การพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว โดยใช้ FPGA

ON THE IMPLEMENTATION OF POCSAG PAGING DECODER USING FPGA

อดุลย์ ขันติชนะกุล
ADUL KANTICHANAKUL

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมอิเล็กทรอนิกส์
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2546

ISBN 974-324-741-0

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

การพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว โดยใช้ FPGA

ON THE IMPLEMENTATION OF POCSAG PAGING DECODER USING FPGA

อดุลย์ ขันติชนะกุล

ADUL KANTICHANAKUL

วิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชาวิศวกรรมอิเล็กทรอนิกส์

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2546

ISBN 974 – 324 –741– 6

เลขหมู่.....
เลขทะเบียน 48377
วัน, เดือน, ปี 24 ต.ค. 2546

ON THE IMPLEMENTATION OF POCSAG PAGING DECODER USING FPGA

ADUL KANTICHANAKUL

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF ENGINEERING IN ELECTRONIC ENGINEERING

SCHOOL OF GRADUATE STUDIES

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2003

ISBN 974 – 324 – 741 – 6

COPYRIGHT 2003

SCHOOL OF GRADUATE STUDIES

KING MONGKUT 'S INSTITUTE OF TECHNOLOGY LADKRABANG

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองวิทยานิพนธ์

หัวข้อวิทยานิพนธ์ การพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว โดยใช้ FPGA
ON THE IMPLEMENTATION OF POCSAG PAGING DECODER USING FPGA

ชื่อนักศึกษา นายอดุลย์ ขันติชนะกุล

รหัสประจำตัว 43061305

ปริญญา วิศวกรรมศาสตรมหาบัณฑิต

สาขาวิชา วิศวกรรมอิเล็กทรอนิกส์

อาจารย์ผู้ควบคุมวิทยานิพนธ์ ศ.ดร.วัลลภ สุระกำพลธร

คณะกรรมการสอบวิทยานิพนธ์		ลายมือชื่อ
ผศ.ประภากร	สุวรรณะ	
ผศ.ขนิษฐา	แซ่ตั้ง	
รศ.ดร.สมศักดิ์	ชุมช่วย	
ผศ.ดร.เกียรติศักดิ์	คมวัชระ	
ศ.ดร.วัลลภ	สุระกำพลธร	

วัน/เดือน/ปี ที่สอบ 29 กรกฎาคม 2546 เวลา 14.00-16.00 น.

สถานที่สอบ ณ อาคาร 12 ชั้น ชั้น 4 (ห้อง E12-404)

บัณฑิตวิทยาลัยรับรองแล้ว

(รศ.ดร.บุญวัฒน์ อัตชู)

คณบดีบัณฑิตวิทยาลัย

วันที่ 1 **เดือน** สิงหาคม **พ.ศ.** 2546

หัวข้อวิทยานิพนธ์	การพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว โดยใช้ FPGA
นักศึกษา	นาย อดุลย์ ขันติชนะกุล
รหัสประจำตัว	43061305
ปริญญา	วิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชา	อิเล็กทรอนิกส์
พ.ศ.	2546
อาจารย์ผู้ควบคุมวิทยานิพนธ์	ศ.ดร.วัลลภ สุระกำพลธร

บทคัดย่อ

วิทยานิพนธ์นี้นำเสนอการพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว โดยใช้เอฟพีจีเอ(FPGA) ซึ่งสามารถแสดงข่าวสารข้อมูลได้ทั้งในรูปแบบตัวอักษรภาษาไทยและภาษาอังกฤษ โดยโครงสร้างของตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัวจะประกอบไปด้วย ภาคการทำงานทางด้านดิจิตอล ซึ่งได้แก่ ภาคถอดรหัสเพจเจอร์ ภาคถอดรหัสหมายเลขเรียกขาน ภาคควบคุมการทำงานระบบ ภาคเชื่อมต่ออุปกรณ์แสดงผล และภาคเชื่อมต่อแสดงผลข้อความ โดยการออกแบบจะอยู่ในชิฟวงจรรวม ซึ่งแนวทางการพัฒนาได้ใช้ภาษาวีเอชดีแอล (VHDL)ในการบรรยายพฤติกรรมของฮาร์ดแวร์ที่ทำการออกแบบแล้วโปรแกรมลงบนชิฟเอฟพีจีเอ และทำการทดสอบการทำงานของตัวถอดรหัสข้อมูลในระบบ POCSAG เพื่อยืนยันการทำงานที่ถูกต้อง

Thesis Title	On the Implementation of Pocsag Paging Decoder using FPGA
Student	Mr. Adul Kantichanakul
Student ID.	43061305
Degree	Master of Engineering
Programme	Electronics Engineering
Year	2003
Thesis Advisor	Prof.Dr.Wanlop Surakampontorn

ABSTRACT

This thesis presents the development of a Pocsag paging decoder with the capability of displaying text information both in Thai and English. The implementation was limited to digital portion of the whole system. This comprises decoder codeword, address codeword, display device interface and LCD Ggraphic displays interface. Behavioral hardware description language (VHDL) design style was invoked. Design procedure included modelling, Simulation hardware programming. The resulted FPGA has been successfully tested for each functions with the good result.

กิตติกรรมประกาศ

ขอกราบพระคุณ ศาสตราจารย์ ดร. วัลลภ สุระกำพลธร อาจารย์ที่ปรึกษา เป็นอย่างสูง ที่ได้มอบโอกาสที่ดีแก่ข้าพเจ้า ในการทำงานวิจัย พร้อมให้กับความเมตตา กรุณา ในการอบรมสั่งสอนในความรู้ในด้านวิชาการ และด้านของการดำเนินการชีวิต ตลอดจนการแนะแนวทางในการแก้ไขปัญหาต่างๆ อย่างใส่ใจ จนทำให้ข้าพเจ้ามีสติ ปัญญา และ แรงใจ ในการทำวิทยานิพนธ์ฉบับนี้ให้สำเร็จสมบูรณ์ขึ้นได้

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร. เกียรติศักดิ์ คมวัชระ ผู้ช่วยศาสตราจารย์ ดร.วรพงษ์ ตั้งศรีรัตน์ ดร.คงศักดิ์ อนันตหิรัญรัตน์ ผู้ช่วยศาตราจารย์ กฤษดา กล่อมการ อาจารย์ชัยวัฒน์ จงกุลสถิตชัย และ นาย เฉลิมภัณฑ์ ฟองสมุทร(นักศึกษาปริญญาเอก) ที่ได้กรุณาให้คำแนะนำและความช่วยเหลือ อีกทั้งขอขอบพระคุณ พี่ๆ เพื่อนๆ และ น้องๆทุกคนในห้องปฏิบัติการ Mixed Signal Processing ที่ให้ความช่วยเหลือ เกื้อหนุน ข้าพเจ้าเป็นอย่างดี

ขอขอบพระคุณ ทศท. คอปอร์เรชั่น จำกัด มหาชน ที่สนับสนุนการศึกษา

ขอขอบพระคุณ บริษัทฟิลิปส์(ประเทศไทย) จำกัดที่ช่วยสนับสนุนไอซี Receiver Decoder Pager ในการทำวิจัย

ขอขอบพระคุณ องค์การร่วมมือระหว่างประเทศของญี่ปุ่น(JICA)ภายใต้โครงการสำนักวิจัยการสื่อสารและเทคโนโลยีสารสนเทศ(ReCCIT) ที่ให้ทุนและเครื่องมือสนับสนุนในการทำวิทยานิพนธ์ และขอขอบคุณสำนักงานกองทุนสนับสนุนการวิจัย (สกว.) ภายใต้โครงการที่ RTA/04/2543 ที่สนับสนุนในการทำงานวิจัย

สุดท้ายนี้ขอกราบขอบพระคุณ บิดา มารดา ตลอดจนครอบครัว ที่มอบความรัก ความห่วงใย กำลังใจ และให้ การสนับสนุน ในการศึกษามาโดยตลอด รวมทั้งคณาญาติ พี่ และ น้อง ที่คอยเป็นกำลังใจเสนอมา

คุณค่าและประโยชน์อันเกิดขึ้นจากวิทยานิพนธ์นี้ ข้าพเจ้าขอมอบแด่ บิดา มารดา ครอบครัว และผู้มีพระคุณทุกท่าน

อดุลย์ ขันติชนะกุล

สารบัญ

หน้า

บทคัดย่อภาษาไทย I

บทคัดย่อภาษาอังกฤษ II

กิตติกรรมประกาศ III

สารบัญ IV

สารบัญตาราง IX

สารบัญรูป X

บทที่ 1 บทนำ 1

1.1 ความเป็นมาและความสำคัญของปัญหา 1

1.2 วัตถุประสงค์ของการวิจัย.....1

1.3 ขอบเขตของการวิจัย 2

1.4 เปรียบเทียบกับหลักการที่มีอยู่เดิม 2

1.5 โครงสร้างของวิทยานิพนธ์ 2

บทที่ 2 ทฤษฎีพื้นฐานที่เกี่ยวข้องกับระบบวิทยุติดตามตัว 4

2.1 โครงสร้างพื้นฐานและการทำงานของระบบวิทยุติดตามตัว 4

2.2 เครื่องรับวิทยุติดตามตัว(Pager) 6

2.3 รูปแบบการฝากข้อความของระบบวิทยุติดตามตัว 7

บทที่ 3 โครงสร้างและรหัสมาตรฐาน POCSAG 9

3.1 รูปแบบของสัญญาณเพจเจอร์ 9

3.1.1 สัญญาณพัลส์ส่วนหน้า 10

3.1.2 โครงสร้างของแบตช์ข้อมูล 10

3.1.2.1 รหัสคำการซิงส์โครไนซ์ 10

3.1.2.2 รหัสคำหมายเลขเรียกขาน 10

3.1.2.3 รหัสคำข่าวสาร 12

3.1.2.3.1 รหัสคำของข่าวสารเพจเจอร์แบบตัวเลข 12

3.1.2.3.2 รหัสคำของข่าวสารเพจเจอร์แบบตัวอักษร-ตัวเลข 13

สารบัญ (ต่อ)

3.1.2.3.3 รหัสคำของข่าวสารเพจเจอร์ตัวอักษรภาษาไทย 17

3.12.4 รหัสคำว่าง 18

บทที่ 4 การออกแบบวงจรในลักษณะโครงสร้างและการบรรยายพฤติกรรม 19

4.1 การออกแบบวงจรดิจิตอล 19

4.2 ประวัติความเป็นมาของภาษาวีเอชดีแอล 21

4.3 ข้อกำหนดของภาษาวีเอชดีแอล 22

4.3.1 ลักษณะทั่วไป 22

4.3.2 สนับสนุนการออกแบบแบบลำดับขั้น 23

4.3.3 ไลบรารี 23

4.3.4 ลำดับคำสั่ง 23

4.3.5 การกำหนดคุณสมบัติ 23

4.3.6 ชนิดของข้อมูล 24

4.3.7 โปรแกรมย่อย 24

4.3.8 การควบคุมเวลา 24

4.3.9 การกำหนดแบบโครงสร้าง 24

4.4 องค์ประกอบพื้นฐานของวีเอชดีแอล 24

4.4.1 การกำหนดการเชื่อมต่อ 25

4.4.2 การกำหนดรูปแบบการบรรยาย 26

4.4.3 หน่วยการออกแบบแพ็กเกจ 27

4.4.3.1 Package Declartion 27

4.4.3.2 Package Body 28

4.4.4 หน่วยการออกแบบ Configuration 28

4.4.5 โปรแกรมย่อย 29

4.4.6 โอเปอร์เรเตอร์ 30

4.4.7 เวลาและความพร้อมเพรียง 31

4.4.8 สัญญาณและตัวแปร 31

4.5 การบรรยายเชิงพฤติกรรม 31

4.6 โปรเซส 31

สารบัญ (ต่อ)

4.7 การกำหนดตัวดำเนินการภายในโปรเซส 32
4.8 การกำหนดการกระทำภายในโปรเซส 33
4.9 การออกแบบจากบนลงล่าง 34
4.10 โครงสร้างของ FPGA ตระกูล Virtex-E 37
4.10.1 สถาปัตยกรรมของ FPGA ในตระกูล Virtex-E 37
4.10.1.1 โครงสร้างของ Virtex –E Array 37
4.10.1.2 กลุ่มของขา IOB 39
4.10.1.3 ขาสัญญาณเมื่อเป็นอินพุท 39
4.10.1.4 ขาสัญญาณเมื่อเป็นเอาท์พุท 40
4.10.1.5 กลุ่มของขาสัญญาณ I/0 Bangking 40
4.10.2 โครงสร้างภายในของชิฟ Virtex-E 41
4.10.2.1 โครงสร้างของ Configurable Logic Block 41
4.10.2.2 โครงสร้างของหน่วยความจำภายใน 43
4.10.2.3 โครงข่ายการเชื่อมต่อของขาสัญญาณอินพุตเอาท์พุท 43
4.10.2.4 โครงข่ายการเชื่อมต่อภายในแบบพิเศษ 44
4.10.2.5 โครงข่ายการเชื่อมต่อของสัญญาณนาฬิกา 44
4.10.2.6 การกระจายของสัญญาณนาฬิกาหลัก 44
4.10.7.7 Digtial Delay – Lock Loop 45
4.11 ปัจจัยที่ทำให้การออกแบบเอฟพีจีเอทำได้ง่ายและสะดวกรวดเร็ว 46
4.12 การออกแบบโดยใช้ภาษาอธิบายพฤติกรรมของฮาร์ดแวร์ 47
4.12.1 การสังเคราะห์วงจร (Logic Synthesis) 47
4.12.2 การแบ่งวงจร (Partitioning) 48
4.12.3 การวางอุปกรณ์(Placement) 48
4.12.4 การเชื่อมต่อสัญญาณ(Routing) 49
4.12.5 ความล่าช้าของเวลา(Delay) 49
4.12.6 การจำลองการทำงานของวงจร(Simulation).....49
4.12.7 การโปรแกรมอุปกรณ์เอฟพีจีเอ(Configuration).....50
4.13 เครื่องมือสำหรับการออกแบบด้วยอุปกรณ์เอฟพีจีเอ 51

สารบัญ (ต่อ)

บทที่ 5 การพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว โดยใช้ FPGA 52

5.1 ลักษณะโครงสร้างโดยรวมของระบบ 52

5.2 ภาครับสัญญาณวิทยุติดตามตัว(RF Receiver) 53

5.3 ภาคถอดรหัสเพจเจอร์ (Decoder Codeword) 55

5.3.1 บล็อกไดอะแกรมภาคถอดรหัสเพจเจอร์ 55

5.3.2 วงจรการทำงานภาคถอดรหัสเพจเจอร์ 56

5.4 ภาคถอดรหัสหมายเลขเรียกขาน (Address Codeword) 58

5.4.1 บล็อกไดอะแกรมภาคถอดรหัสหมายเลขเรียกขาน 58

5.4.2 วงจรภาคถอดรหัสหมายเลขเรียกขาน 59

5.5 ภาคควบคุมการทำงาน (Pager Controller) 60

5.6 ภาคจัดเรียงข้อมูล (Data Bufferr) 61

5.6.1 บล็อกไดอะแกรมภาคจัดเรียงข้อมูล 61

5.6.2 วงจรภาคจัดเรียงข้อมูล 62

5.7 ภาคแสดงผล (Lcd Graphic Display) 63

5.7.1 สัญญาณควบคุมการทำงานของ LCD Graphic Display 63

5.7.2 ภาค LCD Graphic Interface 64

5.7.2.1 บล็อกไดอะแกรมภาค LCD Graphic Interface 64

5.7.2.2 วงจรภาค LCD Graphic Interface 65

5.8 ภาคเชื่อมต่ออุปกรณ์แสดงผล (Display Device Interface) 67

5.9 วงจรกู้สัญญาณนาฬิกา(CLK Discovery) 67

5.9.1 หลักการและวงจรกู้สัญญาณนาฬิกา 67

บทที่ 6 การทดลอง 69

6.1 การทดสอบโดยการจำลองการทำงาน(Simulation) 69

6.1.1 ภาคถอดรหัสข้อมูลเพจเจอร์(Decoder Codeword) 69

6.1.2 ภาคถอดรหัสหมายเลขเรียกขาน(Address Codeword Decoder) 71

6.1.3 ภาคจัดเรียงข้อมูล(Data Buffer) 72

6.1.4 ภาคเชื่อมต่อวงจรแสดงผล(LCD Graphic Display Interface).....73

6.2 การทำ Implementation บน FPGA รุ่น Virtex-E เบอร์ XCV1600E BG560-6 74

สารบัญ (ต่อ)

6.3 ทดสอบการทำงานจริงของวงจร 76

6.3.1 ภาครับสัญญาณวิทยุ(VHF Receiver) 77

6.3.2 ภาคกู้สัญญาณนาฬิกา(Clk Discovery) 78

6.3.2.1 วงจรหาการเปลี่ยนของขอบสัญญาณ 78

6.3.2.2 สัญญาณนาฬิกาที่กู้กลับคืน 79

6.3.3 ฮาร์ดแวร์สำหรับทดสอบการทำงาน.....80

6.3.4 ทดสอบการรับส่งข้อมูลเพจเจอร์ 81

บทที่ 7 สรุปผลการวิจัยและข้อเสนอแนะ 82

7.1 สรุปผลการวิจัย 82

7.2 อุปสรรคที่พบในการวิจัย 83

7.3 ข้อเสนอแนะในการพัฒนา 83

เอกสารอ้างอิง 84

ภาคผนวก 85

ภาคผนวก ก เครื่องรับเพจเจอร์ในระบบ POCSAG 85

ภาคผนวก ข ภาคแสดงผล (LCD Graphic Interface) 87

ภาคผนวก ค ไฟล์ข้อมูลภาษา VHDL ของตัวถอดรหัสเพจเจอร์ 89

ภาคผนวก ง บทความวิจัยที่ได้รับการพิจารณาตีพิมพ์ 102

ประวัติผู้เขียน 108

สารบัญตาราง

ตารางที่ หน้า

3.1 รหัสคำการซิงส์โครไนซ์ 10

3.2 รหัสคำข่าวสาร 12

3.3 ชุดอักษรของข่าวสารเพจเจอร์รุ่นตัวเลข 12

3.4 ชุดอักษรของข่าวสารเพจเจอร์รุ่นตัวอักษร-ตัวเลข 14

3.5 ชุดอักษรของข่าวสารเพจเจอร์ตัวอักษรภาษาไทย(มาตรฐาน สมอ.) 17

3.6 รูปแบบของรหัสคำเทียม 18

4.1 สมรรถนะของวงจรพื้นฐานใน Virtex –E 38

6.1 จำนวนองค์ประกอบภายใน XCV1600E BG560-6 ที่ถูกใช้ในการทำ Implementaion 74

6.2 ค่าเวลาหน่วงที่เกิดขึ้นภายใน FPGA 74

6.3 ค่าเวลาหน่วงและความเร็วสูงสุดที่ระบบสามารถทำงานได้ 85

ข.1 ชุดคำสั่งของ LCD Graphic Interface 88

สารบัญรูป

รูปที่ หน้า

2.1 แสดงโครงสร้างของระบบวิทยุติดตามตัว 5

2.2 แสดงการทำงานของระบบอัตโนมัติ 7

2.3 แสดงการทำงานของระบบที่มีพนักงานรับฝากข้อความ 8

3.1 โครงสร้างรูปแบบของมาตรฐาน POCSAG 9

3.2 รหัสคำหมายเลขเรียกขาน 11

4.1 ขั้นตอนการออกแบบระบบดิจิตอล 19

4.2 การออกแบบระบบเส้นทางของข้อมูล 20

4.3 การกำหนดการเชื่อมต่อและสถาปัตยกรรม 25

4.4 บล็อกไดอะแกรมและการบรรยายการเชื่อมต่อของ Clock_Componet 26

4.5 การบรรยายเชิงพฤติกรรมของ Mux 26

4.6 โครงสร้างทั่วไปของส่วนการประกาศแพ็กเกจ 27

4.7 โครงสร้างของบอดี้แพ็กเกจ 28

4.8 โครงสร้างโดยทั่วไปของหน่วยการออกแบบโครงแบบ 29

4.9 การใช้โพรซีเจอร์ 29

4.10 การใช้ฟังก์ชัน 30

4.11 ตัวดำเนินการในวีเอชดีแอล 30

4.12 รูปแบบของการบรรยายแบบโปรเซส 32

4.13 ตัวอย่างการประกาศตัวดำเนินการภายในโปรเซส 33

4.14 เงื่อนไขการกระทำในโปรเซส 33

4.15 การกระทำในโปรเซส 34

4.16 ขั้นตอนการออกแบบจากบนลงล่าง 35

4.17 สถาปัตยกรรมของ FPGA รุ่น Virtex-E 37

4.18 กลุ่มของ IOBของ Virtex – E 39

4.19 กลุ่มของ I/O แบงก์ภายใน Virtex- E 41

4.20 โครงสร้างแบบ 2 Slice CLB ของ Virtex-E 41

4.21 รายละเอียดภายใน Slice 42

4.22 โครงข่ายของการกระจายสัญญาณ Global Clock 45

4.23 การจัดวางตำแหน่งของ DLL 46

สารบัญรูป(ต่อ)

รูปที่ หน้า

4.24 การโปรแกรมลงในชิฟ 46

5.1 บล็อกไดอะแกรมของระบบตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว.....52

5.2 บล็อกไดอะแกรมภาครับสัญญาณวิทยุ (RF Receiver).....53

5.3 ภาครับสัญญาณวิทยุ(VHF Receiver) 54

5.4 บล็อกไดอะแกรมภาคถอดรหัสเพจเจอร์ 55

5.5 วงจรภาคถอดรหัสเพจเจอร์ 57

5.6 บล็อกไดอะแกรมภาคถอดรหัสหมายเลขเรียกขาน 58

5.7 วงจรภาคถอดรหัสหมายเลขเรียกขาน 59

5.8 บล็อกไดอะแกรมภาคควบคุมการทำงานของระบบ 60

5.9 บล็อกไดอะแกรมภาคจัดเรียงข้อมูล 61

5.10 วงจรจัดเรียงข้อมูล 63

5.11 สัญญาณควบคุม LCD Graphic Display ผ่านทาง 3 Bit Serial 64

5.12 บล็อกไดอะแกรมภาค LCD Graphic Interface 65

5.13 วงจรภาค LCD Graphic Interface 66

5.14 บล็อกไดอะแกรมภาคเชื่อมต่ออุปกรณ์แสดงผลการเรียกเพจเจอร์ 67

5.15 หลักการของวงจรกู้สัญญาณนาฬิกา 68

5.16 วงจรกู้สัญญาณนาฬิกา 68

6.1 แบบจำลองวงจรภาคถอดรหัสข้อมูล 70

6.2 ผลจำลองการทำงานของภาคถอดรหัสเพจเจอร์ในโหมดการเขียนข้อมูล 70

6.3 ผลจำลองการทำงานภาคถอดรหัสเพจเจอร์ในโหมดการอ่านข้อมูล 70

6.4 แบบจำลองภาคถอดรหัสหมายเลขเรียกขาน (Address Codeword) 71

6.5 ผลจำลองการทำงานภาคถอดรหัสหมายเลขเรียกขาน 71

6.6 แบบจำลองการทำงานภาคจัดเรียงข้อมูล(Data Buffer) 72

6.7 ผลจำลองการทำงานภาคจัดเรียงข้อมูล 72

6.8 แบบจำลองการทำงานภาคเชื่อมต่อวงจรแสดงผล 73

6.9 ผลจำลองการทำงานของภาคเชื่อมต่อวงจรแสดงผล 73

6.10 โครงสร้างการเชื่อมต่อภายในระหว่าง CLB 75

6.11 บล็อกไดอะแกรมทดสอบการทำงานจริงของวงจร 76

สารบัญรูป(ต่อ)

รูปที่ หน้า

6.12 สัญญาณเบสด์แบนด์(NRZ)จากภาครับสัญญาณวิทยุ 77
6.13 สัญญาณซิงส์โครไนซ์กับสัญญาณเบสด์แบนด์(NRZ) 78
6.14 สัญญาณนาฬิกาที่กู้กลับคืนมาเมื่อทดสอบกับสัญญาณ Square Wave ที่ความถี่ 1200 Hz 79
6.15 สัญญาณนาฬิกาที่กู้กลับคืนมาเมื่อเปรียบเทียบกับสัญญาณ NRZ ที่รับได้ 79
6.16 ฮาร์ดแวร์สำหรับทดสอบการทำงาน 80
6.17 ผลทดสอบการรับข้อมูลเพจเจอร์ 81
ก.1 บล็อกไดอะแกรมของเครื่องรับเพจเจอร์ 85
ก.2 บล็อกไดอะแกรมภาครับวิทยุติดตามตัว(RF PANEL) 85
ก.3 บล็อกไดอะแกรมภาค AF PANEL 86
ก.4 วงจรภาค AF PANEL 86
ข.1 ตัวแสดงผล(LCD Graphic Display) 88

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันนี้วิทยุติดตามตัว(PAGER) ซึ่งยังคงมีการใช้งานอยู่แต่จำนวนจะลดลงไปเรื่อยๆ เนื่องจากสาเหตุที่เป็นระบบสื่อสารทางเดียวเมื่อเทียบกับโทรศัพท์เคลื่อนที่(Mobile Telephone) ซึ่งเป็นระบบสื่อสารสองทิศทาง (Bidirectional Communication) ดังนั้นเราจึงเล็งเห็นว่าเทคโนโลยีของระบบวิทยุติดตามตัวน่าจะมีการนำไปประยุกต์ใช้ให้เกิดประโยชน์ในแนวทางอื่นๆได้ เพราะว่าเนื่องจากมีขอบเขตที่ให้บริการกระจายครอบคลุมพื้นที่ทั่วประเทศ ตลอดจนความน่าเชื่อถือของการส่งข้อมูลในระบบค่อนข้างสูง โดยการใช้งานเครื่องรับวิทยุติดตามตัวในปัจจุบันส่วนมากจะนำเข้ามาจากต่างประเทศและไม่สามารถที่จะพัฒนาตัวเครื่องเพื่อการนำไปประยุกต์ใช้งานในด้านต่างๆให้มีความหลากหลายเพราะยึดติดกับเทคโนโลยีของผู้ผลิตจำหน่ายดังนั้นวิทยานิพนธ์นี้จึงได้ทำการศึกษาและ พัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัวในระบบ POCSAG (Post Office Code Standardization Advisory Group) ซึ่งเป็นการให้บริการวิทยุติดตามตัวตามมาตรฐานของ CCIR Recommedation 584 ซึ่งจะมีมาตรฐานความเร็วในการส่งข้อมูลอยู่ที่ 512, 1200 และ 2400 บิตต่อวินาที ส่วนภาคการแสดงผลสามารถแสดงได้ทั้งภาษาไทยและภาษาอังกฤษอีกทั้งยังสามารถนำไปประยุกต์ใช้งานนอกจากการใช้งานปกติของวิทยุติดตามตัวในปัจจุบันได้อีกเช่น การนำไปประยุกต์ใช้งานในงานป้องกันโจรกรรมรถยนต์ การรับข่าวสารเพื่อกระจายให้กับบุคลากรภายในองค์กร การแจ้งเหตุเสียในระบบโทรคมนาคมผ่านวิทยุติดตามตัว การควบคุมอุปกรณ์ไฟฟ้า และระบบเตือนภัยผ่านโครงข่ายวิทยุติดตามตัว เป็นต้น

1.2 วัตถุประสงค์ของการวิจัย

1.2.1 เพื่อพัฒนาต้นแบบของตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัวขึ้นมาเพื่อที่จะนำมาใช้งานได้จริง

1.2.2 เพื่อพัฒนาเป็นเทคโนโลยีของตนเอง ในด้านการออกแบบวงจรเฉพาะงานขึ้นมา หรือนำคอร์(Core) ที่ได้ไปใช้ในการออกแบบ ASIC (Application Specific Integrated Circuit) ต่อไป

1.2.3 เพื่อลดจำนวนฮาร์ดแวร์ให้มีขนาดเล็กลงตลอดจนความเร็วในการทำงานที่เร็วขึ้น

1.2.4 สามารถนำไปประยุกต์ใช้งานได้จริง

1.3 ขอบเขตของการวิจัย

เพื่อพัฒนาต้นแบบตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว ซึ่งสามารถแสดงผลได้ทั้งภาษาอังกฤษ และภาษาไทย โดยพัฒนาเป็นชิฟเฉพาะงานในส่วนการทำงานของวงจรดิจิตอล โดยใช้ภาษาวีเอชดีแอล(VHDL)ในการบรรยายอธิบายการทำงานของวงจร และโปรแกรมลงบนอุปกรณ์ เอฟพีจีเอ(FPGA) ซึ่งการพัฒนาการออกแบบวงจรแบบนี้จะมีความยืดหยุ่นในการพัฒนาวงจรให้มีความรวดเร็วในการทำวิจัยและพัฒนาเป็นเครื่องต้นแบบ ตลอดจนความเร็วในการทำงานของวงจรที่สูง ดังนั้นวิทยานิพนธ์นี้จะทำการออกแบบเป็นวงจรรวมให้การทำงานทั้งหมดอยู่ภายในชิฟตัวเดียว

1.4 เปรียบเทียบกับหลักการที่มีอยู่เดิม

แต่เดิมการพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัวจะออกแบบวงจรที่จะเป็นลักษณะ Discrete Component โดยมีตัวควบคุมการทำงานหลักเป็นไมโครโปรเซสเซอร์และอุปกรณ์รอบข้างในการทำงาน ซึ่งวิธีการออกแบบวิธีนี้จะต้องยึดติดกับโครงสร้างกับชิฟที่มีขายจากตัวแทนจำหน่าย ซึ่งจะมีปัญหาในการพัฒนาวงจรต่อไปในอนาคต อาจเนื่องมาจากการหยุดผลิตของผู้ให้จำหน่ายหรืออาจมาจากสาเหตุอื่นๆในการออกแบบวงจร ซึ่งถ้าเทียบกับการออกแบบสมัยใหม่ในการออกแบบวงจรและการพัฒนาโดยใช้ภาษาวีเอชดีแอล(VHDL) เป็นภาษาที่บรรยายพฤติกรรมการทำงานของวงจรดังนั้นเราสามารถที่จะสร้างฟังก์ชันวงจรอะไรก็ได้เพื่อให้วงจรทำงานตามที่เราต้องการ ซึ่งจะเห็นได้ว่าแนวทางการพัฒนาวงจรมีความยืดหยุ่นในการทำงานที่สูงเมื่อเทียบกับวิธีเก่า ซึ่งทดสอบการทำงานโดยโปรแกรมลงชิฟเอฟพีจีเอ(FPAG) ซึ่งทำให้ได้วงจรที่มีขนาดเล็กลง ตลอดจนการพัฒนาวงจรจะทำได้สะดวกและรวดเร็วซึ่งเหมาะกับพัฒนาเป็นเครื่องต้นแบบโดยมีต้นทุนในการวิจัยและพัฒนาที่ต่ำ

1.5 โครงสร้างของวิทยานิพนธ์

วิทยานิพนธ์นี้เป็นส่วนของงานเอกสารหลักของงานวิจัยนี้ซึ่งจะกล่าวถึงรายละเอียดต่างๆ ดังนี้

บทที่ 1 บทนำ

กล่าวถึงปัญหาและความเป็นมาของการออกแบบตัวถอดรหัสเพจเจอร์ในระบบ POCSAG สำหรับวิทยุติดตามตัวโดยใช้เอฟพีจีเอ วัตถุประสงค์และขอบเขตของการวิจัย

บทที่ 2 ทฤษฎีพื้นฐานที่เกี่ยวข้องกับระบบวิทยุติดตามตัว

กล่าวถึงทฤษฎีพื้นฐานของโครงข่ายวิทยุติดตามตัว ตลอดจนโครงสร้างพื้นฐานและการทำงานของระบบวิทยุติดตามตัว

บทที่ 3 โครงสร้างและรหัสมาตรฐาน POCSAG

กล่าวถึงรูปแบบรหัสและมาตรฐานการรับส่งข้อมูลในระบบ POCSAG โดยจะกล่าวรายละเอียดในลักษณะดังต่อไปนี้คือ สัญญาณพัลส์ส่วนหน้า,โครงสร้างและมาตรฐาน POCSAG โครงสร้างของรหัสคำซึ่งได้แก่ รหัสคำการซิงโครไนซ์,รหัสคำหมายเลขเรียกขาน, รหัสคำข่าวสาร เพจเจอร์แบบตัวอักษร-ตัวเลข และแบบอักษรภาษาไทย , รหัสคำเทียม

บทที่ 4 การออกแบบวงจรในลักษณะโครงสร้างและการบรรยายพฤติกรรม

กล่าวถึงการออกแบบวงจรโดยการอธิบายลักษณะพฤติกรรมของวงจรประวัติความเป็นมาของภาษาวีเอชดีแอล ข้อกำหนดต่างๆ และโครงสร้างภายในชิฟเอฟพีจีเอแบบต่างๆ

บทที่ 5 การออกแบบและพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัวโดยใช้เอฟพีจีเอ

กล่าวถึงรายละเอียดของการออกแบบวงจรภายในชิฟเอฟพีจีเอ และขั้นตอนต่างๆของการออกแบบระบบบนชิฟเอฟพีจีเอ

บทที่ 6 ผลการทดลอง

กล่าวถึงการทดสอบการทำงานของตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว แบ่งการทดสอบออกเป็น 2 ส่วนคือ ทดสอบโดยการจำลองการทำงาน (Simulation)และทดสอบการทำงานจริงของวงจร

บทที่ 7 สรุปผลการวิจัยและข้อเสนอแนะ

เป็นการสรุปผลการดำเนินงานทั้งหมด ปัญหาต่างๆที่เกิดขึ้น การแก้ไขปัญหาต่างๆ พร้อมทั้งเสนอแนวทางการพัฒนางานวิจัยต่อไป

ภาคผนวก ก เครื่องรับเพจเจอร์ในระบบ POCSAG

ภาคผนวก ข ภาคแสดงผล(LCD Graphic Display)

ภาคผนวก ค ไฟล์ข้อมูลภาษา VHDL ของตัวถอดรหัสเพจเจอร์

ภาคผนวก ง แสดงบทความวิจัยที่ได้รับการพิจารณาตีพิมพ์

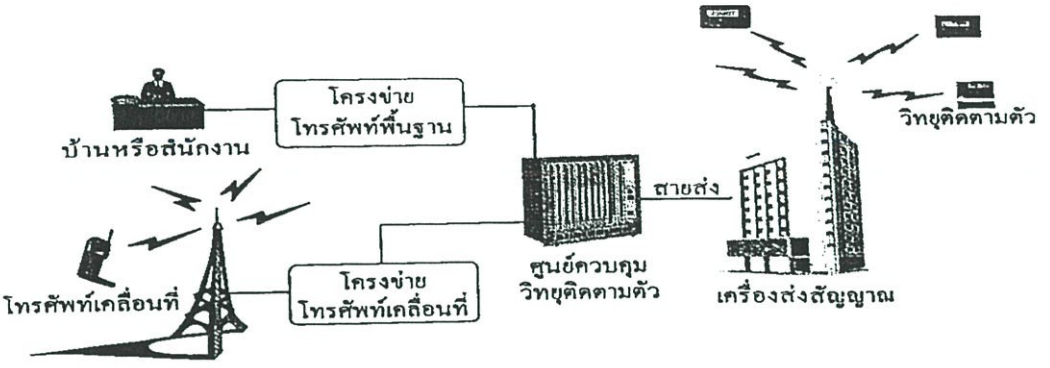
บทที่ 2
ทฤษฎีพื้นฐานที่เกี่ยวข้องกับระบบวิทยุติดตามตัว

วิทยุติดตามตัวหรือเพจเจอร์ เป็นอุปกรณ์สื่อสารแบบเคลื่อนที่ซึ่งมีขนาดเล็ก น้ำหนักเบา สะดวกต่อการพกพาติดตามตัวช่วยอำนวยความสะดวกสบายในการติดต่อสื่อสาร และได้รับความนิยมอย่างกว้างขวาง โดยที่มีลักษณะการติดต่อสื่อสารแบบทิศทางเดียวจากผู้ส่งไปยังผู้รับ โดยในการติดต่อกันนั้นส่วนที่มีความสำคัญมากก็คือโปรโตคอล (Protocal) หรือภาษาที่ใช้ในการสื่อสารกันของระบบโดยจะเป็นตัวกำหนดรูปแบบของการเข้ารหัสสัญญาณ เพื่อที่เครื่องรับวิทยุติดตามตัวจะได้เข้าใจและสามารถอดรหัสสัญญาณได้อย่างถูกต้องนั่นก็คือโปรโตคอลที่ใช้ในการส่งจากระบบวิทยุติดตามตัวและตัวเครื่องรับวิทยุติดตามตัวจะต้องใช้โปรโตคอลชนิดเดียวกัน ในปัจจุบันมีการให้บริการระบบวิทยุติดตามตัวอยู่ 2 ระบบดังนี้คือ ระบบ POCSAG(Post Office Code Standardization Advisory Group) และระบบ FLEX จะสามารถรองรับได้ถึง 2 ล้าน Address ต่อความถี่ สามารถส่งได้ทั้งเสียง(Tone) ตัวเลข(Numeric) ตัวอักษร(Alphanumeric) และตัวอักษรภาษาไทย(Thai) โดยที่ระบบ POCSAG จะมีมาตรฐานความเร็วในการรับส่ง อยู่ที่ 512 ,1200 และ 2400 บิตต่อวินาที ในระบบ POCSAG ซึ่งจะทำการกำหนดวิธีการติดต่อกันระหว่างศูนย์ควบคุมวิทยุติดตามตัวกับเครื่องวิทยุติดตามตัวเท่านั้น แต่จะไม่ได้ทำการกำหนดความถี่ที่ใช้ในการติดต่อกัน ดังนั้นโครงข่ายวิทยุติดตามตัวของแต่ละผู้ให้บริการจะสามารถเลือกใช้ความถี่ใดก็ได้ตามต้องการ แต่ก็มีข้อเสีย คือ การเชื่อมต่อระหว่างโครงข่ายวิทยุติดตามตัว หรือการใช้เครื่องลูกข่ายข้ามระบบ (Roaming)เป็นไปได้ยาก และการให้บริการก็มีความแตกต่างกันไป ทำให้กลายเป็นข้อเสียของระบบนี้ ส่วนระบบ FLEX เป็นระบบใหม่ที่เพิ่งถูกพัฒนาขึ้นมาประมาณปี 1993 โดยบริษัทโมโตโรลาจะมีความเร็วในการส่งข้อมูล 1600,3600,6400 บิตต่อวินาที มีการส่งสัญญาณแบบซิงส์โครไนซ์ (Synchronous) ระหว่างโครงข่ายกับตัวเครื่องวิทยุติดตามตัว ซึ่งต่างจากระบบ POCSAG ที่มีการส่งสัญญาณแบบอะซิงส์โครไนซ์ (Asynchronous)

2.1 โครงสร้างพื้นฐานและการทำงานของระบบวิทยุติดตามตัว[1]

ระบบวิทยุติดตามตัวจะมีลักษณะการทำงานดังนี้ เมื่อผู้ส่งข่าวสารต้องการฝากข่าวสารก็จะทำการโทรไปยังศูนย์ควบคุมวิทยุติดตามตัวเพื่อฝากข่าวสาร ซึ่งอาจจะทำการโทรผ่านโครงข่ายโทรศัพท์พื้นฐาน หรือโครงข่ายโทรศัพท์เคลื่อนที่ก็ได้ และสามารถฝากได้ทั้งตัวเลข ตัวอักษร หรือสัญญาณเสียง เมื่อศูนย์ควบคุมวิทยุติดตามตัวได้รับข่าวสารก็จะทำการส่งข่าวสาร โดยผ่านเครื่องส่ง

สัญญาณไปยังเครื่องรับวิทยุติดตามตัวของผู้รับระบบวิทยุติดตามตัวที่ใช้งานในปัจจุบันนี้ โครงสร้างการทำงานของระบบวิทยุติดตามตัว แสดงดังรูปที่ 2.1



รูปที่ 2.1 แสดงโครงสร้างของระบบวิทยุติดตามตัว

2.1.1 ศูนย์ควบคุมวิทยุติดตามตัว (Paging Control Terminal)

ศูนย์ควบคุมวิทยุติดตามตัวเป็นศูนย์กลางของระบบ ทำหน้าที่รับข่าวสารจากผู้ฝากข้อความและทำการจัดส่งข้อความไปยังผู้รับโดยเปลี่ยนข่าวสารให้เป็นสัญญาณข้อมูลแล้วส่งไปยังสถานีฐานวิทยุติดตามตัว นอกจากนี้ยังจัดการเกี่ยวกับการให้บริการและหน้าที่พิเศษอื่นๆอีกเช่น การเก็บบันทึกการใช้งาน การออกบิลเรียกเก็บเงิน การสอบถามการใช้บริการ เป็นต้น

2.1.2 ระบบโทรศัพท์ (Telephone)

เป็นส่วนที่ผู้ส่งข่าวสารใช้ทำการส่งข่าวสารไปยังศูนย์บริการวิทยุติดตามตัว เพื่อทำการฝากข้อความหรือหมายเลขโทรกลับไปยัง ผู้รับที่ต้องการติดต่อ โดยอาจจะทำการโทรมาจากระบบโทรศัพท์พื้นฐานหรือจากระบบโทรศัพท์เคลื่อนที่

2.1.3 ส่วนส่งสัญญาณ (Transmitter)

เป็นส่วนของเครื่องส่งที่ทำหน้าที่นำเอาข้อมูลที่ได้รับมาจากศูนย์ควบคุมวิทยุติดตามตัวมาแปลงให้เป็นสัญญาณวิทยุแล้วจะทำการกระจายข่าวสารนั้นออกไปยังเครื่องรับวิทยุติดตามตัว โดยที่เครื่องส่งอาจจะติดตั้งอยู่ในที่เดียวกับศูนย์กลางของระบบก็ได้ หรือติดตั้งในที่ที่ต้องการให้บริการในขณะเดียวกันยังส่งสัญญาณติดต่อกับสถานีฐานวิทยุติดตามตัวอื่น เพื่อความสอดคล้องของข้อมูล

2.2 เครื่องรับวิทยุติดตามตัว(Pager)

วิทยุติดตามตัวหรือเพจเจอร์เป็นอุปกรณ์สื่อสารแบบพกพาชนิดหนึ่งเช่นเดียวกับโทรศัพท์เคลื่อนที่ ผิดกันแต่เพียงว่ารูปแบบในการสื่อสารข้อมูลของเพจเจอร์จะเป็นลักษณะของการฝากข่าวสารไว้ที่ศูนย์รับฝากข้อความโดยศูนย์ฯจะทำหน้าที่ส่งข่าวสารนั้นไปยังเครื่องลูกข่ายที่ถูกระบุหมายเลขไว้ หลายท่านเข้าใจว่าเครื่องลูกข่ายมีหน้าที่รับข่าวสารซึ่งถูกส่งจากศูนย์ฯเท่านั้น ในความเป็นจริงแล้วเพจเจอร์บางระบบมีการกำหนดความสามารถของเครื่องลูกข่ายให้ส่งข่าวสารผ่านกลับมาให้ศูนย์ฯ ได้ สำหรับข่าวสารที่สามารถส่งผ่านเครือข่ายเพจเจอร์นั้นมีตั้งแต่ตัวเลข ตัวอักษรและเสียงพูด ทั้งนี้การจะใช้บริการส่งข่าวสารประเภทใด ต้องพิจารณาถึงความสามารถในการรับข่าวสารของเครื่องลูกข่ายด้วย รูปแบบของเครื่องลูกข่ายสามารถแบ่งออกเป็นความสามารถในการรับข่าวสารได้เป็น 5 แบบ คือ

2.2.1 แบบฝากเสียงพูด (Voice Pager)

อีกรูปแบบหนึ่งของเครื่องลูกข่ายโดยผู้ฝากข่าวสารสามารถฝากเสียงพูดของตนผ่านทางคู่สายโทรศัพท์ ไปเก็บยังศูนย์รับฝากข้อความเพื่อให้ศูนย์ฯ ส่งข้อความนั้นไปยังเจ้าของเครื่อง

2.2.2 แบบตัวเลข (Numeric Pager)

ข่าวสารที่รับได้เฉพาะตัวเลข โดยเก็บตัวเลขได้สูงสุด 20 หลัก ต่อหนึ่งข่าวสาร ตัวเลขเหล่านี้อาจจะเป็นหมายเลขโทรศัพท์ที่จะให้ติดต่อกลับ ราคาสินค้า รหัสสินค้า หรือรหัสพิเศษเฉพาะกลุ่ม แล้วแต่ผู้ใช้บริการ

2.2.3 แบบตัวอักษร (Alpha-Numeric and Thai Pager)

เป็นแบบที่ได้รับความนิยมอย่างสูงในบ้านเรา เครื่องลูกข่ายประเภทนี้สามารถรับข้อความได้เต็มที่ 200 ตัวอักษรและแสดงผลได้หน้าละ 80 ตัวอักษรต่อหนึ่งข่าวสาร โดยตัวอักษรในที่นี้หมายถึงตัวอักษรอังกฤษ หากเป็นตัวอักษรภาษาไทยจะได้จำนวนน้อยกว่านี้หน่วยความจำสำรองสามารถเก็บข่าวสารได้ 40 ชุด โดยผู้ใช้สามารถสั่งลบหรือป้องกันการลบได้ด้วยตนเองบริการพิเศษซึ่งได้รับความนิยมมากสำหรับเครื่องลูกข่ายประเภทนี้คือการให้บริการข่าวสารแบบออนไลน์ เช่น อัตราแลกเปลี่ยนการเงิน ข่าวการเมือง ข่าวกีฬา หรืออื่นๆเป็นต้น

2.2.4 แบบใช้เสียง (Tone Pager)

เป็นเครื่องลูกข่ายที่มีราคาถูกที่สุดโดยจะส่งเสียงเตือนเมื่อมีการติดต่อไปยังเครื่องหมายเลขนั้น ทั้งนี้อาจใช้เป็นการเตือนให้เจ้าของเครื่องโทรติดต่อกลับศูนย์ฯ หรือ ใช้เป็นการเตือนให้โทรกลับไปยังหมายเลขใด หมายเลขหนึ่งในกรณีที่มีการตกลงกันล่วงหน้า

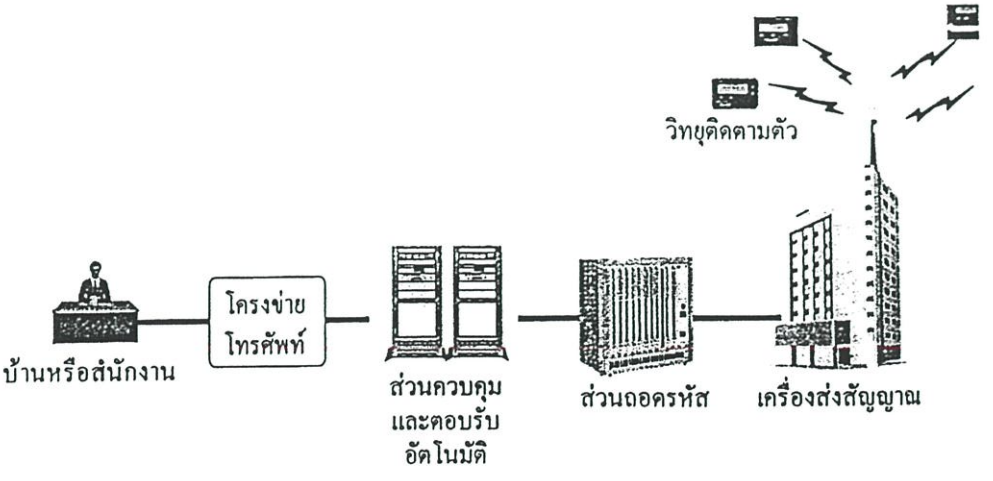
2.2.5 แบบหลายเสียง (Multi Address Pager)

จะทำส่งสัญญาณเสียงเหมือนกัน Tone-Alert Pager แต่จะให้สัญญาณเสียงได้ 2 ลักษณะ เพื่อให้ผู้รับทราบว่าจะติดต่อไปยังที่ใดเวลาที่ใช้ในการส่งข่าวสารด้วยเพจเจอร์ในแต่ละแบบจะแตกต่างกันคือ เพจเจอร์แบบใช้เสียงพูดจะใช้เวลาตั้งแต่ 10 วินาทีลงมา ส่วนเพจเจอร์แบบที่ไม่ใช้เสียงพูดจะใช้เวลาในการส่งเป็นมิลลิวินาที

2.3 รูปแบบการฝากข้อความของระบบวิทยุติดตามตัวจะสามารถแบ่งออกได้เป็น 3 ชนิด

2.3.1 ระบบอัตโนมัติ (Automatic paging system)

