การออกแบบเครื่องจับสัญญาณภาพทางการแพทย์
โดยใช้เอฟพีจีเอ

A DESIGN OF MEDICAL IMAGE CAPTURING SYSTEM USING FPGA

บุญอนันต์ เกียงเอีย
BOONANAN KIANG-EIA

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมอิเล็กทรอนิกส์
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2545
ISBN 974-9546-02-4

การออกแบบเครื่องจับสัญญาณภาพทางการแพทย์ โดยใช้เอฟพีจีเอ

A DESIGN OF MEDICAL IMAGE CAPTURING SYSTEM USING FPGA

บุญอนันต์ เกียงเอีย
BOONANAN KIANG-EIA

วิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมอิเล็กทรอนิกส์
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พศ.2545

ISBN 974-9546-02-4

A DESIGN OF MEDICAL IMAGE CAPTURING SYSTEM USING FPGA

BOONANAN KIANG-EIA

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
MASTER OF ENGINEERING IN ELECTRONIC ENGINEERING
SCHOOL OF GRADUATE STUDIES
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
2002

ISBN 974-9546-02-4

หัวข้อวิทยานิพนธ์	การออกแบบเครื่องจับสัญญาณภาพทางการแพทย์โดยใช้เอฟพีจีเอ
นักศึกษา	นายบุญอนันต์ เกียงเอีย
รหัสประจำตัว	43061310
ปริญญา	วิศวกรรมศาตรมหาบัณฑิต
สาขาวิชา	วิศวกรรมอิเล็กทรอนิกส์
พ.ศ.	2545
อาจารย์ผู้ควบคุมวิทยานิพนธ์	รศ.ดร.มนัส สังวรศิลป์

บทคัดย่อ

ในการนำข้อมูลภาพทางการแพทย์จากเครื่อง MRI หรือ CT เข้ามาประมวลผลในเครื่องคอมพิวเตอร์ส่วนบุคคล สามารถกระทำได้ 2 วิธี โดยการศึกษารูปแบบไฟล์ข้อมูลภาพจากเครื่อง MRI หรือ CT หรืออีกวิธีหนึ่งคือสร้างฮาร์ดแวร์ดึงสัญญาณภาพจากมอนิเตอร์ของระบบแล้วมาทำการดิจิไตซ์ข้อมูล วิธีแรกจะต้องใช้คอมพิวเตอร์ในระบบยูนิกซ์ (UNIX) ในการนำไฟล์ข้อมูลภาพออกมาซึ่งจะมีราคาแพงและเสียเวลาในการศึกษาฟอร์แมตภาพเป็นอย่างมาก ในงานวิจัยนี้จะนำเสนอเทคนิค เพื่อที่จะนำภาพจากสัญญาณภาพของ MRI หรือ CT มาแปลงเป็นสัญญาณดิจิตอล เพื่อนำมาใช้ในคอมพิวเตอร์ส่วนบุคคล โดยใช้ภาษาวีเอชดีแอล (VHDL) ในการบรรยายพฤติกรรมของฮาร์ดแวร์ที่จะทำการออกแบบแล้วโปรแกรมลงในชิพเอฟพีจีเอ (FPGA) เพื่อทดสอบการทำงาน หรือนำคอร์ (CORE) ที่ได้ไปใช้ในการออกแบบ ASIC (Application Specific Integrated Circuit) ต่อไป ข้อดีของวิธีการนี้คือสามารถเลือกบริเวณที่จะทำการจัดเก็บข้อมูลภาพได้ โดยการกำหนดตำแหน่งและขนาดของข้อมูลภาพที่จะทำการจัดเก็บ

Thesis Title	A Design of Medical Image Capturing System Using FPGA
Student	Mr.Boonanan Kiang-eia
Student ID.	43061310
Degree	Master of Engineering
Programme	Electronic Engineering
Year	2002
Thesis Advisor	Assoc.Prof.Dr.Manas Sangworasil

ABSTRACT

Transferring of medical imaging data from imaging systems, such as MRI or CT, to a personal computer for processing can be achieved using two methods—decoding the data format stored on MRI or CT directly, or capturing and digitizing the video signals from imaging systems with a customarily built hardware. The first method requires a costly UNIX-based workstation for processing of medical images, and it is generally more time-consuming. In this thesis, I propose a novel technique for capturing medical imaging data from MRI or CT and transform it to digital signals for use in personal computers. VHSIC Hardware Description Language (VHDL) was used to describe the characteristics of the designed hardware. VHDL codes were then programmed on the Field Programmable Gate Array (FPGA) chip for performance testing. CORE from VHDL can also be utilized for the design of Application Specific Integrated Circuit (ASIC) in the future. The major advantage of the proposed technique is that it allows us to specifically choose the area where we want to store the imaging data by selecting the position and dimensions of the desired image.

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้ลุล่วงไปได้ด้วยดีก็ด้วยความช่วยเหลือและการสนับสนุนจากบุคคลหลายๆ ฝ่ายด้วยกัน ซึ่งผู้เขียนขอขอบพระคุณทุกๆ ท่านดังต่อไปนี้

ขอขอบพระคุณคุณพ่อและคุณแม่ ผู้ซึ่งคอยให้การเลี้ยงดู อบรบสั่งสอน ให้กำลังใจและโอกาสในการศึกษาเล่าเรียนตลอดมา

ขอขอบพระคุณ รศ.ดร.มนัส สังวรศิลป์ อาจารย์ที่ปรึกษา ผู้ซึ่งคอยให้คำปรึกษา ความรู้ความเข้าใจและโอกาสในหลายๆ ด้าน ตลอดระยะเวลาในการศึกษา รวมถึงการเอื้อเฟื้อสนับสนุนเครื่องมือเครื่องใช้และสถานที่ในการทำวิจัยอย่างพร้อมเพรียง ผู้เขียนรู้สึกซาบซึ้งในความเมตตาของท่านเป็นอย่างสูง

ขอขอบพระคุณ ดร.สุธี ผู้เจริญชนะชัย , ผศ.ดร.สุรพันธุ์ เอื้อไพบูลย์ , ดร.สุพันธุ์ ตั้งจิตกุศลมั่น และ ผศ.ประภากร สุวรรณะที่ให้คำแนะนำและแง่คิดต่างๆ ที่เป็นประโยชน์ต่อการทำวิจัย

ขอขอบพระคุณ ผศ.นพ.ราเมศร์ วัชรสินธุ์ ภาควิชารังสีเทคนิค โรงพยาบาลรามาธิบดี และ รศ.ดร.รุจพร ชนะชัย ภาควิชารังสีวิทยา คณะแพทย์ศาสตร์ศิริราชพยาบาล มหาวิทยาลัย-มหิดล ที่เอื้อเฟื้อข้อมูลและสถานที่ในการทดสอบงานวิจัย และให้คำปรึกษาต่างๆ เกี่ยวกับข้อมูลภาพที่ได้จากการทดลอง

ขอขอบคุณ คุณวรเทพ ไพบูลย์รัตนากร คุณประเสริฐ อัศวรุ่งสกุล ที่ให้คำแนะนำเกี่ยวกับการทำงานวิจัย การเขียนวิทยานิพนธ์ และคำปรึกษาต่างๆ

ขอขอบคุณ คุณฐิติรัตน์ สุวรรณเนาว์ ที่คอยให้กำลังใจและช่วยเหลือแกผู้เขียนในการทำงานวิจัยตลอดมา

ขอขอบคุณสมาชิกห้อง Bio Lab ที่คอยให้ความช่วยเหลือต่างๆ ในการทำงานวิจัยนี้

ขอขอบคุณสถาบันบัณฑิตวิทยาศาสตร์และเทคโนโลยีไทย สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ ที่ให้เงินทุนสนับสนุนการศึกษาในระดับปริญญาโทแก่ผู้เขียน

บุญอนันต์ เกียงเอีย

สารบัญ

หน้า

บทคัดย่อภาษาไทย.....I

บทคัดย่อภาษาอังกฤษ.....II

กิตติกรรมประกาศ.....III

สารบัญ.....IV

สารบัญตาราง.....VIII

สารบัญรูป.....IX

บทที่ 1 บทนำ.....1

1.1 ความเป็นมาและความสำคัญของปัญหา.....1

1.2 วัตถุประสงค์ของการวิจัย.....1

1.3 ขอบเขตของการวิจัย.....2

1.4 โครงสร้างของวิทยานิพนธ์.....2

บทที่ 2 การจัดเก็บข้อมูลภาพ.....4

2.1 ลักษณะของสัญญาณภาพ.....4

2.2 การเปลี่ยนข้อมูลจากสัญญาณอนาลอกเป็นสัญญาณดิจิตอล.....10

2.2.1 การสุ่ม (Sampling).....10

2.2.2 การควอนไตซ์ (Quantization).....12

2.3 ลักษณะหน่วยความจำที่ใช้ในการจัดเก็บข้อมูล.....14

บทที่ 3 การออกแบบวงจรในลักษณะโครงสร้างและการบรรยายพฤติกรรม.....16

3.1 การออกแบบระบบดิจิตอล.....16

3.2 ประวัติความเป็นมาของภาษาวีเอชดีแอล.....18

3.3 ข้อกำหนดของภาษาวีเอชดีแอล.....19

3.3.1 ลักษณะทั่วไป.....19

3.3.2 สนับสนุนการออกแบบแบบลำดับขั้น.....20

3.3.3 ไลบรารี.....20

3.3.4 ลำดับคำสั่ง.....20

3.3.5 การกำหนดคุณสมบัติ.....20

สารบัญ(ต่อ)

หน้า

3.3.6 ชนิดของข้อมูล.....21
3.3.7 โปรแกรมย่อย.....21
3.3.8 การควบคุมเวลา.....21
3.3.9 การกำหนดแบบโครงสร้าง.....21
3.4 องค์ประกอบพื้นฐานของวีเอชดีแอล.....21
3.4.1 การกำหนดการเชื่อมต่อ.....22
3.4.2 การกำหนดรูปแบบการบรรยาย.....23
3.4.3 หน่วยการออกแบบแพ็กเก็ต.....24
3.4.4 หน่วยการออกแบบ Configuration.....25
3.4.5 โปรแกรมย่อย.....25
3.4.6 โอเปอร์เรเตอร์.....26
3.4.7 เวลาและความพร้อมเพรียง.....27
3.4.8 สัญญาณและตัวแปร.....27
3.5 การบรรยายเชิงพฤติกรรม.....28
3.6 โปรเซส.....28
3.7 การกำหนดตัวดำเนินการภายในโปรเซส.....29
3.8 การกำหนดการกระทำภายในโปรเซส.....29
3.9 การกระตุ้นและยับยั้งการกระทำของโปรเซส.....30
3.10 การออกแบบจากบนลงล่าง.....32
3.11 โครงสร้างภายในของเอฟพีจีเอ.....34
3.12 ปัจจัยที่ทำให้การออกแบบเอฟพีจีเอทำได้ง่ายและสะดวกรวดเร็ว.....36
3.13 การออกแบบโดยใช้ภาษาอธิบายพฤติกรรมของฮาร์ดแวร์.....37
3.13.1 การสังเคราะห์วงจร (Logic Synthesis).....37
3.13.2 การแบ่งวงจร (Partitioning).....38
3.13.3 การวางอุปกรณ์ (Placement).....38
3.13.4 การเชื่อมต่อสัญญาณ (Routing).....39
3.13.5 ความหน่วงด้านเวลา (Delay).....39
3.15.6 การจำลองการทำงานของวงจร (Simulation).....40

สารบัญ(ต่อ)

หน้า

3.15.7 การโปรแกรมอุปกรณ์เอฟพีจีเอ (Configuration).....40

3.14 เครื่องมือสำหรับการออกแบบเอฟพีจีเอ.....41

บทที่ 4 การออกแบบและสร้างเครื่องแปลงสัญญาณภาพทางการแพทย์โดยใช้เอฟพีจีเอ 42

4.1 ลักษณะโครงสร้างโดยรวมของระบบ.....42

4.1.1 Analog to Digital Converter.....43

4.1.2 Sync Separator.....46

4.1.3 หน่วยความจำ.....47

4.1.4 หน่วงข้อมูล.....48

4.1.5 บัฟเฟอร์.....48

4.1.6 ชุดควบคุมโดยเอฟพีจีเอ.....48

4.2 สถาปัตยกรรมของชุดควบคุมการทำงาน.....48

4.3 สถาปัตยกรรมของชุดอินเตอร์เฟส ISA แบบ 16 บิท.....49

4.4 สถาปัตยกรรมของวงจรกำหนดตำแหน่งและขนาดของข้อมูลภาพ.....67

4.4.1 Lines Select 68

4.4.2 Pixel Select.....69

4.5 สถาปัตยกรรมของชุดกำเนิดสัญญาณควบคุมหน่วยความจำแบบ SRAM.....70

4.6 ขั้นตอนการออกแบบระบบบนชิพเอฟพีจีเอ.....72

4.6.1 Design Entry.....73

4.6.2 Compilation.....74

4.6.3 Simulation.....74

4.6.4 Device Programmming.....74

4.6.5 Hardware Test.....75

บทที่ 5 การทดลอง.....76

5.1 ทดสอบโดยการจำลองการทำงาน (Simulation).....76

5.2 ทดสอบการทำงานจริงของวงจร.....82

สารบัญ(ต่อ)

หน้า

บทที่ 6 สรุปผลการวิจัยและข้อเสนอแนะ.....90
6.1 สรุปผลการวิจัย.....90
6.2 อุปสรรคที่พบในการวิจัย.....91
6.3 ข้อเสนอแนะในการพัฒนา.....91

บรรณานุกรม.....92
ภาคผนวก.....94
ภาคผนวก ก วงจร.....95
ภาคผนวก ข วงจรภายในเอฟพีจีเอ 102
ประวัติผู้เขียน.....112

สารบัญตาราง

ตารางที่ หน้า

2.1 รายละเอียดของสัญญาณภาพจากเครื่อง CT ยี่ห้อ HITACHI รุ่น W-450.....8

2.2 รายละเอียดของสัญญาณภาพที่ได้รับจากเครื่อง CT ยี่ห้อ HITACHI รุ่นต่างๆ.....9

4.1 สัญญาณทางด้าน A และ B.....52

4.2 สัญญาณทางด้าน C และ D.....53

4.3 หน้าที่ของสัญญาณต่างๆ.....54

4.4 รหัสแสดงแอดเดรสของพอร์ตสำหรับการ์ดอินเตอร์เฟส.....66

4.4 คาบเวลาที่ใช้สำหรับการอ่านและเขียนข้อมูลของ SRAM เบอร์ K6R4008C1C-JC12.....72

5.1 ผลการทดลอง.....83

5.2 ผลการทดลอง.....84

5.3 ผลการทดลอง.....85

5.4 ผลการทดลอง.....86

5.5 ผลการทดลอง.....87

5.6 ผลการทดลอง.....88

สารบัญรูป

รูปที่ หน้า

2.1 ลักษณะของสัญญาณวิดีโอ....4

2.2 เปรียบเทียบการสแกนที่หน้าจอของมอนิเตอร์กับสัญญาณที่ใช้ควบคุมการสแกนทั้งสองแกน.5

2.3a ตำแหน่งของพิกเซลทางหน้าจอแสดงผลและตำแหน่งแอดเดรสของหน่วยความจำ....6

2.3b เวลาในการสุ่มสำหรับเส้นสแกนของสัญญาณคอมโพซิทวีดีโอ....6

2.4 ลักษณะสัญญาณภาพที่ได้จากเครื่อง CT ยี่ห้อ HITACHI....7

2.5 ฟังค์ชันของการสุ่ม 2 มิติ....10

2.6a ภาพขนาด 512x512 จุดภาพ....11

2.6b ภาพขนาด 256x256 จุดภาพ....11

2.6c ภาพขนาด 128x128 จุดภาพ....11

2.6d ภาพขนาด 64x64 จุดภาพ....11

2.6e ภาพขนาด 32x32 จุดภาพ....11

2.6f ภาพขนาด 16x16 จุดภาพ....11

2.7a ความละเอียด 8 บิตต่อจุดภาพ....13

2.7b ความละเอียด 7 บิตต่อจุดภาพ....13

2.7c ความละเอียด 6 บิตต่อจุดภาพ....13

2.7d ความละเอียด 5 บิตต่อจุดภาพ....13

2.7e ความละเอียด 4 บิตต่อจุดภาพ....13

2.7f ความละเอียด 3 บิตต่อจุดภาพ....13

2.7g ความละเอียด 2 บิตต่อจุดภาพ....14

2.7h ความละเอียด 1 บิตต่อจุดภาพ....14

2.8 ลักษณะการต่อหน่วยความจำเข้ากับระบบโดยตรง....15

3.1 ขั้นตอนการออกแบบระบบดิจิตอล....16

3.2 การออกแบบระบบเส้นทางของข้อมูล....17

3.3 การกำหนดการเชื่อมต่อและสถาปัตยกรรม....22

3.4 บล็อคไดอะแกรมและการบรรยายการเชื่อมต่อของ clock_ component....23

3.5 การบรรยายเชิงพฤติกรรมของ clock _ component....23

3.6 โครงสร้างทั่วไปของส่วนการประกาศแพ็กเกจ....24

สารบัญรูป(ต่อ)

รูปที่ หน้า

3.7 โครงสร้างของบอดี้แพ็กเกจ....25

3.8 โครงสร้างโดยทั่วไปของหน่วยการออกแบบโครงแบบ....25

3.9 การใช้โพรซีเจอร์....26

3.10 การใช้ฟังก์ชัน....26

3.11 ตัวดำเนินการในวีเอชดีแอล....27

3.12 รูปแบบของการบรรยายแบบโปรเซส....28

3.13 ตัวอย่างการประกาศตัวดำเนินการภายในโปรเซส....29

3.14 เงื่อนไขการกระทำในโปรเซส....30

3.15 การกระทำในโปรเซส....30

3.16 (a) ตัวอย่างโมเดล D-Flip Flop....31

3.16 (b) การบรรยายการเชื่อมต่อของ D-Flip Flop....31

3.17 การบรรยายเชิงพฤติกรรมของ D-FlipFlop....32

3.18 ขั้นตอนการออกแบบจากบนลงล่าง....33

3.19 โครงสร้างภายในของเอฟพีจีเอตระกูล MAX7000S....35

3.20 โครงสร้างภายในของเอฟพีจีเอตระกูล FLEX10K....35

3.21 การโปรแกรมลงในชิพ....36

4.1 บล็อกไดอะแกรมของระบบการเก็บข้อมูลภาพ....42

4.2 วงจรแปลงสัญญาณอนาลอกเป็นดิจิตอล....43

4.3 โครงสร้างภายในของ TDA8708A....44

4.4 วงจรกำเนิดสัญญาณ Gate A และ Gate B โดยใช้โมโนสเตเบิ้ล....45

4.5 วงจรกำเนิดสัญญาณ Gate A และ Gate B อย่างง่ายโดยใช้อินเวอร์เตอร์....46

4.6 สัญญาณ Gate A และ Gate B เมื่อทำการเปรียบเทียบกับสัญญาณ Hor Sync และ สัญญาณ BURST....46

4.7 วงจรแยกสัญญาณซิงค์....47

4.8 วงจรหน่วยความจำ....47

4.9 บล็อกไดอะแกรมโดยรวมของชุดควบคุมการทำงาน....48

4.10 บล็อกไดอะแกรมภายในชุดอินเตอร์เฟสกับสล็อต ISA 16 บิท....49

สารบัญรูป(ต่อ)

รูปที่ หน้า

4.11 ความสัมพันธ์ระหว่างรีจีสเตอร์ต่างๆสำหรับการกำหนดตำแหน่งและขนาดของข้อมูลภาพที่จะจัดเก็บ.....50

4.12 ตำแหน่งขาของสล๊อตคอมพิเตอร์.....51

4.13 บล็อกไดอะแกรมของวงจรกำหนดตำแหน่งและขนาดของข้อมูลภาพ.....67

4.14 ไทมิ่งไดอะแกรมในส่วนของ Line Select.....68

4.15 ไทมิ่งไดอะแกรมในส่วนของ Pixel Select.....69

4.16 บล็อกไดอะแกรมสำหรับการควบคุมหน่วยความจำแบบ SRAM.....70

4.17a ไทมิ่งไดอะแกรมสำหรับขั้นตอนในการอ่านข้อมูลของ SRAM ทั่วๆ ไป.....71

4.17b ไทมิ่งไดอะแกรมสำหรับขั้นตอนในการเขียนข้อมูลของ SRAM ทั่วๆ ไป.....71

4.18 ขั้นตอนการออกแบบบนซิฟเอฟพีจีเอ.....73

4.19a การใช้ภาษาวีเอชดีแอลอธิบายลักษณะพฤติกรรมของรีจิสเตอร์ขนาด 10 บิทและโมดูลของรีจิสเตอร์ 10 บิท.....73

4.19b การเชื่อมต่อโมดูลต่างๆเข้าด้วยกันเป็นวงจรกำหนดตำแหน่งและขนาดของภาพ.....74

5.1 แบบจำลองของวงจรอินเตอร์เฟส ISA แบบ 16 บิท.....76

5.2 ผลการจำลองการทำงานของชุดอินเตอร์เฟส ISA แบบ 16 บิท สำหรับโหมดการเขียนข้อมูล.....78

5.3 ผลการจำลองการทำงานของชุดอินเตอร์เฟส ISA แบบ 16 บิท สำหรับโหมดการอ่านข้อมูล.....79

5.4 แบบจำลองของวงจรเลือกเส้นสะแกนและเลือกพิกเซลที่จะทำการจัดเก็บ.....80

5.5 ผลการจำลองการทำงานของวงจรเลือกเส้นสะแกนและพิกเซลที่จะทำการจัดเก็บ.....81

5.6 ขยายผลการจำลองการทำงานเพื่อดูความสัมพันธ์ของสัญญาณ H-Sync กับ Clock (1)....81

5.7 ขยายผลการจำลองการทำงานเพื่อดูความสัมพันธ์ของสัญญาณ H-Sync กับ Clock (2)....81

5.8 แผงวงจรของเครื่องแปลงสัญญาณภาพทางการแพทย์โดยใช้เอฟพีจีเอ.....82

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

จากความต้องการในการนำข้อมูลภาพทางการแพทย์จากเครื่องมือแพทย์ เช่นภาพตัดขวางที่รับจากเครื่อง MRI หรือ CT มาทำการประมวลผลบนไมโครคอมพิวเตอร์วิธีการที่จะได้มาซึ่งข้อมูลดังกล่าวจะมี 2 วิธีคือ

1.ทำการบันทึกภาพที่ปรากฏยังมอนิเตอร์ของคอนโซลซึ่งจะต้องใช้คอมพิวเตอร์และโปรแกรมที่ซื้อมาจากบริษัทผู้ผลิตเครื่อง MRI หรือ CT ซึ่งมักมีราคาค่อนข้างสูง ข้อมูลภาพจะถูกจัดเก็บลงไฟล์ที่เป็นรูปแบบเฉพาะของบริษัท หากจะนำไฟล์ภาพเหล่านั้นมาใช้งานผู้ใช้จำเป็นจะต้องรู้รูปแบบที่เครื่องดังกล่าวใช้ด้วย นอกจากนี้เครื่องมือแพทย์ระบบดิจิตอลที่มีใช้อยู่ในอดีตจนถึงปัจจุบันจะมีความหลากหลายในเรื่องของรูปแบบข้อมูลที่ทำการจัดเก็บด้วย ดังนั้นการใช้ข้อมูลภาพจำเป็นที่จะต้องพึ่งซอฟท์แวร์ที่ทางบริษัทผู้ผลิตจัดหาให้ซึ่งมักมีราคาค่อนข้างสูง ทำให้เป็นข้อจำกัดของการใช้เครื่องมือแพทย์เหล่านั้น

2.ทำการแปลงสัญญาณจากสัญญาณวิดีโอที่ถูกส่งเข้าสู่จอมอนิเตอร์ของระบบ วิธีนี้มีข้อดีคือไม่จำเป็นต้องไปยุ่งเกี่ยวกับส่วนของฮาร์ดแวร์ของตัวเครื่องมือแพทย์แต่จำเป็นจะต้องทราบความถี่ของสัญญาณวิดีโอที่ส่งออกมาว่ามีค่าเท่าไรเพราะสัญญาณคอมโพซิทวิดีโอของเครื่องมือแพทย์จะไม่เท่ากับคอมโพซิทวิดีโอทั่วไป นอกจากนี้เครื่อง MRI หรือ CT ในแต่ละรุ่นแต่ละบริษัทก็จะมีสัญญาณคอมโพซิทวิดีโอที่แตกต่างกันอีกด้วย

1.2 วัตถุประสงค์ของการวิจัย

ในวิทยานิพนธ์ฉบับนี้เป็นการเสนอการออกแบบเครื่องแปลงสัญญาณภาพจากเครื่องมือแพทย์ โดยในการออกแบบจะใช้ซิฟเอฟพีจีเอเป็นตัวควบคุมกระบวนการทำงานทางดิจิตอลทุกอย่าง ในการออกแบบเครื่องแปลงสัญญาณภาพทางการแพทย์ด้วยเอฟพีจีเอทำให้ต้นทุนในการออกแบบต่ำ [1-2] ในการเก็บข้อมูลภาพสามารถเก็บให้อยู่ในรูปแบบของไฟล์ข้อมูลภาพตามมาตราฐานทั่วไป ซึ่งสามารถนำไฟล์ดังกล่าวไปใช้กับโปรแกรมแสดงภาพหรือประมวลผลภาพที่มีใช้อยู่ทั่วไปได้ ในการจัดเก็บข้อมูลภาพทางการแพทย์จากเครื่องมือแพทย์จะทำการจัดเก็บข้อมูลภาพจากสัญญาณคอมโพสิทวิดีโอที่ถูกส่งมาจากเครื่อง MRI หรือ CT โดยมีความละเอียดและระดับของสัญญาณที่ทำการจัดเก็บเป็นเช่นเดียวกับสัญญาณภาพที่ได้รับคือ 256 ระดับ และทำ

การจัดเก็บในรูปแบบของไฟล์ที่สามารถเปิดดูได้ด้วยโปรแกรมที่มีใช้อยู่ในปัจจุบัน ในการออกแบบเครื่องเก็บข้อมูลภาพจากเครื่องมือแพทย์ไม่ได้ระบุว่าสามารถใช้ได้กับเครื่อง MRI หรือ CT รุ่นใดยี่ห้อใดจะต้องออกแบบให้สามารถใช้ได้กับเครื่อง MRI หรือ CT ชนิดต่างๆได้ และนอกจากนี้ยังสามารถเลือกขนาดและตำแหน่งของข้อมูลภาพที่จะทำการจัดเก็บได้อีกด้วย

1.3 ขอบเขตของการวิจัย

ออกแบบเครื่องแปลงข้อมูลภาพจากเครื่องมือแพทย์โดยใช้ซิฟเอฟพีจีเอเป็นตัวควบคุมการทำงานซึ่งเครื่องเก็บข้อมูลภาพทางการแพทย์จะมีลักษณะเป็นแผงวงจรอิเล็กทรอนิกส์ที่อินเตอร์เฟสกับเครื่องไมโครคอมพิวเตอร์และมีการควบคุมด้วยโปรแกรมบนเครื่องไมโครคอมพิวเตอร์ โดยที่แผงวงจรอิเล็กทรอนิกส์ที่ออกแบบมาจะต้องมีความสามารถในการเก็บข้อมูลภาพได้สูงสุด 1024x1024 จุดต่อภาพ ระดับความเข้มของภาพ 8 บิตต่อจุดภาพ โดยใช้ตัวแปลงสัญญาณอนาลอกเป็นดิจิตอลที่มีความถี่ในการสุ่มสูงสุด 32 MHz หน่วยความจำในการเก็บข้อมูล 2 ชุดๆละ 512 Kbyte เชื่อมต่อกับคอมพิวเตอร์ผ่านทางสล๊อต ISA แบบ 16 บิต มีตำแหน่งในการอ้างอิงกับแผงวงจรที่ 300H ถึง 31FH และสามารถกำหนดขนาดและตำแหน่งของข้อมูลภาพที่จะทำการจัดเก็บได้

1.4 โครงสร้างของวิทยานิพนธ์

วิทยานิพนธ์นี้เป็นส่วนของงานเอกสารหลักของงานวิจัยนี้ซึ่งจะกล่าวถึงรายละเอียดต่างๆ ดังนี้

บทที่ 1 บทนำ

กล่าวถึงปัญหาและความเป็นมาของการออกแบบเครื่องแปลงข้อมูลภาพทางการแพทย์โดยใช้เอฟพีจีเอ วัตถุประสงค์และขอบเขตของการวิจัย

บทที่ 2 การจัดเก็บข้อมูลภาพ

กล่าวถึงลักษณะของสัญญาณภาพทางการแพทย์ที่ส่งมาจากเครื่อง MRI หรือ CT การแปลงสัญญาณภาพที่เป็นสัญญาณอนาลอกเป็นสัญญาณภาพดิจิตอล และการจัดเก็บข้อมูลภาพ

บทที่ 3 การออกแบบวงจรในลักษณะโครงสร้างและการบรรยายพฤติกรรม

กล่าวถึงการออกแบบวงจรโดยการอธิบายลักษณะพฤติกรรมของวงจร ประวัติความเป็นมาของภาษาวีเอชดีแอล ข้อกำหนดต่างๆ และโครงสร้างภายในเอฟพีจีเอแบบต่างๆ

บทที่ 4 การออกแบบและสร้างเครื่องแปลงสัญญาณภาพทางการแพทย์โดยใช้ FPGA

กล่าวถึงรายละเอียดของการออกแบบวงจรภายในชิฟเอฟพีจีเอ และขั้นตอนต่างๆของการออกแบบระบบบนชิฟเอฟพีจีเอ

บทที่ 5 ผลการทดลอง

ผลการทดลองในส่วนต่างๆของวงจรที่ได้ออกแบบด้วยวีเอชดีแอล ซึ่งผลการทดลองในส่วนนี้จะเป็นผลจากการจำลองการทำงาน (Simulation) และผลการทดลองที่ได้ทำการทดสอบกับเครื่อง MRI หรือ CT เปรียบเทียบกับภาพต้นแบบ

บทที่ 6 บทสรุป

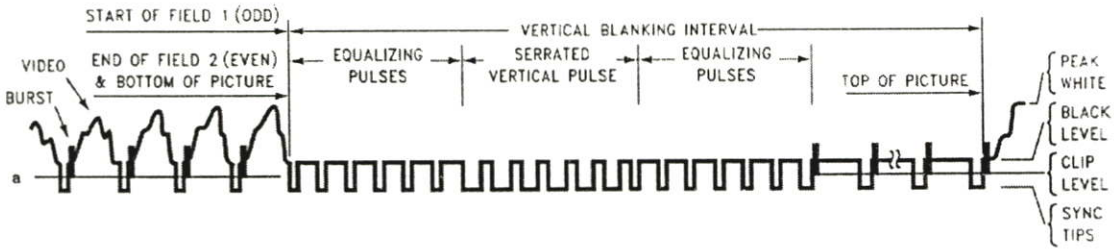
เป็นการสรุปผลการดำเนินงานทั้งหมด ปัญหาต่างๆที่เกิดขึ้น การแก้ไขปัญหาต่าง พร้อมทั้งเสนอแนวทางการพัฒนางานวิจัยต่อไป

บทที่ 2
การจัดเก็บข้อมูลภาพ

ในการออกแบบเครื่องแปลงสัญญาณภาพจำเป็นจะต้องทราบถึงลักษณะของสัญญาณภาพที่จะทำการแปลงข้อมูล เพราะฉะนั้นในบทนี้จะกล่าวถึงลักษณะของสัญญาณภาพ และแนวคิดของการออกแบบระบบการจัดเก็บข้อมูลภาพ

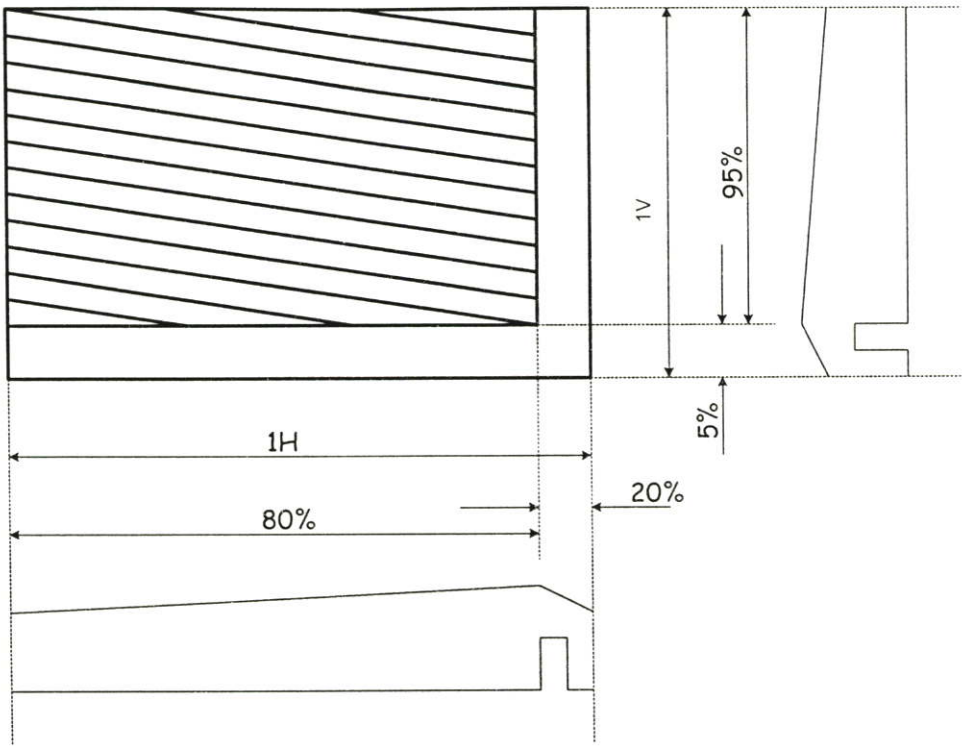
2.1 ลักษณะของสัญญาณภาพ [3]

สัญญาณภาพโดยทั่วไปจะมีลักษณะเป็นสัญญาณคอมโพสิทวิดีโอคือจะประกอบไปด้วยสัญญานภาพ สัญญาณซิงค์ สัญญาณแบลงกิ้ง สัญญาณอิควัลไลซิ่ง โดยลักษณะของสัญญาณดังกล่าวแสดงดังรูปที่ 2.1



รูปที่ 2.1 ลักษณะของสัญญาณวิดีโอ

สัญญาณภาพที่แสดงข้างบนจะเป็นสัญญาณภาพรวมที่ประกอบด้วย เส้นสแกนทางแนวนอนและเส้นสแกนทางแนวตั้ง โดยเส้นสแกนทั้งสองจะประกอบด้วยสัญญาณซิงค์ สัญญาณแบลงกิ้งและสัญญาณภาพ สัญญาณดังกล่าวจะถูกส่งไปยังมอนิเตอร์ ทำให้เกิดการสแกนทางหน้าจอมอนิเตอร์ ทำให้ปรากฏเป็นภาพเมื่อเปรียบเทียบกับการสแกนของเส้นสแกนทั้งแนวนอนและแนวตั้ง สามารถแสดงให้เห็นได้ ดังรูปที่ 2.2



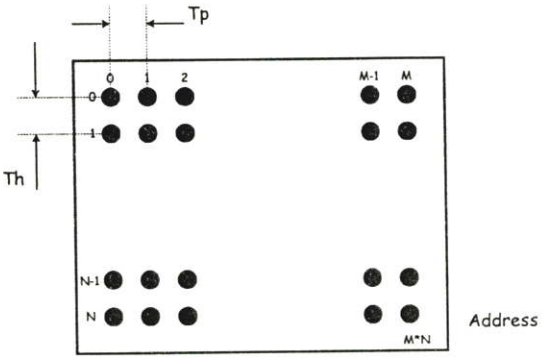
รูปที่ 2.2 เปรียบเทียบการสแกนที่หน้าจอของมอนิเตอร์กับสัญญาณที่ใช้ควบคุมการสแกนทั้งสองแกน

การจัดเก็บสัญญาณภาพจากคอมโพสิทวิดีโอซึ่งมีลักษณะเป็นสัญญาณอนาลอกเข้าไปเก็บไว้ยังหน่วยความจำมีความจำเป็นอย่างยิ่งที่จะต้องทำการแปลงสัญญาณภาพดังกล่าวจากสัญญาณอนาลอกเป็นสัญญาณดิจิตอล เพื่อให้สามารถจัดเก็บเข้าไปยังหน่วยความจำได้ ในการแปลงสัญญาณอนาลอกนั้นก็ต้องประกอบด้วยส่วนของการสุ่มตัวอย่างสัญญาณ และการควัลไตสัญญาณ (Quantization) สัญญาณภาพที่ถูกเปลี่ยนเป็นสัญญาณดิจิตอลแล้ว สามารถที่จะถูกจัดเก็บลงไปยังหน่วยความจำได้ แต่ในการสแกนทางด้านแนวนอนของสัญญาณภาพนั้นจะใช้เวลาในการสแกนที่สั้นมาก ดังนั้นจึงทำให้เกิดปัญหาในขั้นตอนของการเปลี่ยนสัญญาณภาพจากสัญญาณอนาลอกให้เป็นสัญญาณดิจิตอล (Analog to Digital Converter) รวมทั้งการเขียนข้อมูลเข้าไปเก็บยังหน่วยความจำ

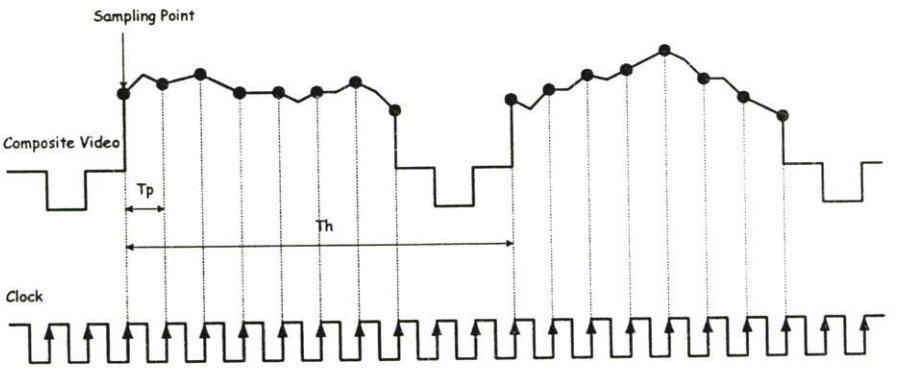
ปัญหาแรกคือปัญหาในเรื่องของการแปลงสัญญาณอนาลอกเป็นดิจิตอล การที่สัญญาณภาพมีเวลาที่ใช้ในการสแกนทางด้านแนวนอนที่สั้นมาก ในการจัดเก็บเพื่อให้ได้ความละเอียดของภาพคงเดิมดังเช่นที่แสดงทางหน้าจอมอนิเตอร์ จำเป็นจะต้องใช้วงจรแปลงสัญญาณ อนาลอกเป็นดิจิตอลที่มีอัตราในการสุ่มตัวอย่างที่สูงเพียงพอกับสัญญาณดังกล่าว

ส่วนปัญหาที่สองนั้นเกิดขึ้นจากข้อมูลที่ถูกแปลงแล้วจะนำไปเขียนยังหน่วยความจำ ซึ่งในการเขียนข้อมูลดังกล่าวนี้จะต้องมีการจัดเวลาอย่างเหมาะสมเพื่อไม่ให้เกิดปัญหาขึ้นในขั้นตอน

ของการเขียนข้อมูลลงหน่วยความจำ ในการจัดเก็บข้อมูลภาพลงสู่หน่วยความจำนั้นจะอาศัยหลักการจัดเก็บภาพลงสู่หน่วยความจำแบบแอดเดรสเป็นแบบต่อเนื่อง ดังแสดงให้เห็นดังรูปที่ 2.3



รูปที่ 2.3a ตำแหน่งของพิกเซลทางหน้าจอแสดงผลและตำแหน่งแอดเดรสของหน่วยความจำ



รูปที่ 2.3b เวลาในการสุ่มสำหรับเส้นสแกนของสัญญาณคอมโพซิทวีดีโอ

โดยที่

Tp : ระยะห่างของแต่ละจุดภาพในแนวแกนนอน

Th : ระยะห่างของแต่ละจุดภาพในแนวแกนตั้ง

M : จำนวณจุดภาพในแนวแกนนอน

N : จำนวณจุดภาพในแนวแกนตั้ง

จากรูปที่ 2.3 ถ้าหากเป็นสัญญาณภาพที่ได้รับจากสัญญาณคอมโพวิทวีดีโอทั่วๆไป ตัวอย่างเช่นสัญญาณวีดีโอในระบบ NTSC (The National Television System Committee) จะพบว่าใน 1 เส้นสแกนทางแนวนอนจะใช้เวลาทั้งสิ้น 64 ไมโครวินาที แต่เนื่องจากส่วนที่เป็นสัญญาณภาพจริงๆ จะประมาณ 80% ของสัญญาณทั้งหมดคือ 64x0.8 = 51.2 ไมโครวินาที ส่วนที่เหลือดังกล่าวจะเป็นสัญญาณในช่วงแบลงกิ้งและถ้าหากต้องการที่จะทำการจัดเก็บสัญญาณในแต่ละเส้น

สแกนทางแนวนอนให้มีจุดภาพทั้งสิ้น 512 จุดภาพ ก็จำเป็นต้องใช้เวลาในการเขียนข้อมูลภาพแต่ละจุดภาพลงสู่หน่วยความจำเท่ากับ

$$\text{เวลาในการเขียนข้อมูลภาพแต่ละจุดภาพ} = \frac{51.2}{512} \text{ ไมโครวินาที}$$

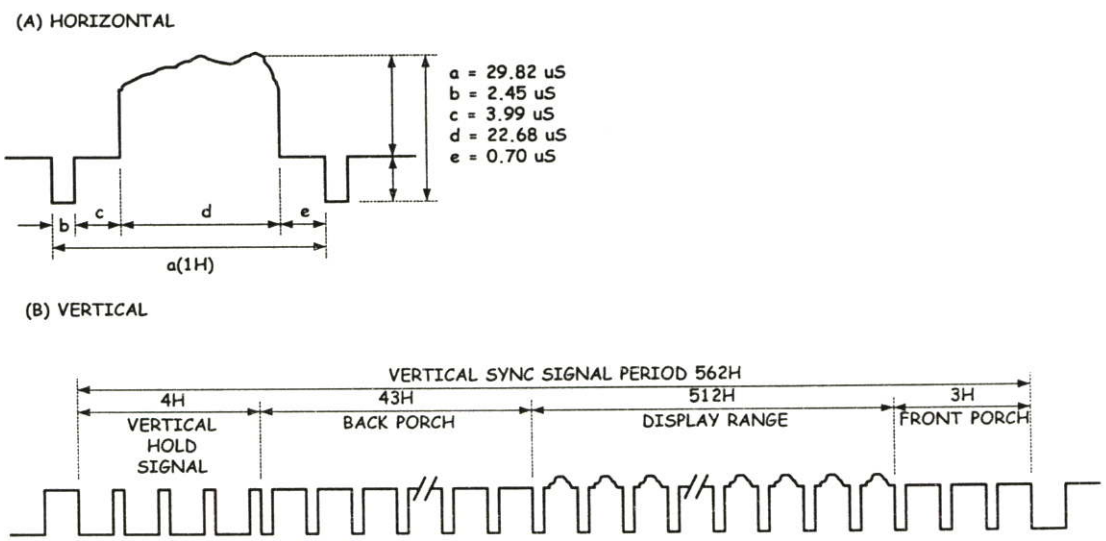
$$= 100 \text{ ไมโครวินาที}$$

เวลาดังกล่าวเป็นค่า Access Time ของหน่วยความจำที่นำมาใช้ โดยค่าความจุของหน่วยความจำที่ใช้จะมีค่าเท่ากับผลคูณของจำนวนเส้นสแกนต่อหนึ่งเฟรมกับจำนวนจุดภาพในหนึ่งเส้นสแกนทางแนวนอน คือ

ความจุของหน่วยความจำ = จำนวณเส้นสแกน x จำนวนจุดภาพในหนึ่งเส้นสแกนทางแนวนอน

= 512x512

= 256 KByte



รูปที่ 2.4 ลักษณะสัญญาณภาพที่ได้จากเครื่อง CT ยี่ห้อ HITACHI [4]

อย่างไรก็ดีลักษณะของสัญญาณภาพที่ได้จากเครื่องมือแพทย์จะมีลักษณะเดียวกันกับสัญญาณวีดีโอดังที่กล่าวมาแล้วในตอนต้น จะมีส่วนที่แตกต่างกันในเรื่องคาบเวลาของสัญญาณ รวมทั้งลักษณะที่ใช้ในการสแกน ลักษณะของสัญญาณจากเครื่อง CT ยี่ห้อ HITACHI รุ่น W-450

แสดงให้เห็นดังรูปที่ 2.4 จากสัญญาณที่แสดงให้เห็นข้างบนทำให้สรุปเป็นค่าต่างๆ ดังในตารางที่ 2.1

ตารางที่ 2.1 รายละเอียดของสัญญาณภาพจากเครื่อง CT ยี่ห้อ HITACHI รุ่น W-450

รุ่น W-450	
ระบบการสะแกนสัญญาณภาพ	Noninterlaced
ความถี่เส้นสะแกนทางแนวนอน	33.5345 KHz
ความถี่เส้นสะแกนทางแนวตั้ง	59.67Hz
ความละเอียดของภาพ	648H x 512V
เส้นสะแกนทางแนวตั้ง 1 เส้น	
จำนวนของเส้นสะแกนทั้งหมด	562H
Front Porch	3H
ความกว้างของสัญญาณซิงค์	4H
Back Porch	43H
จำนวนของเส้นสะแกนที่แสดงทางหน้าจอ	512H
เส้นสะแกนทางแนวนอน 1 เส้น	
คาบเวลาของเส้นสะแกนทางแนวนอน 1 เส้น	29.82uS
Front Porch	0.70uS
ความกว้างของสัญญาณซิงค์	2.45uS
Back Porch	3.99uS
ช่วงระยะเวลาที่ใช้ในการแสดงผล	22.68uS
Aspect Ratio	5.06:4

จากสัญญาณของเครื่อง CT ตัวอย่างที่แสดงข้างต้นนั้น พบว่าความกว้างของเส้นสแกนที่แสดงออกทางหน้าจอมอนิเตอร์มีค่าเท่ากับ 22.68uS และจำนวนพิกเซลที่เกิดขึ้นจะมีค่าเท่ากับ 648 พิกเซล ดังนั้นในการสุ่มตัวอย่างสัญญาณเพื่อไม่ให้สูญเสียในรายละเอียดของสัญญาณ จำเป็นต้องใช้อัตราในการสุ่มข้อมูลเท่ากับ 22.68uS / 648 = 35 nS หรือ 28.57 MHz จากค่าอัตราการสุ่มตัวอย่างนี้ สามารถที่จะเลือกใช้ชิพไอซี A/D เบอร์ TDA8708A ของบริษัทฟิลิปส์ ซึ่งมีอัตราในการสุ่มตัวอย่างสูงสุดที่ 32 MHz จากตัวอย่างที่ผ่านมา เป็นตัวอย่างของเครื่อง CT เพียงรุ่นเดียว ต่อจากนี้เป็นการแสดงให้เห็นเป็นตารางสรุปเกี่ยวกับลักษณะของสัญญาณที่ได้จากเครื่อง CT ยี่ห้อ HITACHI หลายๆ รุ่นที่ใช้อยู่ในปัจจุบัน ดังตารางที่ 2.2

ตารางที่ 2.2 รายละเอียดของสัญญาณภาพที่ได้รับจากเครื่อง CT ยี่ห้อ HITACHI.รุ่นต่างๆ

	W3-10,20,30, W4010-60	W450-10,20 LOW LINE CRT	W450-20HIGHT LINE,W550,W1000
การสะแกนสัญญาณภาพ	Noninterlaced	Interlaced	Noninterlaced
ความถี่ทางแนวนอน	21.04 KHz	15.73 KHz	33.5345 KHz
ความถี่ทางแนวตั้ง	59.7 Hz	56.1 Hz	59.67 Hz
ความละเอียดของภาพ	432Hx320V	648Hx512V	648Hx512V
เส้นสะแกนทางแนวตั้ง 1 เส้น			
จำนวนเส้นสะแกนทั้งหมด	352.5 H	561 H	562 H
Front Porch	3 H	3 H	3 H
ความกว้างสัญญาณซิงค์	3 H	3H	4 H
Back Porch	26 H	15 H	43 H
จำนวนเส้นสะแกนหน้าจอ	320 H	256.5x2 H	512 H
เส้นสะแกนทางแนวนอน 1 เส้น			
คาบเวลาที่สะแกน	47.52 uS	63.56 uS	29.82 uS
Front Porch	0.72 uS	4.76 uS	0.70 uS
ความกว้างสัญญาณซิงค์	3.96 uS	4.76 uS	2.45 uS
Back Porch	3.96 uS	8.68 uS	3.99 uS
คาบเวลาหน้าจอแสดงผล	38.88 uS	45.36 uS	22.68 uS
ค่า Aspect Ratio	1.35 : 1	5.06 : 4	5.06 : 4

