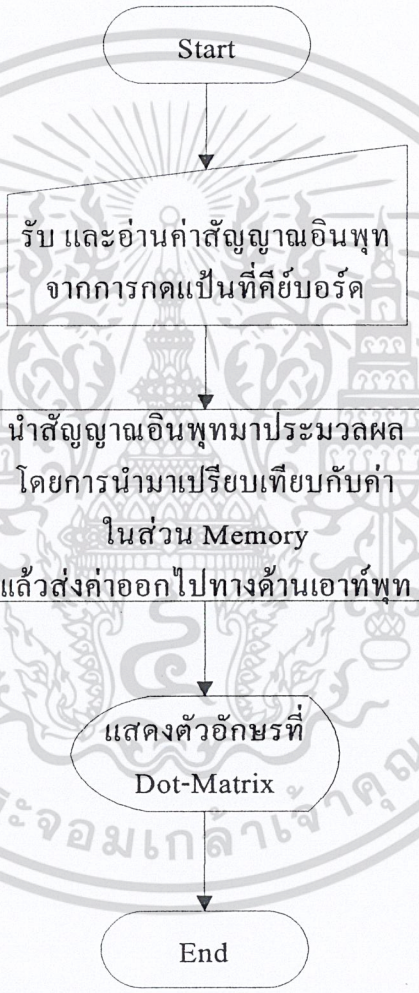
- ตะกั่วบัดกรี
- น้อตสำหรับยึดบอร์ดวงจร
- สาย wire
- โปรแกรม MaxPlusII

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
โปรแกรมควบคุมการทำงานและโฟล์วชาร์ต

5.1 โฟว์ชาร์ตการทำงานของวงจรทั้งหมด

การทำงานของวงจรทั้งหมดแสดงได้ดังรูปที่ 5.1

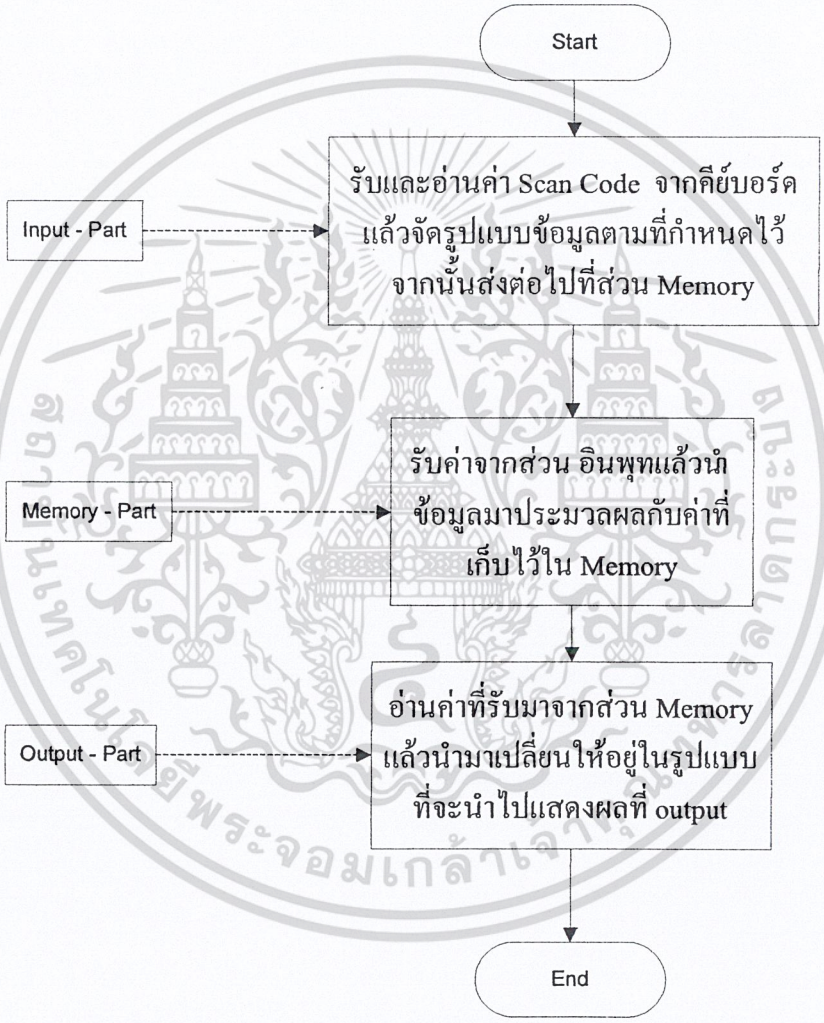


รูปที่ 5.1 แสดงโฟว์ชาร์ตของระบบ

สำหรับโครงงานนี้จะแบ่งโปรแกรมออกเป็น 3 ส่วนใหญ่ๆ คือ ส่วน Input – part ซึ่งจะทำหน้าที่รับและอ่านค่าอินพุท ส่วนต่อมาคือ ส่วน Memory – part ทำหน้าที่รับค่าจากส่วนแรกมาทำ

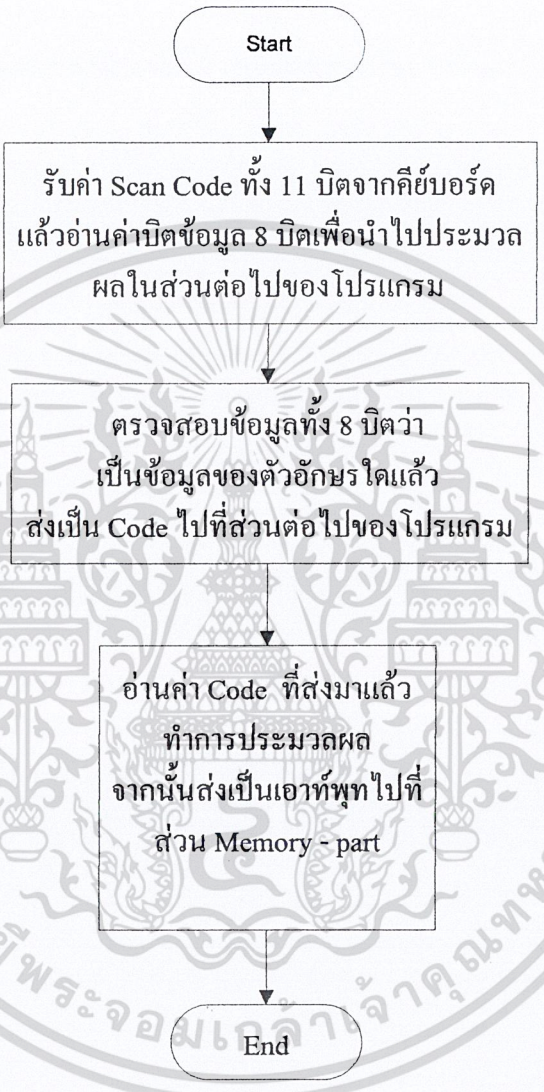
เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ ห้ามนำไปใช้ในการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การประมวลผลแล้วส่งต่อเป็นค่าอินพุทให้กับส่วนสุดท้าย คือ ส่วน Output – part ซึ่งจะควบคุมทำงานเกี่ยวกับการแสดงผลที่ดอทเมตริกซ์ที่ต่อเรียงกันอยู่ แสดงเป็นโฟว์ชาร์ตได้ดังรูปที่ 5.2



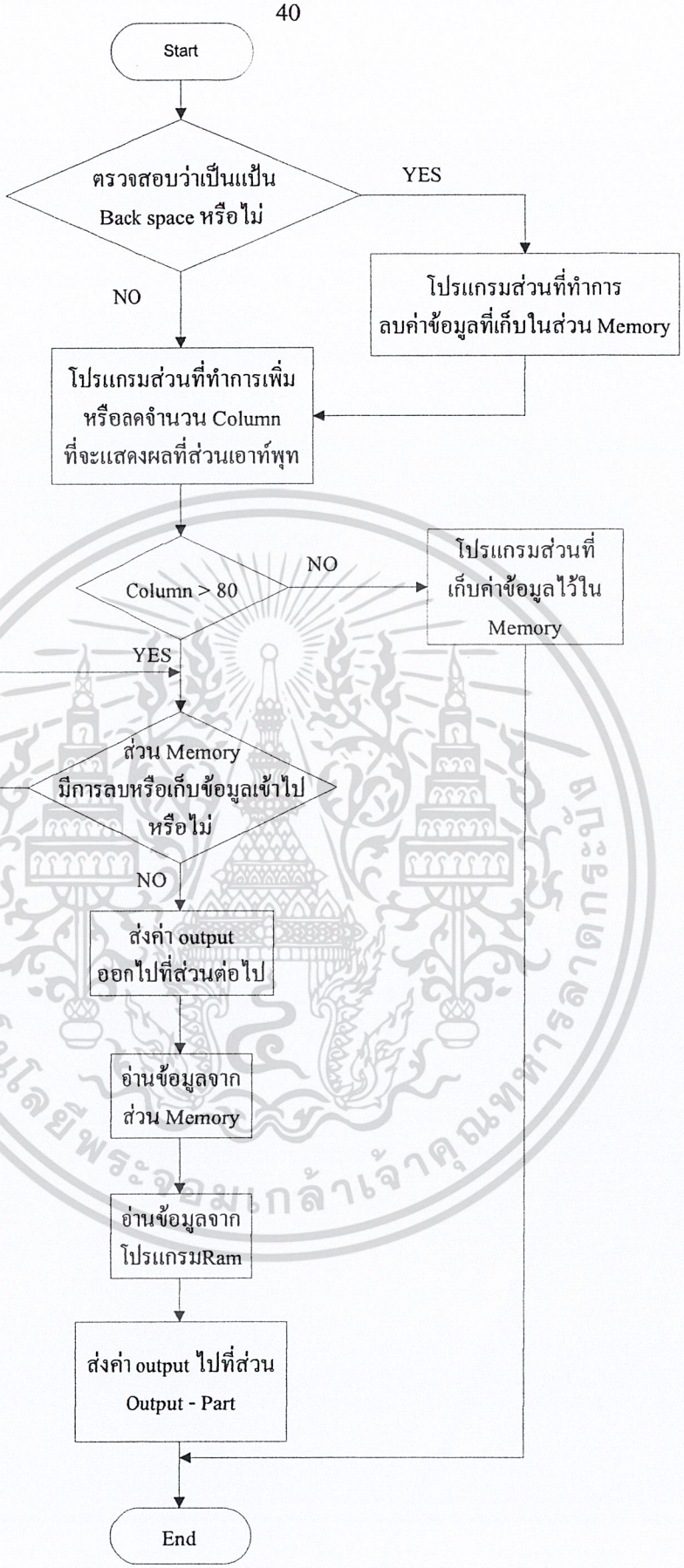
รูปที่ 5.2 แสดงโฟว์ชาร์ตลำดับการทำงานของโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.3 แสดงโฟว์ชาร์ตของส่วน Input - Part

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

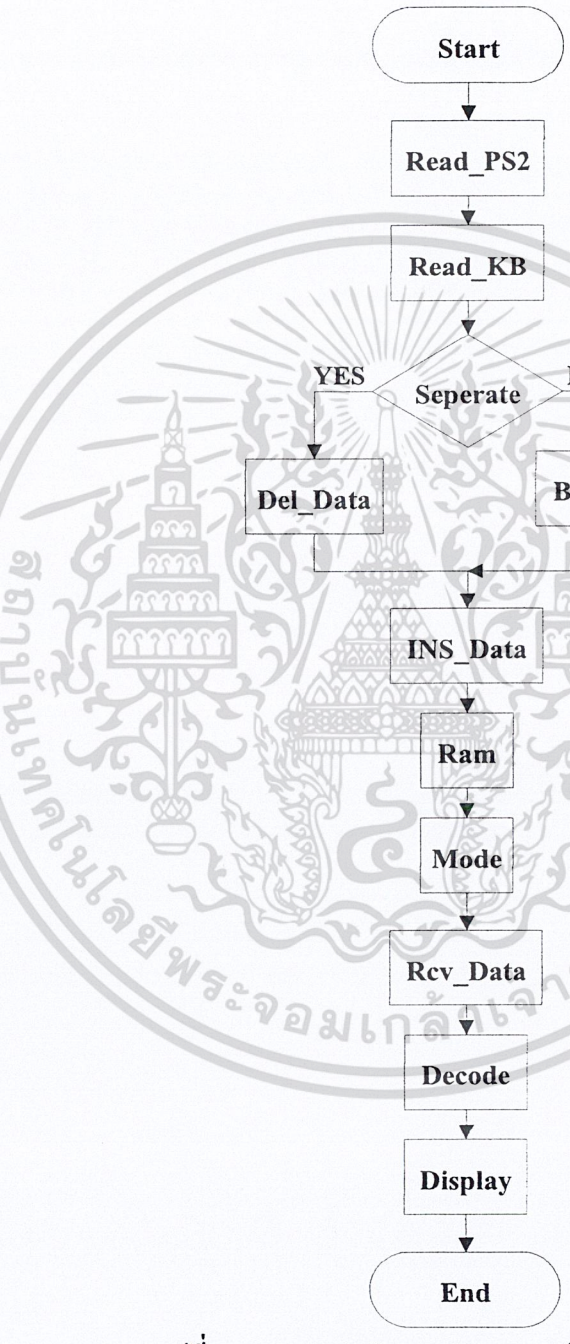


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 5.4 แสดงโฟว์ชาร์ตของส่วน Memory – Part

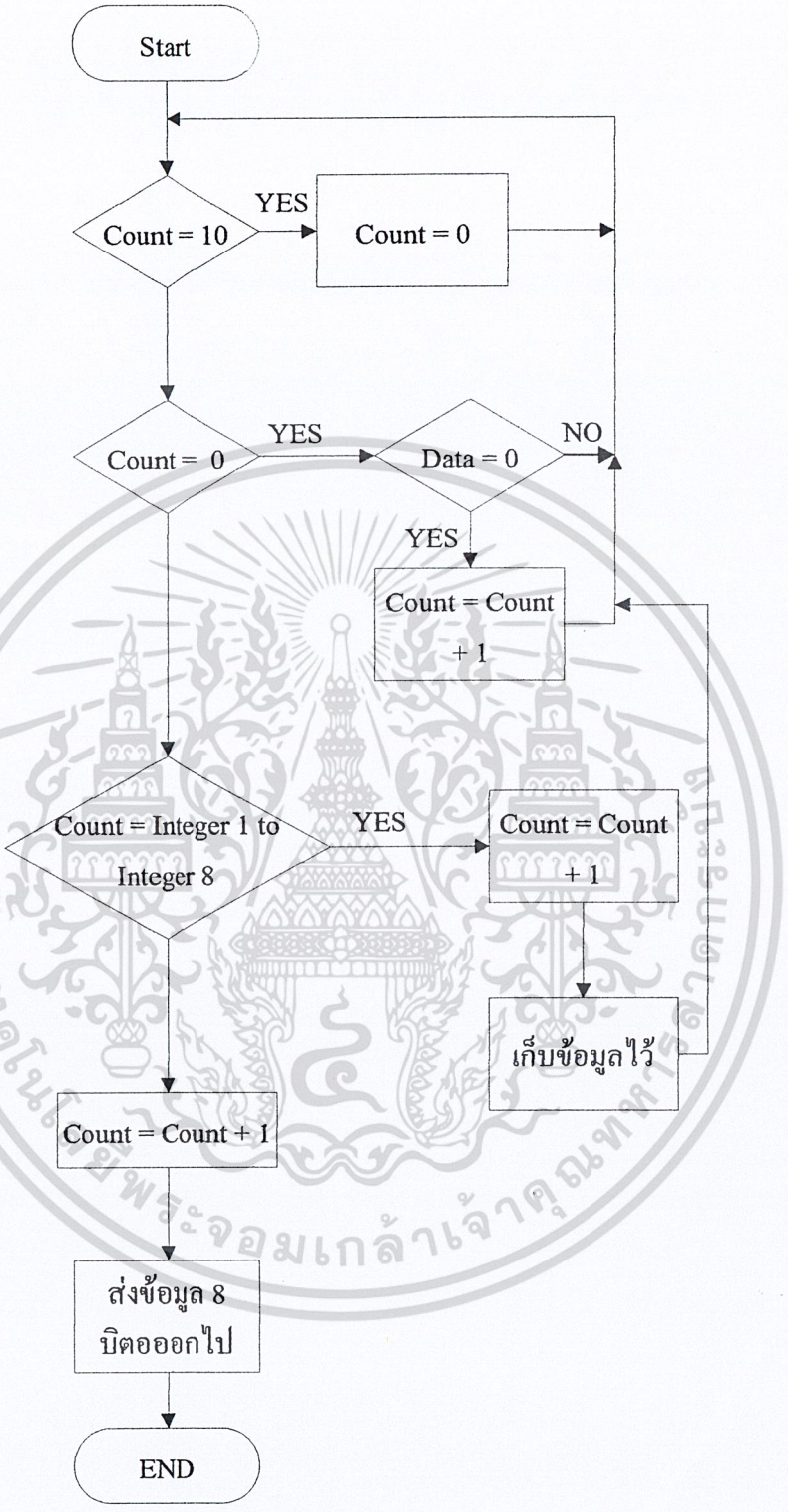
รูปที่ 5.5 แสดงโฟว์ชาร์ตของส่วน Output – Part