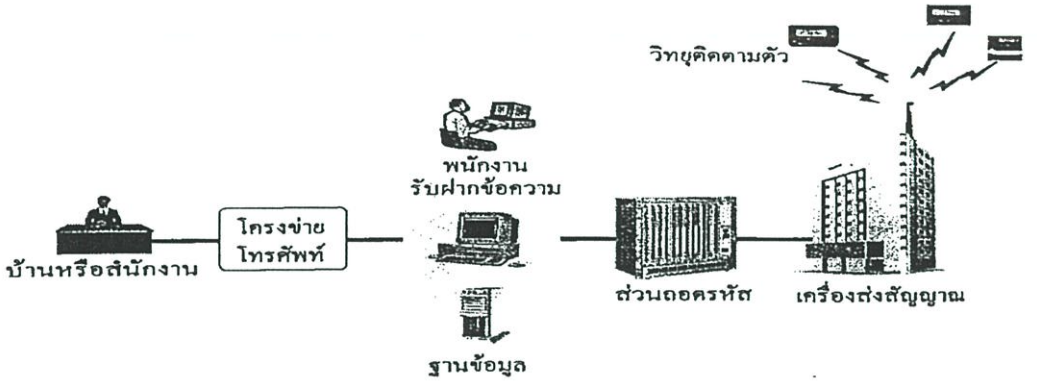
เป็นระบบที่ผู้ส่งข่าวสารสามารถโทรศัพท์เข้าไปที่ศูนย์วิทยุติดตามตัวเพื่อทำการฝากข้อมูลต่างๆโดยจะมีระบบอัตโนมัติทำหน้าที่รับฝากข้อมูล และทำการส่งข้อมูลไปยังเครื่องรับวิทยุติดตามตัวที่ต้องการได้โดยอัตโนมัติซึ่งในประเทศไทยจะมีการให้บริการอยู่สองแบบด้วยกันก็คือ การฝากเลขหรือเบอร์โทรที่ต้องการให้ติดต่อกลับ และการฝากสัญญาณเสียงโดยผู้รับจะต้องโทรกลับไปเช็คข้อความที่ศูนย์บริการอีกทีหนึ่ง



รูปที่ 2.2 แสดงการทำงานของระบบอัตโนมัติ

2.3.2 ระบบที่มีพนักงานรับฝากข้อความ(Manually operator system)

ในระบบนี้จะมีพนักงานทำหน้าที่รับฝากข้อมูลข่าวสารต่างๆจากการโทรศัพท์เข้ามาของผู้ใช้บริการ โดยลักษณะของข่าวสารที่ทำการรับฝากนั้นจะเป็นตัวเลขและตัวอักษร หลังจากนั้น ก็ทำการส่งออกข้อมูลที่ได้ออกไปยังเครื่องรับวิทยุติดตามตัว



รูปที่ 2.3 แสดงการทำงานของระบบที่มีพนักงานรับฝากข้อความ

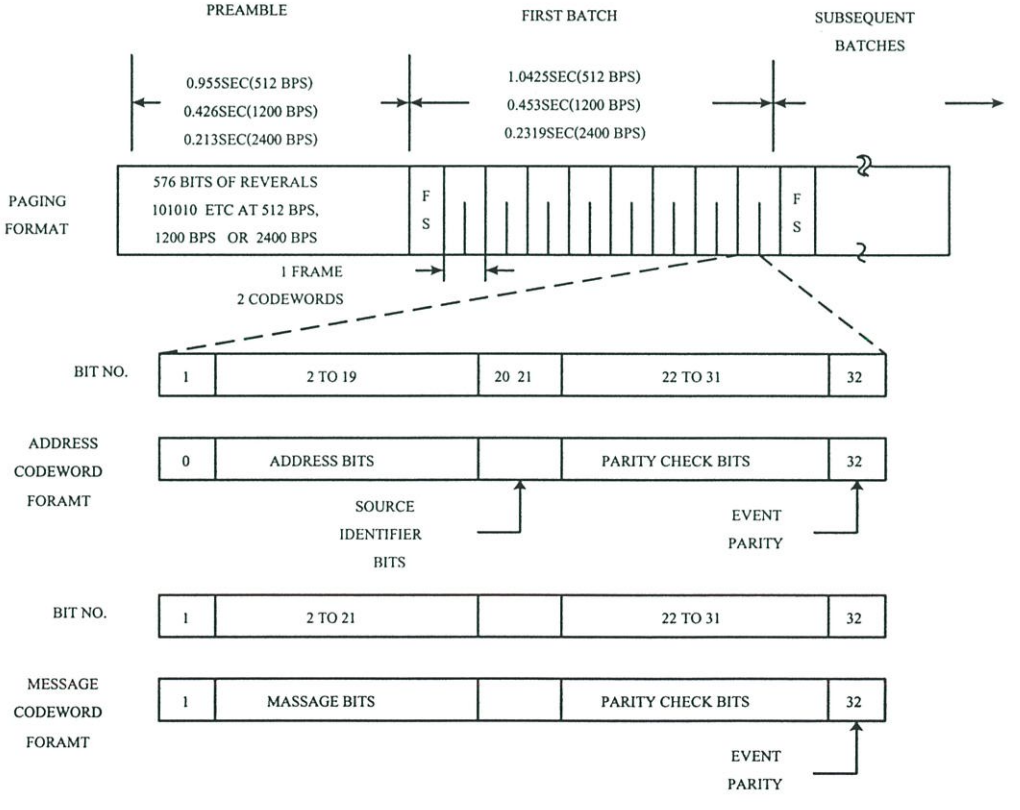
2.3.3 การฝากข้อความผ่านอินเทอร์เนต

การฝากข้อความในระบบนี้ เป็นการให้บริการแบบใหม่ของผู้ให้บริการต่างๆ โดยผู้ส่งสามารถส่งข้อความได้โดยไม่ต้องผ่านพนักงานรับฝากข้อความ แต่มักจะพบว่ามีปัญหาในระบบนี้มาก โดยเมื่อทำการส่งข้อความแล้วทางผู้รับไม่สามารถรับข้อความได้ หรือรับข้อความที่ส่งมาได้แต่อ่านข้อความที่ส่งมาไม่ได้

บทที่ 3
โครงสร้างและรหัสมาตรฐาน POCSAG

3.1 รูปแบบของสัญญาณเพจเจอร์[2]

สัญญาณข่าวสารที่จะส่งไปยังผู้รับนั้นเป็นสัญญาณรหัสข้อมูลเลขฐานสองเรียงต่อเนื่องกัน ซึ่งรหัสข้อมูลที่ใช้เป็นรหัสของ POCSAG (Post Office Code Standardization Advisory Group) ซึ่งเป็นรูปแบบและรหัสมาตรฐานสำหรับวิทยุติดตามตัวที่ถูกกำหนดโดย CCIR รูปแบบของสัญญาณข่าวสารที่ใช้ส่งประกอบไปด้วย สัญญาณพัลส์ส่วนหน้า(Preamble)จำนวน 576 บิต และตามด้วยรหัสคำตั้งแต่หนึ่งชุดขึ้นไป(Batch Structure) ซึ่งแต่ละชุดจะเริ่มต้นด้วยรหัสคำการซิงส์โครไนซ์(Synchronization Codeword) ตามด้วยรหัสคำเทียม(Idle Codeword) รหัสคำหมายเลขเรียกขาน(Address Codeword) และรหัสคำข่าวสาร(Message Codeword) ถ้าข้อมูลที่ส่งไม่เต็ม 1 ชุด (Batch Structure)จะส่งรหัสคำเทียมต่อท้ายมาจนเต็ม 1 ชุดรูปแบบของสัญญาณแสดงได้ดังรูปที่ 3.1



รูปที่ 3.1 โครงสร้างรูปแบบของมาตรฐาน POCSAG

3.1.1 สัญญาณพัลส์ส่วนหน้า (Preamble)

ประกอบด้วยรูปแบบที่มีลักษณะของบิตตรงข้ามกัน คือ "1010.... 1010" จะถูกส่งเป็นจำนวนอย่างน้อย 576 บิต(มีค่าเท่ากับ 1 Batch บวก 1 Codeword) สัญญาณพัลส์ส่วนหน้านี้ถูกใช้เป็นตัวช่วยในการตรวจหาจุดเริ่มต้นของการส่งเพจเจอร์

3.1.2 โครงสร้างของแบตซ์ข้อมูล (Batch Structure)

รหัสคำจะถูกส่งลงในแบตซ์ ซึ่งในแต่ละแบตซ์จะเริ่มด้วยรหัสคำการซิงส์โครไนซ์ และตามด้วยเฟรมของรหัสคำอีก 8 เฟรม โดยที่เฟรมแรกเป็นเฟรม 0 และเฟรมสุดท้ายเป็นเฟรมที่ 7 ซึ่งลักษณะการส่งข้อมูลก็จะส่งเป็นแบตซ์ข้อมูลไปเรื่อยๆจนสิ้นสุดการส่งข้อมูล รหัสคำที่บรรจุอยู่ในแบตซ์แบ่งออกได้เป็น 4 ชนิดดังนี้

3.1.2.1 รหัสคำการซิงส์โครไนซ์ (Synchronization Codeword)

เป็นรหัสคำที่มีรูปแบบเฉพาะเป็นเอกลักษณ์และไม่มีการซ้ำจากรหัสคำอื่นๆซึ่งมีขนาด 32 บิต รหัสคำชนิดนี้จะอยู่เป็นอันดับแรกของแต่ละแบตซ์จึงทำหน้าที่กำหนดจุดเริ่มต้นของแบตซ์ในการส่งข้อมูลมายังเครื่องรับเพจเจอร์ โดยมีรูปแบบของบิตดังตารางที่ 3.1

ตารางที่ 3.1 รหัสคำการซิงส์โครไนซ์

LSB MSB

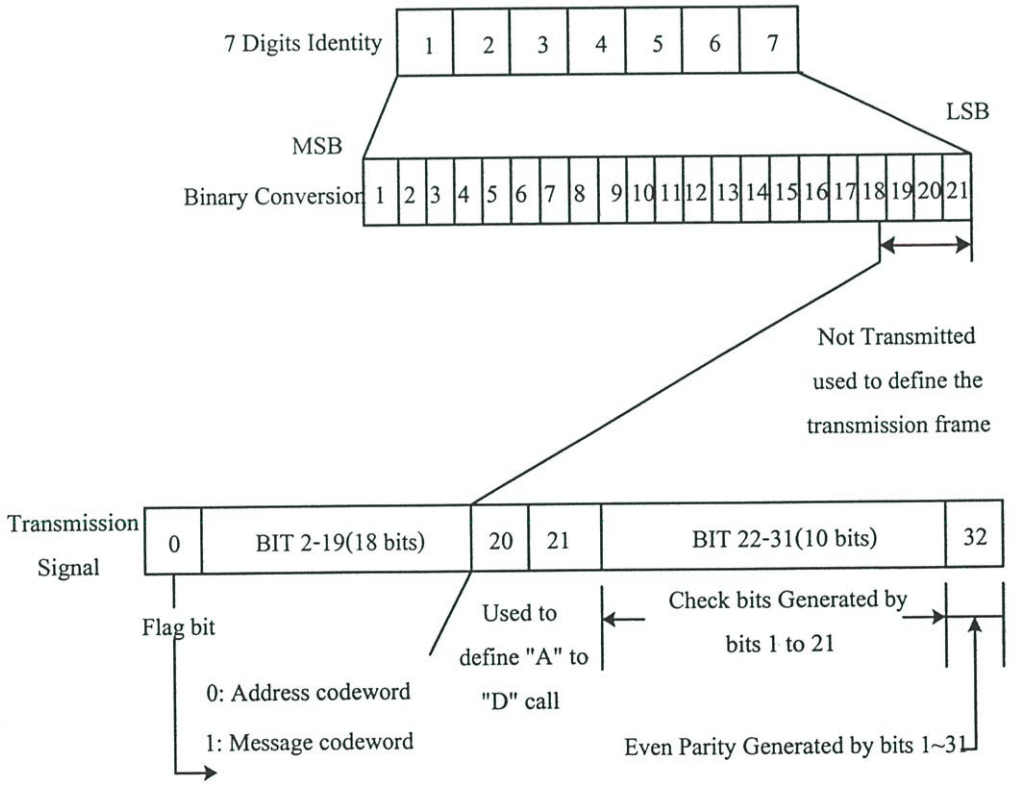
0	1	1	1	1	1	0	0	1	1	0	1	0	0	1	0	0	0	0	1	0	1	0	1	1	1	0	1	1	0	0	0

Synchronization Bits

3.1.2.2 รหัสคำหมายเลขเรียกขาน (Address Codeword)

รหัสคำชนิดนี้ประกอบไปด้วยเลขฐานสองจำนวน 32 บิต ซึ่งบิตแรกเป็น '0' เสมอรูปแบบรหัสจะมีบิตที่ 2 ถึงบิตที่ 19 เป็นรหัสหมายเลขเฉพาะที่กำหนดให้กับเพจเจอร์แต่ละเครื่อง ซึ่งถูกแปลงจากเลขฐานสิบจำนวน 7 หลักมาเป็นเลขฐานสองจำนวน 21 บิต ซึ่งเลขฐานสองที่ถูกแปลงมานั้น จะใช้เพียง 18 บิตที่มีความสำคัญมาก (MSB) ส่งไปกับรหัสของหมายเลขเรียกขาน ส่วนอีก 3 บิตที่มีความสำคัญน้อย (LSB) จะไม่ถูกส่งไปด้วย แต่จะถูกใช้เป็นตัวกำหนดหมายเลขของเฟรมที่

บิตที่มีความสำคัญน้อย (LSB) จะไม่ถูกส่งไปด้วย แต่จะถูกใช้เป็นตัวกำหนดหมายเลขของเฟรมที่จะใช้ในการส่งรหัสของหมายเลขเรียกขานออกไปว่าเป็นเฟรมที่เท่าใดของแบตช์ ตัวอย่าง เช่น ต้องการส่งข่าวสารให้กับเครื่องที่มีหมายเลขเรียกขาน 1965576 สามารถแปลงเป็นฐานสองได้ 11101111111000001000 มาตรฐาน POCSAG กำหนดให้บิตซึ่งมีนัยสำคัญต่ำที่สุด 3 บิตสุดท้ายแสดงถึงหมายเลขกลุ่มของเพจเจอร์ เครื่องลูกข่ายทุกเครื่องที่มีเลขหมาย 3 บิตสุดท้ายเป็น 000 001 010 011 101 110 และ 111 จะทำการติดต่อกับสถานีส่งในช่วงของเฟรมที่ 0 1 2 3 4 5 6 และ 7 ตามลำดับ ซึ่งวิธีการดังกล่าวเป็นการช่วยประหยัดการใช้งานแบตเตอรี่ของเครื่องลูกข่าย เนื่องจากวงจรภาครับของเครื่องลูกข่ายจะทำงานเฉพาะในช่วงที่สถานีส่งทำการส่งกลุ่มข้อมูลซิงส์โครไนซ์และช่วงที่ส่งเฟรมซึ่งเป็นของเครื่องลูกข่ายเท่านั้น อาจกล่าวได้ว่าเครื่องลูกข่ายแต่ละเครื่องจะทำงานเฉพาะ 3 ใน 17 ส่วนของเวลาทั้งหมดจึงช่วยเพิ่มประสิทธิภาพของเครื่องรับให้ดีขึ้นด้วยซึ่งแสดงดังรูปที่ 3.2

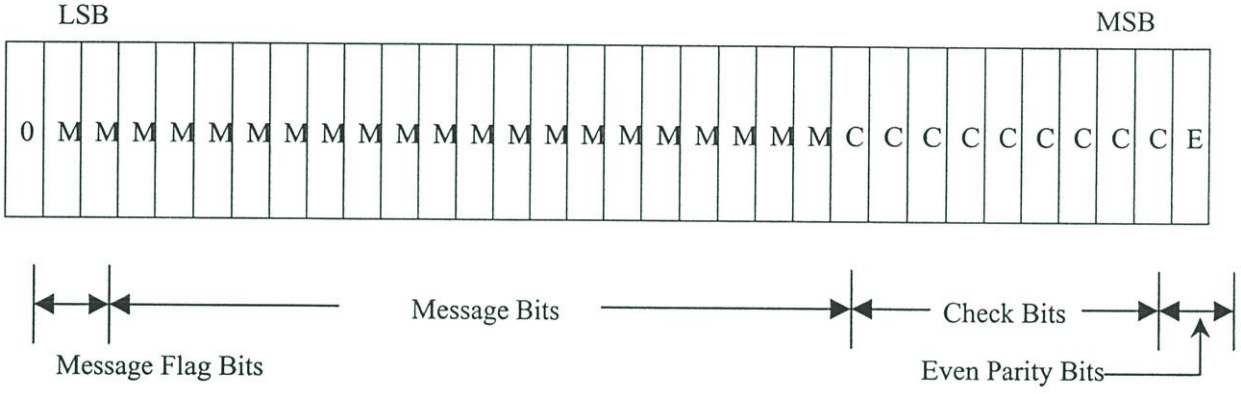


รูปที่ 3.2 รหัสคำหมายเลขเรียกขาน

3.1.2.3 รหัสคำข่าวสาร (Massage Codeword)

รูปแบบข่าวสารข้อมูลที่ใช้ในการรับส่งข้อมูลจะประกอบไปด้วย รหัสคำของข่าวสารเพจเจอร์แบบตัวเลข(Numeric), รหัสคำข่าวสารแบบตัวอักษร-ตัวเลข(Alpha – Numeric) และรหัสข่าวสารแบบตัวอักษรภาษาไทย(Thai) ซึ่งการจัดรูปแบบการส่งข้อมูล โดยมี Message Flag Bit เป็นบิตที่บ่งบอกถึงรูปแบบของข้อมูลที่ทำการรับส่งเพจเจอร์ แสดงดังตารางที่ 3.2

ตารางที่ 3.2 รหัสคำข่าวสาร



3.1.2.3.1 รหัสคำของข่าวสารเพจเจอร์แบบตัวเลข (Numeric)

รูปแบบของรหัสชนิดนี้ถูกแสดงไว้ในตารางที่ 3.2 รหัสคำของข่าวสารจะประกอบด้วยเลขฐานสองจำนวน 32 บิต โดยที่บิตแรกเป็น '0' เสมอ และรหัสคำนี้จะตามหลังรหัสของหมายเลขเรียกขาน รหัสคำข่าวสารจะสิ้นสุดลงเมื่อมีการส่งรหัสคำหมายเลขเรียกขานตัวต่อไป หรือมีการส่งรหัสคำเทียม หรือเมื่อการส่งสัญญาณได้มีการสิ้นสุดลง แต่ถ้าข่าวสารที่จะส่งยาวเลยแบตช์นั้นไป ซึ่งก็หมายความว่าจะถูกแทรกด้วยรหัสคำการซิงส์โครไนซ์ก่อน แล้วข่าวสารส่วนที่เหลือจึงจะถูกส่งในแบตช์ถัดไปจนหมด โดยที่ไม่ต้องเริ่มด้วยรหัสคำหมายเลขเรียกขานอีกครั้ง ข่าวสารในวิทยุติดตามตัวรุ่นตัวเลขจะมีทั้งข่าวสารที่เป็นเลขฐานสิบ เครื่องหมาย Hyphen (-) เครื่องหมาย [,] ช่องว่าง(Space) และสัญลักษณ์ Urgency "U" ซึ่งข้อมูลที่แสดงออกที่หน้าจอวิทยุติดตามตัวก็จะใช้รูปแบบนี้ รูปแบบข่าวสารนี้จะใช้เลขฐานสองจำนวน 4 บิตแทน 1 ตัวอักษร ดังแสดงในตารางที่ 3.3

ตารางที่ 3.3 ชุดอักษรของข่าวสารเพจเจอร์รุ่นตัวเลข

4 Bits Combination Bit No. 4321	Displayed Character
0000	0
0001	1
0010	2

ตารางที่ 3.3 (ต่อ)

4 Bits Combination Bit No. 4321	Displayed Character
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	=
1011	U(Urgency Indicator)
1100	Space
1101	Hyphen
1110	]
1111	[

3.1.2.3.2 รหัสคำของข่าวสารเพจเจอร์แบบตัวอักษร-ตัวเลข (Alpha – Numeric)

สำหรับรหัสคำข่าวสารชนิดนี้จะมีลักษณะโครงสร้างทางบิต และรูปแบบเหมือนกันกับรหัสคำข่าวสารของรุ่นตัวเลขทุกประการดังที่ได้กล่าวมาแล้ว แต่จะมีความแตกต่างของรหัสคำข่าวสารตรงที่จำนวนบิตที่ไม่เท่ากัน เนื่องจากเป็นรุ่นตัวอักษร–ตัวเลข จึงจำเป็นต้องใช้จำนวนบิตเพิ่มมากขึ้น ตามมาตรฐานของ POCSAG 1 อักษร หรือ 1 อักขระ จะใช้จำนวนบิต 7 บิต ด้วยกัน ซึ่งจะต่างกับจำนวนบิตของรุ่นตัวเลขที่มีเพียง 4 บิต เท่านั้น และรุ่นตัวอักษร–ตัวเลขนี้ รหัสข่าวสารที่มีจะมีถึง 80 รูปแบบ โดยมีตัวเลข 0-9 อักษรภาษาอังกฤษพิมพ์ใหญ่ และพิมพ์เล็ก รวมทั้งสัญลักษณ์พิเศษต่างๆลักษณะตำแหน่งข่าวสารในบิตข่าวสารนี้จะมีการเลื่อนไปของข้อมูลตามความยาวของข่าวสารที่ส่งมา หมายความว่า ในบิตข่าวสารของรหัสคำข่าวสารแรกนั้นจะบรรจุได้ 3 ตัวอักษรแต่อักษรตัวที่ 3 จะมีเพียง 6 บิต ส่วนบิตสุดท้ายของอักษรตัวที่ 3 จะไปอยู่ในบิตข่าวสารของรหัสคำข่าวสารถัดไปถ้ามีการส่งเป็นข้อความยาวๆจำนวนบิตของอักษรแต่ละตัวจะถูกเลื่อนออกไป เนื่องจากรูปแบบของบิตข่าวสารกำหนดมีความยาว 20 บิต หรือ 2 ตัวกับอีก 6 บิต นั่นเอง ซึ่งหมายความว่าจะมีอักษรบางตัวที่อยู่คร่อมระหว่างบิตข่าวสารของรหัสคำข่าวสาร 2 รหัสคำซึ่งรหัสข่าวสารของแต่ละตัวอักษรแสดงในตารางที่ 3.4

ตารางที่ 3.4 ชุดอักษรของข่าวสารเพจเจอร์รุ่นตัวเลขตัวอักษร–ตัวเลข

7 Bit Combination Bit No.7654321	Displayed Character
0000110	0
1000110	1
0100110	2
1100110	3
0010110	4
1010110	5
0110110	6
1110110	7
0000110	8
1001110	9
0101110	:
1101110	;
0011110	<
1011110	=
0111110	>
1111110	?
0000001	@
1000001	A
0100001	B
1100001	C
0010001	D
1010001	E
0110001	F
1110001	G
0000001	H

ตารางที่ 3.4 (ต่อ)

7 Bit Combination Bit No.7654321	Displayed Character
1001001	I
0101001	J
1101001	K
0011001	L
1011001	M
0111001	N
1111001	O
0000101	P
1000101	Q
0100101	R
1100101	S
0010101	T
1010101	U
0110101	V
1110101	W
0000101	X
1001101	Y
0101101	Z
1101101	[
0011101	
1011101	]
0111101	^
1111101	-
0000011	“
1000011	a
0100011	b
1100011	c
0010011	d

ตารางที่ 3.4 (ต่อ)

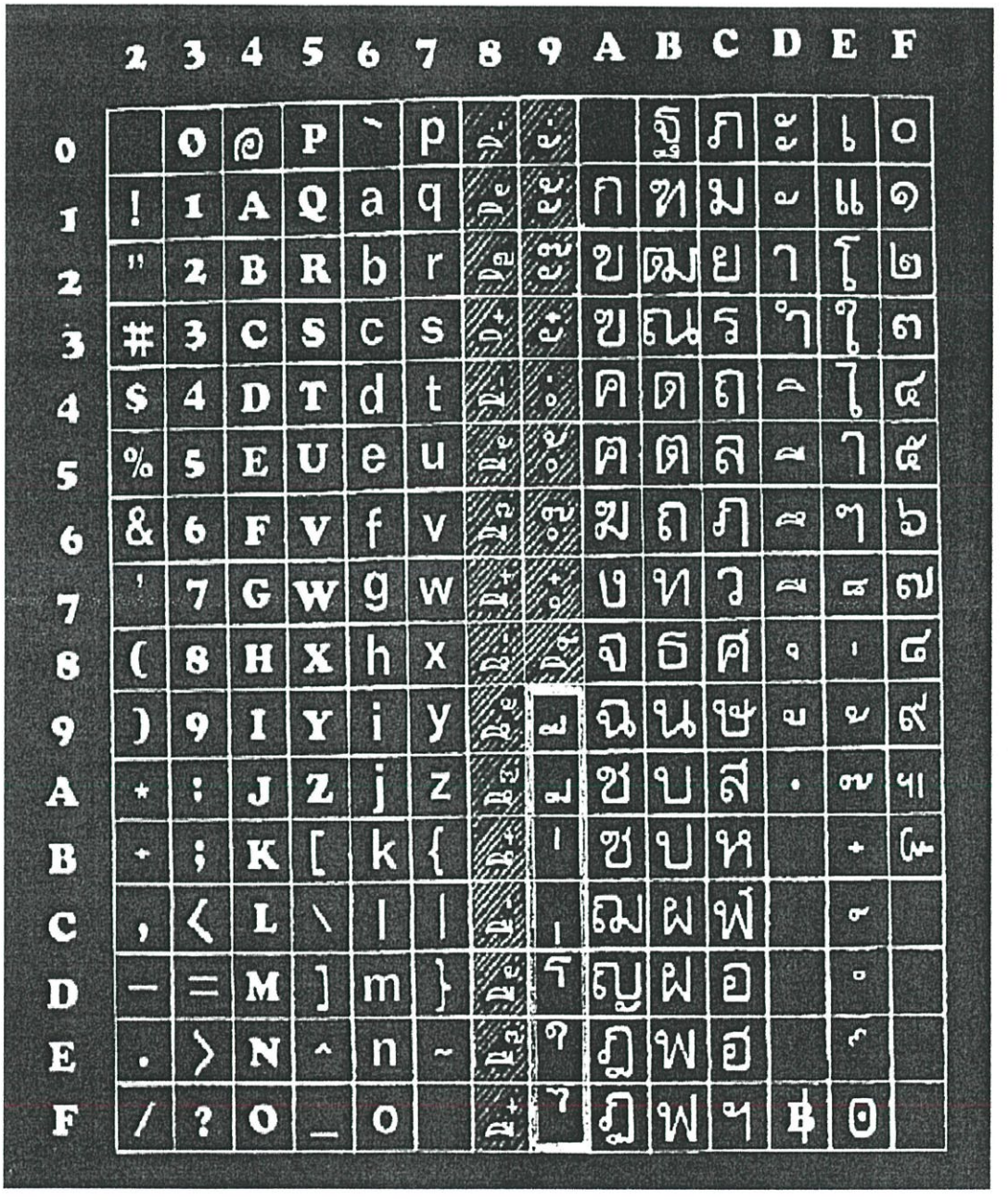
1010011	e
0110011	f
1110011	g
0000011	h
1001011	i
0101011	J
1101011	k
0011011	l
1011011	m
0111011	n
1111011	o
0000111	p
1000111	q
0100111	r
1100111	s
0010111	t
1010111	u
0110111	v
1110111	w
0000111	x
1001111	y
0101111	z
1101111	{
0011111	¦
1011111	}
0111111	~
1111111	#

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

3.1.2.3.3 รหัสคำของข่าวสารเพจเจอร์ตัวอักษรภาษาไทย[3]

สำหรับรหัสคำข่าวสารแบบตัวอักษรภาษาไทย จะใช้มาตรฐานของ สมอ. ในการกำหนดมาตรฐานรูปแบบตัวอักษร ซึ่งจะมีลักษณะโครงสร้างข้อมูลขนาด 8 บิต ในการแสดงผลข้อมูล ดังแสดงไว้ในตารางที่ 3.5

ตารางที่ 3.5 ชุดอักษรของข่าวสารเพจเจอร์ตัวอักษรภาษาไทย(มาตรฐาน สมอ.)



48377

3.1.2.4 รหัสคำว่าง (Idle Codeword)

ถ้าข่าวสารที่ส่งมาหมดลงก่อนการสิ้นสุดลงของเฟรม รหัสคำว่างนี้จะถูกใส่แทนรหัสคำปกติ เพื่อให้เฟรมนั้นมีครบสองรหัสคำ(64 บิต) และในกรณีที่ข่าวสารที่จะส่งหมดลงก่อนที่จะสิ้นสุดแบตซ์ รหัสคำว่างก็จะได้รับการใส่ไว้เพื่อให้ครบแปดเฟรมด้วยเช่นกัน รูปแบบของรหัสคำว่างนี้มีลักษณะแน่นอนและต้องไม่มีกลุ่มข้อมูลประเภทใดใช้รูปแบบเดียวกันกับกลุ่มข้อมูลนี้ดังแสดงไว้ในตารางที่ 3.6

ตารางที่ 3.6 รูปแบบของรหัสคำว่าง

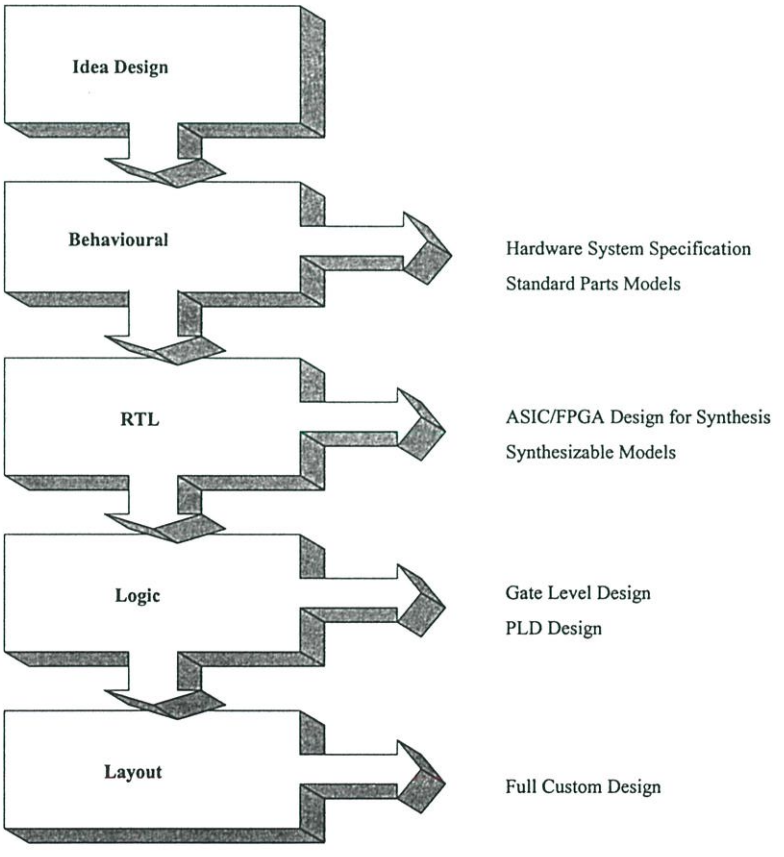
LSB MSB

N	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2
Bit	0	1	1	1	1	0	1	0	1	0	0	0	1	0	0	1	1	1	0	0	0	0	0	1	1	0	0	1	0	1	1	1

บทที่ 4
การออกแบบวงจรในลักษณะโครงสร้าง และการบรรยายพฤติกรรม

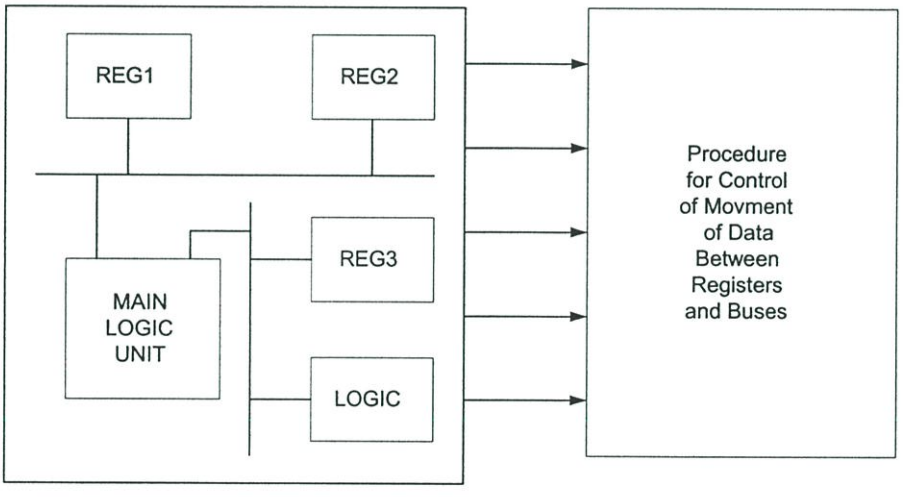
ความซับซ้อนและขนาดของระบบดิจิตอลในปัจจุบันได้เพิ่มมากขึ้นทุกขณะ ส่งผลให้มีการนำคอมพิวเตอร์เพื่อช่วยในการออกแบบหรือ CAD มาใช้ในขบวนการออกแบบฮาร์ดแวร์เพิ่มขึ้นเช่นกัน อีกทั้งอุปกรณ์และวิธีการออกแบบใหม่ๆ ก็ถูกพัฒนาขึ้นมาช่วยอำนวยความสะดวกให้กับนักออกแบบมากขึ้นด้วย สำหรับภาษาบรรยายพฤติกรรมฮาร์ดแวร์ (HDL : Hardware Description Language) ก็เป็นเครื่องมืออย่างหนึ่งที่ได้รับการพัฒนามาอย่างต่อเนื่องเพื่อช่วยให้การปรับปรุงขบวนการออกแบบวงจรดิจิตอลเป็นไปอย่างมีประสิทธิภาพ

4.1 การออกแบบระบบดิจิตอล[4]



รูปที่ 4.1 ขั้นตอนการออกแบบระบบดิจิตอล

ในการออกแบบระบบดิจิตอลเริ่มตั้งแต่การกำหนดแนวความคิดเบื้องต้นจนกระทั่งได้ออกมาเป็นอุปกรณ์ฮาร์ดแวร์ที่ใช้งานได้จะต้องผ่านขั้นตอนมากมาย และในแต่ละขั้นตอนผู้ออกแบบจะต้องตรวจสอบผลลัพธ์ในแต่ละขั้น ก่อนเข้าสู่กระบวนการออกแบบในขั้นต่อไป รูปที่ 4.1 แสดงขั้นตอนปกติที่ใช้ในการออกแบบระบบดิจิตอลทั่วไป ขั้นแรกผู้ออกแบบจะต้องกำหนดแนวความคิดในการออกแบบและทำการพัฒนาให้สามารถนำไปใช้งานได้อย่างสมบูรณ์ ซึ่งภายในขั้นตอน นี้ผู้ออกแบบจำเป็นต้องสร้างรูปแบบระบบในเชิงพฤติกรรมขึ้นมาตรวจสอบซึ่งอาจจะเป็นผังงานแสดงแบบ (Flow Graph) หรือรหัสคำสั่งเทียม (Pseudo Code) ก็ได้ ขั้นตอนต่อไปเป็นการออกแบบระบบเส้นทางข้อมูล (BUS) ผู้ออกแบบจะกำหนดส่วนประกอบของรีจีสเตอร์และวงจรลอจิกที่จำเป็นทั้งหมดเพื่อนำมาประกอบเป็นระบบที่สมบูรณ์ โดยแต่ละองค์ประกอบสามารถเชื่อมต่อกันด้วยบัสหนึ่งหรือสองทิศทาง (Unidirectional or Bidirection Bus) ส่วนกระบวนการในการควบคุมการเคลื่อนย้ายข้อมูลระหว่างรีจิสเตอร์และวงจรลอจิกจะขึ้นอยู่กับพฤติกรรมของระบบที่กำหนดไว้ดังรูปที่ 4.2



รูปที่ 4.2 การออกแบบระบบเส้นทางของข้อมูล

ขั้นตอนถัดมาเป็นการออกแบบวงจรลอจิก ซึ่งจะเกี่ยวข้องกับการนำเกทดิจิตอลพื้นฐานและฟลิปฟลอป (Flip Flop) มาประกอบเป็นอุปกรณ์ย่อยต่างๆ เช่น รีจิสเตอร์เก็บข้อมูลบัสวงจรลอจิก และส่วนควบคุมฮาร์ดแวร์ซึ่งผลลัพธ์ที่ได้ในขั้นตอนนี้จะเป็นเครือข่ายของการโยงใยระหว่างเกทและฟลิปฟลอปนั่นเอง การออกแบบในขั้นตอนถัดไปเป็นการเปลี่ยนเครือข่ายการโยงใยที่ได้จากขั้นตอนที่แล้ว ให้เป็นลำดับของทรานซิสเตอร์ (Transistor List) และ Layout ซึ่งขั้นตอนนี้จะเกี่ยวข้องโดยตรงกับการจัดวางทรานซิสเตอร์หรือไลบรารีเซลล์ เพื่อแทนเกทและฟลิปฟลอปต่างๆ

และในขั้นตอนสุดท้ายจะเป็นการส่งระบบที่ออกแบบไว้ไปทำการเจือสารที่โรงงานเพื่อผลิตออกมาเป็นวงจรรวมในที่สุด

4.2 ประวัติความเป็นมาของภาษาวีเอชดีแอล[5]

วีเอชดีแอล (VHDL) ย่อมาจากคำว่า VHSIC Hardware Description Language (VHSIC: Very High Speed Integrated Circuit) เป็นภาษาโปรแกรมระดับสูง (High Level Language) ที่ใช้สำหรับออกแบบฮาร์ดแวร์ในระบบดิจิตอล ตัวของภาษาสามารถบรรยายพฤติกรรมการทำงานในรูปของลำดับชั้น (Hierarchy) และสามารถเขียนได้หลายรูปแบบ ด้วยเหตุผลนี้จึงทำให้ภาษาวีเอชดีแอลเป็นเครื่องมือที่ใช้ออกแบบตั้งแต่ขั้นตอนบนสุด คือ แนวความคิดที่จะแก้ปัญหา ลงไปที่ละขั้นจนถึงขั้นตอนการสร้างวงจรจริง และตัวภาษาก็เปิดโอกาสให้วิศวกรได้พัฒนาและจำลองการทำงานของรูปแบบฟังก์ชันการทำงานของวงจรอย่างสังเขป โดยยังไม่ต้องคำนึงถึงรายละเอียดเกี่ยวกับโครงสร้างวงจรจริง นอกจากนั้นภาษาวีเอชดีแอล จึงเป็นภาษาที่น่าสนใจในการศึกษา และนำไปใช้งานเป็นอย่างยิ่ง

วิวัฒนาการของวีเอชดีแอลเริ่มต้นประมาณปี คศ 1981 เมื่อกระทรวงกลาโหมสหรัฐอเมริกา หรือ DoD (Department of Defense) ได้พยายามปรับปรุงอุปกรณ์อิเล็กทรอนิกส์และคอมพิวเตอร์ที่ใช้ในกิจการทางทหารให้มีความทันสมัยมากขึ้น ประกอบกับเทคโนโลยีทางด้านไมโครอิเล็กทรอนิกส์มีการพัฒนาไปอย่างรวดเร็วดังจะเห็นได้จากนำเอาวงจรดิจิตอลหลายๆวงจรมาทำการผลิตอยู่บนแผ่นซิลิกอลที่มีพื้นที่เพียง 1–2 ตารางเซนติเมตรเท่านั้น ซึ่งเป็นผลให้ประสิทธิภาพในการทำงานของวงจรสูงขึ้น ตลอดจนความน่าเชื่อถือในการทำงานและความคงทนต่อสภาพแวดล้อมสูงขึ้น แต่เนื่องจากในขณะนั้นขั้นตอนการออกแบบ การผลิต และการตรวจสอบวงจรต้นแบบ เป็นขบวนการที่ต้องใช้วิศวกร และเวลาในการดำเนินการมาก ฉะนั้นทาง DoD จึงจัดตั้งโครงการขึ้นมาเพื่อศึกษาวิธีการที่ช่วยในการพัฒนาวงจรอิเล็กทรอนิกส์ โดยเฉพาะอย่างยิ่งวงจรระบบดิจิตอล ให้สามารถนำไปผลิตได้เร็วขึ้น ซึ่งโครงการดังกล่าวมีชื่อว่า “ Very High Speed Integrated Circuits” หรือ VHSIC โดยในระยะแรกนั้นโครงการนี้ถือว่าเป็นความลับทางด้านความมั่นคงของประเทศ และอยู่ภายใต้ความควบคุมดูแลของ United States International Traffic and Arms Regulations (ITAR)

สำหรับมาตรฐานของภาษาที่ใช้บรรยายพฤติกรรมวงจรหรือฮาร์ดแวร์ของระบบสำหรับโครงการ VHSIC ที่ DoD ได้ให้ไว้สามารถสรุปได้ดังนี้

- ต้องเป็นภาษาที่นำไปเขียนรูปแบบระบบดิจิตอล และมีคุณสมบัติ ที่สามารถเข้าใจได้ทั้งมนุษย์และเครื่องคอมพิวเตอร์โดยไม่ต้องมีการแปลหรือเปลี่ยนแปลงอีก

- สามารถนำไปใช้เป็นเอกสารประกอบโครงการได้
- ต้องเป็นภาษาที่เขียนขึ้นสำหรับใช้จำลองการทำงานของวงจร

ฉะนั้นภาษาดังกล่าวนี้จึงจัดเป็นภาษาโปรแกรมระดับสูงเช่น เดียวกับภาษาปาสคาล หรือภาษาซี ซึ่งในทางวิศวกรรมภาษาที่ใช้ในการออกแบบฮาร์ดแวร์นี้เรียกว่า "Hardware Description Language" หรือ HDL

ในตอนเริ่มแรกนั้น DoD ได้มอบหมายให้บริษัทไอบีเอ็มเท็กซัสอินสตูเมนท์ และอินเตอร์เมทริกซ์ เป็นผู้ศึกษาและพัฒนาโครงการ ซึ่งการดำเนินงานเป็นไปอย่างต่อเนื่อง จนกระทั่งในปี คศ 1985 ทาง ITAR ได้ยกเลิกข้อจำกัดในการถ่ายทอดเทคโนโลยีทางทหารออกจากโครงการนี้ ดังนั้นภาษาวีเอชดีแอล จึงเริ่มเป็นที่รู้จักกันโดยทั่วไป และประมาณปี คศ 1987 IEEE ได้ทำการกำหนดมาตรฐานของภาษานี้เป็น IEEE 1076-1987 และมีชื่อเรียกว่า วีเอชดีแอล ซึ่งมาตรฐานนี้ได้รับการปรับปรุงจนเป็นมาตรฐาน IEEE 1076 – 1993 หรือ VHDL 1993

เนื่องจากในขณะนั้น DoD เป็นลูกค้ารายใหญ่ของอุตสาหกรรมอิเล็กทรอนิกส์และคอมพิวเตอร์ ดังนั้นจึงมีผู้รับโครงการต่างๆ จาก DoD ไปดำเนินการวิจัยและพัฒนาเป็นจำนวนมาก และเพื่อให้ทุกโครงการอยู่ในมาตรฐานเดียวกันหมด ดังนั้นทาง DoD จึงได้กำหนดว่าทุกๆโครงการต้องเขียนอยู่ในรูปของภาษาวีเอชดีแอลเท่านั้น ซึ่งทำให้ DoD สามารถนำโครงการเหล่านี้ไปจำลองกับเครื่องคอมพิวเตอร์ได้หลายๆระบบ

4.3 ข้อกำหนดของภาษาวีเอชดีแอล[6]

4.3.1 ลักษณะทั่วไป

DoD ได้กำหนดให้วีเอชดีแอลเป็นภาษาสำหรับการออกแบบและบรรยายฮาร์ดแวร์ซึ่งหมายถึงความสามารถในการอธิบายและออกแบบในระดับสูง การจำลอง(Simulation)การสังเคราะห์(Synthesis) และการทดสอบ(Testing)นอกจากนั้นภาษาวีเอชดีแอลยังถูกกำหนดไว้สำหรับการบรรยายฮาร์ดแวร์ตั้งแต่ระดับบน ซึ่งก็คือระบบบนจนถึงระดับเกทอีกด้วย

เนื่องจากในการทำงานของระบบดิจิตอลนั้นทุกๆ องค์ประกอบภายในระบบไม่ว่าเล็กหรือใหญ่จะทำงานไปพร้อมๆกัน ซึ่งในเรื่องของความพร้อมเพรียงในการทำงานนี้ก็ถือเป็นข้อกำหนดที่สำคัญอย่างหนึ่งของวีเอชดีแอลด้วย เช่นกัน สำหรับในภาษาที่ใช้ในการบรรยายฮาร์ดแวร์นั้น ความพร้อมเพรียงจะหมายถึงทุกๆคำสั่ง องค์ประกอบ เกทหรือวงจรลอจิกจะถูกนำมาปฏิบัติทั้งหมด ดังนั้นในที่สุดแล้วก็จะดูเหมือนว่าได้มีการปฏิบัติไปพร้อมๆกัน

4.3.2 สนับสนุนการออกแบบแบบลำดับขั้น

การออกแบบแบบลำดับขั้น เป็นลักษณะที่สำคัญอย่างหนึ่งสำหรับการออกแบบระบบที่มีหลายๆระดับ โดยในการออกแบบจะประกอบด้วยส่วนการบรรยายการเชื่อมต่อ และส่วนการบรรยายหน้าที่การทำงาน ซึ่งหน้าที่การทำงานของระบบสามารถกำหนดได้ด้วยตัวเอง หรืออาจถูกกำหนดโดยโครงสร้างที่ประกอบด้วยองค์ประกอบย่อยๆ ลงไปได้เช่นกัน แต่ที่ระดับล่างสุด องค์ประกอบต้องถูกบรรยายหน้าที่การทำงานด้วยตัวมันเอง และไม่สามารถกำหนดการทำงานโดยลักษณะแบบโครงสร้างได้

4.3.3 ไลบรารี

วีเอชดีแอลได้สนับสนุนการมีไลบรารีเพื่อระบบการจัดการที่ดี ผู้ออกแบบสามารถกำหนดลักษณะและการทำงานของอุปกรณ์พื้นฐานไว้ในระบบไลบรารี หรือจะใช้ไลบรารี ที่ระบบได้จัดเตรียมไว้แล้วก็ได้ โมเดลและการบรรยายที่ถูกต้องควรจัดเก็บไว้ในไลบรารี หลังจากที่ได้ผ่านการคอมไพล์เรียบร้อยแล้วเพื่อให้ผู้ออกแบบคนอื่นๆสามารถนำไปใช้ได้ด้วย

4.3.4 ลำดับคำสั่ง

แม้ว่าการปฏิบัติคำสั่งหรือกระบวนการโดยพร้อมเพรียงกันจะเป็นคุณสมบัติ ที่สำคัญของวีเอชดีแอลก็ตาม ตัวภาษาเองก็ยังมีการจัดเตรียมลักษณะการควบคุมแบบลำดับคำสั่งไว้ให้ด้วย เมื่อผู้ออกแบบได้กำหนดหน้าที่ และองค์ประกอบที่ทำงานพร้อมกันของระบบไว้เรียบร้อยแล้ว ผู้ออกแบบยังสามารถบรรยายหน้าที่การทำงานซึ่งเป็นรายละเอียดภายในของแต่ละองค์ประกอบได้ในลักษณะเดียวกับการเขียนโปรแกรมที่ประกอบด้วยโครงสร้างแบบ Case, if–Then–Else และ Loop ทั่วๆไปได้

การบรรยายแบบลำดับคำสั่งทำให้การออกแบบหน้าที่การทำงานของอุปกรณ์กระทำได้สะดวกและง่ายขึ้น อย่างไรก็ตามโครงสร้างทั้งหมดของวีเอชดีแอลก็ยังคงเป็นการทำงานแบบพร้อมเพรียงกันเช่นเดิม

4.3.5 การกำหนดคุณสมบัติ

นอกจากการกำหนดอินพุทและเอาท์พุตแล้ว เงื่อนไขอื่นๆก็มีผลต่อการปฏิบัติหน้าที่ของอุปกรณ์ฮาร์ดแวร์ด้วยเช่นกัน โดยสิ่งนี้รวมถึงสภาพแวดล้อมและลักษณะทางกายภาพของอุปกรณ์นั้นๆด้วย ซึ่งภาษาสำหรับการออกแบบที่ดีควรให้ผู้ออกแบบ กำหนดคุณสมบัติของอุปกรณ์ที่ใช้ได้ด้วย เช่น สามารถกำหนดขนาด ลักษณะทางกายภาพ เวลา โหลด และเงื่อนไขทางสภาพแวดล้อมอื่นๆ ซึ่งความสามารถในการกำหนดคุณสมบัตินี้ก็เป็นส่วนหนึ่งที่มีอยู่ในภาษาวีเอชดีแอลด้วยเช่นกัน

4.3.6 ชนิดของข้อมูล

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

4.3.7 โปรแกรมย่อย

ความสามารถในการใช้ฟังก์ชันและโพรซีเจอร์ (Procedure) ก็เป็นข้อกำหนดอีกอย่างหนึ่งในวีเอชดีแอลซึ่งผู้ออกแบบสามารถนำโปรแกรมย่อยมาใช้ในการเปลี่ยนแปลงชนิดของข้อมูล การกำหนดหน่วยของลอจิก การกำหนดตัวกระทำต่างๆ หรือหน้าที่อื่นๆ ตามที่ต้องการได้เช่นเดียวกับการเขียนโปรแกรมทั่วไป

4.3.8 การควบคุมเวลา

วีเอชดีแอลอนุญาตให้ผู้ออกแบบสามารถกำหนดเวลาในการส่งผ่านข้อมูลหรือสัญญาณได้ตามต้องการ การตรวจสอบ การออกแบบเกทหรือการหน่วงเวลาก็สามารถกระทำได้โดยการกำหนดช่วงเวลาที่แน่นอนหรือกำหนดให้มีการรอคอยเหตุการณ์(Even) นอกจากนี้ก็ยังสามารถกำหนดรูปแบบของสัญญาณนาฬิกาได้อีกด้วย