จากตารางที่แสดงข้างต้นเป็นการยืนยันได้ว่าสัญญาณของเครื่องมือแพทย์ที่มีใช้อยู่นั้นจะมีความหลากหลายไม่ได้มีมาตรฐานที่แน่นอน ดังนั้นในการออกแบบระบบแปลงข้อมูลภาพทางการแพทย์จำเป็นอย่างยิ่งที่จะต้องไม่กำหนดความถี่ใดความถี่หนึ่งตายตัวโดยเฉพาะ จะต้องออกแบบให้สามารถปรับแต่งได้ง่าย เพื่อให้เหมาะสมกับสัญญาณภาพที่ต้องการจัดเก็บ

2.2 การเปลี่ยนข้อมูลจากสัญญาณอนาลอกเป็นสัญญาณดิจิตอล

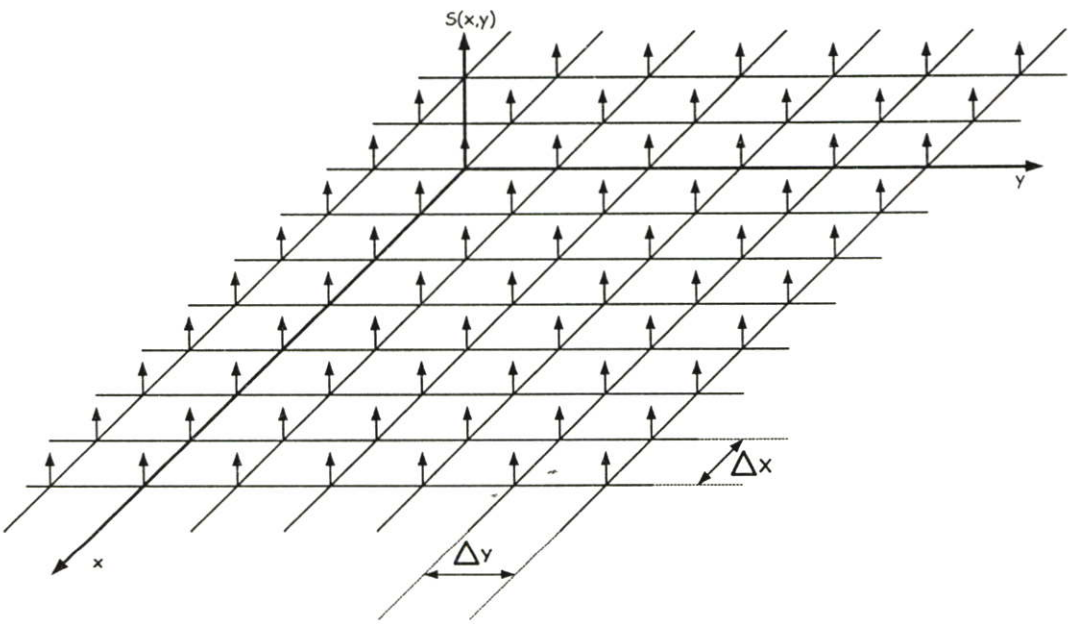
ในการแปลงสัญญาณภาพที่ต่อเนื่องให้เป็นข้อมูลภาพทางดิจิตอลด้วยการดิจิไตซ์ ในการดิจิไตซ์ในสปาเชียลโดเมน (x, y) จะเรียกว่า "การสุ่มภาพ" (image sampling) และในขณะที่การดิจิไตซ์ทางแอมปลิจูดจะเรียกว่า "การควอนไตว์ระดับเทา" (gray-level quantization) [5]

2.2.1 การสุ่ม (Sampling)

ฟังก์ชั่นของการสุ่มสามารถเขียนเป็นฟังก์ชันทางคณิตศาสตร์ ด้วยอิมพัลซ์ฟังก์ชัน $\delta(x, y)$ ที่กระทำกับสัญญาณภาพซึ่งสามารถกำหนดได้ดังนี้

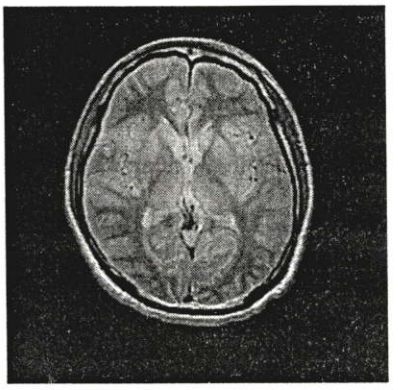
$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\,\delta(x-x_0, y-y_0)\,dxdy = f(x_0, y_0)$$

ฟังก์ชันของการสุ่มสองมิตินั้นจะประกอบด้วยพัลส์เทรนที่ห่างกัน Δx ในทิศทาง x และช่วงห่าง Δy ในทิศทาง y ดังแสดงในรูปที่ 2.5 โดยฟังก์ชันของภาพจะเป็น $f(x, y)$ และมีค่า x และ y ที่ต่อเนื่อง การสุ่มทำได้โดยการคูณฟังก์ชันอิมพัลซ์ $\delta(x, y)$ กับฟังก์ชันภาพ $f(x, y)$ จะได้ $\delta(x, y) f(x, y)$

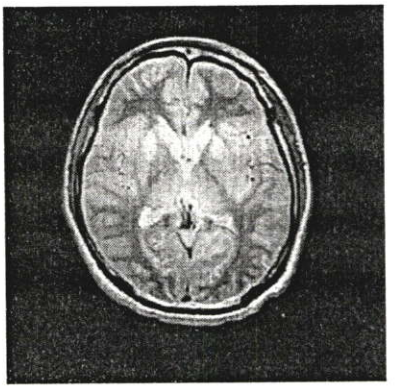


รูปที่ 2.5 ฟังก์ชันของการสุ่ม 2 มิติ

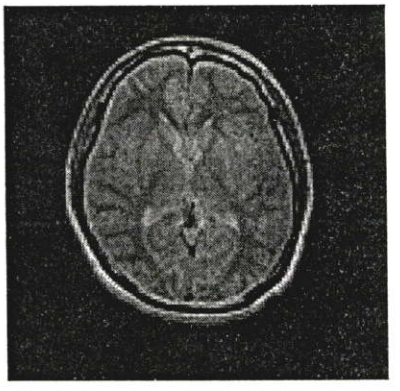
ความถี่ที่ใช้ในการสุ่มในระบบของภาพนั้นจะเป็นตัวบ่งบอกถึงขนาดของภาพ ภาพที่ผ่านการสุ่มด้วยความถี่สูงๆ ก็จะได้จำนวนจุดภาพที่มากขึ้น สามารถเก็บรายละเอียดได้มากขึ้น ในกรณีที่จำนวนจุดภาพมีค่าน้อยจะทำให้เกิดผลอย่างหนึ่งคือ การเกิดซ้ำกันของจุดภาพ (Pixel Replication) ทำให้เห็นภาพเป็นบล็อกๆ (Chechker-Board Effect) [5]



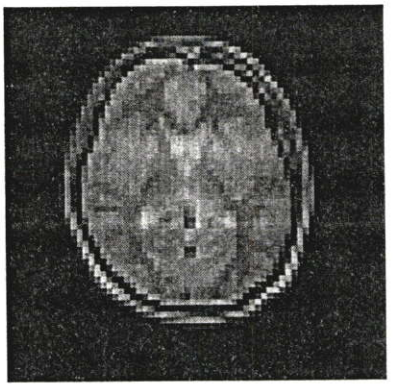
รูปที่ 2.6a ภาพขนาด 512x512 จุดภาพ



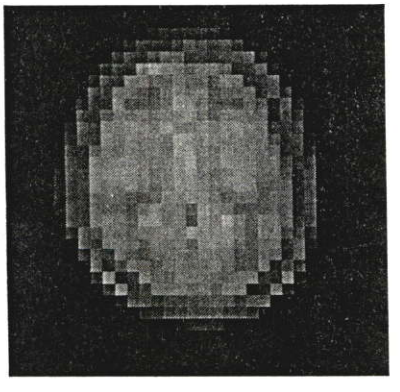
รูปที่ 2.6b ภาพขนาด 256x256 จุดภาพ



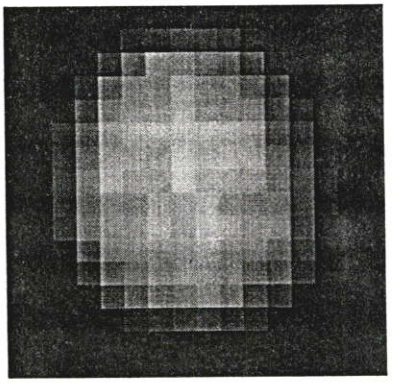
รูปที่ 2.6c ภาพขนาด 128x128 จุดภาพ



รูปที่ 2.6d ภาพขนาด 64x64 จุดภาพ



รูปที่ 2.6e ภาพขนาด 32x32 จุดภาพ



รูปที่ 2.6f ภาพขนาด 16x16 จุดภาพ

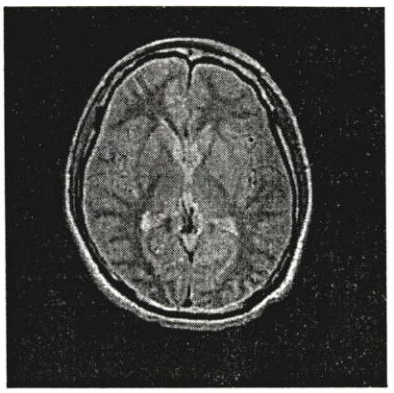
รูปที่ 2.6 แสดงภาพที่มีจำนวนจุดขนาดต่างๆ ซึ่งจะเห็นว่าภาพที่มีจำนวนจุดที่น้อยต่อภาพนั้นจะส่งผลให้เกิดมีการซ้ำกันของจุดภาพส่งผลให้ภาพเกิดเป็นบล็อกๆ ขึ้น ซึ่งสามารถสังเกตเห็นได้จากภาพที่ 2.6 (d) , (e) และ (f) ซึ่งสามารถมองเห็นลักษณะของการเกิดบล็อก ได้อย่างชัดเจน

2.2.2 **การควอนไตซ์** (Quantization)

การควอนไตซ์เป็นการเข้ารหัสของระดับที่ผ่านการสุ่ม เพื่อจัดเข้าระดับที่เป็นมาตรฐานหรือเป็นไปตามที่ต้องการ แต่ในทางด้านการประมวลผลภาพการควอนไตซ์เป็นการจัดระดับของสัญญาณภาพที่ผ่านการสุ่มให้อยู่ในระดับเทา จำนวนระดับเทาที่ใช้นั้นขึ้นอยู่กับจำนวนบิตของข้อมูลดิจิตอล จำนวนของระดับเทาที่ใช้นั้นเท่ากับสองยกกำลังตามจำนวนบิต แสดงตามสมการดังนี้

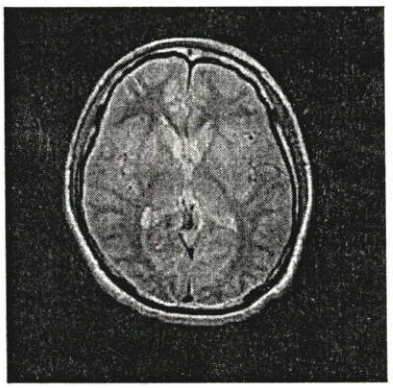
$$G = 2^m$$

เมื่อ G เท่ากับจำนวนระดับเทาและ m เป็นจำนวนบิตของข้อมูลดิจิตอลที่ใช้ ตัวอย่างเช่น ตัวแปลงสัญญาณอนาลอกเป็นดิจิตอลที่ให้ข้อมูลดิจิตอลจากการแปลงแล้ว 8 บิต ทำให้ได้ระดับเทาที่แตกต่างกัน 256 ระดับ ระดับของการควอนไตซ์นั้นจะมีผลต่อภาพที่เก็บ ถ้าใช้ระดับการควอนไตซ์ที่มีจำนวนระดับความแตกต่างน้อยหรือกล่าวอีกนัยหนึ่งคือจำนวนบิตของข้อมูลดิจิตอลที่น้อยกว่าปกตินั้นจะทำให้เกิดความผิดพลาดของข้อมูลสูง สาเหตุที่เป็นเช่นนี้เพราะว่าความห่างของระดับนั้นมีมาก เวลาทำการควอนไตซ์จะเกิดการปรับค่าที่ได้จากการสุ่มให้เข้าสู่ระดับที่กำหนด ถ้าข้อมูลที่ได้จากการสุ่มห่างจากระดับที่กำหนดมากเท่าใดก็จะทำให้เกิดการผิดพลาดมากขึ้นเท่านั้น หรือกล่าวอีกนัยหนึ่งคือเราไม่มีระดับเทาที่แทนค่าของระดับความเข้มของภาพได้หมด ส่วนจำนวนระดับเทาหรือบิตของข้อมูลภาพที่ใช้นั้นปกติไม่ควรต่ำกว่า 64 ระดับเทาหรือจำนวนบิตไม่ควรต่ำกว่า 6 บิตจึงเหมาะสมกับสายตาของคนเราที่จะไม่รู้สึกว่าเกิดการคลาดเคลื่อนขึ้นกับภาพ แต่ถ้าใช้จำนวนระดับที่ต่ำกว่านี้จะทำให้เกิดผลอย่างหนึ่งที่เรียกว่า "ขอบเทียม" [5] (false contour) รูปที่ 2.7 แสดงภาพที่มีการใช้จำนวนบิตต่อจุดภาพขนาดต่างๆ จากรูปที่2.7b และรูปที่2.7c นั้นแม้มีการใช้จำนวนบิตของจุดภาพที่น้อยลง แต่สายตาเราก็ยังไม่สามารถตรวจจับความแตกต่างของภาพได้ แต่ถ้ามีการลดจำนวนบิตของจุดภาพลงไปอีก จะทำให้เราสามารถจับความผิดเพี้ยนของภาพที่เกิดขึ้นได้ ดังที่แสดงในรูปที่ 2.7d , 2.7e , 2.7f , 2.7g และ 2.7h ซึ่งรูปที่ 2.7h นั้นจะกลายเป็นภาพสองระดับไปเพราะมีการใช้จำนวนบิตเพียง 1 บิตต่อจุดภาพ

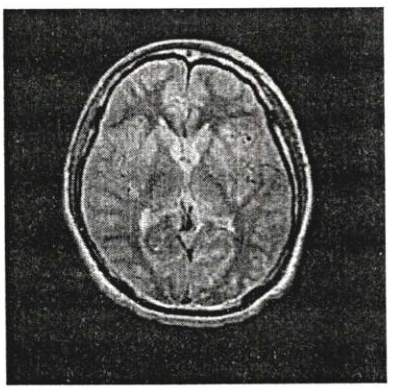


รูปที่ 2.7a ความละเอียด 8 บิตต่อจุดภาพ

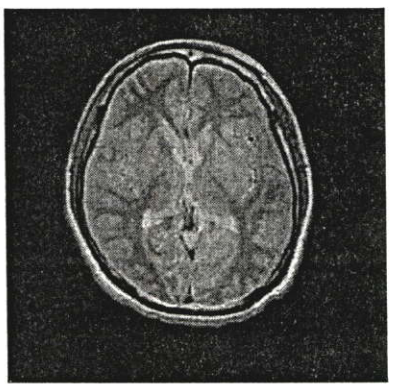
รูปที่ 2.7b ความละเอียด 7 บิตต่อจุดภาพ



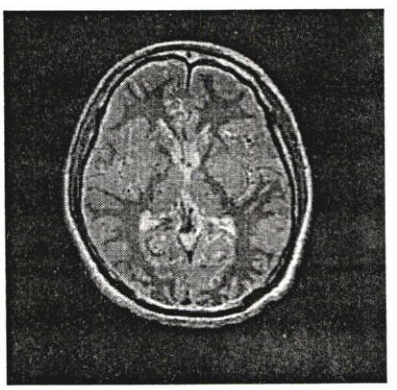
รูปที่ 2.7c ความละเอียด 6 บิตต่อจุดภาพ



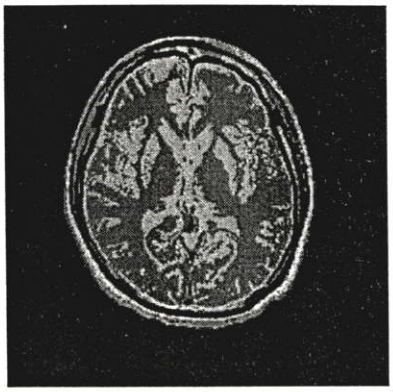
รูปที่ 2.7d ความละเอียด 5 บิตต่อจุดภาพ



รูปที่ 2.7e ความละเอียด 4 บิตต่อจุดภาพ



รูปที่ 2.7f ความละเอียด 3 บิตต่อจุดภาพ



รูปที่ 2.7g ความละเอียด 2 บิตต่อจุดภาพ

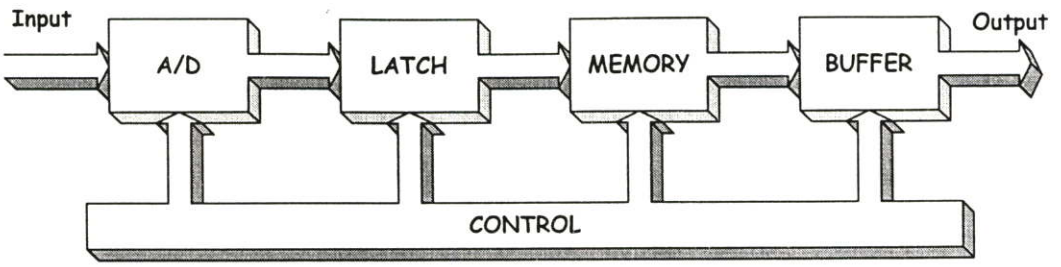
รูปที่ 2.7h ความละเอียด 1 บิตต่อจุดภาพ

ในการกำหนดขนาดของภาพและระดับเทาของภาพสำหรับแผงวงจรเก็บข้อมูลภาพจะต้องพิจารณาให้เหมาะสมกับงานที่จะใช้ซึ่งโดยรวมแล้วเป็นการกำหนดรายละเอียดของภาพ (Resolution of Image) ถ้ากำหนดภาพที่มีรายละเอียดสูงๆ ก็จะได้คุณภาพของภาพที่ดีแต่ทำให้ต้องใช้หน่วยความจำในการเก็บข้อมูลภาพที่มีขนาดใหญ่มาก หรือถ้ากำหนดรายละเอียดของภาพต่ำก็จะสามารถใช้หน่วยความจำขนาดเล็กแต่อาจไม่ได้รายละเอียดเท่าที่ควร ฉะนั้นการออกแบบแผงวงจรเก็บข้อมูลภาพควรให้มีความเหมาะสมในเรื่องคุณสมบัติของภาพที่ต้องการจัดเก็บด้วย

2.3 ลักษณะหน่วยความจำที่ใช้ในการจัดเก็บข้อมูล [6]

การจัดเก็บข้อมูลของสัญญาณภาพที่ได้กล่าวมาแล้วในตอนต้น ปัญหาที่พบทั้งสองปัญหาคือในเรื่องของ A/D และหน่วยความจำที่ใช้ในการจัดเก็บ สามารถทำการแก้ไขได้ จากการรวบรวมข้อมูลเกี่ยวกับรายละเอียดของสัญญาณภาพ ของเครื่องมือแพทย์ที่มีใช้อยู่ ความถี่ที่ใช้ในการสุ่มตัวอย่าง 32 MHz เป็นความถี่ที่สูงเพียงพอที่ครอบคลุมเครื่องมือแพทย์ได้ทั้งหมด ดังนั้นปัญหาในส่วนของ A/D จึงสามารถแก้ไขได้ โดยการใช้ชิพ A/D เบอร์ TDA8708A

ส่วนปัญหาในเรื่องของหน่วยความจำที่จะนำมาใช้ในการจัดเก็บข้อมูลภาพจากเครื่องมือแพทย์นั้นจะใช้หน่วยความจำแบบ Static RAM เบอร์ K6R4008CIC-JC12 ซึ่งเป็นหน่วยความจำที่มีค่า Access Time เท่ากับ 12nS มีความจุเท่ากับ 512 KByte และเป็นการจัดเก็บข้อมูลลงสู่หน่วยความจำแบบโดยตรง วิธีนี้เป็นวิธีที่มีความสะดวกและง่ายที่สุด เพราะจะใช้อุปกรณ์ที่ประกอบในวงจรน้อยที่สุด ทำให้ง่ายในการออกแบบแก้ไข หลักการเก็บข้อมูลภาพด้วยวิธีดังกล่าวนี้แสดงให้เห็นดังรูปที่ 2.8

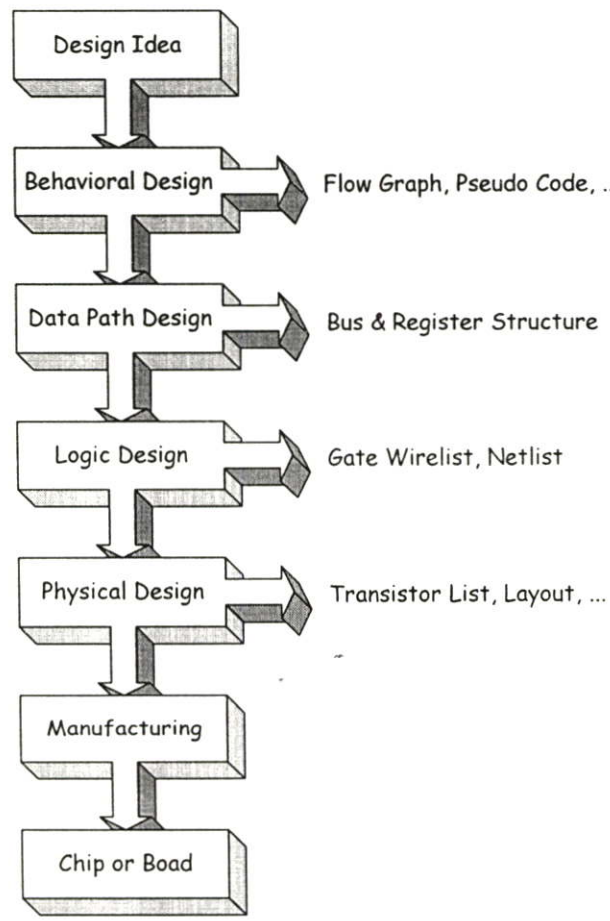


รูปที่ 2.8 ลักษณะการต่อหน่วยความจำเข้ากับระบบโดยตรง

บทที่ 3
การออกแบบวงจรในลักษณะโครงสร้างและการบรรยายพฤติกรรม

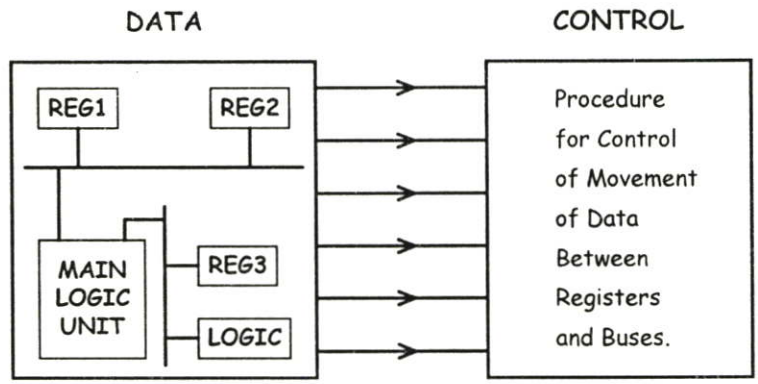
ความซับซ้อนและขนาดของระบบดิจิตอลในปัจจุบันได้เพิ่มมากขึ้นทุกขณะ ส่งผลให้มีการนำคอมพิวเตอร์เพื่อช่วยในการออกแบบหรือ CAD มาใช้ในขบวนการออกแบบฮาร์ดแวร์เพิ่มขึ้นเช่นกัน อีกทั้งอุปกรณ์และวิธีการออกแบบใหม่ๆ ก็ถูกพัฒนาขึ้นมาเพื่อช่วยอำนวยความสะดวกให้กับนักออกแบบมากขึ้นด้วย สำหรับภาษาบรรยายพฤติกรรมฮาร์ดแวร์ (HDL : Hardware Description Language) ก็เป็นเครื่องมืออย่างหนึ่งที่ได้รับการพัฒนามาอย่างต่อเนื่องเพื่อช่วยให้การปรับปรุงขบวนการออกแบบระบบดิจิตอลเป็นไปอย่างมีประสิทธิภาพ

3.1 การออกแบบระบบดิจิตอล [7]



รูปที่ 3.1 ขั้นตอนการออกแบบระบบดิจิตอล

ในการออกแบบระบบดิจิตอล เริ่มตั้งแต่การกำหนดแนวความคิดเบื้องต้นจนกระทั่งได้ออกมาเป็นอุปกรณ์ฮาร์ดแวร์ที่ใช้งานได้จะต้องผ่านขั้นตอนต่างๆ มากมาย และในแต่ละขั้นตอนผู้ออกแบบจะต้องตรวจสอบผลลัพธ์ในแต่ละขั้นก่อนเข้าสู่กระบวนการออกแบบในขั้นต่อไป รูปที่ 3.1 แสดงขั้นตอนปกติที่ใช้ในการออกแบบระบบดิจิตอลทั่วไป ขั้นแรกผู้ออกแบบจะกำหนดแนวความคิดในการออกแบบแล้วทำการพัฒนาให้สามารถนำมาใช้ได้อย่างสมบูรณ์ ซึ่งภายในขั้นตอนนี้ผู้ออกแบบจำเป็นต้องสร้างรูปแบบระบบในเชิงพฤติกรรมขึ้นมาตรวจสอบซึ่งอาจจะเป็นผังงานแสดงแบบ (Flow Graph) หรือรหัสคำสั่งเทียม (Pseudo Code) ก็ได้ ขั้นตอนต่อไปเป็นการออกแบบระบบเส้นทางของข้อมูล (Bus) ผู้ออกแบบจะกำหนดส่วนประกอบของรีจิสเตอร์และวงจรลอจิกที่จำเป็นทั้งหมดเพื่อนำมาประกอบเป็นระบบที่สมบูรณ์ โดยแต่ละองค์ประกอบสามารถเชื่อมต่อกันด้วยบัสหนึ่งหรือสองทิศทาง (Unidirectional or Bidirectional Bus) ส่วนกระบวนการในการควบคุมการเคลื่อนย้ายข้อมูลระหว่างรีจิสเตอร์และวงจรลอจิกจะขึ้นอยู่กับพฤติกรรมของระบบที่กำหนดไว้ดังรูปที่ 3.2



รูปที่ 3.2 การออกแบบระบบเส้นทางของข้อมูล

ขั้นตอนถัดมาเป็นการออกแบบวงจรลอจิก ซึ่งจะเกี่ยวข้องกับการนำเกทดิจิตอลพื้นฐานและฟลิปฟลอป (flip-flop) มาประกอบเป็นอุปกรณ์ย่อยต่างๆ เช่น รีจิสเตอร์เก็บข้อมูล บัสวงจรลอจิก และส่วนควบคุมฮาร์ดแวร์ ซึ่งผลลัพธ์ที่ได้ในขั้นตอนนี้จะเป็นเครือข่ายของการโยงใยระหว่างเกทและฟลิปฟลอปนั่นเอง การออกแบบในขั้นตอนถัดไปเป็นการเปลี่ยนเครือข่ายการโยงใยที่ได้จากขั้นตอนที่แล้วให้เป็นลำดับของทรานซิสเตอร์ (Transistor List) และ Layout ซึ่งขั้นตอนนี้จะเกี่ยวข้องโดยตรงกับการจัดวางทรานซิสเตอร์หรือไลบรารีเซลล์เพื่อแทนเกทและฟลิปฟลอปต่างๆ และในขั้นตอนสุดท้ายจะเป็นการส่งระบบที่ออกแบบไว้ไปทำการเจือสารที่โรงงานเพื่อผลิตออกมาเป็นวงจรรวมในที่สุด

3.2 ประวัติความเป็นมาของภาษาวีเอชดีแอล

วีเอชดีแอล (VHDL) ย่อมาจากคำว่า VHSIC Hardware Description Language (VHSIC : Very High Speed Integrated Circuit) เป็นภาษาโปรแกรมระดับสูง (High Level Language) ที่ใช้สำหรับการออกแบบฮาร์ดแวร์ในระบบดิจิตอล ตัวของภาษาสามารถบรรยายพฤติกรรมการทำงานในรูปของลำดับชั้น (Hierarchy) และสามารถเขียนได้หลายรูปแบบ ด้วยเหตุผลนี้จึงทำให้ภาษาวีเอชดีแอลเป็นเครื่องมือที่ใช้ออกแบบตั้งแต่ขั้นตอนบนสุด คือ แนวความคิดที่จะแก้ปัญหา ลงไปทีละขั้นจนถึงขั้นตอนของการสร้างวงจรจริง และตัวภาษาก็เปิดโอกาสให้วิศวกรได้พัฒนาและจำลองการทำงานของรูปแบบฟังก์ชันการทำงานของวงจรอย่างสังเขป โดยยังไม่ต้องคำนึงถึงรายละเอียดเกี่ยวกับโครงสร้างวงจรจริง นอกจากนั้นวีเอชดีแอลยังเป็นภาษาที่สนับสนุนลักษณะต่างๆ ของระบบดิจิตอลที่มีความซับซ้อนได้ทั้งหมด ดังนั้นวีเอชดีแอลจึงเป็นภาษาที่น่าสนใจในการศึกษาและนำไปใช้งานเป็นอย่างยิ่ง

วิวัฒนาการของวีเอชดีแอลเริ่มต้นประมาณปี ค.ศ. 1981 เมื่อกระทรวงกลาโหมสหรัฐอเมริกา หรือ DoD (Department of Defense) ได้พยายามปรับปรุงอุปกรณ์อิเล็กทรอนิกส์และคอมพิวเตอร์ที่ใช้ในกิจการทางทหารให้มีความทันสมัยมากขึ้น ประกอบกับเทคโนโลยีทางด้านไมโครอิเล็กทรอนิกส์มีการพัฒนาไปอย่างรวดเร็วดังจะเห็นได้จากการนำวงจรดิจิตอลหลายๆ วงจรมาทำการผลิตอยู่บนแผ่นซิลิกอนที่มีพื้นที่เพียง 1 – 2 ตารางเซนติเมตรเท่านั้น ซึ่งเป็นผลให้ประสิทธิภาพในการทำงานของวงจรสูงขึ้น ตลอดจนความน่าเชื่อถือในการทำงานและความคงทนต่อสภาพแวดล้อมสูง แต่เนื่องจากในขณะนั้นขั้นตอนของการออกแบบ การผลิต และการตรวจสอบวงจรต้นแบบ เป็นขบวนการที่ต้องใช้วิศวกร และเวลาในดำเนินการมาก ฉะนั้นทาง DoD จึงจัดตั้งโครงการขึ้นมาเพื่อศึกษาวิธีการที่ช่วยในการพัฒนาวงจรอิเล็กทรอนิกส์ โดยเฉพาะอย่างยิ่งวงจรระบบดิจิตอล ให้สามารถนำไปผลิตได้เร็วขึ้น ซึ่งโครงการดังกล่าวมีชื่อว่า "Very High Speed Integrated Circuits" หรือ VHSIC โดยในระยะแรกนั้นโครงการนี้ถือเป็นความลับทางด้านความมั่งคงของประเทศ และอยู่ภายใต้ความควบคุมดูแลของ United States International Traffic and Arms Regulations (ITAR)

สำหรับมาตรฐานของภาษาที่ใช้บรรยายพฤติกรรมวงจรหรือฮาร์ดแวร์ของระบบสำหรับโครงการ VHSIC ที่ DoD ได้ให้ไว้สามารถสรุปได้ดังนี้

- ต้องเป็นภาษาที่นำไปเขียนรูปแบบระบบดิจิตอล และมีคุณสมบัติที่สามารถเข้าใจได้ทั้งมนุษย์และเครื่องคอมพิวเตอร์โดยไม่ต้องมีการแปลหรือเปลี่ยนแปลงอีก
- สามารถนำไปใช้เป็นเอกสารประกอบโครงการได้
- ต้องเป็นภาษาที่เขียนขึ้นสำหรับใช้จำลองการทำงานของวงจร

ฉะนั้นภาษาดังกล่าวนี้จึงจัดเป็นภาษาโปรแกรมระดับสูง เช่นเดียวกับภาษาปาสคาล หรือภาษาซี ซึ่งในทางวิศวกรรมภาษาที่ใช้ในการออกแบบฮาร์ดแวร์นี้เรียกว่า "Hardware Description Language" หรือ HDL

ในตอนเริ่มแรกนั้น DoD ได้มอบหมายให้บริษัทไอบีเอ็ม เท็กซัสอินสตูเมนท์ และอินเตอร์เมทริกซ์ เป็นผู้ศึกษาและพัฒนาโครงการ ซึ่งการดำเนินงานเป็นไปอย่างต่อเนื่อง จนกระทั่งในปี ค.ศ.1985 ทาง ITAR ได้ยกเลิกข้อจำกัดในการถ่ายทอดเทคโนโลยีทางทหารออกจากโครงการนี้ ดังนั้นภาษาวีเอชดีแอลจึงเริ่มเป็นที่รู้จักกันโดยทั่วไป และประมาณปี ค.ศ. 1987 IEEE ได้ทำการกำหนดมาตรฐานของภาษานี้เป็น IEEE 1076-1987 และมีชื่อเรียกว่าวีเอชดีแอลซึ่งมาตรฐานนี้ได้รับการปรับปรุงจนเป็นมาตรฐาน IEEE 1076-1993 หรือ VHDL 1993

เนื่องจากในขณะนั้น DoD เป็นลูกค้ารายใหญ่ของอุตสาหกรรมอิเล็กทรอนิกส์และคอมพิวเตอร์ ดังนั้นจึงมีผู้รับโครงการต่างๆ จาก DoD ไปดำเนินการวิจัยและพัฒนาเป็นจำนวนมาก และเพื่อให้ทุกโครงการอยู่ในมาตรฐานเดียวกันหมด ดังนั้นทาง DoD จึงได้กำหนดว่า ทุกๆ โครงการต้องเขียนอยู่ในรูปของภาษาวีเอชดีแอลเท่านั้น ซึ่งทำให้ DoD สามารถนำโครงการเหล่านี้ไปจำลองกับเครื่องคอมพิวเตอร์ได้หลายๆระบบ

3.3 ข้อกำหนดของภาษาวีเอชดีแอล

DoD ได้ตั้งข้อกำหนดสำหรับภาษาวีเอชดีแอลในเดือนมกราคมปี ค.ศ.1983 ไว้ดังนี้

3.3.1 ลักษณะทั่วไป

DoD ได้กำหนดให้วีเอชดีแอลเป็นภาษาสำหรับการออกแบบและบรรยายของฮาร์ดแวร์ ซึ่งหมายถึงความสามารถในการอธิบายและออกแบบในระดับสูง การจำลอง (Simulation) การสังเคราะห์ (Synthesis) และการทดสอบ (Testing) นอกจากนั้นวีเอชดีแอลยังถูกกำหนดไว้สำหรับการบรรยายฮาร์ดแวร์ตั้งแต่ระดับบนซึ่งก็คือระบบจนถึงระดับเกทอีกด้วย

เนื่องจากในการทำงานของระบบดิจิตอลนั้น ทุกๆ องค์ประกอบภายในระบบไม่ว่าเล็กหรือใหญ่จะทำงานไปพร้อมๆ กัน ซึ่งในเรื่องของความพร้อมเพรียงในการทำงานนี้ก็ถือเป็นข้อกำหนดที่สำคัญอย่างหนึ่งของวีเอชดีแอลด้วยเช่นกัน (สำหรับในภาษาที่ใช้ในการบรรยายฮาร์ดแวร์นั้น ความพร้อมเพรียงจะหมายถึงทุกๆ คำสั่ง องค์ประกอบ เกทหรือวงจรลอจิกจะถูกนำมาปฏิบัติทั้งหมด ดังนั้นในที่สุดแล้วก็จะดูเหมือนว่าได้มีการปฏิบัติไปพร้อมๆ กัน)

3.3.2 สนับสนุนการออกแบบแบบลำดับขั้น

การออกแบบแบบลำดับขั้นเป็นลักษณะที่สำคัญอย่างหนึ่งสำหรับการออกแบบระบบที่มีหลายๆ ระดับ โดยในการออกแบบจะประกอบด้วยส่วนการบรรยายการเชื่อมต่อ และส่วนการบรรยายหน้าที่การทำงาน ซึ่งหน้าที่การทำงานของระบบสามารถกำหนดได้ด้วยตัวเอง หรืออาจถูกกำหนดโดยโครงสร้างที่ประกอบด้วยองค์ประกอบย่อยๆ ลงไปได้เช่นกัน แต่ที่ระดับล่างสุด องค์ประกอบต้องถูกบรรยายหน้าที่การทำงานด้วยตัวมันเอง และไม่สามารถกำหนดการทำงานโดยลักษณะแบบโครงสร้างได้

3.3.3 ไลบรารี

วีเอชดีแอลได้สนับสนุนการมีไลบรารีเพื่อระบบการจัดการที่ดี ผู้ออกแบบสามารถกำหนดลักษณะและการทำงานของอุปกรณ์พื้นฐานไว้ในระบบไลบรารี หรือจะใช้ไลบรารีที่ระบบได้จัดเตรียมไว้แล้วก็ได้ โมเดลและการบรรยายที่ถูกต้องควรจัดเก็บไว้ในไลบรารีหลังจากที่ได้ผ่านการคอมไพล์เรียบร้อยแล้วเพื่อให้ผู้ออกแบบคนอื่นๆ สามารถนำไปใช้ได้ด้วย

3.3.4 ลำดับคำสั่ง

แม้ว่าการปฏิบัติคำสั่งหรือกระบวนการโดยพร้อมเพรียงกันจะเป็นคุณสมบัติที่สำคัญของวีเอชดีแอลก็ตาม ตัวภาษาเองก็ยังมีการจัดเตรียมลักษณะการควบคุมแบบลำดับคำสั่งไว้ให้ด้วย เมื่อผู้ออกแบบได้กำหนดหน้าที่และองค์ประกอบที่ทำงานพร้อมกันของระบบไว้เรียบร้อยแล้ว ผู้ออกแบบยังสามารถบรรยายหน้าที่การทำงานซึ่งเป็นรายละเอียดภายในของแต่ละองค์ประกอบได้ในลักษณะเดียวกับการเขียนโปรแกรมที่ประกอบด้วยโครงสร้างแบบ case, if - then - else และ loop ทั่วๆ ไปได้

การบรรยายแบบลำดับคำสั่งทำให้การออกแบบหน้าที่การทำงานของอุปกรณ์กระทำได้สะดวกและง่ายขึ้น อย่างไรก็ตามโครงสร้างทั้งหมดของวีเอชดีแอลก็ยังคงเป็นการทำงานแบบพร้อมเพรียงกันเช่นเดิม

3.3.5 การกำหนดคุณสมบัติ

นอกจากการกำหนดอินพุทและเอาท์พุทแล้ว เงื่อนไขอื่นๆ ก็มีผลต่อการปฏิบัติหน้าที่ของอุปกรณ์ฮาร์ดแวร์ด้วยเช่นกัน โดยสิ่งนี้รวมถึงสภาพแวดล้อมและลักษณะทางกายภาพของอุปกรณ์นั้นๆ ด้วย ซึ่งภาษาสำหรับการออกแบบที่ดีควรให้ผู้ออกแบบกำหนดคุณสมบัติของอุปกรณ์ที่ใช้ได้ด้วย เช่น สามารถกำหนดขนาด ลักษณะทางกายภาพ เวลา โหลด และเงื่อนไขทางสภาพแวดล้อม

อื่นๆ ซึ่งความสามารถในการกำหนดคุณสมบัตินี้ก็เป็นส่วนหนึ่งที่มีอยู่ในภาษาวีเอชดีแอลด้วยเช่นกัน

3.3.6 ชนิดของข้อมูล

วีเอชดีแอลสามารถกำหนดชนิดของข้อมูลไม่เพียงแต่ชนิด BIT และ BOOLEAN เท่านั้น แต่ยังสามารถกำหนดชนิดของข้อมูลเป็นจำนวนเต็ม จำนวนจริง จุดทศนิยม และชนิดลำดับการนับ (Enumerate Type) หรือแม้แต่ชนิดของข้อมูลที่ผู้ออกแบบกำหนดขึ้นมาเองก็ได้

3.3.7 โปรแกรมย่อย

ความสามารถในการใช้ฟังก์ชันและโพรซีเจอร์ (Procedure) ก็เป็นข้อกำหนดอีกอย่างหนึ่งในวีเอชดีแอลซึ่งผู้ออกแบบสามารถนำโปรแกรมย่อยมาใช้ในการเปลี่ยนแปลงชนิดของข้อมูล การกำหนดหน่วยของลอจิก การกำหนดตัวกระทำต่างๆ หรือหน้าที่อื่นๆ ตามที่ต้องการได้เช่นเดียวกับการเขียนโปรแกรมทั่วไป

3.3.8 การควบคุมเวลา

วีเอชดีแอลอนุญาตให้ผู้ออกแบบสามารถกำหนดเวลาในการส่งผ่านข้อมูลหรือสัญญาณได้ตามต้องการ การตรวจสอบ การออกแบบเกทหรือการหน่วงเวลาก็สามารถกระทำได้โดยการกำหนดช่วงเวลาที่แน่นอนหรือกำหนดให้มีการรอคอยเหตุการณ์ (Event) นอกจากนี้ก็ยังสามารถกำหนดรูปแบบของสัญญาณนาฬิกาได้อีกด้วย

3.3.9 การกำหนดแบบโครงสร้าง

การกำหนดโครงสร้างขององค์ประกอบต่างๆ สามารถกระทำได้ในทุกระดับของการออกแบบ โดยการกำหนดโครงสร้างขององค์ประกอบร่วมที่เกิดจากองค์ประกอบย่อยซึ่งแตกต่างกันหรือเหมือนกันก็เป็นข้อกำหนดอย่างหนึ่งของวีเอชดีแอลเช่นกัน

3.4 องค์ประกอบพื้นฐานของวีเอชดีแอล [6]

รูปแบบพื้นฐานที่ใช้ในการบรรยายถึงองค์ประกอบของวีเอสดีแอลจะประกอบไปด้วยส่วนกำหนดการเชื่อมต่อ (Interface) และส่วนกำหนดลักษณะเชิงสถาปัตยกรรม (Architecture) ดังแสดงในรูปที่ 3.3 โดยในการบรรยายการเชื่อมต่อจะขึ้นต้นด้วยคำว่า ENTITY แล้วตามด้วยชื่อขององค์ประกอบจากนั้นตามด้วยคำว่า IS และถัดมาจะเป็นการบรรยายถึงพอร์ตการติดต่อ อินพุท –

เอาท์พุท ขององค์ประกอบ ส่วนลักษณะภายนอกอื่น ๆ เช่น เวลา อุณหภูมิก็สามารถรวมเข้าไปในส่วนนี้ได้เช่นกัน

```
ENTITY component_name IS
    Input and output ports
    Physical and other parameters
END [component_name] ;

ARCHITECTURE identifier OF component_name IS
    [declartion]
BEGIN
    specification of  the functionality of the component
    in terms of its input lines and as influenced
    by physical and other parameters
END [identifier] ;
```

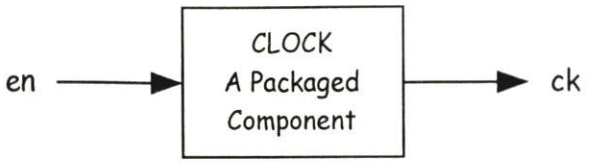
รูปที่ 3.3 การกำหนดการเชื่อมต่อและสถาปัตยกรรม

ในส่วนของการกำหนดลักษณะเชิงสถาปัตยกรรมจะขึ้นต้นด้วยคำว่า ARCHITECTURE ซึ่งเป็นส่วนที่ใช้บรรยายหน้าที่การทำงานขององค์ประกอบ โดยหน้าที่การทำงานนี้จะขึ้นอยู่กับสัญญาณอินพุท – เอาท์พุทและพารามิเตอร์อื่นๆ ที่ได้กำหนดไว้ในส่วนของการเชื่อมต่อดังรูปที่ 3.3 และสำหรับการบรรยายหน้าที่ขององค์ประกอบจะเริ่มต้นหลังจากคำว่า BEGIN เป็นต้นไป

3.4.1 **การกำหนดการเชื่อมต่อ**

การกำหนดการเชื่อมต่อเป็นระดับบนสุดของการออกแบบ โดยในระดับนี้ต้องกำหนดพอร์ตสำหรับการติดต่อกับองค์ประกอบภายนอกอื่นๆ ดังตัวอย่างในรูปที่ 3.4 ซึ่งเป็นบล็อคไดอะแกรม และการบรรยายการเชื่อมต่อขององค์ประกอบสำหรับตัวจ่ายสัญญาณนาฬิกา

ในบรรทัดแรกของการบรรยายการเชื่อมต่อเป็นการกำหนดชื่อขององค์ประกอบซึ่งกำหนดเป็น clock_ component ตามด้วยคำว่า PORT และชื่อของพอร์ตอยู่ภายในวงเล็บ ส่วน IN และ OUT เป็นการกำหนดโหนดของสัญญาณให้เป็นอินพุทหรือเอาท์พุท และ BIT เป็นการแสดงชนิดของข้อมูล




```
ENTITY clock_component IS
    PORT (en : IN BIT;ck : OUT BIT )
END clock_name;
```

รูปที่ 3.4 บล็อคไดอะแกรมและการบรรยายการเชื่อมต่อของ clock_ component

3.4.2 การกำหนดรูปแบบการบรรยาย

หน้าที่การทำงานขององค์ประกอบจะถูกบรรยายภายในส่วนนี้ ซึ่งในการบรรยายสามารถกำหนดค่าของสัญญาณเอาท์พุทในเทอมของอินพุทหรือในรูปขององค์ประกอบอื่นๆ หรือทั้งสองอย่างรวมกันก็ได้ ดังตัวอย่างการบรรยายของ clock_component ในรูปที่ 3.5 ซึ่งเป็นการบรรยายในเชิงพฤติกรรมโดยมี en เป็นอินพุทและ ck เป็นเอาท์พุท

PROCESS เป็นคำที่ใช้ในการเริ่มต้นสำหรับการบรรยายในเชิงพฤติกรรม และภายในโปรเซสกำหนดให้ periodic เป็นตัวแปรที่มีคำเริ่มต้นเป็น "0" ถ้าสัญญาณ en มีค่าเป็น "1"' จะทำให้ตัวแปร periodic ถูกคอมพลีเมนท์ (complement) และส่งค่าให้กับ ck ซึ่งเป็นสัญญาณเอาท์พุท และสำหรับคำสั่ง WAIT จะเป็นการกำหนดให้สัญญาณมีคาบเวลาเท่ากับ 1 ไมโครวินาที

```
ARCHITECTURE behavioral OF clock_component IS

BEGIN
    PROCESS
        VARIABLE periodic : BIT := '0';
    BEGIN
        IF en='1' THEN
         periodic :=  Not periodic;
        END IF;
         ck <= periodic;
         WAIT FOR 1 US;
   END PROCESS;
END behavioral;
```

รูปที่ 3.5 การบรรยายเชิงพฤติกรรมของ clock _ component

3.4.3 หน่วยการออกแบบแพ็กเก็ต

ข้อมูลต่างๆ ตลอดจนโปรแกรมย่อย ที่เป็นประโยชน์ต่อการเขียนรูปแบบการบรรยายระบบดิจิตอล สามารถเก็บไว้ในส่วนของแพ็กเก็ต ซึ่งหน่วยการออกแบบต่างๆ เช่น หน่วยการออกแบบ Entity หน่วยการออกแบบสถาปัตยกรรม หรือหน่วยการออกแบบแพ็กเกจอื่นๆ สามารถเรียกข้อมูลเหล่านี้ไปใช้ได้ นอกจากนั้นสิ่งที่นิยมทำกันมากคือการนำรูปแบบมาตรฐานต่างๆ เช่น อุปกรณ์มาตรฐาน (เช่น ไอซีตระกูล 74XX เป็นต้น) มาเก็บไว้ในรูปของแพ็กเกจ ที่ทุกคนสามารถเข้าถึงได้