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



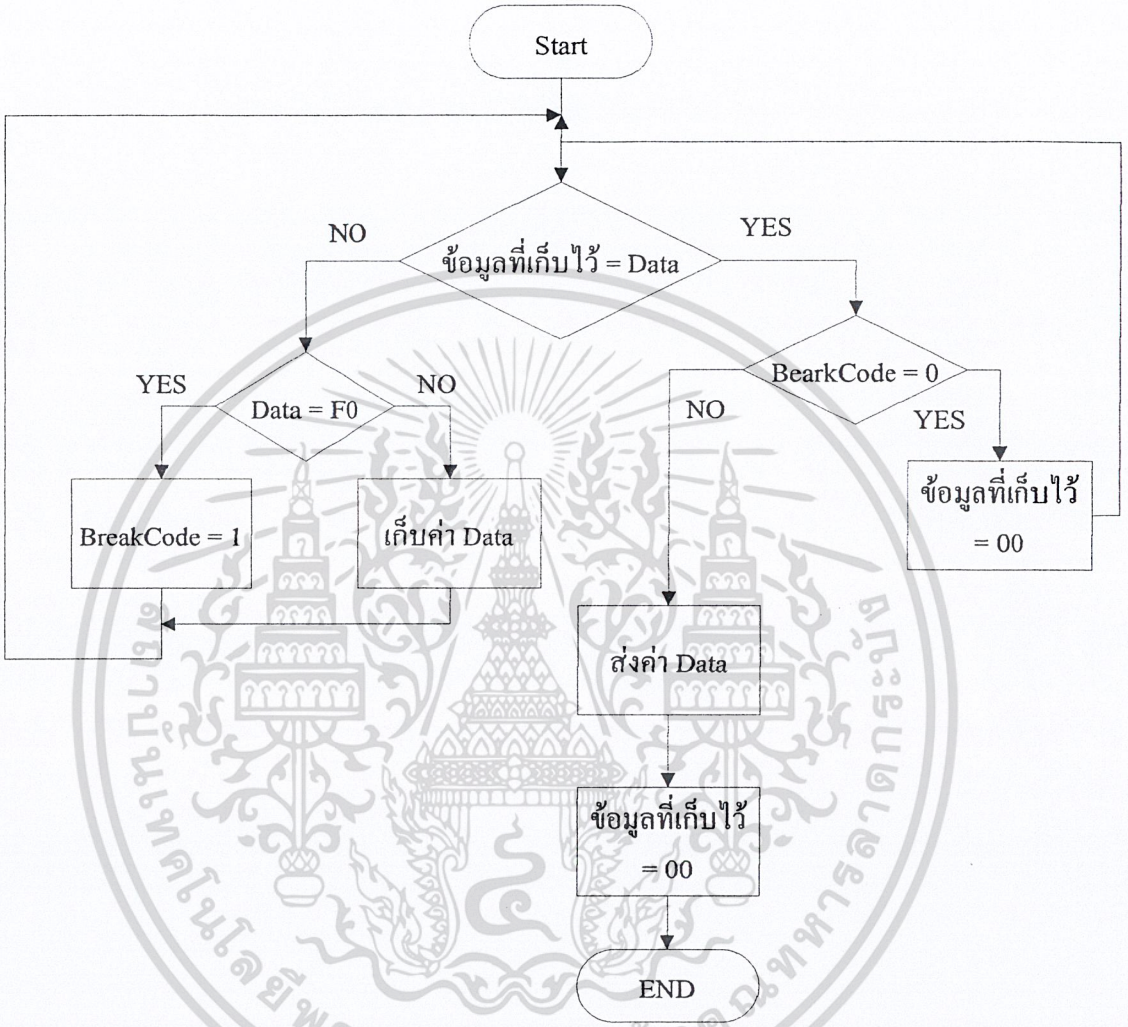
รูปที่ 5.6 แสดงลำดับการทำงานของโปรแกรมทั้งหมด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



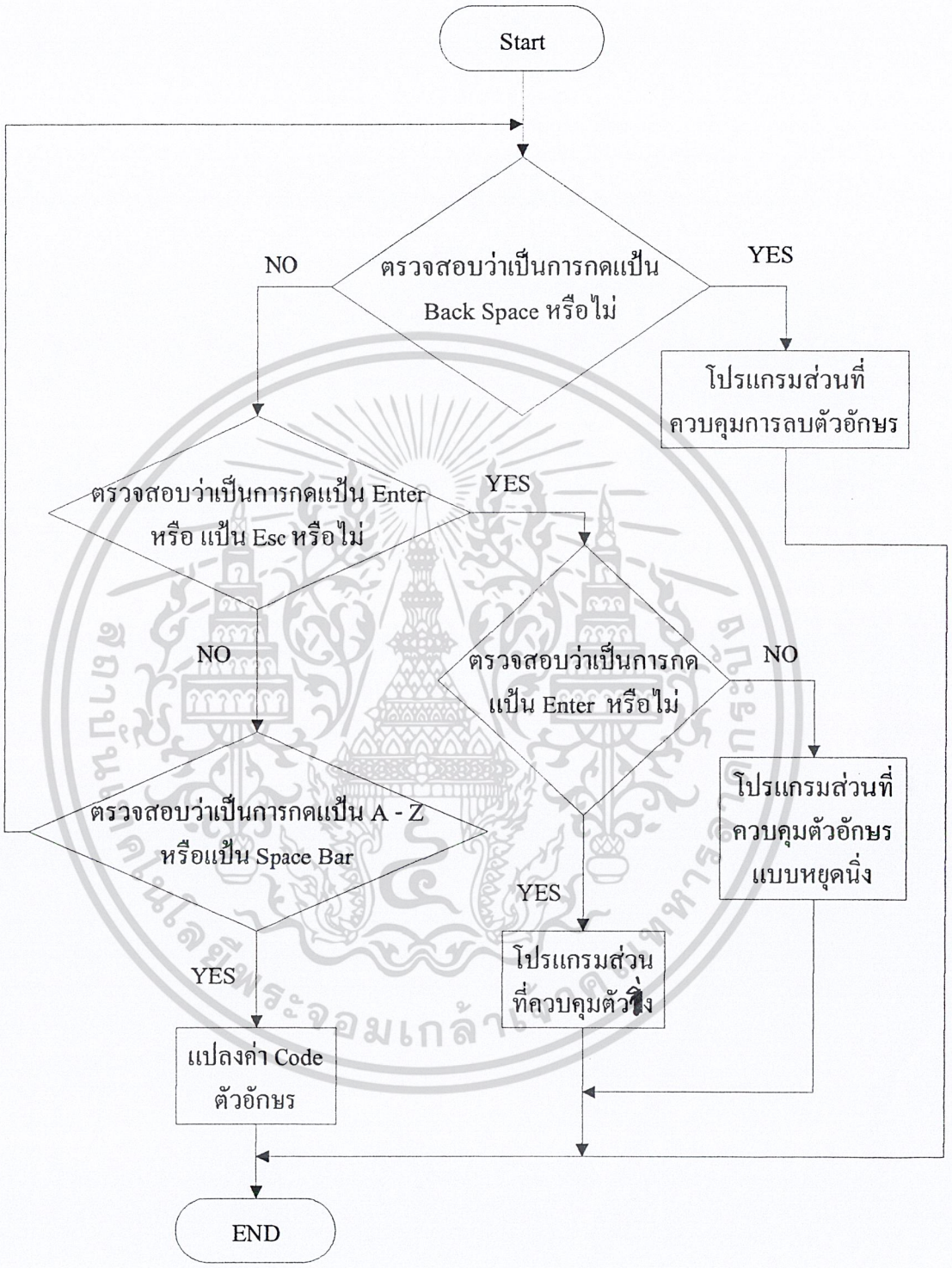
รูปที่ 5.7 แสดงโฟว์ชาร์ตของโปรแกรม READ_PS2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.8 แสดงโฟว์ชาร์ตของโปรแกรม READ_KB

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

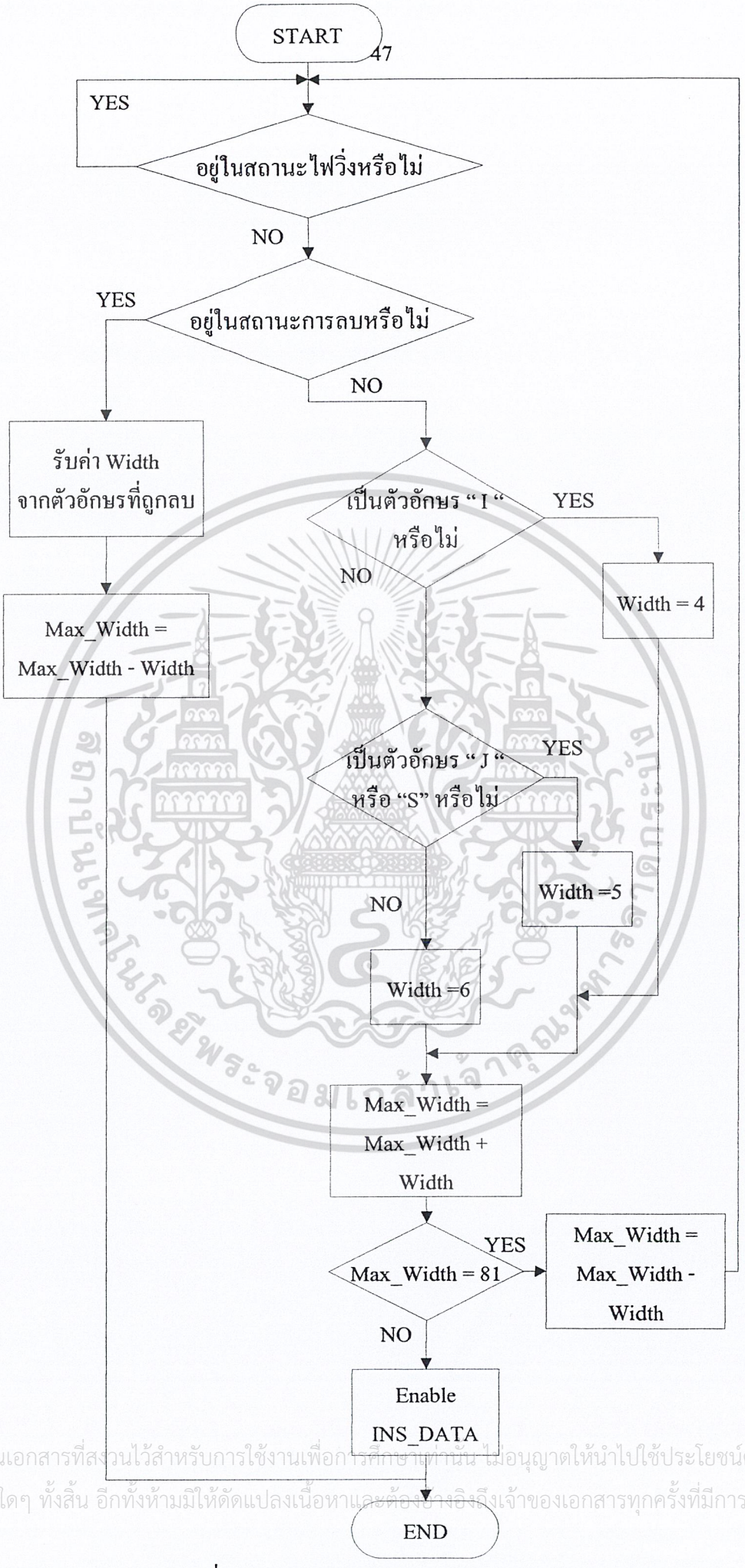


รูปที่ 5.9 แสดงโฟว์ชาร์ตของโปรแกรม SEPERATE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

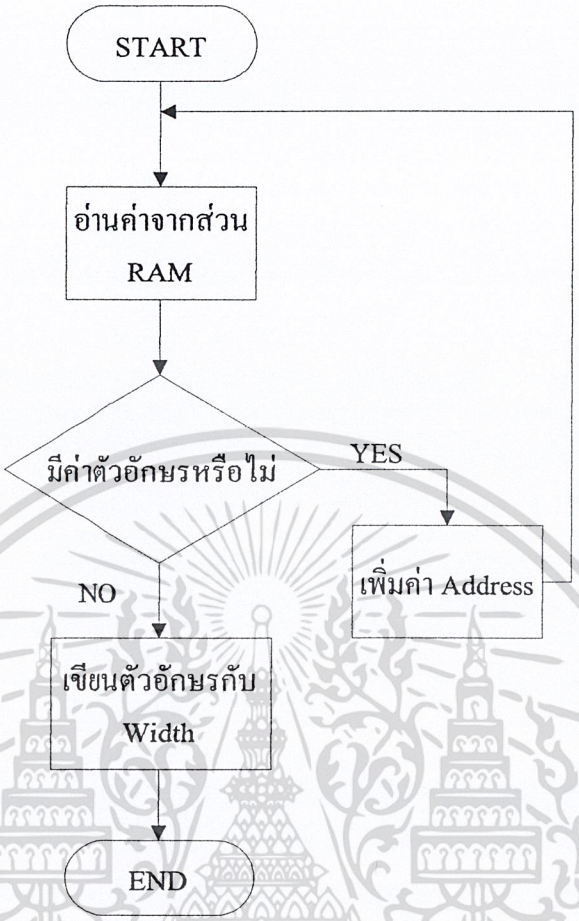
รูปที่ 5.10 แสดงโฟว์ชาร์ตของโปรแกรม DEL_DATA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



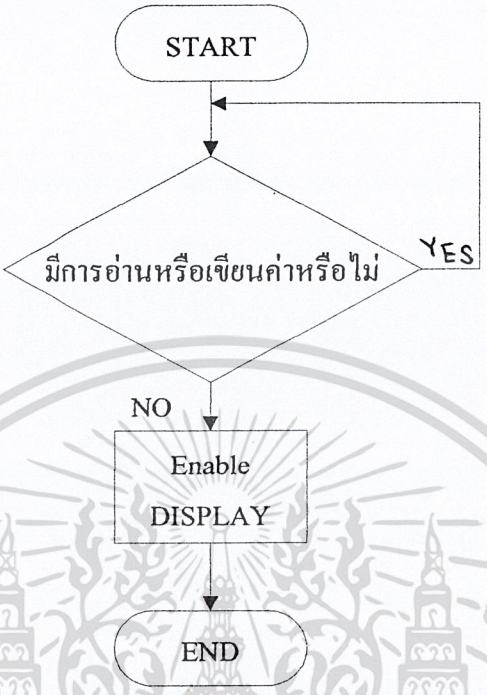
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 5.11 แสดงโฟว์ชาร์ตของโปรแกรม BLK DATA



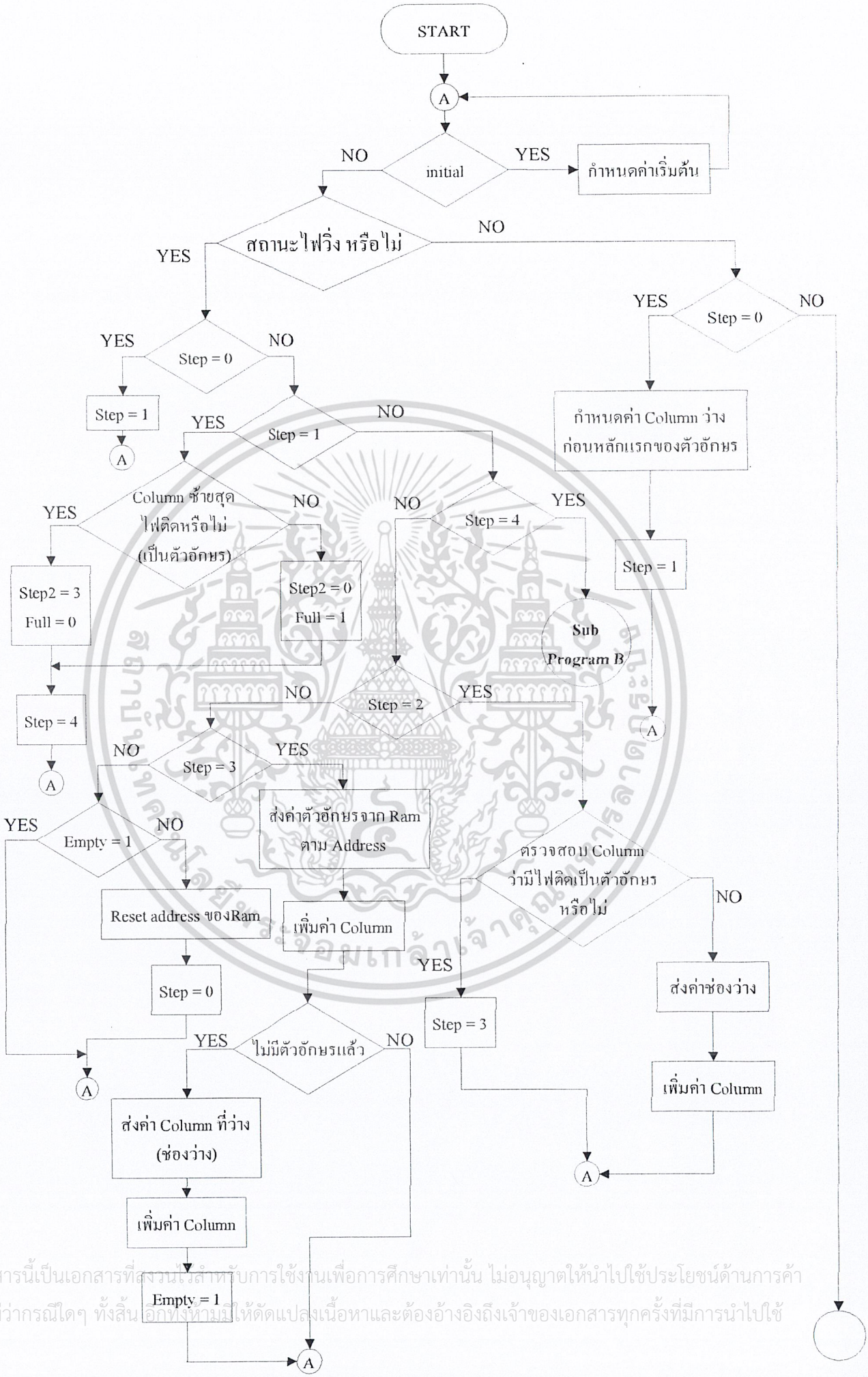
รูปที่ 5.12 แสดงโฟว์ชาร์ตของโปรแกรม INS_DATA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