4.3.9 การกำหนดแบบโครงสร้าง

การกำหนดโครงสร้างขององค์ประกอบต่างๆ สามารถกระทำได้ในทุกระดับของการออกแบบโดยการกำหนดโครงสร้างขององค์ประกอบร่วมที่เกิดจากองค์ประกอบย่อย ซึ่งแตกต่างกัน หรือเหมือนกันก็เป็นข้อกำหนดอย่างหนึ่งของวีเอชดีแอลเช่นกัน

4.4 องค์ประกอบพื้นฐานของวีเอชดีแอล

รูปแบบพื้นฐานที่ใช้ในการบรรยายถึงองค์ประกอบของวีเอชดีแอลจะประกอบไปด้วยส่วนกำหนดการเชื่อมต่อ(Interface) และส่วนกำหนดลักษณะเชิงสถาปัตยกรรม(Architecture) ดังแสดงในรูปที่ 4.3 โดยในการบรรยายการเชื่อมต่อจะขึ้นต้นด้วยคำว่า ENTITY แล้วตามด้วยชื่อขององค์ประกอบจากนั้นตามด้วยคำว่า IS และถัดมาจะเป็นการบรรยายถึงพอร์ตการติดต่ออินพุตเอาท์พุตขององค์ประกอบ ส่วนลักษณะภายนอกอื่นๆ เช่น เวลา อุณหภูมิ ก็สามารถรวมเข้าไปในส่วนนี้ได้เช่นกัน

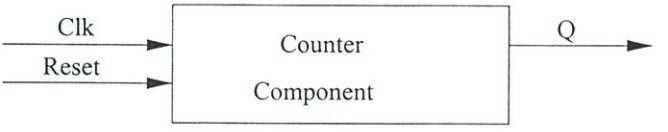
```
ENTITY  entity_name IS
        GENERIC ( generic_list );
        PORT  (  port_name : < Mode> port_type );
END entity_name;

ARCHITECTURE  architecture_name OF entity_name IS
        declarations
BEGIN
        concurrent_statements
END architecture_name;
```

รูปที่ 4.3 การกำหนดการเชื่อมต่อและสถาปัตยกรรม

ในส่วนของการกำหนดลักษณะเชิงสถาปัตยกรรมจะขึ้นต้นด้วยคำว่า ARCHITECTURE ซึ่งเป็นส่วนที่ใช้บรรยายหน้าที่การทำงานขององค์ประกอบ โดยหน้าที่การทำงานนี้จะขึ้นอยู่กับสัญญาณอินพุท–เอาท์พุท และพารามิเตอร์อื่นๆ ที่ได้กำหนดไว้ในส่วนของการเชื่อมต่อดังรูที่ 4.3 และสำหรับการบรรยายหน้าที่ขององค์ประกอบจะเริ่มต้นหลังจากคำว่า BEGIN เป็นต้นไป

4.4.1 การกำหนดการเชื่อมต่อ

การกำหนดการเชื่อมต่อเป็นระดับบนสุดของการออกแบบ โดยในระดับนี้ต้องกำหนดพอร์ตสำหรับการติดต่อกับองค์ประกอบภายนอกอื่นๆ ดังตัวอย่างในรูปที่ 4.4 ซึ่งเป็นบล็อกไดอะแกรม และการบรรยายการเชื่อมต่อขององค์ประกอบสำหรับตัวเคาร์เตอร์ในบรรทัดแรกของการบรรยายการเชื่อมต่อเป็นการกำหนดชื่อขององค์ประกอบซึ่งกำหนดเป็น Counter component ตามด้วยคำว่า PORT และชื่อของพอร์ตอยู่ภายในวงเล็บ ส่วน IN และ OUT เป็นการกำหนดโหนดของสัญญาณให้เป็นอินพุทหรือเอาท์พุท โดยที่ Std_Logic, Integer เป็นการแสดงชนิดของข้อมูล




```
ENTITY counter IS
    PORT (  clk   : in std_logic;
            reset : in std_logic;
            q     : out integer range 0 to 15 );
END counter;
```

รูปที่ 4.4 บล็อกไดอะแกรมและการบรรยายการเชื่อมต่อของ clock_componet

4.4.2 การกำหนดรูปแบบการบรรยาย

หน้าที่การทำงานขององค์ประกอบจะถูกบรรยายภายในส่วนนี้ ซึ่งในการบรรยายสามารถกำหนดค่าของสัญญาณเอาท์พุทในเทอมของอินพุทหรือในรูปขององค์ประกอบอื่นๆ หรือทั้งสองอย่างรวมกันได้ ดังตัวอย่าง การบรรยายของ Mux_Component ในรูปที่ 4.5 ซึ่งเป็นการบรรยายในเชิงพฤติกรรมโดยมี in0, in1, sel เป็นอินพุท และ output เป็นเอาท์พุท PROCESS เป็นคำที่ใช้ในการเริ่มต้นสำหรับการบรรยายในเชิงพฤติกรรม และภายในโปรเซส กำหนดให้ sel เป็นสัญญาณที่ทำหน้าที่เลือกสัญญาณระหว่าง in0 และ in1 โดยถ้า sel เป็น '0' จะให้สัญญาณ output มีค่าเป็น in0, แต่ถ้าไม่ใช้จะให้ output มีค่าเท่ากับ in1

```
ARCHITECTURE  behavioural OF Mux IS
BEGIN
        PROCESS ( in0, in1,sel )
        BEGIN
                if ( sel ='0' ) then
                        output <= in0;
                else
                        output <= in1;
                end if;
        END PROCESS;
END behavioural;
```

รูปที่ 4.5 การบรรยายเชิงพฤติกรรมของ Mux

4.4.3 หน่วยการออกแบบแพ็กเกจ

ข้อมูลต่างๆตลอดจนโปรแกรมย่อย ที่เป็นประโยชน์ต่อการเขียนรูปแบบการบรรยายระบบดิจิตอล สามารถเก็บไว้ในส่วนของแพ็กเกจ ซึ่งหน่วยการออกแบบต่างๆ เช่น หน่วยการออกแบบ Entity หน่วยการออกแบบสถาปัตยกรรม หรือหน่วยการออกแบบแพ็กเกจอื่นๆ สามารถเรียกข้อมูลเหล่านี้ไปใช้ได้ นอกจากนั้นสิ่งที่นิยมทำกันมาก คือ การนำรูปแบบมาตรฐานต่างๆ เช่น อุปกรณ์มาตรฐาน (เช่น ไอซีตระกูล 74XX เป็นต้น) มาเก็บไว้ในรูปของแพ็กเกจ ที่ทุกคนสามารถเข้าถึงได้

ตามปกติแล้วแพ็กเกจจะแบ่งออกเป็น2ส่วนคือ การประกาศแพ็กเกจ(Package Declaration) และส่วนของบอดี้แพ็กเกจ(Package Body) เนื่องจาก แพ็กเกจถูกสร้างขึ้นเป็นส่วนแยกต่างหากออกจากรูปแบบที่กำลังเขียนอยู่ ฉะนั้นการที่นำแพ็กเกจไปใช้นั้นจะต้องมีการเชื่อมโยงหรืออ้างอิงเสียก่อน ซึ่งในภาษาวีเอชดีแอลสามารถกระทำได้ด้วยชุดคำสั่ง USE

4.4.3.1 Package Declaration

ส่วนที่มีความสำคัญที่สุดของแพ็กเกจ ได้แก่ส่วนการประกาศแพ็กเกจ เนื่องจากเป็นส่วนที่ใช้กำหนดชื่อของสิ่งที่ประกาศอยู่ภายในแพ็กเกจ สำหรับนำไปใช้ภายนอกตัวของแพ็กเกจเองถ้ามีการประกาศสิ่งใดๆในส่วนของส่วนบอดี้แพ็กเกจ แต่ไม่ถูกประกาศในส่วนการประกาศแพ็กเกจจะทำให้ค่าและพฤติกรรมไม่สามารถนำไปใช้งานในส่วนนอกได้ ซึ่งเปรียบเทียบได้กับสิ่งที่ประกาศไว้ในส่วนของการประกาศ Entity คือ จุดเชื่อมต่อ หรือ พอร์ต ที่มีหน้าที่ติดต่อกับโลกภายนอก ฉะนั้นโดยทั่วไปแล้วแพ็กเกจสามารถสร้างขึ้นได้โดยไม่จำเป็นต้องมีส่วนบอดี้ และยังสามารถนำไปใช้งานจากรูปแบบภายนอกได้เช่น ใช้สำหรับประกาศชนิด(Type) หรือสัญญาณ เช่นเดียวกับส่วนบอดี้แพ็กเกจที่ไม่จำเป็นต้องมีส่วนของการประกาศแพ็กเกจ แต่แพ็กเกจนั้นจะไม่สามารถนำไปใช้จากรูปแบบอื่นได้แสดงดังรูปที่ 4.6

```
PACKAGE package_name IS
        [ exported_subprogram_declarations ]
        [ exported_constant_declarations ]
        [ exported_components ]
        [ exported_type_declarations
        [ attribut_declarations ]
        [ attribut_specifications ]
END package_name;
```

รูปที่ 4.6 โครงสร้างทั่วไปของส่วนการประกาศแพ็กเกจ

4.4.3.2 Package Body

โครงสร้างซึ่งประกอบด้วยลำดับคำสั่งที่ใช้บรรยายฟังก์ชันการทำงานของโปรแกรมย่อยทั้งหลาย ซึ่งชื่อของโปรแกรมย่อยนั้นๆ ได้ถูกประกาศไปแล้วในส่วนของการประกาศแพ็กเกจจะถูกเก็บไว้ในส่วนของบอดี้แพ็กเกจ ทั้งนี้รวมถึงการกำหนดค่าคงที่ต่างๆ อันได้แก่ ค่าคงที่ที่ถูกประกาศชื่อไว้ก่อนในส่วนของการประกาศแพ็กเกจ และถูกกำหนดค่าในส่วนของบอดี้แพ็กเกจ ฉะนั้นในส่วนของบอดี้แพ็กเกจจึงไม่จำเป็นต้องมี ถ้าในส่วนของการประกาศแพ็กเกจไม่มีการประกาศชื่อที่เป็นโปรแกรมย่อย หรือค่าคงที่ การเขียนบอดี้แพ็กเกจนั้นจะเป็นไปตามกฎเกณฑ์ดังแสดงในรูปที่ 4.7

```
PACKAGE BODY package_name IS
        [ exported_subprogram_declarations ]
        [ exported_constant_declarations ]
        [ exported_components ]
        [ exported_type_declarations
        [ attribut_declarations ]
        [ attribut_specifications ]
END package_name;
```

รูปที่ 4.7 โครงสร้างของบอดี้แพ็กเกจ

4.4.4 หน่วยการออกแบบ Configuration

ดังที่ทราบกันแล้วว่าระบบดิจิตอลรูปแบบหนึ่งไม่ว่าจะเป็นอะไรก็ตาม จะสามารถมีหน่วยการออกแบบ Entity ได้เพียงหนึ่งเดียวเท่านั้น ซึ่งในหน่วยการออกแบบ Entity หนึ่งหน่วยนี้อาจจะมีสถาปัตยกรรมที่เป็นหน่วยรองได้หลายหน่วย ดังนั้นจะต้องมีหน่วยการออกแบบ Configuration มาเพื่อกำหนดการใช้ Configuration ของการประกอบ Entity กับหน่วยการออกแบบสถาปัตยกรรมหน่วยใดๆเข้าด้วยกันดังแสดงในรูปที่ 4.8

```
CONFIGURATION configuration_name OF entity_name IS
        for architecutre_name
END configuration_name;
```

รูปที่ 4.8 โครงสร้างโดยทั่วไปของหน่วยการออกแบบโครงแบบ

4.4.5 โปรแกรมย่อย

การใช้ฟังก์ชันและโพรซีเจอร์ในภาษาวีเอชดีแอลเปรียบได้กับการใช้โปรแกรมย่อยในการเขียนโปรแกรมภาษาชั้นสูงทั่วๆไป ค่าที่ถูกส่งกลับหรือถูกเปลี่ยนแปลงโดยโปรแกรมย่อยอาจจะมีหรือไม่มีผลต่อฮาร์ดแวร์โดยตรงก็ได้ เช่นถ้าใช้ฟังก์ชันแทนการกระทำในสมการบูลีนก็จะมีผลต่อวงจรลอจิกจริงๆ ในขณะที่ถ้าใช้โปรแกรมย่อยในการเปลี่ยนชนิดของข้อมูลหรือในการคำนวณค่าการหน่วงเวลาแล้วก็จะไม่มีผลต่อโครงสร้างของฮาร์ดแวร์ แสดงดังรูปที่ 4.9 และ รูปที่ 4.10

```
PROCEDURE  parity_proc ( x : std_logic_vector;
        signal parity_bit : out std_logic ) IS
        VARIABLE tmp : std_logic;
BEGIN
        tmp := '0';
                FOR j in x' range LOOP
                        tmp := tmp xor x(j);
                END LOOP;
                        parity_bit <= tmp;
END parity_proc;
```

รูปที่ 4.9 การใช้โพรซีเจอร์

```
PROCEDURE parity_proc ( x : std_logic_vector )
                    return  std_logic IS
                    VARIABLE tmp : std_logic;
          BEGIN
                      tmp := '0';
                   FOR j in x' range LOOP
                            tmp := tmp xor x(j);
                   END LOOP;
                            resturn tmp;
```

รูปที่ 4.10 การใช้ฟังก์ชัน

4.4.6 โอเปอร์เรเตอร์

การบรรยายเชิงพฤติกรรมในภาษาวีเอชดีแอลมีตัวดำเนินการหรือโอเปอร์เรเตอร์ทางลอจิกและคณิตศาสตร์เช่นเดียวกับภาษาซอฟต์แวร์ทั่วไปดังรูปที่ 4.11

```
                    PREDEFIND OPERATORS

LOGICAL OPERATORS :  NOT AND OR NAND NOR XOR

          OPERAND TYPE : BIT BOOLEAN
          RESULT    TYPE : BIT BOOLEAN

RELATIONS OPERATORS :  =/  =  <  <=  >  >=
          OPERAND TYPE :  any type
          RESULT    TYPE :  boolean

ARCHITECTURE OPERATORS :  +  -  *  /  ** MOD REM ABS
          OPERAND TYPE : INTEGER REAL Physical
          RESULT    TYPE : INTEGER REAL Physical

CONCANTENATION OPERATION :  &
          OPERAND TYPE : ARRAY of any type
          RESULT    TYPE : ARRAY of any type
          RESULT     TYPE  : ARRAY of any type
```

รูปที่ 4.11 ตัวดำเนินการในวีเอชดีแอล

4.4.7 เวลาและความพร้อมเพรียง

ในวงจรอิเล็กทรอนิกส์อุปกรณ์ทุกๆตัวจะอยู่ในสภาพเตรียมพร้อมเสมอ(Always Active) และจะมีเรื่องของเวลาเข้ามาเกี่ยวข้องในทุกๆเหตุการณ์ที่เกิดขึ้นเสมอ วีเอชดีแอลเป็นภาษาที่ได้รับการออกแบบมาเพื่อให้สามารถบรรยายรูปแบบและการพ้องกันของเวลาสำหรับการทำงานของอุปกรณ์ได้อย่างถูกต้อง การบรรยายการทำงานที่อยู่ภายในส่วนของการบรรยายสถาปัตยกรรม จะมีการทำงานที่พร้อมเพรียงกันเสมอ หรือแม้แต่โปรเซสซึ่งมีการทำงานภายในเป็นแบบลำดับคำสั่งก็ตาม ซึ่งหากมีหลายๆโปรเซสอยู่ภายในโครงสร้างเดียวกัน ทุกๆ โปรเซสก็จะทำงานไปพร้อมๆกันด้วย

4.4.8 สัญญาณและตัวแปร

สัญญาณมีลักษณะเป็นเสมือนตัวกลางฮาร์ดแวร์ที่ใช้ในการส่งผ่านข้อมูลและมีเรื่องของเวลาเข้ามาเกี่ยวข้องด้วย การกำหนดค่าให้กับสัญญาณจะใช้สัญลักษณ์ <= ในการส่งค่าและสามารถใช้คำสั่ง AFTER เพื่อกำหนดช่วงเวลาในการส่งผ่านค่าของสัญญาณเช่น w <= a AFTER 12 NS หมายถึง การกำหนดค่าสัญญาณ a ให้กับ w หลังจากเวลาผ่านไป 12 นาโนวินาที ในทางตรงข้ามตัวแปรมีลักษณะเป็นเสมือนตัวกลางที่ใช้ในการส่งผ่านข้อมูลและไม่มีเรื่องของเวลาเข้ามาเกี่ยวข้องด้วย ซึ่งตัวแปรจะถูกใช้ในส่วนที่มีการทำงานเป็นแบบลำดับคำสั่ง เช่นในฟังก์ชัน โพรซีเจอร์ และโปรเซส สำหรับการกำหนดค่าให้กับตัวแปรจะใช้สัญลักษณ์ :=

4.5 การบรรยายเชิงพฤติกรรม

การบรรยายลักษณะการทำงานของอุปกรณ์ฮาร์ดแวร์ในเชิงพฤติกรรม เป็นการบรรยายลักษณะการเปลี่ยนแปลงของข้อมูลในรูปแบบของอัลกอริธึม สำหรับการคำนวณผลลัพธ์ที่เกิดขึ้นซึ่งสืบเนื่องมาจากการเปลี่ยนแปลงสภาวะของข้อมูลที่เข้ามาโดยไม่คำนึงถึงลักษณะโครงสร้างหรือความสัมพันธ์ของอุปกรณ์ที่อยู่ภายในว่าจะเป็นอย่างไร ในหัวข้อนี้จะแสดงถึงการบรรยายเรื่องพฤติกรรม แทนการใช้โมดูลฮาร์ดแวร์รวมถึงข้อกำหนดต่างๆที่ควรรู้

4.6 โปรเซส

โปรเซสเป็นรูปแบบพื้นฐานอย่างหนึ่งที่ใช้ในการกำหนดให้กับสัญญาณ โปรเซสจะอยู่ในสถานะที่เตรียมพร้อมอยู่เสมอและจะปฏิบัติคำสั่งพร้อมๆกันกับโปรเซสอื่นๆ ที่อยู่ใน

สถาปัตยกรรมบรรยายเดียวกัน โดยโปรเซสจะปฏิบัติงานตามคำสั่งทันทีที่มีเหตุการณ์เกิดขึ้นกับสัญญาณที่อยู่ทางด้านขวามือของสัญลักษณ์กำหนดค่าให้กับสัญญาณ (<=)

```
Optional_label : PROCESS ( optional sensitivity list )
                        Declarations
                 BEGIN
                        Sequential Statement
                 END PROCESS  Optional_label;
```

รูปที่ 4.12 รูปแบบของการบรรยายแบบโปรเซส

การบรรยายโปรเซสจะเริ่มต้นด้วยคำสั่ง PROCESS และจบด้วยคำสั่ง END PROCESS ในรูปที่ 4.12 เป็นการแสดงส่วนประกอบของการบรรยายแบบโปรเซส ซึ่งประกอบด้วยส่วนของการประกาศตัวแปรที่ต้องใช้และส่วนของการปฏิบัติคำสั่งเพื่อให้ได้ผลลัพธ์ที่ต้องการ

4.7 การกำหนดตัวดำเนินการภายในโปรเซส

ตัวดำเนินการภายในโปรเซสมี 3 ชนิดคือ ตัวแปร (Variable) ไฟล์ (File) และตัวคงที่ (Constant) ซึ่งตัวดำเนินการทั้งสามชนิดนี้หากมีการประกาศไว้ในโปรเซสใดก็จะใช้ได้เฉพาะภายในโปรเซสนั้นเท่านั้น สำหรับการติดต่อกับภายนอกหรือระหว่างโปรเซสสามารถทำได้โดยใช้สัญญาณ (Signal) หรือตัวคงที่ที่ได้ประกาศไว้ในส่วนของ ARCHITECTURE ในรูปที่ 4.13 แสดงตัวอย่างการประกาศตัวกระทำภายในโปรเซส ซึ่งจะอยู่ระหว่างคำสั่ง PROCESS และ BEGIN และค่าเริ่มต้นที่ถูกกำหนดให้กับตัวดำเนินการภายในโปรเซสจะถูกนำมาใช้ในตอนเริ่มต้นของการปฏิบัติเพียงครั้งเดียวเท่านั้น ต่างกับค่าเริ่มต้นที่อยู่ภายในโปรแกรมย่อยจะถูกนำมาใช้ทุกครั้งที่มีการเรียกใช้โปรแกรมย่อยนั้นๆ

```
Clock : PROCESS ( Clk )
        VARIABLE periodic : BIT := '0';
    BEGIN
        periodic := NOT ( periodic ) AFTER 50 NS;
        Clk <= periodic;;;
    END PROCESS Clock;
```

รูปที่ 4.13 ตัวอย่างการประกาศตัวดำเนินการภายในโปรเซส

4.8 การกำหนดการกระทำภายในโปรเซส

การกระทำใดๆ ภายในโปรเซสจะเป็นการปฏิบัติแบบลำดับ (Sequential) เสมอ ซึ่งภายในโปรเซสสามารถใช้ประโยคเงื่อนไขหรือการทำซ้ำได้ เช่น IF–THEN–ELSE, CASEE–WHEN, FOR LOOP และ WHILE–LOOP ดังตัวอย่างในรูปที่ 4.14 และ 4.15

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

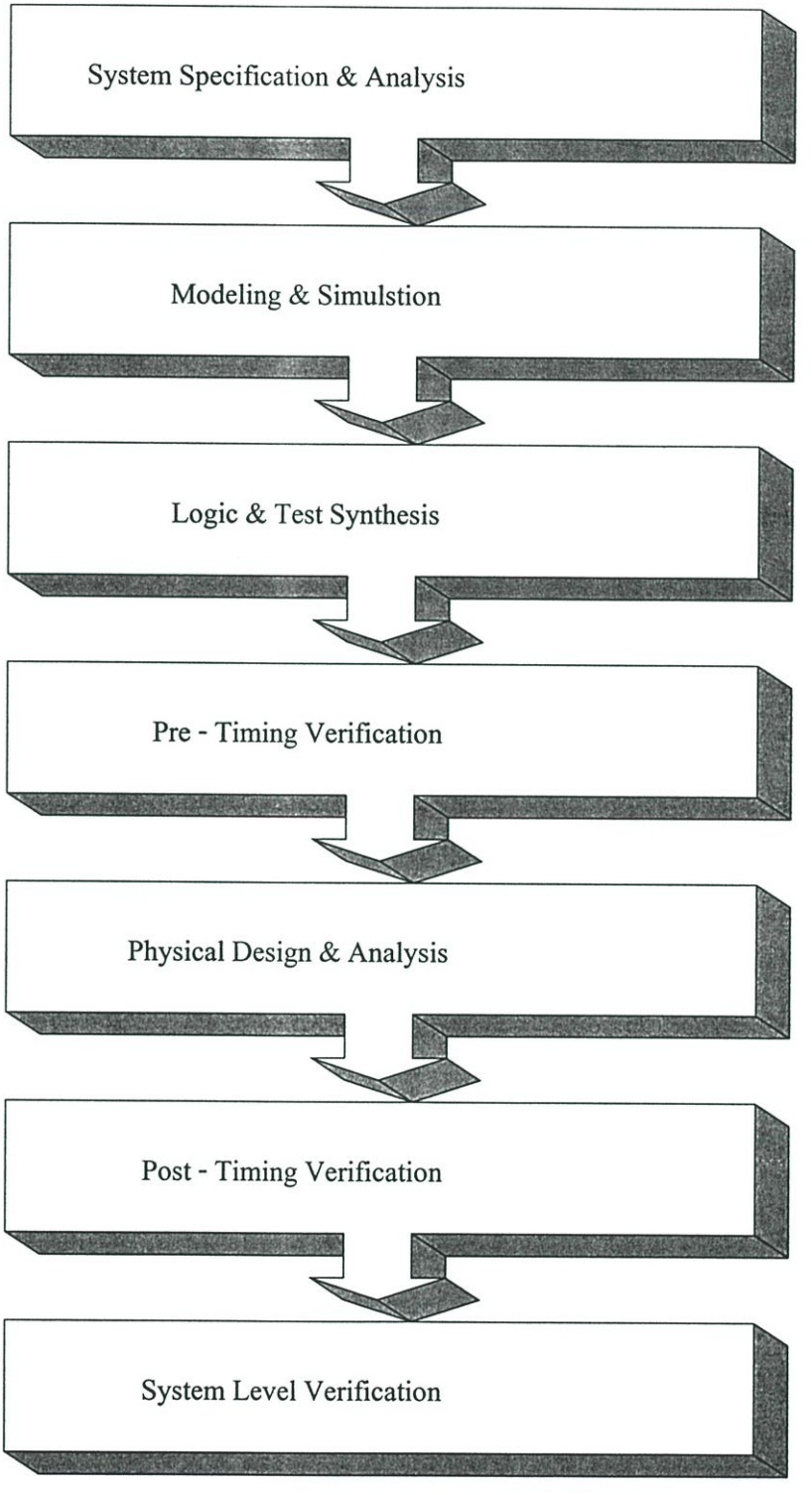
รูปที่ 4.14 เงื่อนไขการกระทำในโปรเซส

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

รูปที่ 4.15 การกระทำในโปรเซส

4.9 การออกแบบจากบนลงล่าง

ในการพัฒนาวงจรรวมดิจิตอลขนาดใหญ่ที่มีความซับซ้อน วิศวกรผู้ออกแบบมักจะมองการออกแบบให้อยู่ในรูปของบล็อกไดอะแกรมก่อนที่จะทำการวิเคราะห์ให้ลึกถึงรายละเอียดต่อไป ซึ่งภาษาวีเอชดีแอลนั้นอนุญาตให้อธิบายและวิเคราะห์การทำงานของแต่ละบล็อกรวมถึงการปรับปรุงการทำงานจากผลการวิเคราะห์เพื่อให้ได้การทำงานตามต้องการ นอกจากนี้ยังสามารถเพิ่มเติมในรายละเอียดในแต่ละขั้นตอนได้ ซึ่งหลักการนี้สอดคล้องกับหลักการออกแบบจากบนลงล่าง (Top– Down Design)นั่นเอง ถ้าทดลองเปรียบเทียบกับการออกแบบจากล่างขึ้นบน (Button–Up Design) จะเห็นได้ว่าการออกแบบจากล่างขึ้นบนจะใช้เวลาการออกแบบมากกว่า 90 % เนื่องจากเป็นการ วาดวงจรด้วยอุปกรณ์ต่างๆ(Schematic Capture)ที่ประกอบกันเข้าเป็นวงจรที่ต้องการออกแบบ ก่อนแล้วจึงทำการจำลองการทำงาน และตรวจสอบความถูกต้อง ดังนั้นการใช้ภาษาวีเอชดีแอล กับหลักการออกแบบจากบนลงล่างจึงเป็นทางออกให้กับวิศวกรให้สามารถออกแบบและพัฒนาวงจรที่มีความซับซ้อนได้มากขึ้น ทั้งยังช่วยลดเวลาและค่าใช้จ่ายในการออกแบบด้วยขั้นตอนการออกแบบดังแสดงในรูปที่ 4.16



รูปที่ 4.16 ขั้นตอนการออกแบบจากบนลงล่าง

จากรูปที่ 4.16 แสดงถึงขั้นตอนของการออกแบบจากบนลงล่าง ทั้งนี้ในทางปฏิบัติอาจมีข้อแตกต่างไปจากนี้บ้างเล็กน้อย เนื่องจากขั้นตอนของการผลิต(Implementation) สามารถกระทำได้หลายเทคโนโลยี สำหรับรายละเอียดของขั้นตอนการออกแบบจากบนลงล่างในแต่ละขั้นตอนมีดังนี้

1. สร้างข้อกำหนดของความต้องการและวิเคราะห์ระบบ เพื่อหาแนวทางความคิดและหลักการ(Idea and Concept)ในการแก้ปัญหา

2. เขียนรูปแบบของระบบที่ต้องการออกแบบโดยใช้ภาษาวีเอชดีแอลหรือภาษา HDL อื่นๆสำหรับการบรรยายพฤติกรรมการทำงาน พร้อมทั้งจำลองการทำงาน เพื่อเปรียบเทียบและตรวจสอบความถูกต้องกับข้อกำหนด

3. หลังจากที่ได้หลักการขั้นตอนพร้อมแนวความคิดที่ผ่านการตรวจสอบแล้วหลักการนี้จะถูกเพิ่มเติมในรายละเอียดลงมาเป็นลำดับขั้นที่สอง จนกระทั่งอยู่ในระดับที่จะนำไปผลิตวงจรจริงหรือสังเคราะห์ในขั้นตอนนี้เองเทคโนโลยีที่จะมารองรับวงจรออกแบบจะถูกกำหนดขึ้นและระบบช่วยการออกแบบจะสังเคราะห์วงจรที่ได้จากรูปแบบที่เขียนขึ้นให้อยู่ในรูปของวงจรที่ประกอบด้วยอุปกรณ์อิเล็กทรอนิกส์ หรือวงจรในระดับเกท และการเชื่อมต่อระหว่างกันของอุปกรณ์เหล่านั้นหรือไม่ก็อยู่ในรูปของ Netlist ที่สามารถนำไปผลิตอุปกรณ์อื่นได้

4. หลังจากการสังเคราะห์วงจรให้อยู่ในระดับเกทหรือ Netlist แล้วข้อมูลนี้จะถูกใช้สำหรับจำลองการทำงานในเรื่องความถูกต้องของฟังก์ชัน พร้อมกับนำข้อมูลที่เกี่ยวข้องกับเวลาเข้ามาประกอบการพิจารณาด้วย ซึ่งตามปกติแล้วอุปกรณ์ทางอิเล็กทรอนิกส์ทุกชิ้นจะมีเวลาหน่วงของการแพร่กระจาย(Propagation Delay Time)เสมอ ถึงแม้ว่าจะเป็นเวลาที่น้อยมากในระดับนาโนวินาทีก็ตาม แต่ถ้าภายในวงจรหนึ่งประกอบด้วยเกทของฟังก์ชันต่างๆจำนวน 10,000 เกทขึ้นไป เวลาดังกล่าวนี้จะสะสมกันมากขึ้น จนอาจทำให้การทำงานของวงจรรวมทั้งหมดผิดพลาดไป หรือไม่สามารถทำงานในย่านความถี่สัญญาณนาฬิกาที่สูงได้

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

6. ทำการตรวจสอบการทำงานและตัวแปรทางด้านเวลาทั้งหมดเพื่อความถูกต้องของวงจรเป็นครั้งสุดท้ายก่อนนำไปรวมเข้ากับอุปกรณ์อื่นๆให้เป็นระบบดิจิตอล เนื่องจากในขั้นตอนนี้วงจรที่ออกแบบ จะประกอบด้วยจุดต่อทางอินพุทและเอาท์พุต ซึ่งเป็นจุดต่อสำหรับการรับและส่งสัญญาณกับภายนอก

7. นำวงจรที่ออกแบบไว้ประกอบเข้ากับอุปกรณ์อื่นๆให้เป็นระบบที่สมบูรณ์ แล้วทำการทดสอบการทำงานทั้งระบบร่วมกับอุปกรณ์อื่นๆ อีกครั้งเพื่อควบคุมคุณภาพของผลิตภัณฑ์

4.10 โครงสร้างของ FPGA ตระกูล Virtex-E [7]

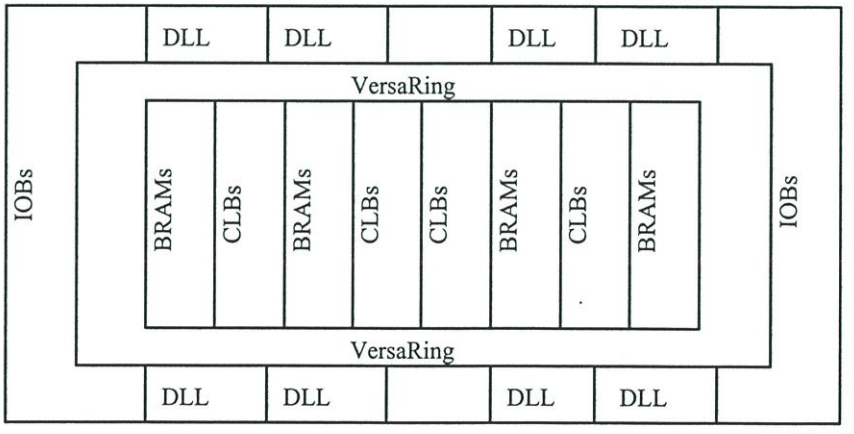
4.10.1 สถาปัตยกรรมของ FPGA ในตระกูล Virtex-E

อุปกรณ์ FPGA ตระกูล Virtex-E ของบริษัท Xilinx ได้ถูกออกแบบให้มีความยืดหยุ่นสูง โดยการให้มีโครงสร้างพื้นฐานเป็นแบบ RAM-Based ที่ประกอบไปด้วยกลุ่มของลอจิกที่สามารถทำการโปรแกรมได้ (Configuration Logic Blocks, CLB) ที่ถูกล้อมด้วยกลุ่มของอินพุทและเอาท์พุท (Input/Output Blocks, IOB) ที่มีการเชื่อมต่อภายในด้วยความเร็วที่สูงมากและสามารถใช้ประโยชน์ได้เอนกประสงค์ ส่วนในการใช้งานสามารถที่จะออกแบบให้มีการทำงานเป็นแบบซิงส์โครนัสที่มีค่าความถี่ของสัญญาณนาฬิกาได้ถึง 240 MHz ในตารางที่ 4.1 จะเป็นการแสดงถึงสมรรถนะการทำงานของวงจรพื้นฐาน(Common Circuit Function) ที่มีอยู่ใน FPGA ตระกูลนี้และค่าของเวลาที่แสดงในตารางจะเป็นค่าพารามิเตอร์ที่อยู่ในกรณีต่ำที่สุดที่จะยอมรับได้

4.10.1.1 โครงสร้างของ Virtex- E Array

โครงสร้างของ Virtex-E จะมีลักษณะเป็นเกทอาร์เรย์ ที่ผู้ใช้สามารถโปรแกรมได้ แสดงดังรูปที่ 4.17 ซึ่งประกอบไปด้วยส่วนที่สามารถกำหนดได้โดยผู้ใช้อยู่สองหลักๆคือ CLBจะเป็นส่วนที่สำหรับใช้ในการทำโครงสร้างทางลอจิกและ IOB จะเป็นส่วนที่สำหรับใช้ในการทำการเชื่อมต่อระหว่างขาของชิฟและCLB ต่างๆ

ตัวCLB จะมีการเชื่อมต่อกันภายในผ่านโครงข่ายของการเชื่อมต่อทั่วไป(General Routing Matrix ,GRM) ใน GRM จะประกอบไปด้วยอาร์เรย์ของสวิตซ์เชื่อมต่อ(Routing Switch) ที่มีตำแหน่งอยู่ระหว่างการเชื่อมต่อภายในของช่องทางการเชื่อมต่อ(Routing Channel)ในทางแนวตั้งกับแนวนอน ในแต่ละ CLB จะวางซ้อนกันเป็นชั้นๆที่เรียกว่า Versa Block ซึ่งก็จะเป็นการทำให้มีการเชื่อมต่อกันได้ระหว่าง CLB กับ GRM



รูปที่ 4.17 สถาปัตยกรรมของ FPGA รุ่น Virtex-E

ตารางที่ 4.1 สมรรถนะของวงจรพื้นฐานใน Virtex- E

Function	Bits	Virtex–E(-6)
Register – to – Register	-	-
Adder	16	4.3ns
	64	6.3 ns
Pipelined Multiplex	8 x 8	16 ns
	16x 16	5.1 ns
Address Decoder	16	3.8 ns
	64	5.5 ns
16 : 1 Multiplexer	-	6.1 ns
Parity Tree	9	3.5 ns
	18	4.4 ns
	36	5.9 ns
Chip – to – Chip	-	-

ในส่วนของ Versa Ring ที่เชื่อมต่อกับ I/O จะเป็นส่วนที่เพิ่มการติดต่อระหว่างภายในตัวชิฟกับสิ่งแวดล้อมภายนอก ซึ่งการเชื่อมต่อนี้จะเพิ่มความยืดหยุ่นของการจัดตำแหน่งขาของอุปกรณ์ได้ นอกจากนั้นได้มีการเพิ่มวงจรดังต่อไปนี้เข้ากับส่วนของ GRM ด้วยคือ

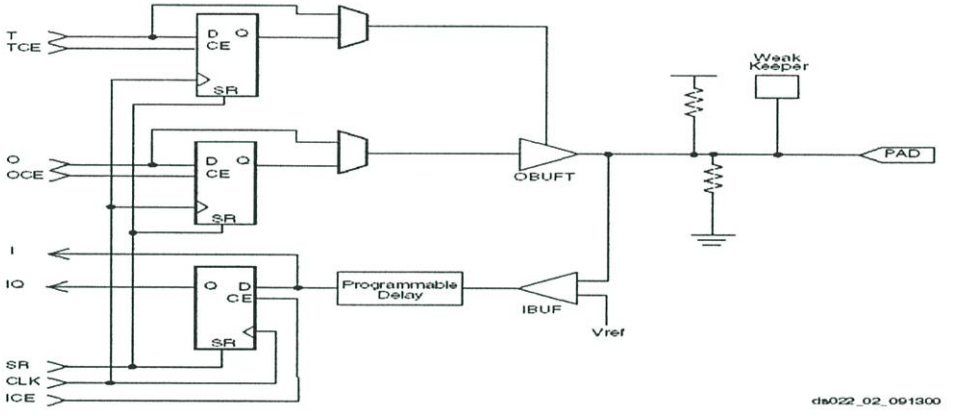
1 บล็อกของหน่วยความจำที่มีขนาด 4096 บิทในแต่ละบล็อก

2 ตัว Clock DLL ซึ่งจะใช้สำหรับการจัดแบ่งกระจายการชดเชยค่าเวลาหน่วงของสัญญาณนาฬิกา

3 บัฟเฟอร์แบบ 3 –State (BUFT) ซึ่งจะช่วยในการเป็นวงจรขับในแต่ละ CLB ที่มีการเชื่อมต่อกันอยู่ในแนวตั้งของภายในชิฟค่าของลอจิกที่ถูกเก็บไว้ในเซลล์หน่วยความจำจะเป็นตัวควบคุมสถานะการทำงานของส่วนประกอบทางลอจิก และการเชื่อมต่อภายในของส่วนประกอบภายในของตัวชิฟซึ่งค่าเหล่านี้จะถูกโหลดเข้าสู่เซลล์หน่วยความจำในตอนจ่ายกำลังงานในตัวชิฟ และสามารถที่จะทำการโหลดซ้ำได้ตามความจำเป็น เมื่อต้องการเปลี่ยนฟังก์ชันการทำงานของตัวอุปกรณ์

4.10.1.2 กลุ่มของขา IOB

ในรูปที่ 4.18 จะแสดงถึงกลุ่ม IOB ของชิฟ Virtex-E การมีรูปแบบที่สามารถให้มีการเลือกเป็นอินพุตและเอาท์พุตได้ทำให้มันสามารถที่จะถูกใช้งานได้ในหลายมาตรฐานครอบคลุมแรงดันตั้งแต่ 0.7V(มาตรฐาน HSTL I)ไปจนถึง 3.3V(มาตรฐาน LVTTL) นอกจากนั้นจะพบว่ามีฟังก์ชันการทำงานอยู่ 3 รูปแบบที่ IOB สามารถกำหนดได้ในแต่ละรูปแบบการทำงานของแลทช์ในแบบที่ใช้ขอบสัญญาณกระตุ้น แบบ D- Type ฟลิปฟลอป หรือแบบใช้ระดับสัญญาณกระตุ้นที่มีอิสระจากกัน รวมทั้งยังมีความอิสระจากกันในเรื่องของทิศทางของการไหลของข้อมูลอีกด้วยองค์ประกอบที่น่าสนใจอีกส่วนหนึ่งก็คือผู้ใช้สามารถทำ Pull-Up และ Pull- Down หรือที่ขาภายในชิฟได้โดยตรง สามารถพิจารณาแยกส่วนของอินพุตและเอาท์พุตได้



รูปที่ 4.18 กลุ่มของ IOBของ Virtex – E

4.10.1.3 ขาสัญญาณเมื่อเป็นอินพุท

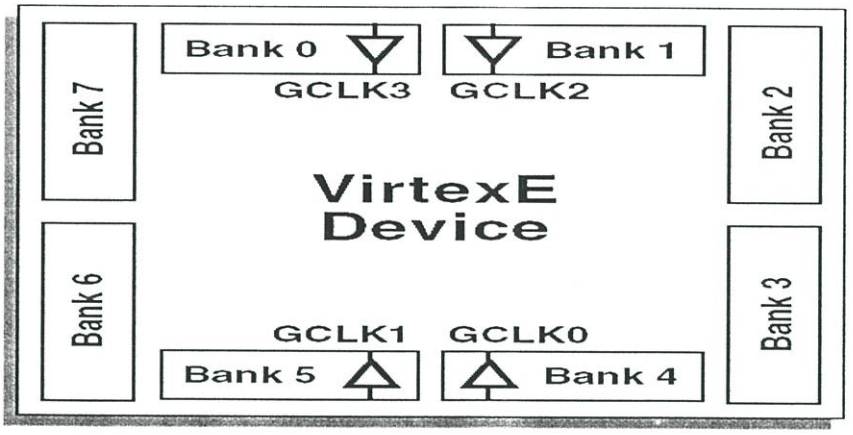
ในส่วนอินพุตของ IOB การเชื่อมต่อของสัญญาณอินพุตทำได้ทั้งในแบบที่เชื่อมต่อได้โดยตรงและแบบที่ต้องผ่านฟลิปฟลอป และในส่วนขององค์ประกอบเวลาหน่วง(Delay Element) ที่ขา D ของฟลิปฟลอป นี้จะคอยกำจัดค่าของเวลาหน่วงที่เกิดจากการเชื่อมต่อขาของอุปกรณ์(Pad-to –Pad Hold Time) ค่าของเวลาหน่วงนี้จะถูกทำให้เหมาะสมกับค่าของเวลาภายในของสัญญาณนาฬิกาใน FPGA ถ้ามีการนำไปใช้งาน ซึ่งก็จะทำให้มั่นใจได้ว่าค่าของเวลาหน่วงนี้จะเป็นศูนย์ นอกจากนั้นในแต่ละขาของอินพุตบัฟเฟอร์สามารถที่จะถูกกำหนดให้มีระดับของแรงดันเป็นไปตามมาตรฐานได้ ซึ่งในบางมาตรฐานก็ยังเปิดโอกาสให้ผู้ใช้งานสามารถที่จะป้อนแรงดันระดับเริ่มเปลี่ยน (Theshold Voltage) ให้แก่ชิฟ เพื่อเป็นแรงดันอ้างอิง Vref ในกรณีของ R-pull-up และ R- pull-down ของแต่ละอินพุตของชิฟ เมื่อถูกกำหนดให้ใช้งานหลังจากการทำคอนฟิกกูเรชันแล้วค่าของมันจะอยู่ในช่วง 50 – 100 KΩ

4.10.1.4 ขาสัญญาณเมื่อเป็นเอาท์พุท

วงจรในส่วนของเอาท์พุตจะประกอบไปด้วยบัฟเฟอร์แบบ 3–State ที่กำหนดให้ขับวงจรในแต่ละขาของชิฟ ขาเอาท์พุตนี้สามารถที่จะกำหนดให้ทำงานได้ทั้งในแบบการต่อโดยตรงกับวงจรลอจิกภายในชิฟ หรือจะต่อผ่าน IOB เอาท์พุตฟลิปฟลอปก็ได้ ส่วนขาควบคุม 3–State ของเอาท์พุตสามารถที่จะเชื่อมต่อได้โดยตรงจากลอจิกภายใน หรือต่อผ่านฟลิปฟลอปที่ได้เตรียมไว้ให้สามารถทำการซิงส์โครนัสหรือไม่ก็ได้ ส่วนในแต่ละเอาท์พุตไดรเวอร์สามารถที่จะถูกโปรแกรมได้อย่างอิสระในการที่จะกำหนดให้มีค่าของแรงดันของสัญญาณที่มีมาตรฐานแตกต่างกันได้ และสามารถที่จะขับกระแสได้สูงได้ถึง 24 mA รับ(Sink)กระแสได้สูงถึง 48 mA ในการที่จะใช้ในระบบบัสต่างๆ เพื่อให้ค่าสัญญาณรบกวนที่เกิดในระบบมีค่าต่ำที่สุด โดยส่วนใหญ่แล้วมักจะให้ระดับมาตรฐานของสัญญาณเอาท์พุตด้าน High ขึ้นอยู่กับระดับแรงดันของ Vcco จึงมีความจำเป็นในการที่จะต้องจ่ายแรงดันของ Vcco นี้ให้มีค่าใกล้เคียงกับค่ามาตรฐานที่ได้กำหนดไว้ให้มากที่สุด ส่วนวงจร Weak Keeper ที่ถูกเชื่อมต่ออยู่ในแต่ละเอาท์พุตเมื่อถูกนำมาใช้งาน จะทำให้วงจรส่วนที่คอยตรวจวัดแรงดันทำการตรวจวัดแรงดันและขับแรงดันที่ขาให้มีค่าสูงขึ้นหรือลดลงให้เหมาะสมกับระดับของสัญญาณทางด้านอินพุต ถ้าในกรณีที่ขาของมันถูกเชื่อมต่ออยู่กับสัญญาณที่มีลักษณะเป็นแหล่งจ่ายหลายชุด (Multiple- source) จะทำให้วงจร Weak- keeper ทำการคงค่าของสัญญาณในสถานะสุดท้ายไว้ถ้าทุกไดรเวอร์ถูกยกเลิกการใช้งาน ดังนั้นการรักษาระดับแรงดันที่ถูกต้องของลอจิกในวิธีการแบบนี้จึงกำจัด Bus chatter ไปได้เป็นอย่างดี เนื่องจากวงจร Weak- keeper จะใช้ IOB อินพุตบัฟเฟอร์มาทำหน้าที่เป็นตัวตรวจวัดระดับของแรงดันอินพุต ดังนั้นการกำหนดและป้อนแรงดัน Vref ที่เหมาะสม จึงเป็นสิ่งที่จำเป็นถ้าต้องการระดับแรงดันในมาตรฐานของสัญญาณในระดับนั้นๆ แต่อย่างไรก็ตามการป้อนแรงดันจำเป็นจะต้องให้เป็นไปตามกฎของ I/O banking rule คงไว้ด้วย

4.10.1.5 กลุ่มของขาสัญญาณ I/O Banking

ในบางมาตรฐานของ I/O มีความจำเป็นต้องใช้แรงดัน Vcco และหรือ Vref ซึ่งระดับแรงดันเหล่านี้จะต้องป้อนจากภายนอกเข้ากับขาของชิฟเพื่อให้กับกลุ่มของ IOB ที่เรียกว่า แบงก์ ดังนั้นการกำหนดระดับของแรงดันมาตรฐานของ I/O จึงสามารถที่ทำได้ภายในแบงก์ที่กำหนดเท่านั้น ภายในชิฟจะมี 8 แบงก์ แยกกันอยู่ด้านละ 2 แบงก์ในแต่ละด้านของชิฟ ดังรูปที่ 4.19 แต่ละแบงก์ก็จะมีขา Vcco อยู่หลายขา และทุกขาจำเป็นจะต้องถูกต่อให้มีระดับแรงดันเดียวกัน ซึ่งระดับแรงดันนี้จะถูกกำหนดโดยค่าของแรงดันมาตรฐานของเอาท์พุตที่ถูกนำมาใช้

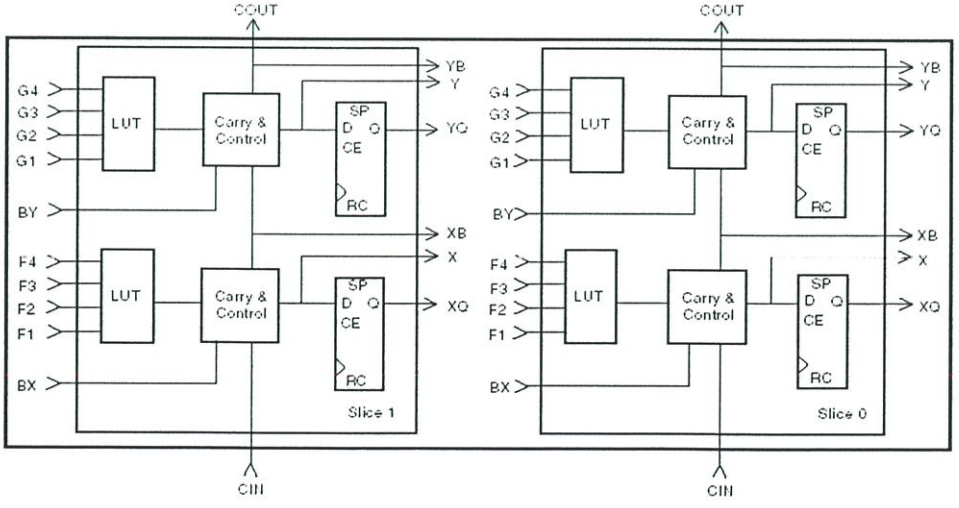


รูปที่ 4.19 กลุ่มของ I/O แบงก์ภายใน Virtex- E

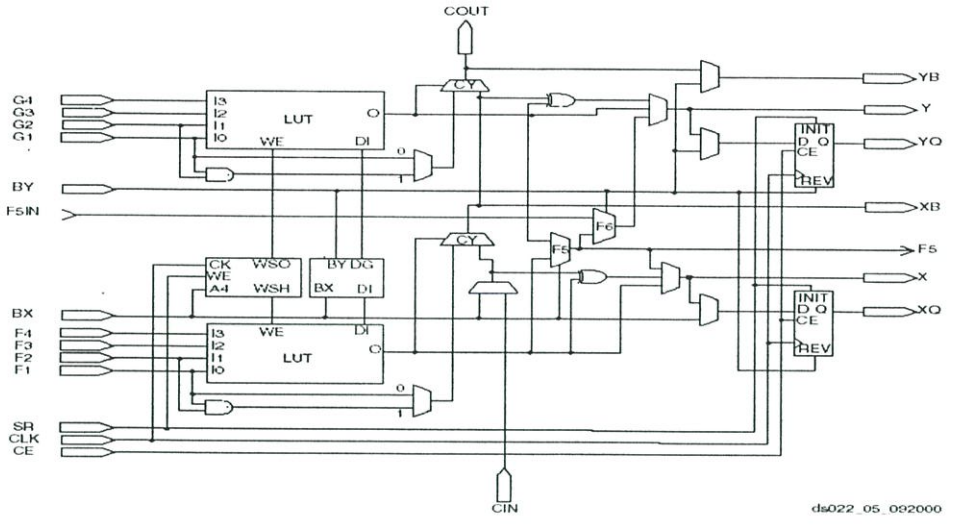
4.10.2 โครงสร้างภายในของชิฟ Virtex- E[8]