ตามปกติแล้วแพ็กเกจจะแบ่งออกเป็น 2 ส่วนคือ การประกาศแพ็กเกจ (Package declaration) และส่วนของบอดี้แพ็กเกจ (Package body) เนื่องจาก แพ็กเกจถูกสร้างขึ้นเป็นส่วนแยกต่างหากออกจากรูปแบบที่กำลังเขียนอยู่ ฉะนั้นการที่นำแพ็กเกจไปใช้นั้นจะต้องมีการเชื่อมโยงหรืออ้างอิงเสียก่อน ซึ่งในภาษาวีเอชดีแอลสามารถกระทำได้ด้วยชุดคำสั่ง USE

3.4.3.1 PACKAGE DECLARATION

ส่วนที่มีความสำคัญที่สุดของแพ็กเกจ (ถ้ามองในแง่ของการนำไปใช้จากภายนอก) ได้แก่ส่วนการประกาศแพ็กเกจ เนื่องจากเป็นส่วนที่ใช้กำหนดชื่อของสิ่งที่ประกาศอยู่ภายในแพ็กเกจ สำหรับนำไปใช้ภายนอกตัวของแพ็กเกจเอง ถ้ามีการประกาศสิ่งใดๆ ในส่วนของส่วนบอดี้แพ็กเกจ แต่ไม่ถูกประกาศในส่วนการประกาศแพ็กเกจ จะทำให้ค่าและพฤติกรรมไม่สามารถนำไปใช้งานในส่วนนอกได้ ซึ่งเปรียบเทียบได้กับสิ่งที่ประกาศไว้ในส่วนของการประกาศ Entity คือ จุดเชื่อมต่อ หรือ พอร์ต ที่มีหน้าที่ติดต่อกับโลกภายนอก ฉะนั้นโดยทั่วไปแล้วแพ็กเกจสามารถสร้างขึ้นได้โดยไม่จำเป็นต้องมีส่วนบอดี้ และยังสามารถนำไปใช้งานจากรูปแบบภายนอกได้เช่น ใช้สำหรับประกาศ ชนิด (Type) หรือสัญญาณ เช่นเดียวกับ ส่วนบอดี้แพ็กเกจที่ไม่จำเป็นต้องมี ส่วนของการประกาศแพ็กเกจ แต่แพ็กเกจนั้นจะไม่สามารถนำไปใช้จากรูปแบบอื่นได้

```
PACKAGE package_name IS
    Package_declarative_part
END package_name;
```

รูปที่ 3.6 โครงสร้างทั่วไปของส่วนการประกาศแพ็กเกจ

3.4.3.2 PACKAGE BODY

โครงสร้างซึ่งประกอบด้วยลำดับคำสั่งที่ใช้บรรยายฟังก์ชันการทำงานของโปรแกรมย่อยทั้งหลาย ซึ่งชื่อของโปรแกรมย่อยนั้นๆ ได้ถูกประกาศไปแล้วในส่วนของการประกาศแพ็กเกจ จะถูกเก็บไว้ในส่วนของบอดี้แพ็กเกจ ทั้งนี้รวมถึงการกำหนดค่าคงที่ต่างๆ อันได้แก่ค่าคง

ที่ที่ถูกประกาศชื่อไว้ก่อนในส่วนของการประกาศแพ็กเกจ และถูกกำหนดค่าในส่วนของบอดี้แพ็กเกจ ฉะนั้นในส่วนของบอดี้แพ็กเกจจึงไม่จำเป็นต้องมี ถ้าในส่วนของการประกาศแพ็กเกจไม่มีการประกาศชื่อที่เป็นโปรแกรมย่อย หรือค่าคงที่ การเขียนบอดี้แพ็กเกจนั้นจะเป็นไปตามกฎเกณฑ์ดังแสดงในรูปที่ 3.7

```
PACKAGE BODY package_name IS
    declarative part
END package_name;
```

รูปที่ 3.7 โครงสร้างของบอดี้แพ็กเกจ

3.4.4 หน่วยการออกแบบ Configuration

ดังที่ทราบกันแล้วว่าระบบดิจิตอลรูปแบบหนึ่งไม่ว่าจะเป็นอะไรก็ตาม จะสามารถมีหน่วยการออกแบบ Entity ได้เพียงหนึ่งเดียวเท่านั้น ซึ่งในหน่วยการออกแบบ Entity หนึ่งหน่วยนี้อาจจะมีสถาปัตยกรรมที่เป็นหน่วยรองได้หลายหน่วย ดังนั้นจะต้องมีหน่วยการออกแบบ Configuration มาเพื่อกำหนดการใช้ Configuration ของการประกอบ Entity กับหน่วยการออกแบบสถาปัตยกรรมหน่วยใดๆ เข้าด้วยกัน

```
COMFIGURTRION identifier OF entiry_name IS
    Configuretion_declarative_part
END ;
```

รูปที่ 3.8 โครงสร้างโดยทั่วไปของหน่วยการออกแบบโครงแบบ

3.4.5 โปรแกรมย่อย

การใช้ฟังก์ชันและโพรซีเจอร์ในภาษาวีเอชดีแอลเปรียบได้กับการใช้โปรแกรมย่อยในการเขียนโปรแกรมภาษาชั้นสูงทั่วๆ ไป ค่าที่ถูกส่งกลับหรือถูกเปลี่ยนแปลงโดยโปรแกรมย่อยอาจจะมีหรือไม่มีผลต่อฮาร์ดแวร์โดยตรงก็ได้ เช่นถ้าใช้ฟังก์ชันแทนการกระทำในสมการบูลีนก็จะมีผลต่อวงจรลอจิกจริงๆ ในขณะที่ถ้าใช้โปรแกรมย่อยในการเปลี่ยนชนิดของข้อมูลหรือในการคำนวณค่าการหน่วงเวลาแล้วก็จะไม่มีผลต่อโครงสร้างของฮาร์ดแวร์ รูปที่ 3.9 แสดงการใช้โพรซีเจอร์เพื่อเปลี่ยนข้อมูลชนิด 8 บิตเป็นค่าจำนวนเต็ม และรูปที่ 3.10 แสดงการใช้ฟังก์ชันโดยกำหนดให้ X เป็นตัวแปรชนิดบิตแทนการกระทำในสมการบูลีน

```
TYPE byte IS ARRAY (7 DOWNTO 0) OF BIT;
 ...
PROCEDURE byte_to_integer (ib : IN byte; oi : OUT INTEGER)IS
        VARIABLE result: INTEGER := 0;
BEGIN
        FOR i IN 0 TO 7 LOOP
                IF ib(i) = '1' THEN
                result := result + 2**i;
                END IF;
        END LOOP;
        oi := result;
END byte_to_integer
```

รูปที่ 3.9 การใช้โพรซีเจอร์

```
FUNCTION f (a, b, c:BIT) RETURN BIT IS
        VARIABLE x:BIT;
BEGIN
        x := ((NOT a) AND (NOT b) AND c) ;
        RETURN x;
END f;
```

รูปที่ 3.10 การใช้ฟังก์ชัน

3.4.6 โอเปอร์เรเตอร์

การบรรยายเชิงพฤติกรรมในภาษาวีเอชดีแอลมีตัวดำเนินการหรือโอเปอร์เรเตอร์ทางลอจิกและคณิตศาสตร์เช่นเดียวกับภาษาซอฟท์แวร์ทั่วไปดังรูปที่ 5.11

PREDEFIND OPERATORS
LOGICAL OPERATORS : NOT AND OR NAND NOR XOR OPERAND TYPE : BIT BOOLEAN RESULT TYPE : BIT BOOLEAN RELATIONAL OPERATORS : = /= < <= > >= OPERAND TYPE : any type RESULT TYPE : Boolen ARITHMETIC OPERATORS : + - * / ** MOD REM ABS OPERAND TYPE : INTEGER REAL Physical RESULT TYPE : INTEGER REAL Physical CONCANTENATION OPERATOR : & OPERAND TYPE : ARRAY of any type RESULT TYPE : array of any type RESULT TYPE : array of any type

รูปที่ 3.11 ตัวดำเนินการในวีเอสดีแอล

3.4.7 เวลาและความพร้อมเพรียง

ในวงจรอิเล็กทรอนิกส์อุปกรณ์ทุกๆ ตัวจะอยู่ในสภาพเตรียมพร้อมเสมอ (Always Active) และจะมีเรื่องของเวลาเข้ามาเกี่ยวข้องในทุกๆ เหตุการณ์ที่เกิดขึ้นเสมอ วีเอชดีแอลเป็นภาษาที่ได้รับการออกแบบมาเพื่อให้สามารถบรรยายรูปแบบและการพ้องกันของเวลาสำหรับการทำงานของอุปกรณ์ได้อย่างถูกต้อง การบรรยายการทำงานที่อยู่ภายในส่วนของการบรรยายสถาปัตยกรรม จะมีการทำงานที่พร้อมเพรียงกันเสมอ หรือแม้แต่โปรเซสซึ่งมีการทำงานภายในเป็นแบบลำดับคำสั่งก็ตาม ซึ่งหากมีหลายๆ โปรเซสอยู่ภายในโครงสร้างเดียวกัน ทุกๆ โปรเซสก็จะทำงานไปพร้อมๆ กันด้วย

3.4.8 สัญญาณและตัวแปร

สัญญาณมีลักษณะเป็นเสมือนตัวกลางฮาร์ดแวร์ที่ใช้ในการส่งผ่านข้อมูลและมีเรื่องของเวลาเข้ามาเกี่ยวข้องด้วย การกำหนดค่าให้กับสัญญาณจะใช้สัญลักษณ์ <= ในการส่งค่าและสามารถใช้คำสั่ง AFTER เพื่อกำหนดช่วงเวลาในการส่งผ่านค่าของสัญญาณเช่น w <= a AFTER 12 NS หมายถึงการกำหนดค่าสัญญาณ a ให้กับ w หลังจากเวลาผ่านไป 12 นาโนวินาที

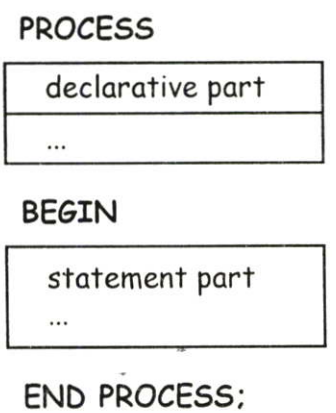
ในทางตรงข้าม ตัวแปรมีลักษณะเป็นเสมือนตัวกลางที่ใช้ในการส่งผ่านข้อมูลและไม่มีเรื่องของเวลาเข้ามาเกี่ยวข้องด้วย ซึ่งตัวแปรจะถูกใช้ในส่วนที่มีการทำงานเป็นแบบลำดับคำสั่ง เช่นใน ฟังก์ชัน โพรซีเจอร์ และโปรเซส สำหรับการกำหนดค่าให้กับตัวแปรจะใช้สัญลักษณ์ :=

3.5 การบรรยายเชิงพฤติกรรม

การบรรยายลักษณะการทำงานของอุปกรณ์ฮาร์ดแวร์ในเชิงพฤติกรรม เป็นการบรรยายลักษณะการเปลี่ยนแปลงของข้อมูลในรูปแบบของอัลกอริธึมสำหรับการคำนวณผลลัพธ์ที่เกิดขึ้น ซึ่งสืบเนื่องมาจากการเปลี่ยนแปลงสภาวะของข้อมูลที่เข้ามาโดยไม่คำนึงถึงลักษณะโครงสร้างหรือความสัมพันธ์ของอุปกรณ์ที่อยู่ภายในว่าจะเป็นอย่างไร ในหัวข้อนี้จะแสดงถึงการบรรยายเชิงพฤติกรรม แทนการใช้โมดูลฮาร์ดแวร์รวมถึงข้อกำหนดต่างๆ ที่ควรรู้

3.6 โปรเซส

โปรเซสเป็นรูปแบบพื้นฐานอย่างหนึ่งที่ใช้ในการกำหนดให้กับสัญญาณ โปรเซสจะอยู่ในสถานะที่เตรียมพร้อมอยู่เสมอและจะปฏิบัติคำสั่งพร้อมๆ กันกับโปรเซสอื่นๆ ที่อยู่ในสถาปัตยกรรมบรรยายเดียวกัน โดยโปรเซสจะปฏิบัติงานตามคำสั่งทันทีที่มีเหตุการณ์เกิดขึ้นกับสัญญาณที่อยู่ทางด้านขวามือของสัญลักษณ์กำหนดค่าให้กับสัญญาณ (<=)



รูปที่ 3.12 รูปแบบของการบรรยายแบบโปรเซส

การบรรยายโปรเซสจะเริ่มต้นด้วยคำสั่ง PROCESS และจบด้วยคำสั่ง END PROCESS ในรูปที่ 3.12 เป็นการแสดงส่วนประกอบของการบรรยายแบบโปรเซส ซึ่งประกอบด้วยส่วนของการประกาศตัวแปรที่ต้องใช้และส่วนของการปฏิบัติคำสั่งเพื่อให้ได้ผลลัพธ์ที่ต้องการ

3.7 การกำหนดตัวดำเนินการภายในโปรเซส

ตัวดำเนินการภายในโปรเซสมี 3 ชนิดคือ ตัวแปร (Variable) ไฟล์ (File) และตัวคงที่ (Constant) ซึ่งตัวดำเนินการทั้งสามชนิดนี้หากมีการประกาศไว้ในโปรเซสใดก็จะใช้ได้เฉพาะภายในโปรเซสนั้นเท่านั้น สำหรับการติดต่อกับภายนอกหรือระหว่างโปรเซสสามารถทำได้โดยใช้สัญญาณ (Signal) หรือตัวคงที่ที่ได้ประกาศไว้ในส่วนของ ARCHITECTURE ในรูปที่ 3.13 แสดงตัวอย่างการประกาศตัวกระทำภายในโปรเซส ซึ่งจะอยู่ระหว่างคำสั่ง PROCESS และ BEGIN และค่าเริ่มต้นที่ถูกกำหนดให้กับตัวดำเนินการภายในโปรเซสจะถูกนำมาใช้ในตอนเริ่มต้นของการปฏิบัติเพียงครั้งเดียวเท่านั้น ต่างกับค่าเริ่มต้นที่อยู่ภายในโปรแกรมย่อยจะถูกนำมาใช้ทุกครั้งที่มีการเรียกใช้โปรแกรมย่อยนั้น ๆ

```
PROCESS
   FILE flush : TEXT IS IN "filename.dat";
   VARIABLE var : BIT;
   CONSTANT n : INTEGER := 0;
BEGIN
  ....
END PROCESS;
```

รูปที่ 3.13 ตัวอย่างการประกาศตัวดำเนินการภายในโปรเซส

3.8 การกำหนดการกระทำภายในโปรเซส

การกระทำใดๆ ภายในโปรเซสจะเป็นการปฏิบัติแบบลำดับ (Sequential) เสมอ ซึ่งภายในโปรเซสสามารถใช้ประโยคเงื่อนไขหรือการทำซ้ำได้เช่น IF-THEN-ELSE , CASE-WHEN , FOR LOOP และ WHILE-LOOP ดังตัวอย่างในรูปที่ 3.14 และ 3.15

```
ARCHITECTURE demo OF partial_process IS
 ....
BEGIN
  PROCESS
  ....
   BEGIN
          ....
          x <= '1' ;
          IF x = '1' THEN
            perform action_1
          ELSE
            perform action_2
          END IF;
          ...
  END PROCESS;
END demo;
```

รูปที่ 3.14 เงื่อนไขการกระทำในโปรเซส

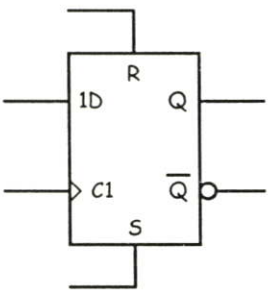
```
ARCHITECTURE demo OF partial_process IS
  ...
BEGIN
  PROCESS
    BEGIN
          ...
          x <= a AFTER 10 NS ;
          y <= b AFTER 6 NS ;
          ...
  END PROCESS;
END demo;
```

รูปที่ 3.15 การกระทำในโปรเซส

3.9 การกระตุ้นและยับยั้งการกระทำของโปรเซส

การกระทำภายในโปรเซสจะอยู่ในสภาพเตรียมพร้อม และมีการปฏิบัติงานอยู่ตลอดเวลาที่มีการเปลี่ยนแปลงของเหตุการณ์เกิดขึ้น อย่างไรก็ดีเราสามารถกระตุ้นหรือยับยั้งการกระทำภายในโปรเซสได้โดยการกำหนดรายการของสัญญาณที่ต้องการให้โปรเซสปฏิบัติงานเมื่อมีเหตุการณ์เกิดขึ้นกับสัญญาณที่กำหนดไว้เท่านั้น ส่วนเหตุการณ์ใดๆ ที่เกิดขึ้นกับสัญญาณที่ไม่ได้กำหนดไว้ในรายการก็จะไม่ส่งผลให้มีการกระทำภายในโปรเซส ซึ่งรายการของสัญญาณนี้เรียกว่า Sensitivity List และจะกำหนดไว้ภายในวงเล็บหลังคำสั่ง PROCESS

รูปที่ 3.16 (a) แสดงตัวอย่างโมเดล และรูปที่ 3.16 (b) เป็นตัวอย่างการบรรยายการเชื่อมต่อของ D-Flip Flop ส่วนรูปที่ 3.17 แสดงถึงการบรรยายเชิงพฤติกรรมของ D-Flip Flop โดยในรูปที่ 3.17 (a) เป็นการใช้ตัวกระทำภายนอกโปรเซส และรูปที่ 3.17 (b) เป็นการใช้ตัวกระทำภายในโปรเซส โดยมีรายการของสัญญาณ (rst, set, clk) เป็นตัวกระตุ้นการปฏิบัติงานภายในโปรเซส



(a)

```
ENTITY component_name IS
    Input and output ports
    Physical and other parameters
END component_name;
```

(b)

รูปที่ 3.16 (a) ตัวอย่างโมเดล D-Flip Flop

(b) การบรรยายการเชื่อมต่อของ D-Flip Flop

```
ARCHITECTURE behavioral OF d_sr_flipflop IS
  SIGNAL state : BIT := '0';
BEGIN
   dff : PROCESS (rst, set, clk )
          BEGIN
        IF set= '1'  THEN
            state <= '1' AFTER sq_delay;
          ELSIF rst = '1' THEN
            state <= '0' AFTER rq_delay;
          ELSIF clk = '1' AND clk ' EVENT  THEN
            state <= d AFTER cq_delay;
        END IF;
   END PROCESS dff;
  q <= state;
  qb <= NOT state;
END behavioral;
```

(a)

```
ARCHITECTURE average_dalay_ behavioral OF d_sr_flipflop IS
BEGIN
   dff : PROCESS (rst, set, clk )
            VARIABLE state : BIT := '0';
          BEGIN
        IF set= '1' THEN
            state <= '1';
          ELSIF rst = '1' THEN
            state <= '0' ;
          ELSIF clk = '1' AND clk ' EVENT THEN
            state <= d ;
        END IF;
            q <= state AFTER (sq_delay + rq_delay + cq_delay)/3;
           qb <= NOT state AFTER(sq_delay + rq_delay + cq_delay)/3;
   END PROCESS dff;
END behavioral;
```

(b)

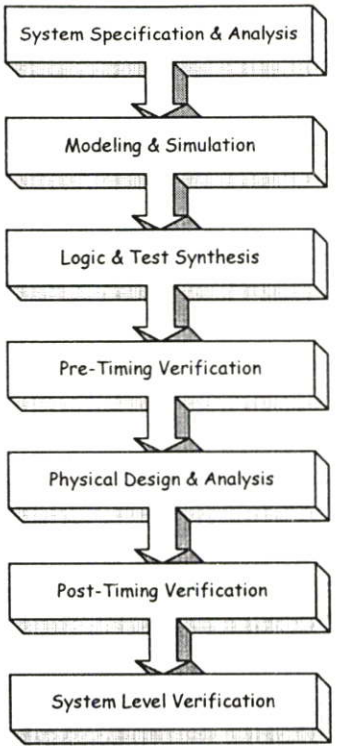
รูปที่ 3.17 การบรรยายเชิงพฤติกรรมของ D-FlipFlop

(a) การใช้ตัวกระทำภายนอกโปรเซส

(b) การใช้ตัวกระทำภายในโปรเซส

3.10 การออกแบบจากบนลงล่าง

ในการพัฒนาวงจรรวมดิจิตอลขนาดใหญ่ที่มีความซับซ้อน วิศวกรหรือผู้ออกแบบมักจะมองการออกแบบให้อยู่ในรูปของบล็อกไดอะแกรมก่อนที่ทำวิเคราะห์ให้ลึกถึงรายละเอียดต่อไป ซึ่งภาษาวีเอชดีแอลนั้นอนุญาตให้อธิบายและวิเคราะห์การทำงานของแต่ละบล็อก รวมถึงการปรับปรุงการทำงานจากผลที่วิเคราะห์เพื่อให้ได้การทำงานตามต้องการ นอกจากนี้ยังสามารถเพิ่มเติมในรายละเอียดในแต่ละขั้นตอนได้ ซึ่งหลักการนี้สอดคล้องกับหลักการออกแบบจากบนลงล่าง (Top - Down Design) นั่นเอง ถ้าทดลองเปรียบเทียบกับการออกแบบจากล่างขึ้นบน (Bottom - Up Design) จะเห็นได้ว่าการออกแบบจากล่างขึ้นบนจะใช้เวลาการออกแบบมากกว่า 90% เนื่องจากเป็นการวาดวงจรด้วยอุปกรณ์ต่างๆ (Schematic capture) ที่ประกอบกันเข้าเป็นวงจรที่ต้องการออกแบบ ก่อนแล้วจึงทำการจำลองการทำงาน และตรวจสอบความถูกต้อง ดังนั้นการใช้ภาษาวีเอชดีแอลกับหลักการออกแบบจากบนลงล่างจึงเป็นทางออกให้กับวิศวกรให้สามารถออกแบบและพัฒนาวงจรที่มีความซับซ้อนได้มากขึ้น ทั้งยังช่วยลดเวลาและค่าใช้จ่ายในการออกแบบด้วย



รูปที่ 3.18 ขั้นตอนการออกแบบจากบนลงล่าง

จากรูปที่ 3.18 แสดงถึงขั้นตอนของการออกแบบจากบนลงล่าง ทั้งนี้ในทางปฏิบัติอาจมีข้อแตกต่างไปจากนี้บ้างเล็กน้อย เนื่องจากขั้นตอนของการผลิต (Implementation) สามารถกระทำได้หลายเทคโนโลยี สำหรับรายละเอียดของขั้นตอนการออกแบบจากบนลงล่างในแต่ละขั้นตอนมีดังนี้

1.สร้างข้อกำหนดของความต้องการ และวิเคราะห์ระบบ เพื่อหาแนวความคิดและหลักการ (Idea and Concept) ในการแก้ปัญหา

2.เขียนรูปแบบของระบบที่ต้องการออกแบบโดยใช้ภาษาวีเอชดีแอลหรือภาษา HDL อื่น ๆ สำหรับบรรยายพฤติกรรมการทำงาน พร้อมทั้งจำลองการทำงาน เพื่อเปรียบเทียบและตรวจสอบความถูกต้องกับข้อกำหนด

3.หลังจากที่ได้หลักการขั้นต้นพร้อมแนวความคิดที่ผ่านการตรวจสอบแล้ว หลักการนี้จะถูกเพิ่มเติมในรายละเอียดลงมาเป็นลำดับขั้นที่สอง จนกระทั่งอยู่ในระดับที่จะนำไปผลิตวงจรจริง หรือสังเคราะห์ในขั้นตอนนี้เองเทคโนโลยีที่จะมารองรับวงจรออกแบบจะถูกกำหนดขึ้น และระบบช่วยการออกแบบจะสังเคราะห์วงจรที่ได้จากรูปแบบที่เขียนขึ้นให้อยู่ในรูปของวงจรที่ประกอบด้วยอุปกรณ์อิเล็กทรอนิกส์ หรือวงจรในระดับเกท และการเชื่อมต่อระหว่างกันของอุปกรณ์เหล่านั้นหรือไม่ก็อยู่ในรูปของ Netlist ที่สามารถนำไปผลิตในอุปกรณ์อื่นได้

4.หลังจากการสังเคราะห์วงจรให้อยู่ในระดับเกทหรือ Netlist แล้ว ข้อมูลนี้จะถูกใช้สำหรับจำลองการทำงานในเรื่องความถูกต้องของฟังก์ชัน พร้อมกับนำข้อมูลที่เกี่ยวข้องกับเวลาเข้ามาประกอบการพิจารณาด้วย ซึ่งตามปกติแล้วอุปกรณ์ทางอิเล็กทรอนิกส์ทุกชิ้นจะมีเวลาหน่วงของการแพร่กระจาย (Propagation Delay Time) เสมอ ถึงแม้ว่าจะเป็นเวลาที่น้อยมากในระดับนาโนวินาทีก็ตาม แต่ถ้าภายในวงจรหนึ่งประกอบด้วยเกทของฟังก์ชันต่างๆ จำนวน 10,000 เกท ขึ้นไป เวลาดังกล่าวนี้จะสะสมกันมากขึ้น จนอาจทำให้การทำงานของวงจรรวมทั้งหมดผิดพลาดไป หรือไม่สามารถทำงานในย่านความถี่สัญญาณนาฬิกาที่สูงได้

5.ผลิตเป็นวงจรจริง (Technology and device mapping) โดยนำข้อมูลที่ได้จากการสังเคราะห์มาผลิต ซึ่งอาจจะอยู่ในรูปของแผงวงจรไฟฟ้า ที่ประกอบด้วยอุปกรณ์หลายๆ ชิ้นหรืออยู่ในรูปของวงจรรวม ASIC

6.ทำการตรวจสอบการทำงานและตัวแปรทางด้านเวลาทั้งหมด เพื่อความถูกต้องของวงจรเป็นครั้งสุดท้ายก่อนนำไปรวมเข้ากับอุปกรณ์อื่นๆ ให้เป็นระบบดิจิตอล เนื่องจากในขั้นตอนนี้วงจรที่ออกแบบ จะประกอบด้วยจุดต่อทางอินพุทและเอาท์พุท ซึ่งเป็นจุดต่อสำหรับการรับและส่งสัญญาณกับภายนอก

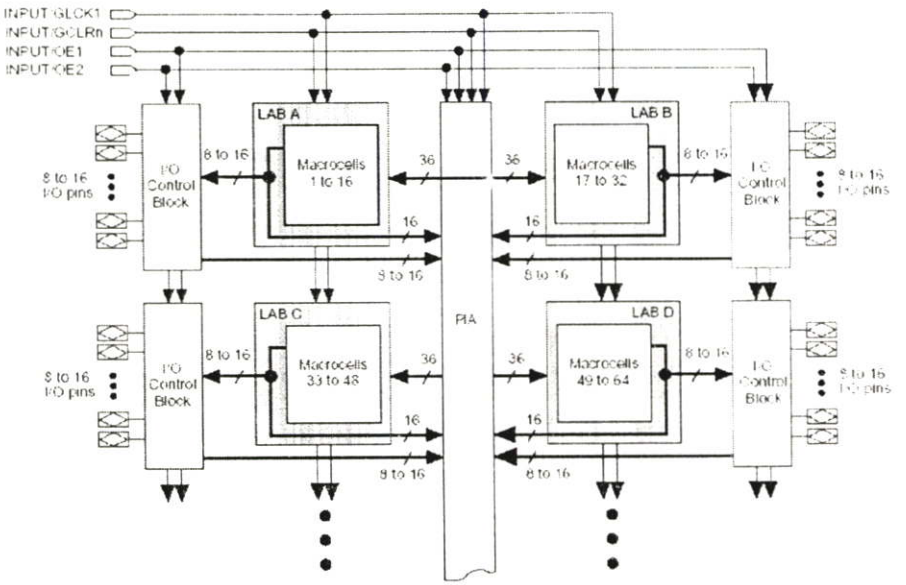
7.นำวงจรที่ออกแบบไว้ประกอบเข้ากับอุปกรณ์อื่นๆ ให้เป็นระบบที่สมบูรณ์ แล้วทำการทดสอบการทำงานทั้งระบบร่วมกับอุปกรณ์อื่นๆ อีกครั้งเพื่อควบคุมคุณภาพของผลิตภัณฑ์

3.11 โครงสร้างภายในของเอฟพีจีเอ

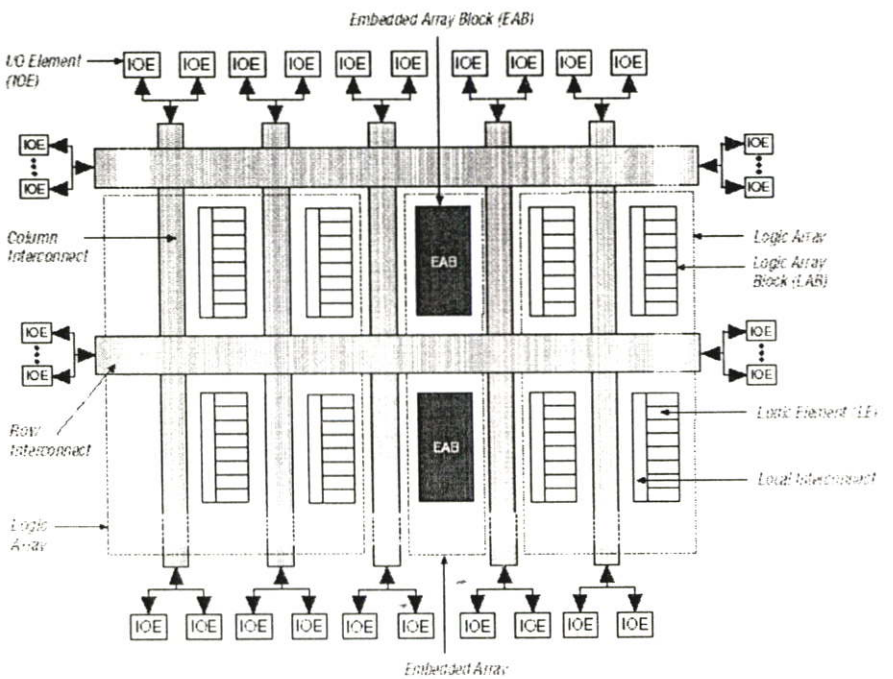
ลักษณะโครงสร้างภายในของเอฟพีจีเอจะเป็นอะเรย์ของบล็อคลอจิกที่สามารถทำการโปรแกรมได้ดังรูปที่ 3.19 และ 3.20

จากรูปที่ 3.19 เป็นโครงสร้างภายในเอฟพีจีเอตระกูล MAX7000S มีลักษณะโครงสร้างเป็นแบบ EEPROM BASE FPGA สามารถเก็บข้อมูลที่โปรแกรมลงไปได้ โดยไม่จำเป็นต้องมีไฟเลี้ยง (NON-Volatile Configuration) ใช้ AND-OR Plane ในการทำลอจิกฟังก์ชัน และมีการจัดสถาปัตยกรรมในรูปแบบอะเรย์ (Array Structure) ในการโปรแกรมสามารถทำซ้ำได้ ประมาณ 10,000 ครั้ง ส่วนรูปที่ 3.20 เป็นโครงสร้างภายในเอฟพีจีเอตระกูล FLEX10K จะมีโครงสร้างภายในเป็นแบบ SRAM-BASE FPGA ใช้เทคโนโลยีในการโปรแกรมเหมือนกับหน่วยความจำแบบ SRAM (Static RAM) ทำให้การโปรแกรมสามารถทำซ้ำได้ โดยไม่จำกัดจำนวณครั้งและใช้เวลาในการโปรแกรมชิฟเอฟพีจีเอน้อยมาก (ระดับ nsec) การโปรแกรมทำได้ง่ายเทียบได้กับการเขียน SRAM ทั่วไป นอกจากนี้ยังมีความจุของเกทประมาณ 10,000 เกท และภายใน FLEX10K ยังมีส่วนของหน่วยความจำภายใน (RAM bits) ให้สามารถใช้งานได้อีก 6,144 บิท เหมาะสำหรับการ

ออกแบบวงจรที่มีความสลับซับซ้อน ส่วนข้อเสียคือไม่สามารถเก็บโปรแกรมในภาวะที่ไม่มีไฟเลี้ยงได้



รูปที่ 3.19 โครงสร้างภายในของเอฟพีจีเอตระกูล MAX7000S [9]



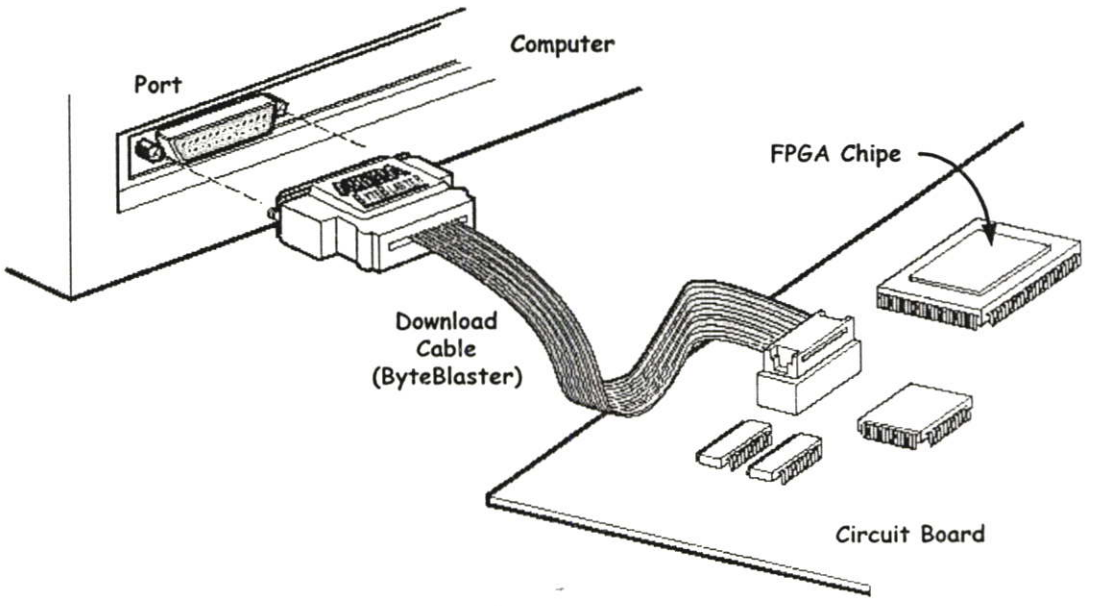
รูปที่ 3.20 โครงสร้างภายในของเอฟพีจีเอตระกูล FLEX10K [10]

3.12 ปัจจัยที่ทำให้การออกแบบเอฟพีจีเอทำได้ง่ายและสะดวกรวดเร็ว

1.ผู้ออกแบบไม่จำเป็นต้องทราบถึงโครงสร้างภายในของตัวชิพ เพียงแต่มีความรู้เกี่ยวกับขั้นตอนการออกแบบลอจิกก็เพียงพอแล้ว ต่างกับการใช้ไมโครโปรเซสเซอร์ซึ่งจำเป็นต้องศึกษาโครงสร้างภายในรวมถึงภาษา Assembly ของไมโครโปรเซสเซอร์ตัวนั้นด้วย

2.มีการออกแบบโดยใช้ภาษาในการอธิบายการทำงานของวงจร หรือ HDL (Hardware Description Language) เป็นเครื่องมือในการออกแบบ ซึ่งเป็นวิธีการที่มีความยืดหยุ่นสูง ทำได้รวดเร็ว และไม่จำเป็นต้องทราบถึงลักษณะของวงจรที่ต้องการว่าจะเชื่อมต่อกันอย่างไร เพียงแต่กำหนดลักษณะการทำงานให้มันจากนั้นตัวซอฟต์แวร์จะทำ Synthesis and Optimize ให้ทั้งหมด นอกจากนี้ภาษาที่ใช้ยังเป็นมาตราฐานเดียวกัน สามารถใช้ได้กับชิพทุกตัวและทุกบริษัท

3.การโปรแกรมสามารถทำได้เองและใช้เวลาไม่นาน เพียงแค่ส่งข้อมูลผ่านสายดาวน์โหลดทางพอร์ตของคอมพิวเตอร์ก็สามารถโปรแกรมตัวชิพขณะที่อยู่ในระบบได้ โดยไม่จำเป็นต้องถอดมาโปรแกรมข้างนอก ดังรูปที่ 3.21 และที่สำคัญสามารถโปรแกรมได้หลายครั้ง จึงทำให้ง่ายในการแก้ไขและพัฒนาโดยไม่ต้องเสียค่าใช้จ่ายเพิ่มแต่อย่างใด



รูปที่ 3.21 การโปรแกรมลงในชิพ [11]

3.13 การออกแบบโดยใช้ภาษาอธิบายพฤติกรรมของฮาร์ดแวร์

ในการออกแบบวงจรดิจิตอลนั้นสามารถทำได้โดยการวาดวงจร (Schematic) หรือใช้ภาษาอธิบายพฤติกรรม (Hardware Description Language) ของฮาร์ดแวร์ ในกรณีของการออกแบบวงจรด้วย ASIC ชนิด Full Custom ผู้ออกแบบจะต้องเขียนวงจรด้วย Schematic จากนั้นจะนำวงจรที่ออกแบบไว้ไปทำการจำลองการทำงาน (Simulate) ซึ่งหากผลออกมาเป็นที่พอใจก็จะต้อง Layout เป็นชั้นสาร และในการออกแบบ ASIC ชนิดนี้ผู้ออกแบบจำเป็นจะต้องทราบถึงเทคโนโลยีที่ใช้ในการสร้างด้วย หลังจากได้ Layout ที่สมบูรณ์แล้วจึงจะส่งไปเข้ากระบวนการสร้างไอซีหรือ Fabrication เพื่อสร้างเป็นชิพไอซีออกมา

แต่ในการออกแบบวงจรด้วยเอฟพีจีเอ โดยการใช้ Schematic หรือใช้ภาษาอธิบายการทำงานของวงจรจะทำได้สะดวกกว่า เนื่องจากวิธีการนี้ผู้ออกแบบไม่จำเป็นต้องคำนึงถึงเทคโนโลยีที่จะใช้สร้างไอซีและที่สำคัญ การออกแบบโดยวิธีนี้สามารถแก้ไขโมเดล (Model) หรือเปลี่ยนแปลงเทคโนโลยีได้สะดวกกว่า เพราะไม่ต้องวาดวงจรใหม่ นั่นคือการออกแบบโดยใช้ภาษาอธิบายฮาร์ดแวร์ จะทำให้โมเดลที่ได้ไม่ขึ้นกับเทคโนโลยี สำหรับภาษาที่ใช้สำหรับอธิบายพฤติกรรมของฮาร์ดแวร์ที่ใช้กันก็มี VHDL, AHDL และ Verilog เป็นต้น ส่วนรายละเอียดของขั้นตอนในการออกแบบสามารถอธิบายได้ดังนี้

3.13.1 การสังเคราะห์วงจร (Logic Synthesis)

ในขั้นตอนนี้จะใช้ซอฟต์แวร์ในการสังเคราะห์วงจร (Synthesis Tools) ทำการสังเคราะห์พฤติกรรมของวงจรที่ได้จากการออกแบบด้วย Schematic หรือวีเอชดีแอลซึ่งต้องทำการตรวจสอบด้วยว่าซอฟต์แวร์นั้นสนับสนุนเทคโนโลยีเอฟพีจีเอ (FPGA Library) ที่ต้องการหรือไม่ ตัวอย่างเช่น เอฟพีจีเอของบริษัท XILINX และบริษัท ALTERA จะมีซอฟต์แวร์หลายตัวที่สามารถใช้ได้ เช่น Max Plus II ในขั้นตอนนี้ซอฟต์แวร์สังเคราะห์วงจรจะทำการแปลงโค้ดวีเอชดีแอล และทำการ Optimize เพื่อให้ได้วงจรตามเทคโนโลยีที่เลือกใช้ ในการสังเคราะห์วงจรนั้นวงจรระดับเกท (Gate Level) จะไม่เหมาะสมกับโครงสร้างที่มีอยู่ในอุปกรณ์เอฟพีจีเอ ดังนั้นในการ Optimize ซอฟต์แวร์สังเคราะห์วงจร จะต้องทำการ Optimize ให้ได้เป็นวงจรที่ประกอบด้วยกลุ่มของลอจิกที่เหมาะสมกับอุปกรณ์เอฟพีจีเอนั้นๆ จึงทำให้ผลที่ได้มีประสิทธิภาพและในขั้นตอนการสังเคราะห์วงจรนี้ ผู้ออกแบบสามารถกำหนดข้อบังคับสำหรับโมเดลแต่ละตัวได้ เช่น ข้อบังคับในเรื่องเวลา (Timing Constraints) หรือข้อบังคับในเรื่องของพื้นที่ (Area) หรือกำหนดชนิดและตำแหน่งของ I/O ซึ่งข้อบังคับเหล่านี้จะถูกนำไปใช้ในขั้นตอน Optimize เพื่อให้วงจรที่ได้เป็นไปตามที่กำหนด ส่วนสำคัญในการ Optimize คือการเทียบ (Mapping) โมเดลให้เข้ากับเทคโนโลยีที่ใช้เพื่อให้ได้วงจรที่เหมาะ

สมกับโครงสร้างและสถาปัตยกรรมภายในอุปกรณ์เอฟพีจีเอ เมื่อทำการสังเคราะห์วงจรเสร็จแล้ว ซอฟแวร์การสังเคราะห์วงจรก็จะมีการรายงานผลว่าโมเดลที่ออกแบบไปนั้นเป็นอย่างไร เช่นมีค่าความล่าช้าของเวลา (Delay) เท่าใด ใช้ทรัพยากรต่างๆในเอฟพีจีเออะไรบ้าง เมื่อมาถึงขั้นตอนนี้ ผู้ออกแบบก็จะทราบว่าโมเดลเป็นไปตามข้อบังคับหรือไม่ ถ้าไม่ก็สังเคราะห์ใหม่จนกว่าจะเป็นไปตามที่กำหนด

3.13.2 **การแบ่งวงจร** (Partitioning)

ขั้นตอนนี้เป็นการแบ่งวงจรที่ได้จากการสังเคราะห์ เป็นส่วนย่อยๆ สำหรับลงใน Configurable Logic Blocks (CLBs) , Input / Output Blocks (IOBs) หรือองค์ประกอบอื่นๆ ภายในอุปกรณ์เอฟพีจีเอ ส่วนของ CLBs จะเป็นส่วนที่บรรจุลอจิกเกตุต่างๆเอาไว้และส่วนของ IOBs จะเป็นส่วนของอินพุทหรือเอาท์พุทที่สามารถโปรแกรมได้ สำหรับเกณฑ์ที่ใช้ในการแบ่งคือให้แต่ละส่วนที่จะแยกออกจากกันมีจำนวนสัญญาณที่เชื่อมต่อระหว่างกันน้อยที่สุดเท่าที่จะทำได้ เพื่อลดความหนาแน่นในตอนทำการเชื่อมต่อสัญญาณ (routing) ในขั้นตอนนี้จะใช้ซอฟต์แวร์ทำโดยซอฟต์แวร์จะเทียบส่วนประกอบของวงจรเช่น เกท (gate), ฟลิป-ฟลอบ (flip-flop) ลงในทรัพยากรต่างๆ ที่มีอยู่ในอุปกรณ์เอฟพีจีเอ

หลังจากทำขั้นตอนนี้เสร็จแล้วผู้ออกแบบสามารถที่จะทราบว่าวงจรใช้จำนวนทรัพยากรภายในอุปกรณ์เอฟพีจีเอไปเท่าไร ส่วนข้อมูลทางเวลานั้นผู้ออกแบบจะทราบเฉพาะความล่าช้าของเวลาภายในแต่ละส่วนเท่านั้น หรือที่เรียกว่าความล่าช้าของเวลาลอจิก (logic delay) ส่วนซอฟต์แวร์จะรวมเอาซอฟต์แวร์ย่อยอื่นๆ อีก เพื่อให้การทำ PPR (Partitioning Placement & Routing) เป็นไปอย่างต่อเนื่อง

3.13.3 **การวางอุปกรณ์** (Placement)

ขั้นตอนนี้เป็นการเลือกทำเลที่ตั้งของแต่ละส่วนของวงจรที่ผ่านการแบ่งวงจร (Partitioning) มาแล้วว่าควรจะอยู่ ณ. ตำแหน่งไหนในอุปกรณ์เอฟพีจีเอ เพื่อให้ได้ผลลัพธ์ดีที่สุด เช่นวงจรส่วนไหนควรอยู่ใกล้กัน เพื่อจะได้ค้นหาเส้นทางได้ (route) ง่ายหรือช่วยลดความล่าช้าของเวลา จะเห็นได้ว่าตำแหน่งภายในอุปกร์ณ์เอฟพีจีเอนั้นมีความสำคัญเพราะถ้าจัดวางวงจรลงในตำแหน่งที่ไม่เหมาะสมแล้วจะทำให้ความล่าช้าของเวลาเพิ่มขึ้นหรือ Router ทำการค้นหาเส้นทางสัญญาณได้ไม่หมด

การวางอุปกรณ์ที่ดีควรวางส่วนต่างๆ ให้อยู่ใกล้กันโดยเฉพาะส่วนที่มีการเชื่อมต่อสัญญาณด้วยกันนอกจากนั้นการกำหนดตำแหน่งขา I/O (I/O pin) ตามตำแหน่งขา I/O ของ เอฟพีจีเอบนแผ่น PCB ก็จะมีผลโดยตรงทันทีคือซอฟต์แวร์จะวาง I/O ลงในตำแหน่งที่ผู้ออกแบบ

กำหนด ซึ่งบางครั้งตำแหน่งที่กำหนดไปไม่เหมาะสม ดังนั้นการกำหนดขา I/O ควรกำหนดตำแหน่งให้เหมาะสม หรือไม่ก็ให้ซอฟต์แวร์จัดการเอง

3.13.4 **การเชื่อมต่อสัญญาณ** (Routing)

ในขั้นตอนนี้เป็นการเชื่อมต่อสัญญาณระหว่างองค์ประกอบต่างๆ ภายในอุปกรณ์เอฟพีจี-เอ ขั้นตอนนี้จะทำต่อเนื่องจากการวางอุปกรณ์ ในกรณีที่ทำการวางอุปกรณ์ไว้ไม่ดีซอฟต์แวร์ก็จะทำการเชื่อมต่อสัญญาณได้ไม่หมด (เนื่องจากจำนวนทรัพยากรสำหรับเชื่อมต่อสัญญาณนั้นมีอยู่จำกัด) หรือเกิดความหน่วงเกินค่าที่กำหนดในข้อบังคับ

ผู้ออกแบบสามารถทำขั้นตอนนี้ได้โดยใช้ซอฟท์แวร์หรือผู้ออกแบบจะทำการเชื่อมต่อสัญญาณด้วยตนเองก็ได้ แต่ทางที่ดีควรใช้ซอฟต์แวร์ทำดีกว่า นอกจากนั้นการกำหนดข้อบังคับทางเวลา จะช่วยให้ผลที่ได้จากการเชื่อมต่อสัญญาณดีขึ้นได้

3.13.5 **ความล่าช้าของเวลา** (Delay)

ในการทำเอฟพีจีเอนั้นความล่าช้าของเวลาที่เกิดขึ้นเป็นความล่าช้าของเวลาที่เกิดจากการวางตำแหน่ง (layout) ของอุปกรณ์ ซึ่งผู้ออกแบบไม่สามารถเข้าไปแก้ไขได้ แต่สามารถทำให้มีความล่าช้าของเวลาน้อยที่สุดได้ สำหรับความล่าช้าของเวลาที่เกิดขึ้นนั้นแยกได้เป็นสองประเภท คือ

- ความล่าช้าของเวลาลอจิก (Logic delay) เป็นความล่าช้าของเวลาภายในองค์ประกอบของอุปกรณ์เอฟพีจีเอเอง

- ความล่าช้าของเวลาที่เกิดจากการเชื่อมต่อสัญญาณ (Routing delay) เป็นความล่าช้าของเวลาที่เกิดจากการเชื่อมต่อสัญญาณระหว่างองค์ประกอบภายในอุปกรณ์เอฟพีจีเอ

โดยปกติแล้วค่าความล่าช้าของเวลาลอจิกไม่ควรเกิน 50% ของค่าความล่าช้าของเวลาที่ยอมรับได้ เพราะความล่าช้าของเวลาที่เกิดจากการเชื่อมต่อสัญญาณมักจะมีค่ามากกว่าค่าความล่าช้าของเวลาลอจิก ดังนั้นในการวางอุปกรณ์ และเชื่อมต่อสัญญาณ ผู้ออกแบบควรกำหนดข้อบังคับทางเวลาเพื่อให้ ซอฟต์แวร์ได้ทำงานอย่างมีประสิทธิภาพเพิ่มขึ้น และเพื่อให้ได้ผลลัพธ์ที่ดีขึ้น