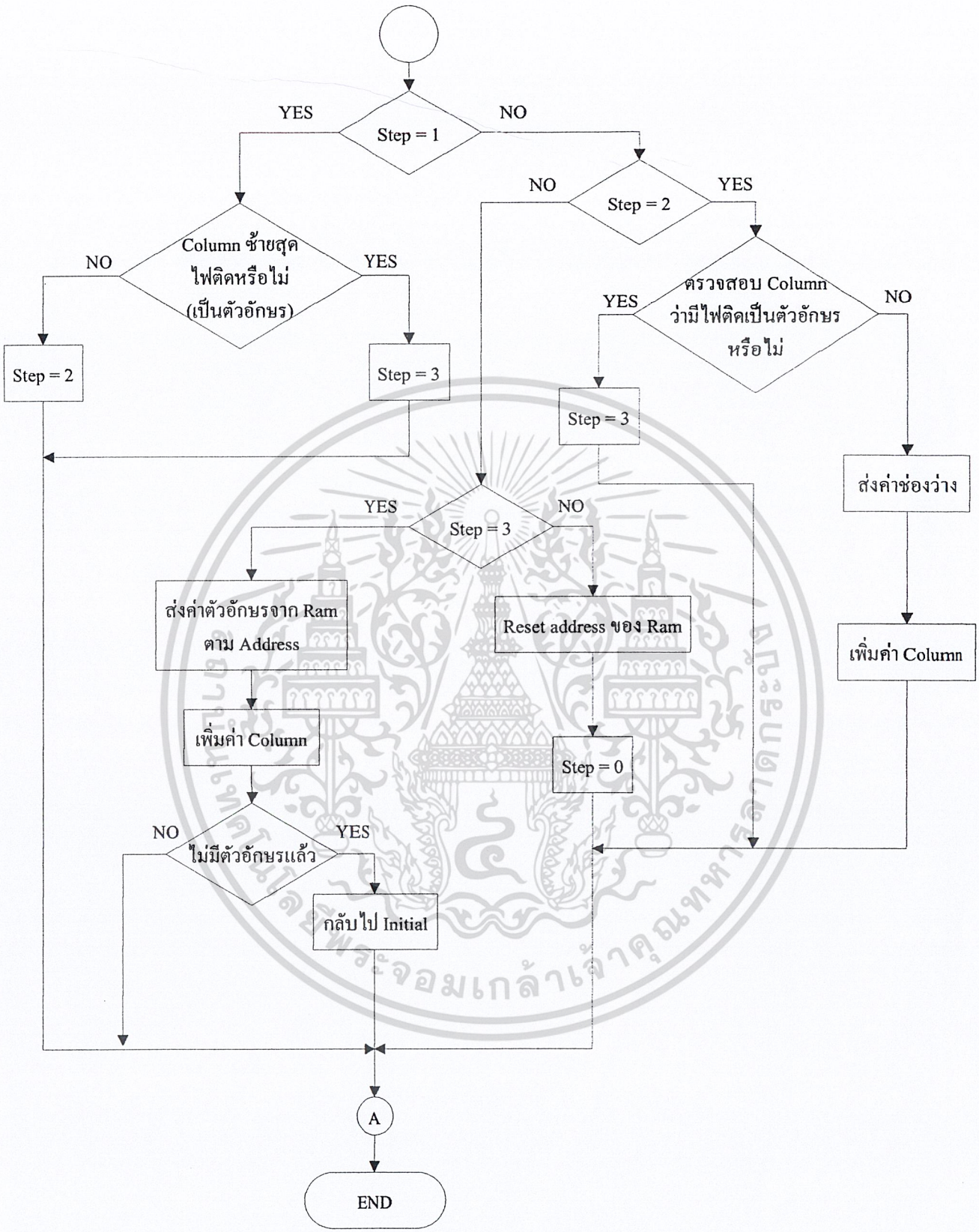
รูปที่ 5.13 แสดงโฟว์ชาร์ตของโปรแกรม RAM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



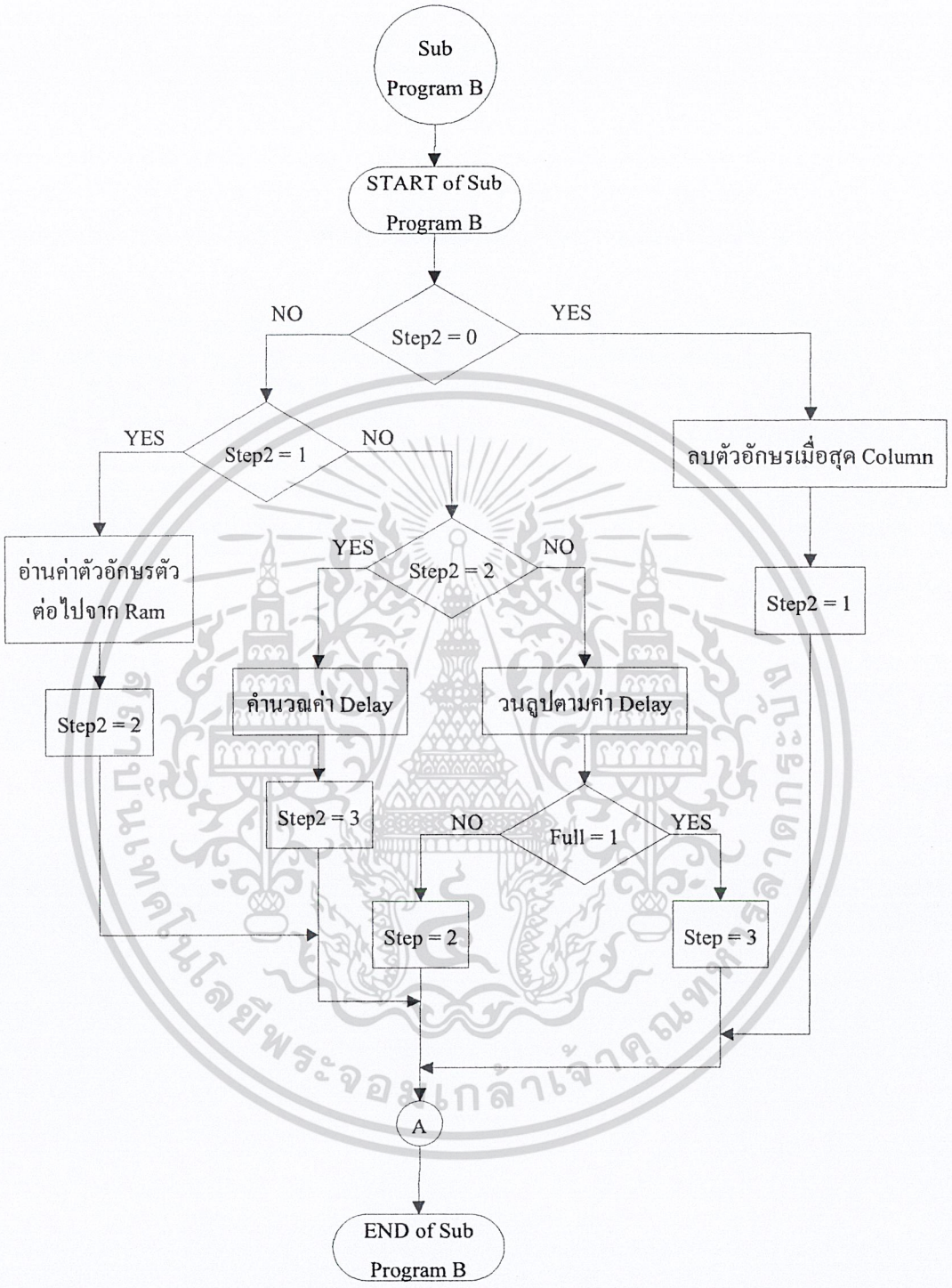
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 5.14 แสดงโฟว์ชาร์ตของโปรแกรม MODE



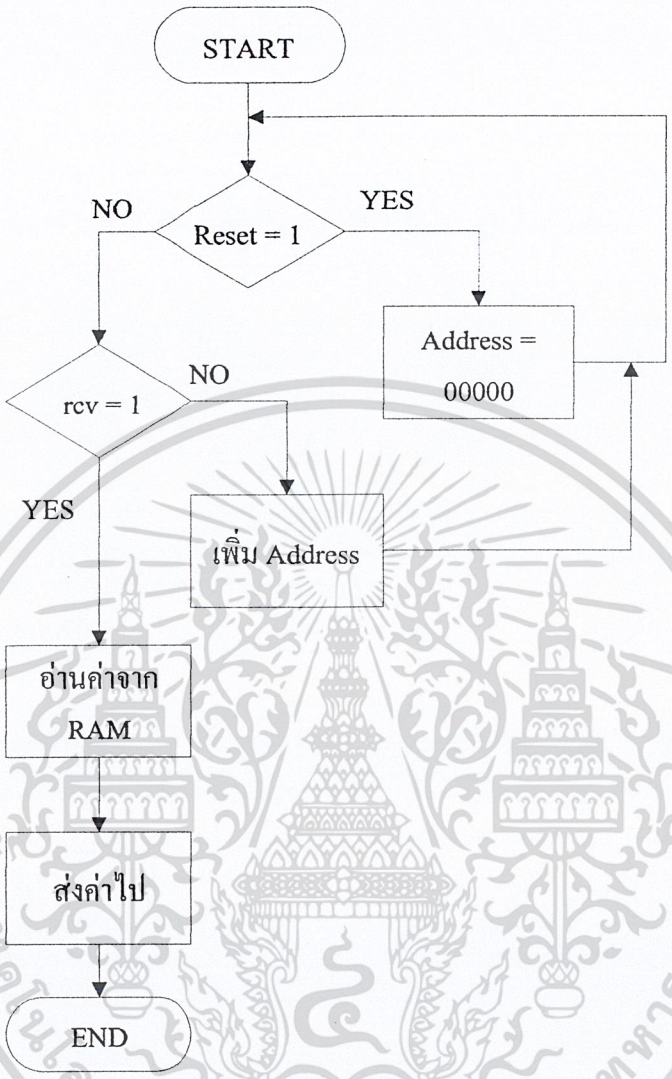
รูปที่ 5.14.1 แสดงโฟว์ชาร์ตของโปรแกรม MODE (ต่อ)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.14.2 แสดงโฟว์ชาร์ตของโปรแกรม MODE (Sub Program B)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

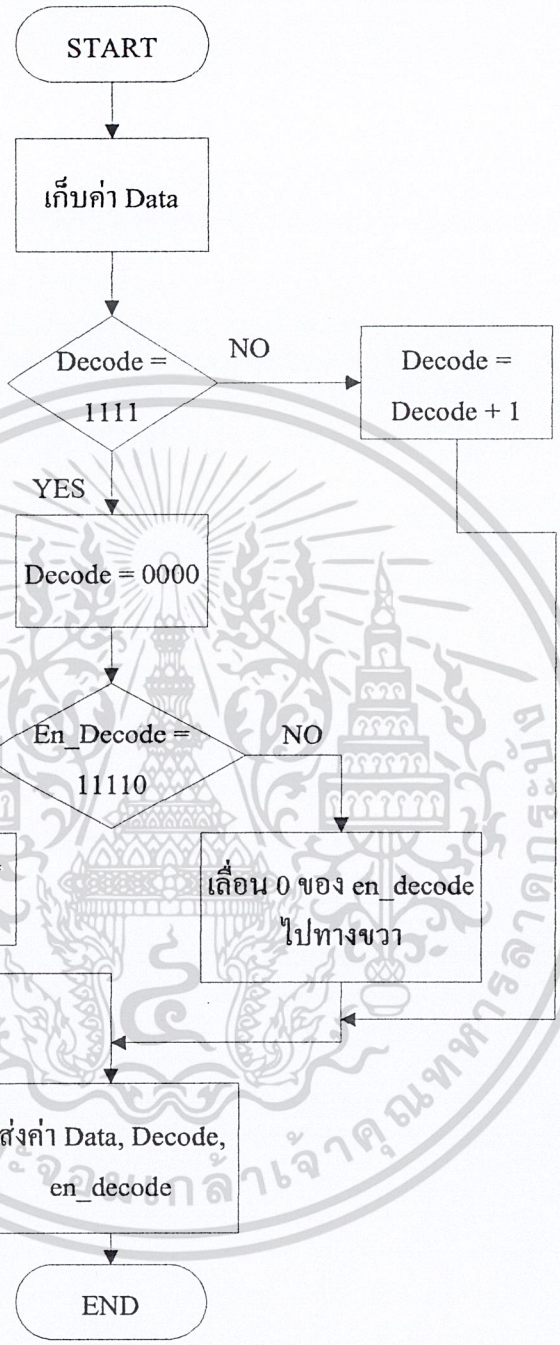


รูปที่ 5.15 แสดงโฟว์ชาร์ตของโปรแกรม RCV_DATA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 5.16 แสดงโฟว์ชาร์ตของโปรแกรม DECODE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

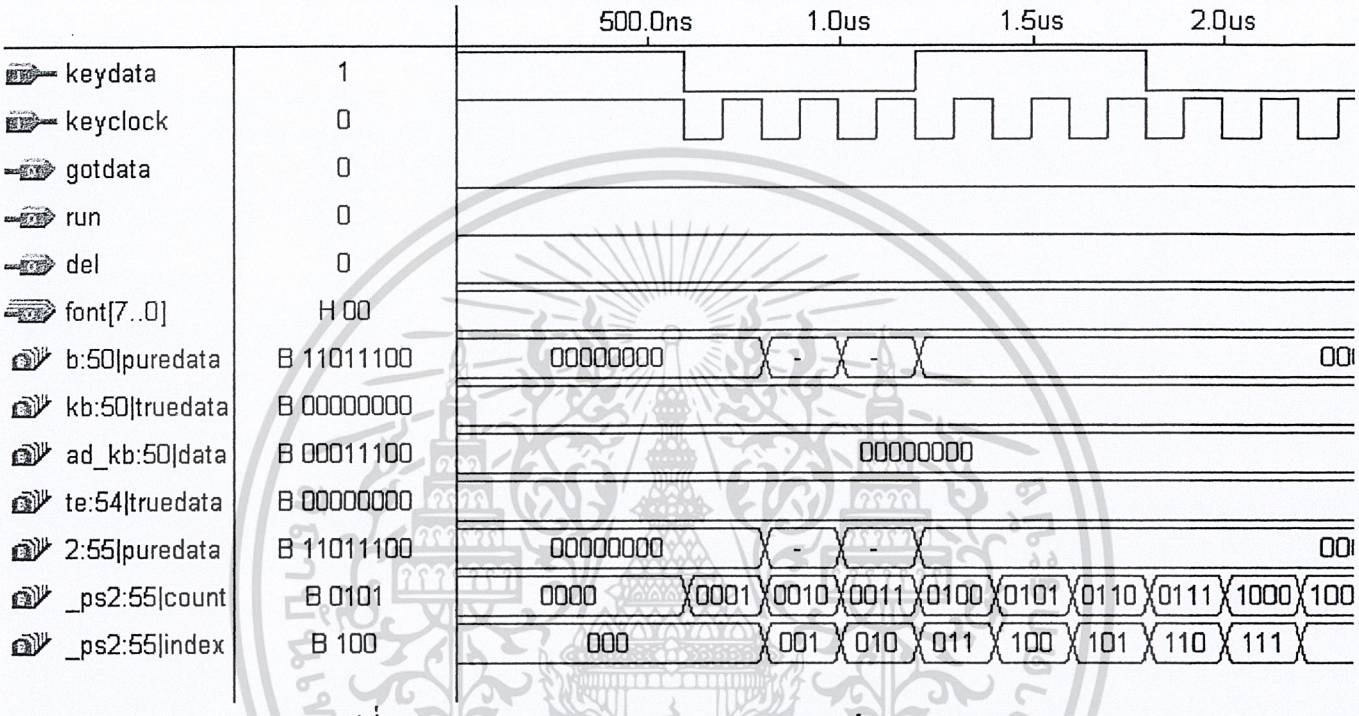


รูปที่ 5.17 แสดงโฟว์ชาร์ตของโปรแกรม DISPLAY

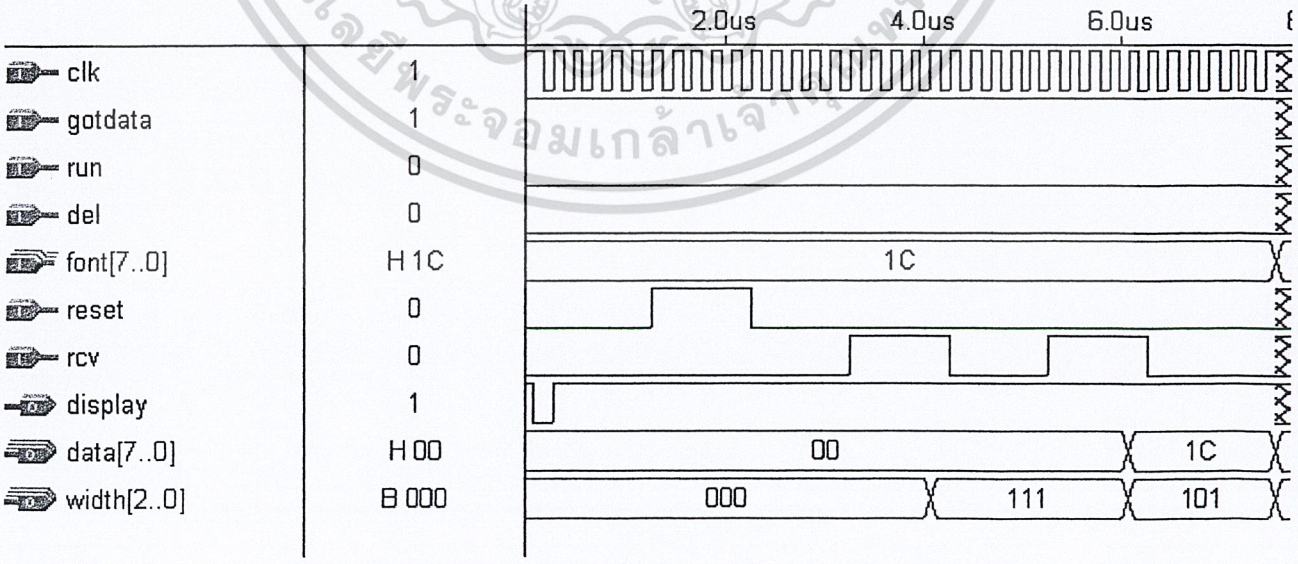
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.2 ผลการจำลองการทำงานของโปรแกรม

ในการจำลองการทำงานของโปรแกรม จะแบ่งเป็น 3 ส่วนใหญ่ๆ คือ การจำลองการทำงานของส่วน Input-Part, ส่วน Memory – Part และส่วน Output – Part ซึ่งแสดงได้ดังรูป

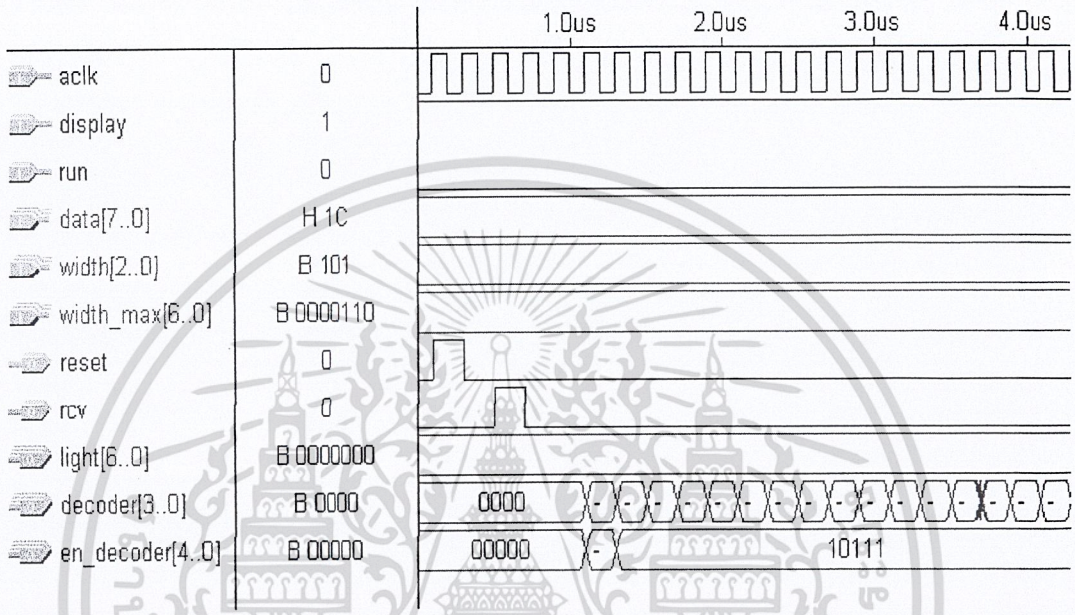


รูปที่ 5.18 แสดงผลการจำลองการทำงานของส่วน Input – Part



รูปที่ 5.19 แสดงผลการจำลองการทำงานของส่วน Memory – Part

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

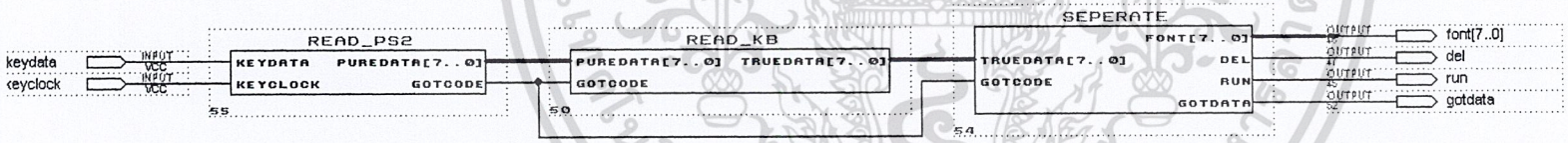


รูปที่ 5.20 แสดงผลการจำลองการทำงานของส่วน Output – Part

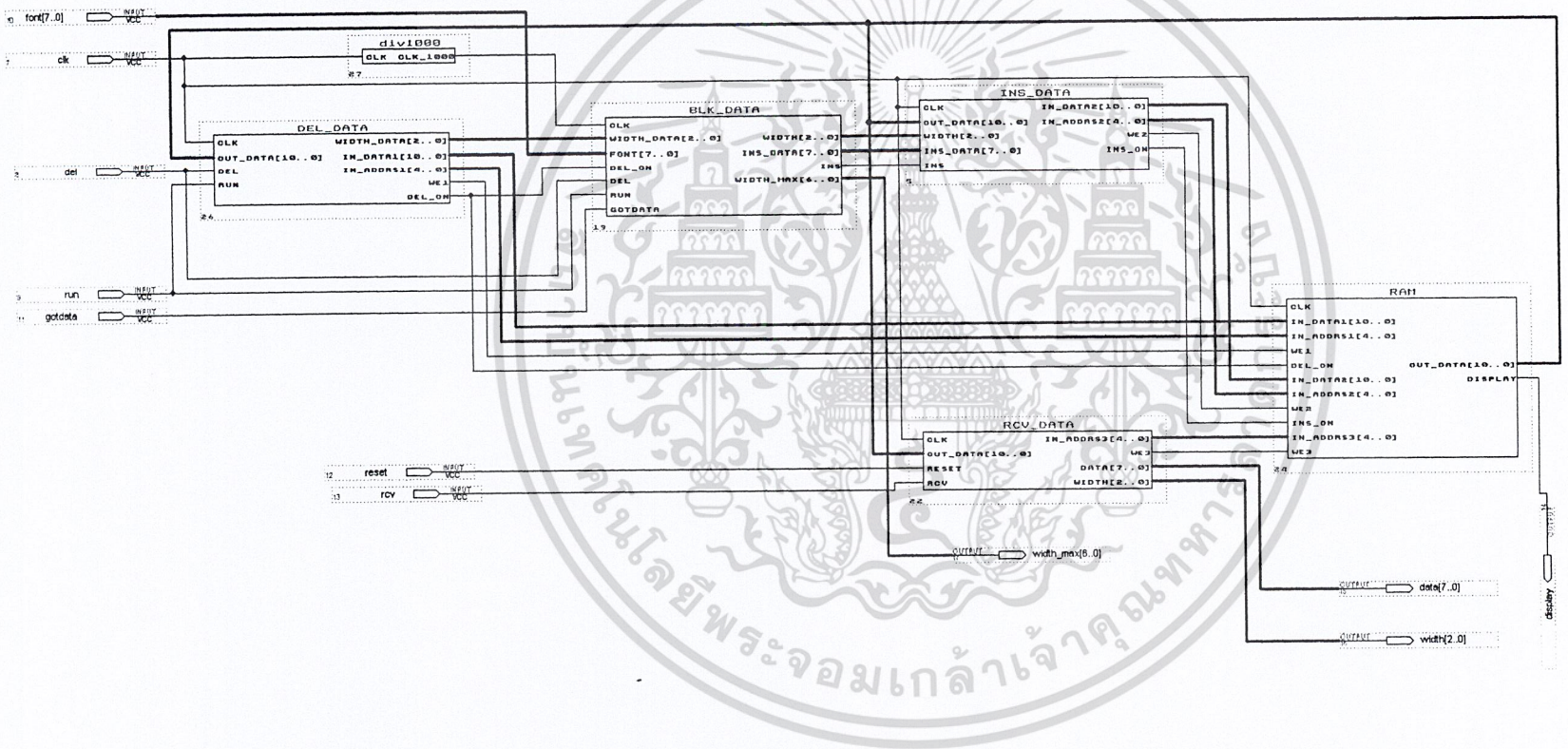
5.3 วงจรที่ได้จากการสังเคราะห์จากโปรแกรม

จากโปรแกรมที่ควบคุมการทำงานที่เขียนด้วยภาษาวีเอชดีแอล เราสามารถใช้ซอฟท์แวร์ทำการสังเคราะห์ลอจิกเกตที่มีการทำงานตามโปรแกรมที่เราออกแบบขึ้นมาได้ ในที่นี้จะแสดงเป็น 5 ส่วน คือ ส่วนของ ส่วนวงจร Input-Part, ส่วนวงจร Memory – Part , ส่วนวงจร Output – Part, ส่วนวงจร Div1000 และส่วนที่แสดงการเชื่อมต่อสัญญาณอินพุท เอาท์พุทของวงจรทั้งหมด ซึ่งแสดงได้ดังรูป

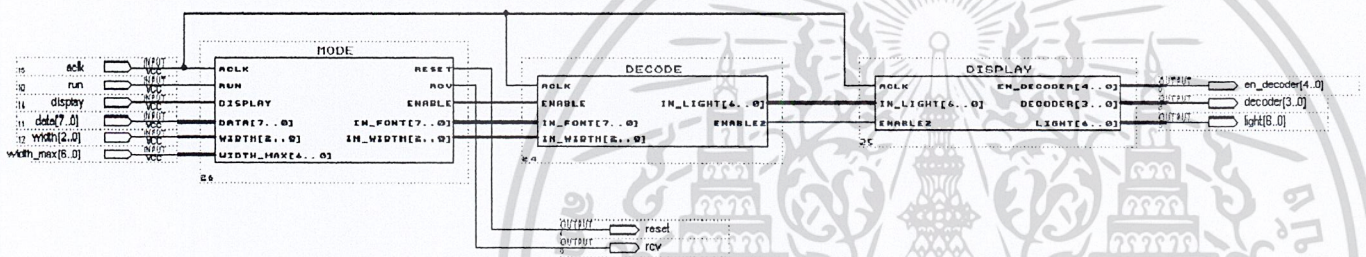
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.21 แสดงวงจรที่ได้จากการสังเคราะห์ของส่วน Input - Part



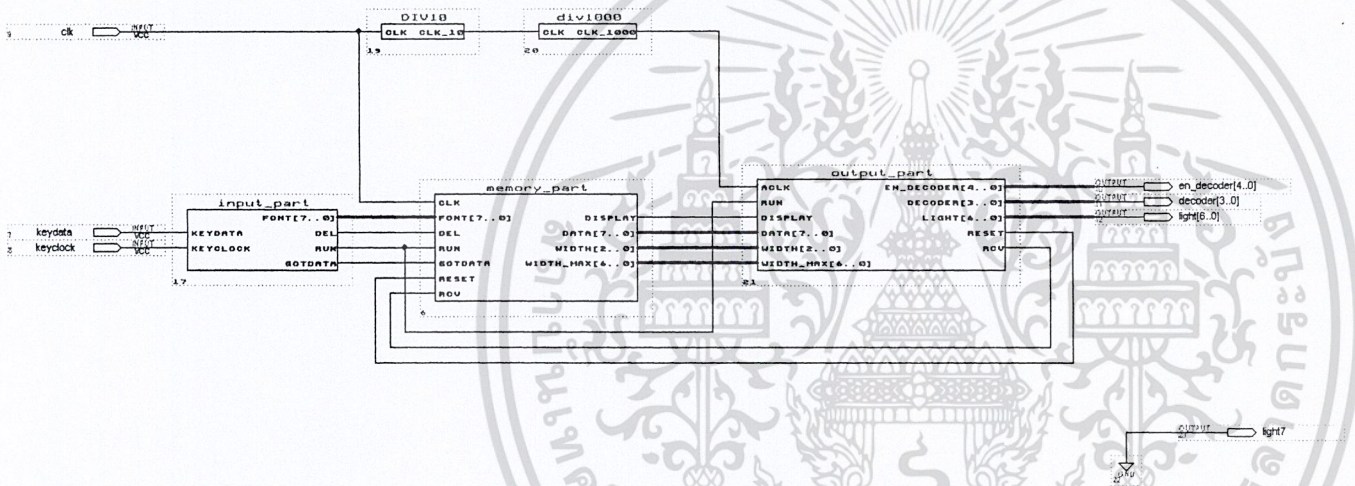
รูปที่ 5.22 แสดงวงจรที่ได้จากการสังเคราะห์ของส่วน Memory - Part



รูปที่ 5.23 แสดงวงจรที่ได้จากการสังเคราะห์ของส่วน Output - Part

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

รูปที่ 5.24 แสดงวงจรที่ได้จากการสังเคราะห์ของส่วน Div 1000



รูปที่ 5.25 แสดงวงจรที่ได้จากการสังเคราะห์ของวงจรทั้งหมด

บทที่ 6
บทสรุปและวิจารณ์

6.1 ข้อจำกัดที่พบในการทำโครงงาน

ในการออกแบบวงจรและการเขียนโปรแกรมควบคุมการทำงานโดยใช้เอฟพีจีเอ ทางผู้จัดทำได้พบข้อจำจัดบางประการ แบ่งได้ 2 ประเภท คือ ข้อจำจัดทางด้านฮาร์ดแวร์และข้อจำกัดทางด้านซอฟท์แวร์ โดยมีรายละเอียดดังต่อไปนี้

6.1.1 ข้อจำกัดด้านฮาร์ดแวร์

ในที่นี้หมายถึง ข้อจำกัดของการออกแบบและสร้างวงจรของบอร์ดแสดงผลแบบดอทเมตริกซ์ พบว่าในการทดสอบวงจรขั้นต้น ดอมเมตริกซ์ที่ใช้มีความสว่างน้อยเกินไป ทางผู้จัดทำได้ทำการแก้ไขโดยการปรับปรุงส่วนวงจรที่ใช้ควบคุมการจ่ายกระแสให้กับดอทเมตริกซ์ใหม่ ผลที่ได้ปรากฏว่า ดอทเมตริกซ์ติดไฟและสว่างมากกว่าเดิม เป็นที่น่าพอใจในระดับหนึ่งและสามารถนำไปใช้งานได้

6.1.2 ข้อจำกัดด้านซอฟท์แวร์

ในที่นี้หมายถึง ข้อจำกัดในการเขียนโปรแกรมภาษาวีเอชดีแอลเพื่อควบคุมการทำงานของวงจรและข้อจำกัดของ FPGA จากที่ได้ตั้งเป้าหมายไว้ว่าในภาคเรียนการศึกษานี้ จะพัฒนาโปรแกรมให้สามารถแสดงผลได้ทุกตัวอักษรที่มีอยู่ในคีย์บอร์ด พบว่าชิฟเอฟพีจีเอ EPF10K20TC144-4 ที่ใช้มีจำนวนลอจิกเกตไม่เพียงพอสำหรับโปรแกรมที่เขียนขึ้นมา จึงต้องทำการปรับปรุงโปรแกรมให้สามารถแสดงผลได้เฉพาะตัวอักษรภาษาอังกฤษตัวพิมพ์ใหญ่ A-Z เท่านั้น เนื่องจากวิธีนี้จะช่วยประหยัดงบประมาณได้ เพราะถ้าเปลี่ยนไปใช้ FPGA ที่มีความจุลอจิกเกตมากๆ ราคาของ FPGA จะสูงมากขึ้นด้วย ผลที่ได้ปรากฏว่าใช้ลอจิกเซลล์ของ FPGA ไปกว่า 90 กว่าเปอร์เซ็นต์ ซึ่งก็กลายเป็นข้อจำกัดต่อมาสำหรับการพัฒนาปรับปรุงโปรแกรมในเวลาต่อมา เพราะมีลอจิกเกตเหลืออยู่ในจำนวนจำกัด จะทำการเปลี่ยนแปลงได้ไม่มากนัก นอกจากนี้ชิฟเอฟพีจีเอที่ใช้ไม่สามารถเก็บโปรแกรมในภาวะที่ไม่มีไฟเลี้ยงได้ ในการทดสอบการทำงานของวงจรจะต้องโปรแกรมชิฟก่อนทุกครั้ง อีกทั้งFPGA ที่มีความจุเกตสูงๆ จะมีราคาที่สูงมาก จึงยังไม่เหมาะสมที่จะนำมาใช้ในเชิงพานิชย์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.2 บทสรุป

ในการทำโครงงานเรื่องบอร์ดแสดงผลตัวอักษรแบบดอทเมตริกซ์ควบคุมด้วยเอฟพีจีเอ สำหรับเทอมนี้จะเป็นการนำเอาความรู้ที่ได้ศึกษามาจากเทอมที่แล้วมาสร้างวงจรและเขียนโปรแกรมควบคุมการทำงานให้สามารถทำงานได้จริง โดยตัวซอฟท์แวร์ในการทำเอฟพีจีเอก็คือโปรแกรม Max+plusII version 10.0 BASELINE ของบริษัท Altera ความรู้ภาษาวีเอชดีแอลเพื่อใช้อธิบายพฤติกรรมของฮาร์ดแวร์และศึกษาทฤษฎีต่างๆที่เกี่ยวข้องกับโครงงาน เช่น การรับส่งข้อมูลผ่านคีย์บอร์ด ความรู้ทั่วไปเกี่ยวกับวงจรทางดิจิตอล ความรู้ทั่วไปเกี่ยวกับคีย์บอร์ดแบบ PS/2 ที่ใช้ในโครงงาน ซึ่งรายละเอียดของทฤษฎีต่างๆ ที่ได้ศึกษามาได้กล่าวถึงในปริญญานิพนธ์นี้แล้ว โดยบอร์ดแสดงผลที่ได้มีคุณสมบัติดังนี้ คือ สามารถแสดงตัวอักษรได้ 2 แบบทั้งแบบหยุดนิ่งกับที่และแบบตัววิ่งโดยจะวิ่งจากขวาไปซ้ายโดยการกดแป้น Enter หลังจากการกดตัวอักษรแล้ว และการกดแป้น Esc จะทำให้ตัวอักษรหยุดวิ่ง และกับผลการจำลองการทำงานที่มีผลน่าพอใจ ส่วนตัวโปรแกรม (source code)ได้อยู่ในภาคผนวก ก

และจากโครงงานนี้ที่ได้มีการออกแบบ สร้างวงจรด้วย FPGA นี้พบว่า โครงสร้างของ FPGA จะทำงานได้อย่างมีประสิทธิภาพในวงจรที่เกี่ยวกับการจัดการข้อมูล (Data-Path Circuit) มากกว่าวงจรที่เกี่ยวกับการควบคุม (Control Logic Circuit) เนื่องจากซอฟท์แวร์ที่ใช้ในการจำลองการทำงานของวงจรที่สร้างบน FPGA ไม่สามารถจำลองการทำงานของไมโครคอนโทรลเลอร์ได้เช่นเดียวกับซอฟท์แวร์ที่ใช้พัฒนาและจำลองการทำงานของไมโครคอนโทรลเลอร์โดยตรง อีกทั้งการออกแบบด้วย FPGA ที่มีวงจรขนาดใหญ่ มีความซับซ้อนและใช้จำนวนลอจิกเกตมาก จะใช้เวลานานในกว่าที่การออกแบบจะเสร็จเรียบร้อย เนื่องจากการจำลองการทำงานและการหาข้อบกพร่องของวงจร (Simulation & Debugging) จะต้องใช้เวลาพอสมควร แต่ FPGA ก็มีข้อดีในการแก้ไขโปรแกรมหรือการทำงานของวงจรที่สามารถทำได้บ่อยครั้งเท่าที่ต้องการ เพื่อให้เกิดประสิทธิภาพสูงสุด ทำให้การออกแบบพัฒนาและการทดสอบสามารถทำได้ง่าย เนื่องจากเราสามารถโปรแกรมได้เองและสามารถโปรแกรมลงชิฟได้หลายๆ ครั้ง ทั้งยังมีความเร็วในการทำงานที่ได้เปรียบอย่างมากอีกด้วยเมื่อเทียบกับไมโครโปรเซสเซอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก

Source Code ของโปรแกรมควบคุมการทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก

Source Code ของโปรแกรมควบคุมการทำงาน

1. ส่วนโปรแกรมภาษาวีเอชดีแอลควบคุมการทำงาน

1.1 โปรแกรม READ_PS2

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity read_ps2 is
        port(keydata                    : in std_logic;
             keyclock                   : in std_logic;
             puredata                   : out std_logic_vector(7 downto 0);
             gotcode                    : out std_logic);
end;
architecture read_serial of read_ps2 is
signal   detect   :       std_logic;
begin
process (keyclock)
variable index    :       natural range 0 to 7 := 0;
variable count    :       natural range 0 to 10 := 0;
begin
        if falling_edge(keyclock) then
             if count = 10 then                          -- stop bit
                  count := 0;
                  gotcode <= '0';                        -- wait for next output
                  detect <= '0';
                  index := 0;
             elsif count = 9 then                        -- parity bit
                  count := count + 1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                gotcode <= '1';                          -- output appear
            elsif (detect = '0') and (keydata = '0') then    -- start bit
                count := count + 1;
                detect <= '1';
            elsif detect = '1' then                      -- data 8 bit
                puredata(index) <= keydata;
                count := count + 1;
                index := index + 1;
            end if;
        end if;
end process;
    end;
```

1.2 ส่วนโปรแกรม READ_KB

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity read_kb is
        port(puredata   : in std_logic_vector(7 downto 0);
             gotcode    : in std_logic;
             truedata   : out std_logic_vector(7 downto 0));
end;
architecture read_key of read_kb is
signal  breakcode       :       std_logic;
signal  data            :       std_logic_vector(7 downto 0);
begin
process (gotcode)
begin
        if rising_edge(gotcode) then
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            if puredata = "11110000" then                    -- check breakcode
                breakcode <= '1';
            elsif puredata = data then
                if breakcode = '1' then                      -- release key
                    truedata <= puredata;
                    data <= "00000000";
                    breakcode <= '0';
                end if;
            else                                             -- puredata <> data condition
                if breakcode = '0' then
                    if data = "00000000" then                -- press key
                        data <= puredata;
                    end if;
                else                                         -- handshake key
                    data <= "00000000";
                    breakcode <= '0';
                end if;
                truedata <= "00000000";
            end if;
        end if;
end process;
end;
```

1.3 ส่วนโปรแกรม SEPERATE

```
        library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity seperate is
        port(truedata      : in std_logic_vector(7 downto 0);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        gotcode     : in std_logic;
        font        : out std_logic_vector(7 downto 0);
        del         : out std_logic;
        run         : out std_logic;
        gotdata     : out std_logic);
end;
architecture seperate_key of seperate is
begin
process (gotcode)
begin
    if falling_edge(gotcode) then
        case truedata is        -- Bksp Enter Esc
    when "01100110" => font <= "00000000";gotdata <= '1';del <= '1'; -- enable del-process
    when "01011010" => font <= "00000000";gotdata <= '1';run <= '1'; -- enable run-process
    when "01110110" => font <= "00000000";gotdata <= '1';run <= '0'; -- disable run-
process
    -- Spacebar
        when "00101001" => font <= "00101001";gotdata <= '1';del <= '0';
    -- A-Z
            when "00011100" => font <= "00011100";gotdata <= '1';del <= '0';
            when "00110010" => font <= "00110010";gotdata <= '1';del <= '0';
            when "00100001" => font <= "00100001";gotdata <= '1';del <= '0';
            when "00100011" => font <= "00100011";gotdata <= '1';del <= '0';
            when "00100100" => font <= "00100100";gotdata <= '1';del <= '0';
            when "00101011" => font <= "00101011";gotdata <= '1';del <= '0';
            when "00110100" => font <= "00110100";gotdata <= '1';del <= '0';
            when "00110011" => font <= "00110011";gotdata <= '1';del <= '0';
            when "01000011" => font <= "01000011";gotdata <= '1';del <= '0';
            when "00111011" => font <= "00111011";gotdata <= '1';del <= '0';
            when "01000010" => font <= "01000010";gotdata <= '1';del <= '0';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                when "01001011" => font <= "01001011";gotdata <= '1';del <= '0';
                when "00111010" => font <= "00111010";gotdata <= '1';del <= '0';
                when "00110001" => font <= "00110001";gotdata <= '1';del <= '0';
                when "01000100" => font <= "01000100";gotdata <= '1';del <= '0';
                when "01001101" => font <= "01001101";gotdata <= '1';del <= '0';
                when "00010101" => font <= "00010101";gotdata <= '1';del <= '0';
                when "00101101" => font <= "00101101";gotdata <= '1';del <= '0';
                when "00011011" => font <= "00011011";gotdata <= '1';del <= '0';
                when "00101100" => font <= "00101100";gotdata <= '1';del <= '0';
                when "00111100" => font <= "00111100";gotdata <= '1';del <= '0';
                when "00101010" => font <= "00101010";gotdata <= '1';del <= '0';
                when "00011101" => font <= "00011101";gotdata <= '1';del <= '0';
                when "00100010" => font <= "00100010";gotdata <= '1';del <= '0';
                when "00110101" => font <= "00110101";gotdata <= '1';del <= '0';
                when "00011010" => font <= "00011010";gotdata <= '1';del <= '0';
                when others => font <= "00000000";gotdata <= '0';del <= '0';
        -- reset when "00000000"
                end case;
        end if;
end process;
    end;
```

1.4 ส่วนโปรแกรม DEL_DATA

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity del_data is
        port(clk        : in std_logic;
             out_data   : in std_logic_vector(10 downto 0);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        del         : in std_logic;
        run         : in std_logic;
        width_data  : out std_logic_vector(2 downto 0);
        in_data1    : out std_logic_vector(10 downto 0);
        in_addrs1   : out std_logic_vector(4 downto 0);
        we1         : out std_logic;
        del_on      : out std_logic);
end;
architecture delete_data of del_data is
signal  del_done    :   std_logic;
signal  addrs       :   std_logic_vector(4 downto 0) := "10010";
signal  data        :   std_logic_vector(10 downto 0);
begin
process (clk)
variable step       :   natural range 0 to 3;
variable delay      :   natural range 0 to 3;
begin
    if rising_edge(clk) then
    if (del = '1') and (run = '0') and (del_done = '0') then
-- delete signal appear and no run signal
                if step = 0 then                    -- read data from memory
                    if delay = 3 then               -- delay from transfering data
                        data <= out_data;
                        step := 1;
                        delay := 0;
                    else
                        del_on <= '1';              -- enable memory
                        we1 <= '0';
                        in_addrs1 <= addrs;
                        delay := delay + 1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                end if;
            elsif step = 1 then                    -- check have data in this address
                if data = "00000000000" then
                    if addrs = "00000" then
                        step := 2;
                    else
                        addrs <= addrs - '1';      -- decrease address
                        step := 0;
                    end if;
                else
                    step := 2;
                end if;
            elsif step = 2 then                    -- found recent data has pressed
width:      for i in 0 to 2 loop                   -- transfer widths 3 bit
                    width_data(i) <= data(i);
                end loop width;
                in_addrs1 <= addrs;
                in_data1 <= "00000000000";         -- delete recent data
                we1 <= '1';
                step := 3;
            else                                   -- disable memory
                del_done <= '1';
                del_on <= '0';
            end if;
        elsif del = '0' then                       -- no delete signal(reset)
            addrs <= "10010";
            del_done <= '0';
            step := 0;
        end if;
    end if;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
end process;
    end;
```

1.5 ส่วนโปรแกรม BLK_DATA

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity blk_data is
        port(clk        : in std_logic;
        width_data      : in std_logic_vector(2 downto 0);
        font            : in std_logic_vector(7 downto 0);
        del_on          : in std_logic;
        del             : in std_logic;
        run             : in std_logic;
        gotdata         : in std_logic;
        width           : out std_logic_vector(2 downto 0);
        ins_data        : out std_logic_vector(7 downto 0);
        ins             : out std_logic;
        width_max       : out std_logic_vector(6 downto 0));
end;
architecture block_data of blk_data is
signal  got_done        :       std_logic;
signal  delete          :       std_logic;
signal  add_width       :       std_logic_vector(2 downto 0);
signal  width_now       :       std_logic_vector(6 downto 0);
begin
process (clk)
variable step   :       natural range 0 to 1;
begin
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
if rising_edge(clk) then
        if (del_on = '0') and (delete = '1') then                -- have delete process
                if step = 1 then                                 -- sending new widths
                        width_max <= width_now;
                        delete <= '0';
                        got_done <= '1';
                        step := 0;
                else                                             -- change widths
                        width_now <= width_now - width_data;
                        step := 1;
                end if;
        elsif (gotdata = '1') and (run = '0') and (got_done = '0') then
-- having data and no run signal
                if del = '1' then                                -- check delete signal
                delete <= '1';
                got_done <= '1';
                elsif step = 0 then                              -- read data
                if font = "00000000" then                        -- no data to insert
                        add_width <= "000";
                        got_done <= '1';
                elsif font = "01000011" then                     -- data has 3 columns
                        add_width <= "100";
                        width_now <= width_now + "100";
                elsif (font = "00111011") or (font = "00011011") then
                        -- data has 4 columns
                        add_width <= "101";
                        width_now <= width_now + "101";
                else                                             -- data has 5 columns
                        add_width <= "110";
                        width_now <= width_now + "110";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                end if;
                step := 1;
            else                                    -- pass data
                if width_now > "1010001" then
                                                    -- widths is over
                    width_now <= width_now - add_width;
                else                                -- widths is okay
                    width <= add_width;
                    ins_data <= font;
                    width_max <= width_now;-- sending widths in now
                    ins <= '1';                     -- enable insert process
                end if;
                got_done <= '1';
            end if;
        elsif gotdata = '0' then                    -- no data come
            ins <= '0';
            got_done <= '0';
            step := 0;
        end if;
    end if;
end process;
    end;
```

1.6 ส่วนโปรแกรม INS_DATA

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity ins_data is
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        port(clk        : in std_logic;
             out_data   : in std_logic_vector(10 downto 0);
             width      : in std_logic_vector(2 downto 0);
             ins_data   : in std_logic_vector(7 downto 0);
             ins        : in std_logic;
             in_data2   : out std_logic_vector(10 downto 0);
             in_addrs2  : out std_logic_vector(4 downto 0);
             we2        : out std_logic;
             ins_on     : out std_logic);
end;
architecture insert_data of ins_data is
signal  ins_done    :   std_logic;
signal  addrs       :   std_logic_vector(4 downto 0);
signal  data        :   std_logic_vector(10 downto 0);
begin
process (clk)
variable step       :   natural range 0 to 3;
variable delay      :   natural range 0 to 3;
begin
        if rising_edge(clk) then
            if (ins = '1') and (ins_done = '0') then        -- insert signal appear
                if step = 0 then                            -- read data from memory
                    if delay = 3 then                       -- delay from transfering data
                        data <= out_data;
                        step := 1;
                        delay := 0;
                    else
                        ins_on <= '1';                      -- enable memory
                        we2 <= '0';
                        in_addrs2 <= addrs;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                delay := delay + 1;
            end if;
        elsif step = 1 then                    -- check non-data in this address
            if data = "00000000000" then
                step := 2;
            else
                addrs <= addrs + '1';          -- increase address
                step := 0;
            end if;
        elsif step = 2 then                    -- write data in this address
            in_addrs2 <= addrs;
            in_data2 <= ins_data & width;
-- write data 8 bit and its width 3 bit
            we2 <= '1';
            step := 3;
        else                                   -- disable memory
            ins_done <= '1';
            ins_on <= '0';
        end if;
    elsif ins = '0' then                       -- no insert signal(reset)
        addrs <= "00000";
        ins_done <= '0';
        step := 0;
    end if;
  end if;
end process;
  end;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.7 ส่วนโปรแกรม RAM

```
library ieee;
use ieee.std_logic_1164.all;
package ram_constants is
  constant data_width : integer := 11;
  constant addr_width : integer := 5;
end ram_constants;
library ieee;
use ieee.std_logic_1164.all;
library lpm;
use lpm.lpm_components.all;
entity ram is
   port(clk                    : in std_logic;
                in_data1       : in std_logic_vector(10 downto 0);
                in_addrs1      : in std_logic_vector(4 downto 0);
                we1            : in std_logic;
                del_on         : in std_logic;
                in_data2       : in std_logic_vector(10 downto 0);
                in_addrs2      : in std_logic_vector(4 downto 0);
                we2            : in std_logic;
                ins_on         : in std_logic;
                in_addrs3      : in std_logic_vector(4 downto 0);
                we3            : in std_logic;
                out_data       : out std_logic_vector(10 downto 0);
                display        : out std_logic);
end;
architecture memory of ram is
signal   in_data        :       std_logic_vector(10 downto 0);
signal   in_addrs       :       std_logic_vector(4 downto 0);
signal   we             :       std_logic;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
begin
  inst_1: lpm_ram_dq
    generic map (lpm_widthad => 5,lpm_width => 11,lpm_outdata => "unregistered")
    port map (data => in_data, address => in_addrs, we => we,
      inclock => clk, q => out_data);
process(clk)
begin
        if rising_edge(clk) then
                if del_on = '1' then                    -- enable memory to delete data
                        display <= '0';                 -- disable display part
                        in_data <= in_data1;
                        in_addrs <= in_addrs1;
                        we <= we1;
                elsif ins_on = '1' then                 -- enable memory to insert data
                        display <= '0';                 -- disable display part
                        in_data <= in_data2;
                        in_addrs <= in_addrs2;
                        we <= we2;
                else                                    -- enable memory to receive data
                        display <= '1';                 -- enable display part
                        in_addrs <= in_addrs3;
                        we <= we3;
                end if;
        end if;
end process;
end;
```

1.8 ส่วนโปรแกรม MODE

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

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity mode is
        port(aclk           : in std_logic;
             run            : in std_logic;
             display        : in std_logic;
             data           : in std_logic_vector(7 downto 0);
             width          : in std_logic_vector(2 downto 0);
             width_max      : in std_logic_vector(6 downto 0);
             reset          : out std_logic;
             rcv            : out std_logic;
             enable         : out std_logic;
             in_font        : out std_logic_vector(7 downto 0);
             in_width       : out std_logic_vector(2 downto 0));
end;
architecture display_mode of mode is
signal  initial  :      std_logic;      -- check for arriving in the first column for each word
signal  empty    :      std_logic;      -- check for end of data arriving in the last column
signal  full     :      std_logic;      -- check for arriving in first column
signal  rcv1     :      std_logic;
signal  first    :      std_logic;
signal  widths   :      std_logic_vector(6 downto 0);       -- width of space
signal  count    :      std_logic_vector(6 downto 0);       -- count width of space
signal  width0   :      std_logic_vector(2 downto 0);       -- count width of data
signal  width1   :      std_logic_vector(2 downto 0);
signal  cwidth   :      std_logic_vector(2 downto 0);
signal  cwidth2  :      std_logic_vector(2 downto 0);
signal  max_width  :    std_logic_vector(6 downto 0);       -- maximum width of output
signal  max_term   :    std_logic_vector(4 downto 0);       -- number of data in memory
signal  term     :      std_logic_vector(4 downto 0);   -- count number of data in memory
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
signal   term1   :        std_logic_vector(4 downto 0);
signal   run_clk :        std_logic_vector(1 downto 0);
signal   run_clk2    :    std_logic_vector(5 downto 0);
signal   run_clk3    :    std_logic_vector(5 downto 0);
signal   run_clk4    :    std_logic_vector(5 downto 0);
begin
process (aclk)
variable step    :        natural range 0 to 4;
variable step2   :        natural range 0 to 3;
variable delay   :        natural range 0 to 2;
begin
        if rising_edge(aclk) then
            if display = '1' then                       -- display signal appear
                if (run ='1') and (initial = '1') then  -- initial case for dynamic mode
                    widths <= "1010000";    -- (determine initial value)
                    reset <= '1';
                    rcv <= '0';
                    rcv1 <= '1';
                    enable <= '0';
                    initial <= '0';
                    empty <= '0';
                    full <= '0';
                    count <= "0000000";
                    width0 <= "000";
                    width1 <= "000";
                    cwidth <= cwidth2;
                    max_width <= "0000000";
                    term <= "00000";
                    term1 <= "00000";
                    run_clk <= "00";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                run_clk2 <= "000000";
                run_clk3 <= "000000";
                run_clk4 <= "000000";
                step := 0;
                step2 := 0;
                delay := 0;
        elsif (run = '0') and (initial = '0') then        -- initial case for static mode
                widths <= "0000000";    -- (determine initial value)
                reset <= '1';
                rcv <= '0';
                rcv1 <= '0';
                enable <= '0';
                initial <= '1';
                empty <= '0';
                full <= '0';
                first <= '0';
                count <= "0000000";
                width0 <= "000";
                width1 <= "000";
                cwidth <= "000";
                cwidth2 <= "000";
                max_width <= "0000000";
                term <= "00000";
                term1 <= "00000";
                run_clk <= "00";
                run_clk2 <= "000000";
                run_clk3 <= "000000";
                run_clk4 <= "000000";
                step := 0;
                step2 := 0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            delay := 0;
        elsif max_width = "1010001" then   -- case for last column in output
            if run = '1' then
                    if widths = "0000000" then
                    else       widths <= widths - '1';
                    end if;
            else  max_term <= term - '1';
            end if;
            in_font <= "00000000";
            reset <= '1';
            rcv1 <= '0';
            width0 <= "000";
            max_width <= "0000000";
            enable <= '0';
            empty <= '0';
            full <= '0';
            term <= "00000";
            run_clk2 <= "000000";
            run_clk3 <= "000000";
            step := 0;
            step2 := 0;
            delay := 0;
        elsif step = 0 then                    -- the first step
            if run = '0' then   widths <= "1010001" - width_max;
            end if;
            reset <= '0';
            step := 1;
        elsif step = 1 then            -- the second step(check the first column)
            if widths = "0000000" then -- data for the first column in output
                    if delay = 0 then
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    rcv <= '1';
                    delay := 1;
                elsif delay = 1 then
                    rcv <= '0';
                    delay := 2;
                else
                    rcv <= '1';
                    if run = '1' then
                        full <= '1';
                        step := 4;  -- move to special step
                        step2 := 0;
                    else    step := 3;
                    end if;   delay := 0;
                end if;
            else                    -- space for the first column in output
                if run = '0' then
widths <= widths - 1;
                    step := 2;
                else    step := 4;      -- move to special step
                    step2 := 3;
                    run_clk2 <= "101100";
                end if;
                rcv <= '1';
            end if;
        elsif step = 4 then             -- special step for dyanamic mode
            if step2 = 0 then
              rcv <= '0';  step2 := 1;
               if run_clk = "10" then
                  if width1 = cwidth then
                                        -- check for equal to the last width of data
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    if term1 = max_term then
                -- check for equal to the last word
                        initial <= '1';
                    else
                -- case for space(between words) arriving
                        run_clk4 <= run_clk4 + "10";
                        width1 <= "000";
                        term1 <= term1 + '1';
                        first <= '0';
                    end if;
                else        width1 <= width1 + '1';
                end if;
            else    run_clk <= run_clk + '1';
            end if;
        elsif step2 = 1 then        -- case for ignore previous words
            if run_clk2 = run_clk4 then
                rcv <= '0';
                step2 := 2;
            else    rcv1 <= not(rcv1);
                rcv <= rcv1;
                run_clk2 <= run_clk2 + '1';
            end if;
        elsif step2 = 2 then            -- case for determine delay
            run_clk2 <= "101000" - run_clk4;
            step2 := 3;
        else                            -- delay time for equal cycle
            if run_clk3 = run_clk2 then
                if (full = '1') then
                    if run_clk = "01" then
                        enable <= '1';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                        in_font <= "00000000";
                                    end if;
                                    step := 3;
                                else    step := 2;
                                end if;
                            else    run_clk3 <= run_clk3 + '1';
                            end if;
                        end if;
                        width0 <= width1;
                        term <= term1;
                    elsif step = 2 then         -- the third step(sending space)
                        enable <= '1';      -- enable decode data
                        if count = widths then              -- next word is data
                            rcv <= '1';
                            in_font <= "00000000";
                            in_width <= width0;
                            max_width <= max_width + '1';
                            term <= term + '1';
                            step := 3;
                            count <= "0000000";
                        else                                -- next word is space
                            rcv <= '0';
                            in_font <= "00000000";
                            in_width <= width0;
                            max_width <= max_width + '1';
                            count <= count + '1';
                        end if;
                    else                                -- the fourth step(sending data)
                        enable <= '1';
                        if empty = '1' then
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                in_font <= "00000000";
                width0 <= "000";
            elsif (run = '1') and (max_width = "1010000") then
                in_font <= "00000000";
                width0 <= "000";
            elsif width0 = width then          -- send space between words
                if (run = '1') and (term = max_term) then
                    empty <= '1';
                end if;
                if first = '0' then
                    if run = '0' then
                        cwidth2 <= width;
                    else    cwidth <= width;
                    end if;  first <= '1';
                end if;
                rcv <= '1';
                in_font <= "00000000";
                width0 <= "000";
                term <= term + '1';
            else                               -- send data with its column
                rcv <= '0';
                in_font <= data;
                in_width <= width0;
                width0 <= width0 + '1';
            end if;  max_width <= max_width + '1';
        end if;
    else                                       -- no display signal(reset)
        widths <= "0000000";
        reset <= '1';
        rcv <= '0';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    rcv1 <= '0';
                    enable <= '0';
                    initial <= '0';
                    empty <= '0';
                    full <= '0';
                    count <= "0000000";
                    width0 <= "000";
                    width1 <= "000";
                    cwidth <= "000";
                    cwidth2 <= "000";
                    max_width <= "0000000";
                    term <= "00000";
                    term1 <= "00000";
                    run_clk <= "00";
                    run_clk2 <= "000000";
                    run_clk3 <= "000000";
                    run_clk4 <= "000000";
                    step := 0;
                    step2 := 0;
                    delay := 0;
                end if;
            end if;
        end process;
        end;
```

1.9 ส่วนโปรแกรม RCV_DATA

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

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
entity rcv_data is
        port(clk          : in std_logic;
             out_data     : in std_logic_vector(10 downto 0);
             reset        : in std_logic;
             rcv          : in std_logic;
             in_addrs3    : out std_logic_vector(4 downto 0);
             we3          : out std_logic;
             data         : out std_logic_vector(7 downto 0);
             width        : out std_logic_vector(2 downto 0));
end;
architecture recieve_data of rcv_data is
signal   rcv_done      :      std_logic;
signal   addrs         :      std_logic_vector(4 downto 0);
signal   data0         :      std_logic_vector(7 downto 0);
signal   width0        :      std_logic_vector(2 downto 0);
begin
process (clk)
variable step          :      natural range 0 to 1;
variable delay         :      natural range 0 to 3;
begin
        if rising_edge(clk) then
           if (rcv = '1') and (rcv_done = '0') and (reset = '0') then    -- receive signal appear
                        if step = 0 then                                 -- receive data from memory
                                we3 <= '0';
                                in_addrs3 <= addrs;
                                delay := delay + 1;
                                if delay = 3 then                        -- delay for transfering data
                                        step := 1;
                                        delay := 0;
                                end if;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                else                                    -- sending data
width_0:        for i in 0 to 2 loop                    -- transfer widths of data
                        width0(i) <= out_data(i);
                end loop width_0;
data_0:         for j in 3 to 10 loop                   -- transfer data
                        data0(j-3) <= out_data(j);
                end loop data_0;
                addrs <= addrs + '1';                   -- increase address
                rcv_done <= '1';
                data <= data0;
                width <= width0 - '1';
                end if;
        elsif reset = '1' then                          -- return to first address
                addrs <= "00000";
                rcv_done <= '0';
                delay := 0;
                step := 0;
        elsif rcv = '0' then                            -- no recieve signal(reset)
                rcv_done <= '0';
                delay := 0;
                step := 0;
        end if;
    end if;
end process;
end;
```

1.10 ส่วนโปรแกรม DIV10

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

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
use ieee.std_logic_unsigned.all;
entity div10 is
port      (clk      : in bit;
           clk_10 : out bit);
end;
architecture div10_clk of div10 is
begin
Process (clk)
variable count    : integer range 0 to 4 := 0;
variable change  : bit := '0';
begin
        if clk'event and clk = '1' then
                if count = 4 then
                        change := not(change);
                        count := 0;
                else      count := count + 1;
                end if;
                clk_10 <= change;
        end if;
end process;
end;
```

1.11 ส่วนโปรแกรม DECODE

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

entity decode is
      port(aclk    : in std_logic;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        enable      : in std_logic;
        in_font     : in std_logic_vector(7 downto 0);
        in_width    : in std_logic_vector(2 downto 0);
        in_light    : out std_logic_vector(6 downto 0);
        enable2     : out std_logic);
end;
architecture decode_data of decode is
begin
process(aclk)
begin
    if rising_edge(aclk) then
        if enable = '1' then                    -- enable signal appear
            case in_font is
                when "00011100" =>              -- A
                    case in_width is
                        when "000" => in_light <= "0111111";
                        when "001" => in_light <= "1001000";
                        when "010" => in_light <= "1001000";
                        when "011" => in_light <= "1001000";
                        when others => in_light <= "0111111";
                    end case;
                when "00110010" =>              -- B
                    case in_width is
                        when "000" => in_light <= "1111111";
                        when "001" => in_light <= "1001001";
                        when "010" => in_light <= "1001001";
                        when "011" => in_light <= "1001001";
                        when others => in_light <= "0110110";
                    end case;
                when "00100001" =>              -- C
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            case in_width is
                when "000" => in_light <= "0111110";
                when "001" => in_light <= "1000001";
                when "010" => in_light <= "1000001";
                when "011" => in_light <= "1000001";
                when others => in_light <= "0100010";
            end case;
        when "00100011" =>                  -- D
            case in_width is
                when "000" => in_light <= "1111111";
                when "001" => in_light <= "1000001";
                when "010" => in_light <= "1000001";
                when "011" => in_light <= "1000001";
                when others => in_light <= "0111110";
            end case;
        when "00100100" =>                  -- E
            case in_width is
                when "000" => in_light <= "1111111";
                when "001" => in_light <= "1010001";
                when "010" => in_light <= "1010001";
                when "011" => in_light <= "1010001";
                when others => in_light <= "1000001";
            end case;
        when "00101011" =>                  -- F
            case in_width is
                when "000" => in_light <= "1111111";
                when "001" => in_light <= "1010000";
                when "010" => in_light <= "1010000";
                when "011" => in_light <= "1010000";
                when others => in_light <= "1000000";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            end case;
        when "00110100" =>                  -- G
            case in_width is
                when "000" => in_light <= "0111110";
                when "001" => in_light <= "1000001";
                when "010" => in_light <= "1001001";
                when "011" => in_light <= "1001001";
                when others => in_light <= "0101110";
            end case;
        when "00110011" =>                  -- H
            case in_width is
                when "000" => in_light <= "1111111";
                when "001" => in_light <= "0001000";
                when "010" => in_light <= "0001000";
                when "011" => in_light <= "0001000";
                when others => in_light <= "1111111";
            end case;
        when "01000011" =>                  -- I
            case in_width is
                when "000" => in_light <= "1000001";
                when "001" => in_light <= "1111111";
                when others => in_light <= "1000001";
            end case;
        when "00111011" =>                  -- J
            case in_width is
                when "000" => in_light <= "0000010";
                when "001" => in_light <= "0000001";
                when "010" => in_light <= "0000001";
                when others => in_light <= "1111110";
            end case;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
when "01000010" =>                          -- K
        case in_width is
                when "000" => in_light <= "1111111";
                when "001" => in_light <= "0001000";
                when "010" => in_light <= "0010100";
                when "011" => in_light <= "0100010";
                when others => in_light <= "1000001";
        end case;
when "01001011" =>                          -- L
        case in_width is
                when "000" => in_light <= "1111111";
                when "001" => in_light <= "0000001";
                when "010" => in_light <= "0000001";
                when "011" => in_light <= "0000001";
                when others => in_light <= "0000001";
        end case;
when "00111010" =>                          -- M
        case in_width is
                when "000" => in_light <= "1111111";
                when "001" => in_light <= "0100000";
                when "010" => in_light <= "0011000";
                when "011" => in_light <= "0100000";
                when others => in_light <= "1111111";
        end case;
when "00110001" =>                          -- N
        case in_width is
                when "000" => in_light <= "1111111";
                when "001" => in_light <= "0010000";
                when "010" => in_light <= "0001000";
                when "011" => in_light <= "0000100";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                when others => in_light <= "1111111";
            end case;
        when "01000100" =>                  -- O
            case in_width is
                when "000" => in_light <= "0111110";
                when "001" => in_light <= "1000001";
                when "010" => in_light <= "1000001";
                when "011" => in_light <= "1000001";
                when others => in_light <= "0111110";
            end case;
        when "01001101" =>                  -- P
            case in_width is
                when "000" => in_light <= "1111111";
                when "001" => in_light <= "1001000";
                when "010" => in_light <= "1001000";
                when "011" => in_light <= "1001000";
                when others => in_light <= "0110000";
            end case;
        when "00010101" =>                  -- Q
            case in_width is
                when "000" => in_light <= "0111110";
                when "001" => in_light <= "1000001";
                when "010" => in_light <= "1000101";
                when "011" => in_light <= "1000010";
                when others => in_light <= "0111101";
            end case;
        when "00101101" =>                  -- R
            case in_width is
                when "000" => in_light <= "1111111";
                when "001" => in_light <= "1001000";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                when "010" => in_light <= "1001100";
                when "011" => in_light <= "1001010";
                when others => in_light <= "0110001";
            end case;
        when "00011011" =>                  -- S
            case in_width is
                when "000" => in_light <= "0110001";
                when "001" => in_light <= "1001001";
                when "010" => in_light <= "1001001";
                when others => in_light <= "1000110";
            end case;
        when "00101100" =>                  -- T
            case in_width is
                when "000" => in_light <= "1000000";
                when "001" => in_light <= "1000000";
                when "010" => in_light <= "1111111";
                when "011" => in_light <= "1000000";
                when others => in_light <= "1000000";
            end case;
        when "00111100" =>                  -- U
            case in_width is
                when "000" => in_light <= "1111110";
                when "001" => in_light <= "0000001";
                when "010" => in_light <= "0000001";
                when "011" => in_light <= "0000001";
                when others => in_light <= "1111110";
            end case;
        when "00101010" =>                  -- V
            case in_width is
                when "000" => in_light <= "1111100";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                when "001" => in_light <= "0000010";
                when "010" => in_light <= "0000001";
                when "011" => in_light <= "0000010";
                when others => in_light <= "1111100";
            end case;
        when "00011101" =>                  -- W
            case in_width is
                when "000" => in_light <= "1111110";
                when "001" => in_light <= "0000001";
                when "010" => in_light <= "0001110";
                when "011" => in_light <= "0000001";
                when others => in_light <= "1111110";
            end case;
        when "00100010" =>                  -- X
            case in_width is
                when "000" => in_light <= "1100011";
                when "001" => in_light <= "0010100";
                when "010" => in_light <= "0001000";
                when "011" => in_light <= "0010100";
                when others => in_light <= "1100011";
            end case;
        when "00110101" =>                  -- Y
            case in_width is
                when "000" => in_light <= "1110000";
                when "001" => in_light <= "0001000";
                when "010" => in_light <= "0000111";
                when "011" => in_light <= "0001000";
                when others => in_light <= "1110000";
            end case;
        when "00011010" =>                  -- Z
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    case in_width is
                        when "000" => in_light <= "1000011";
                        when "001" => in_light <= "1000101";
                        when "010" => in_light <= "1001001";
                        when "011" => in_light <= "1010001";
                        when others => in_light <= "1100001";
                    end case;
                when others =>
                    -- Space("00000000") and Spacebar("00101001")
                    in_light <= "0000000";
            end case;
            enable2 <= '1';                    -- enable display to output
        else                                   -- no enable signal(reset)
            in_light <= "0000000";
            enable2 <= '0';                    -- disable display to output
        end if;
    end if;
end process;
end;
```

1.12 ส่วนโปรแกรม DISPLAY

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity display is
    port(aclk       : in std_logic;
         in_light   : in std_logic_vector(6 downto 0);
         enable2    : in std_logic;
         en_decoder : out std_logic_vector(4 downto 0);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        decoder     : out std_logic_vector(3 downto 0);
        light       : out std_logic_vector(6 downto 0));
end;
architecture display_output of display is
signal  en_decoder0     : std_logic_vector(4 downto 0) := "01111";
signal  decoder0        : std_logic_vector(3 downto 0);
signal  light0          : std_logic_vector(6 downto 0);
begin
process(aclk)
begin
    if rising_edge(aclk) then
        if enable2 = '1' then                       -- enable signal appear
            light0 <= in_light;
            if decoder0 = "1111" then
                decoder0 <= "0000";
                case en_decoder0 is         -- shift dotmatrix
                    when "01111" => en_decoder0 <= "10111";
                    when "10111" => en_decoder0 <= "11011";
                    when "11011" => en_decoder0 <= "11101";
                    when "11101" => en_decoder0 <= "11110";
                    when others => en_decoder0 <= "01111";
                end case;
            else    decoder0 <= decoder0 + '1';     -- shift column of dotmatrix
            end if;
            en_decoder <= en_decoder0;
            decoder <= decoder0;
            light <= light0;
        else                                        -- no enable signal(reset)
            en_decoder0 <= "01111";
            decoder0 <= "1111";
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    light0 <= "0000000";
                    light <= "0000000";
                end if;
            end if;
        end process;
        end;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข

ดาต้าชีท (DATA SHEET)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Octal buffer/line driver; 3-state 74HC/HCT541

FEATURES

- Non-inverting outputs
- Output capability: bus driver
- I_{CC} category: MSI

GENERAL DESCRIPTION

The 74HC/HCT541 are high-speed Si-gate CMOS devices and are pin compatible with low power Schottky TTL (LSTTL). They are specified in compliance with JEDEC standard no. 7A.

The 74HC/HCT541 are octal non-inverting buffer/line drivers with 3-state outputs. The 3-state outputs are controlled by the output enable inputs $\overline{OE}_1$ and $\overline{OE}_2$. A HIGH on $\overline{OE}_n$ causes the outputs to assume a high impedance OFF-state.
The "541" is identical to the "540" but has non-inverting outputs.

QUICK REFERENCE DATA

GND = 0 V; T_{amb} = 25 °C; $t_r = t_f$ = 6 ns

SYMBOL	PARAMETER	CONDITIONS	TYPICAL		UNIT
			HC	HCT	
t_{PHL}/ t_{PLH}	propagation delay A_n to Y_n	C_L = 15 pF; V_{CC} = 5 V	10	12	ns
C_I	input capacitance		3.5	3.5	pF
C_{PD}	power dissipation capacitance per buffer	notes 1 and 2	37	39	pF

Notes

1. C_{PD} is used to determine the dynamic power dissipation (P_D in μW):
   $P_D = C_{PD} \times V_{CC}^2 \times f_i + \Sigma (C_L \times V_{CC}^2 \times f_o)$ where:
   f_i = input frequency in MHz
   f_o = output frequency in MHz
   $\Sigma (C_L \times V_{CC}^2 \times f_o)$ = sum of outputs
   C_L = output load capacitance in pF
   V_{CC} = supply voltage in V
2. For HC the condition is V_I = GND to V_{CC}
   For HCT the condition is V_I = GND to V_{CC} – 1.5 V

ORDERING INFORMATION

See *"74HC/HCT/HCU/HCMOS Logic Package Information"*.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PIN DESCRIPTION

PIN NO.	SYMBOL	NAME AND FUNCTION
1, 19	$\overline{OE}_1$, $\overline{OE}_2$	output enable input (active LOW)
2, 3, 4, 5, 6, 7, 8, 9	A_0 to A_7	data inputs
10	GND	ground (0 V)
18, 17, 16, 15, 14, 13, 12, 11	Y_0 to Y_7	bus outputs
20	V_{CC}	positive supply voltage

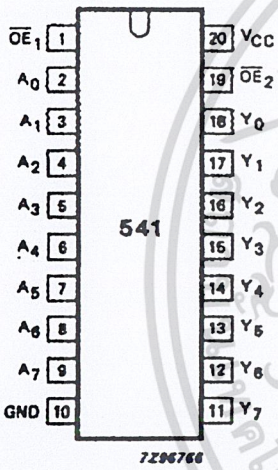


Fig.1 Pin configuration.

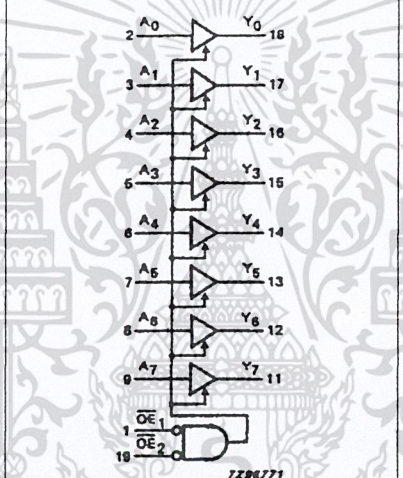


Fig.2 Logic symbol.

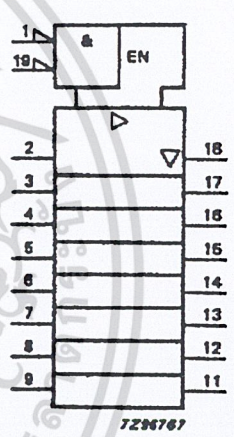


Fig.3 IEC logic symbol.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

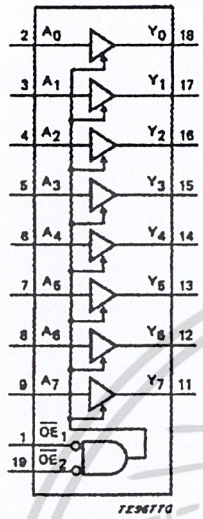


Fig.4 Functional diagram.

FUNCTION TABLE

INPUTS			OUTPUT
$\overline{OE}_1$	$\overline{OE}_2$	A_n	Y_n
L	L	L	L
L	L	H	H
X	H	X	Z
H	X	X	Z

Notes

1. H = HIGH voltage level
   L = LOW voltage level
   X = don't care
   Z = high impedance OFF-state

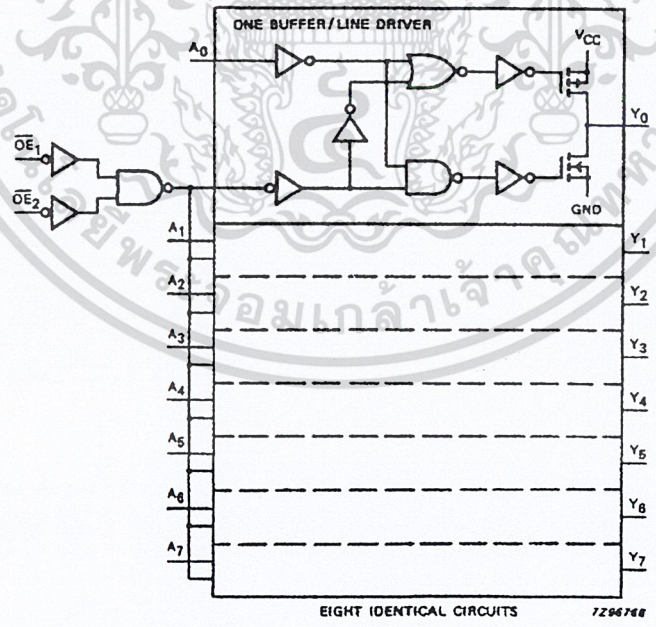


Fig.5 Logic diagram.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DC CHARACTERISTICS FOR 74HC

For the DC characteristics see *"74HC/HCT/HCU/HCMOS Logic Family Specifications"*.

Output capability: bus driver
I_{CC} category: MSI

AC CHARACTERISTICS FOR 74HC

GND = 0 V; $t_r = t_f$ = 6 ns; C_L = 50 pF

SYMBOL	PARAMETER	T_{amb} (°C)							UNIT	TEST CONDITIONS	
		74HC								V_{CC} (V)	WAVEFORMS
		+25			−40 to +85		−40 to +125				
		min.	typ.	max.	min.	max.	min.	max.			
t_{PHL}/ t_{PLH}	propagation delay A_n to Y_n		33 12 10	115 23 20		145 29 25		175 35 30	ns	2.0 4.5 6.0	Fig.6
t_{PZH}/ t_{PZL}	3-state output enable time $\overline{OE}$ to Y_n		55 20 16	160 32 27		200 40 34		240 48 41	ns	2.0 4.5 6.0	Fig.7
t_{PHZ}/ t_{PLZ}	3-state output disable time $\overline{OE}$ to Y_n		61 22 18	160 32 27		200 40 34		240 48 41	ns	2.0 4.5 6.0	Fig.7
t_{THL}/ t_{TLH}	output transition time		14 5 4	60 12 10		75 15 13		90 18 15	ns	2.0 4.5 6.0	Fig.6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DC CHARACTERISTICS FOR 74HCT

For the DC characteristics see *"74HC/HCT/HCU/HCMOS Logic Family Specifications"*.

Output capability: bus driver
I_{CC} category: MSI

Note to HCT types

The value of additional quiescent supply current (ΔI_{CC}) for a unit load of 1 is given in the family specifications. To determine ΔI_{CC} per input, multiply this value by the unit load coefficient shown in the table below.

INPUT	UNIT LOAD COEFFICIENT
$\overline{OE}_1$	1.50
$\overline{OE}_2$	1.00
A_n	0.70

AC CHARACTERISTICS FOR 74HCT

GND = 0 V; $t_r = t_f$ = 6 ns; C_L = 50 pF

SYMBOL	PARAMETER	T_{amb} (°C) 74HCT							UNIT	TEST CONDITIONS	
		+25			−40 to +85		−40 to +125			V_{CC} (V)	WAVEFORMS
		min.	typ.	max.	min.	max.	min.	max.			
t_{PHL}/t_{PLH}	propagation delay A_n to Y_n		15	28		35		42	ns	4.5	Fig.6
t_{PZH}/t_{PZL}	3-state output enable time $\overline{OE}$ to Y_n		21	35		44		53	ns	4.5	Fig.7
t_{PHZ}/t_{PLZ}	3-state output disable time $\overline{OE}$ to Y_n		21	35		44		53	ns	4.5	Fig.7
t_{THL}/t_{TLH}	output transition time		5	12		15		18	ns	4.5	Fig.6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

FAIRCHILD SEMICONDUCTOR™

August 1986
Revised March 2000

DM74LS154
4-Line to 16-Line Decoder/Demultiplexer

General Description

Each of these 4-line-to-16-line decoders utilizes TTL circuitry to decode four binary-coded inputs into one of sixteen mutually exclusive outputs when both the strobe inputs, G1 and G2, are LOW. The demultiplexing function is performed by using the 4 input lines to address the output line, passing data from one of the strobe inputs with the other strobe input LOW. When either strobe input is HIGH, all outputs are HIGH. These demultiplexers are ideally suited for implementing high-performance memory decoders. All inputs are buffered and input clamping diodes are provided to minimize transmission-line effects and thereby simplify system design.

Features

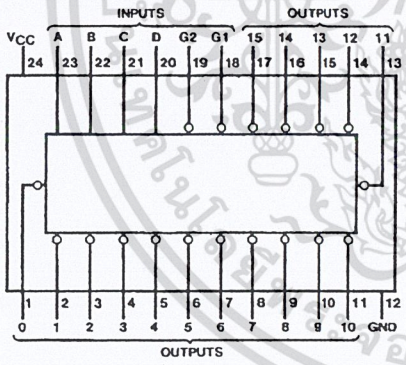
- Decodes 4 binary-coded inputs into one of 16 mutually exclusive outputs
- Performs the demultiplexing function by distributing data from one input line to any one of 16 outputs
- Input clamping diodes simplify system design
- High fan-out, low-impedance, totem-pole outputs
- Typical propagation delay
  - 3 levels of logic 23 ns
  - Strobe 19 ns
- Typical power dissipation 45 mW

Ordering Code:

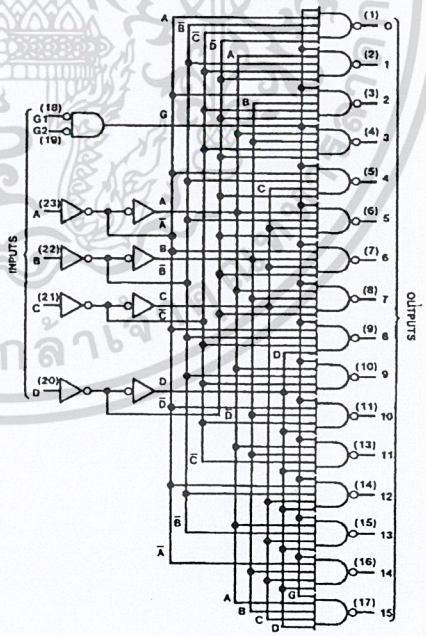
Order Number	Package Number	Package Description
DM74LS154WM	M24B	24-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300 Wide
DM74LS154N	N24A	24-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-010, 0.600 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram



Logic Diagram



© 2000 Fairchild Semiconductor Corporation

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Function Table

Inputs						Outputs														
G1	G2	D	C	B	A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L	L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H
L	L	L	H	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H
L	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H
L	L	H	L	L	L	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H
L	L	H	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H
L	L	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H
L	L	H	H	L	L	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H
L	L	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	L	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H

H = HIGH Level
L = Low Level
X = Don't Care

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Absolute Maximum Ratings(Note 1)

Supply Voltage	7V
Input Voltage	7V
Operating Free Air Temperature Range	0°C to +70°C
Storage Temperature Range	−65°C to +150°C

Note 1: The "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. The device should not be operated at these limits. The parametric values defined in the Electrical Characteristics tables are not guaranteed at the absolute maximum ratings. The "Recommended Operating Conditions" table will define the conditions for actual device operation.

Recommended Operating Conditions

Symbol	Parameter	Min	Nom	Max	Units
V_{CC}	Supply Voltage	4.75	5	5.25	V
V_{IH}	HIGH Level Input Voltage	2			V
V_{IL}	LOW Level Input Voltage			0.8	V
I_{OH}	HIGH Level Output Current			−0.4	mA
I_{OL}	LOW Level Output Current			8	mA
T_A	Free Air Operating Temperature	0		70	°C

Electrical Characteristics

over recommended operating free air temperature range (unless otherwise noted)

Symbol	Parameter	Conditions	Min	Typ (Note 2)	Max	Units
V_I	Input Clamp Voltage	V_{CC} = Min, I_I = −18 mA			−1.5	V
V_{OH}	HIGH Level Output Voltage	V_{CC} = Min, I_{OH} = Max, V_{IL} = Max, V_{IH} = Min	2.7	3.4		V
V_{OL}	LOW Level Output Voltage	V_{CC} = Min, I_{OL} = Max		0.25	0.4	V
		V_{IL} = Max, V_{IH} = Min		0.35	0.5	
		I_{OL} = 4 mA, V_{CC} = Min		0.25	0.4	
I_I	Input Current @ Max Input Voltage	V_{CC} = Max, V_I = 7V			0.1	mA
I_{IH}	HIGH Level Input Current	V_{CC} = Max, V_I = 2.7V			20	µA
I_{IL}	LOW Level Input Current	V_{CC} = Max, V_I = 0.4V			−0.4	mA
I_{OS}	Short Circuit Output Current	V_{CC} = Max (Note 3)	−20		−100	mA
I_{CC}	Supply Current	V_{CC} = Max (Note 4)		9	14	mA

Note 2: All typicals are at V_{CC} = 5V, T_A = 25°C.

Note 3: Not more than one output should be shorted at a time, and the duration should not exceed one second.

Note 4: I_{CC} is measured with all outputs OPEN and all inputs GROUNDED.

Switching Characteristics

at V_{CC} = 5V and T_A = 25°C

Symbol	Parameter	From (Input) To (Output)	R_L = 2 kΩ				Units
			C_L = 15 pF		C_L = 50 pF		
			Min	Max	Min	Max	
t_{PLH}	Propagation Delay Time LOW-to-HIGH Level Output	Data to Output		30		35	ns
t_{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Data to Output		30		35	ns
t_{PLH}	Propagation Delay Time LOW-to-HIGH Level Output	Strobe to Output		20		25	ns
t_{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Strobe to Output		25		35	ns

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ค

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Pin Name	84-Pin PLCC	144-Pin TQFP	208-Pin PQFP/RQFP
MSEL0 (2)	31	77	108
MSEL1 (2)	32	76	107
nSTATUS (2)	55	35	52
nCONFIG (2)	34	74	105
DCLK (2)	13	107	155
CONF_DONE (2)	76	2	2
INIT_DONE (3)	69	14	19
nCE (2)	14	106	154
nCEO (2)	75	3	3
nWS (4)	80	142	206
nRS (4)	81	141	204
nCS (4)	78	144	208
CS (4)	79	143	207
RDYnBSY (4)	70	11	16
CLKUSR (4)	73	7	10
DATA7 (4)	5	116	166
DATA6 (4)	6	114	164
DATA5 (4)	7	113	162
DATA4 (4)	8	112	161
DATA3 (4)	9	111	159
DATA2 (4)	10	110	158
DATA1 (4)	11	109	157
DATA0 (2), (5)	12	108	156
TDI (2)	15	105	153
TDO (2)	74	4	4
TCK (2)	77	1	1
TMS (2)	57	34	50
TRST (2)	56	(6)	51
Dedicated Inputs	2, 42, 44, 84	54, 56, 124, 126	78, 80, 182, 184
Dedicated Clock Pins	1, 43	55, 125	79, 183
DEV_CLRn (3)	3	122	180
DEV_OE (3)	83	128	186
VCCINT	4, 20, 33, 40, 45, 63	6, 25, 52, 53, 75, 93, 123	6, 23, 35, 43, 76, 77, 106, 109, 117, 137, 145, 181
VCCIO	–	5, 24, 45, 61, 71, 94, 115, 134	5, 22, 34, 42, 66, 84, 98, 110, 118, 138, 146, 165, 178, 194
GNDINT	26, 41, 46, 68, 82	16, 57, 58, 84, 103, 127	21, 33, 49, 81, 82, 123, 129, 151, 185
GNDIO	–	15, 40, 50, 66, 85, 104, 129, 139	20, 32, 48, 59, 72, 91, 124, 130, 152, 171, 188, 201
No Connect (N.C.) (7)	–	–	7, 8, 9, 14, 15, 36, 37, 113, 114, 125, 126, 139, 140
Total User I/O Pins (8)	59	102	134

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Notes:

(1) All pins that are not listed are user I/O pins.
(2) This pin is a dedicated pin; it is not available as a user I/O pin.
(3) This pin can be used as a user I/O pin if it is not used for its device-wide or configuration function.
(4) This pin can be used as a user I/O pin after configuration.
(5) This pin is tri-stated in user mode.
(6) The optional JTAG pin TRST is not used in the 100-pin or 144-pin TQFP package.
(7) To maintain pin compatibility when transferring to the EPF10K10 or EPF10K10A device from any other device in the 208-pin PQFP or 256-pin FineLine BGA package, do not use these pins as user I/O pins.
(8) The user I/O pin count includes dedicated input pins, dedicated clock pins, and all I/O pins.
(9) To maintain pin compatibility when transferring to the EPF10K30 device from any other device in the 356-pin BGA or 484-pin FineLine BGA package, do not use these pins as user I/O pins.

(10) To maintain pin compatibility when transferring to the EPF10K50V device from any other device in the 484-pin FineLine BGA package, do not use these pins as user I/O pins.
(11) To maintain pin compatibility when transferring from the EPF10K100 to the EPF10K70 in the 503-pin PGA package, do not use these pins as user I/O pins.
(12) This pin shows the status of the ClockLock and ClockBoost circuitry. When the ClockLock and ClockBoost circuitry are locked to the incoming clock and generate an internal clock, LOCK is driven high. LOCK remains high if a periodic clock stops clocking. The LOCK function is optional; if the LOCK output is not used, this pin is a user I/O pin.
(13) This pin drives the ClockLock and ClockBoost circuitry.
(14) This pin is the power or ground for the ClockLock and ClockBoost circuitry. To ensure noise resistance, the power and ground supply to the ClockLock and ClockBoost circuitry should be isolated from the power and ground to the rest of the device.
(15) To maintain pin compatibility when transferring to the EPF10K100A device from another device in the 600-pin BGA package, do not use these pins as user I/O pins.
(16) The 240-pin QFP packages do not support the MultiVolt I/O feature so there are no VCCIO pins.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Search for "Configuring Multiple FLEX Devices in a FLEX Chain with the BitBlaster, ByteBlaster, or FLEX Download Cable" in MAX+PLUS II Help.

JTAG Mode

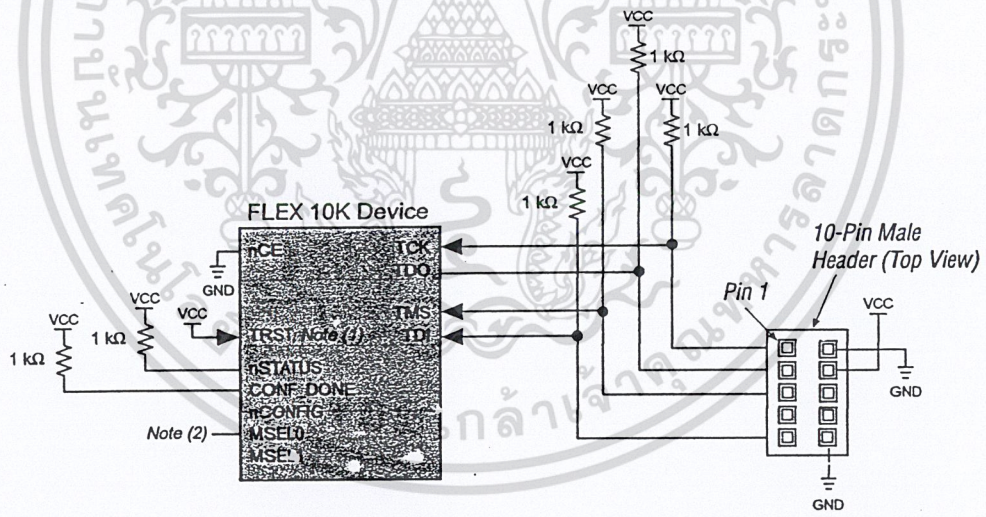
For in-system programming and in-circuit reconfiguration in JTAG mode, the ByteBlaster cable connects to devices on the circuit board via any standard parallel port. This section discusses the following topics:

- JTAG configuration of a single FLEX 10K device
- JTAG programming of a single MAX 9000, MAX 7000S, or MAX 7000A device
- JTAG programming and configuration of multiple devices

JTAG Configuration of a Single FLEX 10K Device

The MAX+PLUS II software downloads the SRAM Object File (.sof), created during compilation, directly to the device via the ByteBlaster cable. Refer to "Software Instructions" on page 312 for more information. Devices are configured via the JTAG pins: TCK, TMS, TDI, and TDO. Figure 9 shows how the ByteBlaster cable interfaces with a single FLEX 10K device. All other I/O pins are tri-stated in this configuration.

Figure 9. JTAG Configuration of a Single FLEX 10K Device



Notes:

(1) FLEX 10K devices in 144-pin TQFP packages do not have a TRST pin. Therefore, the TRST pin can be ignored when using these devices.The nCONFIG, MSEL0, and MSEL1 pins should be connected to support a FLEX configuration scheme. If only JTAG configuration is used, connect nCONFIG to V_{CC}, and MSEL0 and MSEL1 to ground.

(2) The nCONFIG, MSEL0, and MSEL1 pins should be connected to support a FLEX configuration scheme. If only JTAG configuration is used, connect nCONFIG to V_{CC}, and MSEL0 and MSEL1 to ground.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ByteBlaster Parallel Port Download Cable

February 1998, ver. 2.01 — Data Sheet

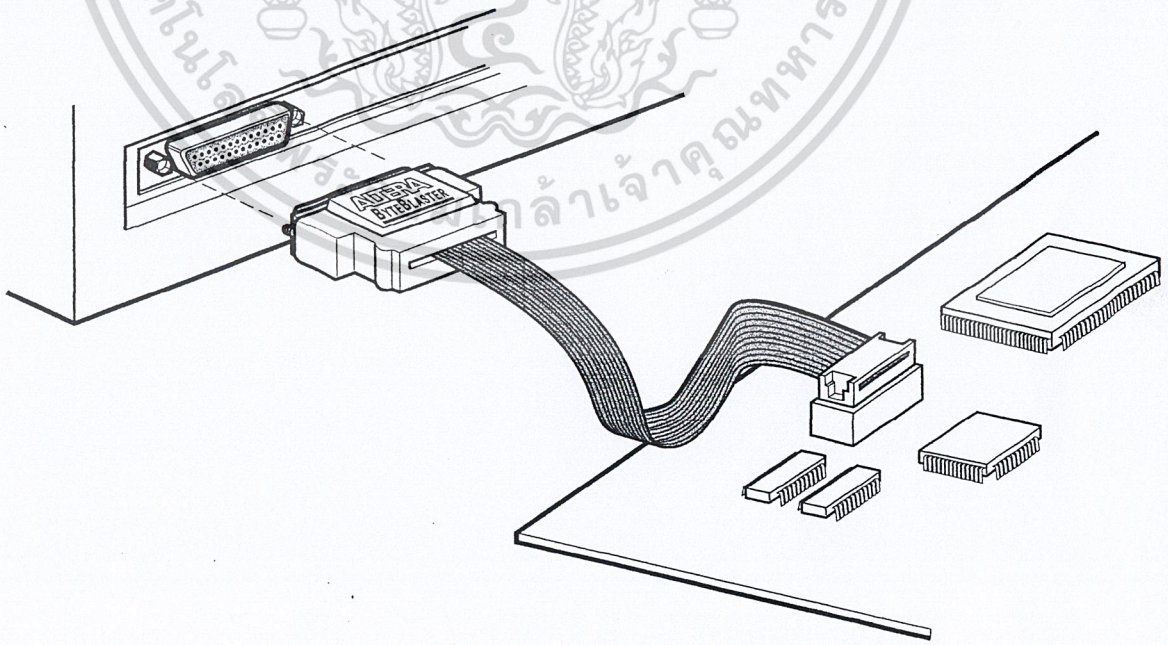
Features

- Allows PC users to perform the following functions:
  - Program MAX® 9000 (including MAX 9000A), MAX 7000S, and MAX 7000A devices in-system via a standard parallel port
  - Configure FLEX® 10K, FLEX 8000, and FLEX 6000 devices in-circuit via a standard parallel port
- Provides a fast and low-cost method for in-system programming
- Downloads data from the MAX+PLUS® II development software
- Interfaces with a standard 25-pin parallel port on PCs
- Uses 10-pin circuit board connector, which is identical to that of the BitBlaster™ serial download cable

Functional Description

The ByteBlaster™ parallel port download cable (ordering code: PL-BYTEBLASTER) is a hardware interface to a standard PC parallel port (also known as an LPT port). This cable channels configuration data to FLEX 10K, FLEX 8000, and FLEX 6000 devices, as well as programming data to MAX 9000, MAX 7000S, and MAX 7000A devices. Because design changes are downloaded directly to the device, prototyping is easy and multiple design iterations can be accomplished in quick succession. See Figure 1.

Figure 1. ByteBlaster Parallel Port Download Cable



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Download Modes

The ByteBlaster cable provides two download modes:

- Passive serial (PS) mode—Used for configuring FLEX 10K, FLEX 8000, and FLEX 6000 devices
- JTAG mode—Industry-standard Joint Test Action Group (JTAG) boundary-scan test (BST) circuitry (compliant with IEEE Std. 1149.1-1990) implemented for configuring FLEX 10K or programming MAX 9000, MAX 7000S, and MAX 7000A devices

ByteBlaster Connections

The ByteBlaster cable has a 25-pin male header that connects to the PC parallel port, and a 10-pin female plug that connects to the circuit board. Data is downloaded from the PC's parallel port through the ByteBlaster download cable to the circuit board via the connections discussed in this section.

☞ To configure / program 3.3-V devices (e.g., FLEX 10KA and MAX 7000A devices) using the ByteBlaster cable, connect the cable's VCC pin to a 5.0-V power supply and the device to a 3.3-V power supply. FLEX 10KA and MAX 7000A devices have 5.0-V tolerant inputs, so the ByteBlaster cable's 5.0-V output will not harm these 3.0-V devices. The pull-up resistors should be connected to the 5.0-V power supply.

ByteBlaster Header & Plug Connections

The 25-pin male header connects to a parallel port with a standard parallel cable. Table 1 identifies the pins and the download modes.

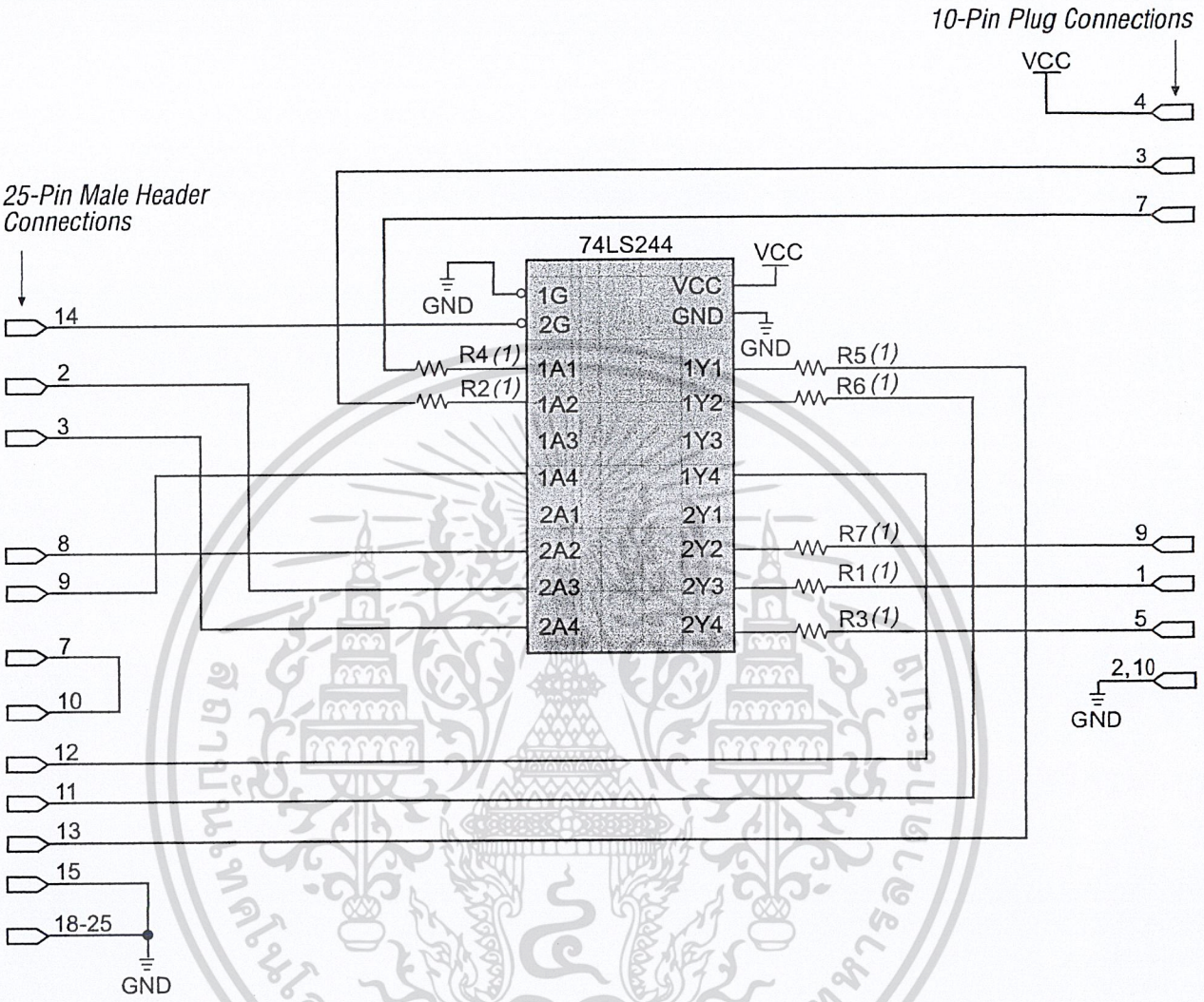
Table 1. ByteBlaster 25-Pin Header Pin-Outs

Pin	JTAG Mode Signal Name	PS Mode Signal Name
2	TCK	DCLK
3	TMS	nCONFIG
8	TDI	DATA0
11	TDO	CONF_DONE
13	NC	nSTATUS
15	GND	GND
18 to 25	GND	GND

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Figure 2 shows a schematic of the ByteBlaster download cable.

Figure 2. ByteBlaster Schematic



Note:
(1) All resistors are 33 Ω.

The 10-pin female plug connects to a 10-pin male header on the circuit board containing the target device(s). Figure 3 shows the dimensions of the female plug.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Figure 3. ByteBlaster 10-Pin Female Plug Dimensions

Dimensions are shown in inches. The spacing between pin centers is 0.1 inch.

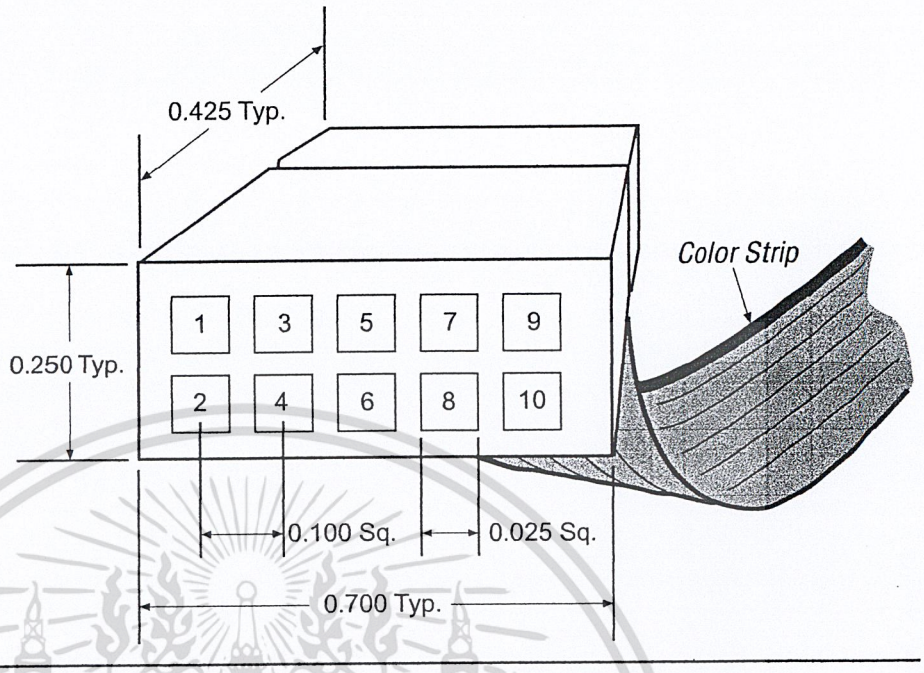


Table 2 identifies the 10-pin female plug's pin names for the corresponding download mode.

Table 2. ByteBlaster Female Plug's Pin Names & Download Modes

Pin	JTAG Mode		PS Mode	
	Signal Name	Description	Signal Name	Description
1	TCK	Clock signal	DCLK	Clock signal
2	GND	Signal ground	GND	Signal ground
3	TDO	Data from device	CONFIG_DONE	Configuration control
4	VCC	Power supply	VCC	Power supply
5	TMS	JTAG state machine control	nCONFIG	Configuration control
6	NC	No connect	NC	No connect
7	NC	No connect	nSTATUS	Configuration status
8	NC	No connect	NC	No connect
9	TDI	Data to device	DATA0	Data to device
10	GND	Signal ground	GND	Signal ground

☞ The circuit board must supply V_{CC} and ground to the ByteBlaster cable.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

โครงงานนี้ดำเนินการไปได้ด้วยดี เพราะได้รับความช่วยเหลือตลอดจนแนวคิดและคำแนะนำที่มีประโยชน์จากอาจารย์ที่ปรึกษาคือ ร.ศ.ดร.มนัส สังวรศิลป์ และคุณศุภชัย งามอเนกรัตน์ (พี่แม็ก) ที่ได้สละเวลามาให้ความช่วยเหลือและแนะแนวทางในการแก้ไขปัญหาต่างๆ เกิดขึ้นในการทำงานครั้งนี้ อีกทั้งความช่วยเหลือจากเพื่อนๆ และน้องๆ ที่ชุมนุมวิชาการ ความช่วยเหลือจากชุมนุมอิเล็กทรอนิกส์และเพื่อน ๆ พี่ ๆ ที่ห้อง ReCCit ทางคณะผู้จัดทำก็ขอขอบคุณทุกๆ ท่านมา ณ ที่นี้ด้วย

นายพีรกิจ ชังเจริญสุข
นายภาสกร เจตกวิน
นางสาวภาสินี เจริญขวัญ
ผู้จัดทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

1. เดชชาญ หาญไชยชนะและเดชา คูหาเกษมสิน,"บอร์ดแสดงผลไฟวิ่ง". วิทยานิพนธ์วิศวกรรมศาสตร์บัณฑิต ภาควิชาวิศวกรรมอิเล็คทรอนิกส์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, ปีการศึกษา 2544.
2. จันจิรา เจือกโว้นและชำนาญ ปัญญาใส,"การออกแบบวงจรด้วย FPGA(FPGA Design)" เอกสารอัดสำเนา.ศูนย์วิจัยเทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ ,ไม่ระบุปีที่พิมพ์.
3. ชาติชาย ดิษฐกุล นอ.,"คู่มือภาษา VHDL".เอกสารอัดสำเนา. ศูนย์วิจัยเทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ ,ไม่ระบุปีที่พิมพ์.
4. อีทีที,ทีมงาน."ET-Display 7x50 บอร์ดแสดงผลตัวอักษรวิ่งแบบ LED แมตริกซ์".วารสารเซมิคอนดัคเตอร์อิเล็กทรอนิกส์, ฉบับที่ 226, 2544, หน้า 144-152.
5. ซีวิคมีเดีย,ทีมงาน."VMS NEO V1 บอร์ดแสดงผลตัวอักษรวิ่งขนาดเล็ก". วารสารเซมิคอนดัคเตอร์อิเล็กทรอนิกส์, ฉบับที่ 229 EBG'2002, 2545, หน้า 124-131.
6. ศิวเมธ มเหศวร."ระบบการมัลติเพล็กซิ่งพื้นฐานและการออกแบบ". ". วารสารเซมิคอนดัคเตอร์อิเล็กทรอนิกส์, ฉบับที่ 197, 2542, หน้า 161-172.
7. Adam Chapweske."PS/2 Mouse/Keyboard Protocol ",1999.
8. Jonas Thor."Interfacing a PS/2 Keyboard",Lab 3.1 in SDM098 , November, 2001.
9. http://panda.cs.ndsu.nodak.edu/~achapwes/PICmicro/keyboard/scancodes2.html
10. http://panda.cs.ndsu.nodak.edu/~achapwes/PICmicro/PS2/ps2.htm
11. http://www.astronlogic.com

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้