4.10.2.1 โครงสร้างของ Configurable Logic Block

พื้นฐานในการสร้างบล็อกของ VirtexE-CLB จะเป็นแบบลอจิกเซลล์(Logic Cell, LC) ภายใน LC จะประกอบไปด้วยฟังก์ชันเจนเนอเรเตอร์ขนาด 4 อินพุต อยู่ 1 ตัว Carry Logic และส่วนของการเก็บข้อมูลสัญญาณจากเอาท์พุตของฟังก์ชันเจนเนอเรเตอร์ในแต่ละ LC จะขับทั้งเอาท์พุตและขา D ของฟลิปฟลอป ในแต่ละ CLB จะประกอบไปด้วยโครงสร้างของ LC 4 ชุดอยู่ในโครงสร้างแบบ Slice 2 ชุด ดังแสดงในรูปที่ 4.20 ส่วนรูปที่ 4.21 แสดงถึงโครงสร้างรายละเอียด



รูปที่ 4.20 โครงสร้างแบบ 2 Slice CLB ของ Virtex-E



รูปที่ 4.21 รายละเอียดภายใน Slice

ของภายใน Slice นั่นก็คือจะพบว่ามี LC อยู่จำนวน 4 ชุด อยู่ภายในแต่ละ CLB ซึ่งจะประกอบรวมกันทั้งหมดแล้วจะทำให้ได้เป็นฟังก์ชันเจนเนอเรเตอร์ขนาด 5 ถึง 6 อินพุต ดังนั้นเมื่อทำการประมาณจำนวนของ System gate ที่อยู่ภายในอุปกรณ์ก็จะทำให้ได้ค่าเป็นว่าในแต่ละ CLB จะมีอยู่ประมาณ 4.5 LCs. จากรูปที่ 4.21 ตัวฟังก์ชันเจนเนอเรเตอร์ของ Virtex-E จะถูกสร้างจากตาราง Look- up Table (LUT) ขนาด 4 อินพุตซึ่งก็จะทำให้การทำงานเป็นฟังก์ชันที่ในแต่ละ LUT มีค่าเป็น 16x1 บิต ซิงโครนัส RAM หรืออาจจะนำในแต่ละ Slice ที่มี LUT อยู่สองชุดมารวมกันเพื่อสร้าง 16x2 บิต ซิงโครนัส RAM ได้ ตัว LUT ยังสามารถที่จะทำให้เป็นชิฟท์รีจีสเตอร์ขนาด 16 บิต ได้อีก ซึ่งเป็นแนวคิดสำหรับการทำ Capturing high-speed หรือ burst mode data ซึ่งโหมดนี้สามารถที่จะถูกใช้งานเพื่อการเก็บข้อมูลในการประยุกต์ใช้งานทางด้าน DSP และส่วนของการเก็บข้อมูลก็สามารถที่จะถูกกำหนดให้เป็นแบบ D–type ฟลิปฟลอป หรือเป็นแบบแลทช์ก็ได้ โดยที่ในส่วนของ D อินพุตสามารถที่จะถูกขับได้จากทั้งฟังก์ชันเจนเนอเรเตอร์ที่อยู่ภายใน Slice เอง หรือขับโดยตรงจากอินพุตของ Slice โดยการบายพาสการทำงานของฟังก์ชันเจนเนอเรเตอร์ส่วนสัญญาณนาฬิกาและสัญญาณเอ็นเนเบิลในแต่ละ Slice จะเป็นแบบซิงส์โครนัสเซ็ต และรีเซ็ต (SR และ BY) ในทางกลับกันสัญญาณทั้งสองนี้สามารถที่จะถูกกำหนดให้ทำงานเป็นแบบอะซิงส์โครนัสก็ได้เช่นกัน นอกจากนี้ตัวมัลติเพล็กเซอร์ F5 ในแต่ละ Slice จะทำให้ฟังก์ชันการทำงานเพิ่มได้เป็นเสมือนมี 5 อินพุต จากตัวมัลติเพล็กเซอร์ขนาด 4 : 1 หรือทำให้สามารถเลือกฟังก์ชันได้สูงถึง 9 อินพุต ในทำนองเดียวกันตัวมัลติเพล็กเซอร์ F6 ก็จะรวมเอาท์พุตของฟังก์ชันเจอเนอเรเตอร์ทั้ง 4 ตัวใน CLB โดยการเลือกตัวใดตัวหนึ่งจากเอาท์พุตของ F5 ทำให้สามารถใช้มัลติเพล็กเซอร์แบบ 6 อินพุต มาทำงานเป็นแบบมัลติเพล็กเซอร์ขนาด 8 : 1 หรือทำให้สามารถเลือกฟังก์ชันได้สูงถึง 19 อินพุต ในแต่ละ CLB นอกจากนั้นใน CLB ยังมี ส่วนที่ใช้สำหรับทำการป้อนผ่านโดยตรงเป็นจำนวน 2 ชุด

ต่อ Slice ทำหน้าที่เป็นทางด่วนข้อมูลแบบพิเศษหรือเป็นส่วนเพิ่มเติมของการเชื่อมต่อที่ไม่ต้องใช้ลอจิกภายใน CLB

ในส่วนการกระทำทางด้านคำนวณก็จะมีตัว Dedicate carry logic ที่มีอยู่ใน CLB จะเป็นตัวที่ทำให้ Carry ที่ได้จากการคำนวณทางคณิตศาสตร์มีความเร็วสูงมากขึ้นให้เหมาะกับการคำนวณทางคณิตศาสตร์ที่ต้องการความเร็วมากๆในแต่ละ CLB จะมีเส้นทางของ Carry อยู่ 2 โครงข่าย(ในแต่ละ Slice จะมี 1 โครงข่าย)

ส่วนของการทำประมวลผลทางคณิตศาสตร์จะประกอบไปด้วย XOR เกททำให้เกิดเป็น Full adder ขนาด 2 บิต ในแต่ละ Slice นอกจากนั้นก็จะมี AND เกทแบบพิเศษที่จะใช้สำหรับการทำวงจรคูณความเร็วสูง นอกจากนี้ส่วนของ Dedicated Carry นั้นสามารถที่จะนำมาใช้สำหรับการต่อ Cascade ให้กับฟังก์ชันเจนเนอเรเตอร์ ซึ่งก็จะทำให้ได้ฟังก์ชันของลอจิกที่ใช้งานได้กว้างขวางมากขึ้น และในแต่ละ CLB จะมีวงจรขับ BUFT อยู่ 2 ตัว ซึ่งจะใช้ขับระบบบัสภายในชิฟและในแต่ละตัวของ BUFT ก็จะมีขาควบคุมและขาอินพุตที่เป็นอิสระจากส่วนอื่นๆภายใน CLB

4.10.2.2 โครงสร้างของหน่วยความจำภายใน (Block select RAM)

ชิฟรุ่น Virtex-E จะมี RAM ให้เลือกใช้ได้ตั้งแต่ขนาด 64 Kbits จนถึง 832 Kbits ขึ้นอยู่กับเบอร์ของ FPGA เกิดจากการรวมหน่วยความจำที่เรียกว่า Block select RAM จำนวนมากไว้ด้วยกัน ซึ่งมันจะมีลักษณะของการกระจายของหน่วยความจำนี้ไปบนโครงสร้างพื้นฐานของแต่ละ CLB ในแต่ละเซลของ Block Select RAM จะเป็นแบบซิงส์โครนัสพอร์ตคู่ขนาด 4096 บิตมีสัญญาณควบคุมที่เป็นอิสระต่อกันในแต่ละพอร์ท ขนาดความกว้างของข้อมูลในแต่ละพอร์ตสามารถที่จะถูกกำหนดได้อย่างเป็นอิสระต่อกัน ซึ่งก็รวมไปถึงการเปลี่ยนขนาดความกว้างของบัสข้อมูลได้ด้วย นอกจากนั้นกลุ่ม RAM เหล่านี้ยังถูกออกแบบให้มีลักษณะเป็นการเชื่อมต่อแบบพิเศษเพื่อที่จะทำให้สามารถนำไปใช้ทำการเชื่อมต่อกับ CLB และ Block RAM อื่น ๆ ได้อย่างพอเพียงกับความต้องการ

4.10.2.3 โครงข่ายการเชื่อมต่อของขาสัญญาณอินพุตเอาท์พุต (I/O Routing)

ตัวชิฟ Virtex-E ได้มีการเพิ่มส่วนของการเชื่อมต่อของอุปกรณ์อื่นๆที่อยู่รอบๆตัวมันและมีรูปแบบเป็นการเชื่อมต่อระหว่าง CLB อาร์เรย์กับ IOB การเพิ่มการเชื่อมต่อในส่วนนี้จะเรียกว่า Versa Ring ซึ่งจะมีความสะดวกมากในการปรับตำแหน่งขาของชิฟ (Pin–swapping, Pin-locking) เช่น ในกรณีที่มีการออกแบบลอจิกใหม่ ก็จะทำให้สามารถปรับเปลี่ยนโครงสร้างภายในได้โดยยังคงขาของอุปกรณ์ไว้ในตำแหน่งเดิมได้บนแบบของแผ่นวงจรพิมพ์เดิม ทำให้ลดเวลาที่ใช้ออกสู่ตลาดทั้งนี้เพราะว่าการออกแบบแผ่นวงจรพิมพ์และระบบอื่นๆ สามารถที่จะออกแบบและทำการผลิตได้แม้กระทั่งในระหว่างที่ส่วนของลอจิกยังอยู่ในระหว่างการออกแบบอยู่ก็ตาม

4.10.2.4 โครงข่ายการเชื่อมต่อภายในแบบพิเศษ (Dedicate Routing)

ในโครงสร้างของ Virtex-E มีการเตรียมชิ้นส่วนพิเศษที่ใช้ในการเชื่อมต่อเป็นสายสัญญาณโดยเฉพาะสำหรับกลุ่มของสายสัญญาณเชื่อมต่อที่เตรียมไว้ คือ กลุ่มของสายสัญญาณชนิดพิเศษที่ใช้ในการทำ Multiple Bus ระหว่าง CLB ซึ่งจะทำให้การทำงานของระบบมีความเร็วสูงขึ้นได้และจะมีเครือข่ายในแต่ละ CLB เพื่อใช้สำหรับการรองรับการเกิดการแผ่การกระจายของสัญญาณ Carry ของ CLB ที่อยู่ติดกัน นอกจากนี้ยังมีระบบสายสัญญาณนาฬิกาแบบพิเศษที่เรียกว่า Global Clock Distribution Network เพื่อป้องกันการเกิดการเบี่ยงเบนที่รุนแรงของสัญญาณนาฬิกาได้

4.10.2.5 โครงข่ายการเชื่อมต่อของสัญญาณนาฬิกา(Clock Routing)

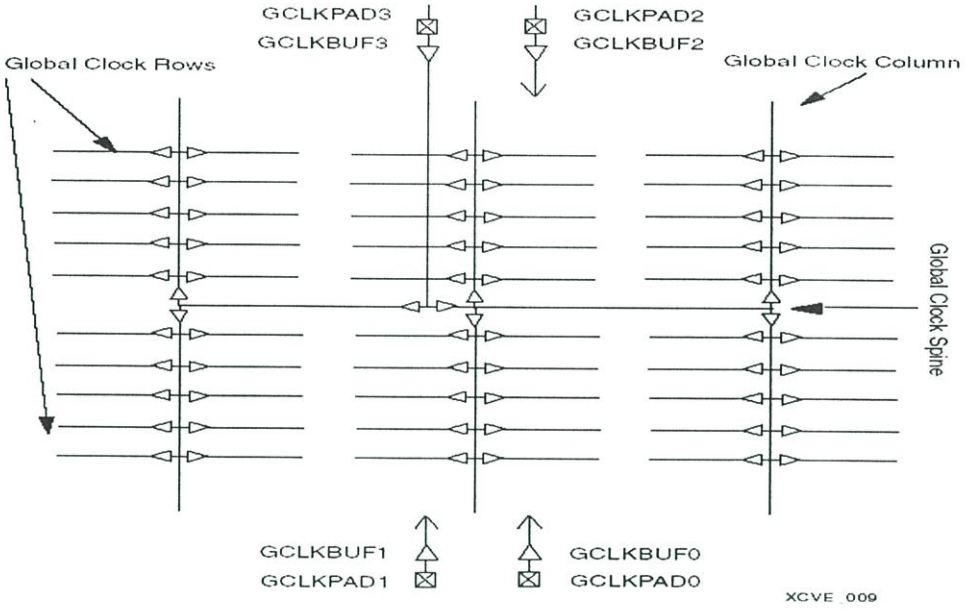
ในส่วนของการเชื่อมต่อของสัญญาณนาฬิกาจะมีการกระจายสัญญาณนาฬิกาและสัญญาณอื่นๆ ด้วยค่าของ Fan out ที่สูงผ่านไปตลอดทั้งชิฟ ซึ่งใน Virtex-E จะประกอบไปด้วยโครงข่ายการเชื่อมต่อสัญญาณนาฬิกาอยู่ 2 แถวคือ Global Routing และ Local Clock Routing โดย

1. Global Routing จะมีโครงข่ายอยู่ 4 ชุด ที่มีลักษณะทางอินพุตเป็นแบบพิเศษที่ออกแบบมาให้มีการกระจายของ Fan out ที่มีค่าสูงของสัญญาณนาฬิกาและมีค่าเบี่ยงเบนต่ำที่สุดในแต่ละโครงข่าย สามารถที่จะนำไปขับ CLB, IOB, Block, RAM ได้พร้อมกันทั้งหมด

2. Local Clock Routing กลุ่มของโครงข่ายนี้มีความยืดหยุ่นมากกว่า Global Routing เพราะว่ามันจะไม่ถูกยึดติดกับการเชื่อมต่อเพียงกับขาสัญญาณนาฬิกาเพียงอย่างเดียวจึงสามารถที่จะประยุกต์ใช้งานกับสัญญาณอื่นในกรณีที่ต้องการให้เกิดค่าเบี่ยงเบนของสัญญาณต่ำได้อีก

4.10.2.6 การกระจายของสัญญาณนาฬิกาหลัก (Global Clock Distribution)

ในชิฟVirtex-E จะมีโครงสร้างของการกระจายสัญญาณนาฬิกาเป็นแบบมีความเร็วสูงความเบี่ยงเบนต่ำผ่านโดยผ่าน Global Routing มีโครงข่ายของการกระจาย แสดงไว้ดังรูปที่ 4.22 จากรูปจะมีบัฟเฟอร์หลัก(GCLK BUF) อยู่ทั้งหมด 4 ตัวโดย 2 ตัวอยู่ด้านบนและส่วนที่เหลืออยู่ด้านล่างตรงกลางของโครงข่าย ซึ่งมันจะทำหน้าที่คอยขับสัญญาณให้โครงข่ายหลักที่จะส่งต่อไปเพื่อขับสัญญาณนาฬิกาให้กับส่วนประกอบอื่นๆต่อไป

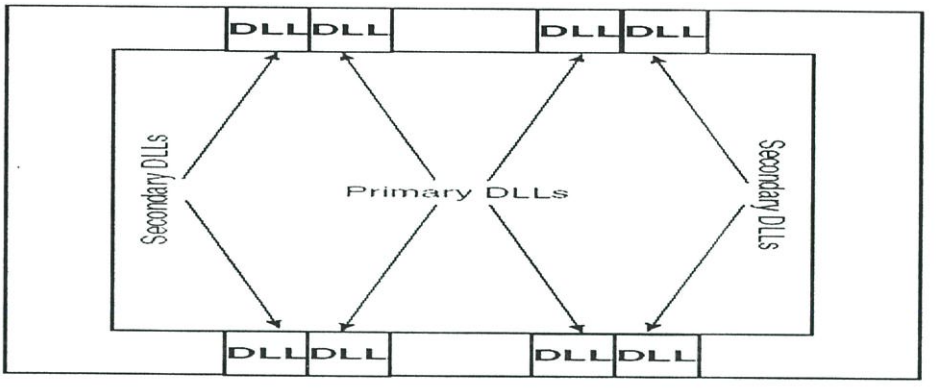


รูปที่ 4.22 โครงข่ายของการกระจายสัญญาณ Global Clock

4.10.2.7 Digital Delay – Lock Loop

ในชิฟแต่ละตัวจะมีอุปกรณ์ Delay–Locked Loop (DLL) อยู่ 8 ตัวจะอยู่ด้านบนและอีก 4 ตัวอยู่ด้ายล่างดังรูปที่ 4.23 ตัว DLL สามารถที่จะถูกนำมาใช้สำหรับการกำจัดความเบี่ยงเบนที่เกิดขึ้นระหว่างขาสัญญาณนาฬิกาของชิฟกับขาสัญญาณนาฬิกาของเซลล์ต่างๆ ที่ต่ออยู่ภายในชิฟโดยแต่ละ DLL สามารถที่จะใช้ขับโครงข่ายหลักของสายสัญญาณได้ 2 วงจร โดยทั่วไปแล้ว DLL จะทำการปรับค่าเวลาหน่วงของสัญญาณนาฬิกาโดยอัตโนมัติ มันจึงถูกใช้เป็นตัวตรวจสอบระบบสัญญาณนาฬิกาของอินพุตและทำหน้าที่แจกจ่ายสัญญาณนาฬิกาให้กับระบบ

นอกจากการกำจัดค่าของเวลาหน่วงในการแจกจ่ายสัญญาณนาฬิกาแล้วตัว DLL ได้รวมเอาส่วนของการทำการควบคุมแบบพิเศษเข้ามาด้วย เพื่อให้สามารถสร้างสัญญาณนาฬิกาได้หลายคาบโดยที่ตัว DLL จะแบ่งเฟสของสัญญาณนาฬิกาออกเป็น 4 ควอดเดรเจอร์ และสามารถที่ทำการทวีคูณความถี่ขึ้นหรือหารความถี่ลงได้ โดยปกติแล้วตัว DLL จะมีการทำงานอยู่ในรูปแบบของ Clock Mirror ดังนั้นด้วยการทำให้เกิดการขับสัญญาณจาก DLL ภายในชิฟแล้วจากนั้นก็ป้อนสัญญาณที่ได้กลับเข้าไปใน DLL อีกครั้ง จึงทำให้มันสามารถที่จะลดค่าเบี่ยงเบนของสัญญาณนาฬิกาบนบอร์ดในกรณีที่มีการใช้ชิฟบนบอร์ดมากกว่า 1 ตัวได้



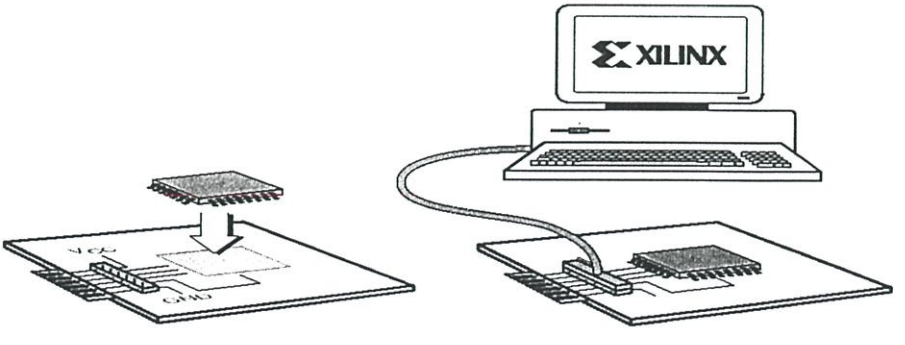
รูปที่ 4.23 การจัดวางตำแหน่งของ DLL

4.11 ปัจจัยที่ทำให้การออกแบบเอฟพีจีเอทำได้ง่ายและสะดวกรวดเร็ว

1. ผู้ออกแบบไม่จำเป็นต้องทราบถึงโครงสร้างภายในของตัวชิฟ เพียงแต่มีความรู้เกี่ยวกับขั้นตอนการออกแบบลอจิกก็เพียงพอแล้ว ต่างกับการใช้ไมโครโปรเซสเซอร์ซึ่งจำเป็นต้องศึกษาโครงสร้างภายในรวมถึงภาษา Assembly ของไมโครโพรเซสเซอร์ตัวนั้นด้วย

2. มีการออกแบบโดยใช้ภาษาในการอธิบายการทำงานของวงจรหรือ HDL (Hardware Description Language) เป็นเครื่องมือในการออกแบบ ซึ่งเป็นวิธีการที่มีความยืดหยุ่นสูงทำได้รวดเร็วและไม่จำเป็นต้องทราบถึงลักษณะของวงจรที่ต้องการว่าจะเชื่อมต่อกันอย่างไร เพียงแต่กำหนดลักษณะการทำงานให้มันจากนั้นตัวซอฟต์แวร์จะทำ Synthesis and Optimize ให้ทั้งหมด นอกจากนี้ภาษาที่ใช้ยังเป็นมาตรฐานเดียวกัน สามารถใช้ได้กับชิฟทุกตัวและทุกบริษัท

3. การโปรแกรมสามารถทำได้เองและใช้เวลาไม่นาน เพียงแค่ส่งข้อมูลผ่านสายดาวน์โหลดทางพอร์ตของคอมพิวเตอร์ก็สามารถโปรแกรมตัวชิฟขณะที่อยู่ในระบบได้ โดยไม่จำเป็นต้องถอดมาโปรแกรมข้างนอก ดังรูปที่ 4.24 และที่สำคัญสามารถโปรแกรมได้หลายครั้ง จึงทำได้ง่ายในการแก้ไข และพัฒนาโดยไม่ต้องเสียค่าใช้จ่ายเพิ่มแต่อย่างใด



รูปที่ 4.24 การโปรแกรมลงในชิฟ

4.12 การออกแบบโดยใช้ภาษาอธิบายพฤติกรรมของฮาร์ดแวร์[9]

ในการออกแบบวงจรดิจิตอลนั้นสามารถทำได้โดยการวาดวงจร(Schematic) หรือใช้ภาษาอธิบายพฤติกรรม (Hardware Description) ของฮาร์ดแวร์ในกรณีของการออกแบบวงจรด้วย ASIC ชนิด Full Custom ผู้ออกแบบจะต้องเขียนวงจรด้วย Schematic จากนั้นจะนำวงจรที่ออกแบบไว้ไปทำการจำลองการทำงาน(Simulate) ซึ่งหากผลออกมาเป็นที่พอใจก็จะต้อง Layout เป็นชั้นสาร และในการออกแบบ ASIC ชนิดนี้ผู้ออกแบบจำเป็นจะต้องทราบถึงเทคโนโลยีที่ใช้ในการสร้างด้วย หลังจากได้ Layout ที่สมบูรณ์แล้วจึงจะส่งไปเข้ากระบวนการสร้างไอซีหรือ Fabrication เพื่อสร้างเป็นซิฟไอซีออกมา

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

4.12.1 การสังเคราะห์วงจร (Logic Synthesis)[10]

ในขั้นตอนนี้จะใช้ซอฟต์แวร์ในการสังเคราะห์วงจร(Synthesis Tools) ทำการสังเคราะห์พฤติกรรมของวงจรที่ได้จากการออกแบบด้วย Schematic หรือวีเอชดีแอลซึ่งต้องทำการตรวจสอบด้วยว่าซอฟต์แวร์นั้นสนับสนุนเทคโนโลยีเอฟพีจีเอ(FPGA Library) ที่ต้องการหรือไม่ ตัวอย่างเช่น เอฟพีจีเอ ของบริษัท XILINX และบริษัท ALTERA จะมีซอฟต์แวร์หลายตัวที่สามารถใช้ได้ ซึ่งในขั้นตอนนี้ซอฟต์แวร์สังเคราะห์วงจรจะทำการแปลงโค้ดวีเอชดีแอล และทำการ Optimize เพื่อให้ได้วงจรตามเทคโนโลยีที่เลือกใช้ ในการสังเคราะห์วงจรนั้นวงจรระดับเกท(Gate Level) จะไม่เหมาะสมกับโครงสร้างที่มีอยู่ในอุปกรณ์เอฟพีจีเอ ดังนั้นในการ Optimize ซอฟต์แวร์สังเคราะห์วงจรจะต้องทำการ Optimize ให้ได้เป็นวงจรที่ประกอบด้วยกลุ่มของลอจิกที่เหมาะสมกับอุปกรณ์เอฟพีจีเอนั้นๆ จึงทำให้ผลที่ได้มีประสิทธิภาพและในขั้นตอนการสังเคราะห์วงจรนี้ ผู้ออกแบบสามารถกำหนดข้อบังคับในเรื่องของพื้นที่(Area) หรือกำหนดชนิดและตำแหน่งของ I/O ซึ่งข้อบังคับเหล่านี้จะถูกนำไปใช้งานในขั้นตอน Optimize เพื่อให้วงจรที่ได้เป็นไปตามที่กำหนด ส่วนสำคัญในการ Optimize คือการเทียบ(Mapping)โมเดลให้เข้ากับเทคโนโลยีที่ใช้เพื่อให้ได้วงจรที่เหมาะกับโครงสร้าง และสถาปัตยกรรมภายในอุปกรณ์เอฟพีจีเอ เมื่อทำการสังเคราะห์วงจรเสร็จแล้วซอฟแวร์การสังเคราะห์วงจรก็จะมีการรายงานผลว่าโมเดลที่ออกแบบไปนั้นเป็นอย่างไร

เช่น มีค่าความล่าช้าของเวลา (Delay) เท่าใด ใช้ทรัพยากรต่างๆในเอฟพีจีเอ อะไรบ้าง เมื่อมาถึงขั้นตอนนี้ผู้ออกแบบก็จะทราบว่าโมเดลเป็นไปตามข้อบังคับหรือไม่ ถ้าไม่ก็สังเคราะห์ใหม่จนกว่าจะเป็นไปตามที่กำหนด

4.12.2 การแบ่งวงจร (Partitioning)

ขั้นตอนนี้เป็นการแบ่งวงจรที่ได้จากการสังเคราะห์ เป็นส่วนย่อยๆ สำหรับลงใน Configurable Logic Blocks(CLBS), Input / Output Blocks (IOBS) หรือองค์ประกอบอื่นๆภายในอุปกรณ์เอฟพีจีเอ ส่วนของ CLBS จะเป็นส่วนที่บรรจุลอจิกเกทต่างๆเอาไว้และส่วนของ IOBS จะเป็นส่วนของอินพุตหรือเอาท์พุต ที่สามารถโปรแกรมได้ สำหรับเกณฑ์ที่ใช้ในการแบ่งคือ ให้แต่ละส่วนที่จะแยกออกจากกันมีจำนวนสัญญาณที่เชื่อมต่อระหว่างกันน้อยที่สุดเท่าที่จะทำได้เพื่อลดความหนาแน่นในตอนทำการเชื่อมต่อสัญญาณ(Routing) ในขั้นตอนนี้จะใช้ซอฟต์แวร์ทำโดยซอฟต์แวร์จะเทียบส่วนประกอบของวงจรเช่น เกท(Gate) , ฟลิปฟลอบ(Flip–flop) ลงในทรัพยากรต่างๆที่มีอยู่ในอุปกรณ์เอฟพีจีเอ

หลังจากทำขั้นตอนนี้เสร็จแล้วผู้ออกแบบสามารถที่จะทราบว่าวงจรใช้จำนวนทรัพยากรภายในอุปกรณ์เอฟพีจีเอไปเท่าไร ส่วนข้อมูลทางเวลานั้นผู้ออกแบบจะทราบเฉพาะความล่าช้าของเวลาภายในแต่ละส่วนเท่านั้น หรือที่เรียกว่าความล่าช้าของเวลาลอจิก(Logic Delay) ส่วนซอฟต์แวร์จะรวมเอาซอฟต์แวร์ย่อยอื่นๆ อีก เพื่อให้การทำ PPR (Partitioning Placement & Routing) เป็นไปอย่างต่อเนื่อง

4.12.3 การวางอุปกรณ์ (Placement)

ขั้นตอนนี้เป็นการเลือกที่ตั้งของแต่ละส่วนของวงจรที่ผ่านการแบ่งวงจร (Partitioning) มาแล้วว่าควรจะอยู่ ณ. ตำแหน่งไหนในอุปกรณ์เอฟพีจีเอ เพื่อให้ได้ผลลัพธ์ดีที่สุดเช่น วงจรส่วนไหนควรอยู่ใกล้กัน เพื่อจะได้ค้นหาเส้นทางได้ (Route) ง่ายหรือช่วยลดความล่าช้าของเวลา จะเห็นได้ว่าตำแหน่งภายในอุปกรณ์เอฟพีจีเอนั้นมีความสำคัญเพราะถ้าจัดวางวงจรลงในตำแหน่งที่ไม่เหมาะสมแล้วจะทำให้ความล่าช้าของเวลาเพิ่มขึ้นหรือ Router ทำการค้นหาเส้นทางสัญญาณได้ไม่หมด

การวางอุปกรณ์ที่ดีควรวางส่วนต่างๆ ให้อยู่ใกล้กันโดยเฉพาะส่วนที่มีการเชื่อมต่อสัญญาณด้วยกันนอกจากนั้นการกำหนดตำแหน่งขา I/O(I/0 pin) ตามตำแหน่งขา I/O ของเอฟพีจีเอบนแผ่น PCB ก็จะมีผลโดยตรงทันทีคือ ซอฟแวร์จะวาง I/O ลงในตำแหน่งที่ผู้ออกแบบกำหนด ซึ่งบางครั้งตำแหน่งที่กำหนดไปไม่เหมาะสม ดังนั้นการกำหนดขา I/O ควรกำหนดตำแหน่งให้เหมาะสมหรือไม่ก็ให้ซอฟต์แวร์จัดการเอง

4.12.4 การเชื่อมต่อสัญญาณ (Routing)

ในขั้นตอนนี้เป็นการเชื่อมต่อสัญญาณระหว่างองค์ประกอบต่างๆภายในอุปกรณ์เอฟพีจีเอ ขั้นตอนนี้ จะทำต่อเนื่องจากการวางอุปกรณ์ ในกรณีที่ทำการวางอุปกรณ์ไว้ไม่ดีซอฟต์แวร์ก็จะทำการเชื่อมต่อสัญญาณได้ไม่หมด (เนื่องจากจำนวนทรัพยากรสำหรับเชื่อมต่อสัญญาณนั้นมีอยู่จำกัด) หรือเกิดความหน่วงเกินค่าที่กำหนดในข้อบังคับ

ผู้ออกแบบสามารถทำขั้นตอนนี้ได้โดยใช้ซอฟต์แวร์หรือผู้ออกแบบจะทำการเชื่อมต่อสัญญาณด้วยตนเองก็ได้ แต่ทางที่ดีควรใช้ซอฟต์แวร์ทำดีกว่า นอกจากนั้นการกำหนดข้อบังคับทางเวลา จะช่วยให้ผลที่ได้จากการเชื่อมต่อสัญญาณดีขึ้นได้

4.12.5 ความล่าช้าของเวลา (Delay)

ในการทำเอฟพีจีเอนั้นความล่าช้าของเวลาที่เกิดขึ้นเป็นความล่าช้าของเวลาที่เกิดจากการวางตำแหน่ง(Layout) ของอุปกรณ์ซึ่งผู้ออกแบบไม่สามารถเข้าไปแก้ไขได้ แต่สามารถทำให้มีความล่าช้าของเวลาน้อยที่สุดได้ สำหรับความล่าช้าของเวลาที่เกิดขึ้นนั้นแยกได้เป็นสองประเภทคือ

- ความล่าช้าของเวลาลอจิก (Logic Delay) เป็นความล่าช้าของเวลาภายในองค์ประกอบของอุปกรณ์เอฟพีจีเอเอง

- ความล่าช้าของเวลาที่เกิดจากการเชื่อมต่อสัญญาณ (Routing Delay) เป็นความล่าช้าของเวลาที่เกิดจากการเชื่อมต่อสัญญาณระหว่างองค์ประกอบภายในอุปกรณ์เอฟพีจีเอ

โดยปกติแล้วค่าความล่าช้าของเวลาลอจิกจากการเชื่อมต่อสัญญาณมักจะมีค่ามากกว่าค่าความล่าช้าของเวลาลอจิก ดังนั้นในการวางอุปกรณ์ และเชื่อมต่อสัญญาณ ผู้ออกแบบควรกำหนดข้อบังคับของเวลาเพื่อให้ ซอฟต์แวร์ได้ทำงานอย่างมีประสิทธิภาพเพิ่มขึ้น และเพื่อให้ผลลัพธ์ที่ดีขึ้นค่าความล่าช้าของเวลาที่ได้หลังจากการวางอุปกรณ์ และเชื่อมต่อสัญญาณแล้วจะมีค่าความล่าช้าของเวลาที่ค่อนข้างแน่นอน ซึ่งผู้ออกแบบสามารถทราบได้ว่าโมเดลที่ออกแบบนั้น เป็นไปตามข้อกำหนดหรือไม่

4.12.6 การจำลองการทำงานของวงจร (Simulation)

ในขั้นตอนนี้เป็นขั้นตอนที่สำคัญอีกขั้นตอนหนึ่ง เพราะเป็นขั้นตอนที่ผู้ออกแบบตรวจสอบฟังก์ชันการทำงานของโมเดลว่าถูกต้องหรือไม่ มีข้อผิดพลาดตรงไหนเพื่อจะได้ทำการแก้ไขให้ถูกต้อง ในขั้นตอนนี้จะมีซอฟต์แวร์ที่ใช้สำหรับทำการจำลองการทำงานของวงจรที่ใช้อยู่ เช่น Model Sim ของบริษัท Model Technology หรือ Max Plus II ของบริษัท Altera ตลอดจน ISE 4.2I ของบริษัท Xilinx ในการจำลองการทำงานของวงจรควรทำทุกครั้งหลังจากที่มีการทำแต่ละขั้นตอนหลักเสร็จแล้ว เพื่อจะได้ทราบว่าข้อผิดพลาดของโมเดลเกิดขึ้นตอนไหน จะได้แก้ไขข้อผิดพลาด

ตรงขั้นตอนนั้นๆ ได้ทันที ไม่ต้องมาคอยตรวจหาขั้นตอนที่ทำให้เกิดข้อผิดพลาด นั่นคือการทำการจำลองการทำงานของวงจร ต้องทำทั้งหลังการเขียนโค้ด, การสังเคราะห์วงจร และการทำ PPR การจำลองการทำงานของวงจรหลังจากที่เขียนโค้ดเสร็จแล้วนั้น ผู้ออกแบบสามารถทราบได้เฉพาะโมเดลทำงานถูกต้องหรือไม่เท่านั้น (Function Test) ยังไม่สามารถตรวจสอบการทำงานในเชิงเวลาได้ถูกต้อง ในการจำลองการทำงานของวงจรหลังจากที่สังเคราะห์เป็นวงจรแล้ว เพื่อตรวจสอบว่าฟังก์ชันการทำงานยังคงถูกต้องหรือไม่ และค่าความล่าช้าของเวลาที่เกิดขึ้นเป็นไปตามข้อบังคับหรือไม่มีข้อผิดพลาดเกิดขึ้นหรือไม่ถ้ามีจะแก้ไขให้ถูกต้อง

ในการจำลองการทำงานของวงจรหลังจากที่ทำการวางอุปกรณ์ การเชื่อมต่อสัญญาณ (Post Layout Simulation) แล้วก็มีความสำคัญเช่นกัน เพราะผลที่ได้จากการจำลองการทำงานของวงจรในตอนนี้จะเป็นผลลัพธ์ของโมเดลเท่านั้น ซึ่งผู้ออกแบบนอกจากจะตรวจสอบฟังก์ชันการทำงานแล้วยังต้องตรวจสอบว่าวงจรรวมสามารถใช้งานที่ความถี่สูงสุดเท่าไรนั่นเอง ในการจำลองการทำงานของวงจร ควรใช้ซอฟต์แวร์ตัวเดียวกันตลอดเพื่อจะได้เปรียบเทียบผลที่ได้จากขั้นตอนต่างๆ

4.12.7 การโปรแกรมอุปกรณ์เอฟพีจีเอ (Configuration)

หลังจากที่โมเดลผ่านขั้นตอนต่างๆจนกระทั่งผ่านการทำ PPR (Partitioning, Placement &Routing) แล้วนั้น ถึงตอนนี้ก็สามารถที่จะดาวน์โหลด (Download) ลงในอุปกรณ์เอฟพีจีเอได้แล้ว ในการดาวน์โหลดนี้ก่อนอื่นต้องแปลงแบบวงจรรวมที่ได้เป็นข้อมูลวงจร (Configuration Data) ซึ่งอยู่ในรูปของบิตสตรีม(Bit Stream) ก่อนแล้วจึงดาวน์โหลดลงไปเพื่อให้อุปกรณ์เอฟพีจีเอ มีฟังก์ชันการทำงานตามโมเดลที่ผู้ออกแบบต้องการ ซึ่งในขั้นตอนนี้จะใช้วิธีที่แตกต่างกันออกไปสำหรับอุปกรณ์เอฟพีจีเอของแต่ละบริษัทผู้ผลิตคือ ในกรณีที่เป็นอุปกรณ์เอฟพีจีเอชนิดที่ต้องโปรแกรมโดยวิธี SRAM เท่านั้น ในการใช้งานผู้ออกแบบจะต้องเก็บข้อมูลวงจรไว้ในหน่วยความจำประเภท EPROM หรือ Serial PROM ด้วยเพื่อจะใช้งานสะดวกขึ้น คือในการใช้งานโมเดลครั้งต่อไปไม่ต้องดาวน์โหลดข้อมูลวงจรจากเครื่องคอมพิวเตอร์อีก เพราะมีข้อมูลวงจรเก็บอยู่ในหน่วยความจำอยู่แล้ว แต่กรณีที่อุปกรณ์เอฟพีจีเอเป็นชนิดที่โปรแกรมโดยใช้วิธี EPROM หรือ Anti Fuse ก็ไม่จำเป็นต้องมีหน่วยความจำสำหรับเก็บข้อมูลวงจร เพราะว่าอุปกรณ์เอฟพีจีเอชนิดนี้เมื่อดาวน์โหลดข้อมูลวงจรลงไปข้อมูลที่ดาวน์โหลดลงไปก็ยังคงอยู่ในอุปกรณ์เอฟพีจีเอและครั้งต่อไปก็ใช้งานโมเดลที่ออกแบบไว้ได้ทันที

4.13 เครื่องมือสำหรับการออกแบบด้วยอุปกรณ์เอฟพีจีเอ

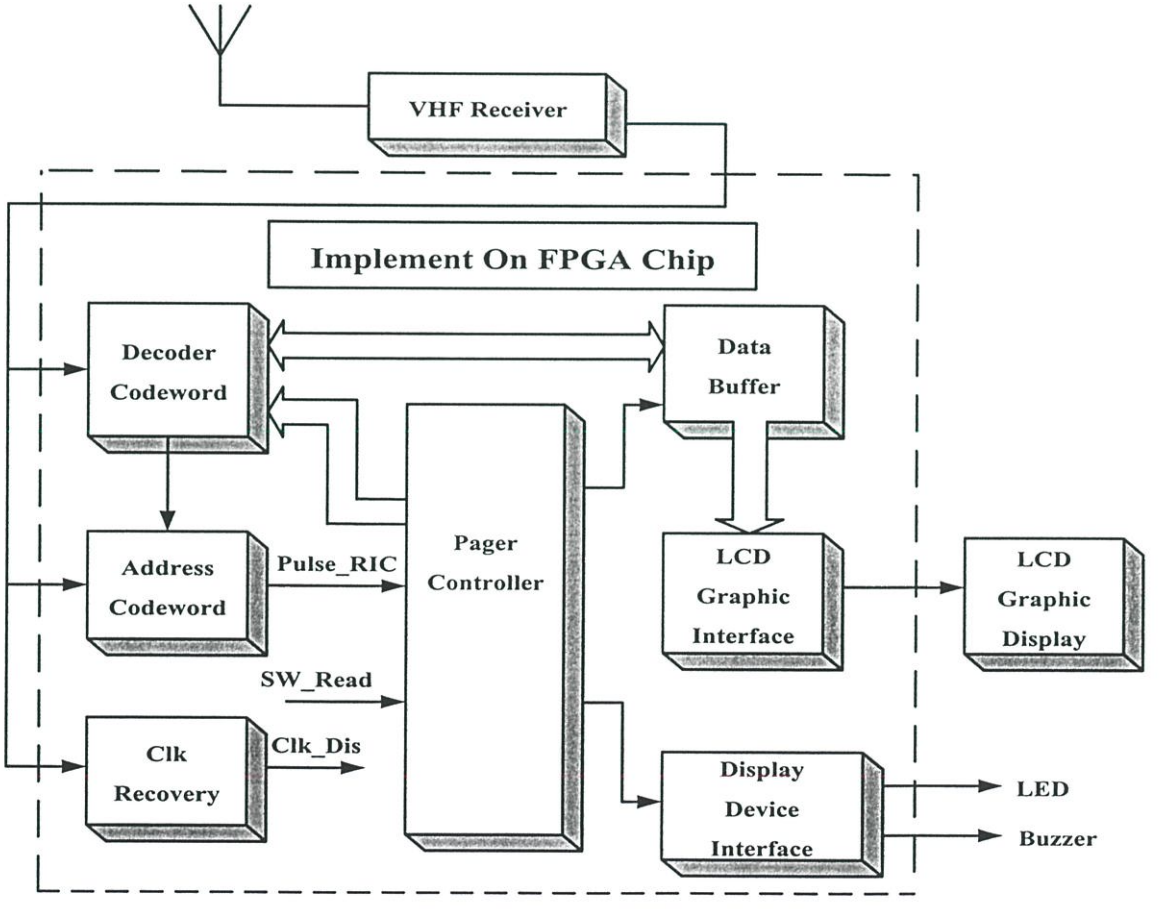
จะเห็นได้ว่าการออกแบบเพื่อทำเอฟพีจีเอนั้นทำได้สะดวกกว่า ASIC มาก เพราะใช้เวลาน้อยกว่ามากด้วย ส่วนสำคัญที่ใช้ในการออกแบบเอฟพีจีเอคือซอฟต์แวร์ที่ใช้ตั้งแต่เขียนโค้ดอธิบายฮาร์ดแวร์จนกระทั่งดาวน์โหลดลงในอุปกรณ์เอฟพีจีเอ ซึ่งซอฟต์แวร์ที่ใช้ต้องเป็นซอฟต์แวร์ที่งานต่อเนื่องกันได้ สำหรับซอฟต์แวร์ที่ใช้ทำการจำลองการทำงานของวงจรนั้น ต้องสามารถใช้งานต่อเนื่องกับซอฟต์แวร์ที่ใช้ทั้งระบบ เพราะโมเดลที่ได้จากการทำขั้นตอนต่างๆ (ด้วยซอฟต์แวร์ต่างๆ) ต้องเอามาจำลองการทำงานได้ และในการจำลองการทำงานของวงจรควรใช้ซอฟต์แวร์ตัวเดียวกันตลอดทั้งระบบ เพื่อจะได้เปรียบเทียบผลได้ง่ายในอดีตซอฟต์แวร์ส่วนใหญ่จะใช้งานอยู่บนคอมพิวเตอร์สมรรถนะสูงอย่างเวิร์กสเตชันเตชั้น(Workstation) ในปัจจุบันมีการพัฒนาซอฟต์แวร์ที่ใช้บนพีซี (PC) มากขึ้นซึ่งสามารถลดค่าใช้จ่ายในด้านอุปกรณ์คอมพิวเตอร์ลงได้

บทที่ 5

การพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว โดยใช้ FPGA

การพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัวซึ่งออกแบบโดยใช้อุปกรณ์ เอฟพีจีเอ ซึ่งพัฒนาเป็นชิฟเฉพาะงานในส่วนของวงจรดิจิตอล ซึ่งทำงานอยู่ภายในชิฟเดียว ก็เพื่อลดความยุ่งยากในการออกแบบวงจรซึ่งทำให้ได้ฮาร์ดแวร์ให้มีขนาดที่เล็กลงตลอดจนความเร็วในการทำงานที่เร็วขึ้น โดยประกอบด้วยภาคถอดรหัสข้อมูล ภาคถอดรหัสหมายเลขเรียกขาน ภาคกู้สัญญาณนาฬิกา ภาคจัดเรียงข้อมูล ภาคเชื่อมต่อแอลซีดีกราฟฟิกดีสเพลย์ ภาคเชื่อมต่ออุปกรณ์แสดงผล และภาคควบคุมระบบ ซึ่งจะมีโครงสร้างการทำงานแสดงดังในรูปที่ 5.1

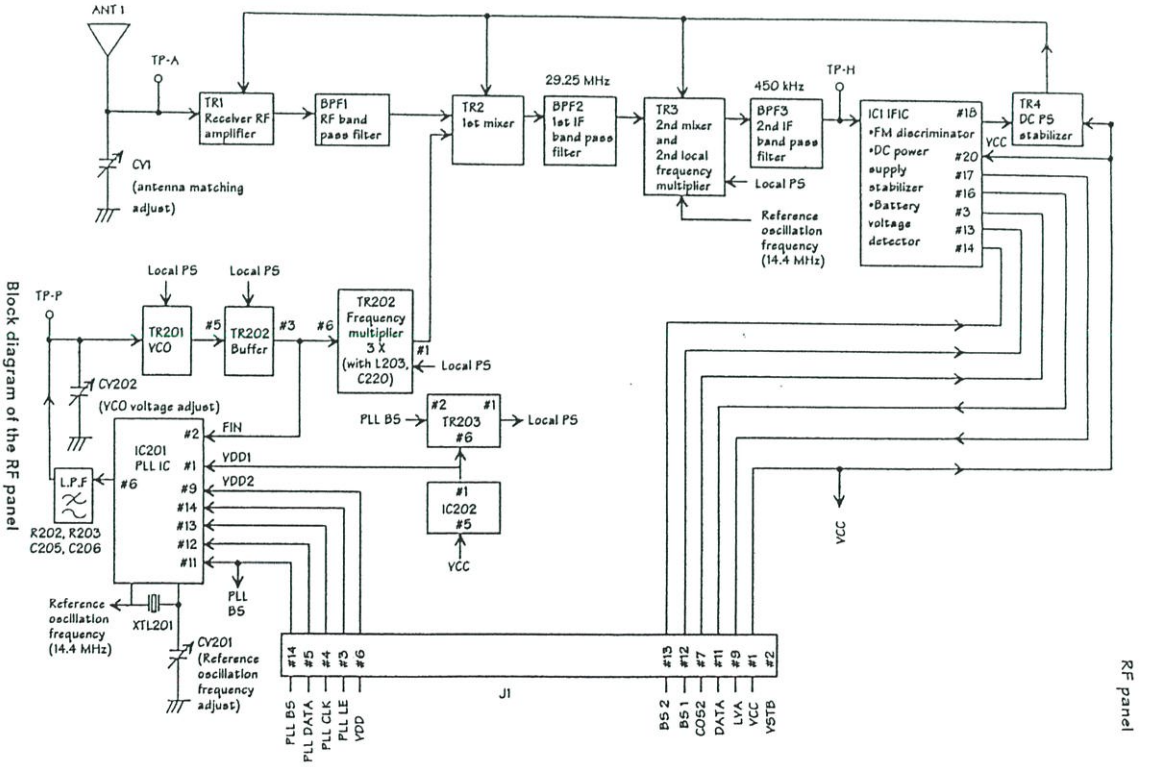
5.1 ลักษณะโครงสร้างโดยรวมของระบบ[11]



รูปที่ 5.1 บล็อกไดอะแกรมของระบบตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว

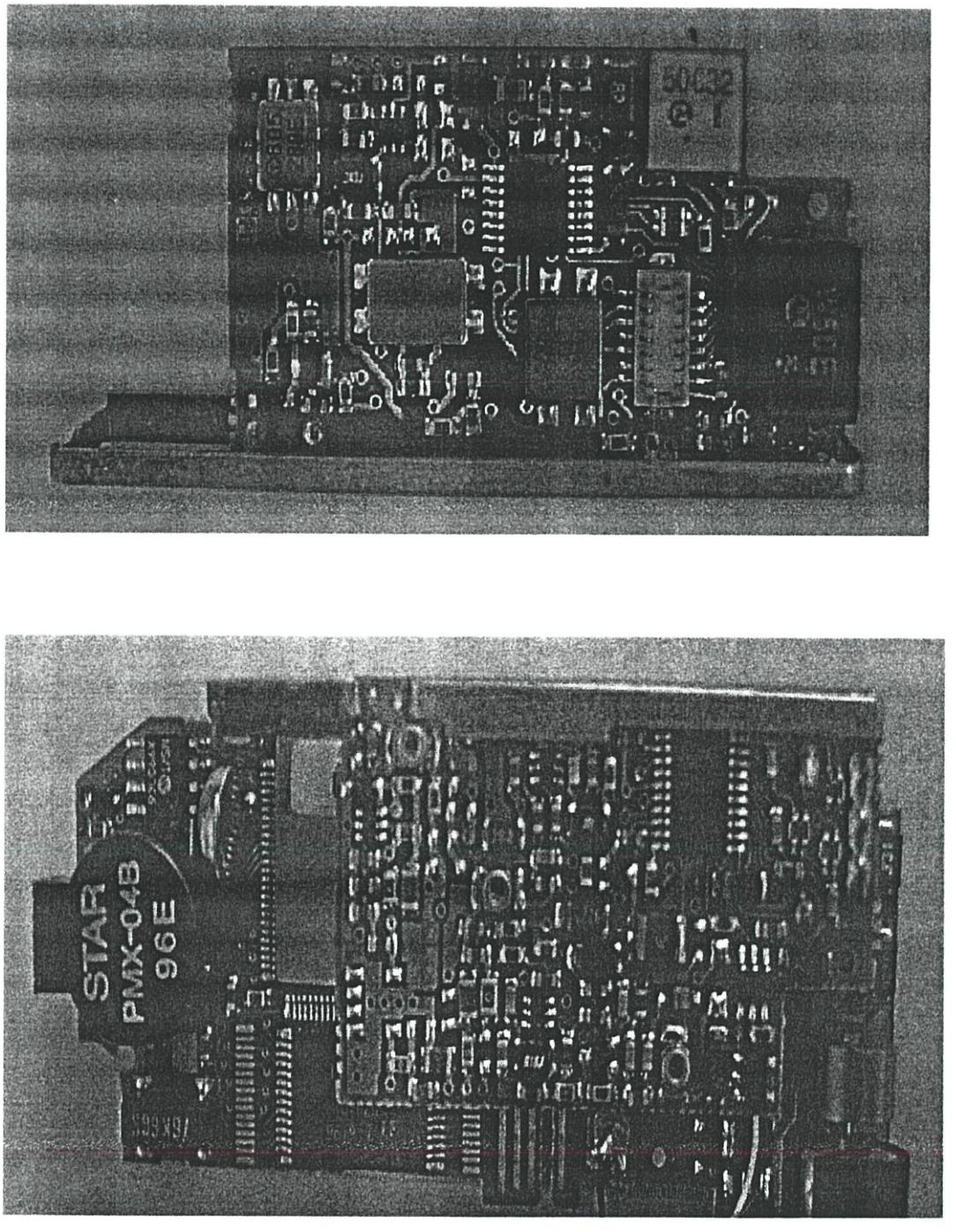
5.2 ภาครับสัญญาณวิทยุติดตามตัว(RF Receiver)[12]

ภาครับสัญญาณวิทยุติดตามตัวซึ่งการทำงานของวงจรจะใช้เครื่องรับเพจเจอร์ของบริษัท NEC Corporation รุ่น PA4PY-26T ใช้ความถี่ 279.550 MHz เป็นระบบ POCSAG โดยมีความเร็วในการรับส่งข้อมูลเท่ากับ1200 บิตต่อวินาที และหมายเลขเรียกขาน(Plugcode)คือ 2061428 ตลอดจนหมายเลขเพจเจอร์ที่ใช้ในการติดต่อคือหมายเลข 3002638 เรียก 141,142โดยผู้ให้บริการคือ บริษัท ทศท.คอร์ปอเรชั่นจำกัด(มหาชน) ซึ่งบล็อกไดอะแกรมการทำงานของวงจรประกอบไปด้วย RF Section (Matching Circuit, RF Amplifier, Band Pass Filter), 1ST Mixer Section (1ST Mixer 1st IF Band Pass Filter), 2nd Mixer Section(2nd Mixer, 2nd IF Band Pass Filter), 1st Local Oscillator Section(VCO Circuit, Multiplier Circuit, PLL Circuit), PLL Local Section(PS Circuit), Reference Crystal Circuit, 2nd Local Oscillator Section(Multiplier Circuit), และ IF IC(Amplifier of 2nd IF Signal, Low Pass Filter(LPF), Amplifier Limiter, Battery Alarm Circuit, Constant Voltage Circuit of RF Section(RF PS), Quadrature Phase Detection, FSK Wave Form Shaping, Battery Saving(BS) ดังแสดงดังรูปที่ 5.2



รูปที่ 5.2 บล็อกไดอะแกรมภาครับสัญญาณวิทยุ (VHF Receiver)

จากบล็อกไดอะแกรมของภาครับสัญญาณวิทยุของเพจเจอร์ ซึ่งจากรูปที่ 5.2 จะเห็นได้ว่าจะมีคอนเนตเตอร์ที่มีสัญญาณเอาท์พุตออกมาซึ่งเป็นสัญญาณเบสแบนด์(NRZ) โดยการต่อใช้งานจะทำการต่อสัญญาณจุดที่ชื่อว่า DATA ของคอนเนตเตอร์ออกไปยังตัวถอดรหัสเพจเจอร์โดยใช้เอฟพีจีเอ ที่ทำการออกแบบ โดยภาครับสัญญาณวิทยุของเพจเจอร์จริงจะแสดงดังรูปที่ 5.3

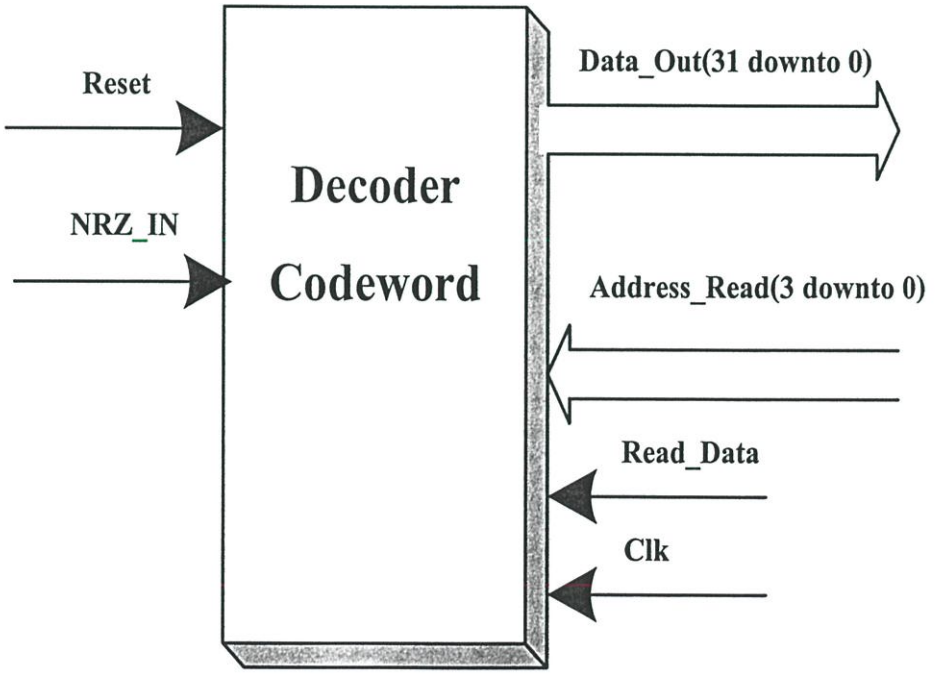


รูปที่ 5.3 ภาครับสัญญาณวิทยุ (VHF Receiver)

5.3 ภาคถอดรหัสเพจเจอร์ (Decoder Codeword)

5.3.1 บล็อกไดอะแกรมภาคถอดรหัสเพจเจอร์

ทำหน้าที่ กู้สัญญาณนาฬิกาเบสแบนด์(NRZ) ที่รับเข้ามาเพื่อทำหน้ากู้และกำเนิดสัญญาณนาฬิกาให้ซิงส์โครไนซ์กับข้อมูลที่รับเข้ามาเพื่อควบคุมการบันทึกข้อมูลทั้งหมดในระบบ สัญญาณเอาท์พุตมาจากภาครับสัญญาณวิทยุซึ่งเป็นสัญญาณเบสแบนด์(NRZ)ให้เก็บอยู่ในหน่วยความจำ (RAM) โดยเก็บข้อมูลครั้งละ 1 รหัสคำ(32 บิต) และจะถูกนำมาแสดงผลโดยแอลซีดีกราฟฟิกดิสเพลย์ (LCD Graphic Display) ซึ่งภาคถอดรหัสเพจเจอร์จะประกอบด้วย วงจรกู้สัญญาณนาฬิกา (Clk Recovery) ทำหน้าสร้างสัญญาณนาฬิกาจากสัญญาณเบสแบนด์(NRZ) ที่รับเข้ามาเพื่อจ่ายให้กับภาคถอดรหัสข้อมูลทั้งหมดโดยที่ วงจรตรวจสอบเฟรมซิงส์โครไนซ์ (Frame Sync Detect) และวงจรหน่วงเวลาเฟรมซิงส์ (Frame Sync Delay) จะทำการตรวจสอบรหัสคำซิงส์โครไนซ์และหน่วงเวลาในการเก็บข้อมูลเข้าไปในหน่วยความจำโดยมีวงจรชิฟรีจีสเตอร์แบบอนุกรมเป็นขนาน (Serial To Parallel Register) ทำหน้าที่ในการจัดเรียงข้อมูลเก็บในหน่วยความจำ (RAM) วงจรนับแอดเดรส (Count Address) และวงจรสร้างสัญญาณเขียนแรม (Write-Ram) เป็นสัญญาณที่ควบคุมการกำหนดตำแหน่งแอดเดรสและสัญญาณการเขียนข้อมูลลงในหน่วยความจำโดยมีวงจรมัลติเพล็กซ์แอดเดรส (Multiplex Address)เป็นตัวควบคุมการเขียนและอ่านข้อมูลจากหน่วยความจำดังแสดงในรูปที่ 5.4



รูปที่ 5.4 บล็อกไดอะแกรมภาคถอดรหัสเพจเจอร์

5.3.2 วงจรการทำงานภาคถอดรหัสเพจเจอร์

วงจรภายในภาคถอดรหัสเพจเจอร์ซึ่งวงจรการทำงานประกอบด้วยวงจรต่างๆดังต่อไปนี้ ดังแสดงในรูปที่ 5.5

5.3.2.1 วงจรกู้สัญญาณนาฬิกา(Clk Discovery)

ทำหน้าที่กู้สัญญาณนาฬิกากลับคืนมาจากสัญญาณเบสแบนด์(NRZ) ซึ่งมีความถี่สัญญาณนาฬิกา 1200 Hz เพื่อซิงส์โครไนซ์และควบคุมการทำงานตลอดเวลาในการรับส่งข้อมูล

5.3.2.2 วงจรตรวจจับเฟรมซิงส์ (Frame Sync Detector)

สัญญาณเฟรมซิงส์นับว่ามีความสำคัญในการส่งข้อมูลเพจเจอร์อย่างมากเพราะจะเป็นตัวบอกให้ทราบจุดเริ่มต้นของการส่งข้อมูลในแต่ละแบตซ์ และเครื่องรับยังอาศัยสัญญาณนี้สำหรับใช้ในการตรวจสอบระยะเวลาการส่งว่าสิ้นสุดลงเมื่อใดอีกด้วย สัญญาณเฟรมซิงส์จะถูกสร้างจากรหัสคำการซิงส์ทั้ง 32 บิต ซึ่งเป็นเอกลักษณ์เฉพาะตัวอยู่แล้ว โดยเอาท์พุตที่ได้จากวงจรจะให้ ค่าเป็นโลจิก High และจะเป็น Low ถ้าค่าที่ตรวจสอบไม่ถูกต้อง

5.3.2.3 วงจรหน่วงเวลาเฟรมซิงส์ (Frame Sync Delay)

เป็นวงจรที่ทำหน้าที่หน่วงเวลาที่ได้จากสัญญาณเฟรมซิงส์เพื่อเป็นสัญญาณควบคุมในการจัดเรียงข้อมูลในวงจรอนุกรมเป็นขนาน (Serial To Parallel) ตลอดจนเป็นสัญญาณควบคุมการอ่านและการเขียนข้อมูลที่รับเข้ามาลงใน หน่วยความจำแรม

5.3.2.4 วงจรชิฟรีจีสเตอร์ SIPO (Serial To Parallel Shift Register)

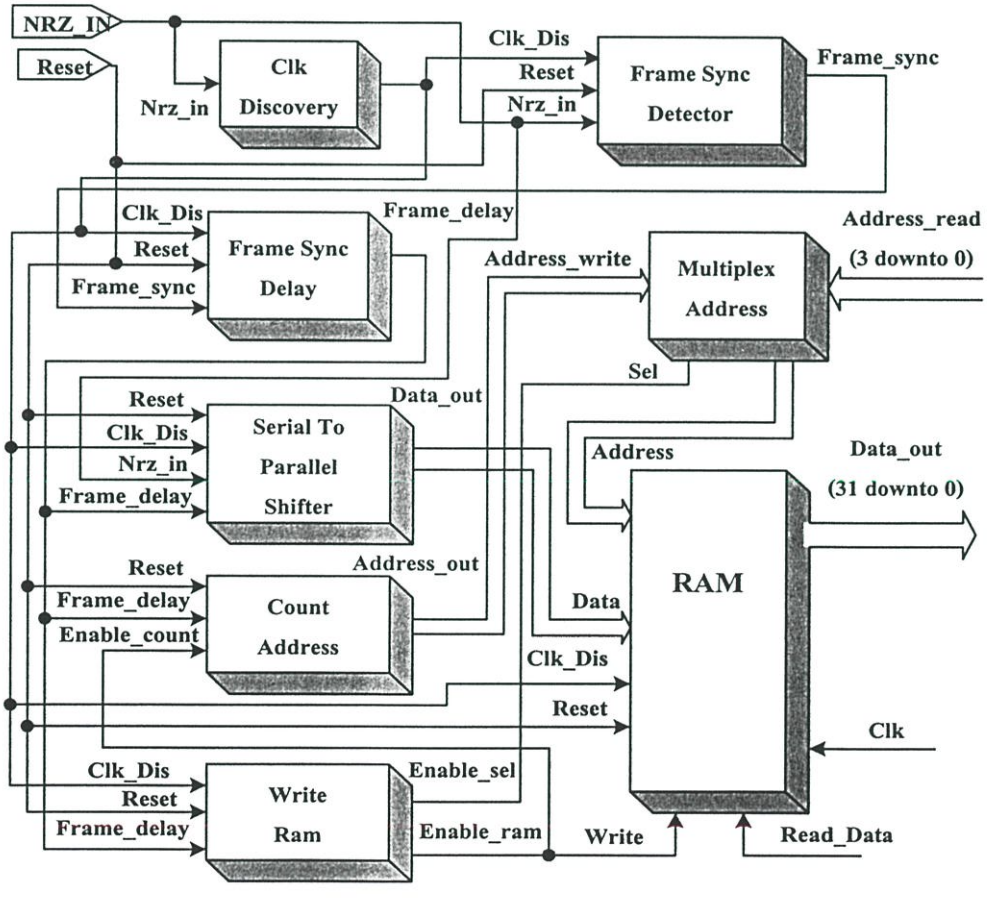
เนื่องจากข้อมูลที่รับได้จากสัญญาณเพจเจอร์เป็นข้อมูลแบบอนุกรม จึงจำเป็นที่ต้องแปลงข้อมูลเหล่านั้นให้เป็นแบบขนาน 32 บิต เพื่อให้สามารถเขียนลงแรมได้ ซึ่งหลักการของSIPO ก็คือการเลื่อนบิตของข้อมูลไปเรื่อยๆจนครบ 32 บิต จากนั้นก็จะทำการเขียนลงในแรมในขนาด 32 บิต โดยใช้สัญญาณนาฬิกาเป็นตัวกำหนดการเลื่อนบิตของข้อมูล เมื่อสัญญาณนาฬิกาเป็นขอบขาขึ้นทำให้ข้อมูลก็จะถูกเลื่อนไปตามจังหวะของสัญญาณนาฬิกาเพื่อกำหนดการเขียนลงในหน่วยความจำ (RAM) ต่อหนึ่งตำแหน่งแอดแดรสทุกๆครั้งที่มีการเลื่อนข้อมูลครบ 32 บิต

5.3.2.5 วงจรกำเนิดแอดแดรส (Count Address) และสัญญาณควบคุมการเขียนแรม (Write Ram)

โดยวงจรกำเนิดแอดเดรสทำหน้าที่กำเนิดสัญญาณแอดเดรสโดยสัญญาณอินพุทจะได้จากเอาท์พุตของวงจรควบคุมสัญญาณการเขียนแรมซึ่งจะเป็นสัญญาณพัลส์โดยวงจรจะทำการนับแอดเดรสขึ้นไปเรื่อยๆตามการเก็บข้อมูลลงในหน่วยความจำ(Ram) โดยจะนับแอดเดรสทุกครั้งหลังจากการเก็บการเก็บข้อมูลครบ 32 บิต พร้อมกันนั้นสัญญาณควบคุมการเขียนสัญญาณของแรมก็จะทำการเขียนเพื่อเก็บข้อมูลลงในแรมในแต่ละแอดแดรส

5.3.2.6 วงจรหน่วยความจำ (RAM)

การเก็บข้อมูลของเพจเจอร์มีการส่งเข้ามาจะถูกเก็บข้อมูลไว้ในหน่วยความจำก็เพราะว่าการส่งข้อมูลในระบบเพจเจอร์จะเป็นการส่งแบบทิศทางเดียวโดยส่งมาเป็นช่วงๆในการส่งข้อมูลแต่ละครั้ง ดังนั้นจึงต้องเก็บข้อมูลไว้ก่อนเพื่อกันไม่ให้ข้อมูลสูญหายโดยจะทำการเก็บข้อมูลจำนวน 1 รหัสคำ(32 บิต) ต่อหนึ่งแอดเดรส ดังนั้นจึงกำหนดการเก็บข้อมูลตามข้อมูลที่ส่งเข้ามาในแต่ละครั้งโดยสัญญาณนาฬิกาจะแยกออกเป็นสองส่วน คือ สัญญาณนาฬิกาที่ควบคุมการเขียนข้อมูลลงในหน่วยความจำซึ่งมีความถี่อยู่ที่ 1200 Hz และส่วนที่สอง จะทำหน้าที่ควบคุมการอ่านข้อมูลไปยังวงจรส่วนอื่นโดยใช้ความถี่ 50 kHz โดยสัญญาณแอดเดรสจะทำการเก็บข้อมูลและอ่านข้อมูลออกไปโดยอาศัยการมัลติเพล็กซ์สัญญาณแอดเดรส โดยมีสัญญาณดาต้าอินพุทขนาด 32 บิต และสัญญาณเขียนข้อมูลเพื่อทำหน้าที่เก็บข้อมูลให้เป็นไปอย่างถูกต้อง ส่วนการอ่านข้อมูลก็จะมีขาควบคุมการอ่านข้อมูลเพื่อทำหน้าที่อ่านข้อมูลออกจากหน่วยความจำตามสัญญาณแอดเดรสที่ถูกกำหนดเพื่อให้ได้ข้อมูลที่ถูกต้องแสดงดังรูปที่ 5.5

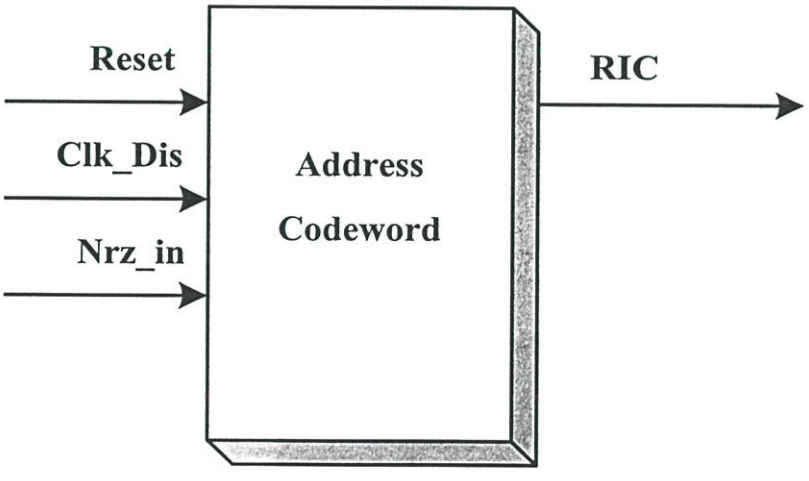


รูปที่ 5.5 วงจรภาคถอดรหัสเพจเจอร์

5.4 ภาคถอดรหัสหมายเลขเรียกขาน(Address Codeword)

5.4.1 บล็อกไดอะแกรมภาคถอดรหัสหมายเลขเรียกขาน

สัญญาณหมายเลขเรียกขานเกิดจากรหัสหมายเลขเฉพาะที่กำหนดให้เพจเจอร์แต่ละเครื่อง ซึ่งมีความสำคัญอย่างมาก เพราะเพจเจอร์แต่ละเครื่องสามารถที่จะรับข่าวสารจากผู้ส่งได้อย่างถูกต้องซึ่งจากการตรวจสอบสัญญาณหมายเลขเรียกขานนี้จะเป็นสัญญาณที่บ่งบอกถึงจุดเริ่มต้นของข่าวสารที่ส่งมาให้เพจเจอร์แต่ละเครื่องด้วย จากการที่มีการกำหนดให้หมายเลขเรียกขานของเครื่องรับเพจเจอร์ แต่ละเครื่องนั้นแตกต่างกันมีผลทำให้สัญญาณหมายเลขเรียกขานของแต่ละเครื่องถูกส่งมาอยู่ในเฟรม ที่แตกต่างกันด้วย ซึ่งจะสามารถทราบตำแหน่งของเฟรม ที่ส่งหมายเลขเรียกขานได้โดยการนำเอาหมายเลขเรียกขาน(PLUGCODE) ซึ่งเป็นหมายเลขประจำตัวเครื่องรับเพจเจอร์ซึ่งเป็นเลขฐานสิบ แปลงให้อยู่ในรูปเลขฐานสอง จะได้ทั้งหมด 21 บิตแต่ 3 บิตหลังที่มีความสำคัญน้อย (LSB) คือ บิตที่ 19-21 ที่ไม่ได้ทำการส่งจะเป็นตัวกำหนดเฟรม ที่ใช้ในการส่งหมายเลขเรียกขานของแต่ละเครื่องรับเพจเจอร์ โดยมีวงจรตรวจจับหมายเลขเรียกขาน(Plugcode Detector) ทำหน้าที่ในการตรวจจับหมายเลขเรียกขานจำนวน 18 บิต โดยจะส่งสัญญาณที่ตรวจจับได้ไปยังวงจรดีเลย์(Plugcode Delay) ซึ่งจะทำการหน่วงเวลาให้ตรงกับเฟรมที่ส่งรหัสหมายเลขเรียกขานแต่ละเครื่องกำหนดมา โดยมีวงจรเฟรมซิ้งส์ (Frame Sync Detector) เป็นจุดเริ่มต้นการนับจำนวนเฟรมที่ส่งมาโดยมี วงจรนับเฟรม(Count Frame)ทำหน้าที่นับจำนวนเฟรมที่ส่งข้อมูลเข้ามาโดยจะให้สัญญาณที่เอาท์พุทก็ต่อเมื่อมีการนับจำนวนเฟรมตรงกับการกำหนดการส่งหมายเลขเรียกขานตรงกับเฟรมนั้นๆ เพื่อให้ได้เอาท์พุตเพื่อบ่งบอกถึงการเริ่มต้นการส่งข้อมูลแสดงดังในรูปที่ 5.6



รูปที่ 5.6 บล็อกไดอะแกรมภาคถอดรหัสหมายเลขเรียกขาน

5.4.1 วงจรภาคถอดรหัสหมายเลขเรียกขาน

ทำหน้าที่ตรวจจับหมายเลขเรียกขานของเพจเจอร์เพื่อบ่งบอกถึงการรับส่งข้อมูลดังแสดงในรูปที่ 5.7 ซึ่งประกอบด้วยวงจรดังต่อไปนี้

5.4.1.1 วงจรตรวจสอบหมายเลขเรียกขาน(Plugcode Detector) และ วงจรหน่วงเวลา(Plugcode Delay)

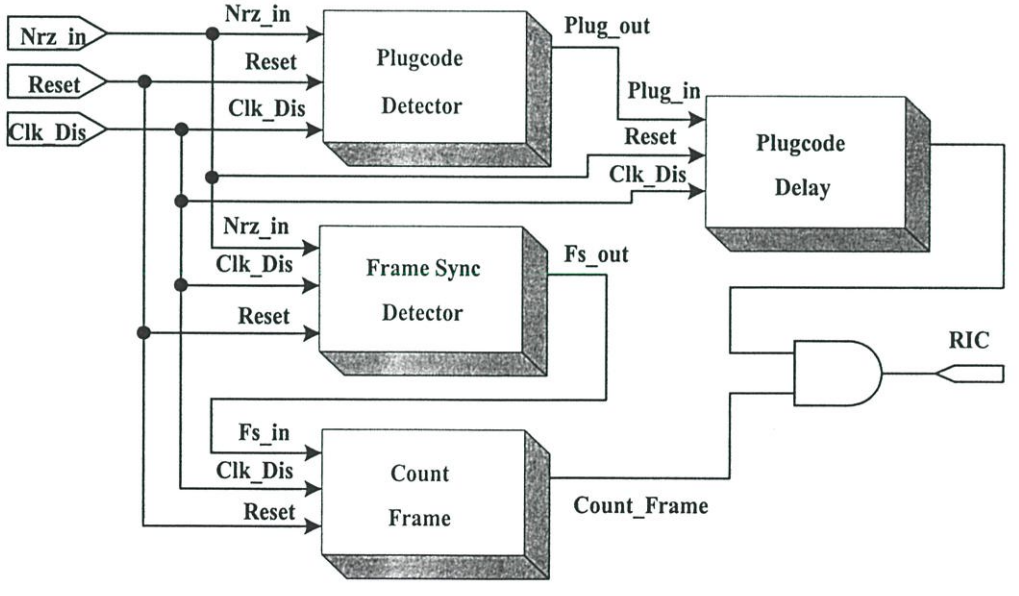
เป็นวงจรตรวจสอบรหัสหมายเลขเรียกขานซึ่งมีขนาด 18 บิต ซึ่งถ้าตรวจสอบรหัสหมายเลขเรียกขานได้ถูกต้องก็จะให้สัญญาณพัลส์หนึ่งออกมาที่เอาท์พุตและทำการหน่วงเวลาเพื่อให้ตรงกับจังหวะที่หมายเลขเรียกขานส่งเข้ามา

5.4.1.2 วงจรตรวจจับเฟรมซิงส์ (Frame Sync Detector)

ทำหน้าที่ตรวจจับรหัสเฟรมซิงส์ซึ่งเป็นรหัสที่บ่งบอกถึงการเริ่มต้นการส่งข้อมูลเพื่อเป็นจุดเริ่มต้นในการตรวจจับหมายเลขเรียกขานในการส่งที่ตำแหน่งเฟรมที่เท่าใดของแบตช์

5.4.1.3 วงจรนับจำนวนเฟรม(Count Frame)

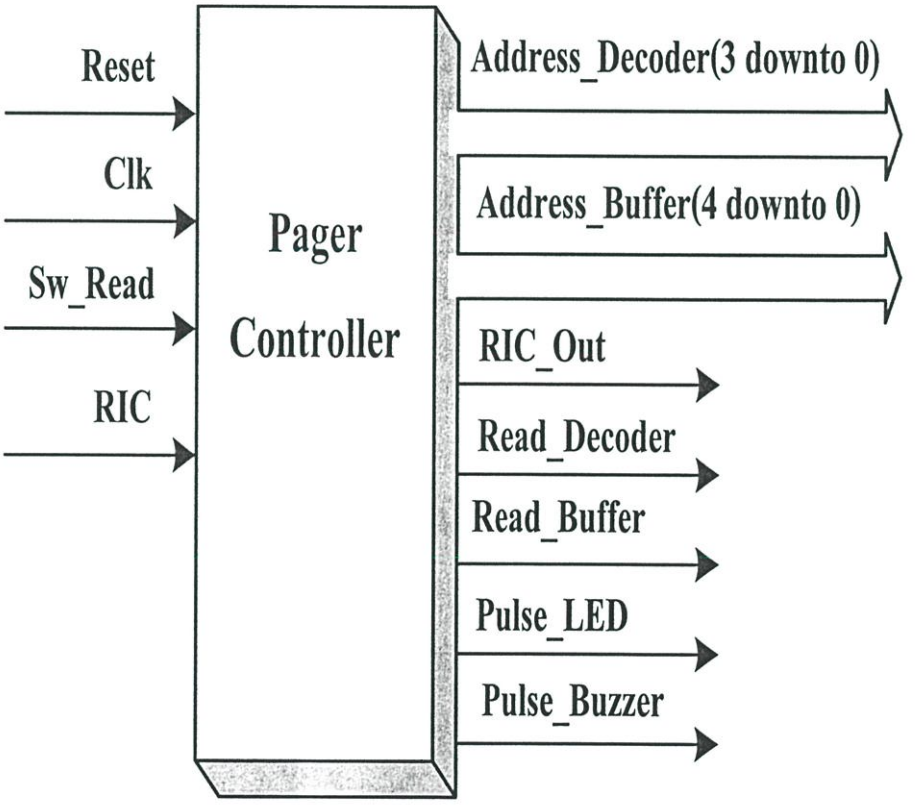
วงจรทำหน้าที่นับจำนวนเฟรมที่เริ่มการส่งหมายเลขเรียกขานเข้ามาเพื่อจะให้สัญญาณที่เอาท์พุตออกมาในจังหวะพอดีกับหมายเลขเรียกขานโดยมีแอนท์เป็นวงจรเปรียบเทียบเพื่อให้ได้เอาท์พุตออกมาในจังหวะการส่งหมายเลขเรียกขานพอดี



รูปที่ 5.7 วงจรภาคถอดรหัสหมายเลขเรียกขาน

5.5 ภาคควบคุมการทำงาน (Pager Controller)

ทำหน้าที่ควบคุมการทำงานทั้งหมดโดยเริ่มต้นการทำงานเมื่อมีการตรวจสอบพบหมายเลขขานหรือสัญญาณ RIC ก็จะแสดงผลการเรียกเพจเจอร์โดยการแสดงผลที่ LED หรือ Buzzer และทำการอ่านข้อมูลจากภาคถอดรหัสเพจเจอร์โดยการควบคุมสัญญาณแอดเดรส(Address Decoder)และ สัญญาณควบคุมการอ่านข้อมูล(Read Decoder) โดยการย้ายข้อมูลมายังวงจรจัดเรียงข้อมูลซึ่งทำการคัดแยกข้อมูลออกมาจากแต่ละเฟรมของแต่แบตซ์ที่รับเข้ามา ดังนั้นเมื่อต้องการดูข้อมูลที่รับเข้า โดยการกดสวิตซ์(SW_Read) เพื่อควบคุมสัญญาณแอดเดรส(Address Buffer) และสัญญาณควบคุมการอ่านข้อมูล(Read Buffer) ในการนำข้อมูลจากภาคจัดเรียงข้อมูลไปยังภาคเชื่อมต่อ LCD Graphic Interface แสดงผลข้อมูลที่รับเข้ามา ดังแสดงดังรูปที่ 5.8

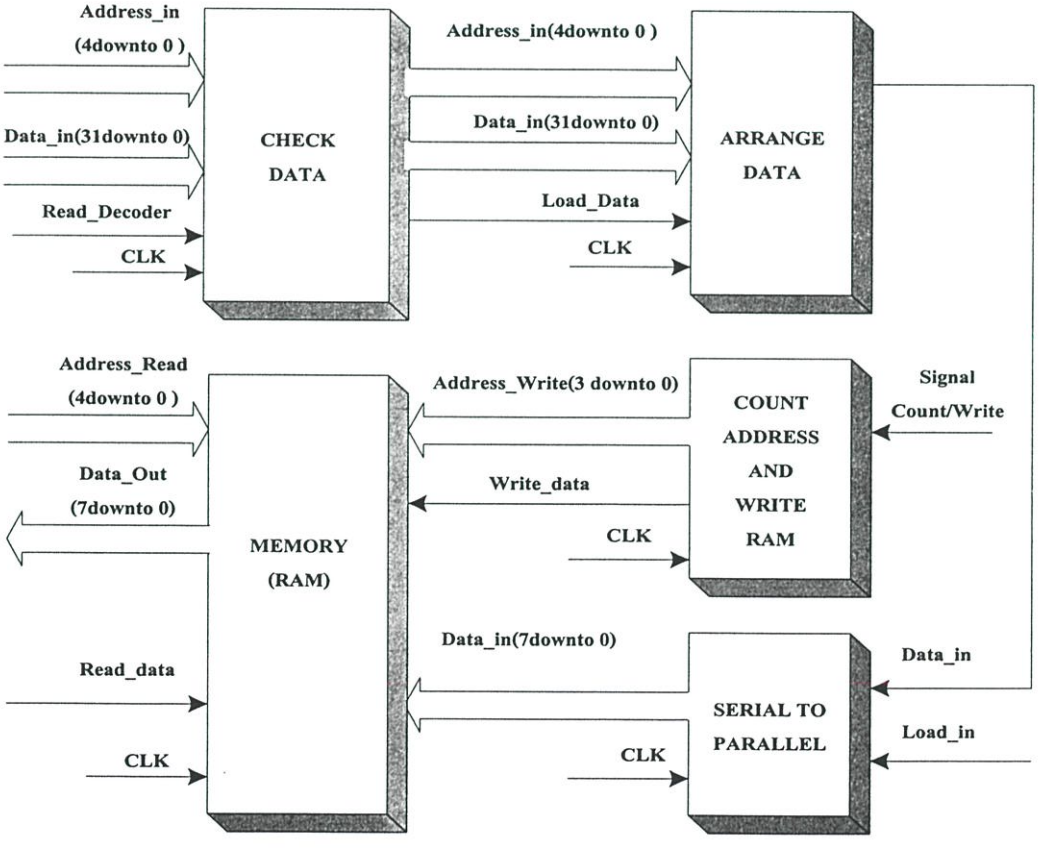


รูปที่ 5.8 บล็อกไดอะแกรมภาคควบคุมการทำงานของระบบ

5.6 ภาคจัดเรียงข้อมูล (Data Buffer)

5.6.1 บล็อกไดอะแกรมภาคจัดเรียงข้อมูล

การจัดเรียงข้อมูลกระทำได้โดยการอ่านข้อมูลจาก ภาคถอดรหัสข้อมูลในหน่วยความจำ (RAM) โดยทำการอ่านข้อมูลในแต่ละแอดเดรสของหน่วยความจำโดย วงจรตรวจสอบข้อมูล (Check Data) จะทำการคัดเอาข้อมูลที่เป็นข่าวสารข้อมูลซึ่งมีขนาด 20 บิตในแต่และเฟรมข้อมูลที่ส่งมายังวงจรเรียงข้อมูล(Arrange Data) ซึ่งทำหน้าที่จัดเรียงข้อมูลข่าวสารทั้งหมดที่ทำการรับเข้ามา หลังจากนั้นจะทำการจัดเก็บข้อมูลขนาด 8 บิตเพื่อจะนำไปแสดงผลข้อมูลลงในส่วนหน่วยความจำ (RAM) ของแต่ละแอดเดรส โดยผ่านวงจร Parallel to Serial ซึ่งจะนำการชิฟข้อมูลทั้งหมดครั้งละ 1 บิต จนครบข่าวสารข้อมูล โดยมีวงจร Serial to Parallel ซึ่งทำการรับข้อมูลเข้ามาโดยทำการจัดเก็บข้อมูลที่เข้ามาจนครบ 8 บิต ก็จะทำการจัดเก็บข้อมูลลงในหน่วยความจำ(RAM) โดยมีวงจร Count Address และวงจร Write Ram ทำหน้าที่ระบุตำแหน่งการจัดเก็บข้อมูลและควบคุมการเขียนข้อมูลลงในหน่วยความจำซึ่งเหตุผลการเก็บข้อมูลครั้งละ 8 บิตก็เพื่อจะนำไปแสดงผลในส่วน LCD Graphic Display ซึ่งรับข้อมูลครั้งละ8 บิตในการแสดงผลข้อมูลดังแสดงดังรูปที่ 5.9



รูปที่ 5.9 บล็อกไดอะแกรมภาคจัดเรียงข้อมูล

5.6.2 วงจรภาคจัดเรียงข้อมูล

ทำหน้าที่จัดเรียงและเก็บข้อมูลที่ได้จากภาคถอดรหัสเพจเจอร์โดยจะทำการคัดแยกข้อมูลของแต่ละเฟรมในแต่ละแบตซ์โดยคัดเฉพาะข่าวสารข้อมูลจำนวน 20 บิตในบิตที่ 2– 21 เพื่อจัดเรียงข้อมูลที่อยู่ในระหว่างเฟรมถัดไปให้ได้ข้อความที่ถูกต้องและมีขนาด 8 บิตเพื่อนำไปแสดงผลข้อมูล ดังแสดงในรูปที่ 5.10

5.6.2.1 วงจรตรวจสอบและเรียงข้อมูล(Check Data /Arrange Data)

ทำหน้าที่ในการคัดข้อมูลโดยการอ่านข้อมูลในแต่ละเฟรมของภาคถอดรหัสเพจเจอร์และทำการเช็คบิต 0 ของแต่ละเฟรมถ้ามีค่าเป็นศูนย์ก็จะเป็นเฟรมข่าวสารข้อมูลก็จะทำการจัดเรียงข้อมูลในบิตที่ 2 – 21(20บิต) จนครบข้อมูลที่รับเข้ามาเพื่อทำการบันทึกข้อมูลลงในหน่วยความจำ

5.6.2.2 วงจรชิฟรีจีสเตอร์ PISO(Parallel To Serial)

วงจรทำการโหลดข้อมูลทั้งหมดที่ได้จากวงจรคัดและเรียงข้อมูลและทำการชิฟข้อมูลออกที่เอาท์พุตที่ละ 1 บิตเพื่อส่งไปยังวงจร SIPO(Serial To Parallel)

5.6.2.3 วงจรชิฟรีจีสเตอร์ SIPO(Serial To Parallel)

ทำหน้าที่รับสัญญาณครั้ง 1 บิตที่ชิฟเข้ามาแล้วทำการจัดเรียงข้อมูลใหม่ให้มีขนาด 8 บิตเพื่อเก็บข้อมูลลงในหน่วยความ(RAM)

5.6.2.4 วงจรนับแอดเดรส(Count Address)และวงจรเขียนแรม(Write Ram)

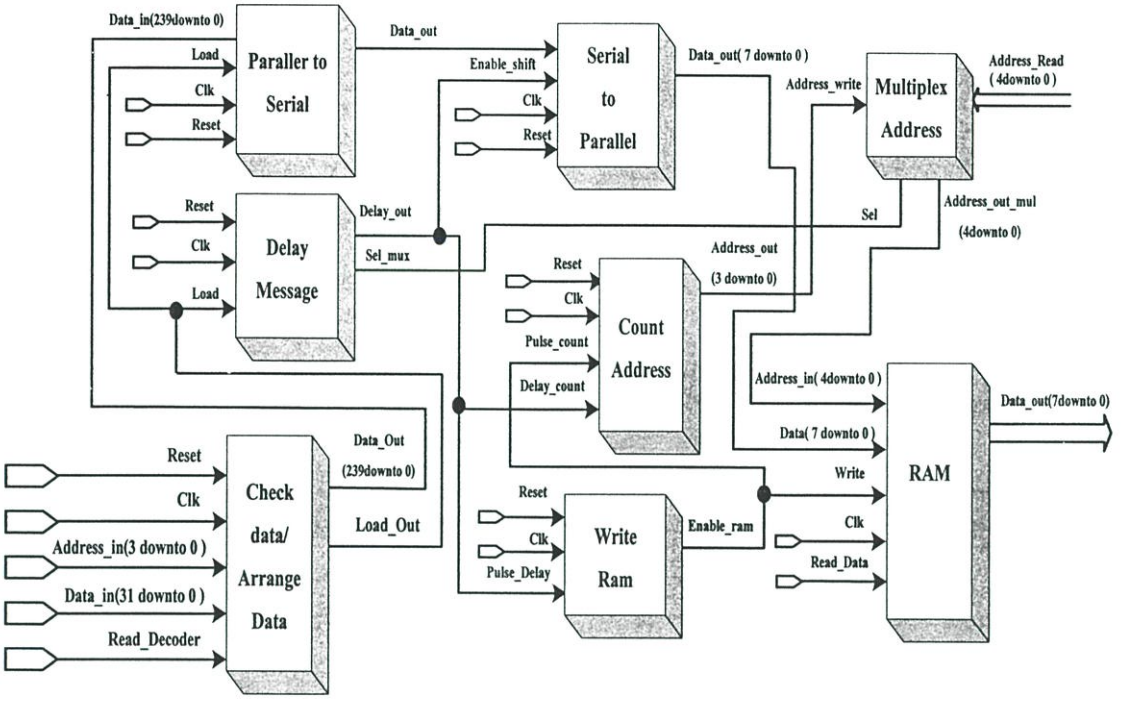
ทำหน้าที่ควบคุมการเก็บข้อมูลลงในหน่วยความจำ(RAM) โดยทำการนับสัญญาณแอดเดรสเพื่อกำหนดตำแหน่งการเก็บข้อมูลลงในตำแหน่งต่างๆภายในหน่วยความจำโดยมีสัญญาณควบคุมการเขียนข้อมูลตามจังหวะตำแหน่งการเก็บข้อมูล

5.6.2.5 วงจรมัลติเพล็กซ์แอดเดรส(Multiplex Address)

เป็นวงจรที่ทำหน้าที่เลือกสัญญาณแอดเดรสในการกำหนดตำแหน่งการเขียนข้อมูลลงในหน่วยความจำและกำหนดตำแหน่งข้อมูลในการอ่านข้อมูลออกไปโดยมีขา SEL เป็นโหมดการเลือกการเขียนและอ่านข้อมูลจากหน่วยความจำ

5.6.2.6 วงจรหน่วยความจำ(RAM)

ทำหน้าที่เก็บข้อมูลซึ่งมีขนาด 8 บิต เพื่อนำข้อมูลที่ได้ไปทำการแสดงผลข่าวสารข้อมูลที่รับเข้ามาโดยมีสัญญาณนาฬิกาขนาดความถี่ 50 kIIz เป็นตัวควบคุมจังหวะการอ่านและเขียนข้อมูลในการทำงานของหน่วยความจำทั้งหมด

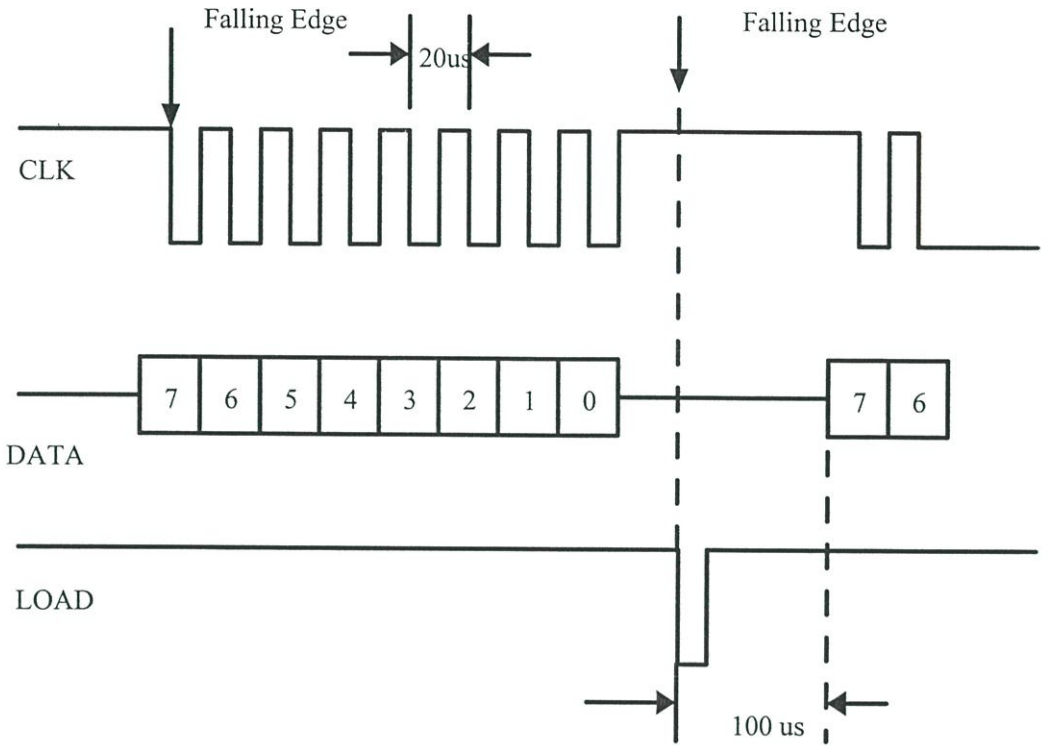


รูปที่ 5.10 วงจรภาคจัดเรียงข้อมูล

5.7 ภาคแสดงผล (LCD Graphic Display)[3]

5.7.1 สัญญาณควบคุมการทำงานของ LCD Graphic Display

การควบคุมการทำงาน LCD Graphic Display จะควบคุมผ่าน 3 Bit Serial ซึ่งการทำงานจะเป็นการส่งข้อมูลแบบ Synchronous กล่าวคือ สัญญาณจะขึ้นอยู่กับ Clock ขอบขาลง โดยไม่ขึ้นกับระยะเวลาของสัญญาณ(เหมือน RS232) ซึ่งจะเหมาะกับงานที่ค่อนข้างซับซ้อน และต้องดูแล INPUT/OUTPUT ที่ไม่แน่นอนจำนวนมาก การควบคุมแบบนี้จะใช้ขาสัญญาณ 3 เส้นคือ DIN (Data In), CLK (Clock) และ LD (Load)โดยรูปแบบการส่งข้อมูลจะกระทำครั้งละ 8 บิต (1 Byte) ทั้งนี้รูปแบบของข้อมูลจะเป็นเหมือนกับ RS232 ทุกประการ การควบคุมแบบนี้จะไม่มีการตอบรับสัญญาณการรับส่งข้อมูล เพราะฉะนั้นจะต้องหน่วงเวลาตาม Busy Time ที่กำหนดเสมอ หลังจากที่ส่งคำสั่งไปแล้วรายละเอียดของสัญญาณแสดงได้ดังรูปที่ 5.11

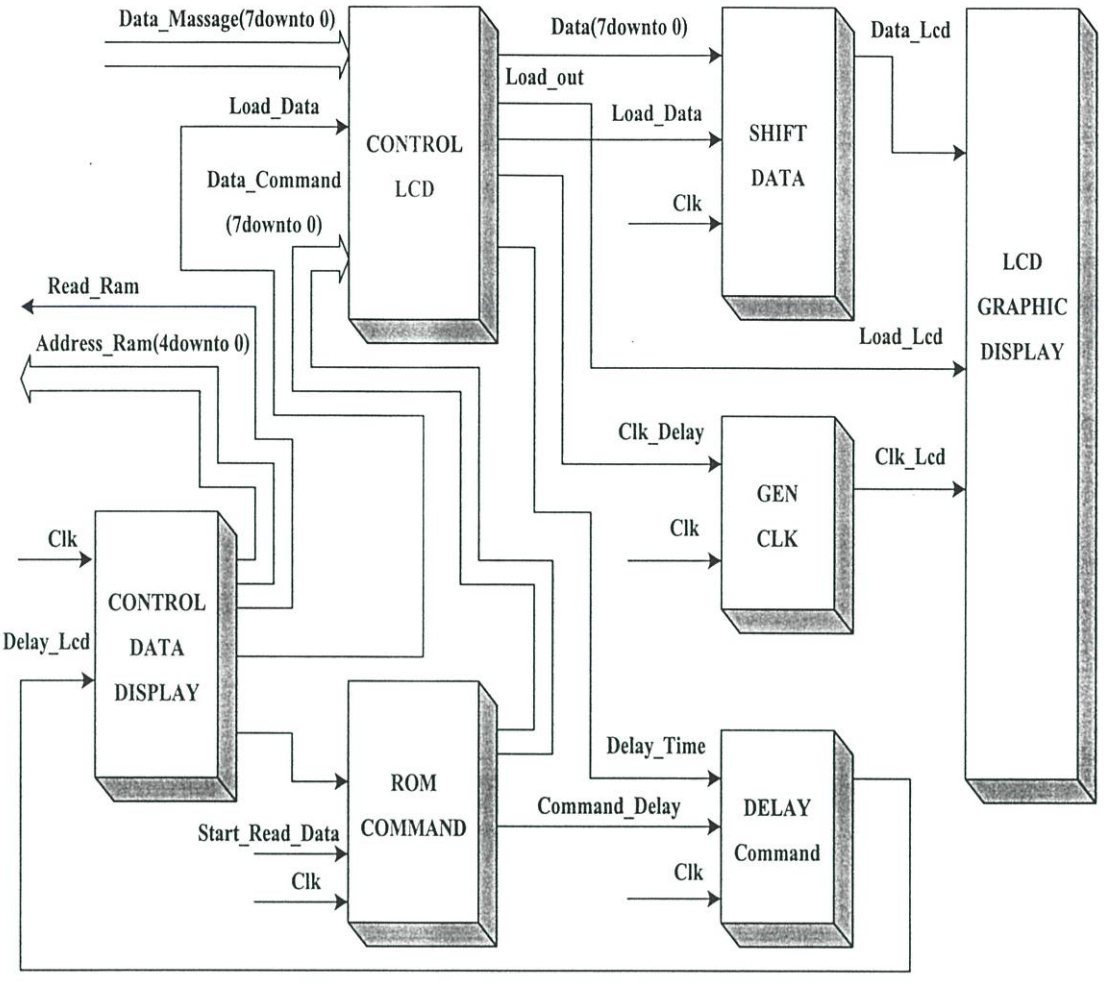


รูปที่ 5.11 สัญญาณควบคุม LCD Graphic Display ผ่านทาง 3 Bit Serial

5.7.2 ภาค LCD Graphic Interface

5.7.2.1 บล็อกไดอะแกรมภาค LCD Graphic Interface

วงจรเชื่อมต่อ LCD Graphic Interface จะทำหน้าที่นำข่าวสารข้อมูลที่ได้จากภาคจัดเรียงข้อมูล(Data Buffer) มาทำการแสดงผลโดยมีวงจรชิฟข้อมูล(Shifter Data) ทำหน้าที่รับข้อมูลเข้ามาครั้งละ 8 บิตและทำการชิฟข้อมูลที่ละบิต เพื่อป้อนให้กับตัวแสดงผล โดยมีวงจรกำเนิดสัญญาณนาฬิกา(Gen Clock) ทำหน้าที่กำเนิดสัญญาณนาฬิกาครั้งละ 8 ลูกเพื่อควบคุมการทำงานโดยจะทำการชิฟข้อมูลแต่ละครั้ง 8 ลูก โดยมีวงจรหน่วงเวลา (LCD Delay) ทำหน้าที่ในการหน่วงเวลาการทำงานในแต่ละคำสั่งในการนำข้อมูลไปแสดงผลซึ่งค่าในการหน่วงเวลาก็ขึ้นอยู่กับคำสั่งในแต่ละคำสั่งของ LCD Graphic Display ซึ่งคำสั่งต่างๆจะอยู่ใน วงจรROM Command โดยมีวงจรควบคุมแอลซีดี(Control LCD) เป็นตัวกำหนดควบคุมการทำงานในการนำคำสั่งต่างๆไปแสดงผล ส่วนวงจรควบคุมการแสดงผลข้อมูล(Control Data Display) ทำหน้าที่ในการควบคุมการนำข้อมูลจากภาคจัดเรียงข้อมูล(Data Buffer) มาแสดงผลดังแสดงดังรูปที่ 5.12