ค่าความล่าช้าของเวลาที่ได้หลังจากการวางอุปกรณ์ และเชื่อมต่อสัญญาณแล้วจะมีค่าความล่าช้าของเวลาที่ค่อนข้างแน่นอน ซึ่งผู้ออกแบบสามารถทราบได้ว่าโมเดลที่ออกแบบนั้น เป็นไปตามข้อกำหนดหรือไม่

3.13.6 **การจำลองการทำงานของวงจร** (Simulation)

ในขั้นตอนนี้เป็นขั้นตอนที่สำคัญอีกขั้นตอนหนึ่ง เพราะเป็นขั้นตอนที่ผู้ออกแบบตรวจสอบฟังก์ชันการทำงานของโมเดลว่าถูกต้องหรือไม่ มีข้อผิดพลาดตรงไหนเพื่อจะได้ทำการแก้ไขให้ถูกต้อง ในขั้นตอนนี้จะมีซอฟต์แวร์ที่ใช้สำหรับทำการจำลองการทำงานของวงจรที่ใช้อยู่ เช่น Model Sim ของบริษัท Model Technology หรือ Max Plus II ของบริษัท Altera ในการจำลองการทำงานของวงจรควรทำทุกครั้งหลังจากที่มีการทำแต่ละขั้นตอนหลักเสร็จแล้ว เพื่อจะได้ทราบว่าข้อผิดพลาดของโมเดลเกิดขึ้นตอนไหน จะได้แก้ไขข้อผิดพลาดตรงขั้นตอนนั้นๆ ได้ทันที ไม่ต้องมาคอยตรวจหาขั้นตอนที่ทำให้เกิดข้อผิดพลาด นั่นคือการทำการจำลองการทำงานของวงจร ต้องทำทั้งหลังการเขียนโค้ด, การสังเคราะห์วงจร และการทำ PPR การจำลองการทำงานของวงจรหลังจากที่เขียนโค้ดเสร็จแล้วนั้น ผู้ออกแบบสามารถทราบได้เฉพาะโมเดลทำงานถูกต้องหรือไม่เท่านั้น (functional test) ยังไม่สามารถตรวจสอบการทำงานในเชิงเวลาได้ถูกต้อง ในการจำลองการทำงานของวงจรหลังจากที่สังเคราะห์เป็นวงจรแล้ว เพื่อตรวจสอบว่าฟังก์ชันการทำงานยังคงถูกต้องหรือไม่ และค่าความล่าช้าของเวลาที่เกิดขึ้นเป็นไปตามข้อบังคับหรือไม่ มีข้อผิดพลาดเกิดขึ้นหรือไม่ถ้ามีจะแก้ไขให้ถูกต้อง

ในการจำลองการทำงานของวงจรหลังจากที่ทำการวางอุปกรณ์ การเชื่อมต่อสัญญาณ (post layout simulation) แล้วก็มีความสำคัญเช่นกัน เพราะผลที่ได้จากการจำลองการทำงานของวงจรในตอนนี้จะเป็นผลลัพธ์ของโมเดลเท่านั้น ซึ่งผู้ออกแบบนอกจากจะตรวจสอบฟังก์ชันการทำงานแล้วยังต้องตรวจสอบคุณสมบัติอื่นๆ เช่น ความล่าช้าของเวลาที่ได้จากการทำ PPR ในรูปแบบค่าความหน่วงมาตรฐาน (Standard Delay Format : SDF) ว่าตรงตามที่กำหนดหรือไม่ หรือตรวจสอบว่าวงจรรวมสามารถใช้งานที่ความถี่สูงสุดเท่าไรนั่นเอง ในการจำลองการทำงานของ วงจรควรใช้ ซอฟต์แวร์ตัวเดียวกันตลอดเพื่อจะได้เปรียบเทียบผลที่ได้จากขั้นตอนต่างๆ

3.13.7 **การโปรแกรมอุปกรณ์เอฟพีจีเอ** (Configuration)

หลังจากที่โมเดลผ่านขั้นตอนต่างๆ จนกระทั่งผ่านการทำ PPR (Partitioning, Placement & Routing) แล้วนั้น ถึงตอนนี้ก็สามารถที่จะดาวน์โหลด (download) ลงในอุปกรณ์เอฟพีจีเอได้แล้ว ในการดาวน์โหลดนี้ก่อนอื่นต้องแปลงแบบวงจรรวมที่ได้เป็นข้อมูลวงจร (configuration data) ซึ่งอยู่ในรูปของบิตสตรีม (bit stream) ก่อนแล้วจึงดาวน์โหลดลงไปเพื่อให้อุปกรณ์เอฟพีจีเอมีฟังก์ชันการทำงานตามโมเดลที่ผู้ออกแบบต้องการ ซึ่งในขั้นตอนนี้จะใช้วิธีที่แตกต่างกันออกไปสำหรับอุปกรณ์เอฟพีจีเอของแต่ละบริษัทผู้ผลิตคือ ในกรณีที่เป็นอุปกรณ์เอฟพีจีเอชนิดที่ต้องโปรแกรมโดยวิธี SRAM นั้น ในการใช้งานผู้ออกแบบจะต้องเก็บข้อมูลวงจรไว้ในหน่วยความจำประเภท EPROM หรือ Serial PROM ด้วยเพื่อจะใช้งานสะดวกขึ้น คือในการใช้งานโมเดลครั้งต่อ

ไปไม่ต้องดาวน์โหลดข้อมูลวงจรจากเครื่องคอมพิวเตอร์อีก เพราะมีข้อมูลวงจรเก็บอยู่ในหน่วยความจำอยู่แล้ว แต่กรณีที่อุปกรณ์เอฟพีจีเอเป็นชนิดที่โปรแกรมโดยใช้วิธี EPROM หรือ Anti fuse ก็ไม่จำเป็นต้องมีหน่วยความจำสำหรับเก็บข้อมูลวงจร เพราะว่าอุปกรณ์เอฟพีจีเอชนิดนี้เมื่อดาวน์โหลดข้อมูลวงจรลงไป ข้อมูลที่ดาวน์โหลดลงไปก็ยังคงอยู่ในอุปกรณ์เอฟพีจีเอและครั้งต่อไปก็ใช้งานโมเดลที่ออกแบบไว้ได้ทันที

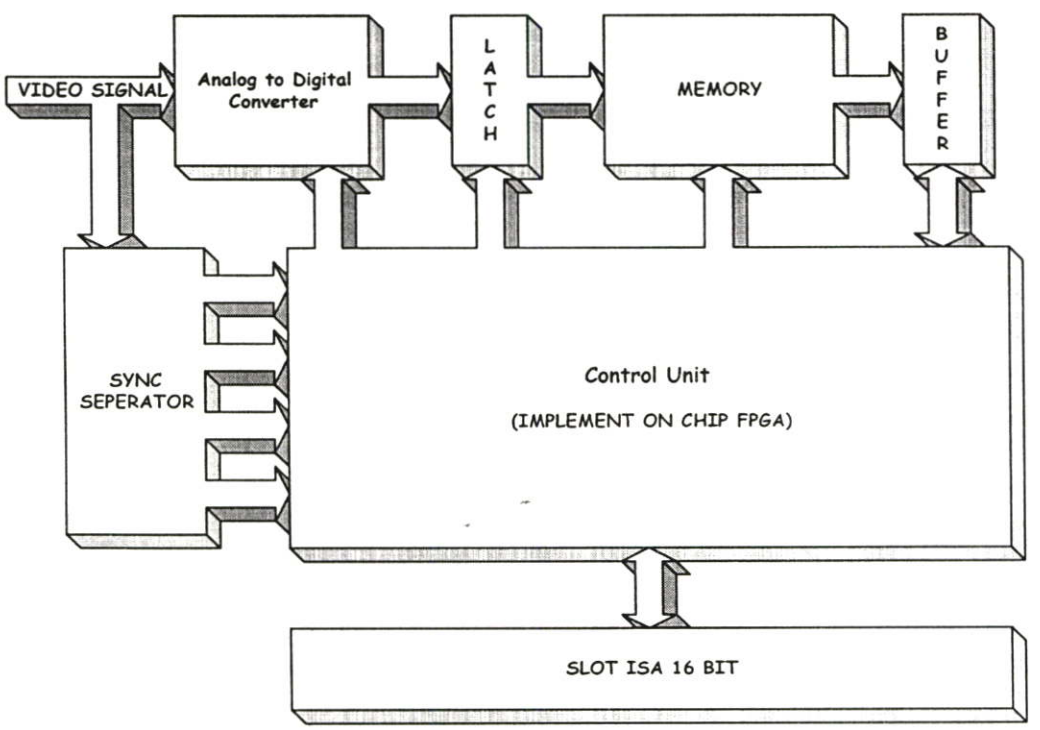
3.14 เครื่องมือสำหรับการออกแบบเอฟพีจีเอ

จะเห็นได้ว่าการออกแบบเพื่อทำเอฟพีจีเอนั้นทำได้สะดวกกว่า ASIC มากเพราะใช้เวลาน้อยกว่ามากด้วย ส่วนสำคัญที่ใช้ในการออกแบบเอฟพีจีเอคือ ซอฟต์แวร์ที่ใช้ตั้งแต่เขียนโค้ดอธิบายฮาร์ดแวร์จนกระทั่งดาวน์โหลดลงในอุปกรณ์เอฟพีจีเอ ซึ่งซอฟต์แวร์ที่ใช้ต้องเป็น ซอฟต์แวร์ที่ทำงานต่อเนื่องกันได้ สำหรับซอฟต์แวร์ที่ใช้ทำการจำลองการทำงานของวงจรนั้น ต้องสามารถใช้งานต่อเนื่องกับซอฟต์แวร์ที่ใช้ทั้งระบบ เพราะโมเดลที่ได้จากการทำขั้นตอนต่างๆ (ด้วยซอฟต์แวร์ต่างๆ) ต้องเอามาจำลองการทำงานได้ และในการจำลองการทำงานของวงจรควรใช้ซอฟต์แวร์ตัวเดียวกันตลอดทั้งระบบ เพื่อจะได้เปรียบเทียบผลได้ง่าย ในอดีตซอฟต์แวร์ส่วนใหญ่จะใช้งานอยู่บนคอมพิวเตอร์สมรรถนะสูงอย่างเวิร์คสเตชั่น (Workstation) ในปัจจุบันมีการพัฒนาซอฟต์แวร์ที่ใช้บนพีซี (PC) มากขึ้นซึ่งสามารถลดค่าใช้จ่ายในด้านอุปกรณ์คอมพิวเตอร์ลงได้

บทที่ 4
การออกแบบและสร้างเครื่องแปลงสัญญาณภาพทางการแพทย์โดยใช้เอฟพีจีเอ

เครื่อง MRI หรือ CT ที่ใช้อยู่ในปัจจุบัน ในการแสดงผลจะใช้วิธีการส่งสัญญาณคอมโพสิทวิดีโอเข้าไปที่จอมอนิเตอร์ของระบบ ในส่วนของสัญญาณคอมโพสิทวิดีโอจะมีลักษณะคล้ายกับสัญญาณคอมโพสิทวิดิโอทั่วๆไปเช่น ระบบ NTSC หรือระบบ PAL แต่ความถี่ที่ใช้ในเครื่อง MRI หรือ CT จะมีค่าความถี่ที่สูงกว่า สัญญาณที่ส่งมาจากเครื่อง MRI หรือ CT จะมีตั้งแต่สัญญาณที่เหมือนมาตรฐานของระบบ NTSC หรือ PAL และสัญญาณที่ไม่เป็นไปตามมาตรฐานทั่วไปดังที่ยกตัวอย่างมาในบทที่ 2 โดยส่วนใหญ่แล้วสัญญาณจากเครื่อง MRI หรือ CT จะมีความถี่ทางด้านฮอร์ริซอลทรัลสูง เหตุผลก็เพื่อที่จะทำให้ภาพที่มอนิเตอร์มีรายละเอียด (Resolution) ที่มากพอที่จะทำการวินิจฉัยได้ ดังนั้นการออกแบบเครื่องแปลงสัญญาณภาพทางการแพทย์ จึงไม่ควรที่จะกำหนดที่ความถี่ใดความถี่หนึ่ง

4.1 ลักษณะโครงสร้างโดยรวมของระบบ

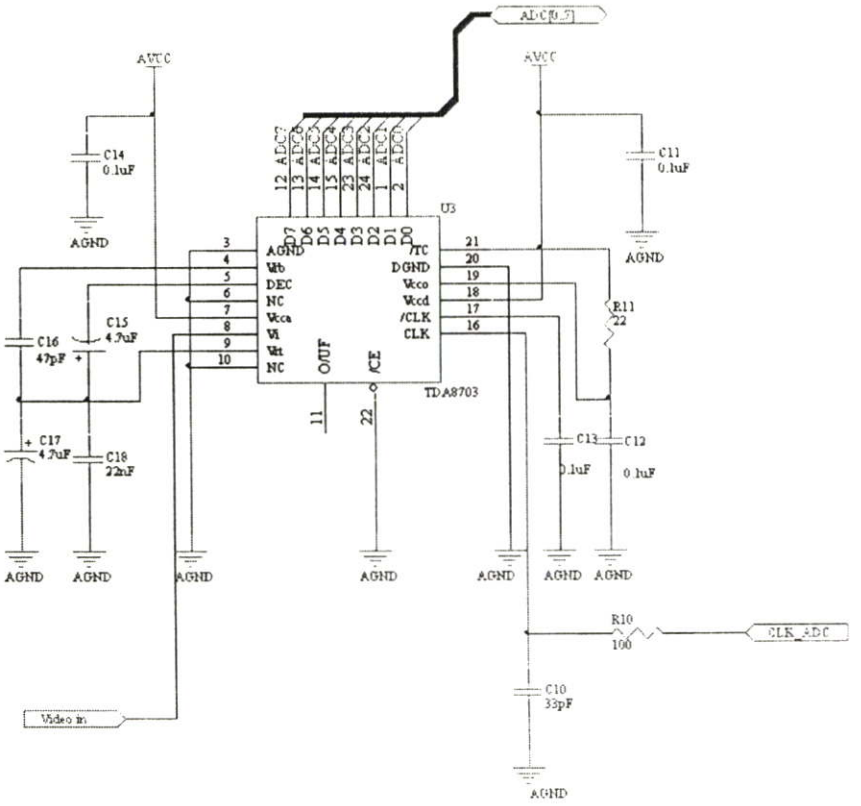


รูปที่ 4.1 บล็อกไดอะแกรมของระบบการเก็บข้อมูลภาพ

จากบล็อกไดอะแกรม ดังรูปที่ 4.1 เป็นบล็อกไดอะแกรมโดยรวมทั้งหมดของระบบการเก็บข้อมูลภาพจากเครื่องมือแพทย์ ซึ่งจะประกอบด้วยส่วนของ Analog to Digital Converter, Sync Separator, Latch, Memory, Buffer และส่วนควบคุมซึ่งสร้างบนชิพเอฟพีจีเอ ซึ่งในแต่ละส่วนก็จะทำหน้าที่แตกต่างกันดังนี้

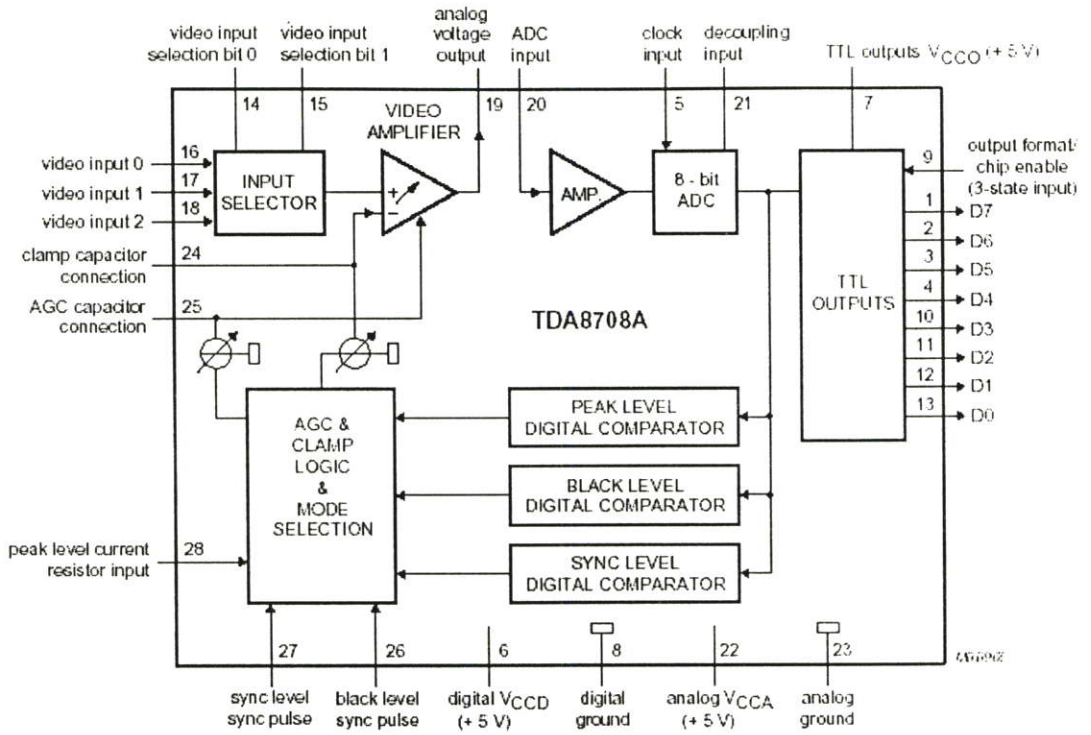
4.1.1 Analog to Digital Converter

ส่วนของ Analog to Digital Converter (A/D) ส่วนนี้จะทำหน้าที่ในการแปลงสัญญาณอนาลอกที่อยู่ในรูปสัญญาณคอมโพสิทวีดีโอให้กลายเป็นสัญญาณดิจิตอล เพื่อที่จะให้สามารถเก็บไว้ในหน่วยความจำได้ วงจรของส่วน A/D จะเป็นดังรูปที่ 4.2



รูปที่ 4.2 วงจรแปลงสัญญาณอนาลอกเป็นดิจิตอล

ไอซี A/D ที่ใช้จะเป็นไอซีเบอร์ TDA8708A เป็นไอซีของบริษัทฟิลลิปส์ ทำหน้าที่ในการแปลงสัญญาณอนาลอกให้เป็นดิจิตอล ซึ่งโครงสร้างภายในของ TDA8708A เป็นดังรูปที่ 4.3



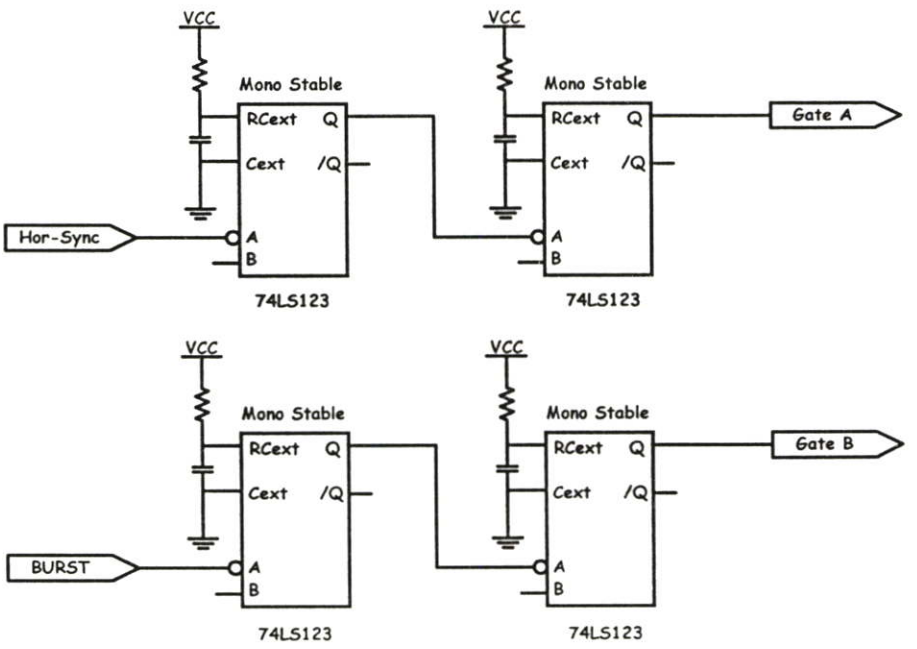
รูปที่ 4.3 โครงสร้างภายในของ TDA8708A

เหตุผลที่เลือกใช้ไอซีเบอร์นี้เพราะว่าเป็นไอซีที่ผลิตมาเพื่อใช้สำหรับการแปลงสัญญาณอนาลอกที่เป็นสัญญาณวีดีโอโดยเฉพาะ โดยที่ภายใน TDA8708A จะประกอบด้วยส่วนของการเลือกสัญญาณอินพุทว่าจะเลือกเอาสัญญาณอินพุทจากอินพุทช่องที่ 1,2หรือ 3 สัญญาณที่ได้จากการเลือกแล้วจะนำไปขยายสัญญาณ โดยภาค Video Amplifier พร้อมทั้งทำการยกระดับของสัญญาณ และควบคุมเกณฑ์การขยายด้วยภาค Clamp and Gain Control หลังจากนั้นจึงนำสัญญาณไปแปลงจากสัญญาณอนาลอกให้เป็นสัญญาณดิจิตอล และไอซีเบอร์นี้ยังมีอัตราการสุ่มข้อมูล (Sampling Rate) ที่ค่อนข้างสูงคืออยู่ที่ 32MHz นอกจากนี้ยังมีโหมดการทำงาน 2 โหมดด้วยกัน คือ

1.โหมดหนึ่งค่าเกณฑ์ของ AGC Amplifier จะถูกทำการปรับอย่างหยาบ โดยทำการตั้งค่าให้ระดับ Sync มีค่าเท่ากับ 0 ส่วนระดับ Peak ของสัญญาณจะถูกกำหนดที่ 255

2.โหมดสองค่าดิจิตอลทางด้านเอาท์พุทของ A/D ถูกนำมาเปรียบเทียบกับค่าอ้างอิงภายใน ดังนั้นจึงทำให้ Peak White Control ทำงานถ้าหากเมื่อไรก็ตามที่ค่า Digital ทางด้านเอาท์พุทมีค่าที่เกินจาก 213 Peak White Control จะทำการควบคุมให้ค่าเอาท์พุทที่ได้รับมีค่าไม่เกินจากค่า 248 เพื่อเป็นการหลีกเลี่ยงสภาวะการเกิด Over-Range ขึ้น

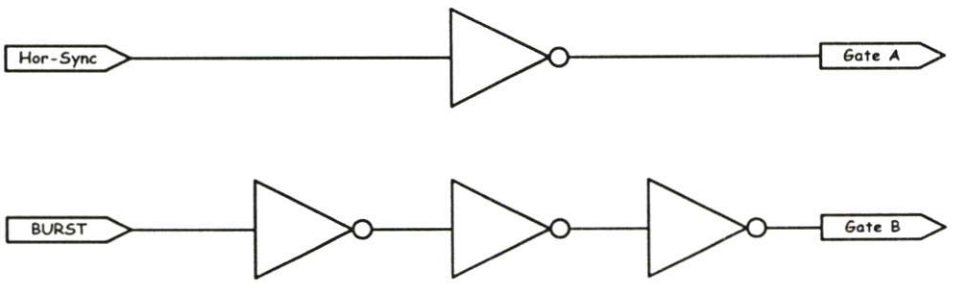
การกำหนดโหมดการทำงานของ TDA8708A จะให้ทำงานในโหมดไหนนั้นก็ขึ้นอยู่กับช่วงขอบขาขึ้นของพัลส์ Gate A และ Gate B สัญญาณ Gate A และ Gate B จะเป็นสัญญาณควบคุมให้ TDA8708A ทำงานในโหมดหนึ่งหรือโหมดสอง โดยที่ Gate A&B Generator จะใช้สัญญาณที่ได้รับมาจากขา 3 และขา 5 ของไอซีเบอร์ LM1881 จากวงจรแยกสัญญาณซิงค์ ซึ่งสัญญาณดังกล่าวจะเป็นสัญญาณ Hor-Sync และ Burst นำสัญญาณดังกล่าวมาทำการกำหนดให้เป็น Gate A และ Gate B ซึ่งก่อนที่จะส่งไปเข้าไอซี TDA8708A จะต้องทำการกำหนดจุดเริ่มต้นและความกว้างของพัลส์ เพื่อให้เกิดความเหมาะสม กับการทำงานของไอซีเบอร์ TDA8708A การกำหนดจะทำโดยการส่งผ่านสัญญาณทั้งสองไปยังชุดของโมโนสเตเบิ้ลที่ทำการต่อคาสเคทกันสองชุด โดยที่ชุดแรกจะกำหนดจุดเริ่มต้นให้กับพัลส์ ในขณะที่ชุดที่สองจะกำหนดความกว้างของพัลส์ ซึ่งไอซีที่จะนำมาใช้เป็นโมโนสเตเบิ้ลคือ 74LS123 ต่อกันเป็นวงจรดังรูปที่ 4.4



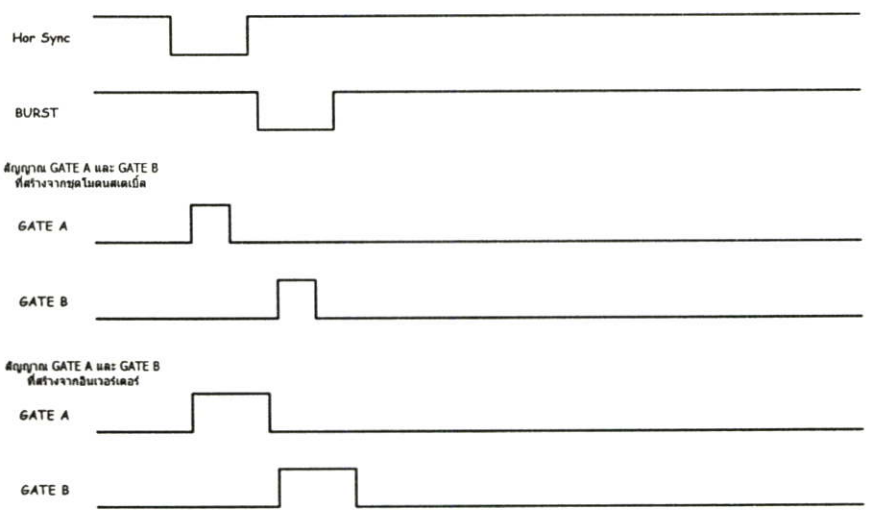
รูปที่ 4.4 วงจรกำเนิดสัญญาณ Gate A และ Gate B โดยใช้โมโนสเตเบิ้ล

ในการต่อดังรูปที่ 4.4 จะมีความสะดวกในการที่จะทำการปรับค่าของ Time ได้ตามต้องการ แต่ถ้าหากว่าไม่มีความประสงค์ในการที่จะปรับค่า Time ก็สามารถที่จะทำการออกแบบวงจรได้ใหม่ซึ่งก็จะเป็นการลดขนาดของวงจรลงได้ โดยการนำเอาสัญญาณ Hor-Sync และ Burst ที่ได้จากการแยกสัญญาณซิงค์มาผ่านวงจรอินเวอร์เตอร์ เพื่อทำการกลับเฟสของสัญญาณที่ได้รับในขณะเดียวกันก็อาศัยคุณสมบัติของอินเวอร์เตอร์ ในเรื่องของเวลาที่ล้าช้าทางด้านเอาท์พุทเมื่อเทียบกับทางด้านอินพุท จึงสามารถนำเอาสัญญาณดังกล่าวมาใช้เป็นสัญญาณ Gate A &

Gate B เพื่อป้อนให้กับไอซี TDA8708A ได้ทันทีลักษณะการต่อวงจรแสดงดังรูปที่ 4.5 ซึ่งในงานวิจัยนี้ได้เลือกใช้งานวงจรกำเนิดสัญญาณ Gate A และ Gate B ตามรูปที่ 4.5 เช่นกัน



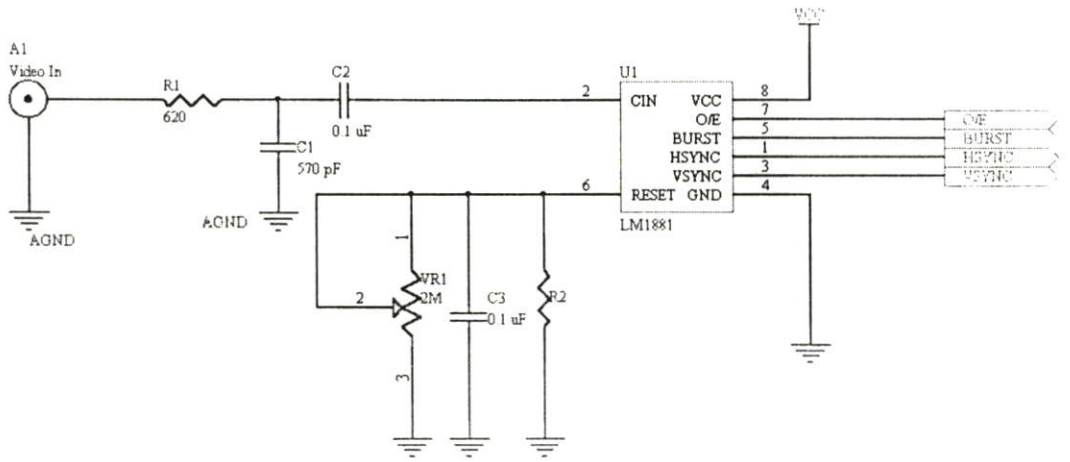
รูปที่ 4.5 วงจรกำเนิดสัญญาณ Gate A และ Gate B อย่างง่ายโดยใช้อินเวอร์เตอร์



รูปที่ 4.6 สัญญาณ Gate A และ Gate B เมื่อทำการเปรียบเทียบกับสัญญาณ Hor Sync และสัญญาณ BURST

4.1.2 Sync Separator

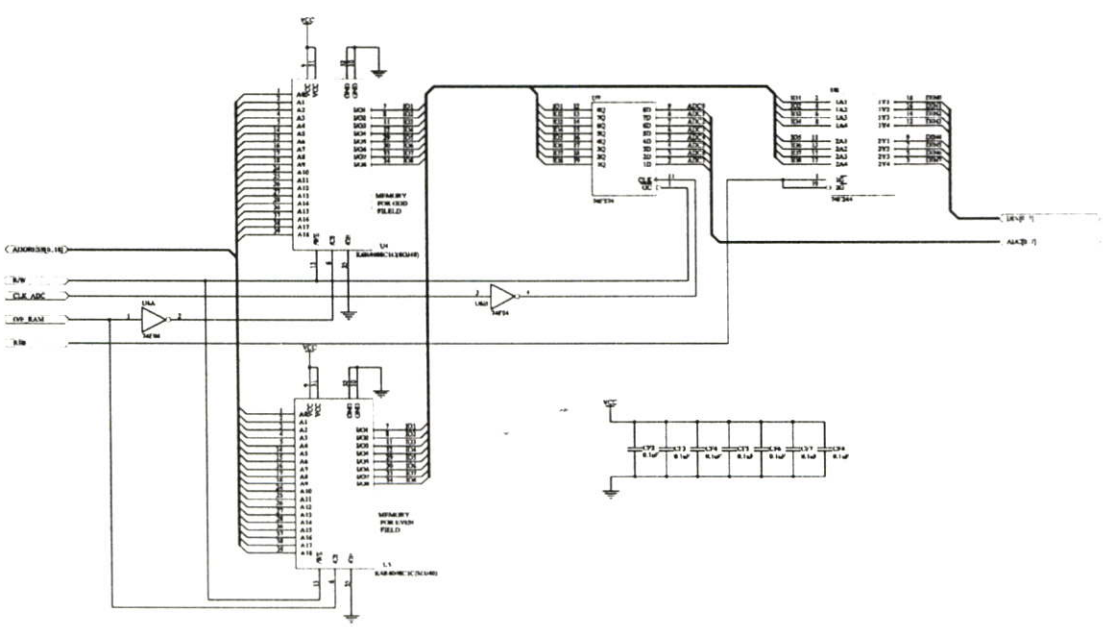
ในส่วนนี้จะเป็นส่วนที่ทำหน้าที่ในการแยกสัญญาณซิงค์ออกจากสัญญาณคอมโพซิทวีดีโอ ซึ่งสัญญาณที่ถูกแยกออกมาแล้ว จะประกอบไปด้วยสัญญาณ Hor Sync, Vertical Sync, Burst/Back Porch, Odd/Even Field ไอซีที่ใช้จะเป็นไอซีเบอร์ LM1881 ของบริษัทเนชั่นแนลจะมีย่านการใช้งานที่ค่อนข้างกว้าง สามารถใช้งานได้กับสัญญาณการสแกนทางด้านแนวนอนสูงถึง 150 KHz ซึ่งในการใช้งานสามารถที่จะทำการเลือกย่านที่เหมาะสมได้ โดยการปรับเปลี่ยนค่าของ Rset ให้เหมาะสมสามารถแสดงการต่อวงจรได้ดังรูปที่ 4.7



รูปที่ 4.7 วงจรแยกสัญญาณซิงค์

4.1.3 หน่วยความจำ

หน่วยความจำที่ใช้จะเป็นหน่วยความจำแบบ Static RAM และจะใช้การเก็บข้อมูลดิจิตอลลงสู่หน่วยความจำโดยตรง โดยจะใช้หน่วยความจำเบอร์ K6R4008C1C-JC12 ซึ่งเป็นหน่วยความจำที่มีความจุ 512 KByte และมีค่า Access Time เท่ากับ 12 nS ในการออกแบบหน่วยความจำเพื่อเก็บข้อมูลภาพ จะต้องออกแบบให้หน่วยความจำ สามารถจัดเก็บข้อมูลภาพได้ทั้งฟิลด์คู่และฟิลด์คี่ สามารถแสดงวงจรสำหรับหน่วยความจำได้ดังรูปที่ 4.8



รูปที่ 4.8 วงจรหน่วยความจำ

4.1.4 **หน่วงข้อมูล** (Latch)

จะเป็นส่วนของวงจรที่ทำหน้าที่ในการหน่วงข้อมูลที่ได้รับจาก A/D ก่อนที่จะส่งเข้าไปยังหน่วยความจำเพื่อทำให้ช่วงจังหวะในการจัดเก็บข้อมูลเข้าไปยังหน่วยความจำมีความเหมาะสมกับคุณลักษณะของหน่วยความจำที่ใช้

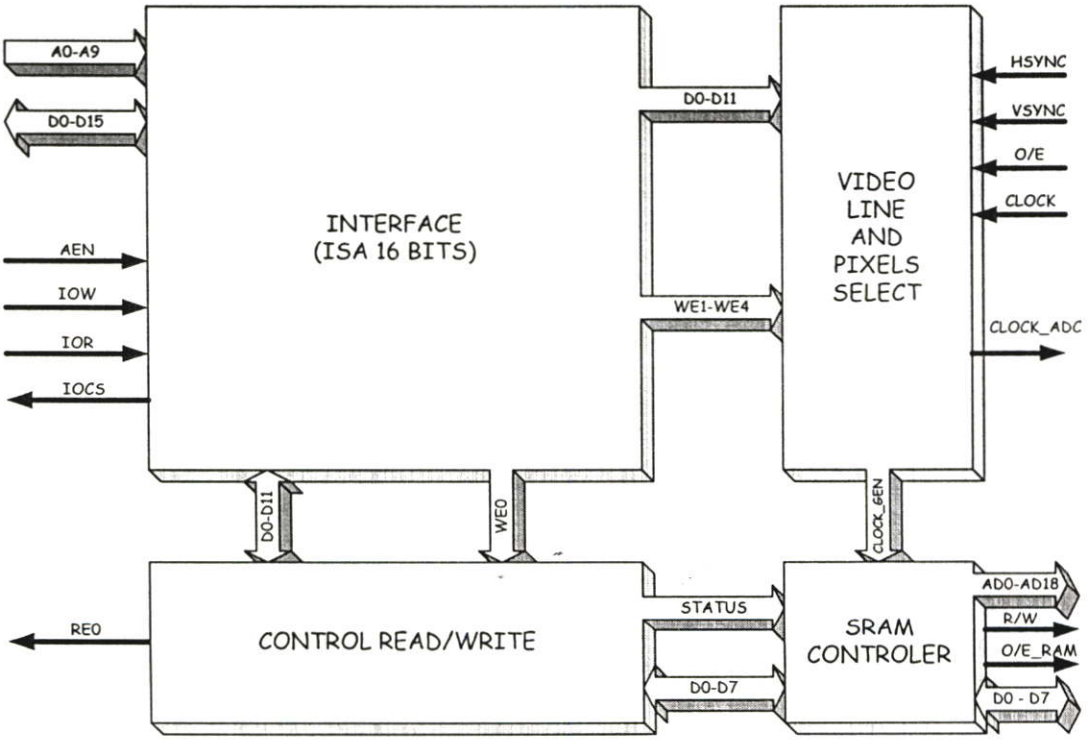
4.1.5 **บัฟเฟอร์**

จะทำหน้าที่ในการขับสัญญาณข้อมูลที่ได้จากหน่วยความจำเพื่อทำการส่งไปยังชุดควบคุมที่ได้ออกแบบด้วยซิพเอฟพีจีเอต่อไป

4.1.6 **ชุดควบคุมโดยเอฟพีจีเอ**

ในส่วนนี้จะเป็นชุดควบคุมที่ควบคุมการทำงานทั้งหมดของวงจรแปลงข้อมูลภาพทางการแพทย์ ซึ่งจะใช้ซิพเอฟพีจีเอในการควบคุมการทำงานทั้งหมด และจะกล่าวถึงรายละเอียดของส่วนควบคุมในหัวข้อถัดไป

4.2 สถาปัตยกรรมของชุดควบคุมการทำงาน

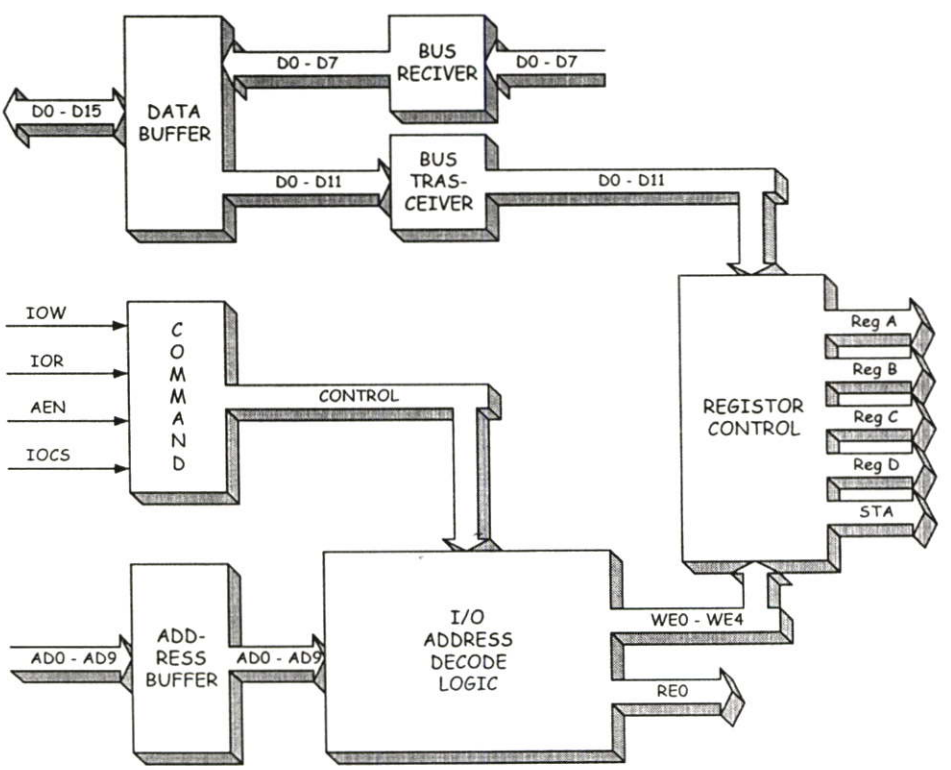


รูปที่ 4.9 บล็อกไดอะแกรมโดยรวมของชุดควบคุมการทำงาน

จากบล็อกไดอะแกรมดังรูปที่ 4.1 จะเห็นว่าชุดควบคุมของเครื่องแปลงข้อมูลภาพทางการแพทย์จะใช้ชิพเอฟพีจีเอ ในการประมวลผลและควบคุมการทำงานของส่วนอื่นๆ หากเลือกใช้อุปกรณ์ TTL ทั่วไปจะทำให้วงจรที่ได้จะมีขนาดใหญ่ [12] ภายในชิพเอฟพีจีเอเองก็ยังประกอบด้วยส่วนประกอบต่างๆ อีกหลายส่วนด้วยกันดังรูปที่ 4.9 ภายในชุดควบคุมการทำงานของเครื่องแปลงข้อมูลภาพทางการแพทย์จะประกอบด้วย ส่วนของการเชื่อมต่อกับคอมพิวเตอร์ผ่านทางสล็อต ISA แบบ 16 บิท ส่วนของการเลือกเส้นสแกนและเลือกพิกเซลที่จะทำการจัดเก็บ ส่วนของชุดควบคุมหน่วยความจำและส่วนของชุดควบคุมการเขียน การอ่านข้อมูลภาพที่จะทำการจัดเก็บ ในส่วนของรายละเอียดของส่วนต่างๆ จะกล่าวในหัวข้อต่อๆ ไป

4.3 สถาปัตยกรรมของชุดอินเตอร์เฟส ISA แบบ 16 บิท

ในส่วนนี้จะเป็นส่วนของการอินเตอร์เฟสกับคอมพิวเตอร์ผ่านทางสล็อต ISA แบบ 16 บิท ซึ่งจะทำหน้าที่ในการรับคำสั่งต่างๆที่ส่งมาจากคอมพิวเตอร์ เพื่อที่จะนำไปกำหนดค่าพารามิเตอร์ให้กับส่วนอื่นๆ ในชิพเอฟพีจีเอ สามารถแสดงโครงสร้างของชุดอินเตอร์เฟส ISA แบบ 16 บิท ได้ดังรูปที่ 4.10



รูปที่ 4.10 บล็อกไดอะแกรมภายในชุดอินเตอร์เฟสกับสล็อต ISA แบบ 16 บิท

ภายในชุดอินเตอร์เฟสจะมีรีจีสเตอร์ที่สำคัญอยู่ 5 ตัว ดังนี้

1. Reg STA เป็นรีจีสเตอร์ขนาด 4 บิท สำหรับควบคุมสถานะต่างๆ ภายในชิฟเอฟพีจีเอ เช่น ควบคุมการเขียน การอ่านข้อมูล การเลือกฟิลด์ของภาพ การสร้าง Address ให้แก่หน่วยความจำเป็นต้น

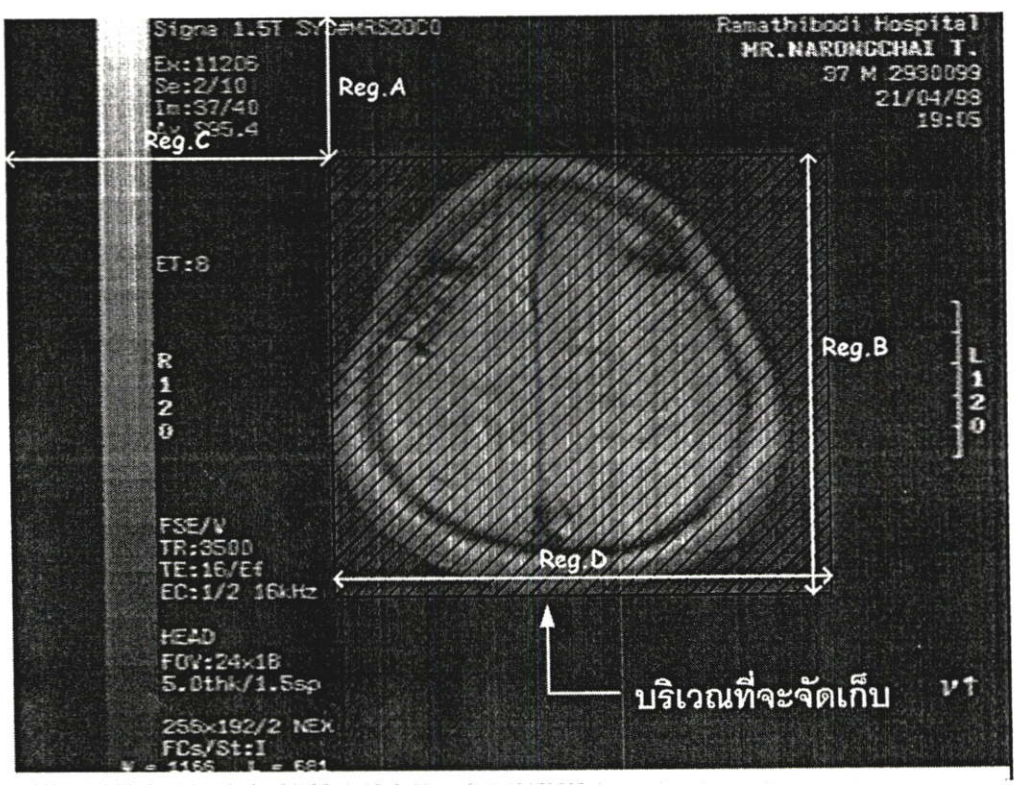
2. Reg A เป็นรีจีสเตอร์ขนาด 10 บิท สำหรับกำหนดตำแหน่งของข้อมูลภาพที่เป็นเส้นสแกนเส้นแรกที่จะทำการจัดเก็บลงสู่หน่วยความจำ

3. Reg B เป็นรีจีสเตอร์ขนาด 10 บิท สำหรับกำหนดจำนวนเส้นสแกนทั้งหมดที่จะทำการจัดเก็บลงสู่หน่วยความจำ

4. Reg C เป็นรีจีสเตอร์ขนาด 10 บิท สำหรับกำหนดตำแหน่งของข้อมูลภาพที่เป็นพิกเซลแรกที่จะทำการจัดเก็บลงสู่หน่วยความจำ

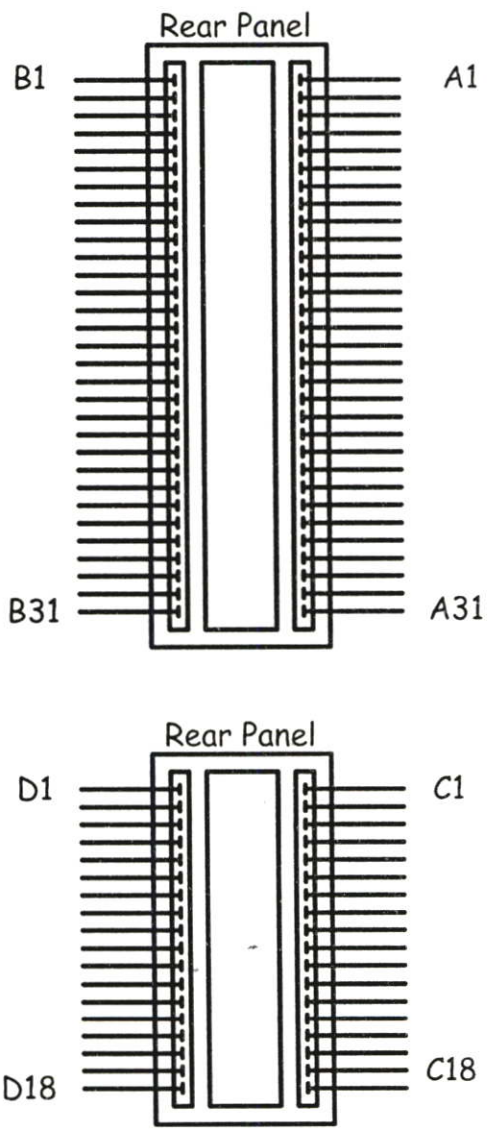
5. Reg D เป็นรีจีสเตอร์ขนาด 10 บิท สำหรับกำหนดจำนวนพิกเซลทั้งหมดในหนึ่งเส้นสแกนที่จะทำการจัดเก็บลงสู่หน่วยความจำ

ความสัมพันธ์ระหว่างรีจีสเตอร์ต่างๆแสดงดังรูปที่ 4.11



รูปที่ 4.11 ความสัมพันธ์ระหว่างรีจีสเตอร์ต่างๆสำหรับการกำหนดตำแหน่งและขนาดของข้อมูลภาพที่จะจัดเก็บ

การ์ดอินเตอร์เฟสที่ได้ทำการออกแบบจะทำการเชื่อมต่อเข้ากับเครื่อง IBM/PC โดยผ่านทางสล็อตที่อยู่บนเมนบอร์ด (MAIN BORD) ภายในสล็อต ISA จะประกอบไปด้วยชุดคอนเนคเตอร์ 2 ชุด คือ ชุดคอนเนคเตอร์ 62 ขากับ 36 ขา ชุดของคอนเนคเตอร์ 62 ขา จะแบ่งออกเป็น 2 ข้างๆ ละ 31 ขา โดยขาที่อยู่ด้านซ้ายของสล็อตจะถูกเรียกโดยใช้ตัวอักษร "B" นำหน้าเลขตำแหน่งของขา ส่วนขาที่อยู่ด้านขวาของสล็อตจะเรียกโดยใช้อักษร "A" นำหน้าเลขตำแหน่งของขา ส่วนชุดคอนเนคเตอร์ 36 ขา จะแบ่งออกเป็น 2 ข้าง เช่นเดียวกัน โดยข้างซ้ายจะถูกเรียกโดยใช้อักษร "D" นำหน้าเลขตำแหน่งของขา ส่วนด้านขวาจะถูกเรียกโดยใช้ตัวอักษร "C" นำหน้าเลขตำแหน่งของขา ชุดของคอนเนคเตอร์ทั้งหมดสามารถแสดงรายละเอียดให้เห็นดังรูป 4.12



รูปที่ 4.12 ตำแหน่งขาของสล๊อตคอมพิเตอร์ [13]

ขาของคอนเนคเตอร์แต่ละขาสามารถที่จะทำการแสดงในรูปความหมายของสัญญาณได้ดังตารางที่ 4.1 และ ตารางที่ 4.2 และสามารถแสดงหน้าที่ของแต่ละสัญญาณได้ดังตารางที่ 4.3

ตาราง 4.1 สัญญาณทางด้าน A และ B

ขา I/O	ชื่อสัญญาณ	I/O	ขา I/O	ชื่อสัญญาณ	I/O
A1	-I/O CHCK	I	B1	GND	Ground
A2	SD7	I/O	B2	RESET DRV	O
A3	SD6	I/O	B3	+5Vdc	Power
A4	SD5	I/O	B4	IRQ9	I
A5	SD4	I/O	B5	-5Vdc	Power
A6	SD3	I/O	B6	DRQ2	I
A7	SD2	I/O	B7	-12Vdc	Power
A8	SD1	I/O	B8	Not Uesd	Not Uesd
A9	SD0	I/O	B9	+12Vdc	Power
A10	-I/O CHRDY	I	B10	GND	Ground
A11	AEN	O	B11	-SMEMW	O
A12	SA19	O	B12	-SMEMR	O
A13	SA18	O	B13	-IOW	O
A14	SA17	O	B14	-IOR	O
A15	SA16	O	B15	-DACK3	O
A16	SA15	O	B16	DRQ3	I
A17	SA14	O	B17	-DACK1	O
A18	SA13	O	B18	DRQ1	I
A19	SA12	O	B19	-Refresh	O
A20	SA11	O	B20	CLK	O
A21	SA10	O	B21	IRQ7	I
A22	SA9	O	B22	IRQ6	I
A23	SA8	O	B23	IRQ5	I
A24	SA7	O	B24	IRQ4	I
A25	SA6	O	B25	IRQ3	I

ตาราง 4.1 (ต่อ)

ขา I/O	ชื่อสัญญาณ	I/O	ขา I/O	ชื่อสัญญาณ	I/O
A26	SA5	O	B26	IRQ2	I
A27	SA4	O	B27	T/C	O
A28	SA3	O	B28	BALE	O
A29	SA2	O	B29	+5Vdc	Power
A30	SA1	O	B30	OSC	O
A31	SA0	O	B31	GND	Ground

ตารางที่ 4.2 สัญญาณทางด้าน C และ D

ขา I/O	ชื่อสัญญาณ	I/O	ขา I/O	ชื่อสัญญาณ	I/O
C1	SBHE	O	D1	-MEM CS 16	I
C2	LA23	O	D2	-I/O CS16	I
C3	LA22	O	D3	IRQ10	I
C4	LA21	O	D4	IRQ11	I
C5	LA20	O	D5	IRQ12	I
C6	LA19	O	D6	IRQ13	I
C7	LA18	O	D7	IRQ14	I
C8	LA17	O	D8	-DACK0	O
C9	-MEMR	O	D9	DRQ0	I
C10	MEMW	O	D10	-DACK5	O
C11	SD08	I/O	D11	DRQ6	I
C12	SD09	I/O	D12	-DACK6	O
C13	SD10	I/O	D13	DRQ6	I
C14	SD11	I/O	D14	-DACK7	O
C15	SD12	I/O	D15	+5Vdc	Power
C16	SD13	I/O	D16	+5Vdc	Power
C17	SD14	I/O	D17	-MASTER	I
C18	SD15	I/O	D18	GND	Ground

ตารางที่ 4.3a หน้าที่ของสัญญาณทางด้าน A

ขา I/O	ชื่อสัญญาณ	I/O	คำอธิบาย
A1	-I/O CHCK	I	แสดงความผิดพลาดเกี่ยวกับพาริตี้ที่เกิดขึ้นในการทำงานของวงจรอินเตอร์เฟสหรืออุปกรณ์ I/O
A2	SD7	I/O	ข้อมูลบิตที่ 7
A3	SD6	I/O	ข้อมูลบิตที่ 6
A4	SD5	I/O	ข้อมูลบิตที่ 5
A5	SD4	I/O	ข้อมูลบิตที่ 4
A6	SD3	I/O	ข้อมูลบิตที่ 3
A7	SD2	I/O	ข้อมูลบิตที่ 2
A8	SD1	I/O	ข้อมูลบิตที่ 1
A9	SD0	I/O	ข้อมูลบิตที่ 0
A10	-I/O CHRDY	I	ใช้เพิ่มช่วงเวลาในบัสไซเคิลในกรณีที่อุปกรณ์ I/O หรือหน่วยความจำที่เกี่ยวข้องกับขบวนการในบัสไซเคิลที่เกิดขึ้นนั้นไม่สามารถทำงานทันตามช่วงเวลาปกติของบัสไซเคิลนั้นๆ
A11	AEN	O	บัสไซเคิลที่เกิดขึ้นในช่วงเวลาที่สัญญาณ AEN แอคตีฟ เป็นบัสไซเคิลของขบวนการ DMA
A12	SA19	O	สัญญาณแอดแดรสบิตที่ 19
A13	SA18	O	สัญญาณแอดแดรสบิตที่ 18
A14	SA17	O	สัญญาณแอดแดรสบิตที่ 17
A15	SA16	O	สัญญาณแอดแดรสบิตที่ 16
A16	SA15	O	สัญญาณแอดแดรสบิตที่ 15
A17	SA14	O	สัญญาณแอดแดรสบิตที่ 14
A18	SA13	O	สัญญาณแอดแดรสบิตที่ 13
A19	SA12	O	สัญญาณแอดแดรสบิตที่ 12
A20	SA11	O	สัญญาณแอดแดรสบิตที่ 11
A21	SA10	O	สัญญาณแอดแดรสบิตที่ 10
A22	SA9	O	สัญญาณแอดแดรสบิตที่ 9

ตารางที่ 4.3a (ต่อ)

ขา I/O	ชื่อสัญญาณ	I/O	คำอธิบาย
A23	SA8	O	สัญญาณแอดแดรสบิตที่ 8
A24	SA7	O	สัญญาณแอดแดรสบิตที่ 7
A25	SA6	O	สัญญาณแอดแดรสบิตที่ 6
A26	SA5	O	สัญญาณแอดแดรสบิตที่ 5
A27	SA4	O	สัญญาณแอดแดรสบิตที่ 4
A28	SA3	O	สัญญาณแอดแดรสบิตที่ 3
A29	SA2	O	สัญญาณแอดแดรสบิตที่ 2
A30	SA1	O	สัญญาณแอดแดรสบิตที่ 1
A31	SA0	O	สัญญาณแอดแดรสบิตที่ 0

ตารางที่ 4.3b หน้าที่ของสัญญาณทางด้าน B

ขา I/O	ชื่อสัญญาณ	I/O	คำอธิบาย
B1	GND	Ground	Ground ของระบบ
B2	RESET DRV	O	ใช้ในการรีเซ็ทวงจรอินเตอร์เฟสหรืออุปกรณ์ I/O ต่างๆ ในช่วงที่เริ่มจ่ายไฟให้กับระบบ
B3	+5Vdc	Power	แหล่งจ่ายไฟ 5 โวลท์
B4	IRQ9	I	สำหรับทำการขออินเตอร์รัพท์
B5	-5Vdc	Power	แหล่งจ่ายไฟ -5 โวลท์
B6	DRQ2	I	สำหรับทำการขอ DMA
B7	-12Vdc	Power	แหล่งจ่ายไฟ -12 โวลท์
B8	Not Uesd	Not Uesd	Not Uesd
B9	+12Vdc	Power	แหล่งจ่ายไฟ 12 โวลท์
B10	GND	Ground	Ground ของระบบ
B11	-SMEMW	O	เขียนข้อมูลลงในหน่วยความจำ
B12	-SMEMR	O	อ่านข้อมูลจากหน่วยความจำ
B13	-IOW	O	บัสไซเคิลที่เกิดขึ้นเป็นบัสไซเคิลของการเขียนข้อมูลลงบนพอร์ท I/O

ตารางที่ 4.3b (ต่อ)

ขา I/O	ชื่อสัญญาณ	I/O	คำอธิบาย
B14	-IOR	O	บัสไซเคิลที่เกิดขึ้นเป็นบัสไซเคิลของการอ่านข้อมูลจากพอร์ท I/O
B15	-DACK3	O	สัญญาณตอบสนองการขอ DMA
B16	DRQ3	I	สำหรับทำการขอ DMA
B17	-DACK1	O	สัญญาณตอบสนองการขอ DMA
B18	DRQ1	I	สำหรับทำการขอ DMA
B19	-Refresh	O	สัญญาณสำหรับรีเฟรสหน่วยความจำ
B20	CLK	O	สัญญาณคล็อกของระบบ
B21	IRQ7	I	สำหรับทำการขออินเตอร์รัพท์
B22	IRQ6	I	สำหรับทำการขออินเตอร์รัพท์
B23	IRQ5	I	สำหรับทำการขออินเตอร์รัพท์
B24	IRQ4	I	สำหรับทำการขออินเตอร์รัพท์
B25	IRQ3	I	สำหรับทำการขออินเตอร์รัพท์
B26	IRQ2	I	สำหรับทำการขออินเตอร์รัพท์
B27	T/C	O	แอคตีฟเมื่อจำนวนไบท์ในการส่งผ่านข้อมูลของขบวนการ DMA ในแชนแนลใดแชนแนลหนึ่งครบตามจำนวนที่กำหนดไว้
B28	BALE	O	แสดงให้อุปกรณ์ภายนอกทราบว่าแอดเดรสที่ CPU ต้องการติดต่อด้วยนั้นถูกส่งออกมาบนบัสแอดเดรสแล้ว
B29	+5Vdc	Power	แหล่งจ่ายไฟ 5 โวลท์
B30	OSC	O	คล็อกความถี่สูงสุดจากเมนบอร์ดไม่ Synchronize กับสัญญาณอื่นๆ
B31	GND	Ground	Ground ของระบบ

ตารางที่ 4.3c หน้าที่ของสัญญาณทางด้าน C

ขา I/O	ชื่อสัญญาณ	I/O	คำอธิบาย
C1	SBHE	O	แสดงให้ทราบว่ามีการส่งผ่านของข้อมูลในส่วนของไบต์สูงลงไปยังบัสข้อมูล
C2	LA23	O	สัญญาณแอดแดรสบิตที่ 23
C3	LA22	O	สัญญาณแอดแดรสบิตที่ 22
C4	LA21	O	สัญญาณแอดแดรสบิตที่ 21
C5	LA20	O	สัญญาณแอดแดรสบิตที่ 20
C6	LA19	O	สัญญาณแอดแดรสบิตที่ 19
C7	LA18	O	สัญญาณแอดแดรสบิตที่ 18
C8	LA17	O	สัญญาณแอดแดรสบิตที่ 17
C9	-MEMR	O	แอคตีฟ (ลอจิก "0") ในระหว่างบัสไซเคิลของการอ่านข้อมูลจากหน่วยความจำ
C10	MEMW	O	แอคตีฟ (ลอจิก "0") ในระหว่างบัสไซเคิลของการเขียนข้อมูลลงหน่วยความจำ
C11	SD08	I/O	ข้อมูลบิตที่ 8
C12	SD09	I/O	ข้อมูลบิตที่ 9
C13	SD10	I/O	ข้อมูลบิตที่ 10
C14	SD11	I/O	ข้อมูลบิตที่ 11
C15	SD12	I/O	ข้อมูลบิตที่ 12
C16	SD13	I/O	ข้อมูลบิตที่ 13
C17	SD14	I/O	ข้อมูลบิตที่ 14
C18	SD15	I/O	ข้อมูลบิตที่ 15

ตารางที่ 4.3d หน้าที่ของสัญญาณทางด้าน D

ขา I/O	ชื่อสัญญาณ	I/O	คำอธิบาย
D1	-MEM CS 16	I	เป็นการแสดงว่ามีการส่งผ่านข้อมูล 16 บิทในช่วงไซเคิลของหน่วยความจำพร้อมกับ 1 WAIT STATE

ตารางที่ 4.3d (ต่อ)

ขา I/O	ชื่อสัญญาณ	I/O	คำอธิบาย
D2	-I/O CS16	I	เป็นการแสดงว่ามีการส่งผ่านข้อมูล 16 บิทในช่วงไซเคิลของ I/O พร้อมกับ 1 WAIT STATE
D3	IRQ10	I	สำหรับทำการขออินเตอร์รัพท์
D4	IRQ11	I	สำหรับทำการขออินเตอร์รัพท์
D5	IRQ12	I	สำหรับทำการขออินเตอร์รัพท์
D6	IRQ13	I	สำหรับทำการขออินเตอร์รัพท์
D7	IRQ14	I	สำหรับทำการขออินเตอร์รัพท์
D8	-DACK0	O	สัญญาณตอบสนองการขอ DMA
D9	DRQ0	I	สำหรับทำการขอ DMA
D10	-DACK5	O	สัญญาณตอบสนองการขอ DMA
D11	DRQ6	I	สำหรับทำการขอ DMA
D12	-DACK6	O	สัญญาณตอบสนองการขอ DMA
D13	DRQ6	I	สำหรับทำการขอ DMA
D14	-DACK7	O	สัญญาณตอบสนองการขอ DMA
D15	+5Vdc	Power	แหล่งจ่ายไฟ 5 โวลท์
D16	+5Vdc	Power	แหล่งจ่ายไฟ 5 โวลท์
D17	-MASTER	I	จะถูกใช้กับสัญญาณ DRQ เพื่อที่จะทำให้การควบคุมของระบบให้ดีขึ้นกว่าเดิม
D18	GND	Ground	Ground ของระบบ

แต่ละขาของสล๊อตเหล่านี้จะเชื่อมต่อกับสัญญาณต่างๆ บนเมนบอร์ด ทำให้การสร้างวงจรอินเตอร์เฟสกับ IBM/PC สามารถทำได้โดยถูกต้อง ซึ่งเส้นสัญญาณที่เชื่อมต่อกับขาของสล๊อตเหล่านี้ประกอบด้วย เส้นสัญญาณของบัสแอสเดรัส (ADDRESS BUS) บัสข้อมูล (DATA BUS) บัสควบคุมสำหรับการเขียน/อ่านข้อมูลจากหน่วยความจำ หรือพอร์ท I/O เส้นสัญญาณสำหรับการขออินเตอร์รัพท์ของวงจรอินเตอร์เฟส เส้นสัญญาณสำหรับการขอ DMA สัญญาณฐานเวลา (TIMING SIGNAL) ต่างๆที่ใช้ในระบบ เส้นสัญญาณแสดงการรีเฟรชหน่วยความจำ และสัญญาณสำหรับการตรวจสอบความผิดพลาด (I/O CHCK)

นอกจากเส้นสัญญาณเหล่านี้แล้ว สล๊อตบนเมนบอร์ดยังเชื่อมต่อกับแหล่งจ่ายไฟต่างๆ ที่ใช้ในระบบอีกด้วย คือ +5Vdc, -5Vdc, +12Vdc และ -12Vdc สามารถแสดงรายละเอียดเกี่ยวกับสัญญาณต่างๆดังนี้

OSC (OSCILLATOR) ขานี้เป็นเอาท์พุทที่เชื่อมต่อกับสัญญาณคล็อกที่มีค่าความถี่สูงสุดบนเมนบอร์ด และมี DUTY CYCLE (ช่วงเวลาใน 1 คาบสัญญาณคล็อกมีลอจิกเป็น "1" หารด้วยคาบเวลาทั้งหมด) ประมาณ 50% สัญญาณคล็อกอื่นๆ นั้นจะถูกสร้างขึ้นโดยการหารสัญญาณคล็อกนี้ อย่างไรก็ตามสิ่งหนึ่งที่จะต้องคำนึงถึงในการใช้สัญญาณ OSC ก็คือสัญญาณนี้จะไม่ SYNCHRONIZE กับสัญญาณอื่นๆ บนบัสของระบบ ดังนั้นจึงไม่ควรที่จะนำสัญญาณจาก OSC นี้ใช้เป็นสัญญาณคล็อกสำหรับวงจรภายนอกอื่นๆ ที่ทำงานร่วมกับระบบ

CLK (CLOCK) ขาสัญญาณนี้เป็นเอาท์พุท ซึ่งต่อกับสัญญาณคล็อกที่ถูกสร้างขึ้นโดยการหารสัญญาณ OSC ด้วย 3 ค่าของ DUTY CYCLE ของสัญญาณนี้จะมีค่าประมาณ 1/3 คือ ใน 1 คาบจะมีช่วงเวลาที่เป็นลอจิก "1" เท่ากับ 1/3 ของคาบเวลาทั้งหมด และช่วงเวลาที่เป็นลอจิก "0" เท่ากับ 2/3 ของคาบเวลาทั้งหมด สัญญาณนี้จะถูกใช้เป็นสัญญาณคล็อกของระบบ

RESET DRV ขาสัญญาณนี้เป็นเอาท์พุท ซึ่งจะแอทตีฟ (ลอจิก "1") ในช่วงที่เริ่มจ่ายไฟให้กับระบบ และจะยังคงแอทตีฟไปจนกว่าระบบต่างๆ ภายใน IBM/PC พร้อมที่จะทำงาน จากนั้นสัญญาณนี้ก็จะเปลี่ยนกลับเป็นลอจิก "0" นอกจากนี้ในระหว่างการทำงานของ IBM/PC ถ้าระดับแรงดันของแหล่งจ่ายไฟตกลง สัญญาณนี้ก็ถูกทำให้แอทตีฟเช่นกัน โดยทั่วไปแล้วสัญญาณนี้จะถูกนำไปใช้ในการรีเซ็ทวงจรอินเตอร์เฟสหรืออุปกรณ์ I/O ต่างๆ ในช่วงที่เริ่มจ่ายไฟให้กับระบบซึ่งจะเป็นการทำให้วงจรหรืออุปกรณ์เหล่านี้ถูกปรับให้อยู่ในสภาวะที่แน่นอน ก่อนที่จะเริ่มต้นการทำงานในระบบ

SA0-SA19 (SYSTEM ADDRESS BUS) ขาสัญญาณทั้ง 20 ขานี้เป็นเอาท์พุท ซึ่งใช้สำหรับกำหนดแอดเดรสของหน่วยความจำหรืออุปกรณ์ I/O ที่ต้องการติดต่อด้วย โดยที่สัญญาณ A0 จะมีนัยสำคัญต่ำสุด (LEAST SIGNIFICANT BIT) และ A19 จะมีนัยสำคัญสูงสุด (MOST SIGNIFICANT BIT) สำหรับค่าแอดเดรสบนบัสแอดเดรส A0-A19 นี้จะถูกกำหนดในช่วงระหว่างขบวนการอ่าน/เขียนข้อมูลลงในหน่วยความจำหรืออุปกรณ์ I/O แต่ในช่วงของขบวนการ DMA นั้น DMA CONTROLLER จะเป็นผู้ทำการกำหนดแอดเดรสบนบัสแอดเดรสเอง (ในช่วงระหว่างดังกล่าวนี้ CPU จะถูกทำการตัดออกจากระบบ)

จะเห็นได้ว่าจำนวนเส้นแอดเดรสนี้จะมีอยู่ 20 เส้น ทำให้สามารถที่จะทำการอ้างแอดเดรสของหน่วยความจำได้ถึง 1 Mbyte แต่อย่างไรก็ตามจะมีแอดเดรสบางแอดเดรสที่ถูกใช้งานโดย IBM/PC อยู่ก่อนแล้ว และแอดเดรสสำหรับหน่วยความจำ ROM อีก 48 Kbyte ซึ่งถูกจัดในช่วงของแอดเดรสบนสุดใน 1 Mbyte คือ 0FC00H จนถึง 0FFFFFH

สำหรับการอ้างแอดเดรสของพอร์ท I/O นั้น จะใช้เส้นแอดเดรสเพียง 16 เส้น คือ A0-A15 ซึ่งจะทำให้อ้างแอดเดรสของพอร์ทได้ 64K พอร์ท โดยผ่านทางชุดคำสั่ง IN และ OUT ส่วนเส้นแอดเดรสที่เหลือคือ A16-A19 นั้นจะไม่ถูกใช้งาน อย่างไรก็ตามภายใน IBM/PC จะมีการใช้เส้นแอดเดรสในการอ้างแอดเดรสของพอร์ทเพียง 10 เส้น คือจาก A0-A9 และแอดเดรสที่ใช้งานจะต้องอยู่ในช่วง 0200H จนถึง 03FFH เท่านั้น

LA17-LA23 ขาสัญญาณนี้จะถูกใช้ในการกำหนดแอดเดรส ให้กับหน่วยความจำและอุปกรณ์ประเภท I/O ที่ต่ออยู่ในระบบ มันจะทำให้ระบบสามารถที่จะทำการอ้างแอดเดรสได้มากถึง 16 Mbyte สัญญาณเหล่านี้จะใช้ได้เมื่อสัญญาณ BALE (ADDRESS LATCH ENABLE) มีค่าเป็น HIGH สัญญาณ LA17-LA23 จะไม่ถูกทำการหน่วงในช่วงระยะไซเคิลของไมโครโปรเซสเซอร์ ดังนั้นจึงใช้ไม่ได้ ในช่วงของไซเคิลดังกล่าววัตถุประสงค์ก็เพื่อที่จะทำการกำเนิดการดีโค้ดหน่วยความจำสำหรับทำให้เกิดไซเคิล WAIT STATE ของหน่วยความจำหนึ่งไซเคิลสัญญาณที่ถูกทำการโค้ดนี้จะถูกทำการหน่วง (LATCH) โดย I/O ADAPTER ตอนช่วงขอบขาลงของสัญญาณ BALE สัญญาณเหล่านี้จะถูกทำการขับโดยไมโครโปรเซสเซอร์อื่นๆ หรือ DMA CONTROLLER ที่อยู่บน I/O CHANNEL

SD0-SD15 (SYSTEM DATA BUS) ขาสัญญาณนี้จะเป็นแบบ BI-DIRECTIONAL ซึ่งต่อกับบัสข้อมูลของระบบ เพื่อทำหน้าที่ในการส่งผ่านข้อมูลระหว่างพอร์ท I/O กับ IBM/PC โดยบิท D0 จะมีนัยสำคัญต่ำสุด และบิท D15 จะมีนัยสำคัญสูงสุด อุปกรณ์ประเภท 8 บิททั้งหมดที่ต่ออยู่บน I/O CHANNEL จะใช้ D0-D7 สำหรับทำการสื่อสารกับไมโครโปรเซสเซอร์ ในขณะที่อุปกรณ์ประเภท 16 บิทก็จะใช้ D0-D15 ในการสื่อสารดังกล่าว ในการสนับสนุนกับอุปกรณ์ประเภท 8 บิท นั้นไมโครโปรเซสเซอร์ขนาด 16 บิทจะอาศัยข้อมูลที่อยู่บน D8-D15 ทำการเปิดเข้าไปยัง D0-D7 ในช่วงระหว่างที่มีการส่งข้อมูลเข้าไปยังอุปกรณ์ดังกล่าว โดยที่ไมโครโปรเซสเซอร์ 16 บิทจะทำการแปลงข้อมูลที่จะทำการส่งผ่านไปยังอุปกรณ์ประเภท 8 บิทออกเป็นสองชุด

สำหรับในบัสไซเคิลของการเขียนข้อมูลนั้น ข้อมูลจะถูกทำการส่งออกมาบนบัสข้อมูลก่อนที่สัญญาณ IOW (ในกรณีที่ต้องการทำการส่งข้อมูลให้กับพอร์ท) หรือ MEMW (ในกรณีที่ต้องการส่งข้อมูลให้กับหน่วยความจำ) จะเปลี่ยนจากลอจิก "0" เป็นลอจิก "1" (ขอบขาขึ้น) ซึ่งโดยทั่วไปขอบขาขึ้นของสัญญาณ IOW หรือ MEMW นี้จะถูกใช้เพื่อที่จะทำการสั่งให้พอร์ท I/O หรือหน่วยความจำที่มีแอดเดรสตรงกับแอดเดรสบนบัสแอดเดรสนั้นทำการรองรับข้อมูลไปทำการเก็บไว้

สำหรับบัสไซเคิลของการอ่านข้อมูลพอร์ท I/O หรือหน่วยความจำที่ถูกอ้างอิงถึงจะต้องส่งข้อมูลออกมาบนบัสข้อมูล ก่อนที่สัญญาณ IOR (ในกรณีที่ต้องการอ่านข้อมูลจากพอร์ท) หรือ

MEMR (ในกรณีที่ต้องการอ่านข้อมูลจากหน่วยความจำ) จะเปลี่ยนจากลอจิก "0" เป็นลอจิก "1" (ขอบขาขึ้น)

BALE ขาสัญญาณนี้เป็นสัญญาณเอาท์พุทที่ถูกสร้างขึ้น เพื่อใช้สำหรับแสดงการเริ่มต้นของบัสไซเคิล และแสดงให้อุปกรณ์ภายนอกทราบว่าแอดเดรสที่ CPU ต้องการติดต่อด้วยนั้นถูกส่งออกมาบนบัสแอดเดรสแล้ว โดยที่สัญญาณ ALE นี้จะเปลี่ยนจากลอจิก "1" เป็น "0" เมื่อค่าแอดเดรสที่ถูกต้องถูกส่งออกมาบนบัสข้อมูลเรียบร้อยแล้ว ดังนั้นขอบขาลงของสัญญาณ ALE นี้จะถูกใช้ในการแลทช์ค่าแอดเดรสจากบัสแอดเดรส / ข้อมูล (ADDRESS/DATA BUS ; AD0-AD7) ของ CPU ทำให้สามารถที่จะทำการแยกค่าแอดเดรส (A0-A19) และข้อมูล (A0-A7) ออกจากกันได้ อย่างไรก็ตามสัญญาณ ALE จะแอคตีฟเฉพาะในบัสไซเคิลทีสร้างขึ้นโดย CPU เท่านั้น โดยจะไม่แอคตีฟในระหว่างขบวนการ DMA

I/O CHCK (I/O CHANNEL CHECK) ขาสัญญาณนี้เป็นอินพุทที่ใช้ในการแสดงความผิดพลาดเกี่ยวกับพาริตี้ ที่เกิดขึ้นในการทำงานของวงจรอินเตอร์เฟสหรืออุปกรณ์ I/O เมื่อขาสัญญาณนี้ได้รับลอจิก "0" ทำให้เกิดสภาวะอินเตอร์รัพท์แบบ NON-MASKABLE (NMI) อย่างไรก็ตามสามารถที่จะกำหนดให้วงจรภายใน IBM/PC ทำการขออินเตอร์รัพท์ (เมื่อได้รับสัญญาณ I/O CHCK) หรือไม่ก็ได้ โดยการกำหนดลอจิกของบิทข้อมูลของพอร์ทที่ควบคุมการขออินเตอร์รัพท์แบบ NMI คือบิท D7 ของพอร์ท 00A0H ในกรณีที่บิท D7 ของพอร์ท 00A0H ถูกเซ็ทเป็น "1" ก็จะทำให้วงจรภายนอกของอินเตอร์รัพท์แบบ NMI ได้ (ENABLE) แต่ถ้าบิท D7 ของพอร์ท 00A0H ถูกเซ็ทเป็น "0" ก็จะเป็นการดิสเอเบิล (DISABLE) การขออินเตอร์รัพท์แบบ NMI ดังนี้

ENABLE : ใช้ทำคำสั่ง OUT ส่งข้อมูล 80H ไปยังพอร์ท 00A0H

DISABLE : ใช้คำสั่ง OUT ส่งข้อมูล 00H ไปยังพอร์ท 00A0H

และเนื่องจากว่ายังมีอุปกรณ์อื่นที่สามารถขออินเตอร์รัพท์แบบ NMI ได้อีก ดังนั้นซอฟท์แวร์ที่ใช้งานจะต้องสามารถตรวจสอบว่าการขออินเตอร์รัพท์นั้นเกิดขึ้นจากแหล่งใดได้ด้วย

I/O CHRDY (I/O CHANNEL READY) ขาสัญญาณนี้เป็นอินพุทที่ใช้เพิ่มช่วงเวลาในบัสไซเคิลในกรณีที่อุปกรณ์ I/O หรือหน่วยความจำที่เกี่ยวข้องกับขบวนการในบัสไซเคิลที่เกิดขึ้นนั้นไม่สามารถทำงานทันตามช่วงเวลาปกติของบัสไซเคิลนั้นๆได้ (ช่วงเวลาของบัสไซเคิลที่เกี่ยวกับหน่วยความจำใช้ช่วงเวลาเท่ากับช่วงเวลาของคล็อก 4 ลูก ในขณะที่บัสไซเคิลที่เกี่ยวกับ I/O จะใช้ช่วงเวลาเท่ากับช่วงเวลาของคล็อก 5 ลูก)

เมื่ออุปกรณ์ I/O หรือหน่วยความจำต้องการที่จะเพิ่มช่วงเวลาในบัสไซเคิลให้นานขึ้นอีกนั้น จะสามารถทำได้โดยการป้อนลอจิก "0" ให้กับขา I/O CHRDY ในช่วงเวลาที่ I/O หรือหน่วยความจำที่ถูกกำหนดนั้น ได้รับสัญญาณจากการดีโค้ดแอดเดรส และสัญญาณ MEMR, MEMW, IOR หรือ IOW แอคตีฟ

IRQ3-IRQ7,IRQ9-IRQ12และIRQ14-IRQ15 (INTERRUPT REQUEST 3 THROUGH7, INTERRUPT REQUEST 9 THROUGH 12 & INTERRUPT REQUEST 14 THROUGH 15) ขาสัญญาณทั้ง 11 ขานี้เป็นขาอินพุทที่ใช้สำหรับทำการขออินเตอร์รัพท์ โดยสัญญาณเหล่านี้จะต่อเข้ากับอุปกรณ์ที่ใช้ในการควบคุมการจัดลำดับความสำคัญ ในการอินเตอร์รัพท์ที่อยู่บนเมนบอร์ดโดยตรง โปรแกรมในส่วน BIOS ของ IBM/PC จะทำการโปรแกรมให้ IRQ9-IRQ12 และ IRQ14-IRQ15 มีลำดับความสำคัญสูง (IRQ9 จะมีลำดับความสำคัญสูงที่สุด (HIGHEST PRIORITY)) และ IRQ3-IRQ7 จะมีลำดับความสำคัญต่ำ (IRQ7 มีลำดับความสำคัญต่ำสุด) ในกรณีที่มีการขออินเตอร์รัพท์เกิดขึ้นคือระดับลอจิกที่ขา IRQ ขาใดขาหนึ่งถูกเปลี่ยนจากลอจิก "0" เป็นลอจิก "1" (ขอบขาขึ้น) อุปกรณ์ที่จัดลำดับความสำคัญในการอินเตอร์รัพท์จะทำการส่งสัญญาณ INT ให้กับ CPU เพื่อทำการขออินเตอร์รัพท์

สิ่งสำคัญในการขออินเตอร์รัพท์โดยผ่านทาง IRQ ต่างๆ เหล่านี้ ก็คืออุปกรณ์ที่ทำการขออินเตอร์รัพท์โดยผ่านทาง IRQ ขาใดก็จะต้องรักษาระดับสัญญาณที่ขา IRQ นั้น ให้แอคตีฟ (ลอจิก "1") อยู่จนกว่าจะได้รับสัญญาณ INTA (INTERRUPT ACKNOWLEDGE) จาก CPU เสียก่อน ถ้าไม่เช่นนั้นการขออินเตอร์รัพท์จะถูกยกเลิกและอินเตอร์รัพท์ LEVEL (IRQ7) จะถูกสร้างขึ้นโดยอัตโนมัติ ไม่ว่าการขออินเตอร์รัพท์ที่ถูกยกเลิกนั้นจะเป็นการขออินเตอร์รัพท์ใน LEVEL หรือขาใด

แต่อย่างไรก็ตามสัญญาณ INTA นี้จะไม่ถูกต่อออกมาที่ขาของสล็อตด้วย ดังนั้นโปรแกรมที่ทำการตอบสนองต่อการขออินเตอร์รัพท์ (INTERRUPT SERVICE ROUTINE) จะต้องทำการรีเซ็ทสัญญาณ IRQ เอง โดยใช้คำสั่ง OUT ไปยังพอร์ท I/O ที่เกี่ยวข้อง

อินเตอร์รัพท์ 13 จะถูกใช้งานโดย SYSTEM BOARD แต่ไม่ได้ถูกใช้งานโดย I/O CHANNEL ส่วนอินเตอร์รัพท์ 8 จะถูกใช้สำหรับการทำ REAL-TIME COLCK

IOR (I/O READ) ขาสัญญาณนี้เป็นเอาท์พุทแอคตีฟทีลอจิก "0" เพื่อจะใช้ในการแสดงว่าบัสไซเคิลที่เกิดขึ้นนี้ เป็นบันไซเคิลของการอ่านข้อมูลจากพอร์ท I/O เพื่อให้พอร์ท I/O ที่มีแอดเดรสตรงกับแอดเดรสบนบัสแอดเดรสนั้นส่งข้อมูลออกมาบนบัสข้อมูล โดยข้อมูลจะต้องถูกส่งออกมาบนบัสข้อมูลก่อนขอบขาขึ้นของสัญญาณ IOR เพื่อที่จะทำให้มั่นใจได้ว่า CPU สามารถรับข้อมูลได้ถูกต้อง สำหรับในขบวนการ DMA DMA CONTROLLER จะทำการสร้างสัญญาณ IOR เอง โดยที่ค่าแอดเดรสที่อยู่บนบัสแอดเดรสจะเป็นค่าแอดเดรสของหน่วยความจำ (แทนที่จะเป็นแอดเดรสของพอร์ท I/O) ที่พอร์ท I/O ที่ขอ DMA ต้องการจะนำข้อมูลไปเก็บ การที่พอร์ทใดจะทำการส่งข้อมูลออกมาบนบัสข้อมูลนั้น จะอาศัยสัญญาณ DACK จาก DMA CONTROLLER เป็นตัวกำหนด เช่นกรณีที่สัญญาณ DACK1 แอคตีฟก็จะแสดงว่าพอร์ท I/O ที่จะต้องส่งข้อมูลออกมาบนบัสข้อมูลก็คือพอร์ท I/O ที่ขอ DMA ผ่านทางแชนแนลที่ 1 (DRQ1) เป็นต้น

IOW (I/O WRITE) ขาสัญญาณนี้เป็นเอาท์พุทแอคตีฟที่ลอจิก "0" ซึ่งถูกสร้างขึ้นโดย BUS CONTROLLER เพื่อใช้แสดงว่าบัสไซเคิลที่เกิดขึ้นเป็นบัสไซเคิลของการเขียนข้อมูลลงบนพอร์ท I/O เพื่อให้พอร์ท I/O ที่มีแอดเดรสตรงกับแอดเดรสบนบัสแอดเดรสนั้น รับข้อมูลที่อยู่บนบัสข้อมูลไปเก็บไว้ อย่างไรก็ตามเนื่องจากในช่วงเวลาที่สัญญาณ IOW นี้แอคตีฟ (ลอจิก "0") นั้นข้อมูลบนบัสอาจจะยังไม่สมบูรณ์ ดังนั้นในการออกแบบจึงควรใช้ขอบขาขึ้นของสัญญาณ IOW แทนขอบขาลงในการทำให้พอร์ท I/O ที่เกี่ยวข้องรับข้อมูลไปเก็บไว้ เพื่อให้ข้อมูลบนบัสข้อมูลสมบูรณ์เสียก่อน สำหรับในขบวนการ DMA นั้น DMA-CONTROLLER จะทำการสร้างสัญญาณ IOW เองโดยที่ค่าแอดเดรสที่อยู่บนบัสแอดเดรสจะเป็นค่าแอดเดรสของหน่วยความจำที่พอร์ท I/O ที่ขอ DMA ต้องการจะอ่านข้อมูล

SMEMW, MEMW (MEMORY WRITE) ขานี้เป็นเอาท์พุทแอคตีฟที่ลอจิก "0" ซึ่ง BUS CONTROLLER สร้างขึ้นในระบบบัสไซเคิลในการเขียนข้อมูลลงในหน่วยความจำสัญญาณ SMEMW จะถูกทำการแอคตีฟก็ต่อเมื่อมีการดีโค้ดหน่วยความจำอยู่ภายใน 1Mbyte ส่วนสัญญาณ MEMW นี้จะถูกส่งออกมาเพื่อให้หน่วยความจำทั้งหมดที่แอดเดรสตรงกับค่าแอดแดรสบนบัสแอดเดรสนั้น ทำการรับข้อมูลที่อยู่บนบัสไปเก็บไว้ โดยทั่วไปหน่วยความจำจะรับข้อมูลในช่วงขอบขาขึ้นของสัญญาณ MEMW

สำหรับในระหว่างกระบวนการ DMA นั้น DMA-CONTROLLER จะทำการควบคุมบัสต่างๆ ของระบบแทน CPU และสัญญาณ MEMW จะถูกใช้ในบัสไซเคิลของการเขียนข้อมูลลงในหน่วยความจำ (ข้อมูลถูกส่งออกจากอุปกรณ์ I/O ไปให้กับหน่วยความจำ)

SMEMR,MEMR (MEMORY READ) ขานี้เป็นสัญญาณเอาท์พุทที่แอคตีฟ (ลอจิก "0") ในระหว่างบัสไซเคิลของการอ่านข้อมูลจากหน่วยความจำ สัญญาณ SMEMR จะถูกทำการแอคตีฟก็ต่อเมื่อทำการติดต่อกับหน่วยความจำที่อยู่ภายใน 1 Mbyte ส่วน MEMR จะทำการแอคตีฟกับหน่วยความจำทั้งหมดเพื่อให้หน่วยความจำที่มีแอดเดรสตรงกับค่าแอดเดรสบนบัสแอดเดรสนั้น ทำการส่งข้อมูลออกมาบนบัสข้อมูล โดยหน่วยความจำนั้นจะต้องส่งข้อมูลออกมา ก่อนที่สัญญาณ MEMW จะกลับเป็นลอจิก "1" ทั้งนี้ก็เพื่อที่จะทำให้ CPU ได้รับข้อมูลที่ถูกต้อง

สำหรับในระหว่างขบวนการ DMA นั้น DMA-CONTROLLER จะทำการควบคุมบัสต่างๆ ของระบบแทน CPU และสัญญาณ MEMR จะถูกใช้ในบัสไซเคิลของการอ่านข้อมูลจากหน่วยความจำ (ข้อมูลถูกส่งจากหน่วยความจำไปให้กับอุปกรณ์ I/O)

DRQ0-DRQ3 และ DRQ6-DRQ7 (DMA REQUEST 0-3 & DMA REQUEST 5-7) ขาสัญญาณทั้งเจ็ดนี้เป็นสัญญาณอินพุทแอคตีฟที่ลอจิก "1" ซึ่งอุปกรณ์ภายนอกสามารถใช้ในการขอ DMA จากระบบ โดยการป้อนระดับสัญญาณลอจิก "1" ให้กับขา DRQ ขาใดขาหนึ่ง

เมื่อ DMA CONTROLLER ได้รับสัญญาณนี้แล้วจะทำการตรวจสอบว่ามีการขอ DMA ในแชนแนลที่มีลำดับความสำคัญ (PRIORITY) สูงกว่าหรือไม่ ถ้าไม่มีก็จะทำการขอ DMA จาก CPU และทำการตอบรับการขอ DMA จากอุปกรณ์ภายนอก (สัญญาณ DACK ของแชนแนลที่ขอ DMA จะแอคตีฟ) แต่ถ้ามี DMA CONTROLLER ก็จะทำการขอ DMA ให้กับแชนแนลที่มีลำดับความสำคัญสูงกว่าก่อนแล้วจึงทำการขอ DMA ให้กับแชนแนลที่มีลำดับความสำคัญต่ำกว่า ภายใน ROM BIOS ของ IBM/PC จะทำการโปรแกรมให้ DMA CONTROLLER จัดลำดับความสำคัญของ DRQ0 มีลำดับความสำคัญสูงสุดและ DRQ7 มีลำดับความสำคัญต่ำสุด ดังนั้นถ้ามีการขอ DMA CONTROLLER ก็จะทำการขอ DMA ให้กับแชนแนลที่ 1 ก่อน จากนั้นเมื่อเสร็จจากขบวนการ DMA ของแชนแนลที่ 1 จึงจะทำการขอ DMA ให้กับแชนแนลที่ 2

สัญญาณ DRQ0-DRQ3 จะถูกนำมาใช้กับขบวนการ DMA แบบ 8 บิท ในขณะที่ DRQ5-DRQ7 จะถูกใช้ในการกระทำแบบ 16 บิท ส่วน DRQ4 ถูกใช้บน SYSTEM BOARD และไม่ถูกนำมาใช้งานบน I/O CHANNEL

ในการขอ DMA นั้นสัญญาณ DRQ นี้จะต้องแอคตีฟอยู่ในช่วงระยะเวลาหนึ่งเท่านั้น ถ้าสัญญาณนี้แอคตีฟอยู่นานเกินไป จะทำให้เกิดขบวนการ DMA ขึ้นมากกว่า 1 ขบวนการได้สำหรับวงจรที่ขอ DMA โดยทั่วไปแล้วจะใช้สัญญาณตอบรับการขอ DMA หรือสัญญาณ DACK ของแชนแนลที่ขอ DMA นั้นในการรีเซ็ทสัญญาณ DRQ เช่นอุปกรณ์ภายนอกที่ทำการขอ DMA ผ่านทางแชนแนลที่ 1 (DRQ1) ก็จะคอยทำการตรวจสอบการตรวจรับในการขอ DMA จากสัญญาณ DACK ของแชนแนลที่ 1 (DACK1) เมื่อได้รับสัญญาณจาก DACK1 แล้วก็จะทำการรีเซ็ทสัญญาณ DRQ1 (เปลี่ยนจากลอจิก "1" เป็น "0")

DACK0-DACK3 และ DACK5-DACK7 (DMA ACKNOWLWDGE 0-3 & DMA ACKNOWLEDGE 5-7) สัญญาณทั้งเจ็ดเป็นเอาท์พุทที่แอคตีฟที่ลอจิก "0" ซึ่ง DMA CONTROLLER สร้างขึ้นเพื่อที่จะเป็นการแสดงให้วงจรภายนอกที่ขอ DMA ทราบว่าการขอ DMA นั้นได้รับการตอบสนองแล้ว และ DMA CONTROLLER จะเข้าสู่ขบวนการ DMA เพื่อที่ทำให้การส่งผ่านข้อมูลระหว่างอุปกรณ์ I/O ที่ขอ DMA กับหน่วยความจำเกิดขึ้นโดยตรง โดยสัญญาณ DACK นี้จะทำการแอคตีฟในแชนแนลไหนนั้นก็ขึ้นอยู่กับว่าขบวนการ DMA ที่เกิดขึ้นนั้น เป็นการตอบสนองต่อการขอ DMA ในแชนแนลใด เช่นถ้าขบวนการ DMA ที่เกิดขึ้นนั้นเป็นการตอบสนองต่อการขอ DMA ในแชนแนลที่ 2 (DRQ2) สัญญาณ DACK2 ก็จะแอคตีฟ เป็นต้น

AEN (ADDRESS ENABLE) สัญญาณนี้เป็นเอาท์พุทที่ใช้ในการแสดงว่าบัสไซเคิลที่เกิดขึ้นในช่วงเวลาที่สัญญาณ AEN แอคตีฟ (ลอจิก "1") นั้นเป็นบัสไซเคิลของขบวนการ DMA

สำหรับเมนบอร์ดของ IBM/PC นั้นจะใช้สัญญาณในการดีสเอเบิล (DISABLE) BUS CONTROLLER และจะใช้ดีสเอเบิลพอร์ท I/O ต่างๆ ที่ไม่เกี่ยวข้องกับขบวนการ DMA ที่เกิดขึ้นนี้

ที่จำเป็นต้องทำเช่นนี้ก็เพราะในระหว่างขบวนการ DMA นั้น DMA CONTROLLER จะทำการส่งแอดเดรสของหน่วยความจำออกมาบนบัสแอดเดรส และจะทำให้สัญญาณ IOR หรือ IOW แอคตีฟด้วย ดังนั้นถ้าไม่ทำการดิสเอเบิลพอร์ท I/O ที่ไม่เกี่ยวข้องไว้ ก็อาจจะทำให้พอร์ท I/O ที่มีแอดเดรสตรงกับค่าแอดเดรสบนบัสแอดเดรส (ซึ่งเป็นแอดเดรสของหน่วยความจำ) นั้น ทำการอ่านหรือส่งข้อมูลออกมาบนบัสข้อมูลทำให้เกิดความผิดพลาดขึ้นได้

SBHE (SYSTEM BUS HIGH ENABLE) สัญญาณนี้จะเป็นการแสดงให้ทราบว่ามีการส่งผ่านของข้อมูลในส่วนของไบต์สูง (UPPER BYTE) ลงไปยังบัสข้อมูล (SD8-SD15) อุปกรณ์ประเภท 16 บิทจะใช้สัญญาณ SBHE ในการกำหนดสถานะของบัฟเฟอร์ สำหรับบัสข้อมูล SD8-SD15

MASTER สัญญาณดังกล่าวนี้จะถูกใช้กับสัญญาณ DRQ เพื่อที่จะทำให้การควบคุมของระบบให้ดีขึ้นกว่าเดิม ตัวโปรเซสเซอร์และ DMA CONTROLLER บน I/O CHANNEL ทำการจ่ายสัญญาณ DRQ เข้าไปยัง DMA CHANNEL ในโหมด CASCADE และทำการรับสัญญาณ "DACK" โดยที่ว่าถ้าหากมีการรับสัญญาณ "DACK" แล้วไมโครโปรเซสเซอร์สำหรับ I/O จะทำการดึงสัญญาณ "MASTER" ให้ลงต่ำซึ่งจะเป็นการยอมรับว่ามันจะทำการควบคุมระบบของแอดเดรสข้อมูลและการควบคุม (สภาพดังกล่าวนี้รู้จักกันดีว่าเป็น "TRI STATE") หลังจากที่สัญญาณ MASTER เป็น "LOW" ไมโครโปรเซสเซอร์ทางด้าน I/O จะทำการคอยในคาบเวลาเท่ากับ CLOCK ของระบบหนึ่งลูก ก่อนที่จะทำการส่งคำสั่งในการอ่านหรือเขียน ถ้าหากสัญญาณดังกล่าวนี้ค้างอยู่ที่ "LOW" มากกว่า 15 uS หน่วยความจำของระบบอาจที่จะสูญเสียอันเนื่องมาจากการขาดซึ่งการรีเฟรช

MEM CS16 (-MEM 16 CHIP SELECT) สัญญาณดังกล่าวนี้เป็นการแสดงว่ามีการส่งผ่านข้อมูล 16 บิทในช่างไซเคิลของหน่วยความจำพร้อมกับ 1 WAIT STATE สัญญาณ "-MEM CS16"ควรที่จะทำการขับโดยตัวขับที่เป็นแบบ OPEN CONTROLLER หรือ TRI-STATE สามารถทำการจ่ายกระแสได้มากถึง 20 mA

I/O CS16 (I/O 16 BIT CHIP SELECT) สัญญาณดังกล่าวนี้เป็นการแสดงว่ามีการส่งผ่านข้อมูล 16 บิทในช่วงไซเคิลของ I/O พร้อมกับ 1 wait state สัญญาณ "I/O CS16" จะทำการแอคตีฟที่ 'LOW" และจะถูกทำการขับโดยตัวขับที่เป็นแบบ OPEN COLOLETTOR หรือ TRI-STATE ที่สามารถทำการจ่ายกระแสได้มากถึง 20 mA