รูปที่ 5.12 บล็อกไดอะแกรมภาค LCD Graphic Interface

5.7.2.2 วงจร LCD Graphic Interface

ทำหน้าที่เป็นวงจรเชื่อมต่อการส่งข้อมูลกับอุปกรณ์แสดงผล(LCD Graphic Display) ซึ่งการแสดงผลข่าวสารข้อมูลโดยการนำข้อมูลจากภาคจัดเรียงข้อมูลซึ่งมีขนาด 8 บิต มาแสดงผลโดยมีวงจรการทำงานดังแสดงดังในรูปที่ 5.13

5.7.2.2.1 วงจร Control Data Display

ทำหน้าที่ควบคุมการทำงานในการนำข้อมูลจาก ROM COMMAND เพื่อกำหนดคำสั่งในการแสดงผลของ LCD Graphic Display และจัดลำดับการส่งข้อมูลโดยการรับสัญญาณจากวงจร Delay Command ซึ่งเป็นการหน่วงเวลาข้อมูลแสดงผลและค่า BUSY TIME ของตัวแสดงผลด้วย

5.7.2.2.2 วงจร Control LCD

ทำหน้าที่ควบคุมการนำข้อมูลไปแสดงผลโดยควบคุมการทำงานผ่านวงจรชิฟข้อมูล (LCD Shifter) และวงจรกำเนิดสัญญาณนาฬิกา(GEN Clk) ตลอดจนสัญญาณหน่วงเวลา (Delay Command) ในการนำข้อมูลไปแสดงผล

5.7.2.2.3 วงจรชิฟข้อมูล(Data Shifter)และวงจรกำเนิดสัญญาณนาฬิกา(GEN Clk)

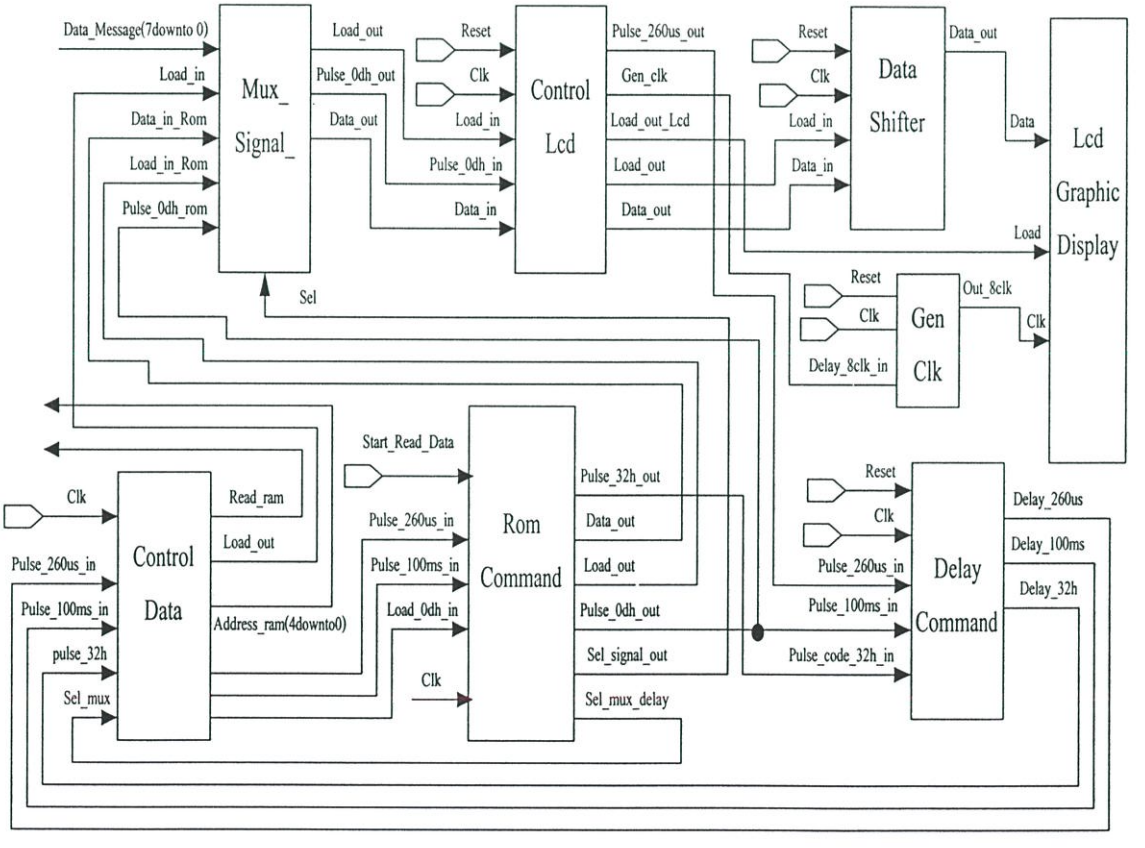
โดยวงจรทำหน้าที่รับข้อมูลเข้ามาขนาด 8 บิตและทำการชิฟข้อมูลครั้งละ 1 บิตโดยวงจรชิฟข้อมูล โดยมีวงจรกำเนิดสัญญาณนาฬิกาชุดละ 8ลูกในการให้จังหวะการชิฟข้อมูลขนาด 8บิตข้อมูลไปยังตัวแสดงผล

5.7.2.2.4 วงจร Rom Command

ทำหน้าที่เก็บคำสั่งในการทำงานของตัวแสดงผล(LCD Graphic Display) เช่นคำสั่งการเคลียร์จอภาพ การกำหนดบรรทัด(Row and Cloum) การกำหนดจำนวนการแสดงผลตัวอักษร การกำหนดให้ Cursor On หรือ OFF ตลอดจนการ Set Graphic ในการแสดงภาพแสดงบนจอภาพ เป็นต้น

5.7.2.2.5 วงจร Mux Signal และ วงจร Delay Command

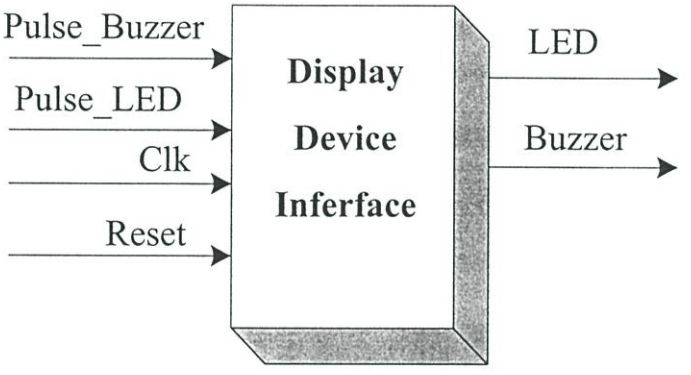
วงจร Mux Signal ทำหน้ามัลติเพล็กซ์การนำข้อมูลใน Rom Command และข้อมูลจากภาคจัดเรียงข้อมูลไปแสดงผล ส่วนวงจร Delay Command ทำหน้าที่หน่วงเวลาการแสดงผลข้อมูลตามค่า Busy Time ของ LCD Graphic Display



รูปที่ 5.13 วงจรภาค LCD Graphic Interface

5.8 ภาคเชื่อมต่ออุปกรณ์แสดงผล (Display Device Interface)

ภาคเชื่อมต่ออุปกรณ์แสดงทำหน้าที่แสดงผลการทำงานที่เกิดการเรียกเพจเจอร์เพื่อส่งข่าวสารข้อมูลโดยจะแสดงการเรียกเพจเจอร์ในสองลักษณะคือการแสดงผลในของ LED จะทำงานในลักษณะกระพริบและ อุปกรณ์ Buzzer จะส่งเสียงเรียกเตือนขึ้นมาเมื่อมีการเรียกเพจเจอร์เช่นกันดังแสดงรูปที่ 5.14

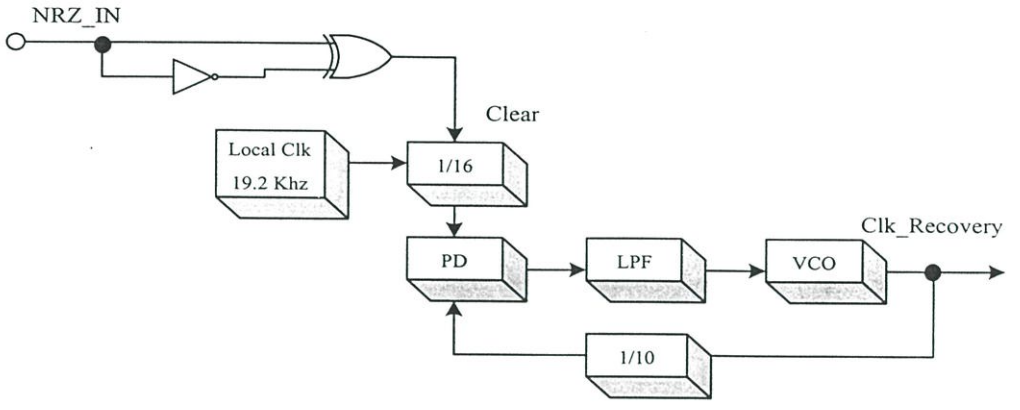


รูปที่ 5.14 บล็อกไดอะแกรมภาคเชื่อมต่ออุปกรณ์แสดงผลการเรียกเพจเจอร์

5.9 วงจรกู้สัญญาณนาฬิกา(Clk Recovery)[13]

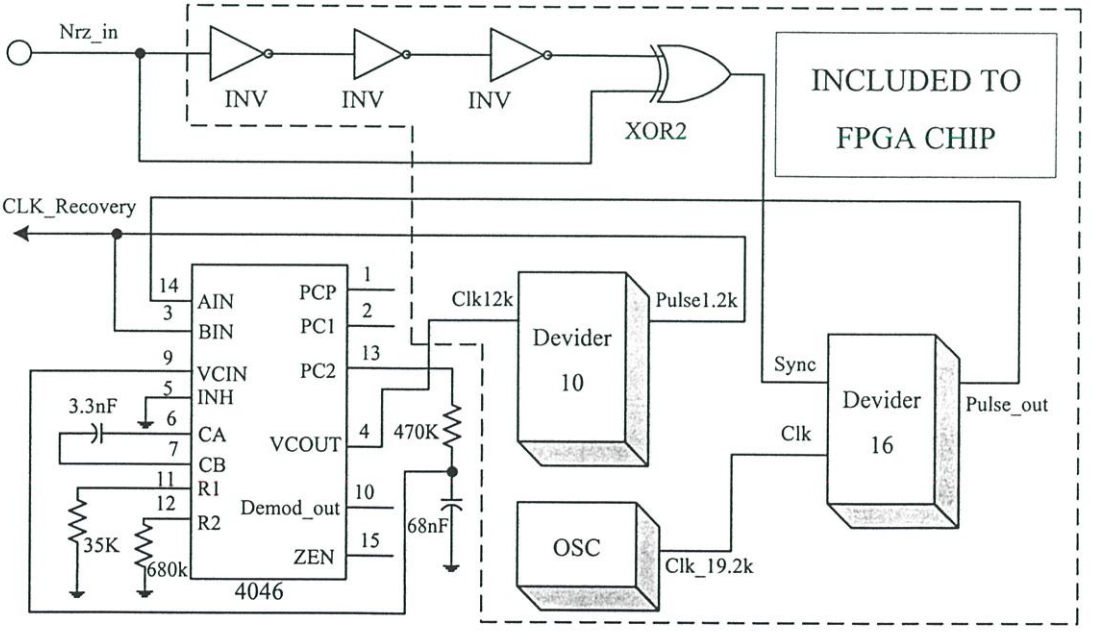
5.9.1 หลักการและวงจรกู้สัญญาณนาฬิกา

ในการกู้สัญญาณนาฬิกาที่ภาครับจะทำการกู้สัญญาณนาฬิกา ขนาดความถี่ 1200 บิตต่อวินาทีให้สามารถซิงส์โครไนซ์กับภาคส่งได้โดยใช้หลักการและวงจรประกอบด้วยวงจรสร้างความถี่อ้างอิง (Local Clock) ซึ่งสร้างความถี่ขนาด16 เท่าของ 1200 Hz, วงจรหาร 16, วงจรหาการเปลี่ยนแปลงของขอบสัญญาณ,วงจรเฟสล็อกลูป(PLL) โดยจากอินพุตสัญญาณ NRZ นำมาหาขอบสัญญาณ (Transition) โดยใช้วงจรนอทเกทหน่วงสัญญาณ ที่อินพุตแล้วทำการเอ็กซ์คลูซีฟออร์เข้ากับสัญญาณ NRZ เดิมซึ่งจะทำให้เอาท์พุตของเอ็กซ์คลูซีฟออร์เกทได้สัญญาณพัลซ์แคบๆซึ่งเกิดขึ้นทุกๆการเปลี่ยนแปลงสัญญาณอินพุต โดยสัญญาณนี้จะเป็นสัญญาณเคลียร์(Clear) ให้กับวงจรหาร 16 โดยที่อินพุตเป็นความถี่ขนาด 16 เท่าของความถี่บิตเรท ดังนั้นความถี่ของสัญญาณนาฬิกาที่เอาท์พุตของวงจรหาร 16 จะเป็นความถี่ของสัญญาณนาฬิกาที่ซิงส์โครไนซ์กับสัญญาณเบสแบนด์ที่รับเข้ามา โดยขนาดความถี่สัญญาณนาฬิกาที่เอาท์พุตวงจรหาร 16 จะถูกปรับอัตราการหารด้วยขนาดต่างๆกัน โดยขึ้นอยู่กับขอบของสัญญาณเบสแบนด์ที่อินพุตของวงจรกู้สัญญาณนาฬิกา โดยเอาท์พุตของวงจรหาร 16 จะเป็นความถี่ขนาด 1200 Hz ดังรูปที่ 5.15



รูปที่ 5.15 หลักการของวงจรกู้สัญญาณนาฬิกา

สัญญาณอินพุตที่เข้ามาจะทำให้ได้สัญญาณเอาท์พุทพัลส์เล็กๆทุกครั้งของวงจรซึ่งเป็นการต่อรวมระหว่างนอทเกทและวงจรเอ็กซ์คลูซีฟออร์ ที่มีการเปลี่ยนแปลงขอบสัญญาณ NRZ ก็เพื่อไปเคลียร์วงจร DEVIDER16 ซึ่งเป็นวงจรหาร 16 ดังนั้นความถี่ของสัญญาณนาฬิกาที่เอาท์พุตของวงจรหาร 16 จะเป็นความถี่ของสัญญาณนาฬิกาที่ซิงส์โครไนซ์กับสัญญาณเบสแบนด์ที่รับเข้ามาโดยเอาท์พุทจะมีความถี่ 1200 Hz สำหรับวงจร PLL ประกอบด้วยวงจรรวม 4046 และวงจร DEVIDER10 โดยวงจรรวม 4046 ประกอบด้วยวงจร VCO และเฟสดีเทคเตอร์โดยขนาดของความถี่ VCO ตั้งให้มีขนาด 10 เท่า ของสัญญาณอินพุตหรือขนาดเท่ากับ 12 kHz โดยวงจร DEVIDER10 ซึ่งเป็นวงจรหาร 10 สำหรับป้อนสัญญาณกลับไปเปรียบเทียบเฟสกับอินพุต โดยเอาสัญญาณป้อนกลับนี้ไปใช้งานเป็นสัญญาณนาฬิกาเพื่อนำไปใช้งานดังแสดงในรูปที่ 5.16



รูปที่ 5.16 วงจรกู้สัญญาณนาฬิกา[13]

บทที่ 6

การทดลอง

ในบทนี้จะเป็นการกล่าวถึงส่วนของการทดสอบการทำงานของวงจรที่ได้ออกแบบมาว่าสามารถทำงานได้จริงตามที่ออกแบบไว้หรือไม่ ในการทดสอบจะแบ่งออกเป็นสองส่วนคือ ส่วนที่หนึ่งจะเป็นส่วนของการจำลองการทำงาน(Simulation)ของวงจรก่อนที่จะโปรแกรมลงชิฟเอฟพีจีเอเพื่อดูผลตอบสนองของวงจรต่ออินพุทต่างๆที่ได้กำหนดให้ ส่วนที่สองจะเป็นส่วนของการทดสอบการทำงานของจริงวงจรทั้งหมดที่ได้ออกแบบมาเพื่อให้ทำงานเป็นตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัว โดยใช้อุปกรณ์เอฟพีจีเอ(FPGA)

6.1 ทดสอบโดยการจำลองการทำงาน(Simulation)

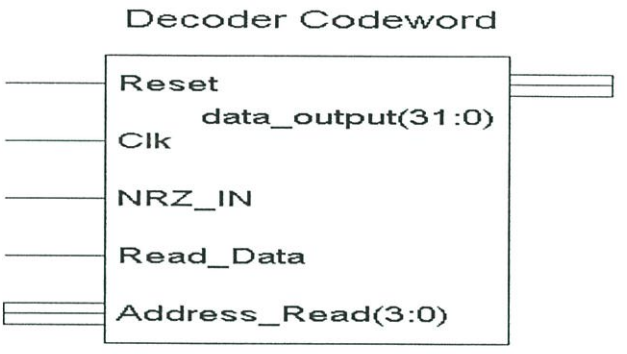
หลังจากได้ออกแบบวงจรโดยการอธิบายลักษณะพฤติกรรมของวงจร(Hardware Description Language) เป็นที่เรียบร้อยแล้ว ผู้ออกแบบจำเป็นจะต้องทำการจำลองการทำงานเพื่อดูผลการตอบสนองของวงจรต่างๆ ต่ออินพุทที่กำหนดให้ซึ่งหากผลลัพธ์ที่ได้ไม่ถูกต้องจะได้ทำการแก้ไขต่อไป ในส่วนของการออกแบบโดยอธิบายลักษณะพฤติกรรมของวงจรจะทำการออกแบบโดยใช้โปรแกรมในการออกแบบชื่อโปรแกรม Xilinx ISE 4.2I ซึ่งเป็นตัว Synthesis Tool และดาวน์โหลดลงในชิฟเอฟพีจีเอโดยใช้ตระกูล Virtex-E เบอร์ XCV1600E BG560 -6C ของบริษัท Xilinx

ในส่วนของการแสดงผลการทดสอบการทำงานของวงจรที่ได้ออกแบบด้วยภาษาวีเอชดีแอลที่สำคัญดังต่อไปนี้

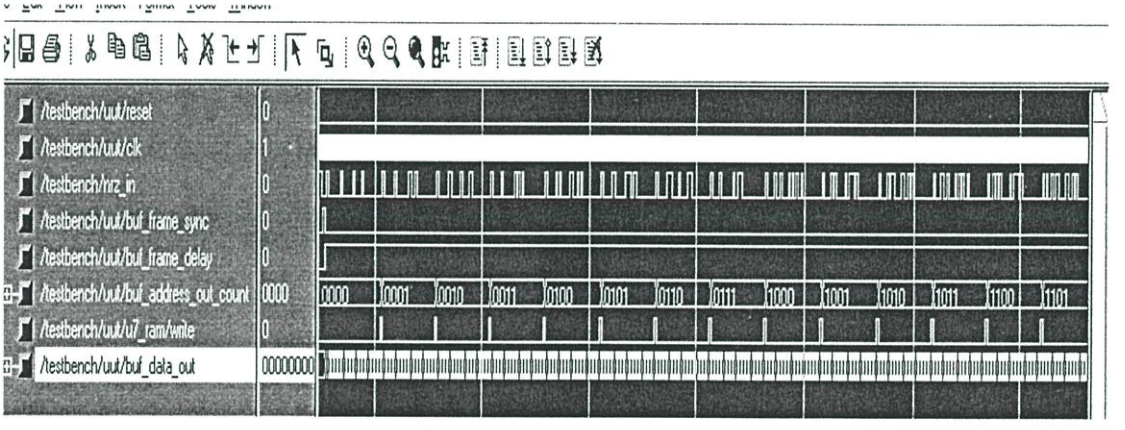
6.1.1 ภาคถอดรหัสเพจเจอร์(Decoder Codeword)

ภาคถอดรหัสเพจเจอร์จะทำหน้าที่เก็บสัญญาณเอาท์พุทมาจากภาครับวิทยุซึ่งเป็นสัญญาณเบสแบนด์แบบ NRZ ให้เก็บอยู่ในหน่วยความจำ (RAM) โดยเก็บข้อมูลครั้งละ 1 รหัสคำ (32 บิต) ซึ่งประกอบด้วย วงจรกู้สัญญาณนาฬิกา(Clk Recovery) ทำหน้าที่กู้สัญญาณนาฬิกาจากสัญญาณเบสแบนด์(NRZ) ที่รับเข้าเพื่อจ่ายให้กับภาคถอดรหัสข้อมูลทั้งหมดโดยที่วงจรตรวจสอบเฟรมซิงส์โครไนส์ (Frame Sync Detect) และวงจรหน่วงเวลาเฟรมซิงส์ (Frame Sync Delay) จะทำการตรวจสอบรหัสคำซิงส์โครไนซ์และหน่วงเวลาในการเก็บข้อมูลเข้าไปในหน่วยความจำโดยมีวงจรชิฟรีจีสเตอร์แบบอนุกรมเป็นขนาน (Serial To Parallel Register) ทำหน้าที่ในการจัดเรียงข้อมูลเก็บในหน่วยความจำ (RAM) วงจรนับแอดเดรส (Count Address) และ วงจรสร้างสัญญาณเขียน

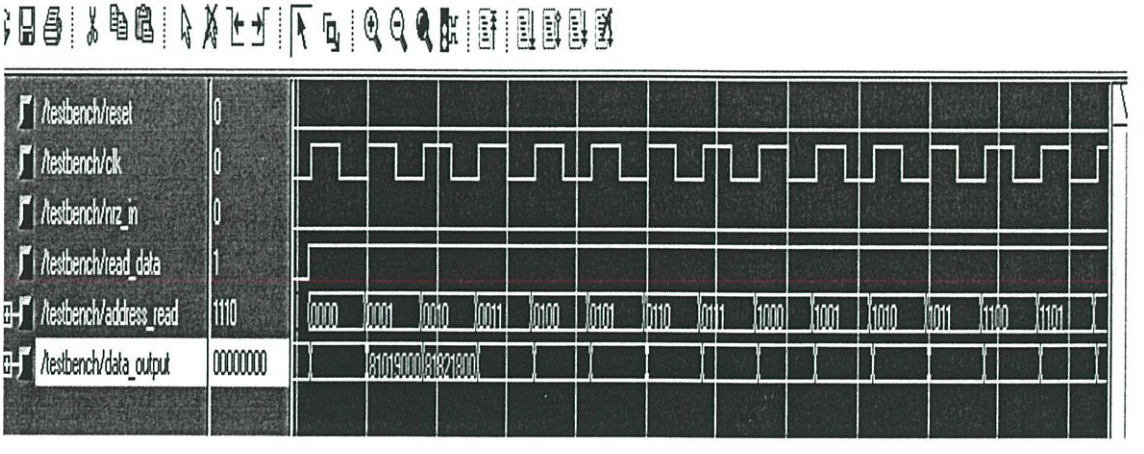
แรม (Write-Ram) เป็นสัญญาณที่ควบคุมการกำหนดแอดเดรสและสัญญาณการเขียนข้อมูลลงในหน่วยความจำ ส่วนวงจรเลือกแอดเดรส (Address Multiplex) เป็นตัวควบคุมการเขียนและการอ่านข้อมูลทั้งหมดจากหน่วยความจำสามารถแสดงผลจำลองการทำงานดังในรูปที่ 6.1,6.2,6.3



รูปที่ 6.1 แบบจำลองวงจรภาคถอดรหัสข้อมูล (Decoder Codeword)



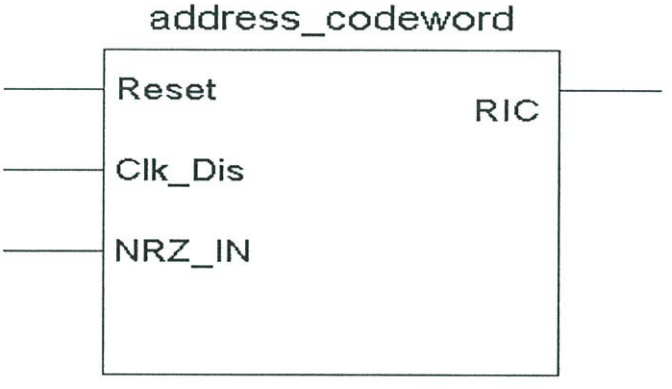
รูปที่ 6.2 ผลจำลองการทำงานของภาคถอดรหัสเพจเจอร์ในโหมดการเขียนข้อมูล



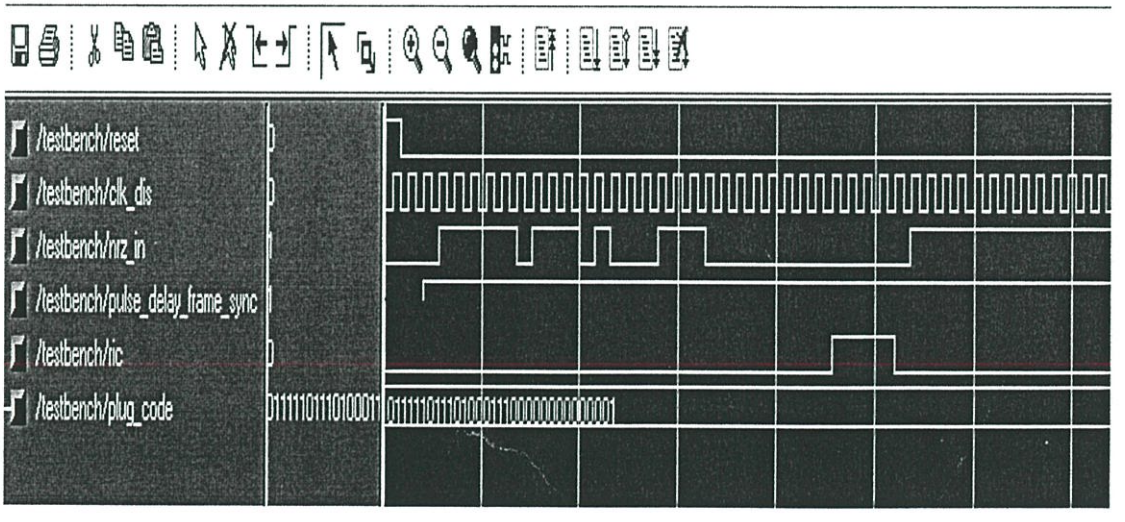
รูปที่ 6.3 ผลจำลองการทำงานภาคถอดรหัสเพจเจอร์ในโหมดการอ่านข้อมูล

6.1.2 ภาคถอดรหัสหมายเลขเรียกขาน (Address Codeword Decoder)

เป็นภาคที่เปรียบเทียบหมายเลขเรียกขานที่รับได้กับหมายเลขเครื่องที่ถูกโปรแกรมไว้ บล็อกไดอะแกรมภาคถอดรหัสหมายเลขเรียกขานแสดงดังรูปที่ 6.4 ซึ่งประกอบไปด้วยวงจรตรวจสอบหมายเลขเรียกขาน(Plugcode Detector) ทำหน้าที่เปรียบเทียบหมายเลขเรียกขานที่ถูกโปรแกรมไว้ เมื่อทำการตรวจสอบตรงกับค่าที่โปรแกรมไว้ ก็จะถูกหน่วงเวลาเพื่อที่จะนำมาทำการแอนกับวงจรนับเฟรมโดยเอาท์พุตของวงจรเฟรมซิงส์โครไนซ์(Frame Synchronize) เป็นจุดเริ่มต้นการนับจำนวนเฟรม เมื่อวงจรนับเฟรมนับค่าได้ตรงกับเฟรมของหมายเลขเรียกขานที่ส่งมาก็จะให้เอาท์พุทเป็น 1 ไปแอนกับเอาท์พุทของวงจรหน่วงเวลาหมายเลขเรียกขานเพื่อให้เกิดสัญญาณอินเตอร์รัพท์ เพื่อเป็นตัวบ่งบอกถึงจุดเริ่มต้นของการรับข่าวสารข้อมูลดังแสดงแบบจำลองการทำงานในรูปที่ 6.4,6.5



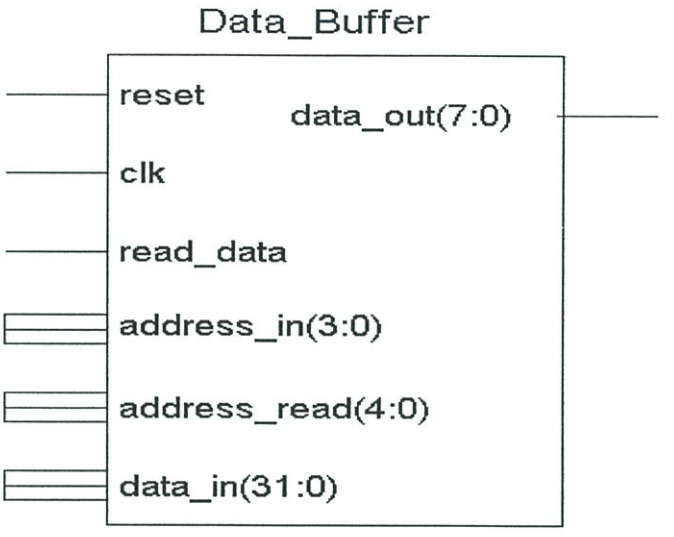
รูปที่ 6.4 แบบจำลองภาคถอดรหัสหมายเลขเรียกขาน (Address Codeword)



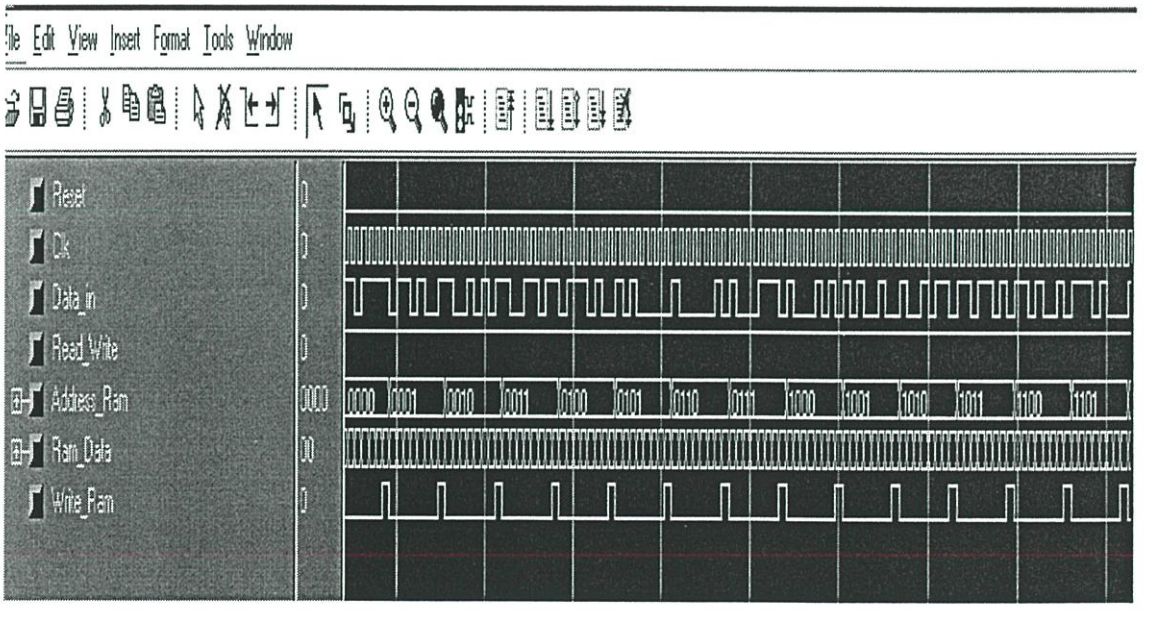
รูปที่ 6.5 ผลจำลองการทำงานภาคถอดหมายเลขเรียกขาน

6.1.3 ภาคจัดเรียงข้อมูล(Data Buffer)

ทำหน้าที่อ่านข้อมูลจากภาคถอดรหัสเพจเจอร์แล้วทำการคัดเอาข่าวสารข้อมูลมาทำการจัดเรียงข้อมูลใหม่ให้มีขนาด 8 บิต เพื่อที่จะนำข้อมูลไปแสดงผลดังแสดงแบบจำลองการทำงานดังในรูปที่ 6.6,6.7



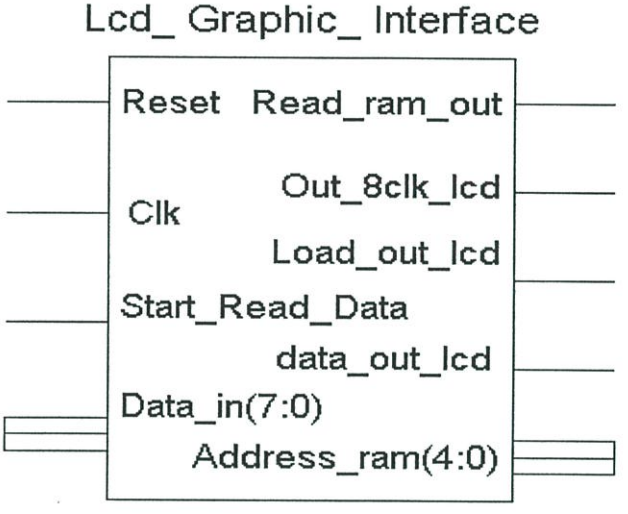
รูปที่ 6.6 แบบจำลองการทำงานภาคจัดเรียงข้อมูล(Data Buffer)



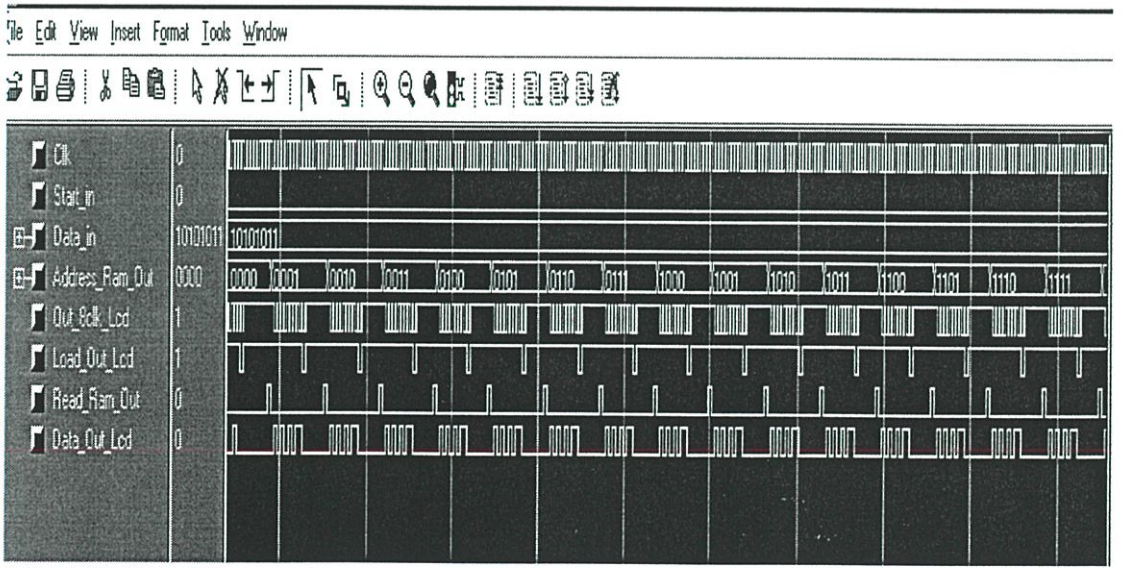
รูปที่ 6.7 ผลจำลองการทำงานภาคจัดเรียงข้อมูล

6.1.4 ภาคเชื่อมต่อวงจรแสดงผล (LCD Graphic Display Interface)

เป็นภาคแสดงข่าวสารข้อมูลโดยใช้ อุปกรณ์กราฟฟิกแอลซีดี เป็นตัวแสดงผลข้อมูลที่รับเข้ามาโดยสามารถแสดงผลได้ทั้งภาษาไทยและภาษาอังกฤษตามมาตรฐานสมอ.โดยมีขนาด128X 64 DOT สามารถแสดงตัวอักษรไทย และภาษาอังกฤษได้ 16X4 บรรทัด ซึ่งลักษณะการส่งข้อมูลจะทำการส่งข้อมูลแบบ 3 Bit Serial โดยมีภาคควบคุมการทำงานแสดงดังรูปที่ 6.8,6.9



รูปที่ 6.8 แบบจำลองการทำงานภาคเชื่อมต่อวงจรแสดงผล



รูปที่ 6.9 ผลจำลองการทำงานของภาคเชื่อมต่อวงจรแสดงผล

6.2 การทำ Implementation บน FPGA รุ่น Virtex-E เบอร์ XCV1600E BG560-6

จากการออกแบบวงจรตัวถอดรหัสในระบบทั้งหมดเมื่อนำไปทำการ Implement ลงบน FPGA เบอร์ XCV1600E BG560 ซึ่งเป็นชิฟขนาด 560ขา โดยจำนวนองค์ประกอบกลุ่มต่างๆ ที่มีอยู่ในชิฟและจำนวนที่ถูกใช้งานไปแสดงดังตารางที่ 6.1 ส่วนตารางที่ 6.2 แสดงถึงค่าของเวลาที่หน่วงที่จะเกิดขึ้นในกรณีต่างๆที่จะต้องคำนึงถึงเมื่อจะนำแบบงานนี้ไปใช้งาน

ตารางที่ 6.1 จำนวนองค์ประกอบภายใน XCV1600E BG560-6 ที่ถูกใช้ในการทำ Implementaion

ลำดับ	องค์ประกอบของชิฟ(Element)	จำนวนที่ใช้	จำนวนที่มี	เปอร์เซ็นต์
1	Number of External GCLKIOBs	2	4	50
2	Number of External IOBs	9	404	2
3	Number of LOCed External IOBs	0	9	0
4	Number of SLICEs	1334	15552	8
5	Number of GCLKs	4	4	50

ตารางที่ 6.2 ค่าเวลาหน่วงที่เกิดขึ้นภายใน FPGA

Delay Summary Report	Delay
Average Connection Delay for this Design is	2.559 ns
Maximum Pin Delay is	14.346 ns
Average Connection Delay on the 10 Worst Nets is	10.791 ns

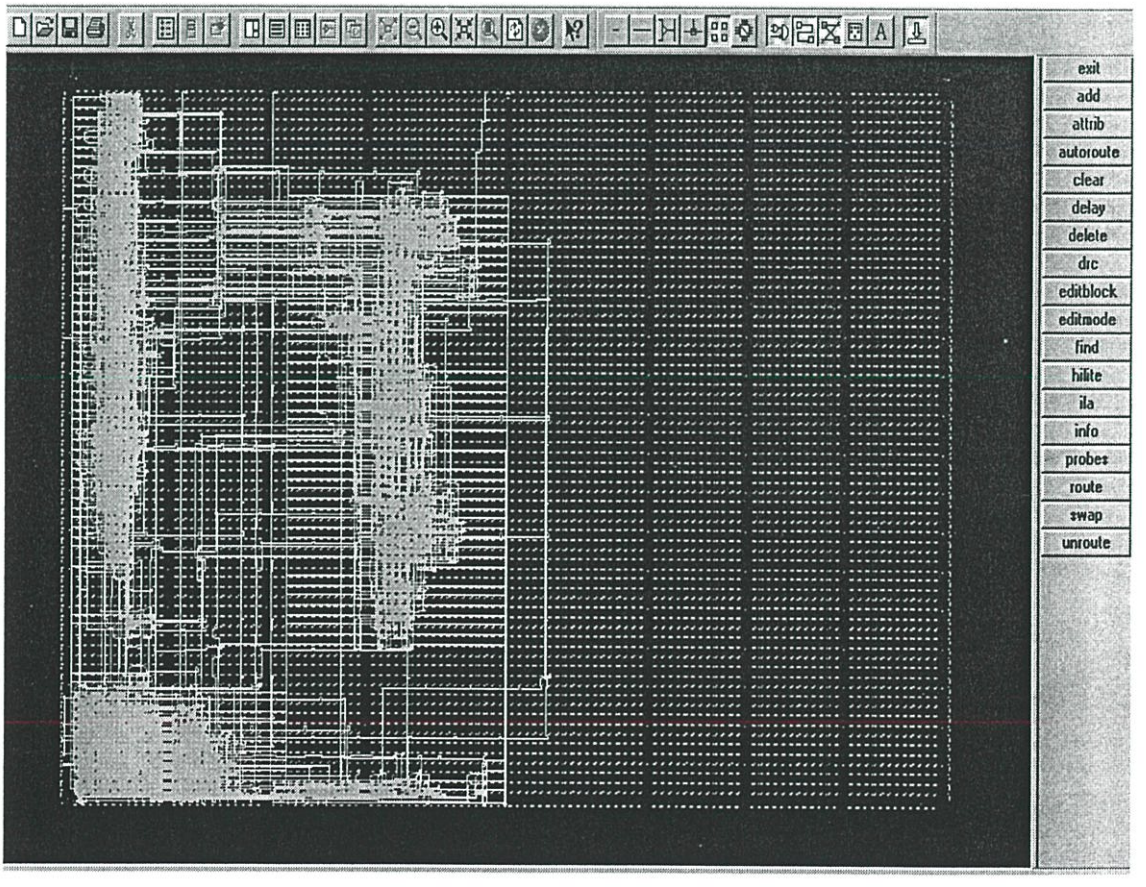
จากตารางที่ 6.1 จะพบว่ามีการใช้งานของ CLB ไปทั้งหมด 1334 Slice คิดเป็น 8% ของปริมาณทั้งหมดที่มีอยู่ภายในชิฟเบอร์ XCV1600E BG560 ดังนั้นถ้าหากจะเลือกใช้ชิฟที่มีขนาดเล็กก็จะให้ประสิทธิภาพในการใช้พื้นที่เพิ่มขึ้นซึ่งทำให้ต้นทุนในการสร้างถูกลงได้มากกว่าหลายเท่า

จากตารางที่ 6.2 เป็นข้อมูลที่เกี่ยวกับค่าของเวลาหน่วงที่ได้จากการทำ Implement บนชิฟที่ได้กำหนดในตอนต้นซึ่งพบว่าจะมีค่าหน่วงเวลาของเส้นทางการเชื่อมต่อข้อมูล ในเส้นทางที่ยาวที่สุดภายในชิฟประมาณ 10.791 ns และค่าเวลาหน่วงของขา I/O ประมาณ 14.346 ns

ส่วนตารางที่ 6.3 แสดงค่าของเวลาหน่วงสูงสุดของระบบที่เกิดขึ้นเป็น 13.213ns ซึ่งเป็นการตอบสนองเพื่อให้เกิดการส่งผ่านของข้อมูลภายในชิฟไปเป็นสัญญาณเอาท์พุตที่จะต้องใช้เวลาที่มากที่สุดภายหลังจากการเปลี่ยนแปลงของสัญญาณนาฬิกาของระบบ แต่อย่างไรก็ตามจากค่าของคาบเวลาที่ใช้ในการทำงานน้อยที่สุดเป็น 18.370ns ทำให้ระบบสามารถทำงานที่ความถี่ได้สูงถึง 54.437 MHz และในรูปที่ 6.4 แสดงให้เห็นถึงโครงสร้างการเชื่อมต่อภายในระหว่าง CLB ต่างๆ

ตารางที่ 6.3 ค่าเวลาหน่วงและความเร็วสูงสุดที่ระบบสามารถทำงานได้

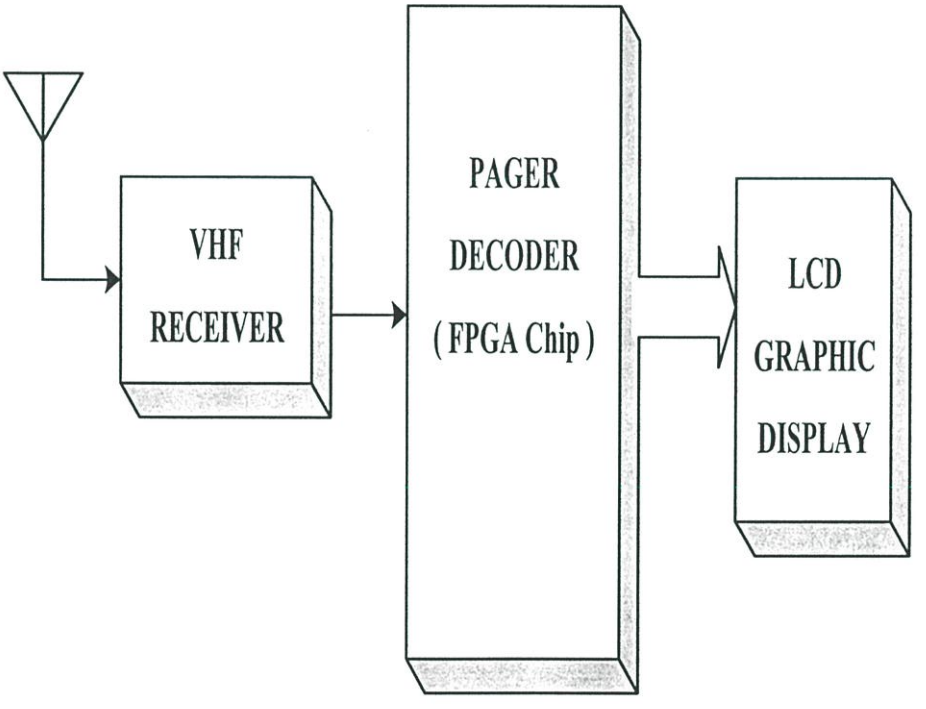
Timing Constraint	
Minimum Period	: 18.370ns (Maximum Frequency : 54.437 MHz)
Minimum input arrival time before clock	: 13.202 ns
Maximum output required time after clock	: 10982 ns
Maximum combination path delay	: 13.213 ns