T/C (TERMINAL COUNT) สัญญาณนี้จะถูกสร้างขึ้นจากการนำเอาสัญญาณเอาท์พุทที่ขา EOP ของ DMA CONTROLLER มากลับลอจิก (โดยใช้เกท INVERTER) ทำให้สัญญาณ T/C นี้แอทตีฟที่ลอจิก "1"

สำหรับสัญญาณนี้จะแอคตีฟเมื่อจำนวนไบท์ในการส่งผ่านข้อมูลของขบวนการ DMA ในแชนแนลใดแชนแนลหนึ่ง ครบตามจำนวนที่กำหนดไว้ โดยทั่วไปแล้วสัญญาณที่จะถูกใช้ในการสิ้นสุดขบวนการ DMA ที่ทำการส่งผ่านข้อมูลเป็นบล็อค เนื่องจากสัญญาณนี้เป็นแอคตีฟโดยไม่แสดงว่าเป็นสัญญาณของแชนแนลใด ดังนั้นจึงต้องทำการนำสัญญาณ T/C นี้ผ่านเกท INVERTER แล้วนำไป OR กับสัญญาณ DACK เพื่อให้สามารถทราบได้ว่า สัญญาณ T/C ที่เกิดขึ้นนั้นเป็นสัญญาณของแชนแนลใด สำหรับในแชนแนลที่ 0 นั้นสัญญาณ T/C จะแอคตีฟในช่วงเวลาที่คงที่

บัสของแหล่งจ่ายไฟของระบบ

+5 Vdc (ขา B3 และ B29) ขาทั้งสองนี้ต่อกับแหล่งจ่ายไฟ DC +5V ของระบบ โดยจะมีค่าความเที่ยงตรง (Regulated) +- 5% คืออยู่ในช่วง +4.75 ถึง +5.25 Vdc

+12 Vdc (ขา B9) ขานี้จะต่อกับแหล่งจ่ายไฟ DC +12V ของระบบค่าความเที่ยงตรง (Regulated) +- 5% คืออยู่ในช่วง +11.4 ถึง +12.6 Vdc

-5 Vdc (ขา B5) ขานี้จะต่อกับแหล่งจ่ายไฟ DC -5V ของระบบค่าความเที่ยงตรง (Regulated) +- 10% คืออยู่ในช่วง + 5.5 ถึง -4.5 Vdc

-12 Vdc (ขา B7) ขานี้จะต่อกับแหล่งจ่ายไฟ DC -12V ของระบบค่าความเที่ยงตรง (Regulated) +- 10% คืออยู่ในช่วง -13.2 ถึง -10.8 Vdc

GND (ขา B1, B10 และ B31) ขาทั้งสามนี้จะต่อเข้ากับกราวด์ (Ground) ของระบบ

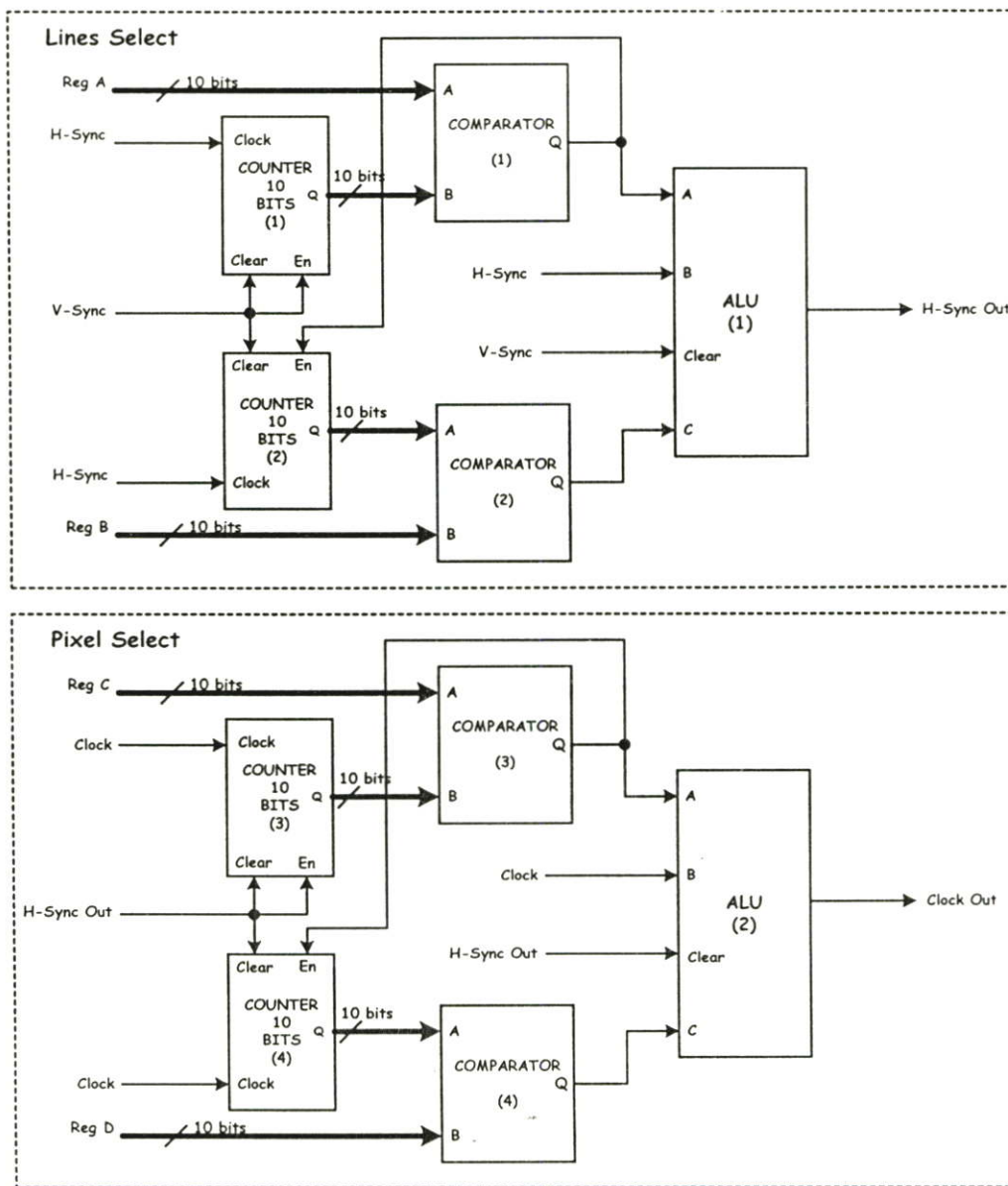
จากข้อกำหนดของเครื่องคอมพิวเตอร์ PC/AT ได้กำหนดแอดเดรสของพอร์ตที่สามารถเลือกตำแหน่งที่จะถูกดีโค้ด ให้อยู่ในช่วงแอดเดรสระหว่าง 300H-31FH หากทำการพิจารณาถึงตำแหน่งแอดเดรสที่จะนำเอามาใช้ในการดีโค้ดก็จะได้เป็นดังตารางที่ 4.3

ตารางที่ 4.4 รหัสแสดงแอดเดรสของพอร์ตสำหรับการ์ดอินเตอร์เฟส

I/O DECODE	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
300H	1	1	0	0	0	0	0	0	0	0
31FH	1	1	0	0	0	1	1	1	1	1
DECODE RANGE	1	1	0	0	0	X	X	X	X	X

4.4 สถาปัตยกรรมของวงจรกำหนดตำแหน่งและขนาดของข้อมูลภาพ [14]

ในส่วนของสถาปัตยกรรมการกำหนดตำแหน่งและขนาดของข้อมูลภาพ จะเป็นส่วนของการเลือกเส้นสแกน และเลือกพิกเซลที่จะทำการจัดเก็บสามารถแสดงเป็นบล็อกไดอะแกรมได้ดังรูปที่ 4.13

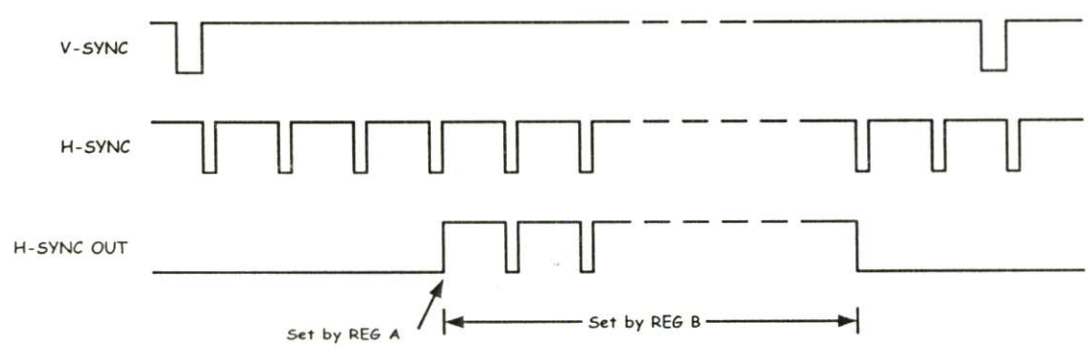


รูปที่ 4.13 บล็อกไดอะแกรมของวงจรกำหนดตำแหน่งและขนาดของข้อมูลภาพ

จากรูปที่ 4.13 ภายในบล็อกไดอะแกรมจะเห็นว่า Reg A - Reg D จะถูกนำมาใช้งานในส่วนนี้ และสามารถแยกการทำงานได้เป็นสองส่วนคือ

4.4.1 Lines Select

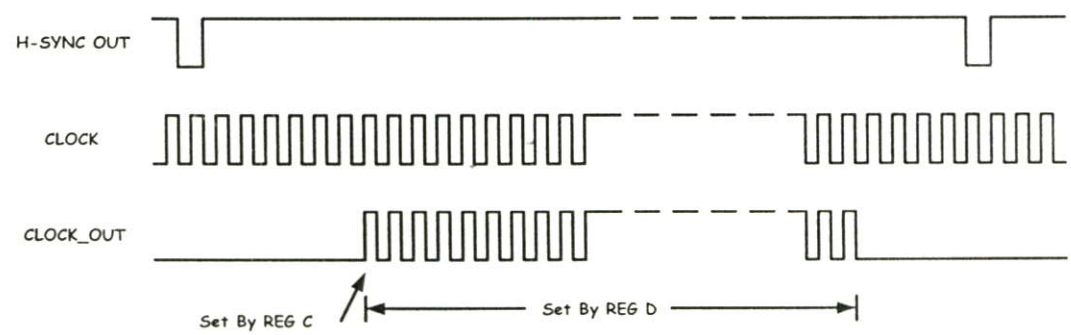
จะทำหน้าที่ในการเลือกเส้นสแกนที่จะทำการจัดเก็บข้อมูลภาพ ภายในบล็อกไดอะแกรม Line Select จะประกอบด้วย Counter 10 bits 2 ชุด Counter ชุดที่ 1 จะทำหน้าที่นับสัญญาณ H- Sync ที่เข้ามา และทำการเปรียบเทียบกับค่าใน Reg A และเมื่อ Counter ชุดที่ 1 นับมาจนมีค่าเท่ากับค่าใน Reg A จะทำให้เอาท์พุทของ Comparator1 เป็นลอจิก "0" ซึ่งขณะนี้แสดงว่า Counter นับมาถึงเส้นสแกนเส้นแรกที่จะทำการจัดเก็บแล้วเอาท์พุทของ Comparator1 จะส่งข้อมูลให้กับ ALU1 และแอนนาเบิ้ลให้ Counter ชุดที่ 2 ทำการนับสัญญาณ H-Sync สัญญาณ H-Sync ที่เข้ามาให้ Counter ชุดที่ 2 ทำการนับนั้น ก็คือสัญญาณ H-Sync ที่จะทำการจัดเก็บลงในหน่วยความจำนั้นเอง เอาท์พุทของ Counter ชุดที่ 2 จะถูกส่งมาทำการเปรียบเทียบกับค่าใน Reg B โดย Comparator2 เมื่อ Counter ชุดที่ 2 นับมาจนกระทั่งมีค่าเท่ากับ Reg B จะทำให้เอาท์พุทของ Comparator2 มีสภาวะเป็น ลอจิก "0" ซึ่งจะส่งค่าเอาท์พุทนี้ไปทำการประมวลผลร่วมกับเอาท์พุทของ Comparator ชุดที่ 1 ,สัญญาณ H-Sync และสัญญาณ V-Sync ที่ ALU1 ในส่วนของ ALU1 นั้นจะทำหน้าที่ในการตัดสินใจที่จะปล่อยสัญญาณ H-Sync ออกมา เมื่อสถานะของอินพุทเป็นไปตามรูปแบบที่กำหนดไว้ เอาท์พุทของ ALU1 จะเป็นสัญญาณ H-Sync ที่จะถูกส่งเข้าสู่ส่วนของ Pixel Select ต่อไป การทำงานของ ALU1 จะทำหน้าที่คล้ายกับสวิตช์เปิด-ปิด ช่องทางให้สัญญาณ H-Sync ปรากฏที่ H-Sync Out โดยจะใช้สัญญาณจาก Comparator1 (เมื่อมีค่าเป็นลอจิก "0") เป็นตัวควบคุมให้เปิดสวิตช์ จะทำให้สัญญาณ H-Sync ปรากฏที่ H-Sync Out และใช้สัญญาณจาก Comparator2 (เมื่อมีค่าเป็นลอจิก "0") เป็นตัวปิดสวิตช์ทำให้ไม่มีสัญญาณปรากฏที่ H-Sync Out สามารถอธิบายการทำงานของภาค Line Select เป็นไทมิ่งไดอะแกรมได้ดังรูปที่ 4.14



รูปที่ 4.14 ไทมิ่งไดอะแกรมในส่วนของ Line Select

4.4.2 Pixel Select

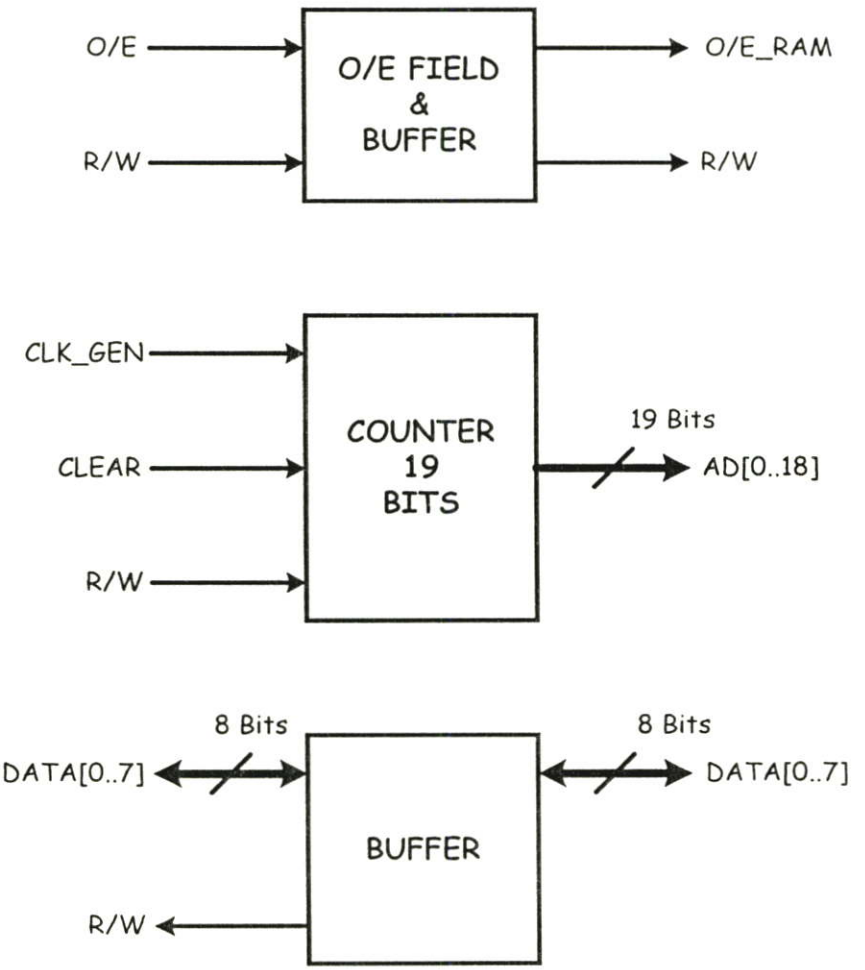
จะทำหน้าที่ในการเลือกพิกเซลในแต่ละเส้นสแกนที่จะทำการจัดเก็บลงสู่หน่วยความจำภายในบล็อกไดอะแกรมของส่วน Pixel Select จะประกอบด้วย Counter 10 bits จำนวน 2 ตัว สัญญาณ H-Sync Out ที่ได้จาก ภาค Line Select จะเป็นตัวแอนนาเบิลให้ Counter ชุดที่ 3 ทำงานซึ่งจะทำการนับสัญญาณ Clock ที่เข้ามา ข้อมูลที่ได้จากการนับจะถูกส่งไปทำการเปรียบเทียบกับค่าใน Reg C โดย Comparator3 เมื่อตัว Counter ทำการนับจนกระทั่งมีค่าเท่ากับค่าใน Reg C จะทำให้เอาท์พุทของ Comparator3 มีค่าเป็นลอจิก "0" ซึ่งจะถูกส่งไปให้ ALU2 และส่งไปแอนนาเบิลให้ Counter ชุดที่ 4 ทำงาน ในขณะที่เอาท์พุทของ Comparator3 มีค่าเป็นลอจิก "0" แสดงว่าเคาเตอร์ได้ทำการนับมาจนถึงพิกเซลแรกที่จะทำการจัดเก็บข้อมูลภาพแล้ว และสั่งงานให้ Counter ชุดที่ 4 เริ่มทำการนับสัญญาณ Clock เอาท์พุทของ Counter ชุดที่ 4 จะถูกส่งไปเข้าที่ Comparator4 เพื่อทำการเปรียบเทียบกับค่าใน Reg D เมื่ออินพุททั้งสองมีค่าที่เท่ากัน แสดงว่าขณะนี้เคาเตอร์ได้ทำการนับพิกเซลที่จะทำการจัดเก็บในหนึ่งเส้นสแกนครบตามจำนวนที่กำหนดแล้ว จะทำให้เอาท์พุทของ Comparator4 มีค่าเป็นลอจิก "0" ซึ่งจะถูกส่งเข้าไปทำการประมวลผลที่ ALU2 และ ALU2 จะทำการประมวลผลสัญญาณจาก H-Sync Out , Comparator3 , Comparator4 และสัญญาณ Clock ซึ่ง ALU2 จะทำการตัดสินใจที่จะให้มีสัญญาณ Clock_Out ออกไป การทำงานของ ALU2 จะทำหน้าที่คล้ายกับสวิตช์เปิด-ปิด ช่องทางให้สัญญาณ Clock ปรากฏที่ Clock_Out โดยจะใช้สัญญาณจาก Comparator3 (เมื่อมีค่าเป็นลอจิก "0") เป็นตัวควบคุมให้เปิดสวิตช์ จะทำให้สัญญาณ Clock ปรากฏที่ Clock_Out และใช้สัญญาณจาก Comparator4 (เมื่อมีค่าเป็นลอจิก "0") เป็นตัวปิดสวิตช์ทำให้ไม่มีสัญญาณปรากฏที่ Clock_Out ในส่วนของสัญญาณ Clock_Out จะใช้เป็นสัญญาณกระตุ้นวงจรแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลและใช้สำหรับนับแอดแดรสของหน่วยความจำที่จะใช้เก็บข้อมูลภาพ สามารถแสดงการทำงานของ Pixel Select เป็นไทมิ่งไดอะแกรม ได้ดังรูปที่ 4.15



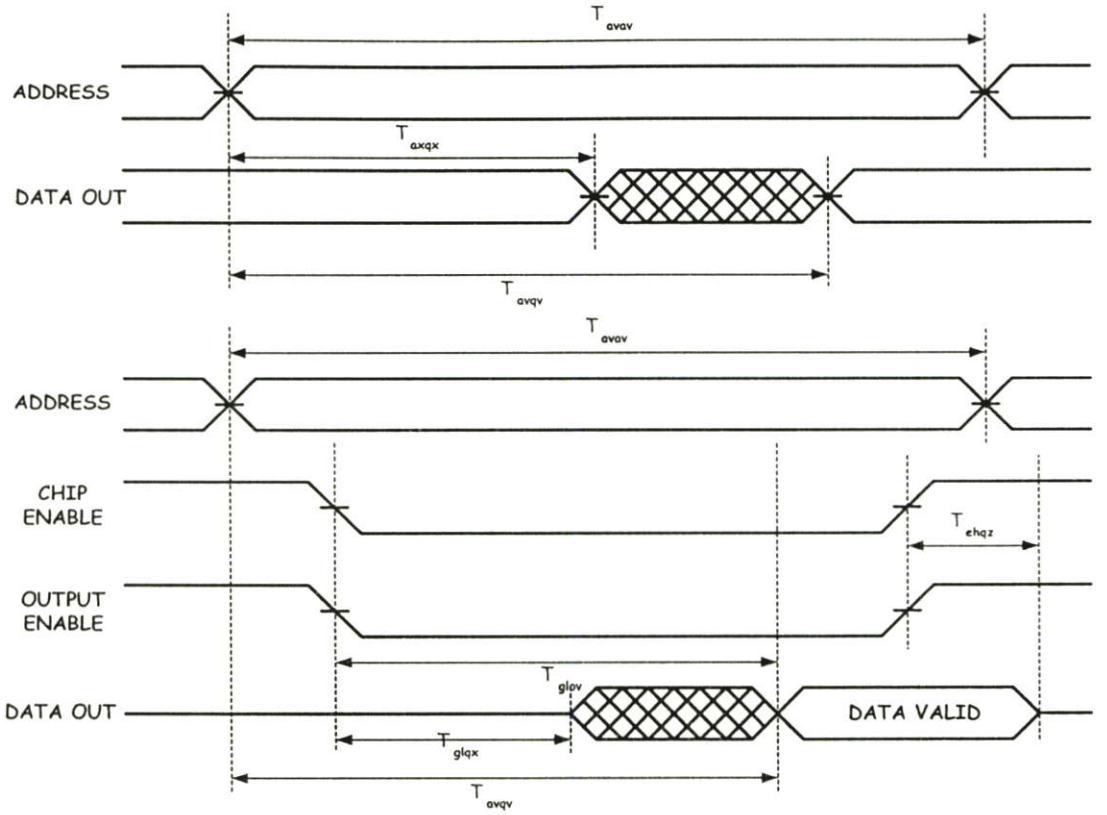
รูปที่ 4.15 ไทมิ่งไดอะแกรมในส่วนของ Pixel Select

4.5 สถาปัตยกรรมของชุดกำเนิดสัญญาณควบคุมหน่วยความจำแบบ SRAM

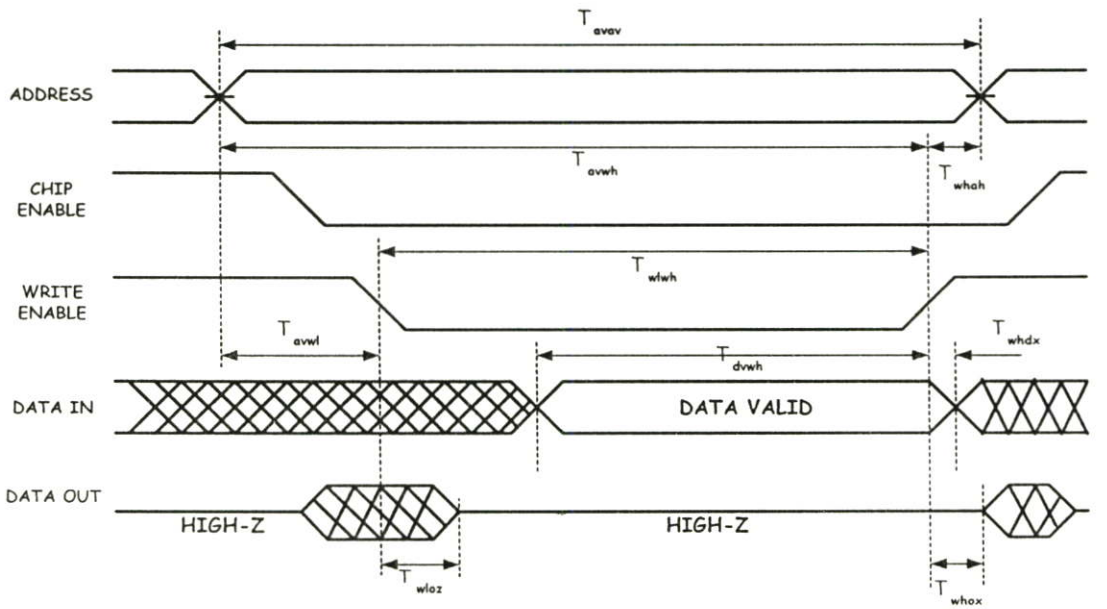
ในส่วนของการกำเนิดสัญญาณเพื่อควบคุมการทำงานของหน่วยความจำแบบ SRAM นั้น จะเป็นการควบคุมการเขียน การอ่าน ข้อมูลของหน่วยความจำ การเลือกหน่วยความจำว่าจะใช้หน่วยความจำฟิลด์คู่หรือฟิลด์คี่ และการควบคุมการอ้างอิงแอดเดรสให้กับหน่วยความจำ ซึ่งสามารถอธิบายเป็นบล็อกไดอะแกรมได้ดังรูปที่ 4.16



รูปที่ 4.16 บล็อกไดอะแกรมสำหรับการควบคุมหน่วยความจำแบบ SRAM



รูปที่ 4.17a ไทมิ่งไดอะแกรมสำหรับขั้นตอนในการอ่านข้อมูลของ SRAM ทั่วๆ ไป



รูปที่ 4.17b ไทมิ่งไดอะแกรมสำหรับขั้นตอนในการเขียนข้อมูลของ SRAM ทั่วๆ ไป

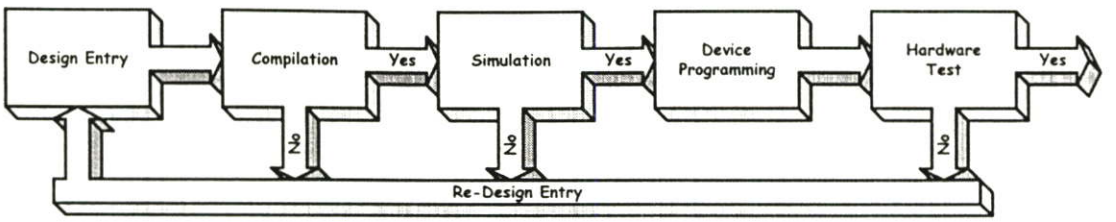
ตารางที่ 4.4 คาบเวลาที่ใช้สำหรับการอ่านและเขียนข้อมูลของ SRAM เบอร์ K6R4008C1C

Symbol	Parameter	Min	Max	Unit
Tavav	Read or Write Cycle Time	12	-	ns
Taxqx	Output Hold from Address	3	-	ns
Tavqv	Address Access Time	-	12	ns
Tehqz	Chip Disable to High-Z Output	0	6	ns
Tglov	Output Enable to Low-Z Output	0	-	ns
Tavwh	Address Valid to End of	8	-	ns
Twhah	Write Recovery Time	0	-	ns
Tavwl	Output Disable to High-Z Output	0	6	ns
Twlwh	Chip Select to End of Write	8	-	ns
Tdvwh	Data to Write Time Overlap	6	-	ns

ในการออกแบบชุดควบคุมหน่วยความจำนั้น สัญญาณที่ใช้ในการควบคุม ขั้นตอนในการอ่านและการเขียนข้อมูล จะต้องมีความคล้องจองกันกับไทมิ่งไดอะแกรมของ SRAM ที่ใช้งานกันทั่วๆ ไป ซึ่งมีลักษณะไทมิ่งไดอะแกรมดังรูปที่ 4.17a , รูปที่ 4.17b ในวิทยานิพนธ์นี้จะใช้หน่วยความจำแบบ SRAM เบอร์ K6R4008C1C-JC12 สามารถแสดงคาบเวลาที่ใช้สำหรับการอ่านและเขียนข้อมูลของ SRAM เบอร์ K6R4008C1C-JC12 ได้ดังตารางที่ 4.4

4.6 ขั้นตอนการออกแบบระบบบนชิฟเอฟพีจีเอ [7]

สำหรับขั้นตอนการออกแบบระบบที่ได้กล่าวมาแล้วทั้งหมดบนชิฟเอฟพีจีเอนั้น เมื่อได้กำหนดแนวคิดหรือโครงสร้างของชุดควบคุมการทำงานต่างๆ ดังหัวข้อที่ 4.2 ถึง 4.5 ที่ผ่านมานั้น หลังจากนั้นจะเป็นส่วนของการอิมพลีเมนต์ลงบนชิฟเอฟพีจีเอ ในส่วนนี้จะเป็นการใช้โปรแกรมช่วยในการออกแบบซึ่งสามารถแสดงขั้นตอนการออกแบบระบบต่างๆบนชิฟเอฟพีจีเอได้ดังรูปที่ 4.18 และสามารถอธิบายขั้นตอนการออกแบบในแต่ละขั้นตอนได้ดังนี้



รูปที่ 4.18 ขั้นตอนการออกแบบบนชิฟเอฟพีจีเอ

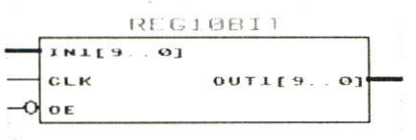
4.6.1 Design Entry

เป็นขั้นตอนของการออกแบบของแต่ละส่วน ในการออกแบบจะพยายามกระจายการออกแบบจากโครงสร้างรวมทั้งหมดในรูปที่ 4.9 ออกเป็นโครงสร้างย่อยๆ หลังจากนั้นจะใช้ภาษาวีเอชดีแอลเป็นตัวอธิบายถึงลักษณะพฤติกรรมของแต่ละโครงสร้างย่อยดังรูปที่ 4.19a หลังจากนั้นใช้จากเขียนแผนผังวงจร (Schematic) ในการเชื่อมต่อโครงสร้างย่อยๆเข้าด้วยกันเป็นโครงสร้างที่ใหญ่ขึ้นดังรูปที่ 4.19b

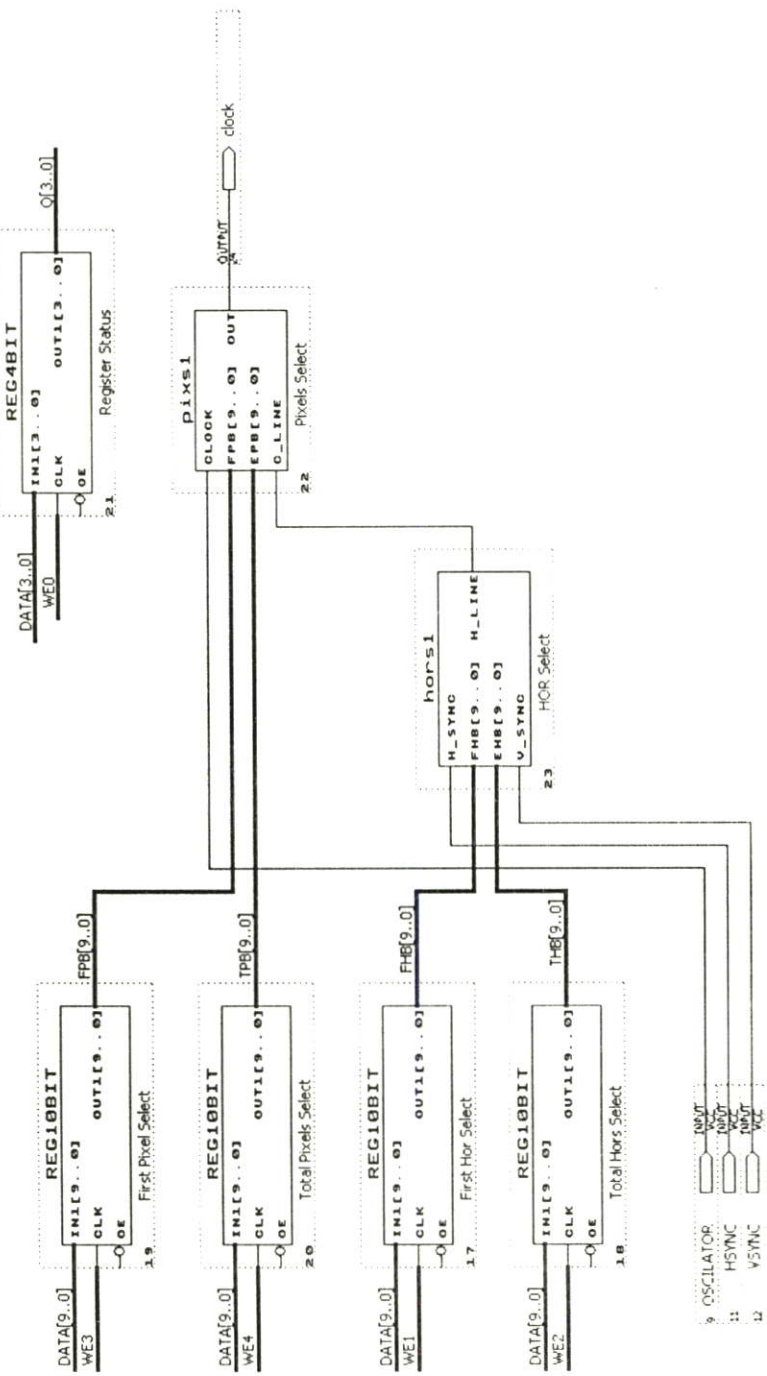
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;

entity Reg10bit is
port (clk : in std_logic; -- Input for Clock
        oe : in std_logic; -- Input for Enable
        in1 : in std_logic_vector(9 downto 0);-- Data In
       out1 : out std_logic_vector(9 downto 0)--Data Out
       );
end Reg10bit;

architecture RTL of Reg10bit is
begin
  process(clk)
  begin
    if (clk'EVENT) and (clk = '1') then --Rise Edge Detect
        if oe = '0' then
          out1 <= in1; --if oe = '0' Set out1 = in1
        else
          out1 <= "ZZZZZZZZZZ"; -- else out1 = Z
        end if;
    end if;
  end process;
end RTL;
```



รูปที่ 4.19a การใช้ภาษาวีเอชดีแอลอธิบายลักษณะพฤติกรรมของรีจิสเตอร์ขนาด 10 บิทและโมดูลของรีจิสเตอร์ 10 บิท



รูปที่ 4.19b การเชื่อมต่อโมดูลต่างๆเข้าด้วยกันเป็นวงจรกำหนดตำแหน่งและขนาดของภาพ

4.6.2 Compilation

หลังจากทำในขั้นตอนออกแบบเสร็จแล้ว ก็จะทำการตรวจสอบความถูกต้อง โดยการคอมไพล์ หากผลการคอมไพล์มีข้อผิดพลาด (error) เกิดขึ้นจะต้องกลับไปแก้ไขในขั้นตอนของ Design Entry ใหม่

4.6.3 Simulation

หากผลการคอมไพล์ไม่มีข้อผิดพลาด (error) ขั้นตอนต่อไปก็คือการจำลองการทำงานของวงจรที่ได้ออกแบบมา ในการจำลองการทำงานจะต้องกำหนดลักษณะของสัญญาณอินพุทที่เข้ามาในวงจรและช่วงเวลาของการจำลองการทำงาน ซึ่งในการจำลองการทำงานจะต้องครอบคลุมเหตุการณ์ทุกเหตุการณ์ที่อาจจะเกิดขึ้นได้ เพื่อดูการตอบสนองของวงจรต่อเหตุการณ์ต่างๆว่าเป็นเช่นไร หากผลการจำลองการทำงานไม่เป็นที่พอใจหรือไม่ถูกต้องก็จะต้องกลับไปทำการแก้ไขในขั้นตอนของ Design Entry ใหม่ หากไม่มีข้อผิดพลาดหรือพอใจในผลการจำลองการทำงานก็ไปทำขั้นตอนต่อไป

4.6.4 Device Programmming

ขั้นตอนนี้เป็นขั้นตอนของการโปรแกรมวงจรที่ได้ออกแบบมาลงในชิฟเอฟพีจีเอเพื่อที่จะนำชิฟไปทดสอบต่อไป

4.6.5 Hardware Test

เป็นขั้นตอนสุดท้ายของการออกแบบจะเป็นการนำชิฟเอฟพีจีเอมาทดสอบการทำงานจริงกับฮาร์ดแวร์ภายนอกที่ได้ออกแบบไว้ว่าสามารถใช้งานได้จริงตามที่ได้ออกแบบไว้หรือไม่ หากผลการทดสอบไม่ถูกต้องก็จะต้องกลับไปทำการแก้ไขที่ Design Entry ใหม่

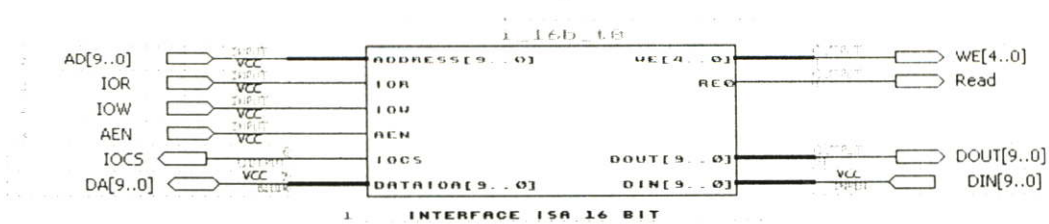
บทที่ 5
การทดลอง

ในบทนี้จะเป็นการกล่าวถึงส่วนของการทดสอบการทำงานของวงจรที่ได้ออกแบบมาว่าสามารถทำงานได้จริงตามที่ออกแบบไว้หรือไม่ ในการทดสอบจะแบ่งออกเป็นสองส่วนคือ ส่วนที่หนึ่งจะเป็นส่วนของการจำลองการทำงาน (Simulation) ของวงจรก่อนที่จะโปรแกรมลงชิฟเอฟพีจีเอเพื่อดูผลตอบสนองของวงจรต่ออินพุทต่างๆ ที่ได้กำหนดให้ ส่วนที่สองจะเป็นส่วนของการทดสอบการทำงานจริงของวงจรทั้งหมดที่ได้ออกแบบมาเพื่อให้ทำงานเป็นเครื่องแปลงสัญญาณภาพทางการแพทย์

5.1 ทดสอบโดยการจำลองการทำงาน (Simulation)

หลังจากได้ออกแบบวงจรโดยการอธิบายลักษณะพฤติกรรมของวงจร(Hardware Description Language) เป็นที่เรียบร้อยแล้ว ผู้ออกแบบจำเป็นจะต้องทำการจำลองการทำงานเพื่อดูผลการตอบสนองของวงจรต่างๆ ต่ออินพุทที่กำหนดให้ซึ่งหากผลลัพธ์ที่ได้ไม่ถูกต้องจะได้ทำการแก้ไขต่อไป ในส่วนของการออกแบบโดยการอธิบายลักษณะพฤติกรรมของวงจรจะทำการออกแบบบนคอมพิวเตอร์ส่วนบุคคล Pentium III 533 MHz หน่วยความจำแบบ SDRAM 64 MB ระบบปฏิบัติการ WINDOWS ME โดยใช้โปรแกรมในการออกแบบชื่อโปรแกรม MAX+plus II ซึ่งเป็น Synthesis Tools ที่ใช้สำหรับการออกแบบวงจรเพื่อทำงานบนชิฟเอฟพีจีเอของบริษัทอัลเทร่าจำกัด

ในส่วนนี้จะแสดงผลการทดสอบการทำงานของวงจรที่ได้ออกแบบด้วยวีเอชดีแอลที่สำคัญสองส่วนเท่านั้นคือในส่วนของวงจรอินเตอร์เฟสกับคอมพิวเตอร์ผ่านทางสล๊อต ISA แบบ 16 บิท และส่วนของวงจรเลือกเส้นสะแกนและพิกเซลที่จะทำการจัดเก็บ สามารถแสดงแบบจำลองของวงจรอินเตอร์เฟส ISA แบบ 16 บิทได้ดังรูปที่ 5.1



รูปที่ 5.1 แบบจำลองของวงจรอินเตอร์เฟส ISA แบบ 16 บิท

จากแบบจำลองในรูปที่ 5.1 จะมีอินพุทและเอาท์พุทดังต่อไปนี้

1.AD[9..0] เป็นสัญญาณทางอินพุทขนาด 10 บิต ใช้สำหรับระบุแอดแดรสของพอร์ท I/O ที่ต้องการจะติดต่อด้วย

2.IOR เป็นสัญญาณอินพุตขนาด 1 บิต เพื่อแสดงว่าบัสไซเคิลที่เกิดขึ้นนี้เป็นบัสไซเคิลของการอ่านข้อมูลจากพอร์ท I/O

3.IOW เป็นสัญญาณอินพุตขนาด 1 บิต เพื่อแสดงว่าบัสไซเคิลที่เกิดขึ้นนี้เป็นบัสไซเคิลของการเขียนข้อมูลสู่พอร์ท I/O

4.AEN เป็นสัญญาณอินพุทขนาด 1 บิต เพื่อแสดงว่าบัสไซเคิลที่เกิดขึ้นเป็นบัสไซเคิลสำหรับพอร์ท I/O

5.IOCS เป็นสัญญาณเอาท์พุทขนาด 1 บิต ใช้สำหรับแสดงว่ามีการส่งผ่านข้อมูล 16 บิต ในไซเคิลของ I/O พร้อมกับ 1 WAIT STATE

6.DA[9..0] เป็นสัญญาณแบบ Bi-directional ขนาด 10 บิต ใช้สำหรับส่งผ่านข้อมูลระหว่างคอมพิวเตอร์กับชิฟเอฟพีจีเอ

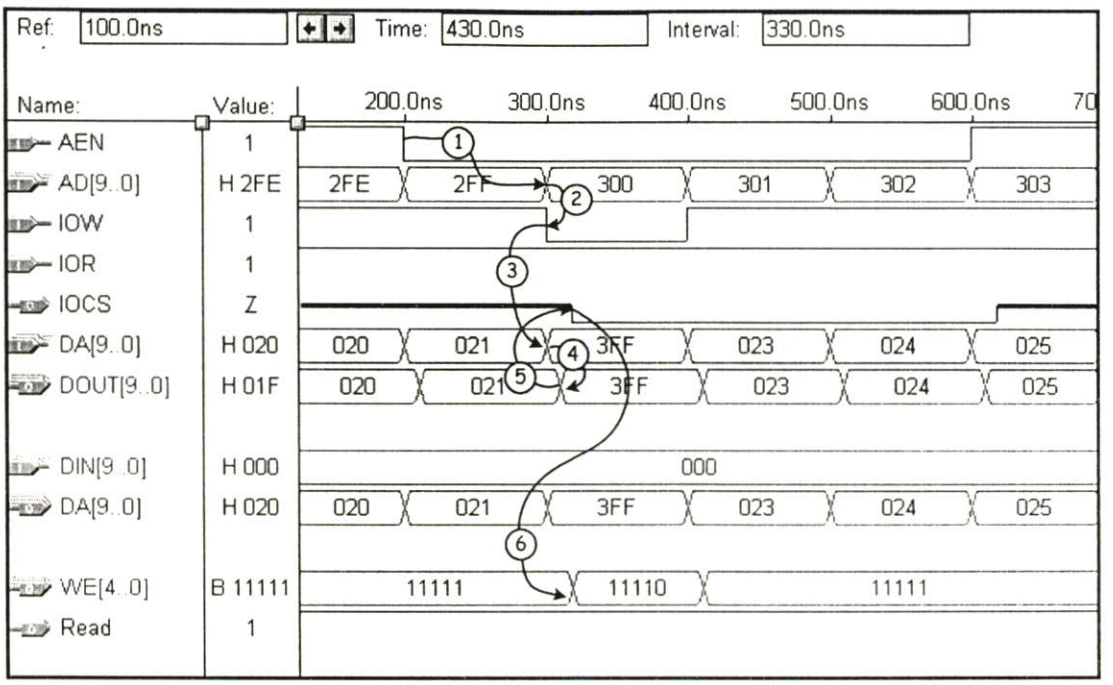
7.WE[4..0] เป็นสัญญาณเอาท์พุทขนาด 5 บิต ใช้สำหรับเลือกพอร์ทของเอาท์พุทที่ต้องการจะติดต่อด้วย

8.RE0 เป็นสัญญาณเอาท์พุทขนาด 1 บิต ใช้สำหรับเลือกพอร์ทของอินพุทที่ต้องการจะติดต่อด้วย

9.DOUT[9..0] เป็นสัญญาณขนาด 10 บิต ใช้สำหรับส่งผ่านข้อมูลให้กับอุปกรณ์ที่เป็นเอาท์พุท

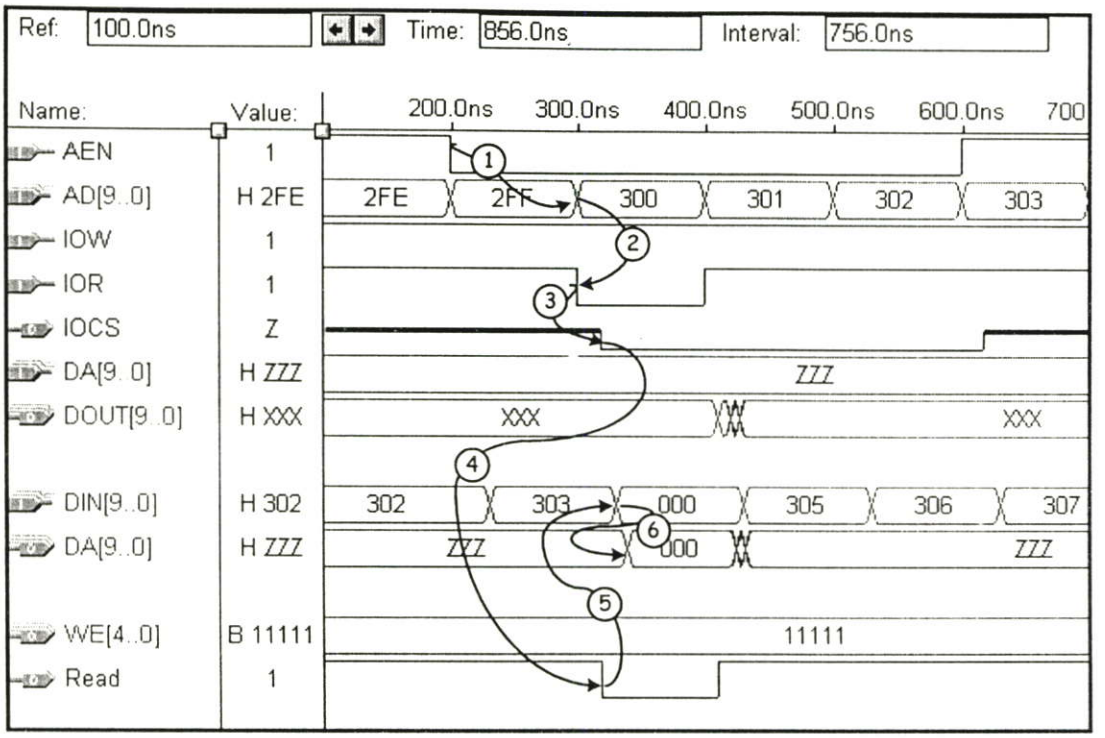
10.DIN[9..0] เป็นสัญญาณขนาด 10 บิต ใช้สำหรับอ่านข้อมูลจากอุปกรณ์ที่เป็นอินพุท

จะเห็นว่าโหมดการทำงานของวงจรอินเตอร์เฟส ISA แบบ 16 บิทจะมีอยู่ 2 โหมดการทำงานคือ โหมดของการเขียนข้อมูลจากบัสข้อมูลลงไปยังตัวอุปกรณ์ และโหมดของการอ่านข้อมูลจากตัวอุปกรณ์ส่งไปยังบัสข้อมูลของคอมพิวเตอร์ ซึ่งจะใช้สัญญาณจากขา IOR และ IOW เป็นสัญญาณควบคุมโหมดการเขียนหรืออ่านข้อมูล ในการจำลองการทำงานจะแสดงผลการจำลองการทำงานของทั้งสองโหมดดังกล่าว สามารถแสดงผลการจำลองการทำงานของโหมดการเขียนข้อมูลได้ดังรูปที่ 5.2 และสามารถแสดงผลการจำลองการทำงานของโหมดการอ่านข้อมูลได้ดังรูปที่ 5.3



รูปที่ 5.2 ผลการจำลองการทำงานของชุดอินเตอร์เฟส ISA แบบ 16 บิท สำหรับโหมดการเขียนข้อมูล

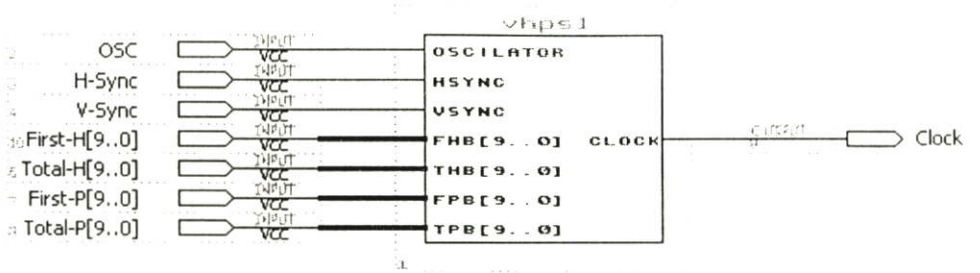
จากรูปที่ 5.2 เป็นผลจากการจำลองการทำงานของวงจรอินเตอร์เฟส ISA แบบ 16 บิท ในโหมดของการเขียนข้อมูล เครื่องหมายวงกลมที่มีตัวเลขกำกับอยู่นั้นจะเป็นขั้นตอนการทำงานของวงจรอินเตอร์เฟส โดยเริ่มต้นจาก ① เมื่อต้องการจะติดต่อกับฮาร์ดแวร์ในโหมดของการเขียนหรืออ่านข้อมูล สัญญาณ AEN จะแอคทีฟเป็นลอจิก "0" หลังจากนั้น ② สัญญาณแอดแดรสจะถูกส่งมาที่บัสแอดแดรส หลังจากนั้นในโหมดของการเขียนข้อมูล ③ สัญญาณ IOW จะแอคทีฟเป็นลอจิก "0" จากนั้น ④ ข้อมูลจากบัสข้อมูลจะถูกส่งมาที่ตัวอุปกรณ์ จากนั้น ⑤ สัญญาณ IOCS จะเปลี่ยนจากสถานะจาก High Impedance เป็นลอจิก "0" เพื่อเป็นการแสดงว่ามีการส่งผ่านข้อมูล 16 บิท ในช่วงไซเคิลของ I/O พร้อมกับ 1 WAIT STATE และ ⑥ เป็นสัญญาณที่จะเลือกตัวอุปกรณ์ให้รับข้อมูลที่ส่งมาจากบัสข้อมูล



รูปที่ 5.3 ผลการจำลองการทำงานของชุดอินเตอร์เฟส ISA แบบ 16 บิท สำหรับโหมดการอ่านข้อมูล

ส่วนรูปที่ 5.3 เป็นผลการจำลองการทำงานของวงจรอินเตอร์เฟส ISA แบบ 16 บิท ในโหมดของการอ่านข้อมูล เครื่องหมายวงกลมที่มีตัวเลขกำกับไว้เป็นการแสดงขั้นตอนการทำงานของวงจรอินเตอร์เฟส โดยเริ่มต้นจาก ① เมื่อต้องการติดต่อกับฮาร์ดแวร์สัญญาณ AEN จะแอคทีฟเป็นลอจิก "0" หลังจากนั้น ② ตำแหน่งของแอดแดรสที่ต้องการจะอ่านข้อมูลจะถูกส่งลงมาที่บัสแอดแดรส ส่วน ③ หากเป็นการติดต่อกับ I/O ในโหมดของการอ่านข้อมูล สัญญาณ IOR จะแอคทีฟเป็นลอจิก "0" หลังจากนั้น ④ สัญญาณ IOCS จะเปลี่ยนสถานะจาก High Impedance เป็นลอจิก "0" เพื่อเป็นการแสดงว่ามีการส่งผ่านข้อมูล 16 บิท ในช่วงไซเคิลของ I/O พร้อมกับ 1 WAIT STATE จากนั้น ⑤ สัญญาณ READ เป็นสัญญาณสำหรับเลือกตัวอุปกรณ์ที่ต้องการให้ส่งข้อมูลเข้ามาที่บัสข้อมูลของคอมพิวเตอร์ และ ⑥ ข้อมูลจาก DIN ถูกส่งเข้ามาที่ DA เป็นอันเสร็จสิ้นขบานการอ่านข้อมูลจากอุปกรณ์มายังบัสข้อมูลของคอมพิวเตอร์

ส่วนต่อไปจะเป็นการจำลองการทำงานของส่วนควบคุมที่สำคัญอีกส่วนหนึ่งก็คือ ส่วนของวงจรเลือกเส้นสะแกนและเลือกพิกเซลที่จะทำการจัดเก็บลงในหน่วยความจำ สามารถแสดงแบบจำลองของวงจรเลือกเส้นสะแกนและเลือกพิกเซลที่จะทำการจัดเก็บได้ดังรูปที่ 5.4



รูปที่ 5.4 แบบจำลองของวงจรเลือกเส้นสะแกนและเลือกพิกเซลที่จะทำการจัดเก็บ

จากแบบจำลองในรูปที่ 5.4 สัญญาณอินพุทที่จะนำเข้ามาประมวลผลร่วมกันจะมีอยู่ 7 สัญญาณอินพุทคือ

1.OSC เป็นสัญญาณอินพุทที่ได้รับมาจากโมดูลออสซิเลเตอร์ เพื่อเป็นตัวกำหนดความถี่ในการสุ่มสัญญาณภาพ

2.H-Sync เป็นสัญญาณ Horizontal Sync ที่ได้รับมาจากวงจรแยกสัญญาณซิงค์ (Sync Separator)

3.V-Sync เป็นสัญญาณ Vertical Sync จะรับสัญญาณมาจากวงจรแยกสัญญาณซิงค์ (Sync Separator) เช่นกัน

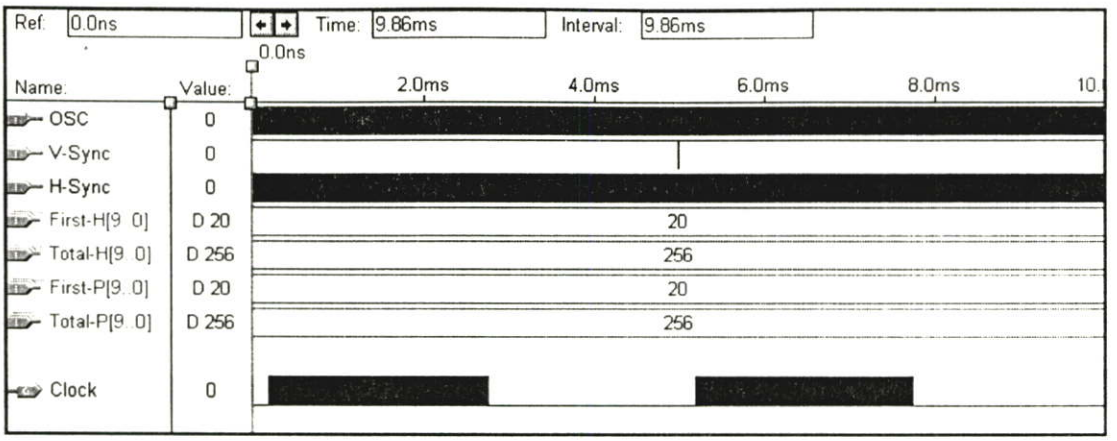
4.First-H[9..0] เป็นสัญญาณอินพุทขนาด 10 บิทใช้สำหรับกำหนดเส้นสะแกนเส้นแรกที่จะทำการจัดเก็บ

5.Total-H[9..0] เป็นสัญญาณอินพุทขนาด 10 บิทใช้สำหรับกำหนดจำนวนเส้นสะแกนทั้งหมดที่จะทำการจัดเก็บ

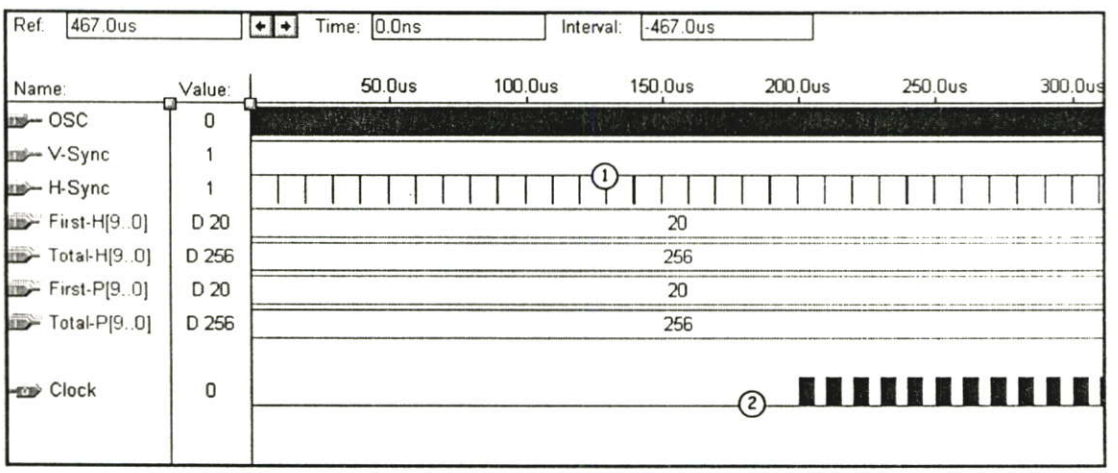
6.First-P[9..0] เป็นสัญญาณอินพุทขนาด 10 บิทใช้สำหรับกำหนดพิกเซลแรกที่จะทำการจัดเก็บ

7.Total-P[9..0] เป็นสัญญาณอินพุทขนาด 10 บิทใช้สำหรับกำหนดจำนวนพิกเซลทั้งหมดที่จะทำการจัดเก็บ

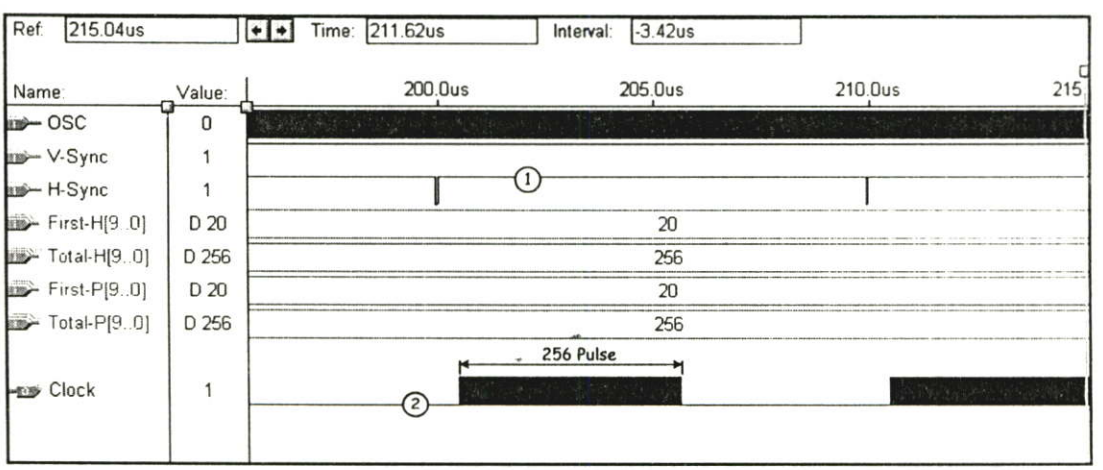
ในการจำลองการทำงานของวงจรเลือกเส้นสะแกนและพิกเซลที่จะทำการจัดเก็บจะต้องทำการกำหนดลักษณะของสัญญาณอินพุทต่างๆให้สอดคล้องกับลักษณะของสัญญาณที่ใช้งานจริง ซึ่งจะทำให้ผลลัพธ์ที่ได้จากการจำลองการทำงานมีลักษณะที่ใกล้เคียงกับการทำงานจริงของวงจรที่ได้ออกแบบมา สามารถแสดงผลการจำลองการทำงานของวงจรเลือกเส้นสะแกนและพิกเซลที่จะทำการจัดเก็บได้ดังรูปที่ 5.5



รูปที่ 5.5 ผลการจำลองการทำงานของวงจรเลือกเส้นสะแกนและพิกเซลที่จะทำการจัดเก็บ



รูปที่ 5.6 ขยายผลการจำลองการทำงานเพื่อดูความสัมพันธ์ของสัญญาณ H-Sync กับ Clock (1)

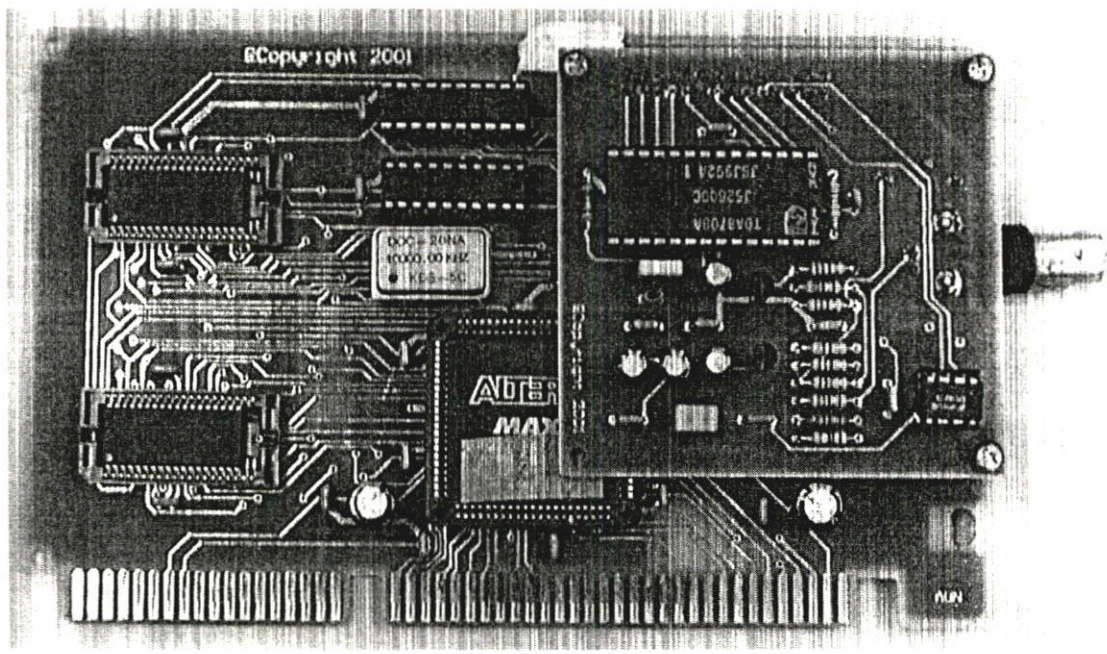


รูปที่ 5.7 ขยายผลการจำลองการทำงานเพื่อดูความสัมพันธ์ของสัญญาณ H-Sync กับ Clock (2)

จากรูปที่ 5.5 สามารถขยายผลเพื่อดูการทำงานให้ละเอียดมากขึ้นดังรูปที่ 5.6 และรูปที่ 5.7 จากรูปที่ 5.6 เป็นการขยายผลการจำลองการทำงานเพื่อดูลักษณะของสัญญาณ H-Sync ① ที่เข้ามาทางอินพุท ในส่วนของ ② เป็นลักษณะของสัญญาณเอาท์พุท Clock จะเห็นว่าสัญญาณ Clock จะปรากฏเป็นช่วงๆ ซึ่งในแต่ละช่วงก็คือสัญญาณคล็อกที่จะส่งไปเป็นตัวกระตุ้นวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอลให้ทำการแปลงสัญญาณคอมโพสิทวิดิโอที่เข้ามา ตำแหน่งและจำนวนของสัญญาณคล็อกที่ปรากฏขึ้นอยู่กับการกำหนดค่าอินพุทของ First-H , Total-H , First-P และ Total-P ส่วนรูปที่ 5.7 เป็นการขยายผลการจำลองการทำงานเพื่อดูลักษณะของสัญญาณคล็อกเปรียบเทียบกับเส้นสะแกน 1 เส้น

5.2 ทดสอบการทำงานจริงของวงจร

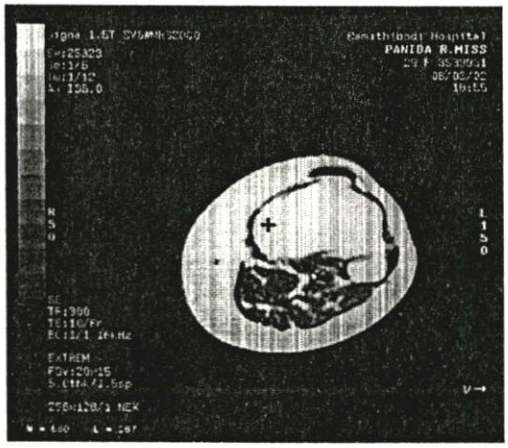
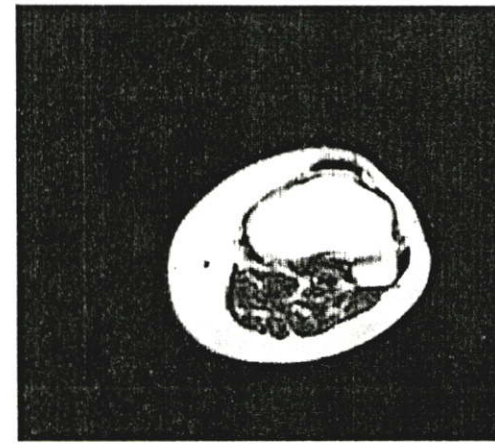
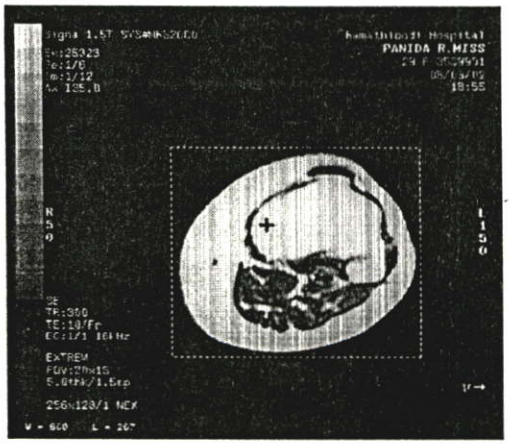
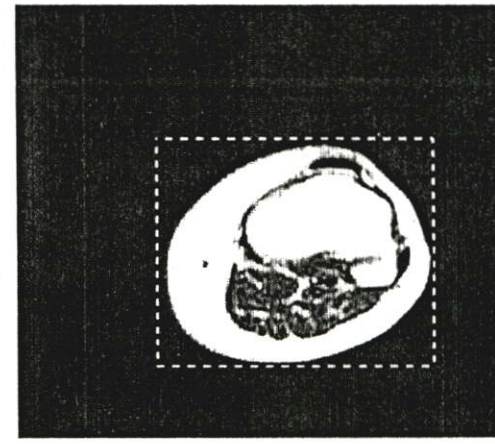
หลังจากทดสอบการทำงานของวงจรที่ได้ออกแบบด้วยภาษาวีเอชดีแอล โดยการจำลองการทำงานจนได้ผลการทำงานเป็นที่พอใจแล้ว หลังจากนั้นจะต้องทำการโปรแกรมข้อมูลทางลอจิกที่ได้ลงในชิฟเอฟพีจีเอ จากนั้นนำชิฟเอฟพีจีเอไปทดสอบกับวงจรอื่นๆ ที่ประกอบกันเป็นเครื่องแปลงสัญญาณภาพทางการแพทย์ สามารถแสดงแผงวงจรของเครื่องแปลงสัญญาณภาพทางการแพทย์ได้ดังรูปที่ 5.8



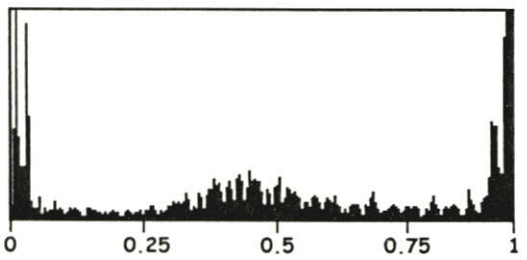
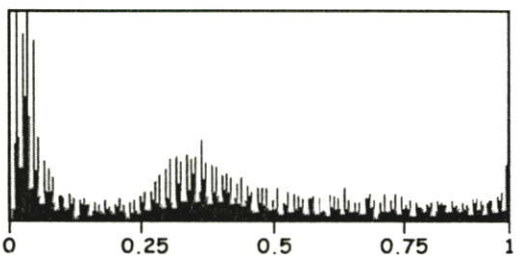
รูปที่ 5.8 แผงวงจรของเครื่องแปลงสัญญาณภาพทางการแพทย์โดยใช้เอฟพีจีเอ

ในการทดสอบการทำงานจริงของเครื่องแปลงสัญญาณภาพทางการแพทย์จะทำการทดลองที่โรงพยาบาลรามาธิบดี , โรงพยาบาลราชวิถี และโรงพยาบาลศิริราช เครื่องมือแพทย์ที่จะใช้ในการทดสอบจะเป็นเครื่อง MRI ข้อมูลภาพที่ได้จากเครื่อง MRI สามารถที่จะทำการคัดลอกภาพต้นฉบับมาทำการเปรียบเทียบกับผลการแปลงภาพได้ เครื่อง MRI ที่ใช้จะเป็นเครื่อง MRI ของบริษัท GENERAL ELECTRIC รุ่น SIGNA NA1.5 ในการเปรียบเทียบกับภาพต้นฉบับจะทำการเปรียบเทียบโดยฮีสโตแกรม (Histogram) จะเห็นว่าลักษณะของฮีสโตแกรมจะมีลักษณะที่คล้ายๆกัน สามารถแสดงผลการทดลองกับเครื่อง MRI ได้ดังตารางที่ 5.1 ถึงตารางที่ 5.6

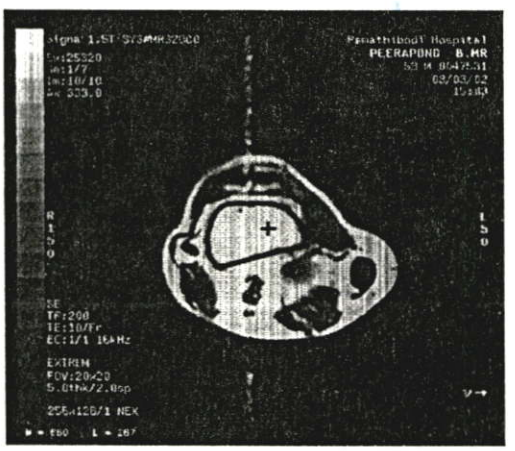
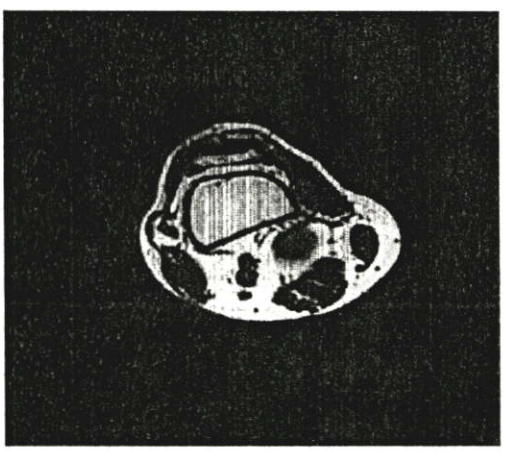
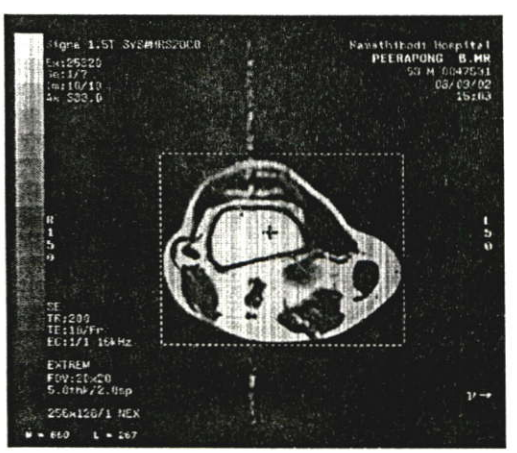
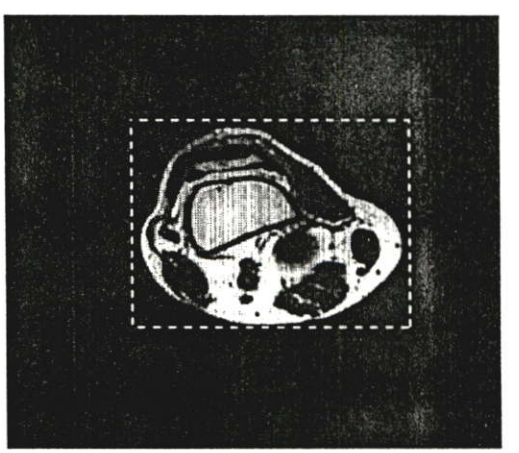
ตารางที่ 5.1 ผลการทดลอง

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1a ภาพจากเครื่องแปลงข้อมูลภาพ	**รูปที่** 2a ภาพต้นฉบับ
	
รูปที่ 1b เลือกบริเวณที่จะหาฮิสโตแกรม	**รูปที่** 2b เลือกบริเวณที่จะหาฮิสโตแกรม

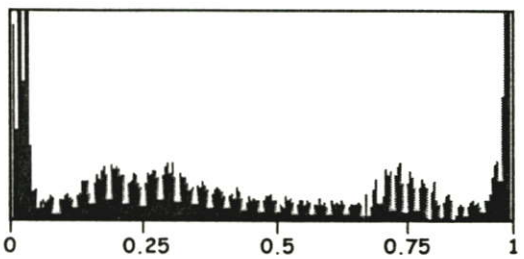
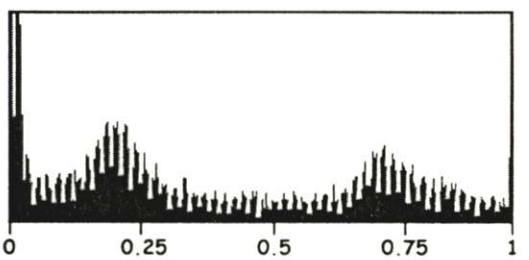
ตารางที่ 5.1 (ต่อ)

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1c ผลการหาฮีสโตแกรม	**รูปที่** 2c ผลการหาฮีสโตแกรม

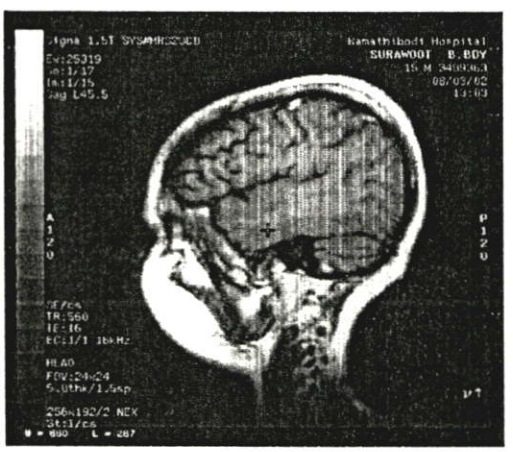
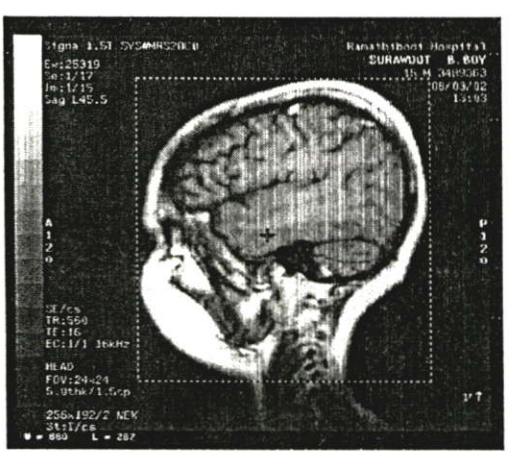
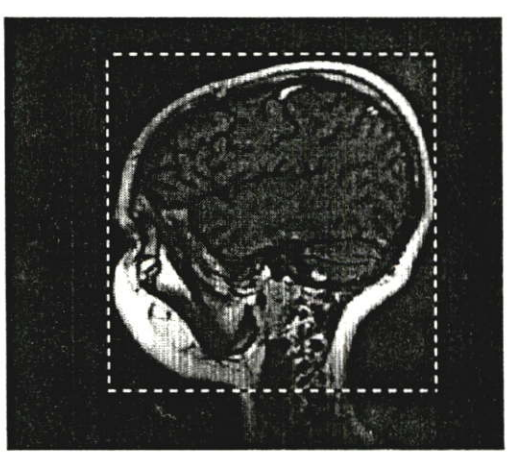
ตารางที่ 5.2 ผลการทดลอง

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1a ภาพจากเครื่องแปลงข้อมูลภาพ	**รูปที่** 2a ภาพต้นฉบับ
	
รูปที่ 1b เลือกบริเวณที่จะหาฮิสโตแกรม	**รูปที่** 2b เลือกบริเวณที่จะหาฮิสโตแกรม

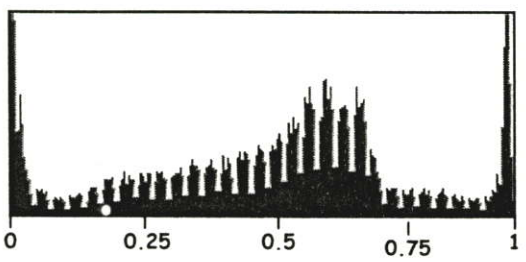
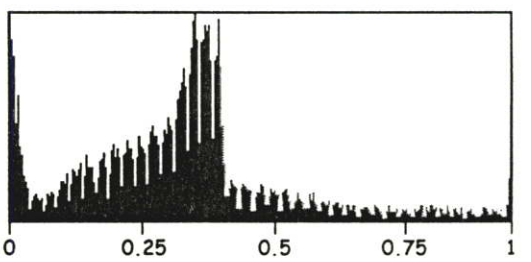
ตารางที่ 5.2 (ต่อ)

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1c ผลการหาฮีสโตแกรม	**รูปที่** 2c ผลการหาฮีสโตแกรม

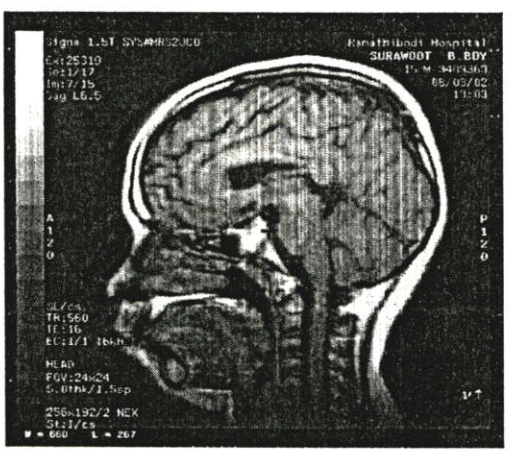
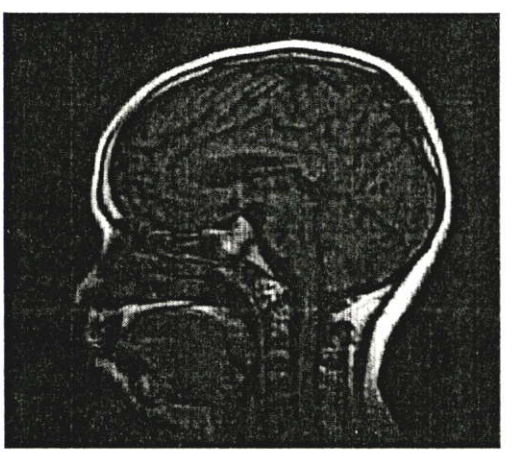
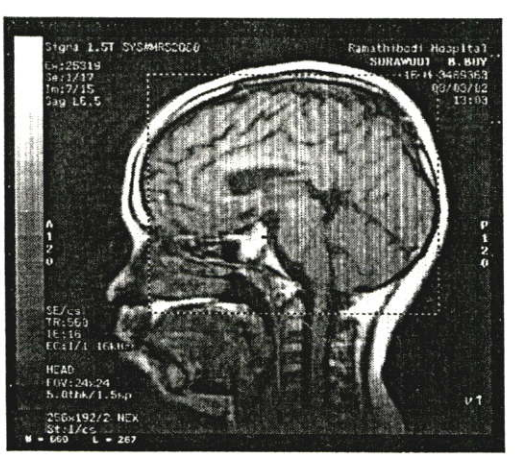
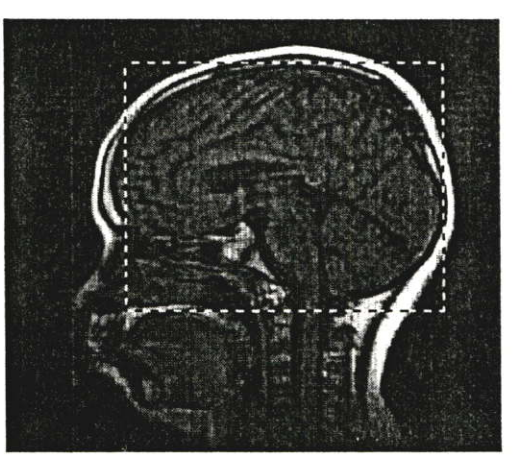
ตารางที่ 5.3 ผลการทดลอง

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1a ภาพจากเครื่องแปลงข้อมูลภาพ	**รูปที่** 2a ภาพต้นฉบับ
	
รูปที่ 1b เลือกบริเวณที่จะหาฮิสโตแกรม	**รูปที่** 2b เลือกบริเวณที่จะหาฮิสโตแกรม

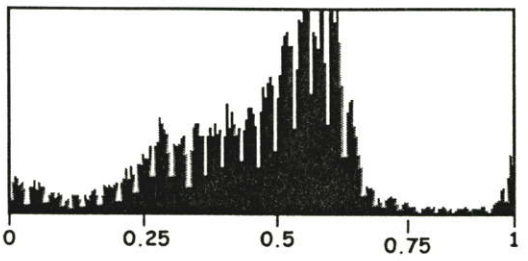
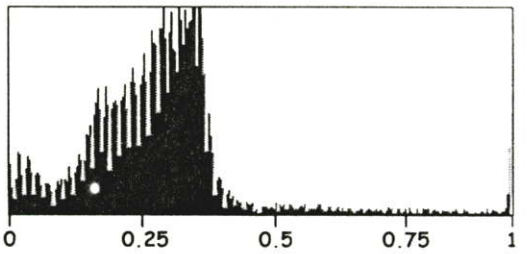
ตารางที่ 5.3 (ต่อ)

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1c ผลการหาฮีสโตแกรม	**รูปที่** 2c ผลการหาฮีสโตแกรม

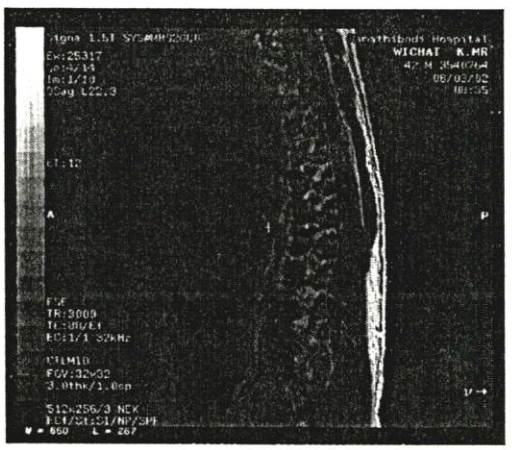
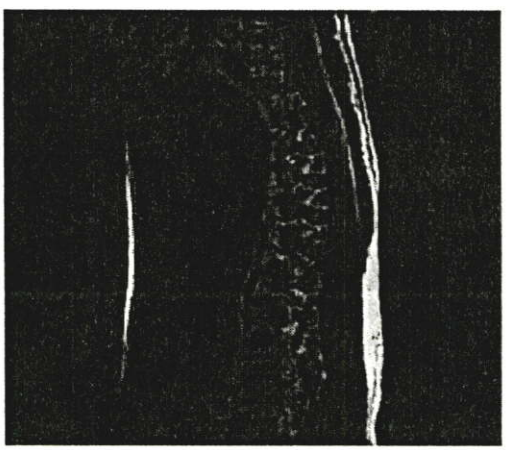
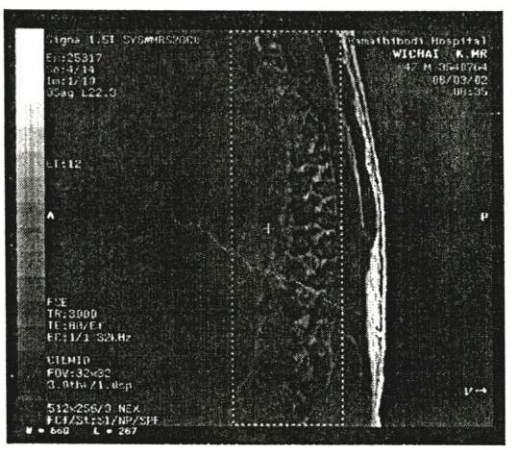
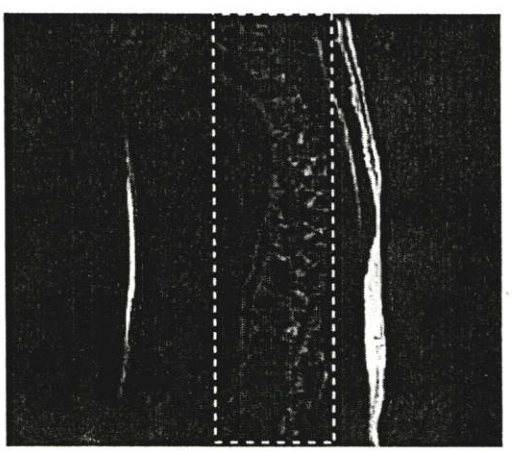
ตารางที่ 5.4 ผลการทดลอง

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1a ภาพจากเครื่องแปลงข้อมูลภาพ	**รูปที่** 2a ภาพต้นฉบับ
	
รูปที่ 1b เลือกบริเวณที่จะหาฮิสโตแกรม	**รูปที่** 2b เลือกบริเวณที่จะหาฮิสโตแกรม

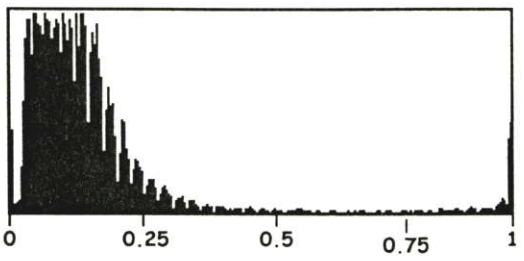
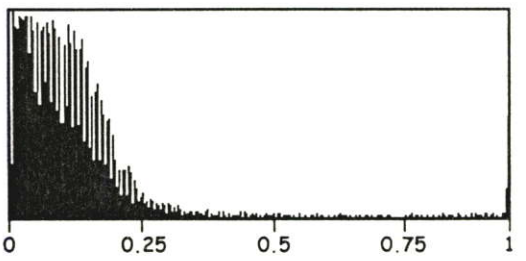
ตารางที่ 5.4 (ต่อ)

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1c ผลการหาฮีสโตแกรม	**รูปที่** 2c ผลการหาฮีสโตแกรม

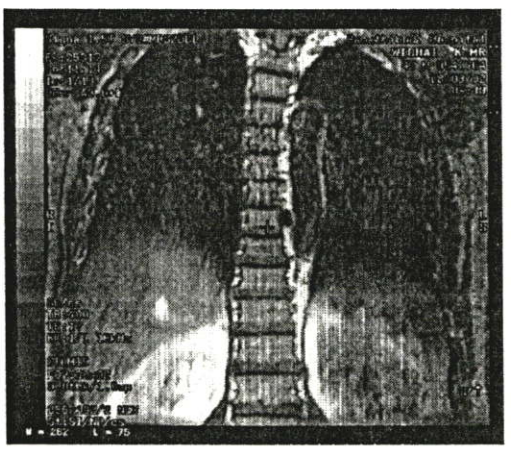
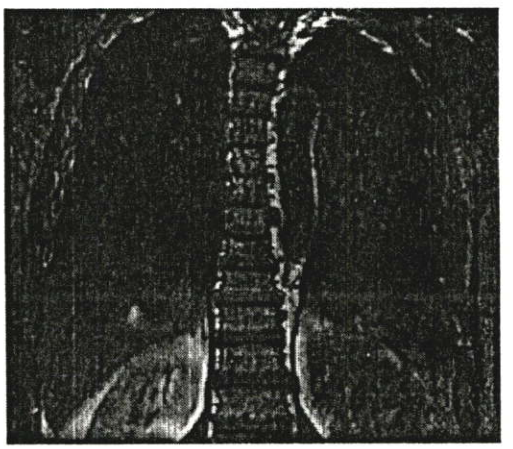
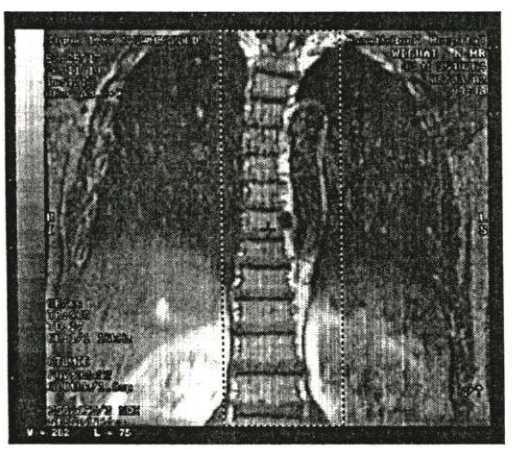
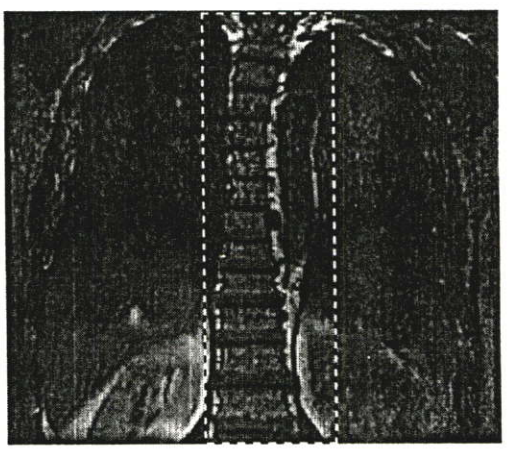
ตารางที่ 5.5 ผลการทดลอง

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1a ภาพจากเครื่องแปลงข้อมูลภาพ	**รูปที่** 2a ภาพต้นฉบับ
	
รูปที่ 1b เลือกบริเวณที่จะหาฮิสโตแกรม	**รูปที่** 2b เลือกบริเวณที่จะหาฮิสโตแกรม

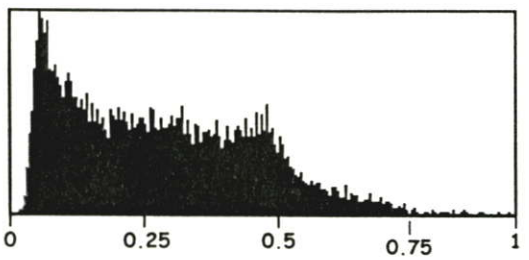
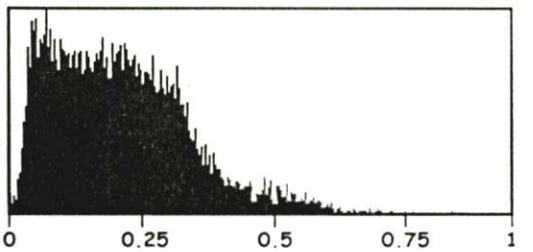
ตารางที่ 5.5 (ต่อ)

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1c ผลการหาฮีสโตแกรม	**รูปที่** 2c ผลการหาฮีสโตแกรม

ตารางที่ 5.6 ผลการทดลอง

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1a ภาพจากเครื่องแปลงข้อมูลภาพ	**รูปที่** 2a ภาพต้นฉบับ
	
รูปที่ 1b เลือกบริเวณที่จะหาฮิสโตแกรม	**รูปที่** 2b เลือกบริเวณที่จะหาฮิสโตแกรม

ตารางที่ 5.6 (ต่อ)

ผลการทดลอง	ภาพต้นแบบ
	
รูปที่ 1c ผลการหาฮีสโตแกรม	**รูปที่** 2c ผลการหาฮีสโตแกรม

จากฮีสโตแกรมในตารางที่ 5.1 ถึงตารางที่ 5.6 เปรียบเทียบกันระหว่างภาพที่ได้จากเครื่องแปลงสัญญาณภาพทางการแพทย์กับภาพต้นฉบับจะเห็นว่าฮีสโตแกรมจะมีลักษณะที่คล้ายคลึงกัน สาเหตุที่ไม่สามารถนำภาพมาเปรียบเทียบกันโดยตรงได้นั้นเพราะว่าขนาดของภาพต้นฉบับจะมีขนาด 256x256 หรือ 512x512 แต่ภาพที่เก็บมาได้โดยเครื่องแปลงสัญญาณภาพทางการแพทย์จะมีขนาดที่ใหญ่กว่าขึ้นอยู่กับการกำหนดขนาดของภาพที่จะทำการจัดเก็บ

บทที่ 6
สรุปผลการวิจัยและข้อเสนอแนะ

6.1 สรุปผลการวิจัย

งานวิจัยนี้นำเสนอแนวทางการออกแบบเครื่องแปลงสัญญาณภาพทางการแพทย์โดยใช้เอฟพีจีเอ ซึ่งมีขั้นตอนการออกแบบต่างๆดังที่กล่าวมาแล้ว จะเห็นว่าสามารถสร้างเครื่องแปลงสัญญาณภาพทางการแพทย์ โดยใช้ชิฟเอฟพีจีเอเป็นตัวควบคุมการทำงานต่างๆ ได้ซึ่งในส่วนของชิฟเอฟพีจีเอได้ถูกออกแบบให้ทำหน้าที่ในการเชื่อมต่อกับคอมพิวเตอร์ผ่านทางสล็อต ISA , ทำหน้าที่ในการเลือกตำแหน่งและขนาดของข้อมูลภาพที่จะทำการจัดเก็บและทำหน้าที่ควบคุมโหมดการทำงานต่างๆ ของวงจรทั้งหมดอีกด้วย ในการออกแบบชิฟเอฟพีจีเอจะใช้ภาษาวีเอชดีแอล ในการอธิบายลักษณะพฤติกรรมของวงจรที่ต้องการ จากนั้นจึงทำการดาวน์โหลดข้อมูลทางลอจิกลงในชิฟเอฟพีจีเอแล้วนำไปทดสอบการทำงาน ชิฟเอฟพีจีเอที่ได้ทำการออกแบบมาสามารถทำงานได้อย่างถูกต้องและมีประสิทธิภาพจะเห็นว่าชิฟเอฟพีจีเอเหมาะสำหรับการออกแบบระบบต้นแบบ เพราะเอฟพีจีเอสามารถกำหนดฟังค์ชันการทำงานตามความต้องการของผู้ออกแบบได้ อีกทั้งการแก้ไขปรับปรุงยังทำได้ง่ายสะดวกและรวดเร็ว ทำให้การออกแบบระบบต้นแบบของวงจรทางดิจิตอลทำได้รวดเร็วยิ่งขึ้น จะใช้ทรัพยากรที่สำคัญภายในชิฟเอฟพีจีเอสำหรับใช้กับเครื่องแปลงสัญญาณภาพทางการแพทย์เป็นดังนี้

- ใช้ลอจิกเซลประมาณ 121 ลอจิกเซลจากทั้งหมด 160 ลอจิกเซลคิดเป็น 75 % ของลอจิกเซลทั้งหมด
- ใช้ขา I/O จำนวน 59 ขา จากทั้งหมด 60 ขา คิดเป็น 98% ของทั้งหมด
- ใช้ฟลิปฟลอบ 87 ตัว

ในส่วนของข้อมูลภาพที่ได้จากเครื่องแปลงสัญญาณภาพทางการแพทย์เป็นการยากที่จะนำภาพดังกล่าวมาทำการเปรียบเทียบกับข้อมูลภาพต้นฉบับโดยตรง เพราะข้อมูลภาพที่ได้จากเครื่องแปลงสัญญาณภาพทางการแพทย์จะมีขนาดและอัตราส่วนของภาพที่ใหญ่กว่าภาพต้นฉบับ ในการนำภาพที่ได้จากการจัดเก็บไปใช้ในการวินิจฉัยโรคให้กับคนไข้ได้รับคำแนะนำจาก ผศ.นพ ราเมศร์ วัชรสินธุ์ ภาควิชารังสี โรงพยาบาลรามาธิบดี ว่าภาพดังกล่าวมีคุณสมบัติเพียงพอที่จะนำไปใช้ในการวินิจฉัยโรคได้

6.2 อุปสรรคที่พบในการวิจัย

ในการทดลองมีปัญหาอย่างมากในเรื่องของความเร็วที่ใช้ในการจัดเก็บ โดยอุปกรณ์ที่นำมาใช้จะต้องมีความเร็วที่สูงเพียงพอที่จะทำงานได้โดยไม่มีข้อผิดพลาด อีกทั้งการจัดหาออสซิเลเตอร์ไม่ได้ตามค่าที่คำนวณไว้ ทำให้ขนาดของภาพที่ได้มีขนาดที่ใหญ่กว่าภาพต้นฉบับ อุปกรณ์ที่สำคัญอีกอย่างหนึ่งคือหน่วยความจำจะต้องใช้หน่วยความจำที่มีความจุสูงและมีค่า Access Time ที่ต่ำอีกด้วย ซึ่งไม่มีขายในเมืองไทยจะต้องสั่งซื้อจากต่างประเทศ

6.3 ข้อเสนอแนะในการพัฒนา

ในการพัฒนางานวิจัยนี้สิ่งที่ควรจะทำการพัฒนาต่อไปคือ

1.แก้ปัญหาเรื่องความคลาดเคลื่อนของระดับความเข้ม (ระดับเทา) ของภาพให้ถูกต้อง

2.พัฒนาให้เป็นระบบที่สามารถทำงานได้ตามเวลาจริง

3.พัฒนาให้สามารถเก็บภาพสีได้

4.พัฒนาโปรแกรมเพื่อใช้ในการประมวลผลภาพที่ได้

5.พัฒนาทางด้านโปรแกรมเพื่อทำการบีบอัดข้อมูลภาพ เพื่อประหยัดเนื้อที่ในการจัดเก็บข้อมูล

6.ทำการแก้ไขเรื่องสัญญาณรบกวน โดยการออกแบบ PCB ที่สามารถลดสัญญาณหรือป้องกันสัญญาณรบกวนได้

7.แก้ไขเรื่องความถี่ออสซิเลเตอร์ โดยเปลี่ยนจากการใช้โมดูลออสซิเลเตอร์เป็นการใช้ Phase Lock Loop [15] เป็นตัวกำเนิดสัญญาณออสซิเลเตอร์ ซึ่งจะทำให้ได้ความถี่ออสซิเลเตอร์ตามที่คำนวณไว้

บรรณานุกรม

[1] Xu. D., Boussakta S., Bentley J.P. "An FPGA-Based Low-Cost Frame Grabber For Image Processing Applications." Electronics, Circuits and Systems, 2000, ICECS 2000. The 7th IEEE International Conference on , Volume: 1 , 2000 PP: 333 –336

[2] Li Qing., Li Wei., Yuan Xin., Xiao Ping., Hu Jingyu., Yang Liangxing. "A TV Graphics Demonstration System Design Using FPGA."ASIC, 1996., 2nd International Conference on , 1996. PP : 228 -230

[3] Scott E Umbaugh. **Computer Vision and Image Processing.** Prentice-Hall International Inc, 1998

[4] Hitachi. **Technical Guide for Whole-Body X-Ray CT System Model CT-W4-40.** Hitachi Company, 1983

[5] R. C. Gonzalez and R. E. Woods. **Digital Image Processing : Digital image fundamentals.** Addison-wesley Publishing Company, 1992

[6] บัณฑิต สมุนวัฒนเดช. "แผงวงจรเก็บข้อมูลภาพที่ประกอบด้วยฟังค์ชันหน่วยความจำ." วิทยานิพนธ์สำหรับปริญญาวิศวกรรมศาสตร์มหาบัณฑิต สาขาวิศวกรรมไฟฟ้า บัณฑิตวิทยาลัย, สถาบันเทคโนโลยีพระจอมเกล้าลาดกระบัง. ปีการศึกษา 2539

[7] Astron Logic Research and Development, "เปิดโลก FPGA กับบอร์ด WIZARD PLD-A01." [Online]. Availble : http://www.astronlogic.com/

[8] Andrew Rushton. **VHDL for Logic Synthesis.** John Wiley & Sons Ltd, 1998

[9] Altera Corporation, "MAX7000 Programmable Logic Device Family." [Online]. Availble : http://www.altera.com/literature/manual/

[10] Altera Corporation, "FLEX10K Embedded Programmable Logic Family." [Online]. Availble : http://www.altera.com/literature/manual/

[11] Altera Corporation, "MAX Plus II Tutorial"; [Online]. Availble : http://www.altera.com/literature/manual/

[12] สหัส ฟักอ่อน. "เครื่องแปลงสัญญาณวิดีโอเป็นข้อมูลภาพสำหรับเครื่องมือแพทย์." วิทยานิพนธ์สำหรับปริญญาวิศวกรรมศาสตร์มหาบัณฑิต สาขาวิศวกรรมไฟฟ้า บัณฑิตวิทยาลัย, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. ปีการศึกษา 2542

[13] Avtar Singh , Walter A. Triebel. **16-Bit and 32 Bits Microprocessors Architecture Software and Interfacing Techniques.** Prentice-Hall International, 1991

[14] บุญอนันต์ เกียงเอีย, สุพันธุ์ ตั้งจิตกุศลมั่น, สุรพันธุ์ เอื้อไพบูลย์, มนัส สังวรศิลป์ "การออกแบบเครื่องแปลงสัญญาณภาพทางการแพทย์โดยใช้เอฟพีจีเอ." เอกสารการประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 24 ปี2544. หน้า 945-950

[15] R. Monleone. "Clock-Recovery PLL Fits Into Single PLD". EDN Europe. November 1996, pp. 36-37

ภาคผนวก

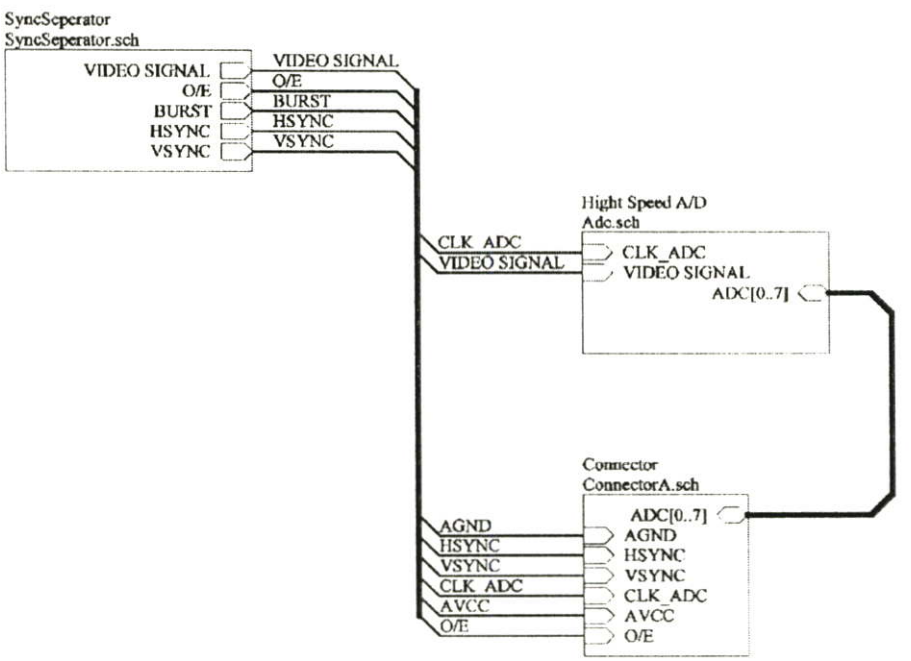
ภาคผนวก ก
วงจร

ในภาคผนวกนี้จะประกอบด้วย วงจรทั้งหมดที่ประกอบขึ้นเป็นเครื่องแปลงสัญญาณภาพทางการแพทย์ด้วยเอฟพีจีเอ ซึ่งจะประกอบด้วย 2 ส่วนคือ

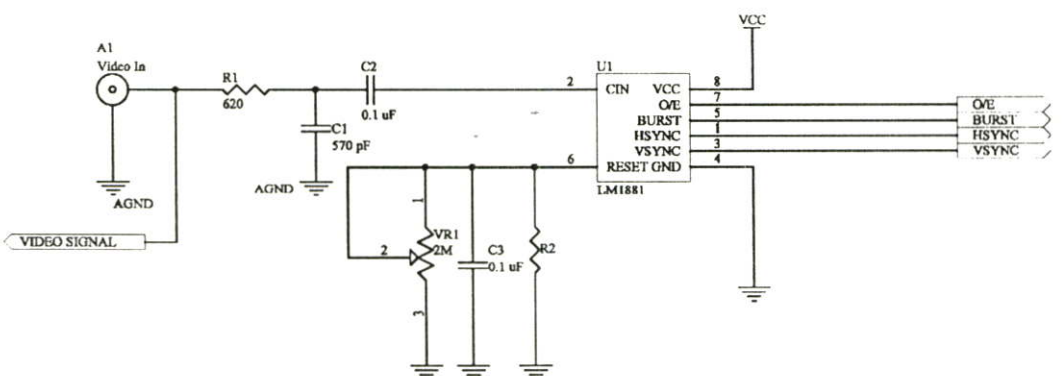
ส่วนที่ 1 วงจรทางอนาลอก

ส่วนที่ 2 วงจรทางดิจิตอล

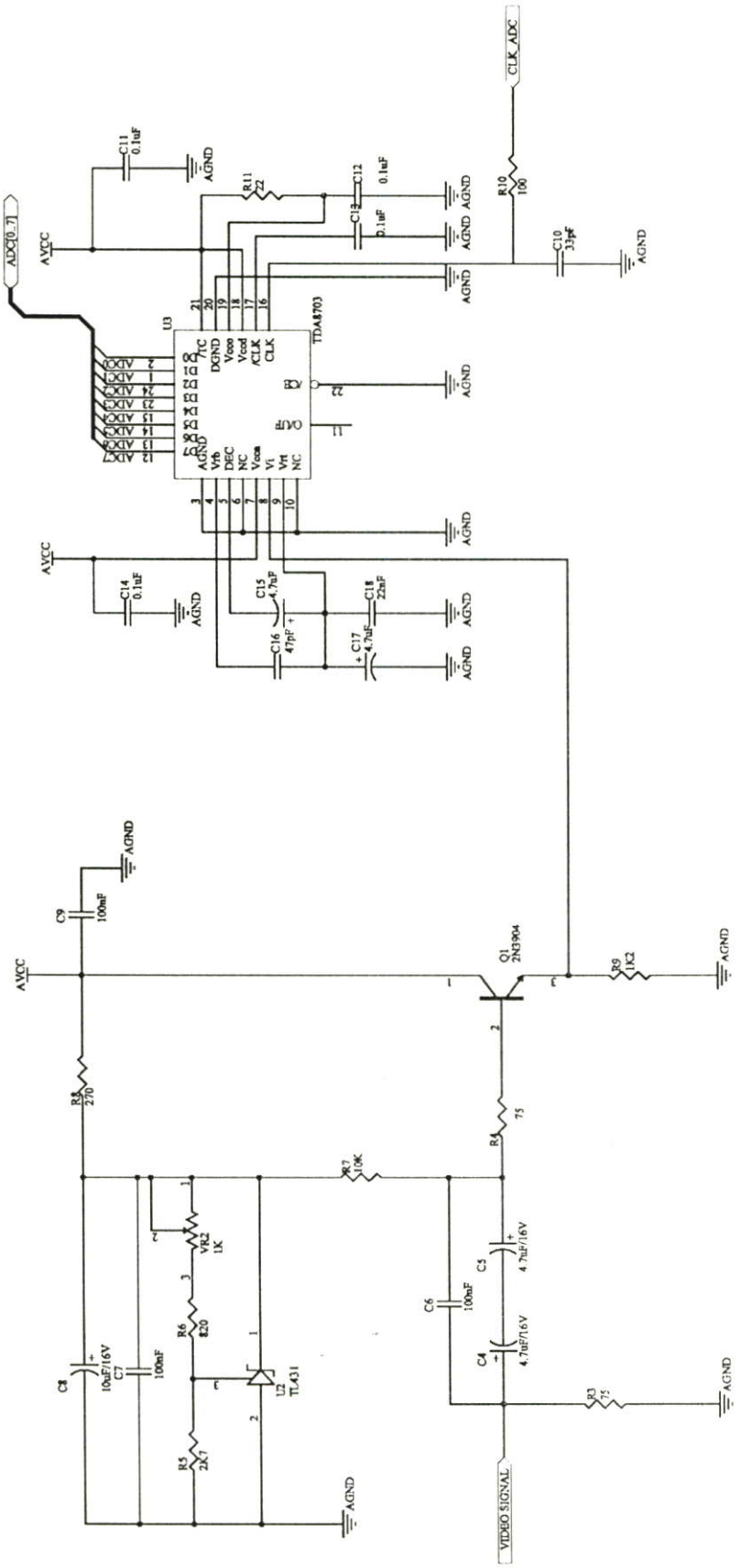
ส่วนที่ 1 วงจรทางอนาลอก



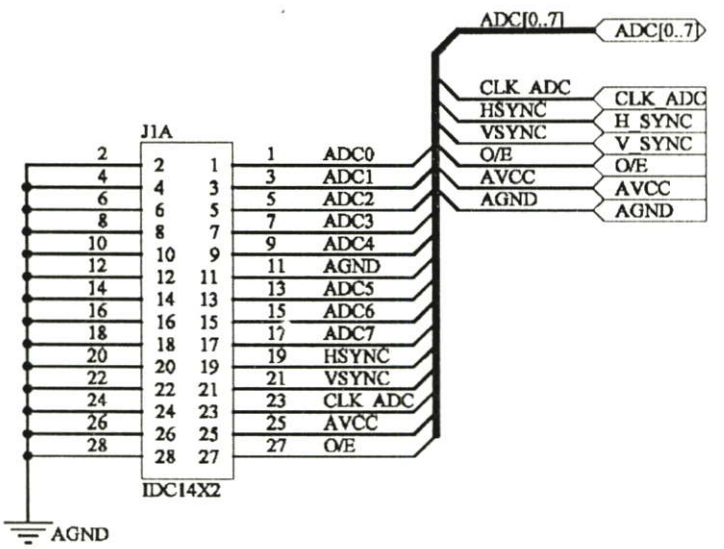
รูปที่ 1 วงจรโดยรวมของส่วนอนาลอก



รูปที่ 2 วงจรแยกสัญญาณซิงค์

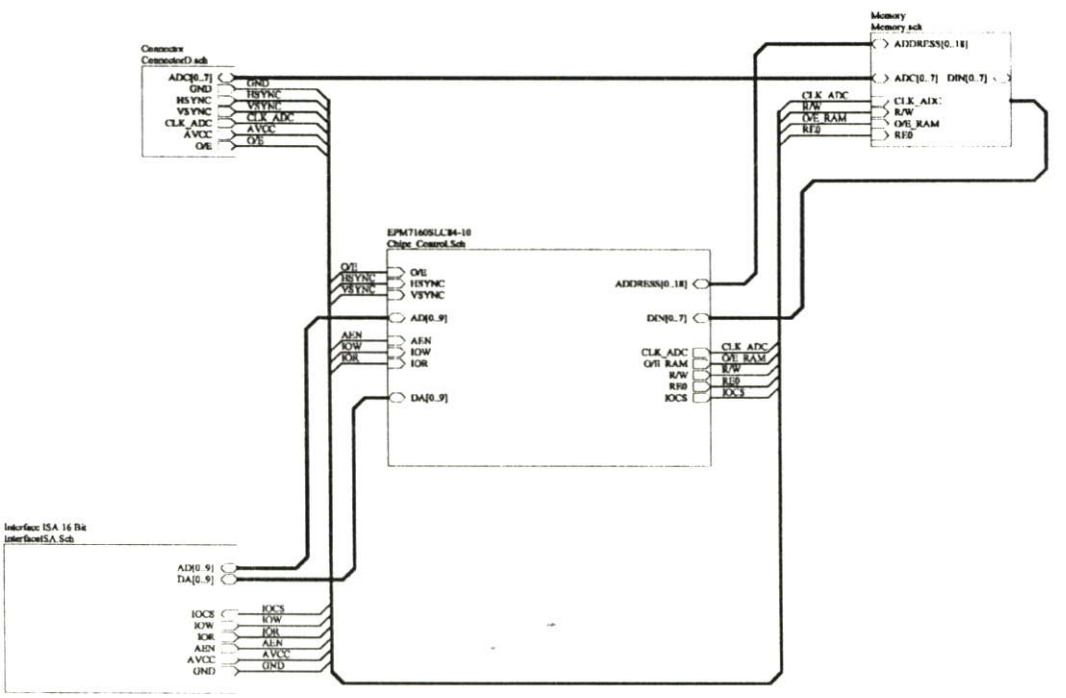


รูปที่ 3 วงจรแปลงสัญญาณอนาลอกเป็นดิจิตอล

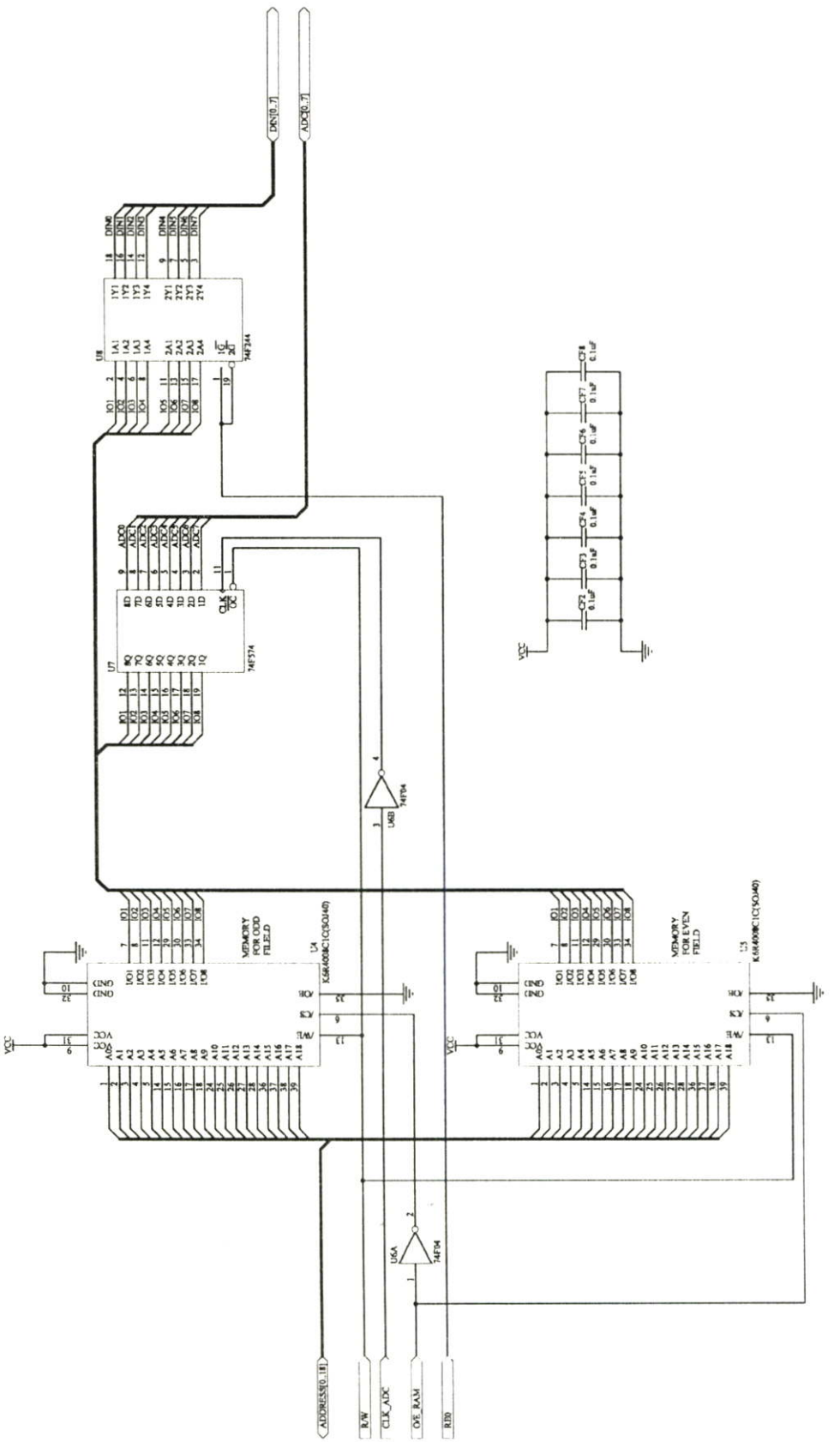


รูปที่ 4 คอนเน็กเตอร์ส่วนอนาลอก

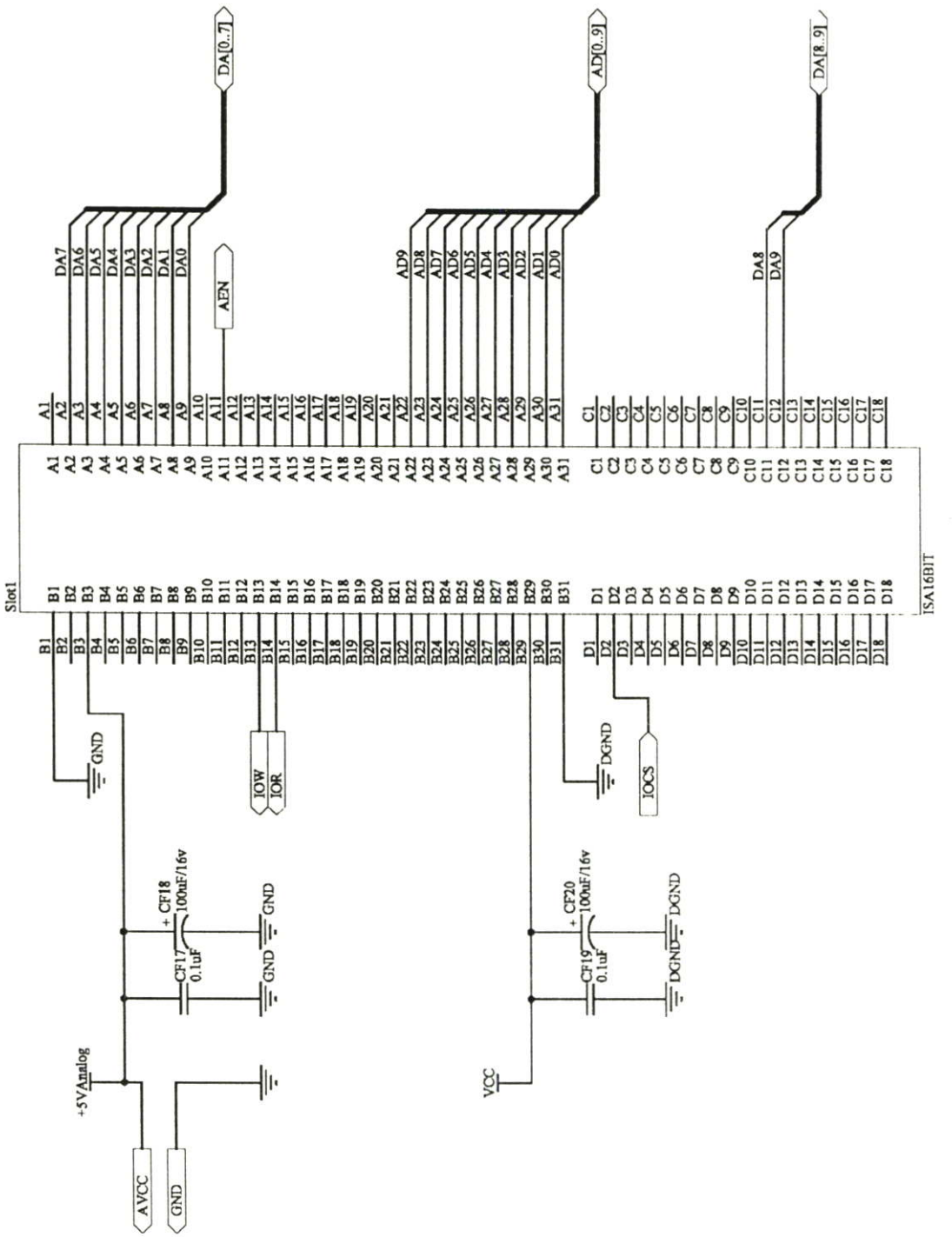
ส่วนที่ 2 วงจรทางดิจิตอล



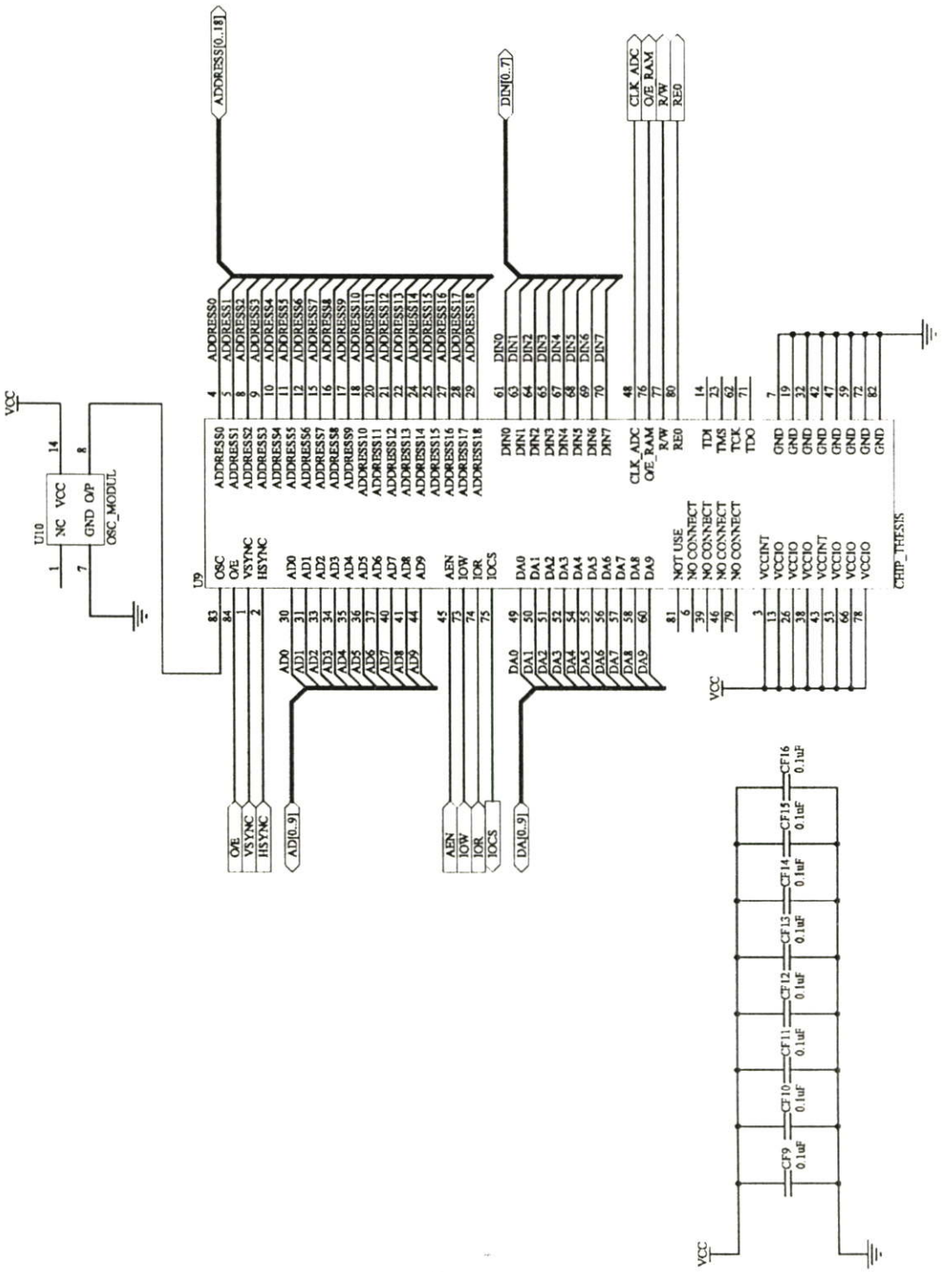
รูปที่ 5 วงจรโดยรวมของส่วนดิจิตอล



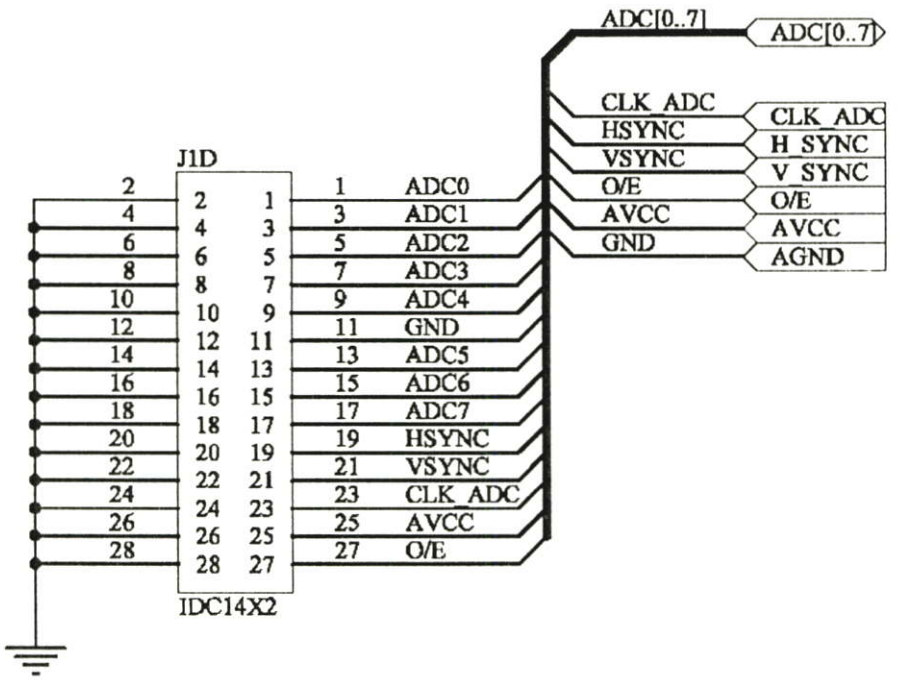
รูปที่ 6 วงจรหน่วยความจำ



รูปที่ 7 วงจรอินเตอร์เฟสกับคอมพิวเตอร์ผ่านทางสล๊อต ISA



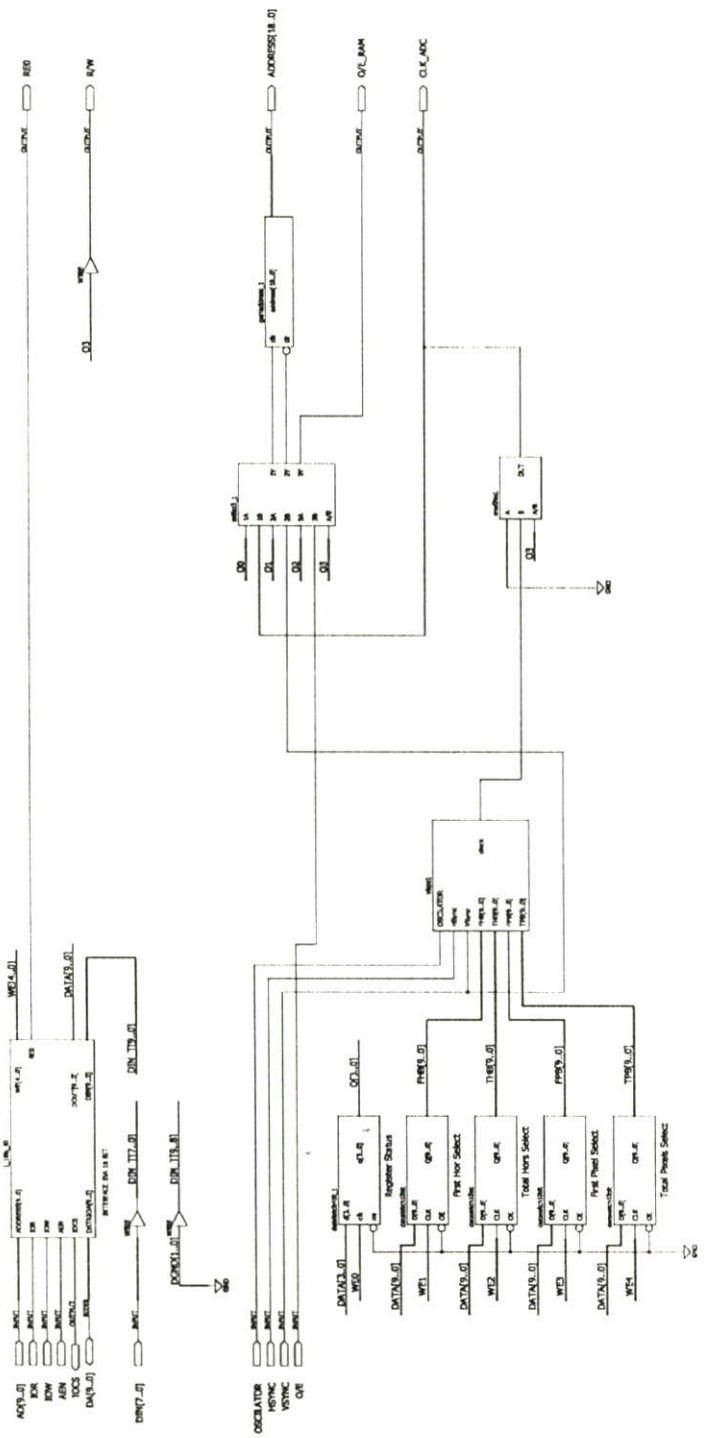
รูปที่ 8 วงจรควบคุมด้วยเอฟพีจีเอ



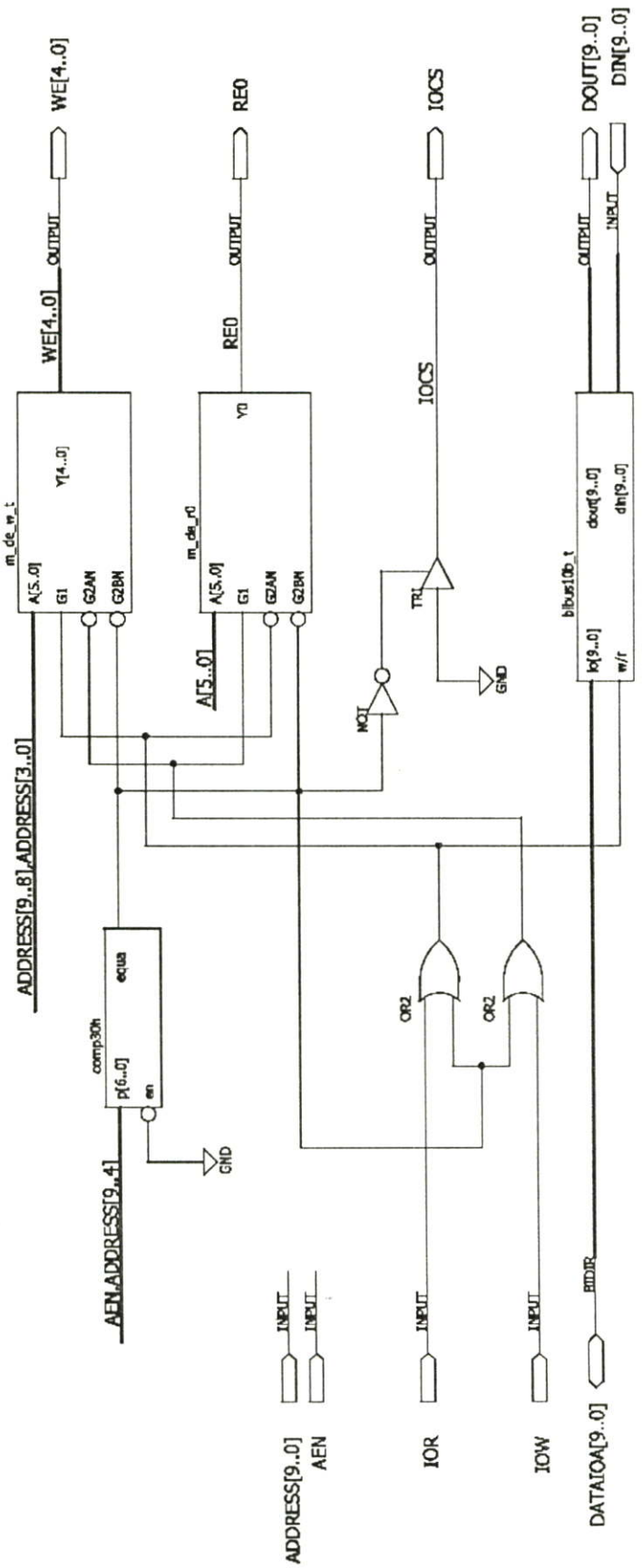
รูปที่ 9 คอนเน็กเตอร์ส่วนดิจิตอล

ภาคผนวก ข
วงจรภายในเอฟพีจีเอ

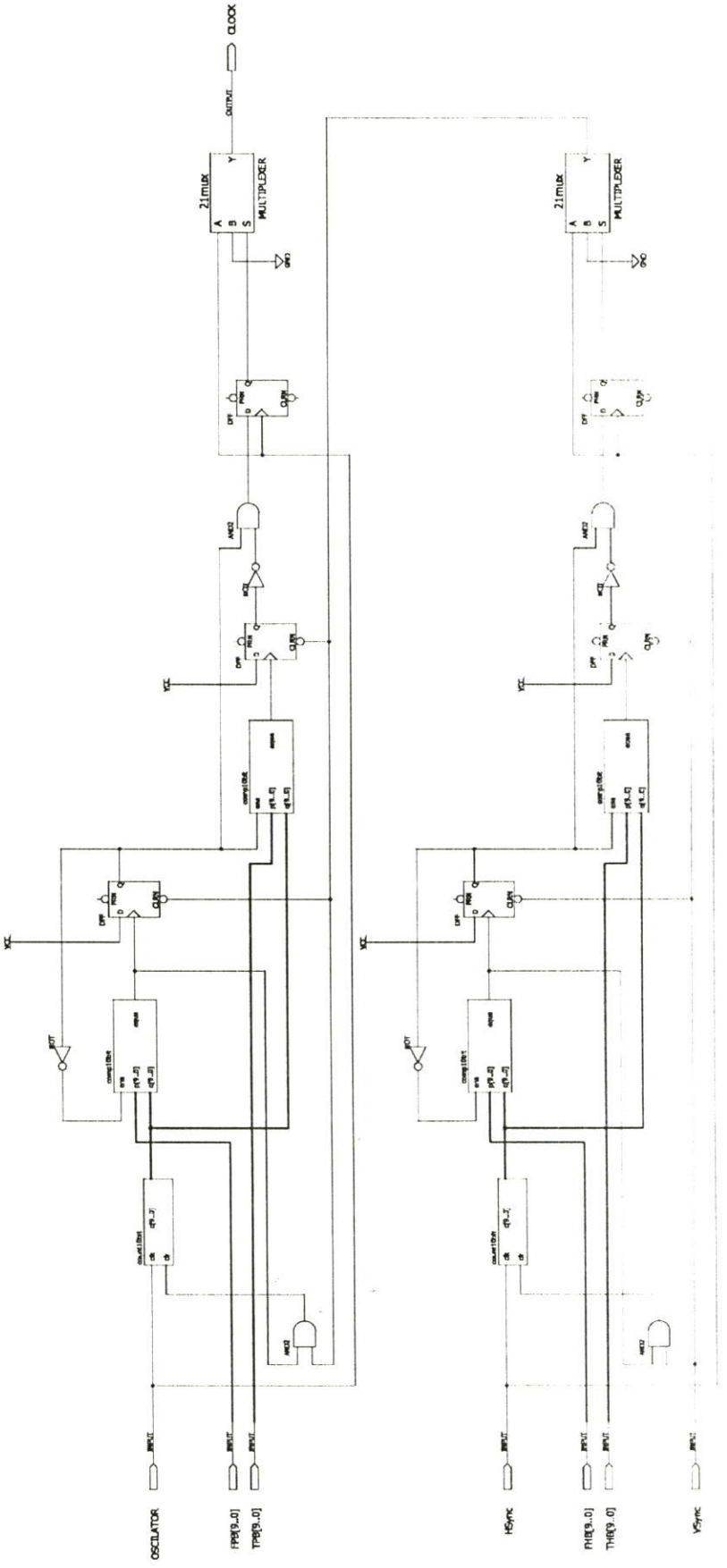
ในภาคผนวกนี้จะประกอบด้วย แผนผังวงจร (Schematics) และภาษา HDL ที่ออกแบบมาสำหรับใช้เป็นชุดควบคุมการทำงานของเครื่องแปลงสัญญาณภาพทางการแพทย์ด้วยเอฟพีจีเอ



รูปที่ 1 วงจรโดยรวมของชุดควบคุมทั้งหมด



รูปที่ 2 วงจรโดยรวมของโมดูลวงจรอินเตอร์เฟส ISA 16 บิต



รูปที่ 3 วงจรโดยรวมของโมดูลวงจรเลือกเส้นสแกนและพิกเซลของข้อมูลภาพที่จะจัดเก็บ

โปรแกรมที่ 1 โมดูล DataLatch4Bit_t ทำหน้าที่เป็นรีจิสเตอร์ขนาด 4 บิต