รูปที่ 6.10 โครงสร้างการเชื่อมต่อภายในระหว่าง CLB

6.3 การทดสอบการทำงานจริงของวงจร

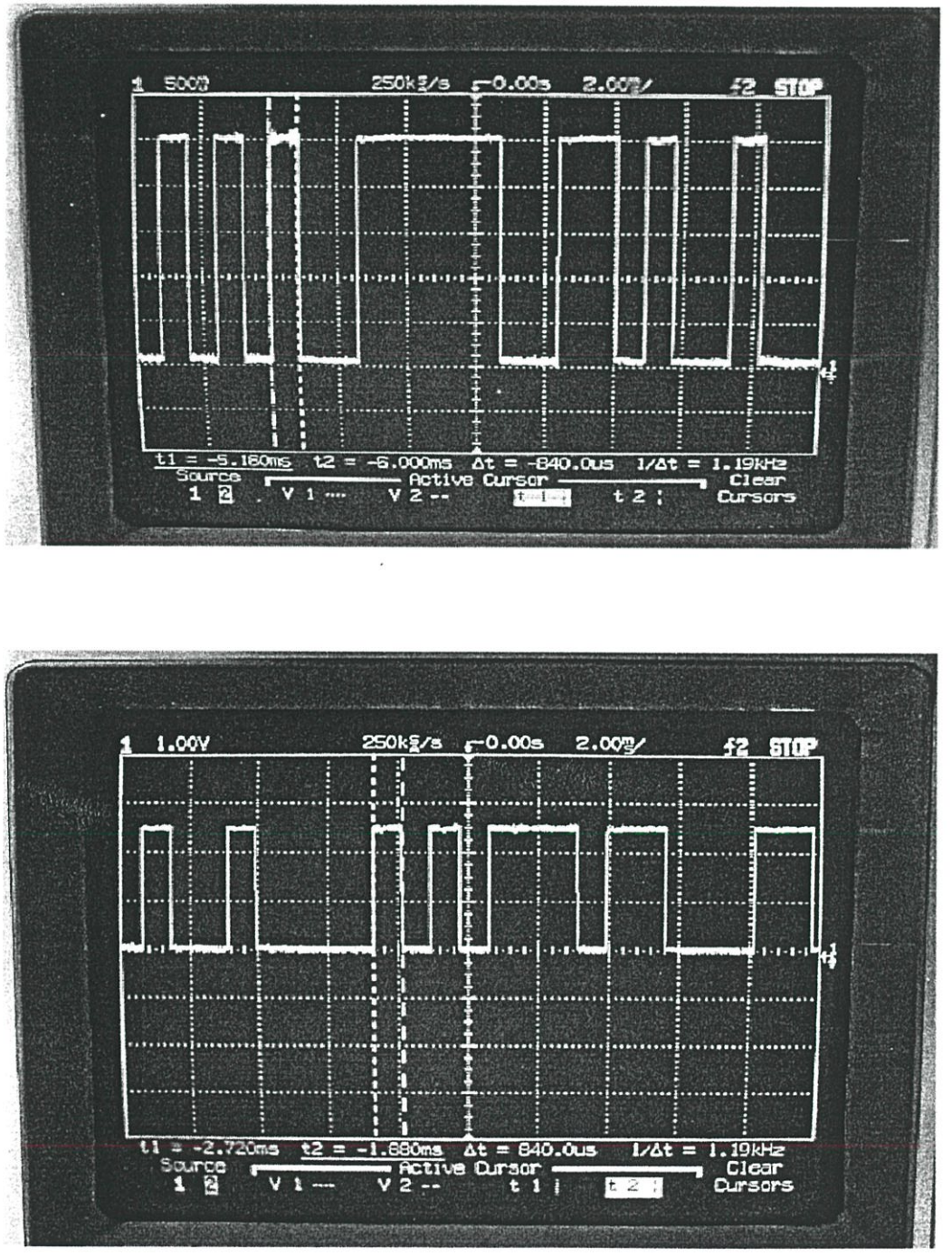
ประกอบไปด้วยบล็อกไดอะแกรมทดสอบการทำงานจริงของวงจรซึ่งประกอบไปด้วยภาครับสัญญาณวิทยุเพจเจอร์โดยใช้ภาครับสัญญาณวิทยุเพจเจอร์จริงของบริษัท NEC รุ่น PA4PY-26T ความถี่ใช้งานที่ 279.550 MHz และหมายเลขเรียกขาน(PLUGCODE) คือ 2061423 เป็นระบบ POCSAG โดยมีความเร็วในการรับส่งข้อมูลที่ 1200 บิตต่อวินาที ทำหน้าที่ในการรับสัญญาณเพจเจอร์เข้ามา ซึ่งข้อมูลที่รับมาได้จะถูกทำการถอดรหัสเพจเจอร์และบันทึกข้อมูลลงในภาคถอดรหัสในระบบ POCSAG โดยใช้เอฟพีจีเอ ซึ่งออกแบบให้ทำงานเป็นวงจรรวมภายในชิฟเดียวโดยใช้อุปกรณ์เอฟพีจีเอ ตระกูล Virtex-E เบอร์ XCV 1600E BG560 –6 ของบริษัท Xilinx ส่วนการแสดงผลข้อมูลข่าวสารจะนำแสดงผลโดยใช้อุปกรณ์ LCD Graphic Display ของบริษัท Sila Research Co., Ltd. ซึ่งแสดงผลเป็นภาษาไทยและภาษาอังกฤษซึ่งเป็นไปตามมาตรฐานของ สมอ. ดังแสดงในรูปที่ 6.11



รูปที่ 6.11 บล็อกไดอะแกรมทดสอบการทำงานจริงของวงจร

6.3.1 ภาครับสัญญาณวิทยุ (VHF Receiver)[12]

โดยแสดงผลการรับสัญญาณเบสแบนด์(NRZ) ซึ่งเป็นข้อมูลจากภาครับสัญญาณวิทยุของเครื่องรับเพจเจอร์จริงโดยมีความเร็วในการส่งข้อมูลที่ 1200 บิตต่อวินาที ดังแสดงในรูปที่ 6.12

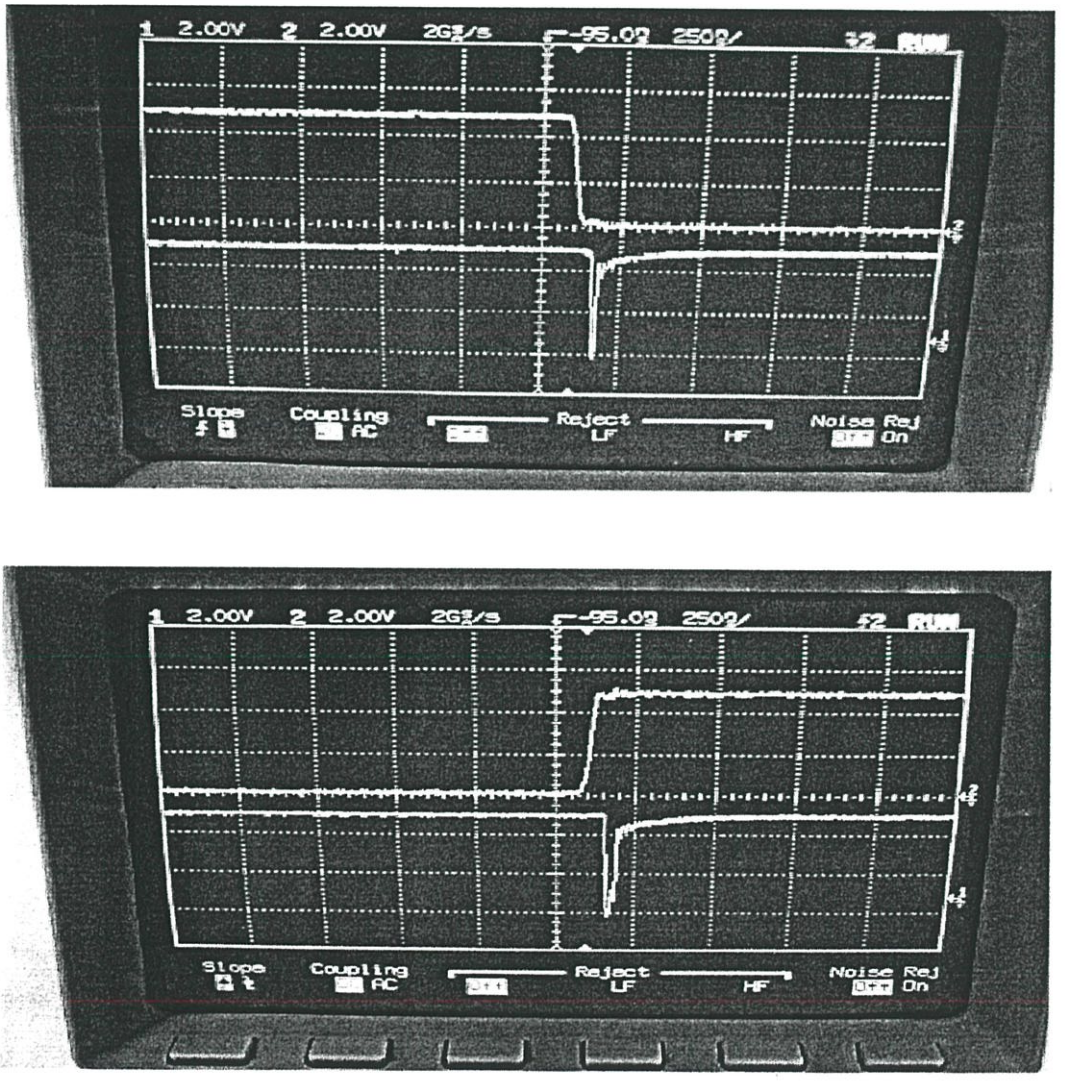


รูปที่ 6.12 สัญญาณเบสด์แบนด์(NRZ)จากภาครับสัญญาณวิทยุ

6.3.2 ภาคกู้สัญญาณนาฬิกา(Clk Recovery)

6.3.2.1 วงจรหาการเปลี่ยนของขอบสัญญาณ

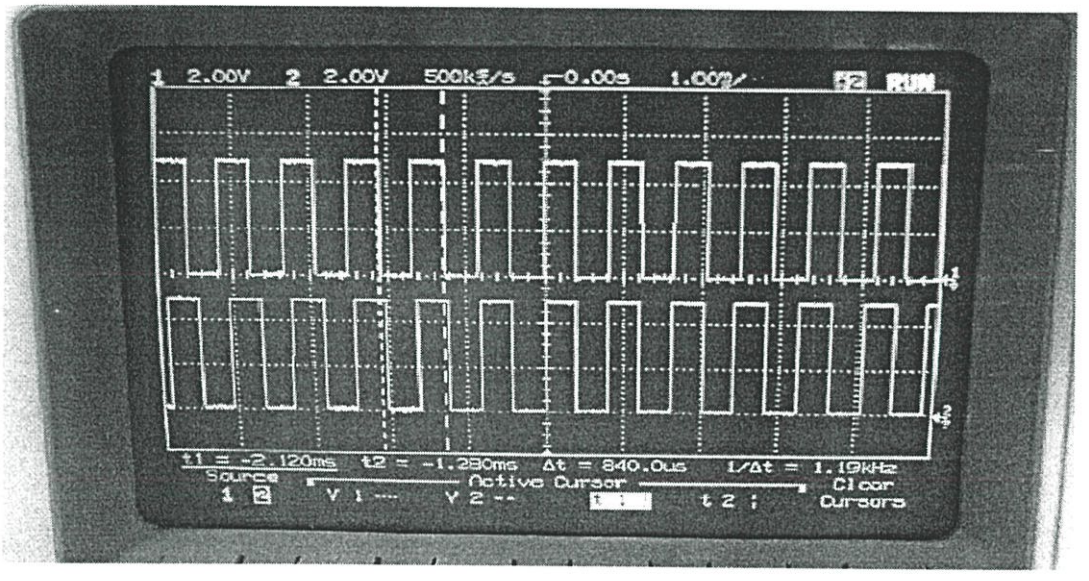
โดยใช้วงจรนอทเกทหน่วงสัญญาณ ที่อินพุทแล้วทำการเอ็กซ์คลูซีฟออร์เข้ากับสัญญาณ NRZ เดิมซึ่งจะทำให้เอาท์พุทของเอ็กซ์คลูซีฟออร์เกทได้สัญญาณพัลซ์แคบๆซึ่งเกิดขึ้นทุกๆการเปลี่ยนแปลงสัญญาณอินพุท โดยสัญญาณนี้จะเป็นสัญญาณเคลียร์(Clear) ให้กับวงจรหาร 16 โดยที่อินพุทเป็นความถี่ 19.2 kHz ซึ่งขนาด 16 เท่าของความถี่บิตเรท ดังนั้นความถี่ของสัญญาณนาฬิกาที่เอาท์พุทของวงจรหาร 16 ซึ่งจะต่อไปยังเฟสล็อกลูป(PLL) ซึ่งทำหน้าที่กำหนดความถี่สัญญาณนาฬิกาขนาด 1200 Hz เพื่อซิงส์โครไนซ์กับสัญญาณเบสแบนด์ที่รับเข้ามาดังแสดงใน รูปที่ 6.13



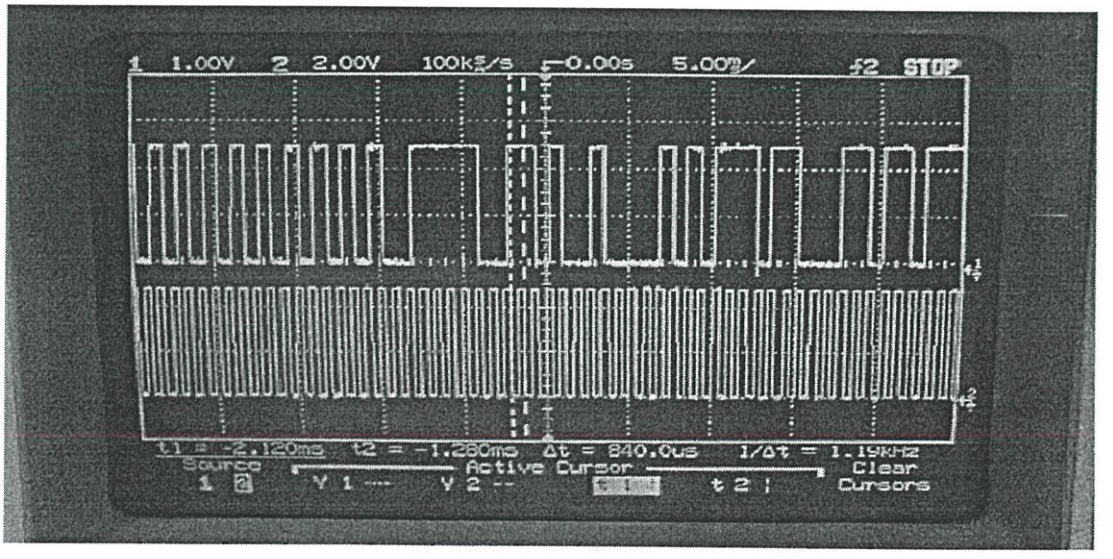
รูปที่ 6.13 สัญญาณซิงส์โครไนซ์กับสัญญาณเบสด์แบนด์(NRZ)

6.3.2.2 สัญญาณนาฬิกาที่กู้กลับคืน

การทดสอบการทำงานโดยการป้อนสัญญาณคลื่นสี่เหลี่ยม(Square Wave) ขนาดความถี่ 1200 Hz เพื่อให้วงจรกู้สัญญาณสร้างสัญญาณนาฬิกากลับคืนมาโดยอาศัยวงจรหาการเปลี่ยนขอบสัญญาณ และวงจรเฟสล็อกลูปเพื่อทำการล็อกความถี่ให้มีความถี่ที่ 1200 Hz คงที่ เพื่อใช้ในการซิงส์โครไนส์กับสัญญาณอินพุทโดยสัญญาณนาฬิกาที่กู้กลับคืนมาได้ใช้ในการจัดเก็บข้อมูลที่รับเข้ามาได้ดังในรูปที่ 6.14 และทำการทดสอบการรับเข้ามูลจริงที่รับมาจากภาครับสัญญาณเพจเจอร์จริงดังแสดงในรูปที่ 6.15



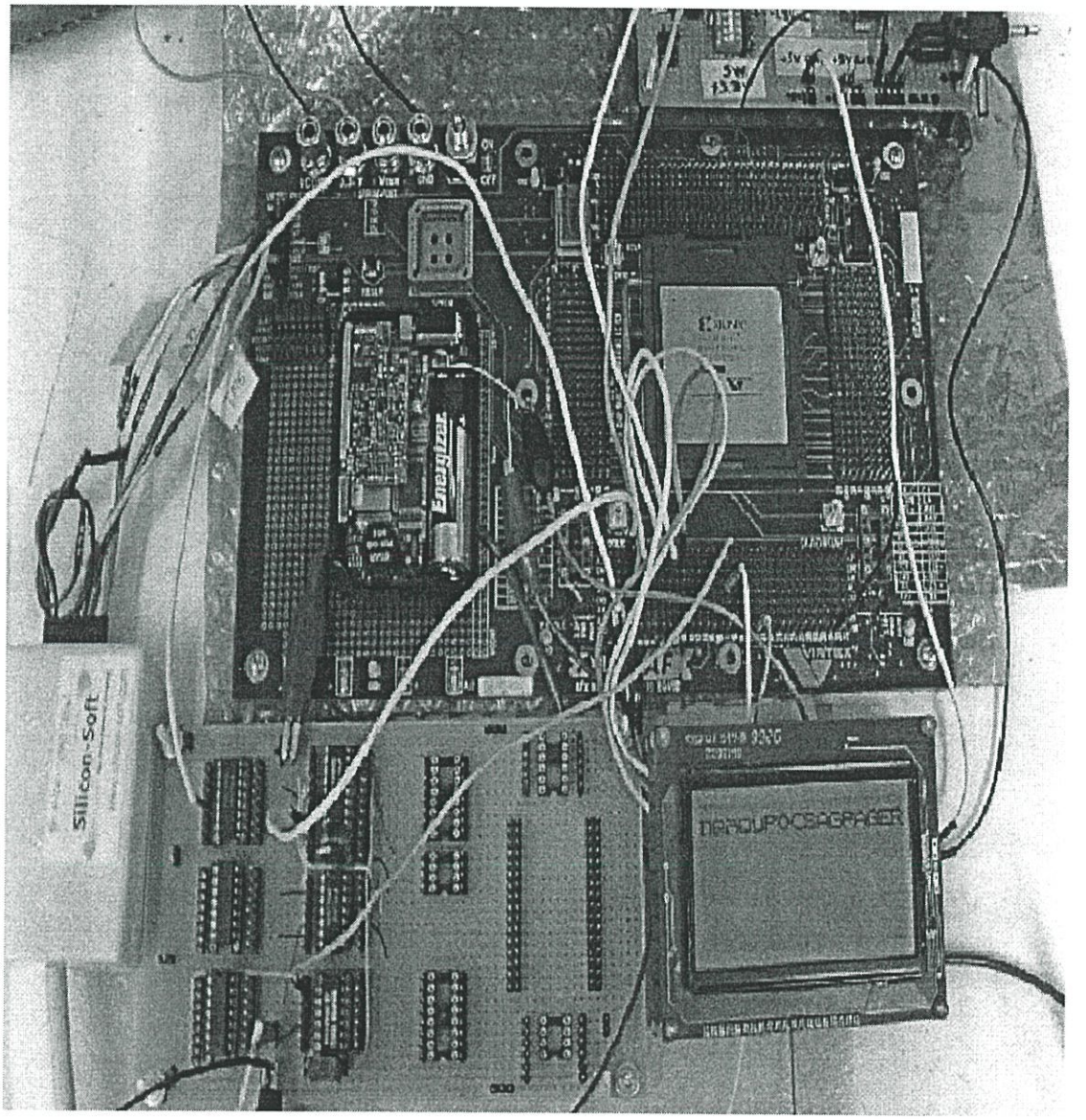
รูปที่ 6.14 สัญญาณนาฬิกาที่กู้กลับคืนมาเมื่อทดสอบกับสัญญาณ Square Wave ที่ความถี่ 1200 Hz



รูปที่ 6.15 สัญญาณนาฬิกาที่กู้กลับคืนมาเมื่อเปรียบเทียบกับสัญญาณ NRZ ที่รับได้

6.3.3 ฮาร์ดแวร์สำหรับทดสอบการทำงาน

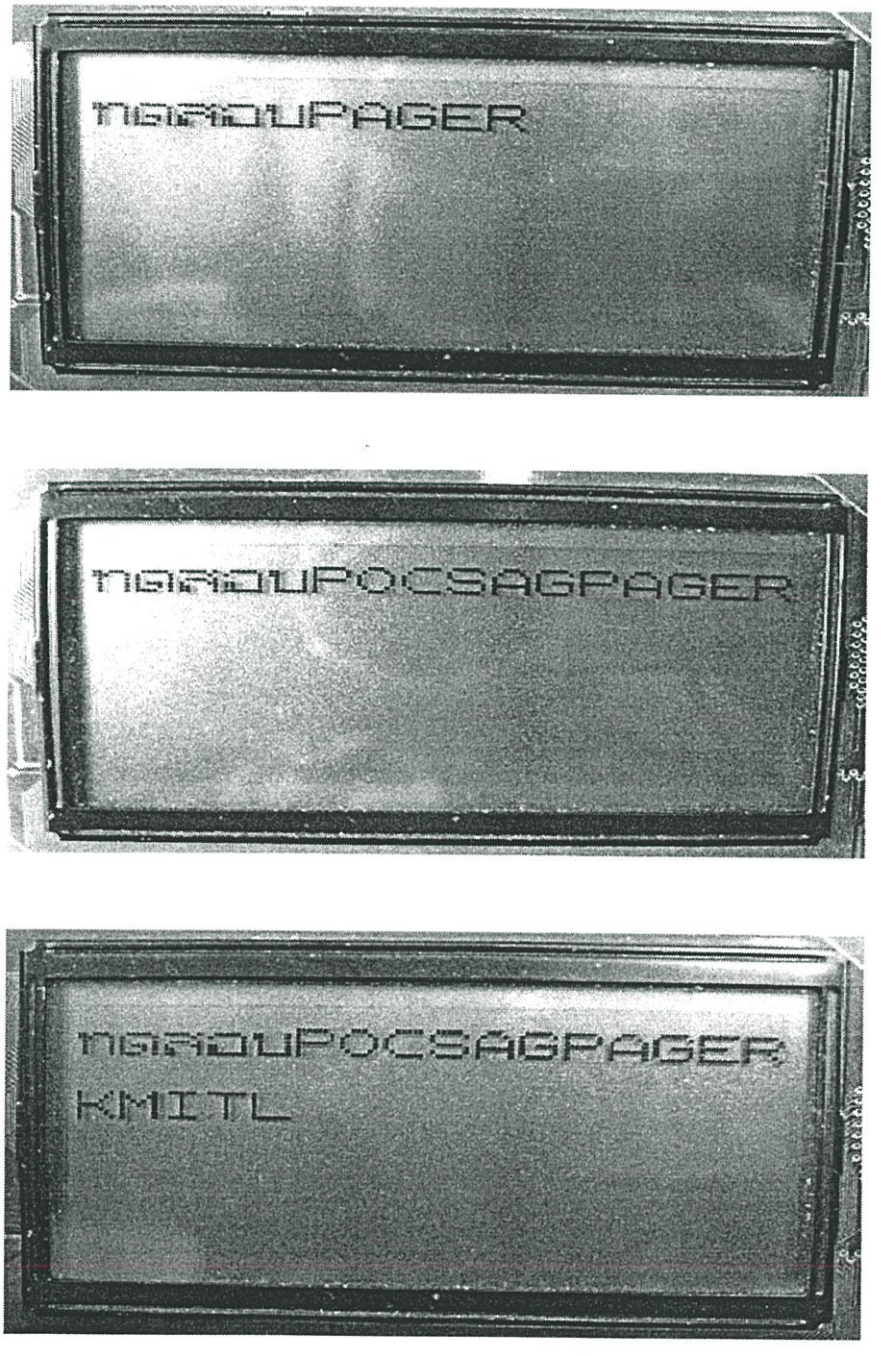
ประกอบด้วยภาครับวิทยุของเพจเจอร์ซึ่งทำหน้าที่ในการรับข่าวสารข้อมูลเพื่อให้ภาคถอดรหัสเพจเจอร์ในการจัดเก็บข้อมูลและนำไปแสดงผลข้อมูลยังภาคแสดงผล(LCD Graphic Display) ดังแสดงในรูปที่ 6.16



รูปที่ 6.16 วงจรทดสอบการทำงาน

6.3.4 ทดสอบการรับส่งข้อมูลเพจเจอร์

โดยการทดสอบการรับข้อมูลที่ส่งมาจากศูนย์เพจเจอร์(Worldpage Center) ของบริษัท ทศท.คอปอเรชั่น จำกัด มหาชน โดยใช้เรียก 141,142 และหมายเพจเจอร์คือ 3002638 ซึ่งเป็นเพจเจอร์ในระบบ POCSAG ซึ่งสามารถรับข่าวสารข้อมูลได้ถูกต้องดังแสดงในรูปที่ 6.17



รูปที่ 6.17 ผลทดสอบการรับข้อมูลเพจเจอร์

บทที่ 7
สรุปผลการวิจัยและข้อเสนอแนะ

7.1 สรุปผลการวิจัย

ในการวิจัยนี้นำเสนอแนวทางการออกแบบตัวถอดรหัสในระบบ POCSAG ในวิทยุติดตามตัวซึ่งมีขั้นตอนการออกแบบวงจรในส่วนต่างๆได้อย่างครบถ้วนดังที่กล่าวมาแล้ว เว้นแต่ในส่วนการตรวจสอบความถูกต้องของข้อมูล(Parity Check Bits) ซึ่งเป็นการตรวจสอบข้อมูลด้วยแบบ BCH(Bose-Chaudhuri-Hocquenghem) และ พาริตี้เช็ค(Event Parity Bits) ซึ่งไม่ได้ออกแบบใช้งานในขณะนี้ เนื่องจากการส่งข้อมูลจากศูนย์เพจเจอร์มีความแรงในการส่งข้อมูลมายังเพจเจอร์ทำให้ข้อมูลที่รับเข้ามาได้มีความถูกต้องสูงในระดับหนึ่ง ดังนั้นในการพัฒนาต่อไปจะต้องสร้างวงจรในส่วนของการตรวจสอบความถูกต้องของข้อมูลขึ้นมาใช้งานเพื่อให้ข้อมูลที่รับเข้ามาได้มีความถูกต้องสูงสุด ซึ่งจะเห็นว่าสามารถพัฒนาตัวถอดรหัสในระบบ POCSAG สำหรับวิทยุติดตามตัวโดยใช้เอฟพีจีเอในขณะนี้ ซึ่งประกอบด้วยภาคการทำงานของแต่ละวงจรหลายๆส่วนโดยแต่ละส่วนการสามารถเปลี่ยนแปลงความจุในการเก็บข้อมูลที่รับเข้ามาได้ดังเช่น การจัดเก็บข้อมูลที่เข้ามาในส่วนของภาค DECODER CODEWORD เราสามารถที่จะขยายจำนวนการจัดเก็บข้อมูลซึ่งสามารถทำได้โดยการเพิ่มความจุของ RAM เพื่อให้ได้ตามความต้องการในการกำหนดการรับข้อมูลเข้ามา และในส่วนของหมายเลขเรียกขานเพจเจอร์ก็สามารถที่จะทำการเปลี่ยนหมายเลขเรียกขานเป็นหมายเลขอะไรก็ได้ซึ่งอยู่ในส่วนของภาค ADDRESS CODEWORD ซึ่งจะเห็นได้ว่าสามารถเปลี่ยนแปลงการทำงานและพัฒนาวงจรได้อย่างรวดเร็ว โดยอาศัยการออกแบบ วงจรโดยใช้ภาษาวีเอชดีแอลในการอธิบายลักษณะพฤติกรรมของวงจร จากนั้นจึงทำการดาวน์โหลดข้อมูลทางลอจิกลงในชิฟเอฟพีจีเอแล้วนำไปทดสอบการทำงาน ซึ่งชิฟเอฟพีจีเอที่ได้ทำการออกแบบสามารถทำงานได้อย่างถูกต้องและมีประสิทธิภาพ

จากการทดสอบการทำงานของตัวถอดรหัสข้อมูลพบว่าสามารถรับส่งข้อมูลข่าวสารที่ส่งมาจากศูนย์ส่งสัญญาณเพจเจอร์สามารถรับข้อมูลข่าวสารได้อย่างถูกต้องและในบางจังหวะการรับข้อมูลเกิดการผิดพลาด สืบเนื่องมาจากขาดวงจรการตรวจสอบข้อมูลที่รับเข้ามาได้ ซึ่งจากผลการทดสอบการทำงาน เราสามารถนำไปประยุกต์ใช้งานในการสั่งงานผ่านระบบวิทยุติดตามตัวได้ในพื้นที่บริเวณปริมณฑลและบริเวณใกล้เคียง เนื่องต้องอาศัยความแรงของสัญญาณที่ส่งมาจากศูนย์เพจเจอร์เพื่อความถูกต้องของข้อมูล

7.2 อุปสรรคที่พบในการวิจัย

ในการทดลองการทำงานของวงจรจะมีปัญหาในส่วนภาครับสัญญาณวิทยุ ซึ่งได้ทดลองใช้ไอซี MC3362 ของบริษัท Motorola ในการทำงานรับส่งข้อมูลซึ่งการทำงานของวงจรที่ไม่เสถียรจึงได้ทำการเปลี่ยนภาครับสัญญาณวิทยุมาใช้ภาครับวิทยุของเครื่องรับเพจเจอร์จริงเพื่อแก้ปัญหาการรับส่งข้อมูล และปัญหาอีกส่วนหนึ่งมาจากการทำงานของวงจรกู้สัญญาณนาฬิกาที่ไม่ซิงส์โครไนซ์กับสัญญาณอินพุทที่เข้ามาในบางจังหวะทำให้การจัดเก็บข้อมูลผิดพลาดไปบ้าง เนื่องมาจากการทำงานของภาคกู้สัญญาณนาฬิกา

7.3 ข้อเสนอแนะในการพัฒนา

1. การทดสอบการทำงานของวงจรทั้งหมดผู้วิจัยได้นำภาครับสัญญาณวิทยุมาจากเครื่องรับเพจเจอร์จริงเนื่องจากทดลองสร้างภาคเครื่องรับวิทยุขึ้นมาโดยใช้ชิฟ MC3362 ของบริษัท Motorola แต่ไม่ประสบความสำเร็จเนื่องจากความไม่เสถียรของวงจรตลอดจนขาดประสบการณ์ในการออกแบบในด้าน RF Design จึงน่าจะมีการออกแบบและพัฒนาภาครับสัญญาณวิทยุขึ้นมาใช้งานเองในการพัฒนาต่อๆไป

2. พัฒนาวงจรในส่วนของการตรวจสอบความถูกต้องของข้อมูลที่รับเข้ามาได้โดยวิธีของ BINARY BCH CODE และ ตรวจเช็คพาริตี้บิท(EVENT PARITY BITS)ทั้งหมดในแต่ละเฟรมข้อมูลที่รับเข้ามา

3. ในงานวิจัยนี้ใช้ชิฟ FPGA เพื่อสร้างเป็นต้นแบบเท่านั้น ผู้วิจัยเลือกใช้ชิฟซึ่งมีขนาดที่ใหญ่เกินความจำเป็นอยู่มาก หากจะนำไปใช้งานจริงควรปรับเปลี่ยน ขนาดหรือตระกูลของชิฟให้มีความเหมาะสมกับงานมากขึ้น ซึ่งจะสามารถลดต้นทุนการผลิตลงได้ประมาณ 40%-80% ทั้งนี้ ขึ้นอยู่กับเบอร์และตระกูลของชิฟที่เลือกใช้ เช่น ชิฟในตระกูล Spartan II ซึ่งเป็นชิฟในตระกูลที่มีองค์ประกอบภายในชิฟที่เกือบจะทดแทนกันได้ทั้งหมด

3. การนำFPGA มาประยุกต์ใช้งานในงานประมวลผลสัญญาณทางดิจิตอลเป็นเทคโนโลยีที่มีความโดดเด่นมากเนื่องจากมีความยืดหยุ่นในการประมวลของสัญญาณที่มีความซับซ้อนสูง แต่การประยุกต์ใช้งานด้วย FPGA ให้มีประสิทธิภาพสูงสุดได้นั้น ผู้ออกแบบต้องมีความรู้และเชี่ยวชาญในหลายๆด้านได้แก่ เทคนิคการออกแบบระดับสูง(High Level Design) เทคนิคการออกแบบด้วยภาษาวีเอชดีแอล(VHDL) กฎการออกแบบ(Design Rule) ความเชี่ยวชาญเฉพาะทางที่ต้องการนำไปประยุกต์ใช้ ความเข้าใจในเรื่องฮาร์ดแวร์ของวงจรดิจิตอล ตลอดจนความรู้เกี่ยวกับรูปแบบและคุณสมบัติของ FPGA แบบต่างๆ เป็นต้น

เอกสารอ้างอิง

[1] วารสารเซมิคอนดักเตอร์ ฉบับที่ 73 ตุลาคม 2529 หน้าที่ 162 - 169 และฉบับที่ 159 พฤษภาคม 2539 หน้าที่ 66 – 78 และ ศูนย์เพจเจอร์(Worldpage) บริษัท ทศท.คอปอเรชั่นจำกัด มหาชน

[2] British Telecom " A standard Code for Radio Paging report of Post Office Code Standard Advisory Group (POCSAG) " June 1978, CCIR Recommendation 584, Radiopaging Code 1

[3] TLCD-164 Version1.0 Thai Lcd Module,Sila Research Co., Ltd.

[4] Astron Logic Reasearch and Development, " เปิดโลก FPGA กับบอร์ด WIZARD PLD- A01" Available: http ://www.astronlogic.com/

[5] ชาติชาย ดิษฐกุล, Introduction to VHDL. กรุงเทพฯ ภาควิชาวิศวกรรมคอมพิวเตอร์

[6] บุญอนันต์ เกียงเอีย "A Design Of Medical Image Capturing System Using FPGA" ปริญญาวิศวกรรมศาสตรมหาบัณฑิต สาขา วิศวกรรมอิเล็กทรอนิกส์ สถาบัน เทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2545

[7] The Programmable Logic Data Book 2000, Web: http://www.xilinx.com

[8] วัชระ ภัคมาตร์ "Design And Implementation Of and FPGA Based For Voice Echo Canceller" ปริญญาวิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมไฟฟ้า สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณ ทหารลาดกระบัง ปีการศึกษา 2545

[9] Stefan Sjoholm ABB Industrial System,Sweden and Lennart Lindh Malardalens University Sweden"VHDL for Designer",Prentice-Hall

Charles H. Roth, Jr , Digital Systems Design Using VHDL. Boston : PWS Publishing, 1998

[10] Xilinx Corporation " Foundation ISE 4.2i "

"FPGA Design Workshop" ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ ฝ่ายออกแบบวงจรรวม(TMEC)

[11] อุดมพร สนันทชัยกุล,กฤดากร กล่อมการ,ดลชัย สุขเจริญผล, วิชัย สุรพัฒน์ "การพัฒนาเพจเจอร์การ์ดบนไมโครคอมพิวเตอร์" การประชุมวิชาการทางวิศวกรรมไฟฟ้าครั้งที่ 18

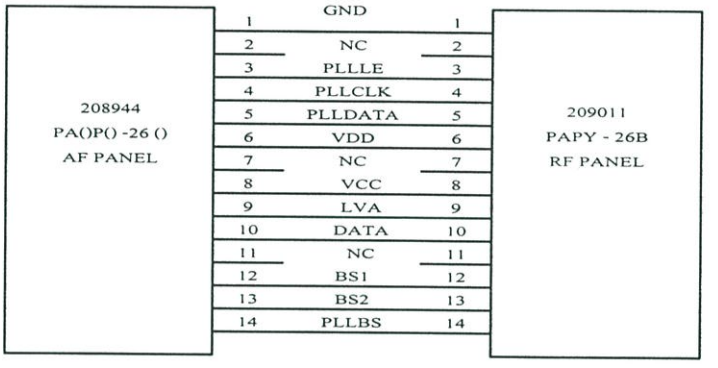
[12] บริษัท ทศท.คอปอเรชั่น จำกัด มหาชน ผู้ให้บริการเพจเจอร์(Worldpage) รุ่น NEC PA4PY-26T 279.550 เรียกหมายเลข 142 3002638 ของ NEC Corporation

[13] FM SCA Pager ปริญญาวิศวกรรมศาสตร์บัณฑิต สาขาวิศวกรรมโทรคมนาคม สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2543

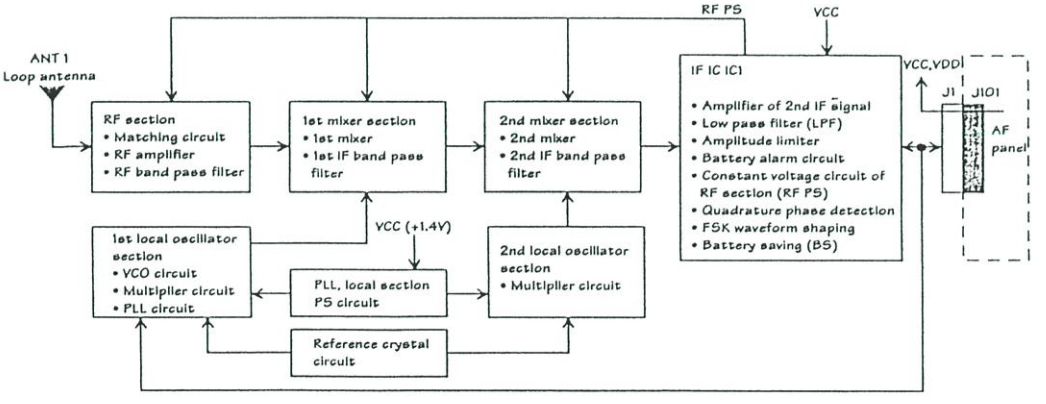
ภาคผนวก ก

เครื่องรับเพจเจอร์ในระบบ POCSAG

เครื่องรับเพจเจอร์ที่ใช้ภาครับสัญญาณวิทยุในการทดสอบการทำงานจะเป็นรุ่น PA4PY-2T ใช้ความถี่ 279.550 MHz โดยหมายเลขเรียกขานคือ 2061428 เป็นของบริษัท NEC Corporation โดยมีบล็อกไดอะแกรมและวงจรการทำงานของเครื่องรับเพจเจอร์

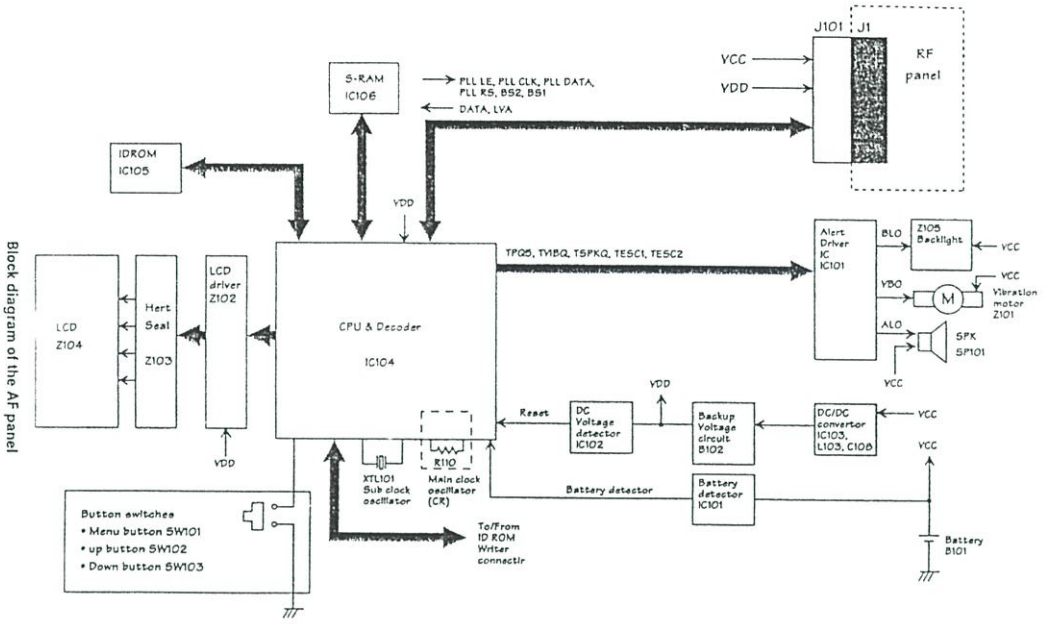


รูปที่ ก.1 บล็อกไดอะแกรมของเครื่องรับเพจเจอร์

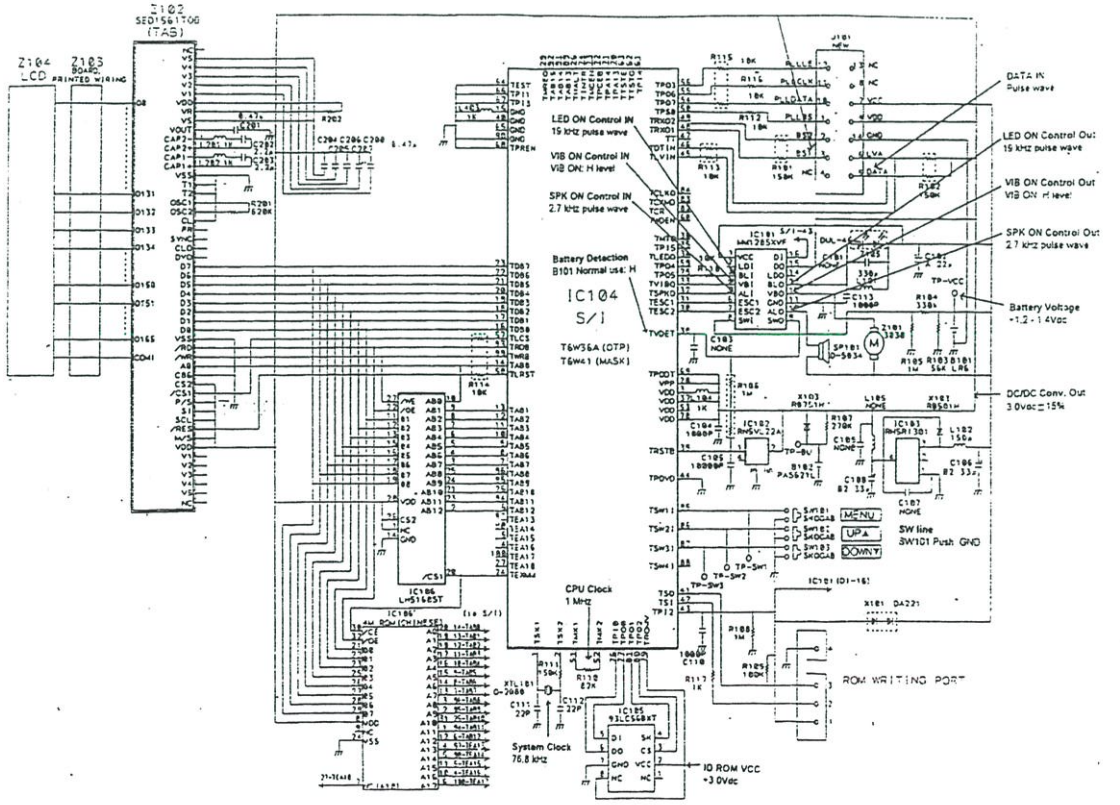


Overview of the RF panel

รูปที่ ก.2 บล็อกไดอะแกรมภาครับวิทยุติดตามตัว(RF PANEL)



รูปที่ ก.3 บล็อกไดอะแกรมภาค AF PANEL



รูปที่ ก.4 วงจรภาค AF PANEL

ภาคผนวก ข

ภาคแสดงผล(LCD Graphic Display)

ภาคแสดงผลซึ่งเป็น LCD รุ่นกราฟฟิกขนาด 128 x 64 Dots รุ่น TLCD-164 VERSION 1.0 THAI LCD MODULE ของบริษัท ศิลารีเสิร์ช จำกัด ที่มีส่วนไมโครคอนโทรล เลอร์อยู่ในชุดเดียวกัน ทำให้ใช้งานง่ายและสะดวกมาก ต่อควบคุมผ่านสัญญาณ RS232 หรือ 3 BIT Serial ซึ่งแสดงผลเป็นภาษาไทย (รหัส สมอ.) ที่มีการจัดบรรทัดอย่างถูกต้องได้ในตัวเอง ซึ่งมีคุณสมบัติการทำงานคือ

1. รับข้อมูลและคำสั่งควบคุมได้ 2 รูปแบบ คือแบบ RS232 และแบบ 3 Bit Serial
2. แสดงอักษรไทย-อังกฤษได้ 16 อักษร 4 บรรทัดหรือในแบบกราฟฟิกขนาด128x64 Dots
3. มี VR ปรับความเข้มของ LCD
4. ใช้แหล่งจ่ายไฟ + 5VDC เท่านั้น
5. สามารถเลือกสัญญาณควบคุมได้ 3 แบบ คือ RS232 มาตรฐาน, RS232 แบบ LOGIC และ 3 BIT SERIAL
6. มีชุดคำสั่งในการควบคุม 8 คำสั่ง
7. สามารถแสดงข้อมูลในแบบกราฟฟิกได้ พร้อมทั้งมีคำสั่งสำหรับ On/Off Cursor และทำ Backspace ได้

ชุดคำสั่งการควบคุม

คำสั่งในการควบคุมจะมีทั้งหมด 8 คำสั่ง โดยมีรายละเอียดจะเป็นดังนี้

: CXX…. X <CR>

: คือรหัสนำของคำสั่ง(3AH)

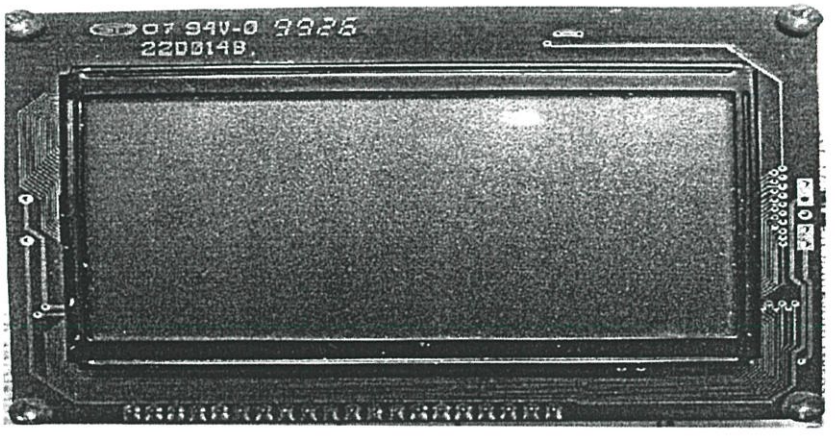
C คือรหัสคำสั่งตั้งแต่ 1-8 (31H – 38H)

XX….. X คือข้อมูลติดตามของแต่ละคำสั่ง ซึ่งอาจจะมีหรือไม่มีก็ได้ รวมมีความยาวตามกำหนดในแต่ละคำสั่งด้วย

<CR> คือรหัสลงท้ายของคำสั่ง (ODH)

ตารางที่ ข.1 ชุดคำสั่งของ LCD Graphic Interface

คำสั่ง	รายละเอียด	BUSY TIME
:1RCC<CR>	Set Character Row & Colum R = 0-3 CC = 00 - 15	4 mS
:2XX .. X <CR>	Set Character (Max = 25 Char)	115 mS
:3PPYY <CR>	Set Graphic Page & Y PP = 00 –15 YY = 00 – 63	6mS
:4XX .. X <CR>	Set Graphic (Max = 25 Data)	3mS
:5 <CR>	Clear LCD	50 mS
:6 <CR>	Cursor ON	3 mS
:7 <CR>	Cursor OFF	6 mS
: 8 <CR>	Back Space	6 mS



รูปที่ ข.1 ตัวแสดงผล(LCD Graphic Display)

ภาคผนวก ค

ไฟล์ข้อมูลภาษา VHDL ของตัวถอดรหัสเพจเจอร์

DECODER_CODEWORD_ALL.VHD

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

ENTITY DECODER_CODEWORD IS
   PORT(RESET                 : in std_logic;
        CLK_NRZ_IN            : in std_logic;
        CLK_READ              : in std_logic;
        NRZ_IN                : in std_logic;
        READ                  : in std_logic;
        ADDRESSS_READ         : in std_logic_vector ( 5 downto 0 );
        PLUG_CODE_SYNC        : out std_logic;
        DATA_OUTPUT           : out std_logic_vector( 31 downto  0 ));
END DECODER_CODEWORD;

ARCHITECTURE  Behavioral of DECODER_CODEWORD IS
        COMPONENT frame_sync_detect_decoder IS
        PORT  ( RESET         : in std_logic;
                CLK           : in std_logic;
                NRZ_IN        : in std_logic;
                PLUG_CODE_SYNC : out std_logic;
                FRAME_SYNC    : out std_logic  );
        END COMPONENT;
```

```
COMPONENT  frame_sync_delay_decoder IS
PORT ( RESET                : in std_logic;
       CLK                  : in std_logic;
       FRAME_SYNC           : in std_logic;
       FRAME_DELAY          : in std_logic;
END COMPONENT;
COMPONENT  serial_parallel_shift_decoder IS
PORT ( RESET                : in std_logic;
       CLK                  : in std_logic;
       NRZ_IN               : in std_logic;
       FRAME_DELAY          : in std_logic;
       DATA_OUT             : out  std_logic_vector(31 downto  0 ));
END COMPONENT;
COMPOMENT  count_address_decoder IS
PORT  (RESET                    : in std_logic;
       FRAME_DELAY          : in std_logic;
       ENABLE_COUNT         : in std_logic;
       ADDRESS_OUT_COUNT : out std_logic_vector(5 downto  0));
END COMPONENT;
COMPONENT  write_ram_decoder IS
PORT  (RESET                : in std_logic;
       CLK                  : in std_logic;
       FRAME_DELAY          : in std_logic;
       ENABLE_SEL           : in std_logic;
       ENANLE_RAM           : out std_logic );
END COMPONENT;
COMPONENT  multiplex_address_decoder IS
PORT ( ADDRESS_WRITE     : in std_logic_vector( 5 downto  0 );
       ADDRESS_READ      : in std_logic_vector( 5 downto  0 );
       SEL               : in std_logic;
       ADDRESS_OUT_MUL : out std_logic_vector ( 5 downto 0 ));
END COMPONENT;
```

```
        COMPONENT  ram_decoder IS
        PORT  (RESET                : in std_logic;
               CLK                  :in std_logic;
               CLK_READ             : in std_logic;
               ADDRESS              : in std_logic_vector( 5 downto 0 );
               DATA                 : in std_logic_vector( 31 downto 0);
               WRITE                : in std_logic;
               READ                 : in std_logic;
               DATA_OUTPUT          : out std_logic_vector ( 31 downto  0 ) );
        END COMPONENT;
SIGNAL buf_frame_sync               : std_logic;
SIGNAL buf_frame_delay              : std_logic;
SIGNAL buf_data_out                 : std_logic_vector( 31 downto 0 );
SIGNAL buf_enable_ram               : std_logic;
SIGNAL buf_address_out_count        : std_logic_vector( 5 downto 0 );
SIGNAL  buf_enable_sel              : std_logic;
SIGNAL buf_address_out_mul          : std_logic_vector ( 5 downto 0 );

BEGIN
u1_frame_detec :    frame_sync_detect_decoder PORT MAP
                    (RESET              => reset,
                    CLK                 => clk_nrz_in,
                    NRZ_IN              => nrz_in,
                    PLUG_CODE_SYNC      => plug_code_sync,
                    FRAME_SYNC          => buf_frame_sync );
u2_frame_delay :    frame_sync_delay_decoder PORT MAP
                    (RESET              => reset,
                    CLK                 => clk_nrz_in,
                    FRAME_SYNC          => buf_frame_sync,
                    FRAME_DELAY         => buf_frame_delay );
```

```
u3_serial_parallel : serial_parallel_shift_decoder  PORT MAP
                    ( RESET              => reset,
                      CLK                => clk_nrz_in,
                      NRZ_IN             => nrz_in,
                      FRAME_DELAY        => buf_frame_delay,
                      DATA_OUT           => buf_data_out );
u4_count_address :  count_address_decoder  PORT MAP
                    (RESET               => reset,
                    FRAME_DELAY          => buf_frame_delay,
                    ENABLE_COUNT         => buf_enable_ram,
                    ADDRESS_OUT_COUNT=> buf_address_out_count);
u5_write_ram :  write_ram_decoder PORT MAP
                    (RESET               => reset,
                    CLK                  => clk_nrz_in,
                    FRAME_DELAY          => buf_frame_delay,
                    ENABLE_SEL           => buf_enable_sel,
                    ENABLE_RAM           => buf_enable_ram );
u6_multiplex_address :  multiplex_address_decoder PORT MAP
                    (ADDRESS_WRITE       => buf_address_out_count,
                    ADDRESS_READ         => address_read,
                    SEL                  => buf_enable_sel,
                    ADDRESS_OUT_MUL => buf_address_out_mul );
u7_ram :          ram_decoder PORT MAP
                    (RESET               => reset,
                    CLK                  => clk_nrz_in,
                    CLK_READ             => clk_read,
                    ADDRESS              => buf_address_out_mul,
                    DATA                 => buf_data_out,
                    WRITE                => buf_enable_ram,
                    READ                 => read,
                    DATA_OUTPUT          => data_output );
END Behavioral;
```

LCD Graphic_Interace_ALL.VHD

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

    ENTITY LCD_Graphic_Interface IS
    PORT ( RESET                : in std_logic;
           CLK                  : in std_logic;
           START_IN             : in std_logic;
           DATA_IN              : in std_logic_vector ( 7 downto 0 );
           READ_RAM_OUT         : out std_logic;
           ADDRESS_RAM_OUT      : out std_logic_vector ( 6 downto 0 );
           OUT_8CK_LCD          : out std_logic;
           LOAD_OUT_LCD         : out std_logic;
           DATA_OUT_LCD         : out std_logic );
    END LCD_Graphic_Interface;

ARCHITECTURE Behavioral of LCD_Graphic_Interface is
    COMPONENT  control_data_dis IS
    PORT ( RESET                : in std_logic;
           CLK                  : in std_logic;
           PULSE_32H_IN         : in std_logic;
           PULSE_260US_IN       : in std_logic;
           PULSE_100ms_IN       : in std_logic;
           CLEAR_PULSE_IN       : in std_logic;
           LOAD_OUT             : out std_logic;
           READ_RAM_OUT         : out std_logic;
           LOAD_ODH_OUT         : out std_logic;
           ADDRESS_OUT          : out std_logic_vector ( 6 downto 0 ) );
    END COMPONENT;
```

```
        COMPONENT delay_0dh_dis IS
        PORT  (RESET                    : in std_logic;
               CLK                      : in std_logic;
               PULSE_ODH_IN             : in std_logic;
               PULSE_0DH_OUT            : out std_logic );
        END COMPONENT;
        COMPONENT  include_all2_dis IS
        PORT  (RESET                    : in std_logic;
               CLK                      : in std_logic;
               SWITCH_IN                : in std_logic;
               LOAD_0DH_IN              : in std_logic;
               LOAD_IN                  : in std_logic;
               OFF_PULSE_260US_IN       : in std_logic;
               DATA_IN                  : in std_logic_vector ( 7 downto 0 );
               PULSE_32H_OUT            : out std_logic;
               PULSE_100MS_OUT          : out std_logic;
               PULSE_260US_OUT          : out std_logic;
               OUT_8CLK_LCD             : out std_logic;
               LOAD_OUT_LCD             : out std_logic;
               DATA_OUT_LCD             : out std_logic );
        END COMPONENT;
        COMPONENT  start_delay_dis IS
        PORT ( RESET                    : in std_logic;
               CLK                      : in std_logic;
               START_IN                 : in std_logic;
               DELAY_OUT                : out std_logic );
        END COMPONENT;
SIGNAL buf_pulse_32h                    : std_logic;
SIGNAL buf_pulse_260us                  : std_logic;
SIGNAL buf_pulse_100ms                  : std_logic;
SIGNAL buf_load_out                     : std_logic;
SIGNAL  buf_load_0dh_in                 : std_logic;
```

```
SIGNAL buf_0dh_out                  : std_logic;
SIGNAL buf_switch_out               : std_logic;

BEGIN
u1 :    control_data_dis PORT MAP
            (RESET                  => reset,
            CLK                     => clk,
            PULSE_32H_IN            => buf_pulse_32h,
            PULSE_260US_IN          => buf_pulse_260us,
            PULSE_100MS_IN          => buf_pulse_100ms,
            CLEAR_PULSE_IN          => buf_0dh_out,
            LOAD_OUT                => buf_load_out,
            READ_RAM_OUT            => read_ram_out,
            LOAD_ODH_OUT            => buf_load_0dh_in,
            ADDRESS_OUT             => address_ram_out );
u2 :    delay_0dh_dis PORT MAP
            (RESET                  => reset,
            CLK                     => clk,
            PULSE_0DH_IN            => buf_load_0dh_in,
            PULSE_0DH_OUT           => buf_0dh_out );
u3 :    include_all2_dis port map
            (RESET                  => reset,
            CLK                     => clk,
            SWITCH_IN               => buf_switch_out,
            LOAD_ODH_IN             => buf_0dh_out,
            LOAD_IN                 => buf_load_out,
            OFF_PULSE_260US_IN      => buf_load_0dh_in,
            DATA_IN                 => data_in,
            PULSE_32H_OUT           => buf_pulse_32h,
            PULSE_100MS_OUT         => buf_pulse_100ms,
            PUSLE_260US_OUT         => buf_pulse_260us,
            OUT_8CK_LCD             => out_8clk_lcd,
```

```
                LOAD_OUT_LCD            => load_out_lcd,
                DATA_OUT_LCD            => data_out_lcd );
u4 :    start_delay_dis PORT MAP
                (RESET                  => reset,
                CLK                     => clk,
                START_IN                => start_in,
                DELAY_OUT               => buf_switch_out );
end Behavioral;
```

INCLUDE_PAGER_DECODER_ALL.VHD

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;

ENTITY INCLUDE_PAGER_DECODER IS
  PORT (RESET                   : in std_logic;
        CLK_NRZ_IN              : in std_logic;
        CLK                     : in std_logic;
        CLK_READ                : in std_logic;
        START_IN                : in std_logic;
        NRZ_IN                  : in std_logic;
        LED                     : out std_logic;
        BUZZER_OUT              : out std_logic;
        OUT_BIT31               : out std_logic;
        OUT_8CLK_LCD            : out std_logic;
        LOAD_OUT_LCD            : out std_logic;
        DATA_OUT_LCD            : out std_logic );
END INCLUDE_PAGER_DECODER;
```

```
ARCHITECTURE Behavioral of INCLUDE_PAGER_DECODER IS
    COMPONENT include_pager_decoder_ctrl_include_buffer_all IS
    PORT (REESET                    : in std_logic;
          CLK_NRZ_IN                : in std_logic;
          CLK                       : in std_logic;
          CLK_READ                  : in std_logic;
          NRZ_IN                    : in std_logic;
          READ_DATA                 : in std_logic;
          ADDRESS_READ              : in std_logic_vector ( 4 downto 0 );
          BUZZER_OUT                : out std_logic;
          LED                       : out std_logic;
          OUT_BIT31                 : out std_logic;
          DATA_OUTPUT               : out std_logic_vector ( 7 downto 0 ) );
    END COMPONENT;
    COMPONENT include_data_to_lcd_graphic_dis IS
    PORT ( RESET                    : in std_logic;
          CLK                       : in std_logic;
          START_IN                  : in std_logic;
          DATA_IN                   : in std_logic_vector ( 7 downto 0 );
          READ_RAM_OUT              : out std_logic;
          ADDRESS_RAM_OUT           : out std_logic_vector ( 4 downto 0 );
          OUT_8CLK_LCD              : out std_logic;
          LOAD_OUT_LCD              : out std_logic;
          DATA_OUT_LCD              : out std_logic );
    END COMPONENT;

SIGNAL  buf_read_data               : std_logic;
SIGNAL buf_address_read             : std_logic_vector ( 4 downto 0 );
SIGNAL buf_data_output              : std_logic_vector ( 7 downto 0 );
```

```
BEGIN
u_inc_dec_ctrl_buff_all : include_pager_decoder_ctrl_include_buffer_all PORT MAP
                          (RESET                => reset,
                          CLK_NRZ_IN            => clk_nrz_in,
                          CLK                   => clk,
                          CLK_READ              => clk,
                          NRZ_IN                => nrz_in,
                          READ_DATA             => buf_read_data,
                          ADDRESS_READ          => buf_address_read,
                          BUZZER_OUT            => buzzer_out,
                          LED                   => led,
                          OUT_BIT31             => out_bit31,
                          DATA_OUTPUT           => buf_data_output);

u_inc_data_lcd_grap_dis : include_data_to_lcd_graphic_dis PORT MAP
                          (RESET                => reset,
                          CLK                   => clk,
                          START_IN              => start_in,
                          DATA_IN               => buf_data_output,
                          READ_RAM_OUT          => buf_read_data,
                          ADDRESS_RAM_OUT       => buf_address_read,
                          OUT_8CLK_LCD          => out_8clk_lcd,
                          LOAD_OUT_LCD          => load_out_lcd,
                          DATA_OUT_LCD          => data_out_lcd );
END Behavioral;
```

SERIAL TO PARALLEL.VHD

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;

ENTITY serial_parallel_shift_decoder IS
PORT ( RESET          : in std_logic;
       CLK            : in std_logic;
       NRZ_IN         : in std_logic;
       FRAME_DELAY    : in std_logic;
       DATA_OUT       : out std_logic_vector( 31 downto 0 ) );
END serial_parallel_shift_decoder;

ARCHITECTURE arc_serial_parallel_shift_decoder of serial_parallel_shift_decoder IS
       SIGNAL reg : std_logic_vector ( 31 downto 0 );
BEGIN
       PROCESS (reset,clk,nrz_in,reg,frame_delay)
       BEGIN
                    if ( reset = '1' ) then
                           reg <= ( others => '0' );
                    elsif ( clk'event and clk ='1' ) then
                           if ( frame_delay = '1' ) then
                                  reg <= reg(30 downto 0)& nrz_in;
                           else
                                  reg <= ( others => '0' );
                           end if;
                    end if;
       END PROCESS;
                           data_out <= reg;
END arc_serial_parallel_shift_decoder;
```

RAM.VHD

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;
USE ieee.std_logic_arith.all;


ENTITY ram_decoder IS
        PORT ( RESET            : in std_logic;
               CLK_NRZ_IN       : in std_logic;
               CLK_READ         : in std_logic;
               ADDRESS          : in std_logic_vector ( 3 downto 0 );
               DATA             : in std_logic_vector ( 31 downto 0 );
               WRTIE            : in std_logic;
               READ             : in std_logic;
               DATA_OUTPUT      : out std_logic_vector ( 31 downto 0 ) );
END ram_decoder;
ARCHITECTURE arc_ram_decoder of ram_decoder IS
   TYPE  memory_array_type is array ( 0 to ( 2**(4)-1 ) ) of std_logic_vector ( 31 downto 0 );
   SIGNAL  memory_array : memory_array_type;
BEGIN
u1_write_ram:  PORCESS (reset,clk_nrz_in,write,data,address )
                    variable temp_var : integer range 0 to 16;
               BEGIN
                    if ( reset = '1' ) then
                         memory_array <= (others => ( others => '0' ) );
                    elsif ( clk_nrz_in = '1' and clk_nrz_in 'event ) then
                         if ( write = '1' ) then
                              temp_var := conv_integer ( address );
                              memory_array ( temp_var ) <= data;
                         end if;
                    end if;
               END PROCESS;
```

```
u2_read_data    :  PROCESS ( read,clk_read,reset,memory_array,address )
                           variable temp_var : integer range 0 to 16;
                   BEGIN
                        if ( reset = '1' ) then
                                data_output <= ( others => '0' );
                        elsif ( clk_read = '1' and clk_read 'event ) then
                                if ( read = '1' ) then
                                        temp_var := conv_integer ( address );
                                        data_output <= memory_array ( temp_var );
                                else
                                        temp_var := 0;
                                end if;
                        end if;
                    END PROCESS;
END arc_ram_decoder;
```

ภาคผนวก ง

บทความวิจัยที่ได้รับการพิจารณาตีพิมพ์

การพัฒนาวิทยุติดตามตัวในระบบ POCSAG โดยใช้เอฟพีจีเอ

On the Implementation of Pocsag Paging using FPGA

อดุลย์ ขันติชนะกุล เฉลิมภัณฑ์ ฟองสมุทร ศ.ดร.วัลลภ สุระกำพลธร
คณะวิศวกรรมศาสตร์ และ สำนักวิจัยการสื่อสารและเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง กรุงเทพมหานคร 10520
โทร (02-3264224) (01-3831797) E-mail: Adul_kan@yahoo.com

บทคัดย่อ

บทความนี้เสนอการพัฒนาวิทยุติดตามตัวในระบบ Pocsag ซึ่งสามารถแสดงข่าวสารข้อมูลได้ทั้งในรูปแบบตัวอักษรภาษาไทยและแบบภาษาอังกฤษ โดยระบบจะประกอบไปด้วย ภาครับสัญญาณวิทยุย่านความถี่สูง และภาคการทำงานทางด้านดิจิตอล ซึ่งได้แก่ ภาคถอดรหัสเพจเจอร์ ภาคถอดรหัสหมายเลขเรียกขาน และภาคแสดงผล โดยการออกแบบอยู่ในชิฟวงจรรวม ซึ่งแนวทางการพัฒนาใช้ภาษาวีเอชดีแอลในการบรรยายพฤติกรรมของฮาร์ดแวร์ที่ทำการออกแบบแล้วโปรแกรมลงในชิฟเอฟพีจีเอ

คำสำคัญ : โดยออกแบบเป็นชิฟวงจรรวม,วีเอชดีแอล,เอฟพีจีเอ

Abstract

This paper presents the development of a Pocsag pager which has the capability of displaying text both in Thai and English. The pager system consists of the VHF receiver part and the digital part, Which comprises pager decoder, address codeword decoder and LCD graphic display, is designed using VHDL to describe the behavior of the designed hardware, and programmed onto the FPGA chip.

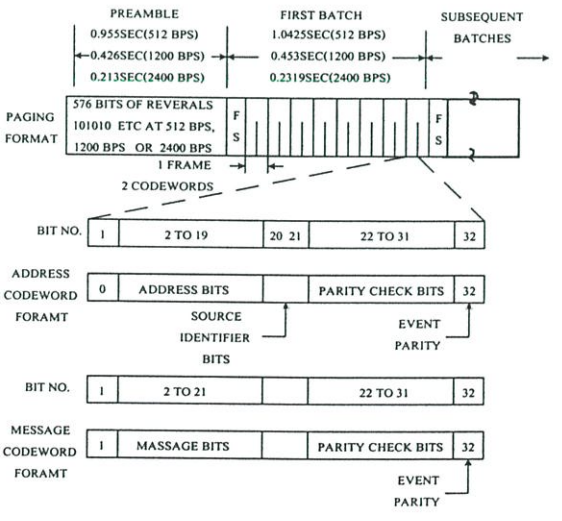
Keywords : VHDL, FPGA ,POCSAG

1. คำนำ

ในปัจจุบันการออกแบบชิฟวงจรรวมทางด้านวงจรดิจิตอลโดยใช้ภาษาวีเอชดีแอลในการอธิบายการทำงานของวงจรจะมีประสิทธิภาพมากยิ่งขึ้นทั้งในด้านขนาดของวงจรที่เล็กลงและมีการทำงานที่เร็วขึ้น ซึ่งบทความนี้จะเสนอการพัฒนาวิทยุติดตามตัวในระบบ POCSAG ซึ่งมีอัตราในการส่งข้อมูล 1200 บิทต่อวินาที โดยจุดประสงค์ก็เพื่อพัฒนาเป็นชิฟวงจรรวมในภาคการทำงานทางด้านวงจรดิจิตอลของวิทยุติดตามตัวเพื่อสามารถที่จะนำไปประยุกต์ใช้งานกับระบบอื่นๆให้มีความหลากหลายในการใช้งานมากยิ่งขึ้น เช่นการสั่งงานผ่านวิทยุติดตามตัวในการป้องกันความปลอดภัยจากการโจรกรรมรถหรือการควบคุมการสั่งงานอุปกรณ์ไฟฟ้าภายในบ้านเป็นต้น ซึ่งข้อดีของระบบสื่อสารแบบ เพจเจอร์ที่สามารถส่งข้อมูลได้ทั่วประเทศและมีความถูกต้องในการรับส่งข้อมูลตลอดจนการใช้งานของการให้บริการก็ยังมีอยู่อย่างต่อเนื่องอยู่ในหลายๆบริษัท จึงสามารถนำไปประยุกต์ใช้งานได้หลายรูปแบบโดยไม่ต้องยึดติดกับเครื่องรับวิทยุติดตามตัวที่สั่งเข้าจากต่างประเทศจึงมีแนวความคิดที่จะทำการออกแบบขึ้นมาเพื่อพัฒนาเทคโนโลยีภายในประเทศ

2. รูปแบบรหัสของเพจเจอร์

รหัสเพจเจอร์ที่ใช้กันอยู่ในประเทศปัจจุบันจะมีมาตรฐานในการสื่อสารข้อมูลที่แตกต่างกัน ซึ่งรหัสเพจเจอร์แบบ POCSAG (Post Office Code Standard Advisory Group)[1] ก็เป็นมาตรฐานหนึ่งที่มีรูปแบบการส่งสัญญาณแบบอะซิงโครไนซ์โดยรูปแบบของสัญญาณจะประกอบไปด้วย 2 ส่วน คือส่วนแรกเป็นพรีแอมเบิ้ล (Preamble) และตามด้วยรหัสคำที่เป็นส่วนของข้อมูลหรือหมายเลขเรียกขานซึ่งจะเริ่มต้นเฟรมด้วยรหัสซิงโครไนซ์ หรือเฟรมซิงโครไนซ์ (Frame Synchronize) โดย รายละเอียดในส่วนต่างๆนั้นได้แสดงไว้รูปที่ 1



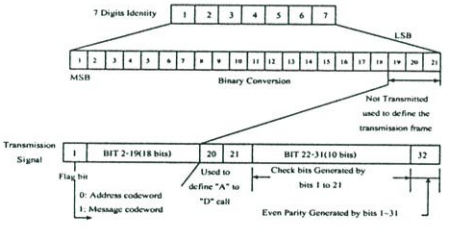
รูปที่ 1 รูปแบบรหัสเพจเจอร์แบบ POCSAG

EL

2.1 พรีแอมเบิ้ล (Preamble) จะมีขนาด 576 บิท จะทำการส่ง "0" และ "1" สลับกันไป ใช้สำหรับกำหนดจุดเริ่มต้นการทำงานและการกู้สัญญาณนาฬิกาที่ภาครับ

2.2 รหัสคำ(Code Word) รหัสคำที่บรรจุแต่ละแบตช์จะเริ่มต้นรหัสคำด้วยรหัสคำซิงโครไนซ์ ซึ่งเป็นรหัสเฉพาะที่มีขนาด 32 บิท และตามด้วยเฟรมของรหัสคำอีก 8 เฟรม โดยแต่ละเฟรมจะมีขนาด 64 บิท หรือ 2 รหัสคำซึ่ง แบ่งรหัสคำได้เป็น

2.2.1 รหัสคำหมายเลขเรียกขาน (Address Codewords) รหัสคำหมายเลขเรียกขานจะมีบิทแรกเป็น "0" เสมอ โดยบิทที่ 2 ถึง 19 เป็นหมายเลขเรียกขาน ซึ่งสร้างขึ้นจากรหัสประจำเครื่อง (Radio Identity Code : RIC) โดยจะมีขนาด 7 หลัก แล้วนำเลข 7 หลักนี้มาแปลงเป็นเลขฐานสองจำนวน 21 บิท ซึ่ง 18 บิท ที่มีความสำคัญมากจะถูกนำมาใช้และส่งออกไป ในส่วนของอีก 3 บิท ที่มีความสำคัญน้อยจะไม่ส่งออกไป แต่จะใช้เป็นตัวกำหนดหมายเลขของเฟรม(0–7)ว่ารหัสเรียกขานถูกส่งไปเฟรมใด ส่วนบิทที่ 20 และ 21 ใช้สำหรับเลือกกลุ่มของหมายเลขเรียกขานซึ่งมีอยู่ 4 กลุ่ม ส่วนบิทที่ 22 ถึง 31 จะเป็นบิทที่ใช้สำหรับตรวจสอบข้อมูลและแก้ไขการผิดล่วงหน้าโดยใช้การตรวจสอบแบบ BCH สำหรับการใช้บิทตรวจสอบ 10 บิทสำหรับข่าวสาร 21 บิททำให้ตรวจสอบการผิดของบิทได้ 4 บิท และแก้ไขการผิดของบิทได้ 2 บิท สำหรับบิทสุดท้ายเป็นบิทพาริตี้ ซึ่งเป็นแบบพาริตี้คู่ (Even Parity) รูปแบบรหัสคำหมายเลขเรียกขานแสดงได้ดังรูปที่ 2



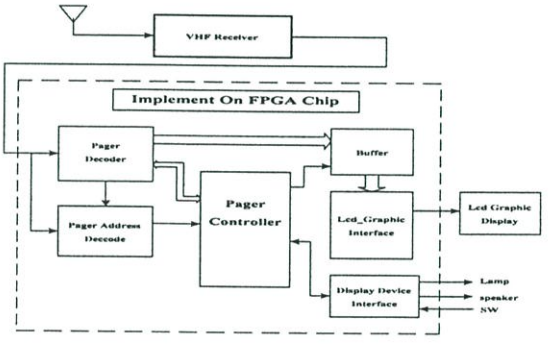
รูปที่ 2 รูปแบบรหัสหมายเลขเรียกขาน

2.3 รหัสคำข่าวสาร (Message Codeword) รหัสคำข่าวสารจะเริ่มต้นบิทแรกเป็น 1 โดยข่าวสารข้อมูลซึ่งบรรจุอยู่ในบิทที่ 2 ถึง บิทที่ 21 จะมีขนาด 8 บิท ต่อ 1 ตัวอักษรโดยมี รูปแบบตัวอักษรภาษาอังกฤษเป็นไปตามมาตรฐาน ASCII Code และ อักษรภาษาไทย อยู่ในรูปแบบตัวอักษรตามมาตรฐานของ สมอ. จากการที่แต่ละตัวอักษรมีขนาด 8 บิททำให้ใน 1 รหัสคำบรรจุอักษรได้ 2 ตัวอักษรกับอีก 4 บิทซึ่งหมายความว่าบิทของตัวอักษรตัวท้ายจะต้องถูกเลื่อนไปอยู่ในรหัสคำอื่น

2.4 รหัสคำว่าง (Idle Codeword) รหัสคำว่างเป็นรหัสใช้สำหรับใส่รหัสลงในแบตช์ให้ครบ 8 เฟรม ในกรณีที่ข่าวสารส่งมาสิ้นสุดก่อนที่จะหมดแบตช์โดยรูปแบบของคำคือ " 01111010100010011100000110010111 "

3. การออกแบบฮาร์ดแวร์

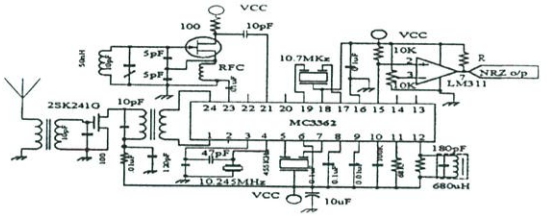
สำหรับแนวทางการพัฒนาวิทยุติดตามตัวในระบบ POCSAG [2] จะประกอบไปด้วยส่วนต่างๆคือภาครับสัญญาณวิทยุ (Receiver) ทำหน้าที่รับสัญญาณวิทยุที่ส่งข้อมูลเพจเจอร์ในย่านความถี่ VHF แล้วแปลงสัญญาณวิทยุเป็นสัญญาณดิจิตอลเบสแบนด์ ส่วนภาคถอดรหัสเพจเจอร์ (Pager Decoder) จะถอดรหัสดิจิตอลเบสแบนด์เป็นรูปแบบรหัส POCSAG ให้เก็บในหน่วยความจำ (Static RAM : SRAM) ส่วนถอดรหัสสัญญาณเรียกขาน (Address Codeword Decoder) สามารถโปรแกรมค่า Address ใดๆก็ได้เมื่อมีสัญญาณเพจเจอร์เข้ามาส่วนการถอดรหัสสัญญาณเรียกขานจะเปรียบเทียบค่ารหัสของเครื่องที่ตั้งไว้กับสัญญาณที่รับเข้ามาโดย ถ้าหากสัญญาณเรียกขานรับมาตรงกับค่าที่ตั้งไว้ก็จะเกิดสัญญาณอินเตอร์รัพท์ (Interrupt) เพื่อให้ภาคเชื่อมต่ออุปกรณ์แสดงผล (Display Device - Interface) จะแสดงผลเมื่อเริ่มรับข่าวสารข้อมูลเข้ามาส่วนภาคแสดงข้อมูลจะใช้กราฟฟิกแอลซีดี (Lcd Graphic Display) เป็นตัวแสดงข่าวสารข้อมูลที่รับเข้ามา แสดงดังรูปที่ 3



รูปที่ 3บล็อกไดอะแกรมเครื่องรับวิทยุติดตามตัวระบบ POCSAG

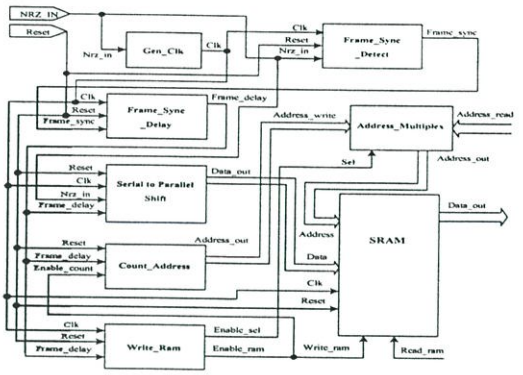
3.1 ภาครับสัญญาณวิทยุ (VHF Receiver)[3] การออกแบบภาครับสัญญาณวิทยุได้ออกแบบให้มีการรับสัญญาณเพจเจอร์ในย่านความถี่ 279.55 MHz ซึ่งเป็นความถี่ให้บริการในโหมดตัวอักษรโดยตัวอักษรจะถูกมอดดูเลทแบบ Frequency Shift Keying โดยมีความถี่เบี่ยงเบนเท่ากับ $\pm$ 4.5 KHz ด้วยขนาดของบิทเรทข้อมูลเท่ากับ 1200 บิทต่อวินาที โดยการออกแบบเลือกใช้วงจรรวมเบอร์ MC3362 ของบริษัท Motola ซึ่งเป็นวงจรที่ทำหน้าที่ถอดสัญญาณวิทยุ FM แบบแบนด์แคบมีเอาท์พุทของภาคดีเทคสัญญาณ FSKโดยภายในจะเป็นภาครับแบบดูออลคอนเวอร์ชั่น (Dual Conversion) ใช้ความถี่ IF ภาคที่ 1 เท่ากับ 10.7 MHz และภาคที่ 2 เท่ากับ 455KHz โดยมีวงจรออสซิเลเตอร์ภายในสำหรับผลิตความถี่เพื่อสร้างสัญญาณ IF ภาคที่ 1 สามารถผลิตความถี่ได้ถึง 200 MHz และภาคผลิตความถี่สำหรับสร้างสัญญาณ IF ภาคที่ 2 ซึ่งผลิตความถี่ขนาด 10.245 MHz โดยควบคุมความถี่ด้วยคริสตอลส่วนภาคดีเทคสัญญาณเป็นวงจรแบบควอดคราเจอร์ดิสคริบมิเนเตอร์ (Quadrature Discriminator)

โดยวงจรรวม MC3362 ต้องการสัญญาณอินพุทขนาด 0.7 μV สำหรับทำให้ค่า S+N / N มีขนาด 20 dB หรือทำให้การดีเทคข้อมูลมีความน่าเชื่อถือ ดังนั้นที่ภาคอินพุทจึงมีการเพิ่มวงจรขยายสัญญาณ RF โดยใช้ FET เบอร์ 2SK241G ซึ่งมีอัตราการขยาย 10 dB ซึ่งก็เพียงพอที่จะขยายสัญญาณ RF ที่ปลายสายอากาศให้มีความแรงที่จะป้อนเข้าอินพุทของวงจรรวม MC3362 สำหรับวงจรผลิตความถี่เพื่อมาผสมให้ได้ความถี่ IF ภาคแรก นั้นเนื่องจากในที่นี้วงจรออกแบบรับสัญญาณความถี่ขนาด 279.8MHz จึงใช้วงจรผลิตความถี่ภายนอกซึ่งใช้วงจรผลิตความถี่แบบโคพิท (Colpit Osicllator) โดยใช้เฟทเบอร์ 2SK125 สำหรับเซรามิกฟิลเตอร์จะใช้สำหรับกรองสัญญาณ IF เลือกใช้ค่าโดยทั่วไป ซึ่งมีขนาดแบนด์วิดท์ประมาณ 15 KHz ซึ่งมากพอที่จะให้สัญญาณ FSK ของเพจเจอร์ที่มีขนาดความถี่เบี่ยงเบน $\pm$ 4.5 KHz ผ่านได้ สำหรับเอาท์พุทภาคสุดท้ายของวงจรควอดคราเจอร์ดิสคริมิเนเตอร์ต่อเข้ากับวงจรรวม LM311 ทำหน้าที่เป็นวงจรเปรียบเทียบเพื่อทำให้สัญญาณเอาท์พุท NRZ ที่มอดดูเลทได้ระดับสัญญาณดิจิตอลโดยวงจรภาครับสัญญาณวิทยุแสดงได้ดังรูปที่ 4



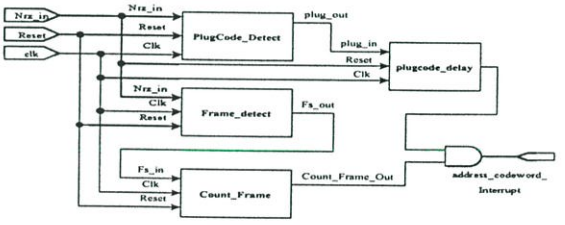
รูปที่ 4 วงจรภาครับสัญญาณวิทยุ

3.2 ภาคถอดรหัสเพจเจอร์ (Pager Decoder) โดยบล็อกไดอะแกรมแสดงดังในรูปที่ 5 เป็นการนำสัญญาณเอาท์พุทมาจากภาครับวิทยุซึ่งเป็นสัญญาณเบสแบนด์แบบ NRZ ให้เก็บอยู่ในหน่วยความจำ (SRAM) โดยเก็บข้อมูลครั้งละ 1 รหัสคำ(32 บิท) และจะถูกนำมาแสดงผลโดย กราฟฟิกแอลซีดี (LCD Graphic Display) ซึ่งภาคถอดรหัสเพจเจอร์จะประกอบ ด้วย วงจรกู้สัญญาณนาฬิกา (Clk Recovery) ทำหน้าสร้างสัญญาณนาฬิกาจากสัญญาณเบสแบนด์ NRZ ที่รับเข้าเพื่อจ่ายให้กับภาคถอดรหัสข้อมูลทั้งหมดโดยที่วงจรตรวจสอบเฟรมซิงส์โครไนส์ (Frame Sync Detect) และวงจรหน่วงเวลาเฟรมซิงส์ (Frame Sync Delay) จะทำการตรวจสอบรหัสคำซิงโครไนซ์และหน่วงเวลาในการเก็บข้อมูลเข้าไปในหน่วยความจำโดยมีวงจรชิฟรีจิสเตอร์แบบอนุกรมเป็นขนาน (Serial To Parallel Register) ทำหน้าที่ในการจัดเรียงข้อมูลเก็บในหน่วยความจำ (Static RAM : SRAM) วงจรนับแอดเดรส (Count Address) และวงจรสร้างสัญญาณเขียนแรม (Write-Ram) เป็นสัญญาณที่ควบคุมการกำหนดแอดเดรสและสัญญาณการเขียนข้อมูลลงในหน่วยความจำ ส่วนวงจรเลือกแอดเดรส (Address Multiplex) เป็นตัวควบคุมการเขียนและการอ่านข้อมูลทั้งหมดจากหน่วยความจำ



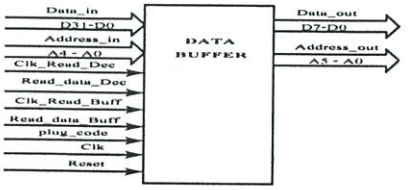
รูปที่ 5 บล็อกไดอะแกรมภาคถอดรหัสเพจเจอร์

3.3 ภาคถอดรหัสหมายเลขเรียกขาน (Address Codeword Decoder) [2] เป็นภาคที่เปรียบเทียบหมายเลขเรียกขานที่รับได้กับหมายเลขเครื่องที่ถูกโปรแกรมไว้บล็อกไดอะแกรมภาคถอดรหัสหมายเลขเรียกขานแสดงดังรูปที่ 6 ซึ่งประกอบไปด้วยวงจรตรวจสอบหมายเลขเรียกขาน (Address Codeword Detect) ทำหน้าที่เปรียบเทียบหมายเลขเรียกขานและถูกหน่วงเวลาด้วยวงจรหน่วงเวลา (Address Codeword Delay) เพื่อทำการแอนกับวงจรนับเฟรมโดยที่วงจรเฟรมซิงโครไนซ์ (Frame Synchronize) เป็นจุดเริ่มต้นการนับเฟรม สำหรับวงจรเลือกเฟรม (Count Frame) หมายเลขเรียกขานจะส่งมาในเฟรมใดขึ้นอยู่กับ 3 บิทหลัง เมื่อวงจรนับเฟรมนับค่าได้ตรงกับเฟรมที่ตั้งไว้ก็จะให้เอาท์พุทเป็น 1 ไปแอนกับเอาท์พุทของวงจรหน่วงเวลาหมายเลขเรียกขานเพื่อให้เกิดสัญญาณอินเตอร์รัพท์ เพื่อเป็นตัวบ่งบอกถึงจุดเริ่มต้นของการรับข่าวสารข้อมูล



รูปที่ 6 บล็อกไดอะแกรมภาคถอดรหัสหมายเลขเรียกขาน

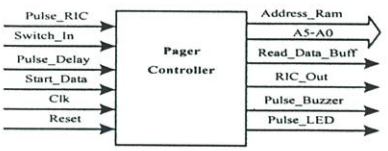
3.4 ภาคพักข้อมูลชั่วคราว (Data Buffer) ทำหน้าที่จัดเรียงข้อมูลที่ได้จากหน่วยความจำเพื่อนำไปแสดงผลข่าวสารข้อมูลที่ได้รับแสดงดังรูปที่ 7



รูปที่ 7 บล็อกไดอะแกรมภาคพักข้อมูล

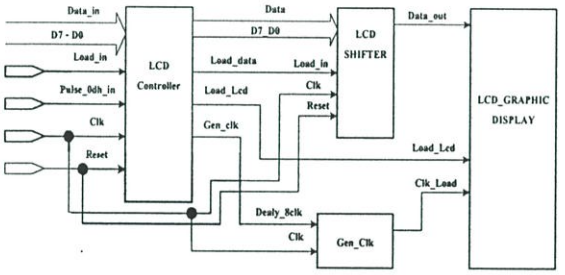
EL

3.5 ภาคควบคุมระบบเพจเจอร์ (Pager Controller) ทำหน้าที่ควบคุมการการเขียน-อ่านข้อมูลและสัญญาณควบคุมต่างๆในระบบดังรูปที่ 8



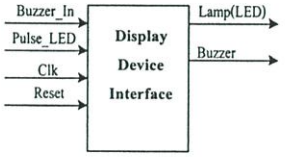
รูปที่ 8 บล็อกไดอะแกรมภาคควบคุมระบบ

3.6 ภาคแสดงผล (Display) เป็นภาคแสดงข่าวสารข้อมูลโดยใช้อุปกรณ์กราฟฟิกแอลซีดี [6] เป็นตัวแสดงผลข้อมูลที่รับเข้ามาโดยสามารถแสดงผลได้ทั้งภาษาไทยและภาษาอังกฤษตามมาตรฐานสมอ.โดยมีขนาด 128×64 DOT สามารถแสดงตัวอักษรไทย และภาษาอังกฤษได้ 16×4 บรรทัด ซึ่งลักษณะการส่งข้อมูลจะทำการส่งข้อมูลแบบ 3 Bit Serial โดยมีภาคควบคุมการทำงานแสดงดังรูปที่ 9



รูปที่ 9 บล็อกไดอะแกรมภาคเชื่อมต่อกราฟฟิกแอลซีดี

3.7 ภาคเชื่อมต่ออุปกรณ์แสดงผล (Display Device Interface) ทำหน้าที่แสดงผลการเรียกเพจเจอร์โดยจะแสดงไฟและส่งเสียงเตือนเมื่อมีการเรียกเพจเจอร์ แสดงดังรูปที่ 10

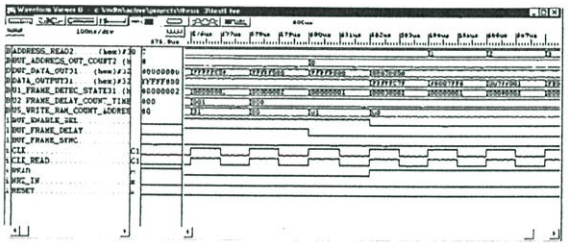


รูปที่ 10 บล็อกไดอะแกรมภาคเชื่อมต่ออุปกรณ์แสดงผล

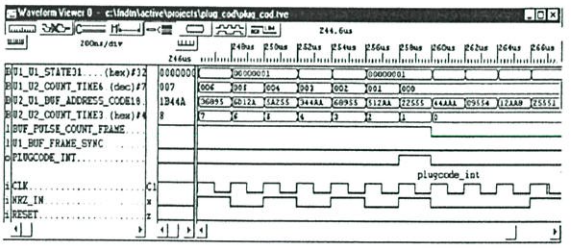
4. ผลการทดสอบ

ศึกษาการทำงานทั้งหมดของวงจรจากนั้นสร้างเป็นโมดูลของวงจร โดยใช้ภาษาวีเอชดีแอล (VHDL) ในการบรรยายพฤติกรรมของวงจร (Behavioral) [4] ในแต่ละส่วนการทำงานของวงจรที่แบ่งไว้และทำการสังเคราะห์ (Synthesis Tools) โมดูลของวงจรต่างๆเพื่อให้ได้ฮาร์ดแวร์ที่เหมาะตลอดจนความเร็วในการทำงานของวงจรเพื่อนำไปทดสอบกับชิฟเอฟพีจีเอต่อไปส่วนโปรแกรมการออกแบบวงจรจะพัฒนาด้วยภาษาวีเอชดีแอลแล้วนำมาสังเคราะห์วงจรตลอดจนการโปรแกรมลงชิฟเอฟพีจีเอโดยใช้โปรแกรม Xilinx Foundation F2.1i และชิฟเอฟพีจีเอ (FPGA) ของบริษัท Xilinx

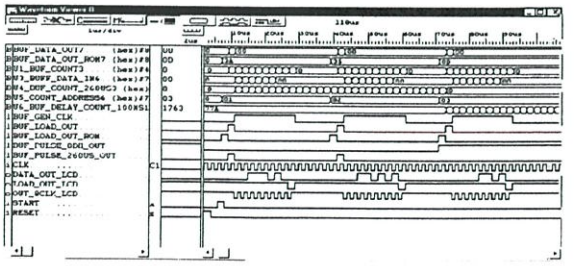
4.1 VHDL Simulation ทำการจำลองการทำงาน (Simulation) ของวงจรทั้งหมดที่ได้จากการใช้ภาษาวีเอชดีแอลอธิบายพฤติกรรมของวงจรโดยกำหนดรูปแบบสัญญาณอินพุทตามมาตรฐานรูปแบบของรหัส POCSAG แล้วป้อนเข้าไปในระบบเพื่อดูผลจำลองการทำงานของระบบโดยภาคถอดรหัสสัญญาณจะทำการเก็บข้อมูลและอ่านข้อมูลขนาด 32 บิทจาก SRAM ซึ่งแสดงผลดังรูปที่ 11 ส่วนภาคถอดรหัสหมายเลขเรียกขานทำหน้าที่ ตรวจสอบหมายเลขเรียกหมายโดยจะกำเนิดสัญญาณพัลซ์เพื่อแจ้งให้ทราบถึงการรับส่งข้อมูลแสดงดังรูปที่ 12 และภาคแสดงผลกราฟฟิกแอลซีดี จะรับข้อมูลเข้ามาขนาด 8 บิทและทำการส่งข้อมูลเป็นแบบ 3 Bit Serial ซึ่งแสดงดังรูปที่ 13



รูปที่ 11 เขียน-อ่านข้อมูลจาก SRAM ในภาคถอดรหัสสัญญาณ

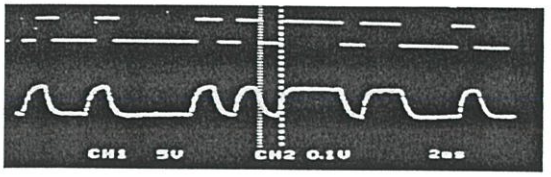


รูปที่ 12 สัญญาณรหัสหมายเลขเรียกขาน



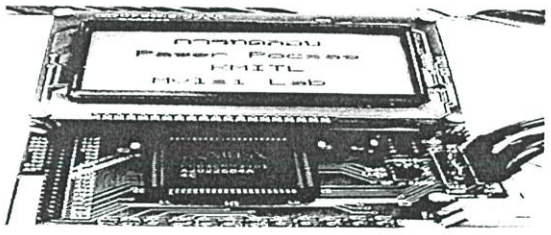
รูปที่ 13 ข่าวสารข้อมูลภาคแสดงผลกราฟฟิกแอลซีดี

4.2 ทดสอบภาครับสัญญาณวิทยุ (VHF Receiver) เป็นการทดสอบการทำงานโดยทำการวัดสัญญาณเอาท์พุทของ LM311 แสดงข่าวสารข้อมูลที่รับเข้ามาแสดงในรูปที่ 14



รูปที่ 14 สัญญาณข่าวสารข้อมูลในภาค VHF Receiver

4.3 ทดสอบบนชิฟเอฟพีจีเอ เป็นการทดสอบฮาร์ดแวร์หลังจากได้ผลจากการจำลองการทำงานแล้วสามารถทำการโปรแกรมข้อมูลทางลอจิกที่ได้จากการสังเคราะห์ ลงในชิฟเอฟพีจีเอของบริษัทXilinxโดยเลือกใช้ชิฟในตระกูล XC4000E เบอร์ 4010EPC84 ในการโปรแกรมผ่านสายดาวน์โหลด X-Checker ของบริษัท Xilinx เพื่อทดสอบการทำงานโดยการรับข่าวสารข้อมูลจริงมาจากภาครับสัญญาณวิทยุที่ส่งมาแสดงดังรูปที่ 16



รูปที่ 16 เครื่องต้นแบบวิทยุติดตามตัว

5. สรุปผลการทดลอง

จากการทดลองพบว่าสามารถสร้างเครื่องต้นแบบเพื่อใช้งานสำหรับการรับสัญญาณเพจเจอร์ในระบบ POCSAG ซึ่งสามารถแสดงผลเป็นภาษาไทยและภาษาอังกฤษได้ข้อมูลที่ถูกต้อง ซึ่งการพัฒนาการออกแบบโดยใช้ภาษาวีเอชดีแอลแล้วโปรแกรมลงชิฟเอฟพีจีเอสามารถกำหนดฟังก์ชันการทำงานตามความต้องการของผู้ออกแบบซึ่งมีความสะดวกและรวดเร็วทำให้การออกแบบวงจรดิจิตอลสามารถทำได้รวดเร็วยิ่งขึ้น เมื่อเทียบกับการวิธีที่ผ่านมา[2] โดยแนวทางที่จะพัฒนาวิทยุติดตามตัวต่อไปคือการนำคอร์(Core) ที่ได้จากการสังเคราะห์ไปออกแบบเป็น ASIC ต่อไป

6. กิตติกรรมประกาศ

ขอขอบคุณบริษัทฟิลิปส์(ประเทศไทย)จำกัดที่ช่วยสนับสนุนไอซี Receiver - Decoder Pager องค์การโทรศัพท์แห่งประเทศไทย (ศูนย์เวิลด์เพจ) และJICA ที่ให้การสนับสนุนงานวิจัยในครั้งนี้

เอกสารอ้างอิง

[1] British Telecom " A standard Code for Radio Paging report of Post Office Code standard Advisory Group (POCSAG) " june 1978

[2] อุดมพร สนันทชัยกุล,กฤคากร กล่อมการ,คลชัย สุขเจริญผล, วิชัย สุรพัฒน์ "การพัฒนาเพจเจอร์การ์ดบนไมโครคอมพิวเตอร์ " การประชุมวิชาการทางวิศวกรรมไฟฟ้า ครั้งที่ 18

[3] "VHF Narrowband FM Receiver Design Using The MC3362 and The MC3362 Dual Conversion Receiver AN980 ",CMOS Application-Specific ICS,Motorola INC,

[4] Stefan Sjoholm ABB Industrial System,Sweden and Lennart Lindh Malardalens University ,Sweden"VHDL for Designes",Prentice-Hall

[5] Xilinx Corporation " Foundation Serial 2.1i Tutorial",

[6] TICD-164 Version1.0 Thai Lcd Module,Sila Research Co., Ltd.

อดุลย์ ขันติชนะกุล สำเร็จการศึกษาในระดับปริญญาตรี จากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปี พ.ศ 2537 คณะวิศวกรรมศาสตร์ ภาคเทคนิคอุตสาหกรรม และกำลังศึกษาอยู่ในระดับปริญญาโท สจล.

เฉลิมภัณฑ์ ฟองสมุทร สำเร็จการศึกษาในระดับปริญญาตรีและปริญญาโท วิศวกรรมศาสตร์บัณฑิต สาขาอิเล็กทรอนิกส์ และวิศวกรรมศาสตร์มหาบัณฑิต สาขาวิศวกรรมไฟฟ้า จากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ในปีพ.ศ.2538และ พ.ศ.2541 ตามลำดับ ซึ่งสนใจงานวิจัยทางด้าน digital signal processing และ instrumentation.

วัลลภ สุระกำพลธร สำเร็จการศึกษาในระดับปริญญาตรีและระดับปริญญาโท วิศวกรรมศาสตร์บัณฑิต และวิศวกรรมศาสตร์มหาบัณฑิตสาขาวิศวกรรมไฟฟ้า จากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ในปีพ.ศ. 2519 และ พ.ศ. 2521 ตามลำดับ และระดับปริญญาเอก (Ph.D.) ในสาขาอิเล็กทรอนิกส์ จาก the University of Kent at Canterbury, Kent, ประเทศสหราชอาณาจักร ในปี พ.ศ. 2526

ปัจจุบันเป็นศาสตราจารย์ ในภาควิชาอิเล็กทรอนิกส์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ซึ่งทำงานวิจัยเกี่ยวกับ analog and digital integrated circuit designs, real-time application of PC computer and microprocessors, digital signal processing, electronic instrumentation, และ VLSI signal processing.

EL

ประวัติผู้เขียน

นาย อดุลย์ ขันติชนะกุล เกิดเมื่อวันที่ 15 เมษายน พ.ศ.2513 ที่จังหวัดสุราษฎร์ธานี สำเร็จการศึกษา ภาคเทคนิคอุตสาหกรรม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยี พระจอมเกล้าเจ้าคุณทหารลาดกระบัง ในปีการศึกษา 2537 และเข้าศึกษาต่อในระดับปริญญาโท หลักสูตร วิศวกรรมศาสตร์มหาบัณฑิต สาขา วิศวกรรมอิเล็กทรอนิกส์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง เมื่อปี พ.ศ. 2543

ปัจจุบันทำงานที่ ทศท.คอปอเรชั่นจำกัด มหาชน ตำแหน่ง วิศวกร 5 ฝ่ายวิจัยและพัฒนาเทคนิค