```
SUBDESIGN DataLatch4Bit_t
(
d[3..0],oe,clk : INPUT = GND;
q[3..0] : OUTPUT;
)
VARIABLE
        dd[3..0] : DFF;
dtri[3..0] : TRI;
BEGIN
dd[].clk = clk;
dd[].d = d[];
dtri[].oe = !oe;
dtri[].in = dd[].q;
q[] = dtri[].out;
END;
```

โปรแกรมที่ 2 โมดูล DataLatch10Bit_t ทำหน้าที่เป็นรีจิสเตอร์ขนาด 10 บิต

```
SUBDESIGN DataLatch10Bit
(
d[9..0],oe,clk : INPUT = GND;
q[9..0] : OUTPUT;
)
VARIABLE
        dd[9..0] : DFF;
dtri[9..0] : TRI;
BEGIN
dd[].clk = clk;
dd[].d = d[];
dtri[].oe = !oe;
dtri[].in = dd[].q;
```

```
q[] = dtri[].out;
END;
```

โปรแกรมที่ 3 โมดูล Mul2To1 ทำหน้าที่เป็น Multiplex 2 line to 1 line

```
SUBDESIGN Mul2To1
(
        A,B : INPUT;
        A/B : INPUT;
        OUT : OUTPUT
)
VARIABLE
N1 : NODE;

BEGIN
if A/B == B"1" then
        N1 = A;
else
        N1 = B;
end if;
OUT = N1;
END;
```

โปรแกรมที่ 4 โมดูล Mul6To3_t ทำหน้าที่เป็น Multiplex 6 line to 3 line

```
SUBDESIGN M6to3_t
(
        1A,1B,2A,2B,3A,3B,A/B : INPUT;
        1Y,2Y,3Y : OUTPUT;
)

VARIABLE
N1,N2,N3 : NODE;
```

```
BEGIN
if a/b == B"1" then
        N1 = 1A;
        N2 = 2A;
        N3 = 3A;
else
        N1 = 1B;
        N2 = 2B;
        N3 = 3B;
end if;
1Y = N1;
2Y = N2;
3Y = N3;
END;
```

โปรแกรมที่ 5 โมดูล GenAddress_t ทำหน้าที่กำเนิดสัญญาณแอดแดรสให้หน่วยความจำ

```
SUBDESIGN GenAddress_t
(
        clk : INPUT;
        clr : INPUT;
        address[18..0] : OUTPUT;
)
VARIABLE
N_add[18..0] : DFF;

BEGIN
N_add[].clk = clk;
N_add[].clrn = clr;
N_add[].d = N_add[].q+1;
address[] = N_add[];
end;
```

โปรแกรมที่ 6 โมดูล Comp30H ทำหน้าที่เปรียบเทียบสัญญาณอินพุทกับค่า 30H

```
SUBDESIGN Comp30H
(
p[6..0] : INPUT = VCC;
en : INPUT = VCC;
equa : OUTPUT;
)
VARIABLE
e : LATCH;
BEGIN
DEFAULTS
        e.d = VCC;
END DEFAULTS;
e.ena = VCC;
IF p[] == H"30" THEN
        e.d=GND;
END IF;
equa = e.q # en;
END;
```

โปรแกรมที่ 7 โมดูล m_de_w_t ทำหน้าที่เลือกพอร์ต I/O สำหรับเขียนข้อมูล

```
SUBDESIGN m_de_w_t
(
A[5..0] : INPUT;
G1,G2AN,G2BN : INPUT;
Y[4..0] : OUTPUT;
)
VARIABLE
re[4..0] : NODE;
en : NODE;
BEGIN
```

```
en = G1 & !G2AN & !G2BN ;
IF (A[] == H"30") & en THEN re[]    = B"11110";
ELSIF (A[] == H"32") & en THEN re[] = B"11101";
ELSIF (A[] == H"34") & en THEN re[] = B"11011";
ELSIF (A[] == H"36") & en THEN re[] = B"10111";
ELSIF (A[] == H"38") & en THEN re[] = B"01111";
ELSE re[] = B"11111";
END IF;
Y[] = re[];
END;
```

โปรแกรมที่ 8 โมดูล m_de_r0 ทำหน้าที่เลือกพอร์ต I/O สำหรับอ่านข้อมูล

```
SUBDESIGN m_de_r0
(
A[5..0] : INPUT;
G1,G2AN,G2BN : INPUT;
Y0 : OUTPUT;
)
VARIABLE
re0 : NODE;
en : NODE;

BEGIN
en = G1 & !G2AN & !G2BN ;
IF (A[] == H"30") & en THEN re0 = B"0";
ELSE re0 = B"1";
END IF;
Y0 = re0;
END;
```

โปรแกรมที่ 9 โมดูล BiBus10B_T ทำหน้าที่เป็น Bi-directional Bus

```
SUBDESIGN BiBus10B_T
(
din[9..0] : INPUT = VCC;
w/r : INPUT = VCC;
dout[9..0] : OUTPUT;
io[9..0] : BIDIR = VCC;
)
VARIABLE
inlatch[9..0] : LATCH;
intri[9..0] : TRI;

BEGIN
intri[].in = din[];
intri[].oe = !w/r;
io[] = intri[].out;

inlatch[].d = io[];
inlatch[].ena = w/r;
dout[] = inlatch[].q;
END;
```

โปรแกรมที่ 10 โมดูล Count10Bit ทำหน้าที่เป็นเคาเตอร์แบบ 10 บิต

```
SUBDESIGN Count10Bit
(
clk,clr : INPUT;
c[9..0] : OUTPUT;
)
VARIABLE
count[9..0] : DFF;
BEGIN
```

```
count[].clk = clk;
count[].clrn = clr;
count[].d = count[].q + 1;
c[] = count[];
END;
```

โปรแกรมที่ 11 โมดูล Comp10Bit ทำหน้าที่เปรียบเทียบสัญญาณทางอินพุทขนาด 10 บิต

```
SUBDESIGN Comp10Bit
(
ena,p[9..0],q[9..0] : INPUT;
equa : OUTPUT;
)
VARIABLE
a : LATCH;
BEGIN
DEFAULTS
        a.d = VCC;
END DEFAULTS;
equa = a.q;
a.ena = VCC;

IF ena == B"1" THEN
        IF p[] == q[] THEN
                a.d = GND;
        ELSE
                a.d = VCC;
        END IF;
ELSE
        a.d = VCC;
END IF;
END;
```

ประวัติผู้เขียน

ชื่อ นายบุญอนันต์ เกียงเอีย

เกิดวันที่ 1 เมษายน 2519

สถานที่เกิด รพ.ปะเหลียน จ.ตรัง

การศึกษา

- ปี 2532-2534 ศึกษาในระดับมัธยมศึกษาชั้นปีที่ 1 ถึงมัธยมศึกษาชั้นปีที่ 3 โรงเรียนปะเหลียนผดุงศิษย์
- ปี 2535-2537 ศึกษาในระดับประกาศนียบัตรวิชาชีพ (ปวช) สาขาวิชาอิเล็กทรอนิกส์ วิทยาลัยเทคนิคตรัง
- ปี 2538-2539 ศึกษาในระดับประกาศนียบัตรวิชาชีพชั้นสูง (ปวส) สาขาอิเล็กทรอนิกส์ ภาควิชาอิเล็กทรอนิกส์ วิทยาลัยเทคโนโลยีอุตสาหกรรม สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ
- ปี 2540-2542 ศึกษาในระดับปริญญาตรี สาขาอิเล็กทรอนิกส์ ภาควิชาวิศวกรรมอิเล็กทรอนิกส์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
- ปี 2543-2544 ศึกษาในระดับปริญญาโท สาขาอิเล็กทรอนิกส์ ภาควิชาวิศวกรรมอิเล็กทรอนิกส์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง