สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

การศึกษาระบบและสมรรถภาพของ DS-CDMA ในช่องสัญญาณไร้สาย

System and performance analysis of DS-CDMA in wireless channel

โดย

นายปฤชน ศิริมงคลลาวัลย์

นายพีรพงศ์ นิติเขตต์ปรีชา

นายพีระพล สุวรรณโชติ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เลขหมู่.....
เลขทะเบียน 61942
วัน,เดือน,ปี 25 ก.ค. 2549

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

ภาควิชา
วิศวกรรมโทรคมนาคม

b.....
i.....

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การศึกษาระบบและสมรรถภาพของ DS-CDMA ในช่องสัญญาณไร้สาย

System and performance analysis of DS-CDMA in wireless channel

โดย

นายปฤชน ศิริมงคลลาวัลย์ 44010306

นายพีรพงศ์ นิติเขตต์ปรีชา 44010340

นายพีระพล สุวรรณโชติ 44010343

อาจารย์ที่ปรึกษา

ดร. พรชัย ทรัพย์นิธิ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การศึกษาระบบและสมรรถภาพของ DS-CDMA ในช่องสัญญาณไร้สาย

System and performance analysis of DS-CDMA in wireless channel

โดย นายปุณชน ศิริมงคลลาวัลย์ 44010306
นายพีรพงศ์ นิติเขตต์ปรีชา 44010340
นายพีระพล สุวรรณโชติ 44010343

อาจารย์ที่ปรึกษา ดร.พรชัย ทรัพย์นิธิ

บทคัดย่อ

โครงงานนี้นำเสนอการศึกษาระบบและสมรรถภาพของ DS-CDMA ในช่องสัญญาณไร้สาย โดยใช้โปรแกรม MATLAB ศึกษาและจำลองการทำงานของระบบในส่วนของ การเข้ารหัส การถอดรหัส การมอดูเลต และการดีมอดูเลต รวมถึงการออกแบบวงจรดิจิตอลของระบบร่วมกับภาษา VHDL และทดสอบด้วยการทำงานบน FPGA

Abstract

This project is to study the DS-CDMA (Direct Sequence CDMA) system and performance in a wireless channel using MATLAB program to simulate the DS-CDMA system. The digital circuit design is implemented with VHDL (Very High Speed Integrated Circuit Hardware Description Language) then tested on FPGA (Field Programmable Gate Array).

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2547

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง **การศึกษาระบบและสมรรถภาพของ DS-CDMA ในช่องสัญญาณไร้สาย**

System and performance analysis of DS-CDMA in wireless channel

ผู้จัดทำ

1. นายปุณชน ศิริมงคลลาวัลย์ 44010306
2. นายพีรพงศ์ นิติเขตต์ปรีชา 44010340
3. นายพีระพล สุวรรณโชติ 44010343

..... อาจารย์ที่ปรึกษา

(ดร. พรชัย ทรัพย์นิธิ)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

หน้า

บทที่ 1 บทนำ

1.1 ที่มาของปริญญานิพนธ์ 1

1.2 วัตถุประสงค์ของปริญญานิพนธ์ 1

บทที่ 2 ทฤษฎีและหลักการ

2.1 ทฤษฎีและหลักการระบบเข้าถึงหลายทางแบบการแบ่งรหัส 3

2.1.1 ระบบการเข้าช่องสัญญาณ (Multiple access system) 3

2.1.2 การเข้าถึงหลายแบบการแบ่งรหัส (CDMA) 3

2.1.3 การมอดูเลตแบบสเปกตรัมแผ่

(Spread Spectrum Modulation : SS Modulation) 5

2.1.4 การจัดลำดับโดยตรง (Direct Sequence : DS)

- พื้นฐานของ DS-CDMA 5

2.1.5 การสเปรด (Spreading) 9

- รหัสวอล์ช (Walsh Code) 9

- รหัสสเครมบลิง (Scrambling Code) 12

2.1.6 การซิงโครไนซ์ของระบบ 14

2.2 ทฤษฎีและหลักการที่ใช้ในการออกแบบวงจรดิจิตอล 14

บทที่3 การคำนวณและการสร้าง

3.1 การออกแบบและการสร้างด้วยโปรแกรม MATLAB

3.1.1 แบบจำลองเครื่องส่ง (Transmitter model) 18

3.1.2 แบบจำลองช่องสัญญาณ (Channel model) 19

3.1.3 แบบจำลองเครื่องรับ (Receiver model) 20

3.1.4 การวิเคราะห์สมรรถนะของระบบ 21

3.1.5 วิเคราะห์สมรรถภาพของระบบเมื่อมีการเปลี่ยนแปลงของรหัสวอล์ช 21

3.1.6 วิเคราะห์การซิงโครไนซ์ 21

3.2 การออกแบบวงจรและการสร้างด้วยภาษา VHDL 22

3.2.1 ด้านส่ง

3.2.1.1 วงจรหารความถี่ 25

3.2.1.2 วงจรซีเรียลอินพาราแลลเอ้าท์ (Parallel in Serial out) 26

3.2.1.3 วงจรเปลี่ยน 1 บิต เป็น 8 บิต 27

3.2.1.4 วงจรเข้ารหัส 28

3.2.1.5 วงจรแมปปิ้ง (mapping) 29

3.2.1.6 วงจรบวก (ADDER) 30

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.2 ด้านรับ

3.2.2.1 วงจรถอดรหัส 31

3.2.2.2 วงจรบวก 31

3.2.2.3 วงจรตัดสินค่า(DECISION) 32

3.2.2.4 วงจรซีเรียลอินพาราแลล เอ้าท์ 33

3.2.2.5 วงจรหน่วยควบคุม (control unit) 34

3.2.3 ระบบแบบหลายผู้ใช้ผ่านพอร์ทอนุกรม 35

3.2.3.1 วงจรรับข้อมูลจากพอร์ทอนุกรม (RX-SERIAL) 37

3.2.3.2 วงจรอินเทอร์เฟส 1 (interface 1) 37

3.2.3.3 วงจรอินเทอร์เฟส 2 (interface 2) 39

3.2.3.4 วงจรส่งข้อมูลออกพอร์ทอนุกรม 40

3.2.3.5 ส่วนส่งข้อมูล และรับสัญญาณผ่านพอร์ทอนุกรม 44

บทที่ 4 การทดลองและผลการทดลอง

4.1 ผลการทดลองที่ได้จากโปรแกรม MATLAB 46

4.1.1 ผลการทดลองสมรรถภาพของระบบ DS-CDMA 46

4.1.2 ผลการทดลองสมรรถภาพเมื่อขนาดของรหัสเพิ่ม 52

4.1.3 ผลการวิคราะห์ทดลองการซิงโครไนซ์ 53

4.2 ผลการทดลองที่ได้จากโปรแกรม MAX+plus II

4.2.1 ด้านส่ง 53

4.2.1.1.1 วงจรหารความถี่ 53

4.2.1.1.2 วงจรพาราแลลอินซีเรียลเอ้าท์ (Parallel in Serial out) 53

4.2.1.1.3 วงจรเข้ารหัส 54

4.2.1.1.4 วงจรแมปปิ้ง 54

4.2.1.1.5 วงจรบวก 55

4.2.1.2 ด้านรับ

4.2.1.2.1 วงจรถอดรหัส 55

4.2.1.2.2 วงจรบวก 55

4.2.1.2.3 วงจรตัดสินค่า 56

4.2.1.2.4 วงจรซีเรียลอินพาราแลลเอ้าท์ 56

4.2.1.2.5 วงจรหน่วยควบคุม 57

4.2.1.2.6 วงจรรวม 57

4.2.1.2.7 สัญญาณอินพุตจากเครื่องกำเนิดความถี่ 59

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.2 ผลการทดลองระบบแบบหลายผู้ใช้ผ่านพอร์ทอนุกรม
4.2.2.1 วงจรรับข้อมูลจากพอร์ทอนุกรม 62
4.2.2.2 วงจรส่งข้อมูลออกพอร์ทอนุกรม 63
บทที่ 5 บทวิจารณ์และบทสรุป 64
วิจารณ์ผลการทดลอง 65
ภาคผนวก ก. 66
ภาคผนวก ข. 69
ภาคผนวก ค. 70
ภาคผนวก ง. 74
กิตติกรรมประกาศ 88
บรรณานุกรม 89

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

	หน้า
รูปที่ 1.1 บล็อกไดอะแกรมด้านส่งและด้านรับของระบบ DS-CDMA	2
รูปที่ 2.1 บล็อกไดอะแกรมของ ทางด้านส่ง CDMA	6
รูปที่ 2.2 บล็อกไดอะแกรมของ ทางด้านรับCDMA	6
รูปที่ 2.3 ไดอะแกรมอธิบายลักษณะสัญญาณรวมที่มาจากหลายผู้ใช้ที่มาถึงด้านรับ	7
รูปที่ 2.4 อธิบายถึงกระบวนการที่จะได้สัญญาณที่ต้องการ	8
รูปที่ 2.5 4 บิต ชิฟรีจิตเตอร์	12
รูปที่ 2.6 ฟังก์ชันอัตสหสัมพันธ์	13
รูปที่ 2.7 หลักการ Orthogonal Code ที่ใช้ในการออกแบบวงจรดิจิตอล	16
รูปที่ 3.1 แบบจำลองระบบ DS-CDMA ที่ศึกษา	18
รูปที่ 3.2 แบบจำลองภาคส่ง DS-CDMA ที่ศึกษา	19
รูปที่ 3.3 แบบจำลองภาครับ DS-CDMA ที่ศึกษา	20
รูปที่ 3.4 บล็อกไดอะแกรม การซิงโครไนซ์	21
รูปที่ 3.5 บล็อกไดอะแกรมด้านส่งของวงจรดิจิตอลที่ออกแบบ	23
รูปที่ 3.6 บล็อกไดอะแกรมด้านรับของวงจรดิจิตอลที่ออกแบบ	24
รูปที่ 3.7 บล็อกวงจรหารความถี่ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	27
รูปที่ 3.8 บล็อกวงจร PISO ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	28
รูปที่ 3.9 บล็อกวงจรวงจรเปลี่ยน 1 บิต เป็น 8 บิต ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	29
รูปที่ 3.10 บล็อกวงจรเข้ารหัสที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	30
รูปที่ 3.11 บล็อกวงจรmapping ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	31
รูปที่ 3.12 บล็อกวงจร adder ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	32
รูปที่ 3.13 บล็อกวงจรถอดรหัสที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	33
รูปที่ 3.14 บล็อกวงจร adder ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	35
รูปที่ 3.15 บล็อกวงจรตัดสินที่ได้จากการ synthesis จากโปรแกรม MAX+plus	36
รูปที่ 3.16 บล็อกวงจร SIPO ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	37
รูปที่ 3.17 บล็อกวงจร control unit ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	38
รูปที่ 3.18 บล็อกไดอะแกรมด้านส่งของวงจรดิจิตอลที่ออกแบบ	39
รูปที่ 3.19 บล็อกวงจรรับข้อมูลจากพอร์ทอนุกรม ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	38
รูปที่ 3.20 บล็อกวงจร อินเทอร์เฟส 1 ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	39
รูปที่ 3.21 บล็อกวงจร อินเทอร์เฟส 2 ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II	40

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

รูปที่ 3.22 บล็อกวงจรส่งข้อมูลออกพอร์ทอนุกรมที่ได้จากการ synthesis จากโปรแกรม MAX+plus II 44
รูปที่ 4.1 อัตราความผิดพลาดของบิตสัญญาณกับพลังงานสัญญาณต่อสัญญาณรบกวน 46
รูปที่ 4.2 อัตราความผิดพลาดของบิตข้อมูลกับพลังงานสัญญาณต่อสัญญาณรบกวน 47
รูปที่ 4.3 ภาพข้อมูลเริ่มต้นที่1 48
รูปที่ 4.4 ภาพข้อมูลที่1 หลังผ่านระบบ CDMA เมื่อ SNR=1 48
รูปที่ 4.5 ภาพข้อมูลที่1หลังผ่านระบบ CDMA เมื่อ SNR=6 49
รูปที่ 4.6 ภาพข้อมูลที่1หลังผ่านระบบ CDMA เมื่อ SNR=12 49
รูปที่ 4.7 ภาพข้อมูลเริ่มต้นที่ 2 50
รูปที่ 4.8 ภาพข้อมูลที่2หลังผ่านระบบ CDMA เมื่อ SNR=1 50
รูปที่ 4.9 ภาพข้อมูลที่2หลังผ่านระบบ CDMA เมื่อ SNR=6 51
รูปที่ 4.10 ภาพข้อมูลที่2หลังผ่านระบบ CDMA เมื่อ SNR=12 51
รูปที่ 4.11 อัตราความผิดพลาดของบิตข้อมูลกับพลังงานสัญญาณต่อสัญญาณรบกวน 52
รูปที่ 4.12 สัญญาณที่ได้จากวงจรหารความถี่ 53
รูปที่ 4.13 สัญญาณที่ได้จากวงจรพาราแลลอินซีเรียลเอ้าท์ 54
รูปที่ 4.14 สัญญาณที่ได้จากวงจรเข้ารหัส 54
รูปที่ 4.15 สัญญาณที่ได้จากวงจรแมปปิ้ง 54
รูปที่ 4.16 สัญญาณที่ได้จากวงจรบวก (full adder) 6 บิต 55
รูปที่ 4.17 สัญญาณที่ได้จากวงจรถอดรหัส 55
รูปที่ 4.18 สัญญาณที่ได้จากวงจรบวก 56
รูปที่ 4.19 สัญญาณที่ได้จากวงจรตัดสินค่า 56
รูปที่ 4.20 สัญญาณที่ได้จากวงจรซีเรียลอินพาราแลลเอ้าท์ 57
รูปที่ 4.21 สัญญาณที่ได้จากวงจรหน่วยควบคุม 57
รูปที่ 4.22 สัญญาณที่ได้จากวงจรรวมเมื่อรหัสเป็น “01010101” 58
รูปที่ 4.23 สัญญาณที่ได้จากวงจรรวมเมื่อรหัสเป็น “00110011” 58
รูปที่ 4.24 สัญญาณที่ได้จากวงจรรวมเมื่อรหัสเป็น “01011010” 58
รูปที่ 4.25 สัญญาณที่ได้จากวงจรรวมเมื่อรหัสเป็น “00111100” 59
รูปที่ 4.26 สัญญาณที่ได้จากวงจรรวมเมื่อรหัสเป็น “01100110” 59
รูปที่ 4.27 สัญญาณที่ได้เมื่อรหัสเป็น “01010101” 60
รูปที่ 4.28 สัญญาณที่ได้เมื่อรหัสเป็น “00111100” 61
รูปที่ 4.29 สัญญาณที่ได้เมื่อรหัสเป็น “01100110” 62
รูปที่ 4.30 สัญญาณที่ได้จากวงจรรับข้อมูลจากพอร์ทอนุกรม 62
รูปที่ 4.31 สัญญาณที่ได้จากวงจรส่งข้อมูลออกพอร์ทอนุกรม 63

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

หน้า

ตารางที่ 2.1 รหัสตั้งฉาก 64x64 11

ตารางที่ 2.2 ความสัมพันธ์ระหว่างระยะห่างบิตสัญญาณ
กับค่าอัตสหสัมพันธ์รหัสวอล์ชขนาด 8x8 bit 14

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 ที่มาของปริญญานิพนธ์

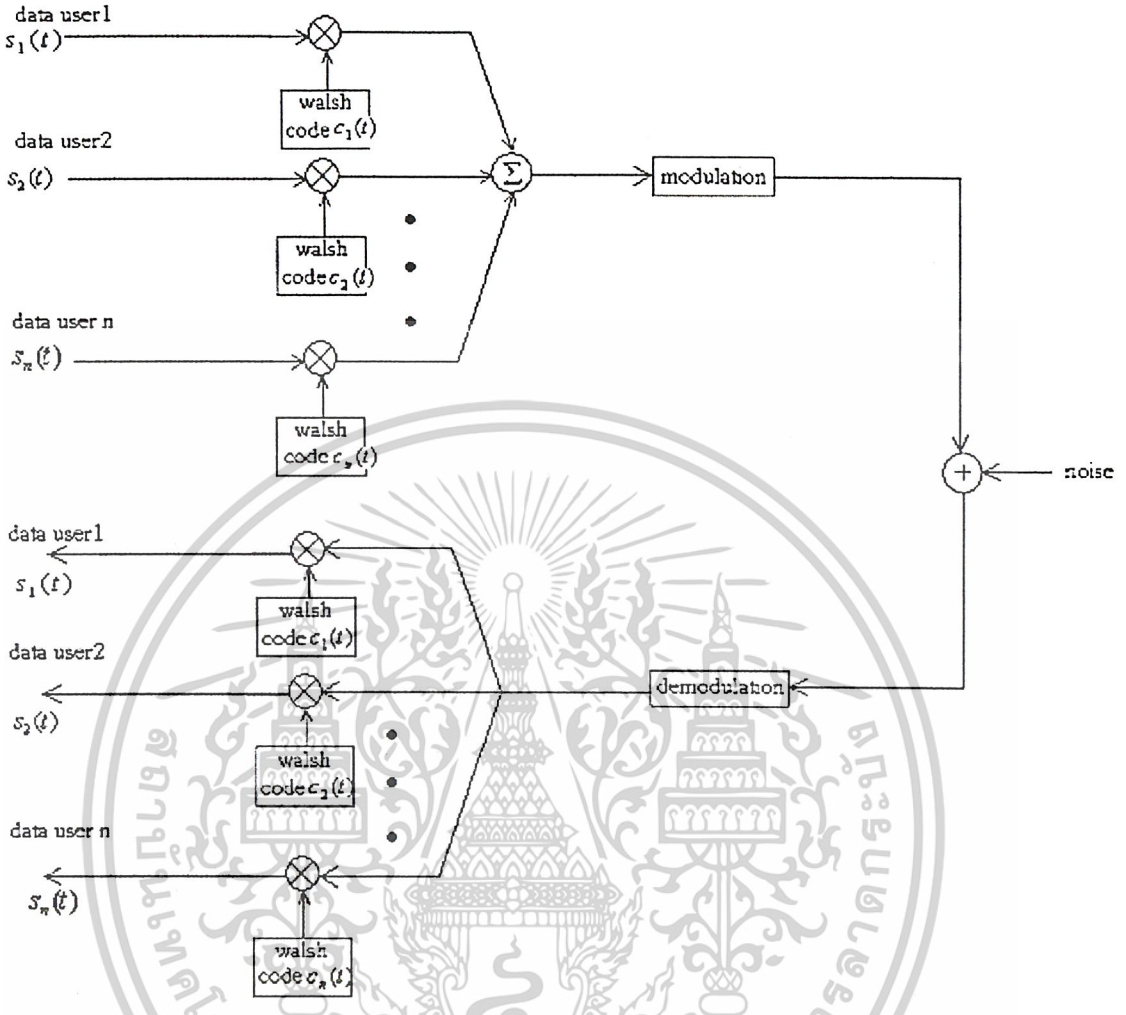
เนื่องจากการสื่อสารของโลกในปัจจุบันทุกวันนี้มีการพัฒนาอย่างรวดเร็ว ซึ่งการสื่อสารที่ถูกพัฒนาขึ้นมานี้มีความสำคัญมากขึ้นเรื่อยๆ สำหรับการพัฒนาต่าง ๆ ของประเทศ โดยเฉพาะทางด้านเศรษฐกิจ ซึ่งรูปแบบหนึ่งของการสื่อสารที่จะกล่าวถึงในปริญญานิพนธ์ฉบับนี้นั้นได้เริ่มมาจาก การมอดูเลตแบบสเปกตรัมแผ่ (spread-spectrum : ss) เริ่มถูกนำมาใช้ครั้งแรกในระบบสื่อสารทางทหาร หลังจากสงครามโลกครั้งที่ 2 ได้มีการวิเคราะห์การมอดูเลตแบบสเปกตรัมแบบสเปกตรัมแผ่เพื่อใช้ในระบบที่ต้องการป้องกันการรบกวนจากสัญญาณภายนอกสูง (jamming signal) ซึ่งผลการวิเคราะห์ดังกล่าวทำให้ทราบถึงคุณสมบัติของการมอดูเลตแบบสเปกตรัมแผ่กล่าวคือการมอดูเลตดังกล่าวลดผลของการแทรกสอด และสัญญาณรบกวนอันเนื่องมาจากการสื่อสารในช่องสัญญาณหลายวิถี ซึ่งเป็นคุณสมบัติที่เหมาะสมในระบบการเข้าถึงหลายทาง (multiple access) ทำให้มีการพัฒนาเทคนิคการเข้าถึงหลายทางแบบแบ่งรหัส (Code Division Multiple Access : CDMA) ขึ้น โดยระบบ CDMA สามารถรองรับผู้ใช้งานได้มากกว่าทั้งระบบ TDMA และ FDMA เนื่องจากการเพิ่มผู้ใช้งานในระบบจะทำให้สมรรถนะของระบบลดลงเล็กน้อยเท่านั้น นอกจากนี้ระบบ CDMA ยังไม่จำเป็นที่จะต้องทราบเวลาและความถี่ในการเข้าถึงข้อมูลที่ถูกต้องและเที่ยงตรงในการใช้งานเหมือนกับระบบ TDMA และ FDMA รูปแบบการสื่อสาร CDMA ที่นิยมนำมาใช้งานในเชิงพาณิชย์มากที่สุดคือการจัดลำดับโดยตรง (Direct Sequence)

ปริญญานิพนธ์ฉบับนี้จะแสดงถึงการจำลองการทำงานของระบบ CDMA บนโปรแกรม MATLAB เพื่อศึกษาเกี่ยวกับรหัสที่ใช้งาน ลักษณะของสัญญาณเมื่อมีคลื่นรบกวน และการออกแบบวงจรดิจิตอลของระบบร่วมกับภาษา VHDL

1.2 วัตถุประสงค์ของปริญญานิพนธ์

เพื่อศึกษาพื้นฐานการทำงานของระบบการเข้าถึงหลายทางแบบแบ่งรหัสลำดับตรง(Direct Sequence CDMA : DS-CDMA)โดยใช้โปรแกรม MATLAB ในการจำลองการทำงานในส่วนของการเข้ารหัส การถอดรหัส และผลของสัญญาณรบกวนที่มีต่อระบบโดยสามารถอธิบายหลักการและเหตุผลของแบบจำรองระบบ และทำการออกแบบวงจรดิจิตอลของระบบลงบน FPGA เพื่อนำไปประยุกต์ใช้งานจริงได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่1.1 บล็อกไดอะแกรมด้านส่งและด้านรับของระบบ DS-CDMA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎีและหลักการ

2.1 ทฤษฎีและหลักการระบบเข้าถึงหลายทางแบบแบ่งรหัส

2.1.1 ระบบการเข้าช่องสัญญาณ (Multiple access system)

ระบบสื่อสารอาจแบ่งได้เป็นระบบสื่อสารอนาล็อกและระบบสื่อสารดิจิตอล เมื่อระบบสื่อสารของอนาล็อกเริ่มเกิดปัญหาเกี่ยวกับความจุช่องสัญญาณที่ไม่พอ ทำให้การติดต่อผ่านเครือข่ายระบบสื่อสารอนาล็อกเกิดปัญหาและหยุดชะงัก ดังนั้นจึงเริ่มมีการพัฒนาระบบสื่อสารดิจิตอลขึ้นมาเพื่อเพิ่มความจุของช่องสัญญาณให้มากขึ้น สามารถแบ่งวิธีการเข้าถึงระบบเครือข่ายช่องสัญญาณดังนี้

1) การเข้าถึงหลายทางแบบแบ่งความถี่ (Frequency Division Multiple Access : FDMA) แต่ละช่องสัญญาณจะผลัดเปลี่ยนกันใช้งานที่ความถี่ไม่ตรงกัน
2) การเข้าถึงช่องสัญญาณแบบแบ่งช่วงเวลา (Time Division Multiple Access : TDMA) แต่ละช่องสัญญาณจะผลัดเปลี่ยนกันใช้แบนด์วิดท์ที่มีช่วงคาบเวลา (Time-sharing) ไม่ตรงกัน
3) การเข้าถึงสัญญาณแบบแยกความแตกต่างของรหัส (Code Division Multiple Access : CDMA) แต่ละช่องสัญญาณจะใช้สัญญาณแบนด์วิดท์เดียวกันและเวลาเดียวกันก็ได้และแต่ละช่องสัญญาณจะมีรหัสที่ต่างกัน
4) การเข้าถึงช่องสัญญาณแบ่งแยกด้วยโพลาไรซ์เซชัน (Polarization Division Multiple Access : PDMA) แต่ละช่องสัญญาณใช้โพลาไรซ์ต่างกัน ซึ่งวิธีนี้ไม่ถูกนำมาใช้กับระบบวิทยุโทรศัพท์เคลื่อนที่
5) การเข้าถึงช่องสัญญาณแบบแยกด้วยสเปซ (Space Division Multiple Access : SDMA) วิธีนี้ถูกกำหนดเพื่อนำมาใช้กับดาวเทียมที่มีสายอากาศแบบกำหนดตำแหน่งลำคลื่นได้ (spot beam) ซึ่งการติดต่อที่อยู่ในพื้นที่การควบคุมของลำคลื่นของสายอากาศต่างกัน สามารถใช้ความถี่เดียวกันก็ได้

2.1.2 การเข้าถึงหลายแบบแบ่งรหัส (CDMA)

ระบบ CDMA เป็นระบบที่มีคุณสมบัติของการเข้าถึงหลายทางโดยการเข้ารหัส (Coding) โดยแต่ละผู้ใช้งานจะมีรหัสเฉพาะที่แตกต่างกันในแต่ละผู้ใช้งาน เพื่อใช้ในการเข้ารหัสข้อมูลที่ต้องการส่ง โดยที่เครื่องรับจะรู้รหัสของผู้ใช้งานและจะทำการถอดรหัสหลังจากรับสัญญาณได้และนำไปแปลงเพื่อให้ได้สัญญาณเดิมกลับมา เมื่อแบนด์วิดท์ของสัญญาณรหัสมากกว่าแบนด์วิดท์ของข้อมูลที่ต้องการส่ง ซึ่งการเข้ารหัสจะเป็นการแผ่สเปกตรัมของสัญญาณและเรียกว่า การมอดูเลตแบบสเปกตรัมแผ่(Spread spectrum modulation)

ข้อกำหนดในการบ่งบอกว่าเป็น การมอดูเลตแบบสเปกตรัมแผ่มี 2 ข้อ คือ

1. แบนด์วิดท์ของสัญญาณที่ส่งออกไปต้องมากกว่าแบนด์วิดท์ของข้อมูลที่ยังไม่เข้ารหัสสเปกตรัมแผ่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. แบนด์วิดท์ของสัญญาณที่ส่งออกไปไม่ขึ้นอยู่กับข้อมูลที่ยังไม่เข้ารหัสสเปกตรัมแผ่

นั่นคือการมอดูเลตแบบสเปกตรัมแผ่จะแปลงสัญญาณข้อมูลข่าวสารไปเป็นสัญญาณส่งออกอากาศที่มีแบนด์วิดท์มากขึ้น ซึ่งทำได้โดยการเข้ารหัสสัญญาณข้อมูลโดยสัญญาณรหัสนี้จะไม่ขึ้นอยู่กับข้อมูลและมีความกว้างของสเปกตรัมมากกว่าสัญญาณข้อมูลและจะกระจายกำลังของสัญญาณตลอดช่วงแบนด์วิดท์กว้างทำให้ มีความหนาแน่นกำลังที่ต่ำ อัตราส่วนของแบนด์วิดท์ที่ทำการส่งกับแบนด์วิดท์ของข้อมูลเรียกว่า อัตราขยายประมวลผล (Processing Gain : Gp) ของระบบสเปกตรัมแผ่

$$G_p = \frac{B_t}{B_i} \qquad (1)$$

โดย B_t คือแบนด์วิดท์ที่ทำการส่งหลังการเข้ารหัสและ B_i คือแบนด์วิดท์ของข้อมูลก่อนการเข้ารหัส

ที่เครื่องรับจะเปรียบเทียบสัญญาณที่รับได้กับสัญญาณรหัสเพื่อให้ได้ข้อมูลเดิมก่อนการเข้ารหัสออกมานั่นคือเครื่องรับจะต้องรู้รหัสที่ใช้ในการมอดูเลตก่อน

สัญญาณสเปกตรัมแผ่จะมีคุณสมบัติที่แตกต่างจากสัญญาณแถบแคบดังนี้

1) ความสามารถของการเข้าถึงหลายทาง

ถ้ามีการส่งสัญญาณที่มีการแผ่สเปกตรัม เมื่อมีผู้ใช้งานหลายคนในเวลาเดียวกัน ทางเครื่องรับก็จะสามารถแยกผู้ใช้งานแต่ละรายได้โดยใช้รหัสที่แตกต่างกันของแต่ละผู้ใช้งานซึ่งรหัสดังกล่าวต้องมีค่าสหสัมพันธ์ข้าม (Cross correlation) ที่ต่ำพอการหาค่าสหสัมพันธ์ของสัญญาณที่รับได้ด้วยรหัสสเปกตรัมแผ่จะดีสเปสสัญญาณของผู้ใช้งานนั้นแต่จะทำการแผ่สเปกตรัมของสัญญาณของผู้ใช้รายอื่นแทน นั่นคือกำลังของสัญญาณในแบนด์วิดท์ของข้อมูลจะมากกว่ากำลังของสัญญาณการแทรกสอดซึ่งทำให้แยกสัญญาณที่ต้องการได้ โดยความสามารถของการเข้าถึงหลายทาง

2) การป้องกันการแทรกสอดแบบหลายวิถี

ในช่องสัญญาณวิทยุสัญญาณระหว่างเครื่องส่ง-รับ จะมีจำนวนเส้นทางมากกว่า 1 วิถี อันเนื่องมาจากการสะท้อนและการหักเห ซึ่งสัญญาณในวิถีต่างๆ คือสัญญาณที่ถูกส่งมาต่างกันที่ขนาดแอมพลิจูดและเฟสเมื่อรวมสัญญาณดังกล่าวที่เครื่องรับก็จะเกิดการเสริมกันบางความถี่และหักล้างกันบางความถี่ ซึ่งการมอดูเลตแบบสเปกตรัมแผ่สามารถแก้ปัญหาดังกล่าวได้ โดยขึ้นกับชนิดของวิธีการมอดูเลตที่ใช้

3) เป็นระบบส่วนบุคคล

นั่นคือเครื่องรับจะดีสเปสได้ต้องรู้รหัสของสัญญาณที่ต้องการเท่านั้น

4) ลดผลจากการสอดแทรก

สหสัมพันธ์ข้ามของสัญญาณรหัสกับสัญญาณแถบแคบจะเป็นการกระจายกำลังงานของสัญญาณแถบแคบนั่นคือ ไปลดกำลังของสัญญาณแทรกสอดในแบนด์วิดท์ของข้อมูล โดยสัญญาณที่มีการแผ่กระจายสเปกตรัม (S) กับ สัญญาณการแทรกสอดแถบแคบ (I) ที่เครื่องรับ สัญญาณที่มีการแผ่สเปกตรัม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะดีสเปสในขณะที่สัญญาณแทรกสอดจะถูกแผ่สเปกตรัมกลายเป็นสัญญาณรบกวนพื้นหลัง(background noise) เมื่อเทียบกับสัญญาณที่ดีสเปส

2.1.3 การมอดูเลตแบบสเปกตรัมแผ่ (Spread Spectrum Modulation : SS Modulation)

ระบบ CDMA ใช้หลักการมอดูเลตแบบสเปกตรัมแผ่ ทำให้ข้อมูลข่าวสารที่ใช้ในการสื่อสารถูกแผ่กระจายไปบนแกนความถี่ทำให้มีแบนด์วิดท์ของสัญญาณข้อมูลที่ต้องการจะส่งมาก ซึ่งสามารถแบ่งได้ ดังนี้

2.1.3.1 การจัดลำดับโดยตรง (Direct Sequence : DS) โดยสัญญาณข้อมูลจะถูกคูณโดยตรงกับรหัสข้อมูลที่มีความเร็วสูง

2.1.3.2 การกระโดดทางความถี่ (Frequency Hopping : FH) โดยความถี่คลื่นพาห์ของสัญญาณข้อมูลที่ถูกส่งไปจะมีการเปลี่ยนแปลงอย่างรวดเร็วอันเนื่องมาจากรหัสข้อมูล

2.1.3.3 การกระโดดทางเวลา (Time Hopping : TH) สัญญาณข้อมูลไม่ได้ส่งอย่างต่อเนื่องแต่จะส่งเป็นเบิรสต์สั้นๆ โดยเวลาที่ส่งจะเป็นไปตามรหัสข้อมูล

2.1.3.4 วิธีมอเลตแบบไฮบริด(Hybrid System) เป็นการรวมเอาเทคนิค 2 หรือมากกว่าของเทคนิคที่กล่าวไว้ข้างต้น ใช้ด้วยกันเพื่อแก้ไขข้อบกพร่องของแต่ละวิธี

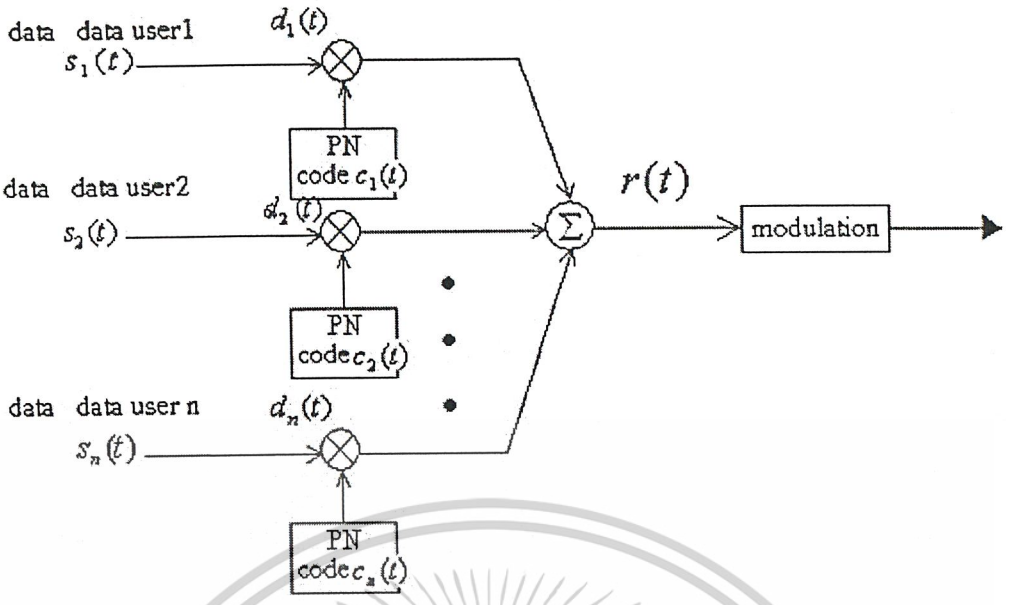
2.1.4 การจัดลำดับโดยตรง (Direct Sequence : DS)

ใน DS-CDMA สัญญาณข้อมูลจะถูกมอดูเลตโดยตรงกับสัญญาณรหัสดิจิตอลโดยสัญญาณข้อมูล อาจจะเป็นสัญญาณดิจิตอลหรืออนาลือกก็ได้ สัญญาณข้อมูลจะถูกคูณกับสัญญาณรหัสและได้สัญญาณที่มอดูเลตเป็นคลื่นพาห์แบบแถบกว้างซึ่งเป็นเหตุผลที่เรียกวิธีนี้ว่า DS-CDMA

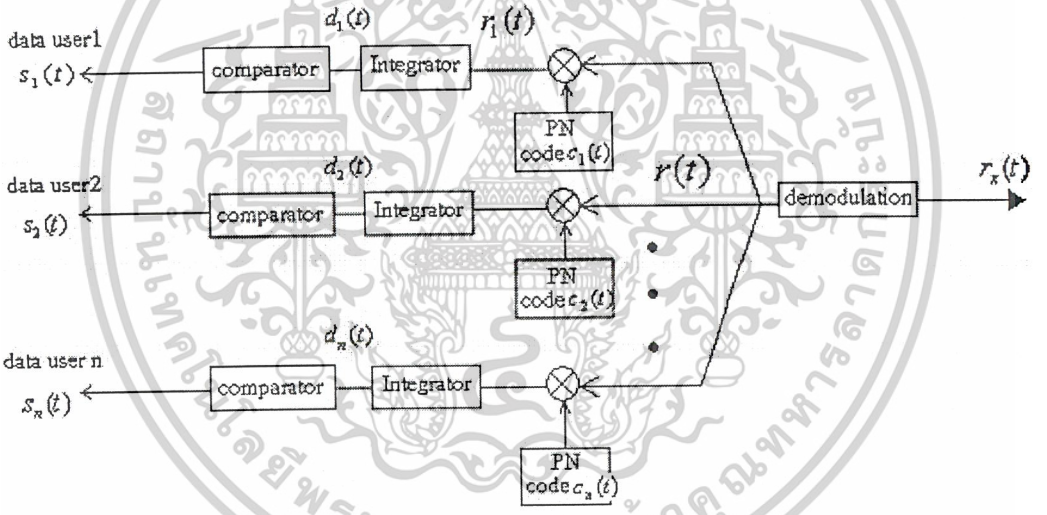
พื้นฐานของ DS-CDMA

รหัส PN (Psudorandom Noise code) คือรหัสตั้งฉากที่มีคุณสมบัติเฉพาะตัว โดยนำรหัส PN ที่ต่างกันนำไปคูณกับข้อมูลของผู้ใช้งานเป็นการแยกข้อมูลของแต่ละผู้ใช้งานออกจากกัน เมื่อนำรหัส PN แผ่ข้อมูลของผู้ใช้ที่ทางด้านส่ง ข้อมูลเดิมนี้จะสามารถถูกระบบที่ทางด้านรับ รับได้โดยการเอาสัญญาณที่รับเข้ามาหลังจากดีมอดูเลตแล้ว คูณกับรหัส PN ตัวเดิม จะเห็นได้จากรูปที่ 2.3 ว่าทางด้านรับสามารถจับสัญญาณจากผู้ต้องการข้อมูลที่อยู่ในสัญญาณที่นำเสนอออกจากข้อมูลของผู้ใช้อื่น ๆ ในระบบ CDMA พิจารณารูป 2.1 เป็นบล็อกไดอะแกรมง่าย ๆ ของ ทางด้านส่ง CDMA จะแผ่ข้อมูลด้วยรหัส PN และทางด้านรับก็จะคูณด้วยรหัส PN ตัวเดิม แล้วเข้าวงจรอินทริเกรเตอร์ (Integrator) และวงจรคอมแพเรเตอร์ (comparator) เพื่อให้ได้ข้อมูลเดิมกลับคืน รูปที่ 2.2 เป็นบล็อกไดอะแกรมง่าย ๆ ของ ทางด้านรับ CDMA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.1 บล็อกไดอะแกรมของ ทางด้านส่ง CDMA



รูปที่ 2.2 บล็อกไดอะแกรมของ ทางด้านรับCDMA

สมมติว่าทางด้านรับต้องการข้อมูลของผู้ใช้ที่ 1 (user 1) สัญญาณ $r_x(t)$ ที่ส่งเข้ามาจะถูกดีมอดูเลตแล้วเอ้าพุท ที่ได้เป็น สัญญาณ $r(t)$ ออกมาจะมาคูณกลับรหัส PN ของผู้ใช้ที่ 1 และผลได้จะนำมาผ่านวงจรอินทริเกรเตอร์ แล้วนำมาแปลงเป็นข้อมูลฐานสอง จะได้ข้อมูลของผู้ใช้ที่ 1 ออกมา

$$r(t) = \sum_{i=1}^{n} d_i(t)c_i(t) \tag{2}$$

$$= d_1(t) * c_1(t) + d_2(t) * c_2(t) + d_3(t) * c_3(t) + \ldots\ldots \tag{3}$$

$$r_1(t) = r(t) * c_1(t) \tag{4}$$

$$= d_1(t) * c_1(t) * c_1(t) + d_2(t) * c_2(t) * c_1(t) + d_3(t) * c_3(t) * c_1(t) + \ldots. \tag{5}$$

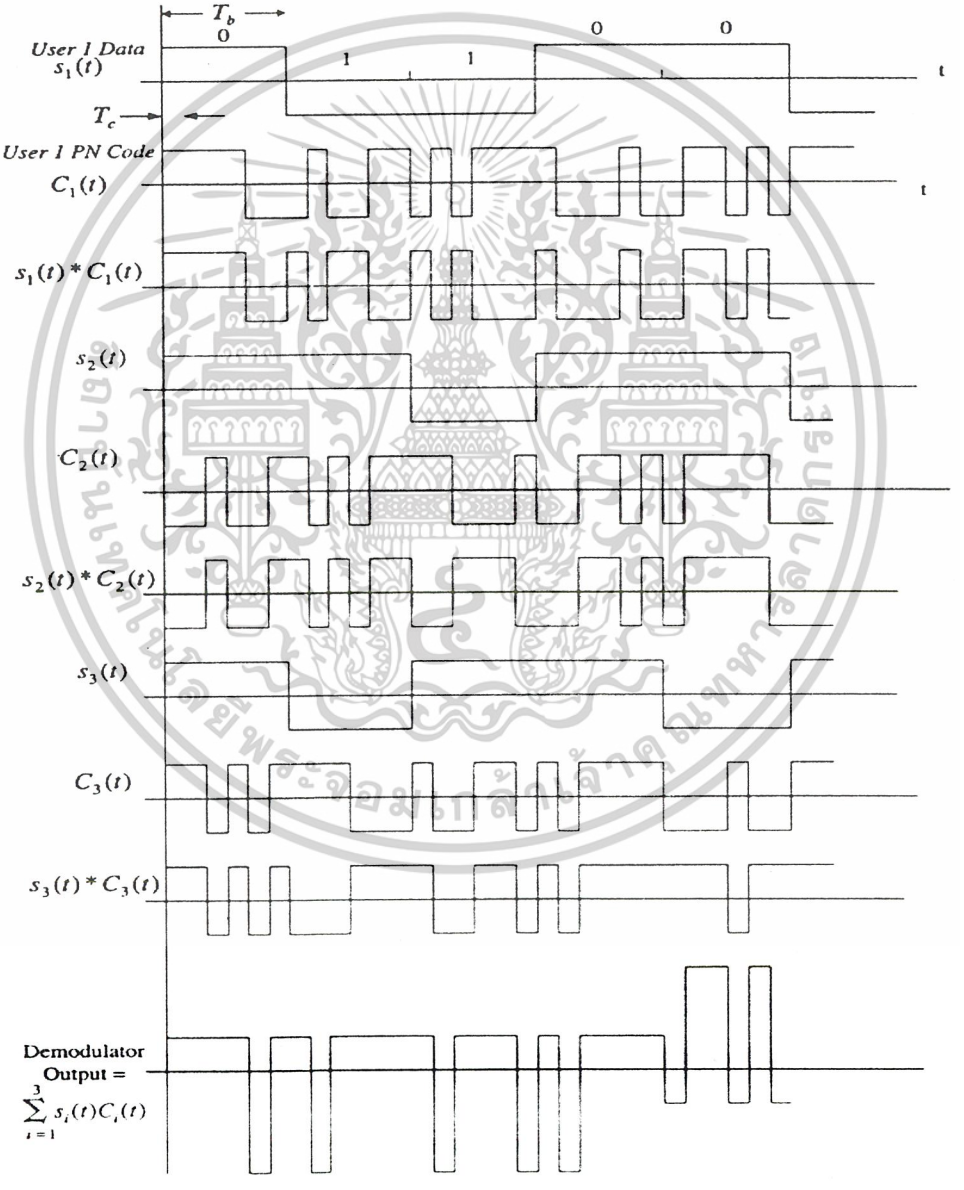
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความสัมพันธ์ระหว่าง $c_1(t)$ กับ $c_2(t)$ และ $c_1(t)$ กับ $c_3(t)$ มีค่าน้อยมาก ดังนั้นจะได้ว่าเทอมที่ 2 และ 3 มีค่าเหมือนเป็นสัญญาณรบกวน (noise) เมื่อผ่านการอินทริเกรเตอร์ แล้วจะมีค่าเป็น 0 ส่วนเทอมที่1 เมื่อผ่านการอินทริเกรเตอร์ แล้วจะได้ค่าประมาณ $d_1(t)$

เพราะว่า

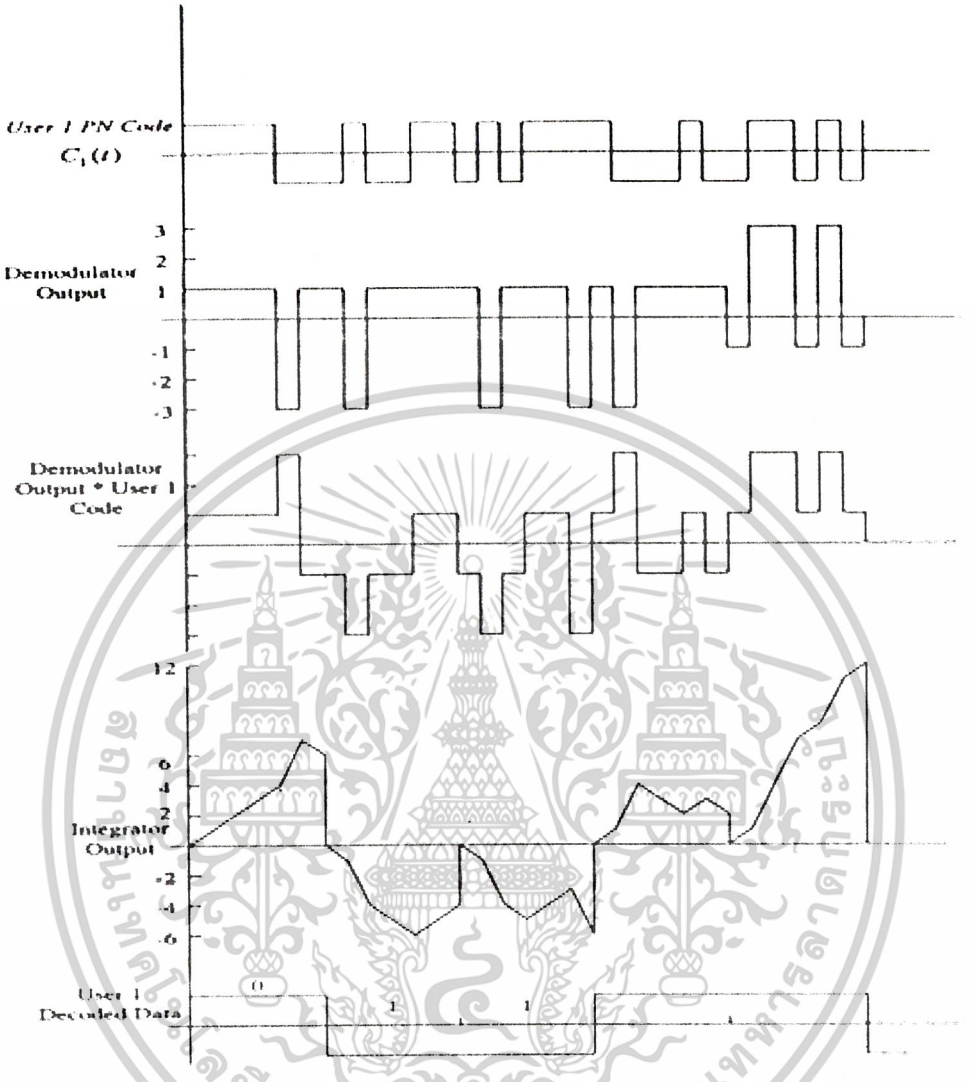
$$c_1(t) * c_1(t) = 1 \tag{6}$$

ซึ่งทั้งหมดนี้อธิบายได้โดยรูปที่ 2.3



รูปที่ 2.3 ไดอะแกรมอธิบายลักษณะสัญญาณรวมที่มาจากหลายผู้ใช้ที่มาถึงด้านรับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.4 อธิบายถึงกระบวนการที่จะได้สัญญาณที่ต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.5 การสเปรด (Spreading)

ใน UMTS (Universal Mobile Telecommunications) และ cdma2000 signaling และข้อมูลผู้ใช้ (user data) จะมีการแผ่ 2 ครั้ง เริ่มด้วยกับรหัสการเข้าถึงช่องสัญญาณ (Channelization codes) ซึ่งเป็นโครงสร้างของรหัสวอล์ช (Walsh code) ค่อนข้างที่จะทนทานต่อการรบกวนที่มาจากหลายผู้ใช้ (multiple user) ได้มากกว่า หลังจากนั้นกับรหัสสเครมบลิง (Scrambling code) ซึ่งไม่จำเป็นที่จะต้องตั้งฉากเรียกว่ารหัส PN

- รหัสวอล์ช (Walsh Code)

การแพร่แบบตั้งฉากโดยใช้ฟังก์ชันวอล์ช (Orthogonal spreading using Walsh Function)

แต่ละรหัสช่องสัญญาณที่ถูกส่งบนช่องสัญญาณฟอร์เวอร์ด (Forward) ถูกแพร่โดยใช้ ฟังก์ชันวอล์ช เพื่อที่จะจัดเตรียมการจัดช่องสัญญาณแบบตั้งฉากระหว่างรหัสช่องสัญญาณทั้งหมด

การมอดูเลตสำหรับช่องสัญญาณรีเวอร์ด (Reverse) เป็นการมอดูเลตแบบตั้งฉาก 64 อาเรย์(array)

สัญญาณหลังจากการมอดูเลต (Modulation symbol) เป็น 1 ใน 64 รูปสัญญาณซึ่งสร้างโดยฟังก์ชันวอล์ช 1 ใน 64 รูปสัญญาณจะถูกส่งสำหรับแต่ละ 6 สัญญาณ(บิต)

เช่น $c_i, 0 \le i \le 5$

ฟังก์ชันวอล์ช ถูกสร้างโดยใช้ เมตริกฮาดามาร์ด (Hadamard matrix) H_n ซึ่ง n คือกำลังของ 2 ; $n = 2^m$ โดย เมตริกฮาดามารด H_n คือเมตริกซ์ตั้งฉาก $n \times n$ ซึ่งสมาชิกในเมตริกซ์เป็น +1 และ -1

สมาชิกทั้งหมดในแถวและหลักแรกของ H_n เป็น +1 ทั้งหมด โดยแถวอื่นๆบรรจุด้วย +1 และ -1 เท่าๆกัน เมตริกฮาดามาร์ด $n \times n$ สามารถมีจริงได้โดย n ต้องเป็นกำลังของ 2 ($n = 2^m$)

โดย เปลี่ยนจาก +1 เป็น 0 และ -1 เป็น 1 , เมตริกฮาดามาร์ด สำหรับ $n = 2$ ถูกแสดงโดย

$$H_2 = \begin{bmatrix} +1+1 \\ +1-1 \end{bmatrix} = \begin{bmatrix} 00 \\ 01 \end{bmatrix} \tag{7}$$

เมตริกฮาดามาร์ด สำหรับ $n = 2^6$ เป็น 64x64 ฟังก์ชันวอล์ช ตั้งฉาก แสดงในตารางที่ 2.1

64x64 เมตริกซ์นี้สามารถสร้างได้โดยกระบวนการวนลูป สำหรับ $n = 2^m$, $0 \le m$ (จำนวนเต็มบวก)

$$H_1 = 0 \quad for \quad m = 0 \tag{8}$$

$$H_2 = \begin{bmatrix} H_1 H_1 \\ H_1 \overline{H_1} \end{bmatrix} = \begin{bmatrix} 00 \\ 01 \end{bmatrix} \quad for \quad m = 1 \tag{9}$$

$$H_4 = \begin{bmatrix} H_2 H_2 \\ H_2 \overline{H_2} \end{bmatrix} = \begin{bmatrix} 0000 \\ 0101 \\ 0011 \\ 0110 \end{bmatrix} \quad for \quad m = 2 \tag{10}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$H_8 = \begin{bmatrix} H_4 H_4 \\ H_4 \overline{H_4} \end{bmatrix} = \begin{bmatrix} 00000000 \\ 01010101 \\ 00110011 \\ 01100110 \\ 00001111 \\ 01101010 \\ 00111100 \\ 01101001 \end{bmatrix} \quad for \;\; m = 3 \tag{11}$$

.

.

.

$$H_{2n} = \begin{bmatrix} H_n H_n \\ H_n \overline{H_n} \end{bmatrix} \tag{12}$$

ซึ่ง $2n = 2^{m+1}$ และ $\overline{H_n}$ หมายถึง คอมพิเมนท์ของ H_n

ดังนั้นรหัสช่องสัญญาณที่ถูกแพร่โดยใช้ ฟังก์ชันวอล์ช ถูกกำหนดค่ารหัสช่องสัญญาณให้มีหมายเลข n โดย $0 \le n \le 63$

รหัสช่องสัญญาณหมายเลข 0 ถูกกำหนดให้เป็นช่องสัญญาณนำร่อง (pilot channel)

รหัสช่องสัญญาณหมายเลข 1-7 ถูกกำหนดให้เป็นช่องสัญญาณซิงค์ (sync channel)

รหัสช่องสัญญาณหมายเลข 32 ถูกกำหนดให้เป็นช่องสัญญาณเพจจิง (paging channel)

รหัสช่องสัญญาณที่เหลือสำหรับกำหนดให้เป็นช่องสัญญาณฟอร์เวอร์ด (Forward Traffic)

สำหรับช่องสัญญาณรีเวอร์ด CDMA สัญญาณที่ได้จากการมอดูเลตจะเป็น 1 จาก 64 สัญญาณที่ตั้งฉากที่สร้างโดย ฟังก์ชันวอล์ช สัญญาณที่ได้จากการมอดูเลตจะมีหมายเลข 0 ถึง 63 ซึ่งจะสัมพันธ์กับสูตร

$$MSI = c_0 + 2c_1 + 4c_2 + 8c_3 + 16c_4 + 32c_5 \tag{13}$$

ซึ่ง MSI : Modulation symbol index หมายถึง มอดูเลชัน ซิมโบล อินเด็กซ์

และ $c_i, 0 \le i \le 5$ แทน 6 สัญญาณเอาท์พุทของแต่ละกลุ่ม โดย 6 สัญญาณรหัส จะเป็น 1 สัญญาณที่ถูกมอดูเลต

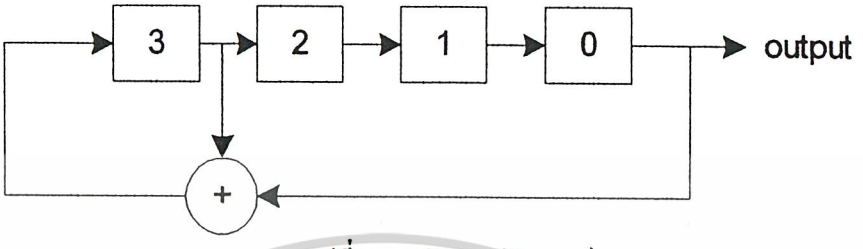
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
2	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
3	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
4	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
5	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
6	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
7	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
8	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
9	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
10	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
11	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1
12	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1	1	1
13	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0
14	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0
15	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
17	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
18	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
19	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1
20	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
21	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
22	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
23	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1
24	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
25	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
26	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
27	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
28	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
29	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
30	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0
31	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
33	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
34	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
35	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
36	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
37	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0
38	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0
39	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
40	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
41	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
42	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
43	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
44	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
45	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
46	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
47	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
48	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
49	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
50	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
51	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
52	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
53	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
54	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
55	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
56	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
57	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0
58	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
59	0	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1

- รหัสสเครมบลิง (Scrambling Codes)

รหัส PN จากการสร้างพื้นฐานของรหัสสเครมบลิง ซึ่งถูกสร้างโดยมัลติบิตชิฟรีจิสเตอร์ (multibit shift register) ที่ซึ่งเอาท์พุต บางตัวถูกบวกแบบมอดูโล 2 (modulo2) หรือวงจร X-OR แล้วป้อนกลับไปยังอินพุต

ทฤษฎีของรหัส PN



รูปที่ 2.5 4 บิต ชิฟรีจิสเตอร์

จากรูปแถวของชิฟรีจิสเตอร์ ประกอบ ไปด้วย 4 สัญญาณบิตชิฟรีจิสเตอร์, เอาท์พุตของ ชิฟรีจิสเตอร์ 3 และ 0 จะรวมกันแบบมอดูโล 2 แล้วป้อนกลับไปยังอินพุตของ ชิฟรีจิสเตอร์ 3

สมมติ สถานะเริ่มต้นของทั้งหมดคือ 1,1,1และ1 ที่เวลา t = 0 ผลลัพธ์ของ ชิฟรีจิสเตอร์ 3 กับชิฟรีจิสเตอร์ 0 คือ 0 เมื่อเกิดสัญญาณนาฬิกาลูกแรกขึ้นสถานะของ 4 บิต ชิฟรีจิสเตอร์ เปลี่ยนเป็น 0 1 1 1 ต่อไปเรื่อยๆ จะได้สถานะเป็น

1 1 1 1
0 1 1 1
1 0 1 1
0 1 0 1
1 0 1 0
1 1 0 1
0 1 1 0
0 0 1 1
1 0 0 1
0 1 0 0
0 0 1 0
0 0 0 1
1 0 0 0
1 1 0 0
1 1 1 0
1 1 1 1 (ไม่จำเป็นต้องเขียนลำดับเอาท์พุต มากไปกว่า 15 คล็อกพัลซ์ (clock pulse) เพราะว่าจะซ้ำ)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อสังเกต(จากตัวอย่าง)

1. ลำดับของเอาท์พุตจะออกมาเป็นคาบ แต่บิตข้างในจะเกิดแบบสุ่ม ดังนั้นอาจจะถูกใช้เป็นรูปแบบของซูโดแรนดอม (pseudo random pattern) จากตัวอย่าง คาบ = 15 = 2^4- 1 : จาก 4 บิตชิฟรีจิตเตอร์ ลำดับที่ยาวที่สุดของคาบที่เป็นไปได้ที่สามารถถูกกำเนิดโดยชิฟรีจิตเตอร์ ถูกเรียกว่า ลำดับ (sequence) หรือ ลำดับยาวที่สุด (maximum length sequence)

2. ในแต่ละคาบจะมี 0 = 7 ตัว ละ 1 = 8 หมายความว่าถ้าไม่นับบิตเริ่มต้นแล้วจำนวนบิต 0 และ 1 จะเท่ากัน

ลำดับบางลำดับของชิฟรีจิตเตอร์ มีคุณสมบัติครบตามนี้

1. ในแต่ละคาบถ้าไม่นับบิตเริ่มต้นแล้วจำนวนบิต 0 และ 1 จะเท่ากัน

2. ฟังก์ชันอัตสหสัมพันธ์ (auto-correlation function) *C(j)* ของลำดับคาบ { b_n } ของคาบ N จะถูกจำกัดความโดย

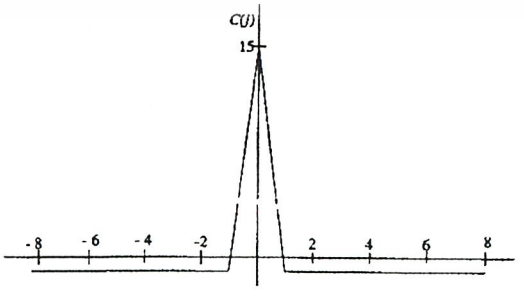
$$C(j) = \sum_{i=1}^{N} b_i b_{i+1} \qquad (14)$$

อธิบายอีกอย่างหนึ่งว่า ผลบวกของตัวเลขแต่ละตัวในลำดับซึ่งมันจะถูกหน่วงและเก็บค่าไว้ สมมติว่าแต่ละ b_i เป็นได้ทั้ง 1 และ -1 , ฟังก์ชันอัตสหสัมพันธ์ (C_j) ของลำดับชิฟรีจิตเตอร์ จะมี 2 ค่า

$$C(j) = N \text{ ถ้า } j = 0 \qquad (15)$$

$$= K \text{ ถ้า } 0 < j < N \qquad (16)$$

เห็นว่าคุณสมบัติฟังก์ชันอัตสหสัมพันธ์ (C_j) ของลำดับชิฟรีจิตเตอร์ รูป 2.5 คือเมื่อไม่เลื่อนเฟส คือ $C_{(0)}$ = + 15 ของลำดับเอาท์พุต = 1 1 1 1 0 1 0 1 1 0 0 1 0 0 0 , เมื่อเลื่อนเฟส 1 บิต ลำดับของ เอาท์พุต = 0 1 1 1 1 0 1 0 1 1 0 0 1 0 0 จะได้ฟังก์ชันอัตสหสัมพันธ์ *C(+1)* = -1และเมื่อเลื่อนเฟส 1 บิต ไปทางซ้ายลำดับของเอาท์พุต = 1 1 1 0 1 0 1 1 0 0 1 0 0 0 1 จะได้ฟังก์ชันอัตสหสัมพันธ์ *C(-1)* = -1 ทำอย่างนี้เรื่อย ๆ จะได้ผลดังรูปที่ 2.6



รูปที่ 2.6 ฟังก์ชันอัตสหสัมพันธ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.6 การซิงโครไนซ์ของระบบ

ในระบบ CDMA ทางด้านรับ เมื่อต้องการที่จะถอดรหัสสัญญาณเพื่อจะถอดรหัสสัญญาณเพื่อที่จะให้ได้ข้อมูลกลับคืนมานั้น วงจรทางด้านรับไม่จำเป็นที่จะต้องมีสัญญาณนาฬิกาเพื่อที่จะทำให้วงจรทำงานซิงโครไนซ์ (Synchronize) แต่ระบบ CDMA จะสามารถทำการซิงโครไนซ์ ได้โดยตรวจสอบสัญญาณเบื้องต้นที่เรียกว่าช่องสัญญาณซิงค์ (Sync channel) แล้วแต่ทำการหาอัตสหสัมพันธ์ ซึ่งค่าอัตสหสัมพันธ์นี้มีค่าที่บ่งชี้ว่ารหัสสัญญาณที่ได้รับกับรหัสตั้งฉากทางด้านรับมีความคลาดเคลื่อนห่างกันเท่าใด (จำนวนบิต) ดังนั้นเมื่อทราบระยะทางคลาดเคลื่อนแล้วระบบก็จะทำการเลื่อนบิตให้ตรงกันแล้วทำการตรวจสอบอีกครั้งหนึ่ง และเมื่อสัญญาณตรงกันแล้ว (Synchronize) ระบบก็จะสามารถถอดรหัสข้อมูลที่ส่งมาได้อย่างถูกต้องแสดงตัวอย่างการหาค่าอัตสหสัมพันธ์ของรหัสวอล์ชขนาด 8x8 ดังตารางที่ 2.2

Walsh Code	ระยะห่างสัญญาณและค่าอัตสหสัมพันธ์																
	0	1	2	3	4	5	6	7	8	7	6	5	4	3	2	1	0
11111111	-8	-6	-4	-2	0	2	4	6	8	8	8	8	8	8	8	8	8
10101010	0	-2	2	-4	4	-6	6	-8	8	-8	8	-8	8	-8	8	-8	8
11001100	0	-2	-4	0	4	-2	-8	0	8	0	-8	0	8	0	-8	0	8
10011001	0	2	-2	-2	4	2	-6	-2	8	0	-8	0	8	0	-8	0	8
11110000	0	-2	-4	-4	-8	-4	0	-4	8	4	0	-4	-8	-4	0	4	8
10100101	0	2	-2	-2	-4	0	2	-6	8	-4	0	4	-8	4	0	-4	8
11000011	0	2	4	4	-4	-4	-4	-2	8	4	0	-4	-8	-4	0	4	8
10010110	0	-2	2	2	-4	4	4	-4	8	-4	0	4	-8	4	0	-4	8

ตารางที่ 2.2 ความสัมพันธ์ระหว่างระยะห่างบิตสัญญาณกับค่าอัตหสัมพันธ์ของรหัสวอล์ชขนาด 8x8 bit

ในบางกรณีการหาค่าอัตสหสัมพันธ์จะไม่สามารถหาได้ตรงตามตัวอย่างเนื่องจากผลของการรวมกันของสัญญาณที่มีมากกว่า 1 สัญญาณ (มากกว่า 1 ข้อมูลมารวมกัน) ทำให้ค่าอัตสหสัมพันธ์ที่หาได้มีขนาดลดลง ซึ่งจะกล่าวต่อไปในตัวอย่างการทดลองที่ 3.1.6

2.2 ทฤษฎีและหลักการที่ใช้ในการออกแบบวงจรดิจิตอล

รหัสตั้งฉาก (Orthogonal Code) เป็นรหัสที่มีลักษณะพิเศษ คือ การพยายามถอดรหัสอื่นที่ไม่ถูกต้องจะทำให้ข้อมูลที่ได้ออกมามีความสัมพันธ์เป็นศูนย์ หรือไม่ได้ข้อมูลอะไรเลย หากอธิบายในเชิงพีชคณิตแล้วอาจอธิบายได้ด้วยความสัมพันธ์ที่ว่าผลลัพธ์ของการ X-OR (Exclusive OR) ของรหัสตั้งฉากชุดใดชุดหนึ่ง จะมีจำนวนหนึ่งกับศูนย์เท่ากันซึ่งการที่ให้ผลลัพธ์เช่นนี้จะมีผลดีในแง่ของการเข้ารหัสคือค่าอัตสหสัมพันธ์ ของรหัสเป็นศูนย์ หมายความว่าการพยายามถอดรหัสด้วยข้อมูลชุดอื่นที่ไม่ตรงกัน จะไม่สามารถให้ข้อมูลอะไรออกมาได้เลย ตัวอย่างของรหัสตั้งฉาก ความยาว 4 บิต ดังต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Code Sequence 1	0 0 0 0
Code Sequence 2	0 1 0 1
Code Sequence 3	0 0 1 1
Code Sequence 4	0 1 1 0

ถ้านำเอารหัสชุดเดียวกันมา X-OR กันจะพบว่าผลลัพธ์ที่ได้มีค่าเป็น ศูนย์หมด ดังนี้

0 1 0 1
<u>0 1 0 1</u>
0 0 0 0 (XOR เหมือนกันเป็น 0 ต่างกันเป็น 1)

แต่หากลองนำเอารหัสชุดที่ 1 และ 2 มา X-OR กัน จะได้ผลลัพธ์ที่มีเลขหนึ่งและศูนย์เท่ากันดังนี้

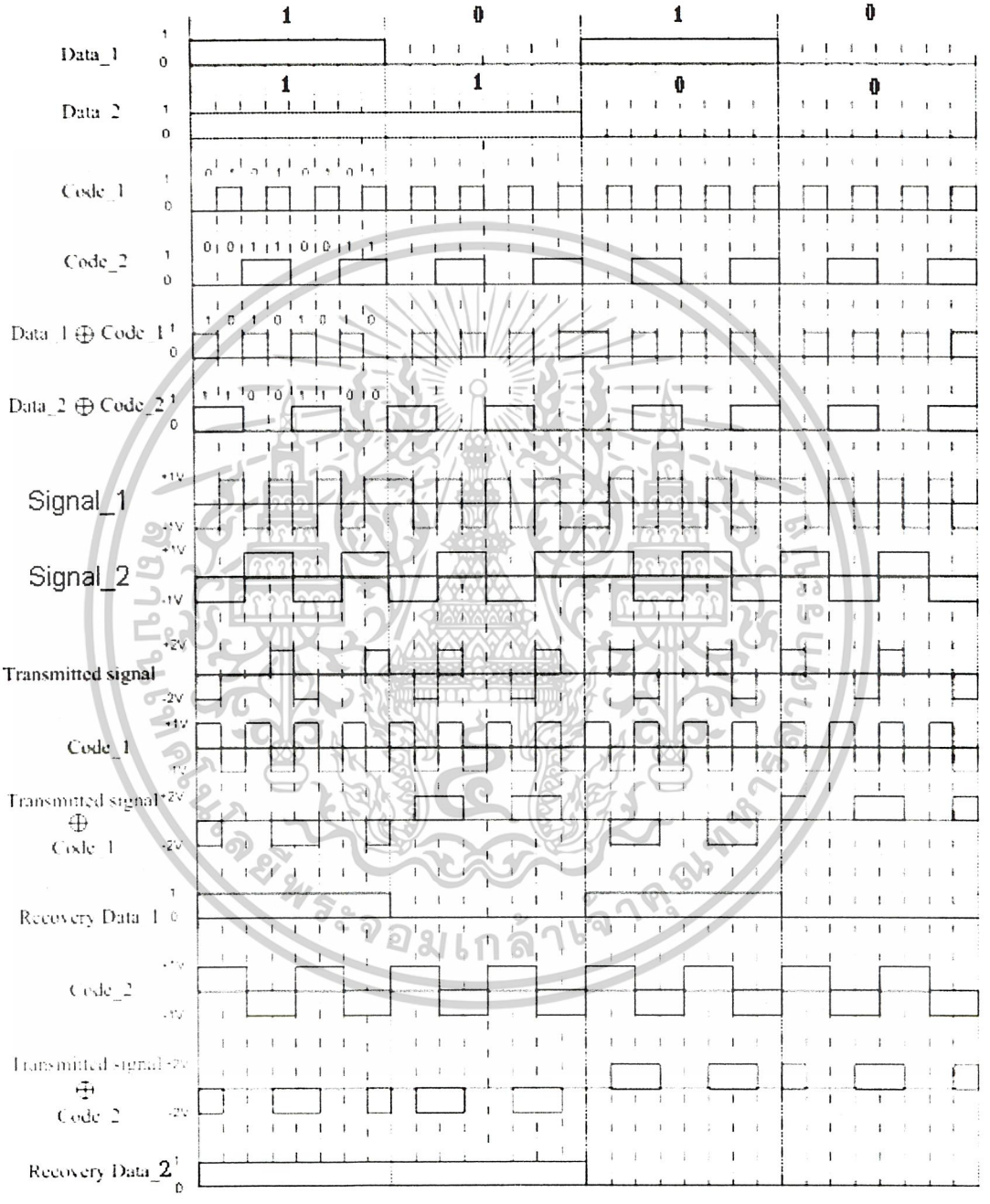
0 0 0 0
<u>0 1 0 1</u>
0 1 0 1

ในแง่ของระบบสื่อสารนั้น ข้อมูลดิจิตอล ที่เป็น 0 หรือ 1 นี้ จะถูกส่งผ่านระบบเครือข่ายสื่อสารด้วยศักย์ไฟฟ้า +1 และ-1 โวลต์ ที่ด้านรับสัญญาณจะถูกตีความออกมาโดยอาศัยการ X-OR สัญญาณข้อมูลที่ได้รับ และนำเอาค่าศักย์ไฟฟ้าที่ได้มารวมกันในช่วงเวลาใดเวลาหนึ่ง ผลลัพธ์ที่ได้ คือ ค่าสหสัมพันธ์ข้าม ซึ่งจะถูกนำไปแปลเป็นบิตของข้อมูลอีกครั้งหนึ่ง จากตัวอย่างทั้งสองข้างต้นนั้น ในกรณีที่เป็นรหัสคนละชุด ผลลัพธ์ที่ได้จากการ X-OR นั้น จะมี 0 และ 1 เท่าๆกัน ซึ่งเมื่อคิดผลรวมของศักย์ไฟฟ้าในช่วงความยาว 4 บิตข้อมูลนี้ จะได้ศักย์ไฟฟ้าศูนย์โวลต์ ซึ่งไม่สามารถนำไปตีความเป็นข้อมูลได้เนื่องจากไม่ใกล้เคียงทั้ง +1 และ -1 แต่ในตัวอย่างหลังที่ใช้รหัสชุดเดียวกัน จะได้ผลลัพธ์เป็น 0 บิตทั้งหมด ซึ่งเมื่อรวมค่าศักย์ไฟฟ้าทั้ง 4 บิต จะได้เป็น +4 โวลต์ และเฉลี่ยออกมาได้ +1 โวลต์ต่อบิต ซึ่งสามารถตีความข้อมูลออกมาได้เป็นบิต 0 ได้

โดยปกติแล้วค่าสหสัมพันธ์ข้าม คือค่าที่บอกถึงความสัมพันธ์ของรหัสหรือสัญญาณสองชุดว่ามีความเกี่ยวข้องสัมพันธ์กันมากน้อยเพียงใด ในกรณีของรหัสตั้งฉาก นั้นจะพบว่า รหัสแต่ละชุดที่ต่างกันจะไม่มีความสัมพันธ์เกี่ยวข้องกัน นั่นคือรหัสชุดหนึ่งจะไม่สามารถให้ข้อมูลเกี่ยวกับรหัสชุดอื่นได้ ดั้งนั้น การนำเอารหัสคนละชุดมาพยายามถอดรหัสข้อมูลชุดอื่นจะได้ค่าสหสัมพันธ์ข้าม เป็นศูนย์ ซึ่งไม่มีประโยชน์ และไม่สามารถตีความออกมาได้ แต่ถ้าใช้รหัสชุดที่ถูกต้อง คือรหัสชุดเดียวกัน จะได้ค่าสหสัมพันธ์ข้าม ที่มีความหมายและสามารถตีความข้อมูลออกมาคืนได้ คุณสมบัติของรหัสตั้งฉาก ดังที่กล่าวมาข้างต้นนี้ ทำให้ระบบ CDMA เลือกใช้รหัสดังกล่าวสำหรับแยกแยะช่องสัญญาณแต่ละช่องออกจากกัน และส่งรวมกันออกไปภายใต้ความถี่เดียว ที่ด้านรับสามารถแปลงเป็นข้อมูลที่ต้องการได้ โดย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อาศัยการถอดรหัสด้วยรหัสตั้งฉาก ที่ถูกต้อง ระบบ CDMA จึงสามารถใช้ความถี่ที่มีอยู่ร่วมกันได้ (ในทางทฤษฎี มีค่าองค์ประกอบความถี่ใช้ใหม่ (Frequency Reuse Factor) เป็น 1) ซึ่งเป็นลักษณะสำคัญของระบบนี้ดีกว่าระบบอื่น และเพื่อให้เข้าใจถึงลักษณะการนำเอารหัสตั้งฉาก ไปใช้ในการแยกช่องสัญญาณ สามารถพิจารณาได้จากรูปที่ 2.7



รูปที่ 2.7 หลักการรหัสตั้งฉาก ที่ใช้ในการออกแบบวงจรดิจิตอล

ซึ่งสามารถอธิบายได้ดังนี้คือ สมมุติให้ระบบมีผู้ใช้งาน 2 คน โดยแต่ละคนจะมีรหัสที่ตั้งฉากกัน โดยรหัสที่ใช้คือรหัสวอล์ชที่สร้างจากเมตริกฮาดามาร์ด 8x8 ซึ่งจะได้รหัส 8 บิต โดยกำหนดให้ผู้ใช้คน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แรกมีรหัส “01010101” (Code_1) และผู้ใช้คนที่สองมีรหัส “00110011” (Code_2) และในกรณีนี้ผู้ใช้แต่ละคนมีชุดข้อมูลที่จะส่ง 4 บิตคือ Data_1 และ Data_2 ตามลำดับ ภายหลังการเข้ารหัส ข้อมูลที่ถูกส่งออก (Transmitted signal) มากเป็น 8 เท่าของข้อมูลเดิม ในแง่ของแบนด์วิดท์ในการส่งข้อมูลนั้นก็จะกว้างขึ้นเป็น 8 เท่าเป็นเหตุผลว่าทำไมระบบ CDMA จึงต้องการแบนด์วิดท์มากกว่าปกติ เช่น 1.23 MHz ในกรณีของ IS-95 A และทำให้ระบบ CDMA ได้ชื่อว่าเป็น การมอดูเลตแบบสเปกตรัมแผ่ (Spread Spectrum) เนื่องจากมีการกระจายแบนด์วิดท์ให้กว้างขึ้นด้วยการเข้ารหัสนั่นเอง หลังจากที่ได้ทำการแผ่ข้อมูลด้วยรหัสตั้งฉาก แล้วและเมื่อแทนบิท ‘0’ ด้วยศักย์ไฟฟ้า +1V และบิต ‘1’ ด้วยศักย์ไฟฟ้า -1V จะได้เป็นสัญญาณ Signal_1 และ Signal_2 ตามลำดับ

ซึ่งถ้าจะส่งทั้ง 2 สัญญาณนี้เลยจะใช้ช่องสัญญาณ 2 ช่องหรือ 2 ความถี่ แต่ถ้าต้องการส่งโดยใช้ช่องสัญญาณเพียงช่องเดียวจะนำสัญญาณไฟฟ้าของข้อมูลที่แผ่แล้วมารวมกัน สัญญาณรวมของสัญญาณทั้งสองชุดได้เป็นสัญญาณข้อมูลที่ถูกส่งออก ซึ่งมีศักย์ไฟฟ้าเพิ่มขึ้นเป็น $\pm 2V$ ในบางจุด (ในกรณีระบบมีผู้ใช้งาน 2 คน)

ที่ด้านรับเมื่อได้รับสัญญาณดังกล่าว ก็จะทำการดีสเปสข้อมูลซึ่งก็คือการถอดรหัส (เข้ารหัสอีกครั้ง) ด้วยรหัสตั้งฉาก ชุดที่ถูกต้อง เช่นถ้าผู้ใช้งานคนที่ 1 ต้องการข้อมูลกลับคืนก็จะนำรหัสที่ 1 (Code_1) มาถอดรหัสโดยอาศัยการคูณ (Multiplication) จะทำให้ได้สัญญาณไฟฟ้าอีกชุดหนึ่งซึ่งใน ที่นี้คือสหสัมพันธ์ข้าม ระหว่างสัญญาณที่ได้รับกับสัญญาณรหัสที่ 1 นั่นเอง จะได้เป็นสัญญาณกู้คืน (Recovery) Data_1 ซึ่งมีแบนด์วิดท์ของลดลงเหลือเท่าเดิมเช่นเดียวกับด้านส่ง ทำให้เรียกการถอดรหัสด้วยรหัสตั้งฉากที่ด้านรับว่าเป็นการดีสเปส และถ้าต้องการข้อมูลของผู้ใช้งานคนที่ 2 ก็ทำเช่นเดียวกันโดยการนำเอารหัสที่ 2 (Code_2) มาคูณกับสัญญาณที่ได้รับ ก็จะได้สัญญาณกู้คืน Data_2 ซึ่งเป็นข้อมูลเดิมของตัวเองกลับคืน

หากพิจารณาในแง่ของพลังงานที่ใช้ในการส่งนั้น จะพบว่าด้านที่ส่งนั้นจะมีการกระจายพลังงานในหนึ่งบิตของข้อมูล กล่าวอีกนัยหนึ่งคือข้อมูลนั่นเอง ออกไปเป็นบิตย่อย 8 บิต ที่ด้านรับจะนำเอาพลังงานย่อยเหล่านี้มารวมกันอีกครั้ง และตีความออกมาเป็นข้อมูลเดิม ซึ่งจะรวมพลังงานจาก 8 บิต ย่อยและเฉลี่ยออกมาเพื่อตีความเป็นข้อมูลเดิม 1 บิต การทำเช่นนี้มีส่วนช่วยให้ระบบ CDMA มีความทนทานต่อสัญญาณรบกวน โดยหากมีข้อมูลบิตย่อยใดโดนรบกวนหรือสูญหายไประหว่างทาง ก็ยังคงมีบิตย่อยอื่นเหลืออยู่ ซึ่งสามารถเอาพลังงานที่เหลือมารวมกันและตีความออกมาเป็นข้อมูลออกมาได้

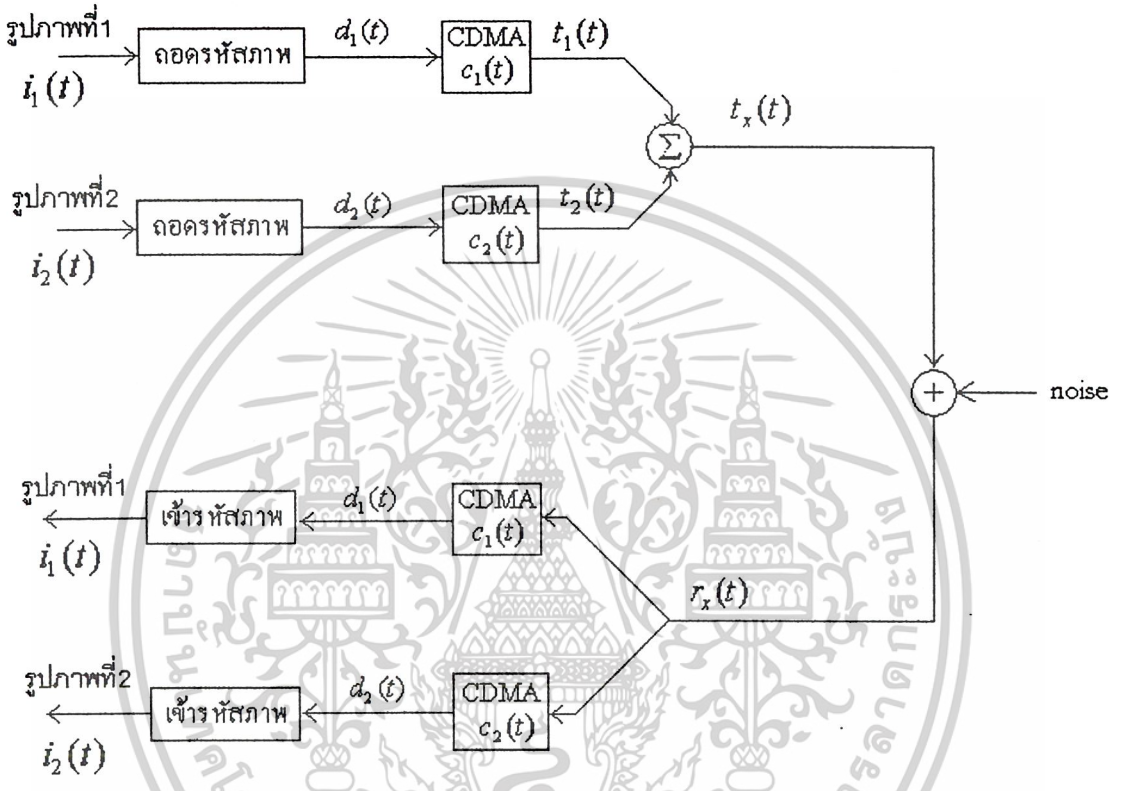
ในทางปฏิบัติ ความยาวของรหัสตั้งฉาก จะต้องมีความเหมาะสม นั่นคือไม่ยาวไม่สั้นจนเกินไป การที่รหัสนั้นสั้นเกินไป จะทำให้ชุดรหัสมีจำนวนน้อยเกินไปจนไม่เพียงพอกับช่องสัญญาณอีกทั้งยังมีความปลอดภัยต่ำด้วย แต่ถ้ายาวเกินไปจะทำให้มีความซับซ้อนจนอุปกรณ์มีราคาแพงจนเกินไป ในระบบ IS-95A นั้นจะอาศัยรหัสวอล์ชซึ่งเป็นรหัสตั้งฉาก ที่มีความยาว 64 บิต

61942
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
การออกแบบและการสร้าง

3.1 การออกแบบและการสร้างด้วยโปรแกรม MATLAB

ระบบ DS-CDMA ที่ศึกษาประกอบด้วยภาคส่ง ช่องสัญญาณ ภาครับ แสดงด้วยบล็อกไดอะแกรมดังรูปที่ 3.1

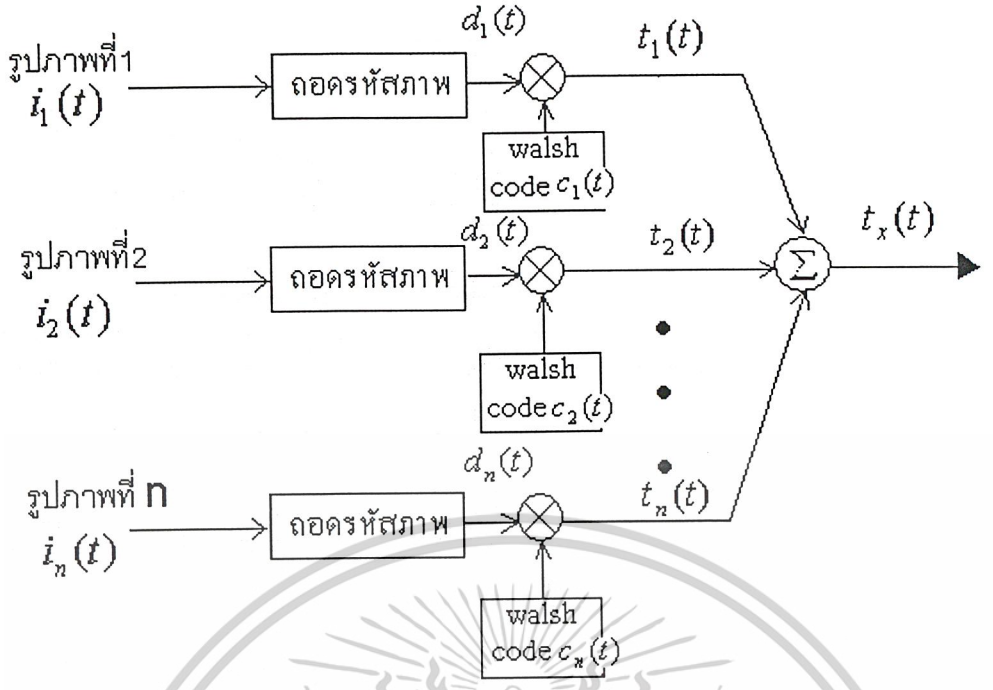


รูปที่ 3.1 แบบจำลองระบบ DS-CDMA ที่ศึกษา

3.1.1 แบบจำลองเครื่องส่ง (Transmitter model)

ระบบ DS-CDMA ที่ศึกษากำหนดให้มีจำนวนผู้ใช้งานเท่ากับ 2 ให้ข้อมูล (information) ของผู้ใช้งานเป็นข้อมูลภาพ ดังนั้นต้องนำข้อมูลรูปภาพของผู้ใช้งานมาถอดรหัสภาพก่อนแล้วจึงนำข้อมูล (data) ที่ได้ไปเข้ารหัส PN ซึ่งรหัส PN ที่ใช้นั้นคือ รหัสวอล์ชกำหนดให้ข้อมูลกับรหัสวอล์ชที่พิจารณาเป็นแบบเลขฐานสอง ซึ่งหลักการของ ระบบ DS-CDMA จะนำลำดับรหัส (code sequence)และลำดับข้อมูล (data sequence) มาทำการคูณกันโดยตรง แล้วนำเอาข้อมูลที่คูณกับรหัสวอล์ชขนาด 8บิต แล้วของแต่ละผู้ใช้งานมารวมกันเพื่อส่งออกไป แบบจำลองภาคส่ง DS-CDMA แสดงดังรูปที่ 3.2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.2 แบบจำลองภาคส่ง DS-CDMA ที่ศึกษา

สัญญาณที่ถูกส่งออกไปสำหรับผู้ใช้งาน n คน แสดงในสมการที่ 17

$$T_x = \sum_{i=1}^{n} d_i * c_i \qquad (17)$$

โดย c_i คือลำดับรหัสของผู้ใช้งานลำดับที่ i

d_i คือลำดับข้อมูลที่ต้องการส่งของผู้ใช้งานลำดับที่ i

n คือจำนวนผู้ใช้งานทั้งหมด

3.1.2 แบบจำลองช่องสัญญาณ (Channel model)

สัญญาณที่ทำการพิจารณานั้นจะออกแบบการทดลองให้ผ่านช่องสัญญาณที่มีส่วนประกอบของสัญญาณรบกวนเกาส์ขาวแบบบวก (Additive White Gaussian Noise : AWGN)

ในการทดลองจะทำปรับการปรับเปลี่ยนระดับของสัญญาณรบกวนเพื่อศึกษาผลเปรียบเทียบที่ได้โดยการปรับเปลี่ยนระดับของสัญญาณรบกวนนั้นจะใช้ระดับของอัตราส่วนสัญญาณที่เราต้องการต่อสัญญาณรบกวน (SNR : Signal to noise ratio) ดังสมการที่ 19

$$SNR = \frac{Average\_signal\_energy}{Noise\_\mathrm{var}iance} \qquad (18)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$SNR = 10\log_{10}\frac{E_b}{\sigma^2} \tag{19}$$

โดยที่ ค่าพลังงานเฉลี่ยของสัญญาณ (Average signal energy : Eb) ดังสมการที่ 20

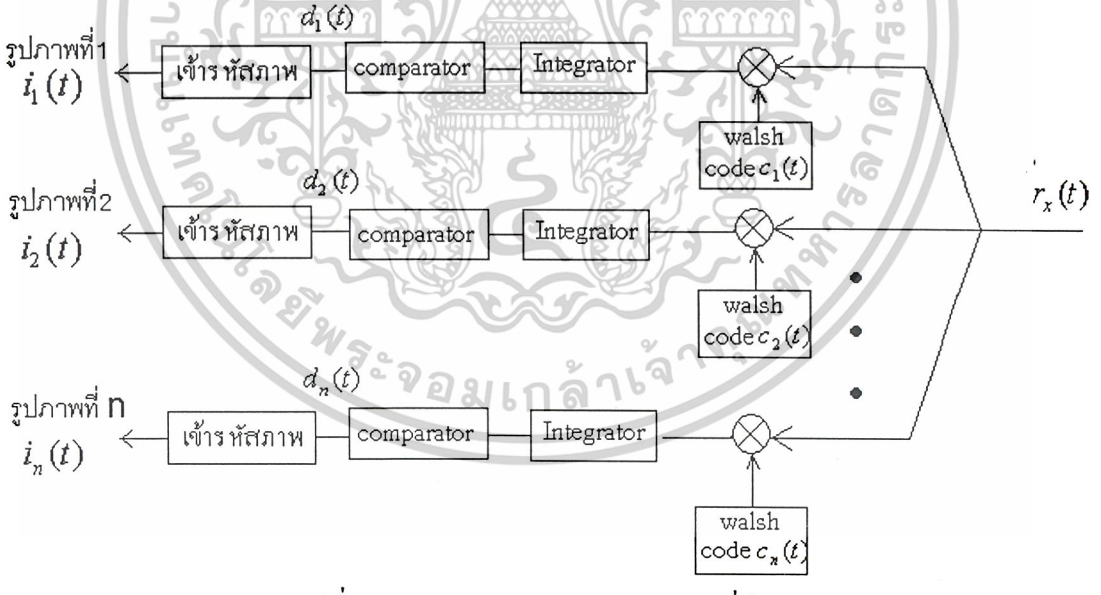
$$E_b = \frac{\sum_{n=1}^{N}(Tx)^2}{N} \tag{20}$$

ค่าความแปรปรวนของสัญญาณรบกวน (Noise variance) จะอ้างอิงระดับของ SNR ดังสมการที่ 21

$$\sigma = \sqrt{\frac{E_b}{10^2 * SNR}} \tag{21}$$

3.1.3 แบบจำลองเครื่องรับ (Receiver model)

แบบจำลองภาครับ DS-CDMA แสดงดังรูปที่ 3.3



รูปที่ 3.3 แบบจำลองภาครับ DS-CDMA ที่ศึกษา

โดยสัญญาณที่รับได้ที่เครื่องรับจะประกอบไปด้วยสัญญาณที่มาจากผู้ใช้หลายๆคน และสัญญาณรบกวนเกาส์ขาวแบบบวก นำสัญญาณที่รับมาได้คูณด้วยรหัสวอล์ชตัวเดิม ซึ่งเมื่อผ่านวงจรอินทริเกรเตอร์ และ วงจรคอมแพเรเตอร์ แล้วก็จะได้ข้อมูลเดิมของผู้ใช้งานแต่ละคน จากนั้นนำไปเข้ารหัสภาพ ก็จะได้ข้อมูลที่เป็นรูปภาพกลับคืน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.4 การวิเคราะห์สมรรถนะของระบบ

อัตราความผิดพลาดบิตของระบบ จากการสับเปลี่ยนระดับสัญญาณในช่องสัญญาณตามสมการที่ 3.2 ในแต่ละระดับสัญญาณจะทำการทดลอบอัตราความผิดพลาดบิต (Bit error rate : BER)

ซึ่งในระบบ DS-CDMA ที่ศึกษานี้สามารถแสดงได้ 2 แบบคือ

1. อัตราความผิดพลาดของบิตข้อมูล (data bit error rate)
2. อัตราความผิดพลาดของบิตสัญญาณ (signal bit error rate)

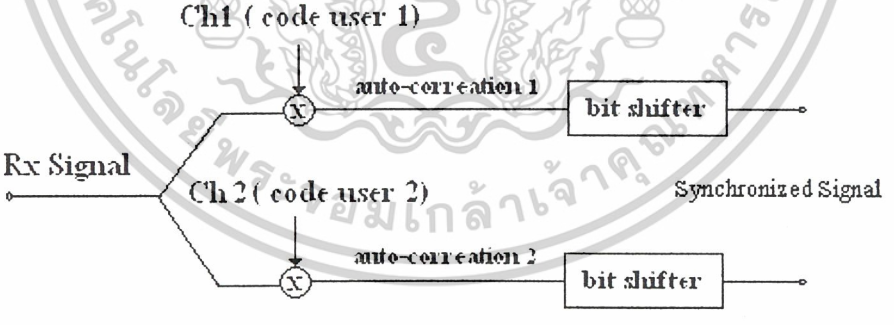
3.1.5 วิเคราะห์สมรรถภาพของระบบเมื่อมีการเปลี่ยนแปลงของรหัสวอล์ช

เมื่อทำการศึกษาสมรรถภาพของระบบเมื่อรหัส มีขนาดเพิ่มขึ้นนอกจากจะทำให้ระบบสามารถรองรับจำนวนผู้ใช้ได้มากขึ้นตามขนาดของรหัสแล้ว ยังช่วยให้ระบบสารมารถถอดรหัสข้อมูลกับคืนได้ง่ายแม้สัญญาณอยู่ในระดับต่ำ หรืออีกในหนึ่งก็คือระบบสามารถป้องกันสัญญาณรบกวนได้มากขึ้น

ทำการเปรียบเทียบทดลองโดยการเขียนโปรแกรมแสดงผลของระบบที่ใช้รหัสวอล์ชขนาด 8-128 bit ที่ได้ทำการแก้ไขจากโปรแกรมแรก ซึ่งผลการทดลองที่ได้แสดงในหัวข้อที่ 4.1.2

3.1.6 วิเคราะห์การซิงโครไนซ์

จากการทดลองที่ผ่านมาจะทำการส่งข้อมูล 2 ข้อมูลผ่านระบบไปพร้อมๆ กัน ดังนั้นการหาค่าอัตสหสัมพันธ์ของแต่ละสัญญาณจะทำให้ได้ค่าอัตสหสัมพันธ์ที่ต่ำลงซึ่งจะหาได้โดยนำสัญญาณที่ได้รับไม่ว่าจะเริ่มที่ตำแหน่งให้ก็ตามมาทำการถอดรหัสกับสัญญาณข้อมูลนั้นโดยมีบล็อกไดอะแกรม ตามรูปที่ 3.4



รูปที่ 3.4 บล็อกไดอะแกรม การซิงโครไนซ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 การออกแบบวงจรและการสร้างด้วยภาษา VHDL

3.2.1 ระบบแบบหลายผู้ใช้

การทดลองนี้กำหนดให้มีผู้ใช้งาน 4 คน โดยข้อมูลของแต่ละคนนำมาจากเครื่องกำเนิดสัญญาณ (waveform generator) ซึ่งเป็นสัญญาณซายน์ คนละความถี่ โดยก่อนที่จะนำสัญญาณทั้ง 4 มาเข้าสู่ระบบที่อยู่ภายในชิป FPGA จะเปลี่ยนข้อมูลสัญญาณซายน์ ซึ่งเป็นสัญญาณอนาล็อก ให้เป็นสัญญาณดิจิตอล ที่วงจร A/D (Analog to Digital) ก่อน ซึ่งภายในชิป FPGA จะนำสัญญาณทั้ง 4 ไปประมวณผลโดยใช้หลักการของ DS-CDMA ทั้งทางด้านส่งและด้านรับ ทางด้านส่งเรากำหนดให้ผู้ใช้งานทั้ง 4 มีรหัสของตัวเองโดยรหัสที่ใช้เป็นรหัสวอล์ชที่สร้างมาจาก เมตริกฮาดามาร์ด 8x8 ซึ่งจะได้ 8 รหัสแต่ในที่นี้เราจะนำมาใช้เพียง 4 รหัสโดยแต่ละผู้ใช้งานจะมีรหัสของตัวเองดังนี้

ผู้ใช้งานคนที่ 1 “01010101”

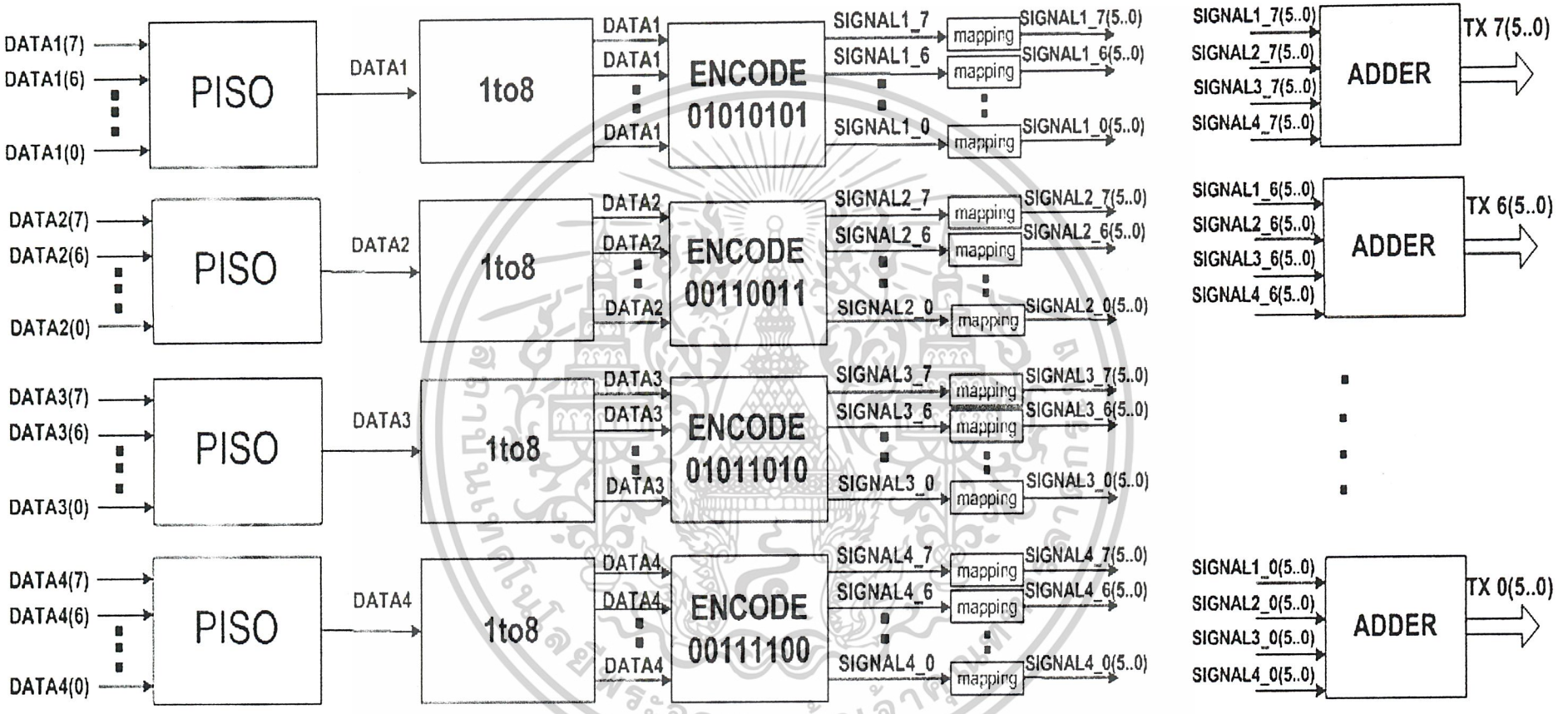
ผู้ใช้งานคนที่ 2 “00110011”

ผู้ใช้งานคนที่ 3 “01011010”

ผู้ใช้งานคนที่ 4 “00111100”

และทางด้านรับเราจะให้รับค่ารหัส 8 บิตซึ่งถ้ารหัสเป็นของผู้ใช้คนไหน ข้อมูลที่ได้ก็จะเป็นของผู้ใช้คนนั้น และเมื่อสัญญาณออกจากชิป FPGA ก็จะนำไปแปลงเป็นสัญญาณอนาล็อกตามเดิม เพื่อนำไปเปรียบเทียบกับสัญญาณซายน์ ที่อินพุตว่าตรงกันกับเจ้าของรหัสหรือไม่ รูปบล็อกไดอะแกรมของทางด้านส่งและด้านรับของวงจรดิจิตอลที่ออกแบบเป็นดังรูปที่ 3.5 และ 3.6 ตามลำดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.5 บล็อกไดอะแกรมด้านส่งของวงจรดิจิตอลที่ออกแบบ

รูปที่ 3.6 บล็อกไดอะแกรมด้านรับของวงจรดิจิตอลที่ออกแบบ

ซึ่งภายในชิป FPGA ประกอบด้วยวงจรต่างๆดังนี้

3.2.1 ด้านส่ง

3.2.1.1 วงจรหารความถี่

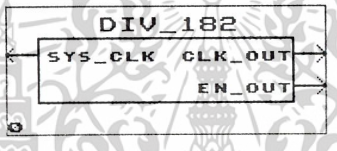
ในวงจร D/A นั้นเราต้องมีความถี่ในการแซมปลิง ซึ่งความถี่ที่จะใช้ในการแซมปลิงนั้นจะได้จากการนำความถี่จาก crystal oscillator บนบอร์ด FPGA ซึ่งในที่นี้เราใช้ crystal oscillator ที่มีความถี่ 16.257 MHz มาทำการหารความถี่ที่วงจรหารความถี่ ความถี่ที่เราต้องการจะสัมพันธ์กับจำนวนสเตจ (state) ในการทำงานครบ 1 รอบ ซึ่งจำนวนสเตจเราสามารถกำหนดได้ในวงจรควบคุมการทำงาน (control unit) โดยแต่ละสเตจจะใช้เวลาเท่ากับสัญญาณนาฬิกาที่ใช้ในระบบ 1 ลูก โดยสัญญาณนาฬิกาที่ใช้ในระบบในการทดลองนี้เราใช้ 5.4 MHz (นำความถี่จาก crystal oscillator มาหาร 3) ซึ่งเป็นความถี่ที่เหมาะสม โดยสัญญาณนาฬิกาที่ใช้ในการแซมปลิงจะต้องมีคาบที่มากกว่าหรือเท่ากับเวลาที่ใช้ในการทำงานครบ 1 รอบ เพื่อป้องกันไม่ให้สัญญาณอินพุตตัวใหม่เข้าสู่ระบบโดยที่ยังประมวณผลสัญญาณเก่าไม่เสร็จ ตัวอย่างเช่น ในการทดลองนี้การทำงานทั้งหมดของระบบใช้ 57 สเตจซึ่งแต่ละเสตจใช้เวลา 185 ns ($\frac{1}{5.4MHz}$) เพราะฉะนั้นใช้เวลาทั้งหมด10.56 us แปลงเป็นความถี่เท่ากับ 94.7 kHz เพราะฉะนั้นความถี่ในการแซมปลิงจะต้องน้อยกว่าหรือเท่ากับ 94.7 kHz นี้และเพื่อให้สัญญาณอินพุตที่จะเข้าสู่ระบบสัมพันธ์กับการทำงานของระบบ โดยถ้าหารได้ความถี่ที่น้อยกว่าเราจะใช้สัญญาณจากวงจรหารความถี่นี้ไปเป็นสัญญาณ enable ที่วงจรควบคุมการทำงาน เพื่อให้วงจรควบคุมการทำงานจะทำงานได้ก็ต่อเมื่อวงจรหารความถี่ทำการแซมปลิงได้อินพุตตัวใหม่แล้วเท่านั้น ในการทดลองนี้เราให้วงจรหารความถี่เท่ากับ 182 ซึ่งเมื่อหารความถี่ 16.257 MHz แล้วจะได้ความถี่ในการแซมปลิงประมาณ 90 kHz ซึ่งวงจรหารความถี่สามารถบรรยายด้วย VHDL code ได้ดังนี้

```
Library ieee;
Use ieee.std_logic_1164.ALL;
Use ieee.std_logic_unsigned.ALL;

Entity div_182 is
Port (   sys_clk                : in  std_logic;
         clk_out,en_out  : out std_logic);
End div_182  ;

Architecture behavior  of div_182 is
SIGNAL state  : integer range 0 to 1;
SIGNAL count  : integer range 0 to 91;

Begin
   PROCESS (sys_clk)
     BEGIN
       if sys_clk'event and sys_clk='1' then
            case state IS
            when 0 =>     if count<90 then
                               count<=count+1;
                               clk_out<='1';
                               state<=0;
                               if count>2 then
                                    en_out<='1';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        end if;
                    else
                        count<=0;
                        state<=1;
                    end if;
            when others => if count<90 then
                        count<=count+1;
                        clk_out<='0';
                        state<=1;
                        else
                        count<=0;
                        state<=0;
                        en_out<='0';
                    end if;
        End case;
    End if;
  End PROCESS ;
End behavior;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.7 บล็อกวงจรหารความถี่ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

3.2.1.2 วงจรพาราแลลอินซีเรียล เอ้าท์ (Parallel In Serial Out : PISO)

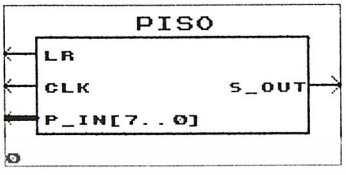
ข้อมูลของแต่ละผู้ใช้งานจะนำมาผ่านวงจร A/D โดยสัญญาณที่ได้จะได้เป็นข้อมูลดิจิตอลทีละ 8 บิต ซึ่งจะเป็นอินพุตที่เข้าสู่ชิป FPGA แต่ภายในชิป เป็นการประมวณผลสัญญาณทีละบิตดังนั้นจึงต้องนำสัญญาณอินพุตมาแปลงจาก 8 บิตให้ออกทีละบิตที่วงจรพาราแลลอินซีเรียลเอ้าท์ โดยวงจรนี้สามารถบรรยายด้วย VHDL Code ได้ดังนี้

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_Unsigned.ALL;

ENTITY piso IS
PORT( lr,clk : IN  std_logic;
      p_in   : IN std_logic_vector(7 downto 0);
      s_out  : OUT std_logic);
END piso;

ARCHITECTURE behavioral OF piso IS
      SIGNAL  sh : std_logic_vector(7 downto 0);
BEGIN
      PROCESS (clk)
            BEGIN
                  IF lr='0' THEN
                  sh<=p_in;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                ELSIF clk'event AND clk = '1' THEN
                s_out<=sh(0);
                sh(6 downto 0)<=sh(7 downto 1);
            END IF;
        END PROCESS;
END behavioral;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.8 บล็อกวงจร PISO ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

3.2.1.3 วงจรเปลี่ยน 1 บิต เป็น 8 บิต

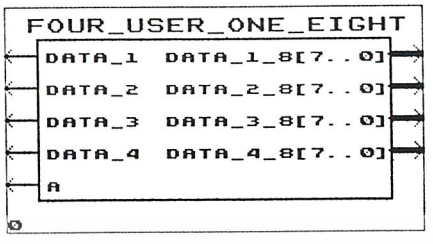
ก่อนที่จะนำเอาสัญญาณแต่ละบิต ของผู้ใช้งานแต่ละคนมาทำการเข้ารหัสของตัวเอง ซึ่งเป็นรหัส 8 บิต เราจะทำการแปลง 1 บิตของผู้ใช้งานไปเป็น 8 บิต เพื่อให้พอดีกับรหัสที่ใช้งาน ซึ่งวงจรแปลง 1 บิต ไปเป็น 8 บิต นี้สามารถบรรยายด้วย VHDL Code ได้ดังนี้

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

ENTITY four_user_one_eight IS
      PORT (data_1,data_2,data_3,data_4,a       : IN std_logic;
            data_1_8,data_2_8,data_3_8,data_4_8 : OUT
            std_logic_vector(7 DOWNTO 0));
END four_user_one_eight;


ARCHITECTURE behavior OF four_user_one_eight IS
SIGNAL a_1_8,a_2_8,a_3_8,a_4_8 : std_logic_vector(7 DOWNTO 0);
BEGIN
      PROCESS(a)
      BEGIN
            IF (a'event and a = '1') THEN
            a_1_8 <= data_1 & data_1 & data_1 & data_1 & data_1 &
data_1 & data_1 & data_1;
            a_2_8 <= data_2 & data_2 & data_2 & data_2 & data_2 &
data_2 & data_2 & data_2;
            a_3_8 <= data_3 & data_3 & data_3 & data_3 & data_3 &
data_3 & data_3 & data_3;
            a_4_8 <= data_4 & data_4 & data_4 & data_4 & data_4 &
data_4 & data_4 & data_4;
            END IF;
      END PROCESS;
      data_1_8<=a_1_8;
      data_2_8<=a_2_8;
      data_3_8<=a_3_8;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      data_4_8<=a_4_8;
END behavior;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.9 บล็อกวงจรวงจรเปลี่ยน 1 บิต เป็น 8 บิต ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

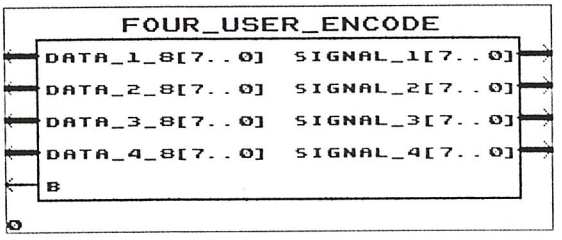
3.2.1.4 วงจรเข้ารหัส

ที่วงจรเข้ารหัสนี้เราจะนำเอาสัญญาณที่แปลงจาก 1 บิตเป็น 8 บิตแล้วของแต่ละผู้ใช้งานมาเข้ารหัสโดยจะนำเอาข้อมูลของตัวเองมาเข้ารหัสโดยการ X-OR (exclusive or) กับรหัสของตัวเองซึ่งวงจรเข้ารหัสนี้สามารถบรรยายด้วย VHDL Code ได้ดังนี้

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

ENTITY four_user_encode IS
      PORT (data_1_8,data_2_8,data_3_8,data_4_8 : IN
      std_logic_vector(7 DOWNTO 0);
      b                                         : IN std_logic;
      signal_1,signal_2,signal_3,signal_4       : OUT
      std_logic_vector(7 DOWNTO 0));
END four_user_encode;


ARCHITECTURE behavior OF four_user_encode IS
      SIGNAL      b_1,b_2,b_3,b_4   : std_logic_vector(7 DOWNTO 0);

BEGIN
      PROCESS(b)
      BEGIN
            IF (b'event and b = '1') THEN
                  b_1   <= data_1_8 XOR "01010101" ;
                  b_2   <= data_2_8 XOR "00110011" ;
                  b_3   <= data_3_8 XOR "01011010" ;
                  b_4   <= data_4_8 XOR "00111100" ;
            END IF;
      END PROCESS;
      signal_1<=b_1;
      signal_2<=b_2;
      signal_3<=b_3;
      signal_4<=b_4;
END behavior;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ซึ่งจะได้ component ดังนี้



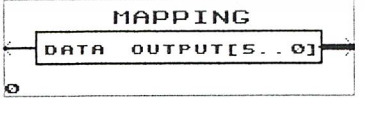
รูปที่ 3.10 บล็อกวงจรเข้ารหัสที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

3.2.1.5 วงจรแมปปิ้ง (mapping)

ก่อนที่เราจะบวกข้อมูลที่เข้ารหัสแล้วทั้ง 4 ชุดนี้เข้าด้วยกันนั้น เราต้องแปลงข้อมูลแต่ละบิตไปเป็นระดับโวลเตจเสมือน โดยแปลงข้อมูลบิต '0' ไปเป็น +1V ซึ่งแทนโดย "000001" และแปลงข้อมูลบิต '1' ไปเป็น -1V ซึ่งแทนโดย "111111" ซึ่งวงจรแมปปิ้ง นี้สามารถบรรยายด้วย VHDL code ได้ดังนี้

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY mapping IS
      PORT (  data      : IN std_logic;
            output : OUT std_logic_vector(5 DOWNTO 0));
END mapping;

ARCHITECTURE behavioral OF mapping IS
BEGIN
            output <= "111111" WHEN data='1' ELSE
                      "000001" WHEN data='0' ELSE
                      "000000";

END behavioral;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.11 บล็อกวงจรแมปปิ้ง ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

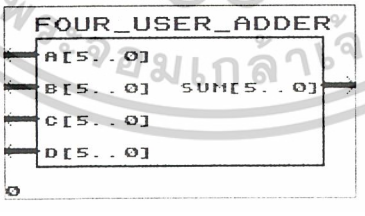
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.1.6 วงจรบวก (ADDER)

เราจะนำเอาข้อมูลที่แปลงเป็นระดับโวลเตจเสมือนของทั้ง 4 ผู้ใช้งานมาบวกกันที่วงจรบวก ซึ่งสัญญาณหลังวงจรบวก นี้จะได้สัญญาณ 8 ชุดที่แต่ละชุดมี 6 บิต โดยสัญญาณที่ได้จะมี 5 ระดับคือ $\pm 4V$, $\pm 2V$ และ $0V$ ข้อมูลทั้ง 8 ชุดนี้จะถูกส่งไปที่ด้านรับ ซึ่งวงจรadder นี้สามารถบรรยายด้วย VHDL Code ได้ดังนี้

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

entity      four_user_adder is
port  ( a,b,c,d  : in std_logic_vector(5 DOWNTO 0);
        sum      : out std_logic_vector(5 DOWNTO 0));
end;

architecture      strutural of four_user_adder is
COMPONENT   full_adder6bit
      port  (a,b : in std_logic_vector(5 DOWNTO 0);
             sum : out std_logic_vector(5 DOWNTO 0));
end COMPONENT;
signal s_1,s_2,s_3:std_logic_vector(5 DOWNTO 0);
begin
            box1 : full_adder6bit port map(a,b,s_1);
            box2 : full_adder6bit port map(c,d,s_2);
            box3 : full_adder6bit port map(s_1,s_2,s_3);

            sum <= s_3;
end strutural;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.12 บล็อกวงจรบวก ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

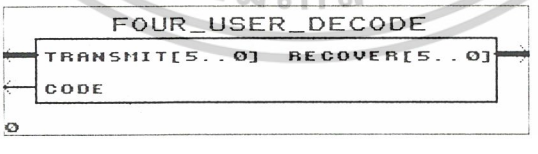
3.2.2 ด้านรับ

3.2.2.1 วงจรถอดรหัส

เมื่อสัญญาณที่รวมกันระหว่างผู้ใช้งานทั้ง 4 คน ซึ่งเป็นสัญญาณ 8 ชุด ชุดละ 6 บิตมาถึงด้านรับก็จะนำสัญญาณที่รับได้มาถอดรหัส โดยเริ่มที่วงจรถอดรหัส จะทำการคูณกันระหว่างสัญญาณที่รับมาได้กับรหัสที่ป้อนเข้าโดยในการทดลองนี้เราจะให้ป้อนรหัสของผู้ใช้งานโดยการปรับดิพสวิตช์ ถ้ารหัสตรงกับรหัสของผู้ใช้งานคนใดคนหนึ่ง ก็จะได้ข้อมูลของผู้ใช้งานคนนั้นๆ โดยรหัสแต่ละบิตนั้นเราจะแปลงให้เป็นระดับโวลเตจเสมือนเหมือนกัน ซึ่งการคูณกันจะเป็นการคูณแบบบิตที่มีเครื่องหมาย วงจรถอดรหัสนี้สามารถบรรยายด้วย VHDL Code ได้ดังนี้

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_arith.ALL;
USE ieee.std_logic_signed.ALL;

ENTITY four_user_decode IS
            PORT( transmit     : IN std_logic_vector(5 DOWNTO 0);
                  code         : IN std_logic;
                  recover      : OUT std_logic_vector(5 DOWNTO 0));
END four_user_decode;

ARCHITECTURE mixer OF four_user_decode IS
SIGNAL code_5 : std_logic_vector(5 DOWNTO 0);
COMPONENT two_comp
PORT (data        : IN std_logic;
      output      : OUT std_logic_vector(5 DOWNTO 0));
END COMPONENT;

BEGIN
      box1: two_comp port map(code,code_5);
      recover<=transmit * code_5;

END mixer;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.13 บล็อกวงจรถอดรหัสที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

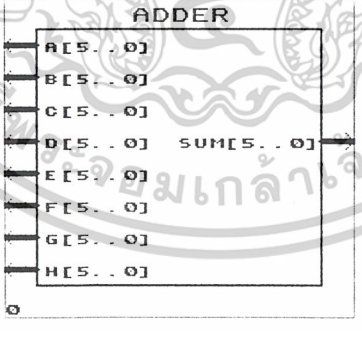
3.2.2.2 วงจรบวก

สัญญาณที่ออกจากวงจรถอดรหัส ซึ่งจะนำมาหาค่าเฉลี่ย จะนำมาบวกกันทั้ง 8 ค่าที่วงจรบวกสามารถบรรยายด้วย VHDL Code ได้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.std_logic_arith.ALL;
USE ieee.std_logic_signed.ALL;

ENTITY adder IS
PORT  (a,b,c,d,e,f,g,h  :IN std_logic_vector(5 DOWNTO 0);
       sum              :OUT std_logic_vector(5 DOWNTO 0));
END adder;

ARCHITECTURE structural OF adder IS
COMPONENT   full_adder6bit
      SIGNAL s_1,s_2,s_3,s_4,s_5,s_6,s_7:std_logic_vector(5 DOWNTO 0);
      PORT  (a,b : IN std_logic_vector(5 DOWNTO 0);
             sum : OUT std_logic_vector(5 DOWNTO 0));
      END COMPONENT;
BEGIN
             Box1:full_adder6bit port map(a,b,s_1);
             Box2:full_adder6bit port map(s_1,c,s_2);
             Box3:full_adder6bit port map(s_2,d,s_3);
             Box4:full_adder6bit port map(s_3,e,s_4);
             Box5:full_adder6bit port map(s_4,f,s_5);
             Box6:full_adder6bit port map(s_5,g,s_6);
             Box7:full_adder6bit port map(s_6,h,s_7);

             sum <= s_7;
END structural;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.14 บล็อกวงจรบวกที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

3.2.2.3 วงจรตัดสินค่า (DECISION)

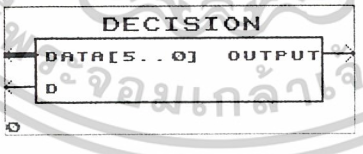
สัญญาณที่ออกจากวงจรบวกแล้ว จะมีอยู่ 3 ค่าคือ +8V (“001000”) ซึ่งเมื่อเฉลี่ย (หาร8) แล้วจะได้ +1V นั่นคือข้อมูลบิต ‘0’, -8V (“111000”) ซึ่งเมื่อเฉลี่ยแล้วจะได้ -1V นั่นคือข้อมูลบิต ‘1’ นั่นเอง และ 0V จะไม่ได้สัญญาณอะไรออกมานั้นคือเป็นบิต ‘0’ ตลอด ดังนั้นเมื่อรู้ว่าอินพุตของวงจรตัดสินคือ +8V

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หรือ -8V เราก็ทำการตัดสินว่าข้อมูลคือบิต '0' และบิต '1' ตามลำดับ วงจรตัดสินนี้สามารถบรรยายด้วย VHDL Code ได้ดังนี้

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

ENTITY decision IS
            PORT( data          : IN std_logic_vector(5 DOWNTO 0);
                  d             : IN std_logic;
                  output        : OUT std_ulogic);
END decision;

ARCHITECTURE behavioral OF decision IS
      SIGNAL xxx : std_ulogic;
BEGIN
            PROCESS (data,e)
            BEGIN
                  IF (d'event and d='1') THEN
                        IF (data = "000000") THEN
                              xxx <= 'X';
                        ELSIF (data = "111000") THEN
                              xxx <= '1';
                        ELSIF (data = "001000") THEN
                              xxx <= '0';
                        END IF;
                  END IF;
            END PROCESS;
      output <= xxx;
END behavioral;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.15 บล็อกวงจรตัดสินที่ได้จากการ synthesis จากโปรแกรม MAX+plus

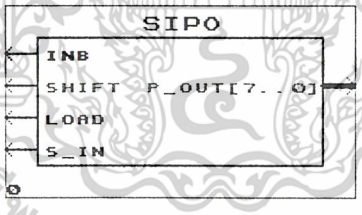
3.2.2.4 วงจรซีเรียลอินพาราแลล เอ้าท์ (Serial in Parallel out)

หลังจากที่ได้ข้อมูลเดิมกลับมาแล้ว จะนำเข้าไปที่วงจร ซีเรียลอินพาราแลล เอ้าท์ เพื่อให้สัญญาณที่ออกจากชิปเป็นสัญญาณขนาน 8 บิต เพื่อนำเข้าสู่วงจร D/A เพื่อให้กลับเป็นสัญญาณซานย์ดังเดิมซึ่งวงจรสามารถบรรยายด้วย VHDL Code ได้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
library ieee;
Use ieee.std_logic_1164.ALL;
Use ieee.std_logic_Unsigned.ALL;
Entity sipo is
  port( inb,shift,load : in  std_logic;
          s_in             : in  std_logic;
          p_out            : out std_logic_vector(7 downto 0));
  End sipo;

Architecture behavior of sipo is
  Signal  sh : std_logic_vector(7 downto 0);
   Begin
     Process (inb,shift,load)
      Begin
            if inb = '1' then
               sh(7)<=s_in;
            end if;
            if  shift'event and shift = '1' then
                  sh(6 downto 0)<=sh(7 downto 1);
            end if;
            if load='0' then
                     p_out<=sh;
            end if;
     end process;
end behavior;
```

ซึ่งจะได้ component ดังนี้



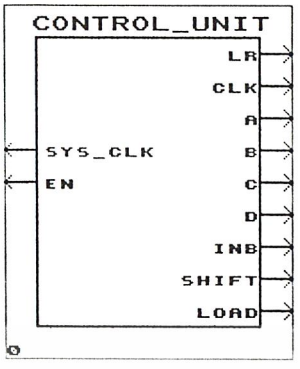
รูปที่ 3.16 บล็อกวงจร SIPO ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

3.2.2.5 วงจรหน่วยควบคุม (control unit)

เนื่องจากในแต่ละส่วนต้องมีการทำงานที่สัมพันธ์กัน ดังนั้นจำเป็นที่ต้องมีวงจรหน่วยควบคุม เพื่อให้ได้วงจรทั้งหมดที่ทำงานสัมพันธ์กันและสัมพันธ์กันกับอินพุตด้วยซึ่งสามารถดูการอธิบายด้วยภาษา VHDL ได้ที่ ภาคผนวก ง.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ซึ่งจะได้ component ดังนี้



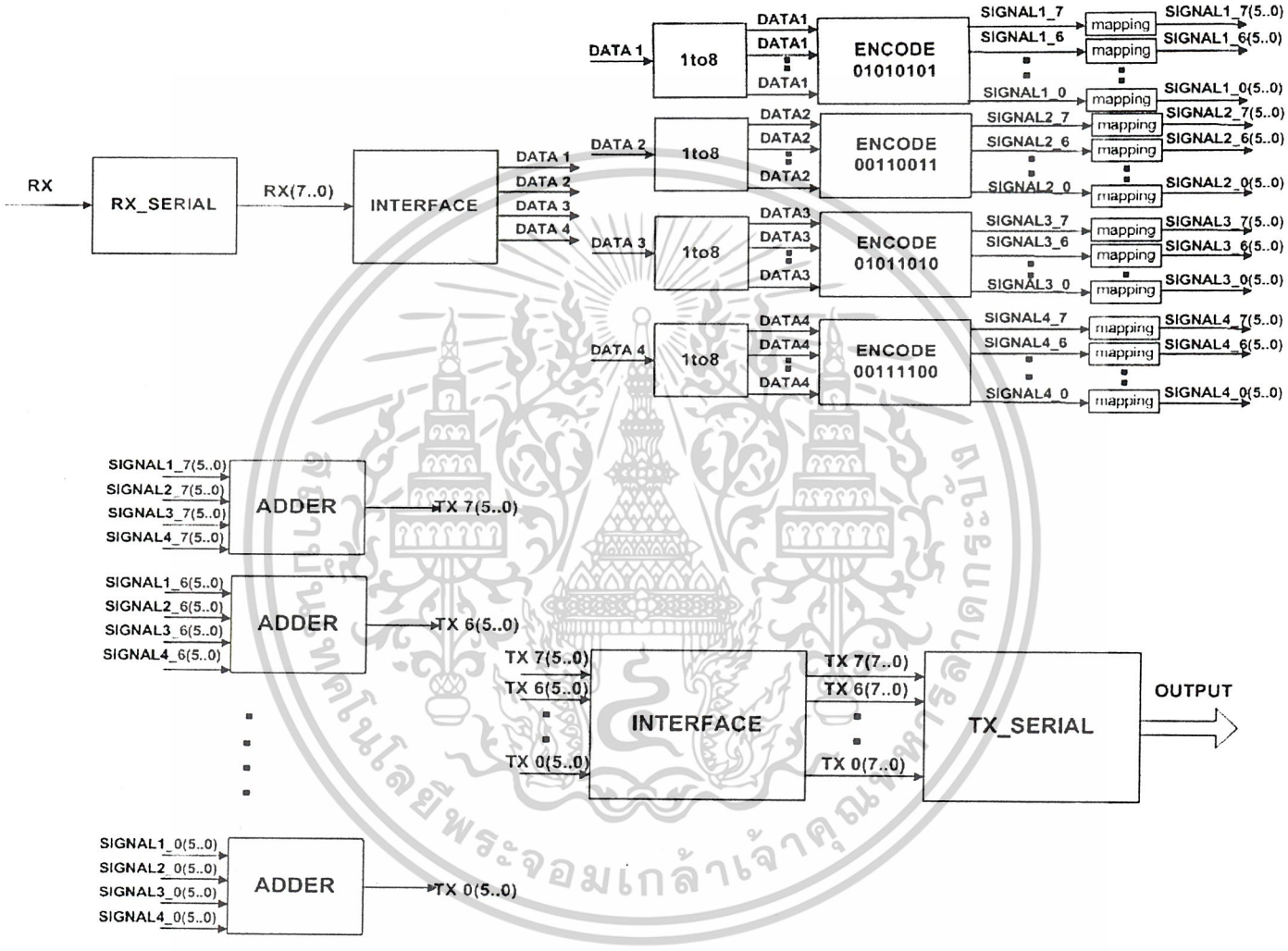
รูปที่ 3.17 บล็อกวงจรหน่วยควบคุมที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

3.2.3 ระบบแบบหลายผู้ใช้ผ่านพอร์ทอนุกรม

การทดลองนี้เราจะให้ด้านส่งและด้านรับอยู่แยกกันโดยด้านส่งอยู่ในชิป FPGA ซึ่งจะทำการรับข้อมูลจากคอมพิวเตอร์โดยผ่านทางพอร์ทอนุกรมที่กำหนดค่าความเร็วบอดเรต (baud rate) เท่ากับ 4800 และเมื่อประมวลผลทางด้านส่งเสร็จก็จะส่งข้อมูลผ่านทางพอร์ทอนุกรมไปให้ทางด้านรับซึ่งอยู่ในคอมพิวเตอร์โดยทางด้านรับจะใช้โปรแกรม MATLAB ในการประมวลผล

โดยด้านส่งมีวงจรรวมดังรูปที่ 3.18

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.18 บล็อกไดอะแกรมด้านส่งของวงจรดิจิตอลที่ออกแบบ

โดยมีวงจรที่ต่างจากการทดลองที่ 3.2.1 ดังนี้

3.2.3.1 วงจรรับข้อมูลจากพอร์ทอนุกรม (Rx-serial)

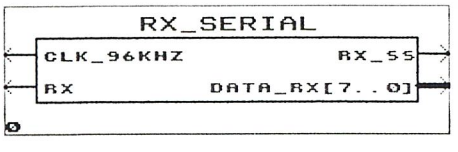
วงจรรับข้อมูลจากพอร์ทอนุกรมนี้จะทำการรับข้อมูลจากพอร์ทอนุกรมที่ส่งออกมาจากคอมพิวเตอร์โดยข้อมูลที่ออกจากวงจรนี้เราจะได้ข้อมูล 8 บิตซึ่งวงจรรับข้อมูลจากพอร์ทอนุกรมนี้สามารถบรรยาย ด้วย VHDL Code ได้ดังนี้

```
library ieee;
use ieee.std_logic_1164.ALL;
Use ieee.std_logic_Unsigned.ALL;

Entity rx_serial is
Port( clk_48kHZ   : in  std_logic;
      RX          : in  std_logic;
      RX_SS       : out std_logic;
      DATA_RX     : out std_logic_vector(7 downto 0));
End;

Architecture rtl of rx_serial is
signal State_RX : integer range 0 to 7 := 0;
Begin
      Process(clk_48kHZ,RX)
            variable RX_Data_Count : integer range 0 to 7;
            variable Buffer_RX     : std_logic_vector(7 downto 0);
            variable wait_count    : integer range 0 to 31;
            variable acc_var       : std_logic_vector(3 downto 0);
            Begin
            if clk_48kHZ'Event and clk_48kHZ = '1' then
                  CASE State_RX is
                  WHEN 0 =>   RX_SS <= '0';
                              wait_count:=0;
                              acc_var:=(others=>'0');
                              Buffer_RX:=(others=>'0');
                                    if RX = '0' then
                                          RX_Data_Count := 0;
                                          State_RX <= 1;
                                          else
                                          RX_Data_Count := 0;
                                          State_RX <= 0;
                                          end if;

                  WHEN 1 =>   if wait_count = 9 then
                                    wait_count := 0;
                                    State_Rx <= 2;
                              else

      wait_count:=wait_count+1;
                              end if;

                  WHEN 2 =>   if wait_count = 9 then
                                    State_Rx <= 3;
                                    wait_count := 1;
                                    acc_var:=acc_var+RX;
                              else
                                    wait_count:=wait_count+1;
                                    acc_var:=acc_var+RX;
                              end if;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                WHEN 3 =>   if RX_Data_Count = 7 then
                                  RX_SS <= '0';
                                  State_Rx <= 4;
                            if acc_var >= "101" then
                                  Buffer_RX(RX_Data_Count) := '1';
                                  acc_var:=(others=>'0');
                            else
                                  Buffer_RX(RX_Data_Count) := '0'
                                  acc_var:=(others=>'0');
                            end if;
                            RX_Data_Count:=RX_Data_Count+1;
                      else
                            State_Rx <= 2;
                      if acc_var >= "101" then
                            Buffer_RX(RX_Data_Count) := '1';
                            acc_var:=(others=>'0');
                      else
                            Buffer_RX(RX_Data_Count) := '0'
                            acc_var:=(others=>'0');
                      end if;
                            RX_Data_Count:=RX_Data_Count+1;
                      end if;

                WHEN 4 =>   if  wait_count=22 then
                                  RX_Data_Count := 0;
                                  DATA_RX <= Buffer_RX;
                                  RX_SS <= '1';
                                  State_RX <= 0;
                            else
                                  RX_Data_Count := 0;
                                  DATA_RX <= Buffer_RX;
                                  RX_SS <= '1';
                                  State_RX <= 4;
                                  wait_count:=wait_count+1;
                            end if;
                 WHEN others =>
                            RX_Data_Count := 0;
                            RX_SS <= '0';
                            State_RX <= 0;
                 end case;
          end if;
     end process;
end;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.19 บล็อกวงจรรับข้อมูลจากพอร์ทอนุกรม ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

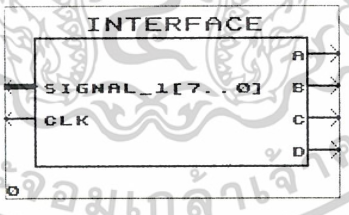
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.3.2 วงจรอินเทอร์เฟส 1 (interface 1)

เนื่องจากการประมวลผลของระบบจะประมวลผลที่ละ 4 บิต แต่ข้อมูลที่ได้จากพอร์ทอนุกรมจะได้ทีละ 8 บิตดังนั้นเราจะเลือกใช้เฉพาะ 4 บิตโดยใช้บิตที่ 7(MSB), 6, 5, 4 ซึ่งวงจรอินเทอร์เฟส 1 นี้สามารถบรรยายด้วย VHDL Code ได้ดังนี้

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

ENTITY interface IS
      PORT (signal_1     : IN std_logic_vector(7 DOWNTO 0);
            clk          : IN std_logic;
            a,b,c,d      : OUT std_logic);
END;

ARCHITECTURE behavior OF interface IS
BEGIN
      process(clk)
      BEGIN
            IF (clk'event and clk = '1') THEN
                  a<=signal_1(7);
                  b<=signal_1(6);
                  c<=signal_1(5);
                  d<=signal_1(4);
            END IF;
      END process;
END;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.20 บล็อกวงจรอินเทอร์เฟส 1 ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

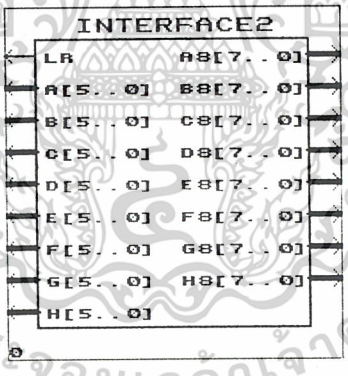
3.2.3.3 วงจรอินเทอร์เฟส 2 (interface 2)

เมื่อได้อินพุตเป็นข้อมูลทีละ 4 บิตแล้วก็ประมวณผลเหมือนการทดลองที่ 3.2.1 แล้วเมื่อได้สัญญาณที่จะส่งออกเป็นข้อมูล 8 ชุด ชุดละ 6 บิตแล้วเราจะส่งไปทางพอร์ทอนุกรมไปให้ทางด้านรับซึ่งอยู่ในคอมพิวเตอร์ ซึ่งการส่งผ่านทางพอร์ทอนุกรมจะส่งทีละ 8 บิตดังนั้นเราจึงต้องแปลงข้อมูลชุดละ 6 บิตเป็นชุดละ 8 บิตเสียก่อนโดยแปลงที่วงจรนี้ ซึ่งวงจรอินเทอร์เฟส 2 นี้สามารถบรรยายด้วย VHDL Code ได้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
library ieee;
Use ieee.std_logic 1164.ALL;
Use ieee.std_logic_Unsigned.ALL;
Entity interface2 is
  port( lr : in  std_logic;
        a,b,c,d,e,f,g,h   : in  std_logic_vector(5 downto 0);
        a8,b8,c8,d8,e8,f8,g8,h8:out std_logic_vector(7 downto 0));
  End;

Architecture rtl of interface2 is
Begin
     Process (lr)
      Begin
        if lr='0' then
                  a8 <=a&"00";
            b8<=b&"00";
            c8<=c&"00";
            d8<=d&"00";
            e8<=e&"00";
            f8<=f&"00";
            g8<=g&"00";
            h8<=h&"00";
        end if;
     end process;
end;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.21บล็อกวงจรอินเทอร์เฟส 2 ที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

3.2.3.4 วงจรส่งข้อมูลออกพอร์ทอนุกรม (Tx-serial)

เมื่อได้ข้อมูลที่จะส่งออกพอร์ทขนานทีละ 8 บิตจำนวน 8 ชุดแล้วก็จะทำการส่งออพอร์ทอนุกรมทีละชุด โดยวงจรนี้จะทำการเพิ่มสตาร์ทบิต 1 บิตซึ่งเป็นบิต '0' และสต๊อปบิตอีก 1 บิตซึ่งเป็นบิต '1' โดยวงจรส่งข้อมูลออกพอร์ทอนุกรม นี้สามารถบรรยายด้วย VHDL Code ได้ดังนี้

```
library ieee;
Use ieee.std_logic_1164.ALL;
Use ieee.std_logic_unsigned.ALL;
Entity Tx_serial is
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      port( BAUD_RATE    : in std_logic;
            TX_EN        : in std_logic;
            DATA_1TX,DATA_2TX,DATA_3TX,DATA_4TX,DATA_5TX,DATA_6TX,
            DATA_7TX,DATA_8TX : in std_logic_vector(7 downto 0);
            TX                : out std_logic  );
end;

Architecture rtl of Tx_serial is
             type State_Type_
            (Idel1a,Start1a,TransData1a,Stop1a,Idel2a,Start2a,TransDa
            ta2a,Stop2a,Idel3a,Start3a,TransData3a,Stop3a,Idel4a,Star
            t4a,TransData4a,Stop4a,Idel5a,Start5a,TransData5a,Stop5a,
            Idel6a,Start6a,TransData6a,Stop6a,Idel7a,Start7a,TransDat
            a7a,Stop7a,Idel8a,Start8a,TransData8a,Stop8a);
      signal State_TX : State_Type_TX;

begin
process(BAUD_RATE,TX_EN,DATA_1TX,DATA_2TX,DATA_3TX,DATA_4TX,DATA_5TX,
DATA_6TX,DATA_7TX,DATA_8TX)
            variable TX_Data_Count : integer range 0 to 7 := 0;
            begin
        if TX_EN = '1' then
           State_Tx<=Start1a;
           elsif BAUD_RATE'Event and BAUD_RATE = '1' then
                 case State_TX is
                 when Idel1a =>
                       TX <= '1';
                       TX_Data_Count := 0;
                       State_TX <= Idel1a;
                 when Start1a =>
                       TX <= '0';
                       TX_Data_Count := 0;
                       State_TX <= TransData1a;
                 when TransData1a =>
                       if TX_Data_Count = 7 then
                             TX <= DATA_1TX(TX_Data_Count);
                             State_TX <= Stop1a;
                       else
                             TX <= DATA_1TX(TX_Data_Count);
                             TX_Data_Count := TX_Data_Count + 1;
                             State_TX <= TransData1a;
                       end if;
                 when Stop1a =>
                       TX <= '1';
                       TX_Data_Count := 0;
                       State_TX <= Idel2a;
                 when Idel2a =>
                       TX <= '1';
                       TX_Data_Count := 0;
                       State_TX <= Start2a;
                 when Start2a =>
                       TX <= '0';
                       TX_Data_Count := 0;
                       State_TX <= TransData2a;
                 when TransData2a =>
                       if TX_Data_Count = 7 then
                             TX <= DATA_2TX(TX_Data_Count);
                             State_TX <= Stop2a;
                       else
                             TX <= DATA_2TX(TX_Data_Count);
                             TX_Data_Count := TX_Data_Count + 1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                State_TX <= TransData2a;
        end if;
when Stop2a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Idel3a;
when Idel3a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Start3a;
when Start3a =>
        TX <= '0';
        TX_Data_Count := 0;
        State_TX <= TransData3a;
when TransData3a =>
        if TX_Data_Count = 7 then
                TX <= DATA_3TX(TX_Data_Count);
                State_TX <= Stop3a;
        else
                TX <= DATA_3TX(TX_Data_Count);
                TX_Data_Count := TX_Data_Count + 1;
                State_TX <= TransData3a;
        end if;
when Stop3a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Idel4a;
when Idel4a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Start4a;
when Start4a =>
        TX <= '0';
        TX_Data_Count := 0;
        State_TX <= TransData4a;
when TransData4a =>
        if TX_Data_Count = 7 then
                TX <= DATA_4TX(TX_Data_Count);
                State_TX <= Stop4a;
        else
                TX <= DATA_4TX(TX_Data_Count);
                TX_Data_Count := TX_Data_Count + 1;
                State_TX <= TransData4a;
        end if;
when Stop4a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Idel5a;
when Idel5a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Start5a;
when Start5a =>
        TX <= '0';
        TX_Data_Count := 0;
        State_TX <= TransData5a;
when TransData5a =>
        if TX_Data_Count = 7 then
                TX <= DATA_5TX(TX_Data_Count);
                State_TX <= Stop5a;
        else
```

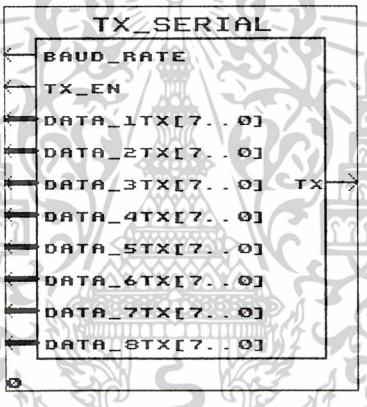
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                TX <= DATA_5TX(TX_Data_Count);
                TX_Data_Count := TX_Data_Count + 1;
                 State_TX <= TransData5a;
        end if;
when Stop5a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Idel6a;
when Idel6a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Start6a;
when Start6a =>
        TX <= '0';
        TX_Data_Count := 0;
        State_TX <= TransData6a;
when TransData6a =>
        if TX_Data_Count = 7 then
                TX <= DATA_6TX(TX_Data_Count);
                State_TX <= Stop6a;
        else
                TX <= DATA_6TX(TX_Data_Count);
                TX_Data_Count := TX_Data_Count + 1;
                State_TX <= TransData6a;
        end if;
when Stop6a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Idel7a;
when Idel7a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Start7a;
when Start7a =>
        TX <= '0';
        TX_Data_Count := 0;
        State_TX <= TransData7a;
when TransData7a =>
        if TX_Data_Count = 7 then
                TX <= DATA_7TX(TX_Data_Count);
                State_TX <= Stop7a;
        else
                TX <= DATA_7TX(TX_Data_Count);
                TX_Data_Count := TX_Data_Count + 1;
                State_TX <= TransData7a;
        end if;
when Stop7a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Idel8a;
when Idel8a =>
        TX <= '1';
        TX_Data_Count := 0;
        State_TX <= Start8a;
when Start8a =>
        TX <= '0';
        TX_Data_Count := 0;
        State_TX <= TransData8a;
when TransData8a =>
        if TX_Data_Count = 7 then
                TX <= DATA_8TX(TX_Data_Count);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        State_TX <= Stop8a;
                  else
                        TX <= DATA_8TX(TX_Data_Count);
                        TX_Data_Count := TX_Data_Count + 1;
                        State_TX <= TransData8a;
                  end if;
            when Stop8a =>
                  TX <= '1';
                  TX_Data_Count := 0;
                  State_TX <= Idel1a;
            when others =>
                  TX_Data_Count := 0;
                  TX <= '1';
                  State_TX <= Idel1a;
            end case;
end if;
end process;
end;
```

ซึ่งจะได้ component ดังนี้



รูปที่ 3.22 บล็อกวงจรส่งข้อมูลออกพอร์ทอนุกรมที่ได้จากการ synthesis จากโปรแกรม MAX+plus II

3.2.3.5 ส่วนส่งข้อมูลและรับสัญญาณผ่านพอร์ทอนุกรม

ส่วนนี้จะทำการส่ง และรับข้อมูลโดยใช้โปรแกรมคอมพิวเตอร์ MATLAB ผ่านทางพอร์ทอนุกรม กำหนดค่าความเร็วบอดเรต (baudrate) เท่ากับ 4800 ที่เชื่อมต่ออยู่กับอุปกรณ์ FPGA และเมื่อประมวลผลทางด้านส่งเสร็จก็จะส่งข้อมูลกลับผ่านทางพอร์ทอนุกรมไปให้ทางด้านรับซึ่งอยู่ในคอมพิวเตอร์โดยทางด้านรับจะใช้โปรแกรม MATLAB ในการประมวลผล

ในการส่งส่งข้อมูลจะต้องส่งพร้อมกัน 4 ผู้ใช้ ที่จะทำการพิจารณาในระบบ ดังนั้นจึงต้องแปลงการลำดับข้อมูลให้เป็นไปตามมาตรฐานของพอร์ทอนุกรม และทำการรับสัญญาณที่ส่งกลับมาจากอุปกรณ์ FPGA ที่มีการสเปรดสเปคตรัม มาทำการประมวลผลเพื่อให้ได้ข้อมูลเดิมต่อไป ดังโปรแกรมต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
clc;
clear all;
w = [1 -1  1 -1  1 -1  1 -1;
     1  1 -1 -1  1  1 -1 -1;
     1 -1  1 -1 -1  1 -1  1;
     1  1 -1 -1 -1 -1  1  1];

s = serial('COM1');
set(s,'BaudRate',4800);
set(s,'Databits',8);
fopen(s);
%%user 1 2 3 4
%data=[1 1 1 1 0 0 0 0;
%      1 1 0 0 0 0 0 0;
%      1 0 1 0 0 0 0 0;
%      1 0 0 1 0 0 0 0];
data=[240;192;160;144]
sd=size(data);
for i=1:sd(1)
        fwrite(s,data(i));%%% Tx serial
        x=fread(s,8);%%% Rx serial
        dec2bin(x,8)
    for j=1:8
   %%%binary signed %%%
                if x(j)==16
                    Rx_hat(j)=4;
                else
                    if x(j)==8
                        Rx_hat(j)=2;
                    else
                        if x(j)==248
                            Rx_hat(j)=-2;
                        else
                            if x(j)==240
                                Rx_hat(j)=-4;
                            else
                                Rx_hat(j)=0;
                            end
                        end
                    end
                end

    end
            decode=0;
            for j=1:8                !!!
                detect=Rx_hat(j)*w( ,j);
                decode=decode+detect;
            end
                if decode>=0
                    dat_in(i)=0;
                else
                    dat_in(i)=1;
                end
end

                dat_in
fclose(s)
delete(s)
instrreset
```

สามารถเลือกผู้ใช้ที่จะแสดงผลได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

การทดลองและผลการทดลอง

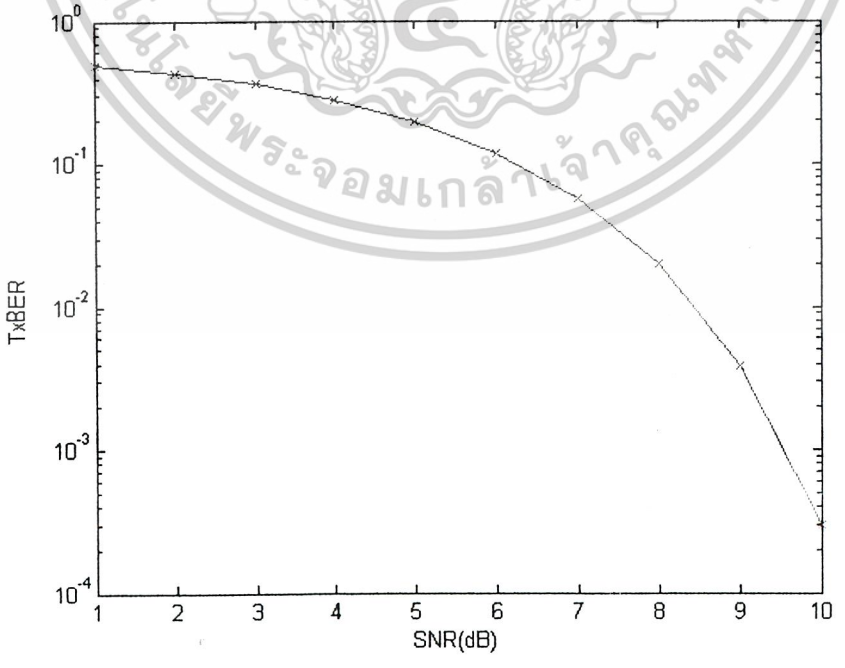
4.1 ผลการทดลองที่ได้จากโปรแกรม MATLAB

4.1.1 ผลการทดลองสมรรถภาพของระบบ DS-CDMA

บทนี้จะนำเสนอการทดลองและผลการทดลองเชิงเลขโดยใช้โปรแกรม MATLAB ในการจำลองการทำงาน โปรแกรมที่ใช้ในการทดลองนี้จะออกแบบโดยอ้างอิงทฤษฎีของระบบ DS-CDMA ดังที่ได้นำเสนอไว้แล้วในบทที่ 2 ซึ่งจะใช้การเข้ารหัสช่องสัญญาณแบบของรหัสวอล์ช โดยได้โปรแกรมที่ใช้ในการทดลองแสดงในภาคผนวก ค.

ส่วนผลการทดลองเชิงเลขเมื่อใช้โปรแกรม MATLAB ในการจำลองการทำงานของการส่งภาพข้อมูล 2 ภาพพร้อมกันและศึกษาอัตราการผิดพลาดบิตของระบบ DS-CDMA ในรายงานฉบับนี้ได้ศึกษาผลของค่า SNR ซึ่งมีผลต่อค่าสมรรถนะของระบบที่มีขั้นตอนตามบทที่ 3 โดยทำการเปลี่ยนแปลงค่า SNR แล้วสังเกตผลของอัตราความผิดพลาดบิต ซึ่งได้ทำการศึกษาในสองแบบ คือ อัตราความผิดพลาดของบิตข้อมูล และ อัตราความผิดพลาดของบิตสัญญาณ

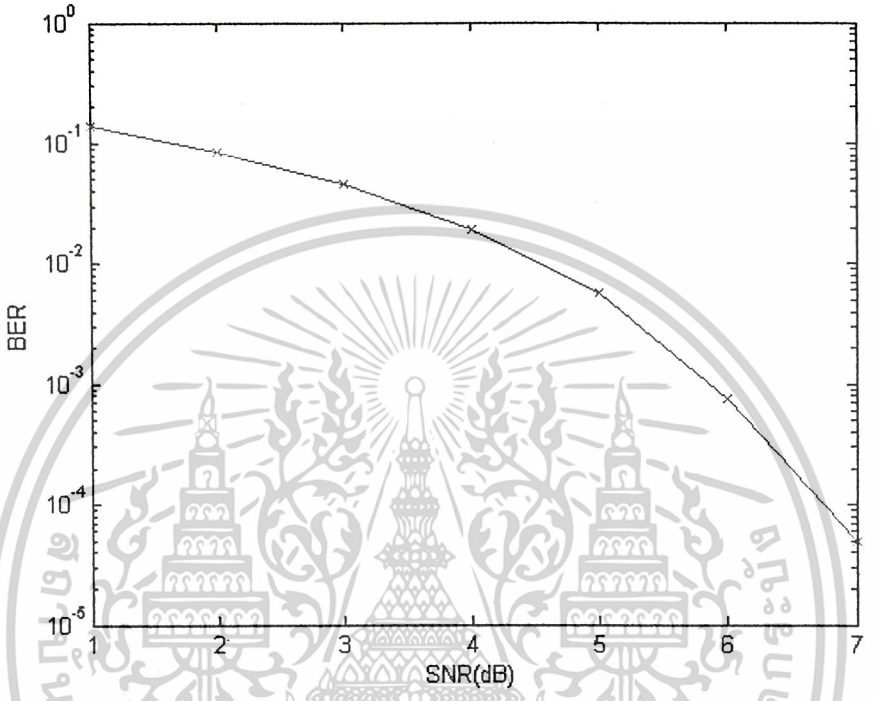
กราฟแสดงอัตราการผิดพลาดบิตสัญญาณกับพลังงานสัญญาณต่อสัญญาณรบกวนดังรูปที่ 4.1 ได้มาจากการนำภาพข้อมูลทั้ง 2 ภาพมาทำการถอดรหัสให้อยู่ในรูปบิตเดี่ยวแล้วนำข้อมูลที่ได้ไปเข้าระบบ DS-CDMA ตามลำดับดังต่อไปนี้ เข้ารหัสสัญญาณกับรหัสวอล์ช, รวมสัญญาณที่จะทำการส่ง, เพิ่มสัญญาณรบกวนระหว่างการส่ง โดยที่สัญญาณรบกวนนั้นมีการเปลี่ยนแปลงที่แตกต่างกันด้วนการกำหนดค่า SNR และเมื่อทำการเปรียบเทียบบิตสัญญาณที่ได้กับบิตสัญญาณที่ส่งพร้อมนับจำนวนบิตที่แตกต่างกันแล้ว
นำมาแสดงเป็นกราฟดังรูปที่ 4.1



รูปที่ 4.1 อัตราความผิดพลาดของบิตสัญญาณกับพลังงานสัญญาณต่อสัญญาณรบกวน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อทางด้านรับจะทำการแยกสัญญาณโดยใช้รหัสวอล์ชของแต่ละข้อมูลภาพ นำสัญญาณที่ได้มาทำการอินทิเกรทและเปรียบเทียบระดับสัญญาณเพื่อให้ได้ค่าของบิตข้อมูลแล้วทำการเปรียบเทียบกับข้อมูลเริ่มต้น ทำการนับจำนวนบิตข้อมูลที่ผิดพลาดนำมาแสดงเป็นกราฟอัตราการผิดพลาดของบิตข้อมูลกับพลังงานสัญญาณต่อสัญญาณรบกวนดังรูปที่ 4.2

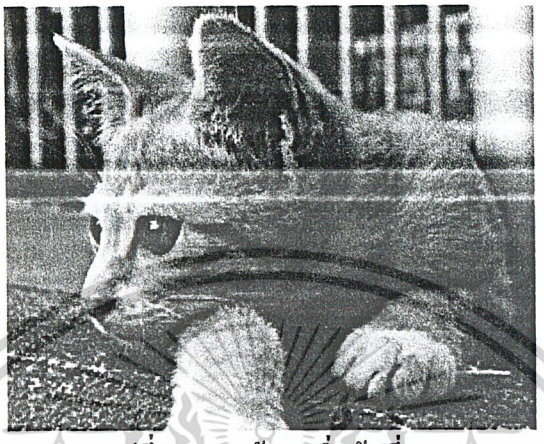


รูปที่ 4.2 อัตราความผิดพลาดของบิตข้อมูลกับพลังงานสัญญาณต่อสัญญาณรบกวน

จากกราฟในรูปที่ 4.2 จะเห็นได้ชัดว่าอัตราความผิดพลาดของบิตข้อมูลกับพลังงานสัญญาณต่อสัญญาณรบกวน จะมีอัตราการเปลี่ยนแปลงของค่า BER ที่ลดลงมากกว่า อัตราความผิดพลาดของบิตสัญญาณกับพลังงานสัญญาณต่อสัญญาณรบกวน ซึ่งเป็นผลจากการเข้ารหัสช่องสัญญาณวอล์ช

รูปที่ 4.3 และ 4.7 แสดงภาพข้อมูลเริ่มต้นทั้ง 2 ภาพ ที่มีขนาด 200x250 พิกเซล เมื่อนำสัญญาณข้อมูลที่ได้จากการเข้าระบบ DS-CDMA และนำสัญญาณที่ผ่านระบบมาถอดเข้ารหัสกลับคืนเป็นรูปภาพได้ดังรูปที่ 4.4 - 4.6 และรูปที่ 4.8 - 4.10

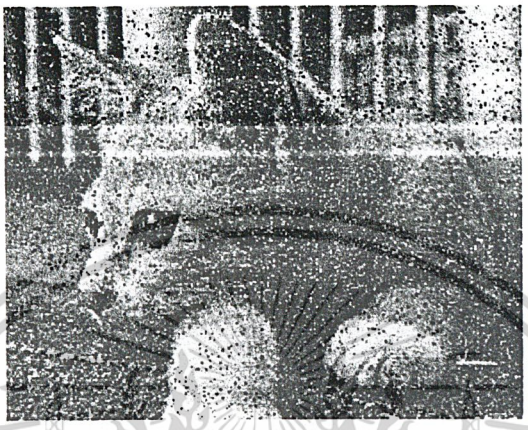
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.3 ภาพข้อมูลเริ่มต้นที่ 1

รูปที่ 4.4 ภาพข้อมูลที่ 1 หลังผ่านระบบ CDMA เมื่อ SNR=1

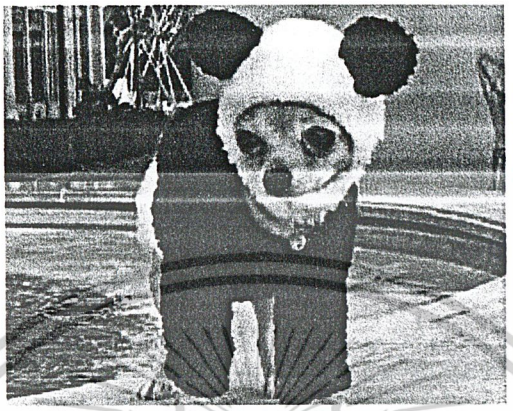
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



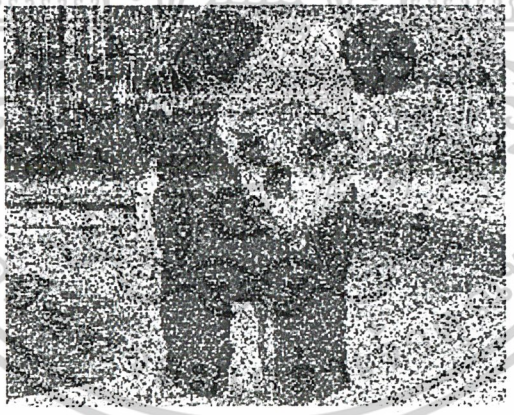
รูปที่ 4.5 ภาพข้อมูลที่ 1 หลังผ่านระบบ CDMA เมื่อ SNR=6

รูปที่ 4.6 ภาพข้อมูลที่ 1 หลังผ่านระบบ CDMA เมื่อ SNR=12

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.7 ภาพข้อมูลเริ่มต้นที่ 2



รูปที่ 4.8 ภาพข้อมูลที่ 2 หลังผ่านระบบ CDMA เมื่อ SNR=1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

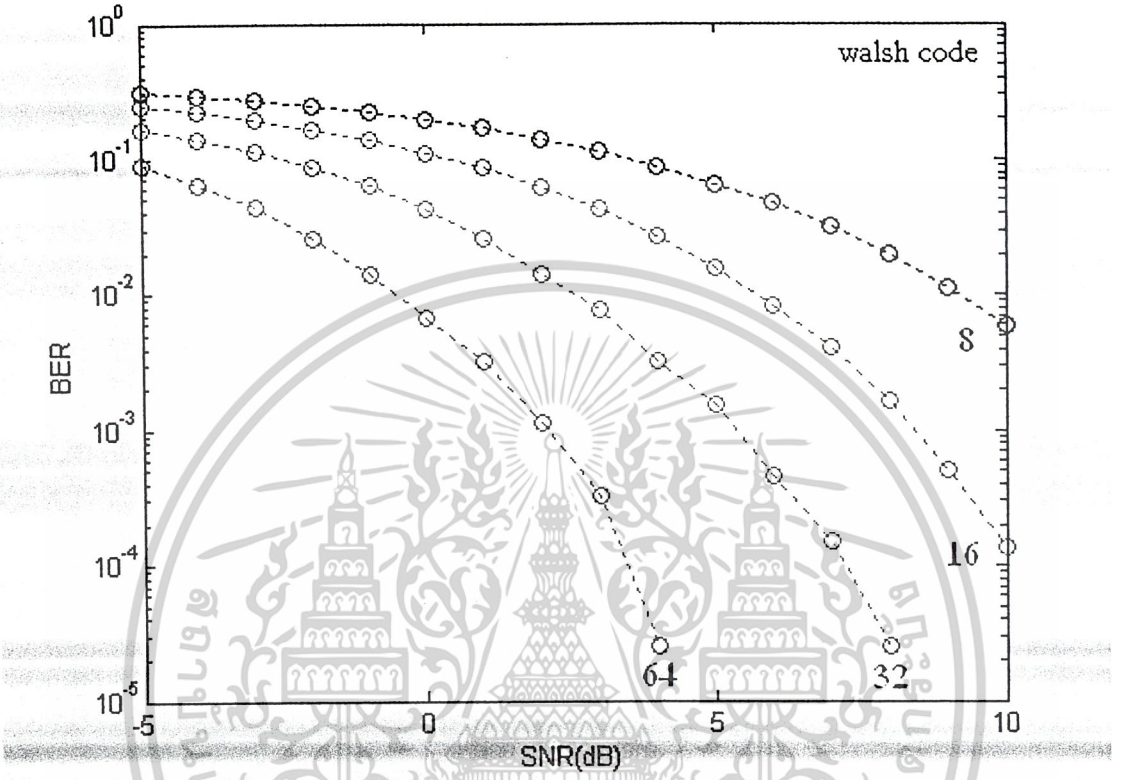
รูปที่ 4.9 ภาพข้อมูลที่ 2 หลังผ่านระบบ CDMA เมื่อ SNR=6

รูปที่ 4.10 ภาพข้อมูลที่ 2 หลังผ่านระบบ CDMA เมื่อ SNR=12

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.2 ผลการทดลองสมรรถภาพเมื่อขนาดของรหัสเพิ่ม

จากการทดลองได้ทำการศึกษาทั้งอัตราความผิดพลาดบิตสัญญาณและอัตราความผิดพลาดบิตข้อมูลซึ่งอัตราความผิดพลาดบิตข้อมูลจะลดลงอย่างชัดเจนเมื่อเปลี่ยนขนาดของรหัสวอล์ช



รูปที่ 4.11 อัตราความผิดพลาดของบิตข้อมูลกับพลังงานสัญญาณต่อสัญญาณรบกวน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.3 ผลการวิเคราะห์ทดลองการซิงโครไนซ์

จากการทดลองการซิงโครไนซ์ของระบบสามารถแสดงผลและค่าตัวแปรต่างๆดังนี้

```
data =
  0   0   0   0   0   0   0   0
  1   1   1   1   1   1   1   1
  1   1   1   1   1   1   1   1

t =
  0   0   0   0   0   0   0   0
  1   1  -1  -1   1   1  -1   1
  1  -1  -1   1   1  -1  -1   1


Rx_hat =
  1   0  -1   0   1   0  -1   0
```

```
auto =                  ค่าอัตสหสัมพันธ์ที่ระยะต่างๆ
   4   4                0 bit
   4  -4                1 bit
  -4  -4                2 bit
  -4   4                3 bit
   4   4                4 bit
   4  -4                5 bit
  -4  -4                6 bit
  -4   4                7 bit
   4   4                8 bit
```

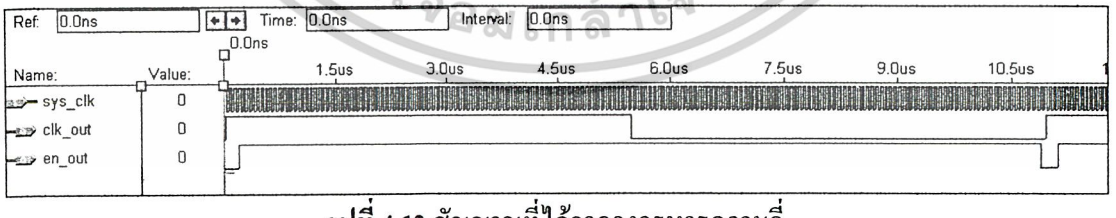
4.2 ผลการทดลองที่ได้จากโปรแกรม MAX+plus II

4.2.1 ผลการทดลองระบบแบบหลายผู้ใช้

4.2.1.1 ด้านส่ง

4.2.1.1.1 วงจรหารความถี่

วงจรหารความถี่นี้ เป็นการหารความถี่ของสัญญาณนาฬิกา 16.257 MHz ด้วย 182 เพื่อให้เหลือความถี่ 90 kHz เพื่อใช้ในการแซมปลิง ซึ่งผลจากการจำลองการทำงาน จะได้รูปที่ 4.12

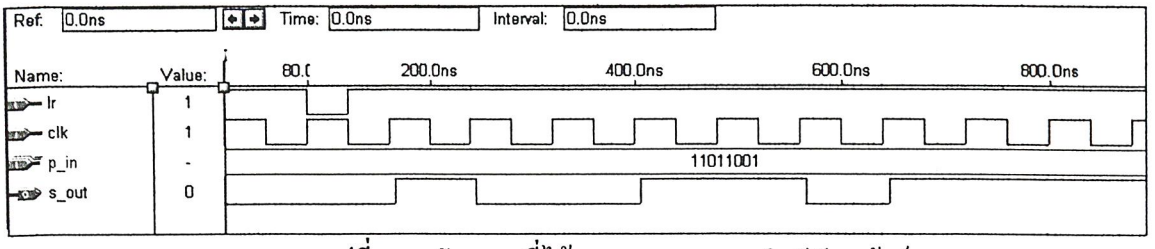


รูปที่ 4.12 สัญญาณที่ได้จากวงจรหารความถี่

4.2.1.1.2 วงจรพาราแลลอินซีเรียล เอ้าท์

วงจรจะโหลดข้อมูล (p_in) เมื่อบิต lr มีค่าเป็น '0' และจะเลื่อนบิต LSB ออกทุกขอบขาขึ้นของสัญญาณ clk ซึ่งผลจากการจำลองการทำงาน จะได้รูปที่ 4.13

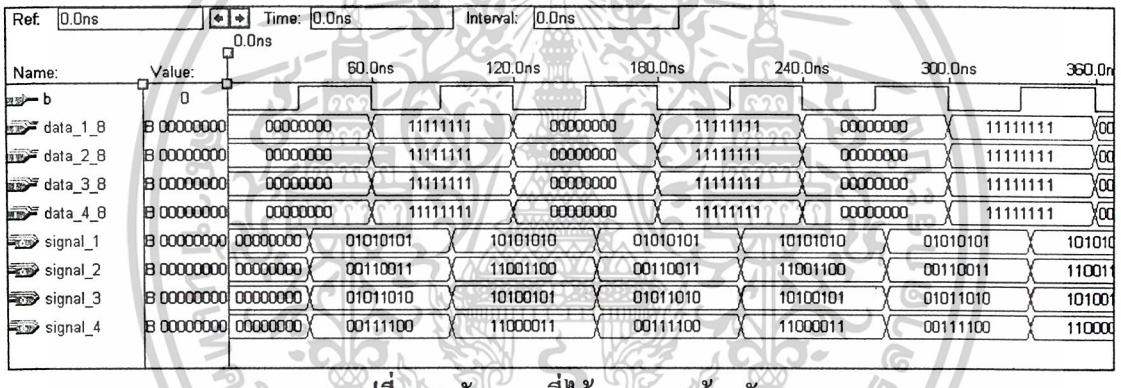
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.13 สัญญาณที่ได้จากวงจรพาราแลลอินซีเรียลเอ้าท์

4.2.1.1.3 วงจรเข้ารหัส

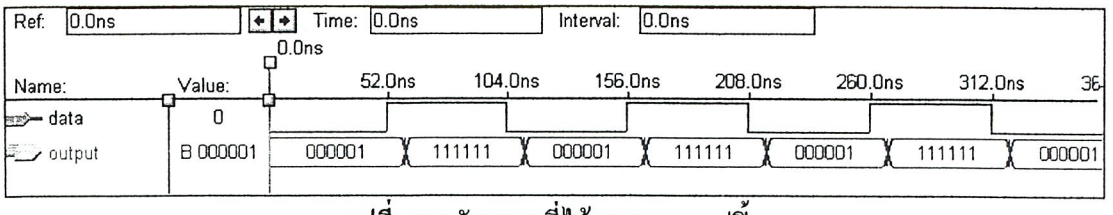
วงจรนี้นำเอาข้อมูลของผู้ใช้งานที่แปลงเป็น 8 บิตแล้วของแต่ละคน (data_1_8, data_2_8, data_3_8, data_4_8) มาเข้ารหัสโดยการ X-OR กับรหัสของตัวเองโดยผู้ใช้งานคนที่ 1 มีรหัสคือ "01010101", ผู้ใช้งานคนที่ 2 มีรหัสที่ 2 คือ "00110011" ,ผู้ใช้งานคนที่ 3 มีรหัสที่ 3 คือ "01011010" และผู้ใช้งานคนที่ 4 มีรหัส คือ "00111100" สัญญาณของผู้ใช้งานแต่ละคนที่เข้ารหัสแล้วจะเป็นดัง signal_1, signal_2, signal_3, signal_4 ซึ่งผลจากการจำลองการทำงานของวงจรเข้ารหัส จะได้ดังรูปที่ 4.14



รูปที่ 4.14 สัญญาณที่ได้จากวงจรเข้ารหัส

4.2.1.1.4 วงจรแมปปิ้ง

สัญญาณที่ได้จากการเข้ารหัส เราแปลงไปเป็นระดับโวลเตจเสมือน โดยแปลงข้อมูลบิต '0' ไปเป็น +1V ซึ่งแทนโดย "000001" และแปลงข้อมูลบิต '1' ไปเป็น -1V ซึ่งแทนโดย "111111" ผลจากการจำลองการทำงาน จะได้ดังรูปที่ 4.15



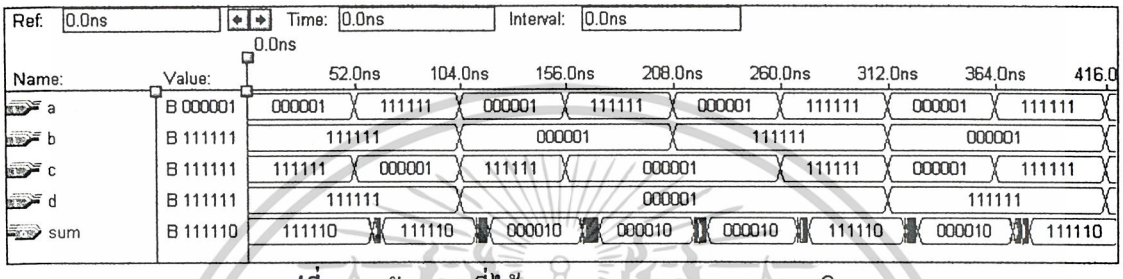
รูปที่ 4.15 สัญญาณที่ได้จากวงจรแมปปิ้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.1.1.5 วงจรบวก

เมื่อแปลงข้อมูลไปเป็นระดับโวลท์เตจเสมือนแล้ว นำเอาสัญญาณของผู้ใช้งานทั้ง 4 มารวมกันที่วงจรบวก

ผลของการรวมกันจะได้สัญญาณที่มี 5 ระดับคือ $\pm 4V, \pm 2V$ และ $0V$ แต่จากรูปเป็นการจำลองการทำงานเมื่อข้อมูลของผู้ใช้ทั้ง 4 (a,b,c,d) เป็นบิต '0', '1', '1', '1' ตามลำดับและผ่านวงจรเข้ารหัสและวงจรแมปปิ้ง แล้วซึ่งเมื่อสัญญาณทั้ง 4 เข้าสู่วงจรบวก จะทำให้ได้สัญญาณ 2 ระดับคือ $+2V(000010)$ และ $-2V(111110)$ รวมกัน 8 ชุด (sum) ซึ่งผลจากการจำลองการทำงาน จะได้ดังรูปที่ 4.16

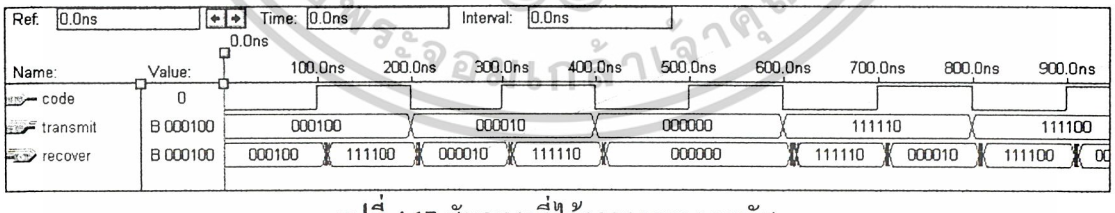


รูปที่ 4.16 สัญญาณที่ได้จากวงจรบวก (full adder) 6 บิต

4.2.1.2 ด้านรับ

4.2.1.2.1 วงจรถอดรหัส

เมื่อสัญญาณจากด้านส่งซึ่งเป็นข้อมูล 8 ชุดชุดละ 6 บิต (transmit) มาถึงด้านรับ ที่ด้านรับจะนำเอาสัญญาณที่รับได้มาคูณกับรหัสของผู้ใช้ โดยนำรหัสของผู้ใช้แปลงเป็นระดับโวลเตจเสมือนก่อน ซึ่งการคูณกันจะเป็นการคูณแบบบิตที่มีเครื่องหมาย คือถ้าบิตเครื่องหมายของ 2 สัญญาณเหมือนกันผลจากการคูณจะเป็นบวก (บิตเครื่องหมายเป็น 0) และถ้าบิตเครื่องหมายของ 2 สัญญาณต่างกันผลจากการคูณจะเป็นลบ (บิตเครื่องหมายเป็น 1) เป็นดังสัญญาณกู้คืน โดยผลการจำลองการทำงาน ดังรูปที่ 4.17

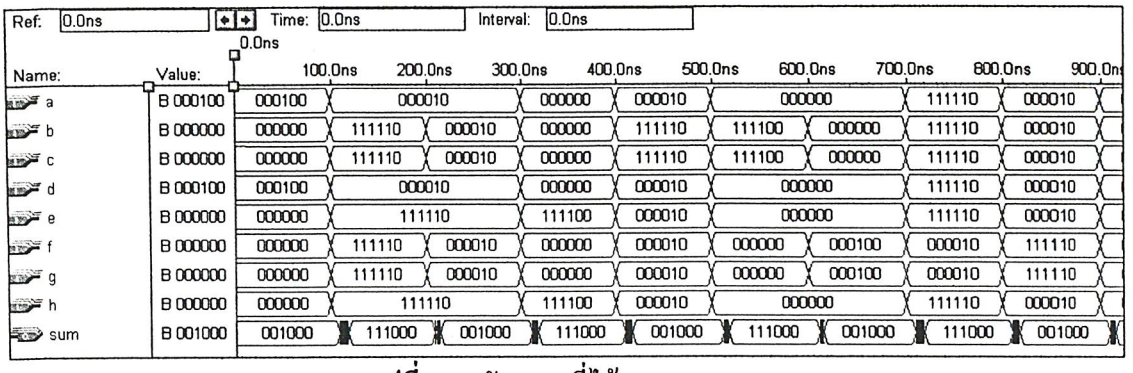


รูปที่ 4.17 สัญญาณที่ได้จากวงจรถอดรหัส

4.2.1.2.2 วงจรบวก

นำทั้งสัญญาณทั้ง 8 ชุดมาหาค่าเฉลี่ย โดยนำมาบวกกันที่วงจรบวก ก่อนที่จะนำไปตัดสินว่าเป็นบิต '0' หรือ '1' โดยผลจากการจำลองการทำงาน จะได้ดังรูปที่ 4.18

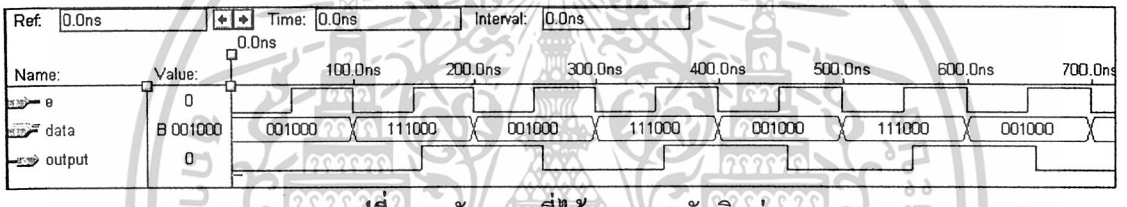
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.18 สัญญาณที่ได้จากวงจรบวก

4.2.1.2.3 วงจรตัดสินค่า

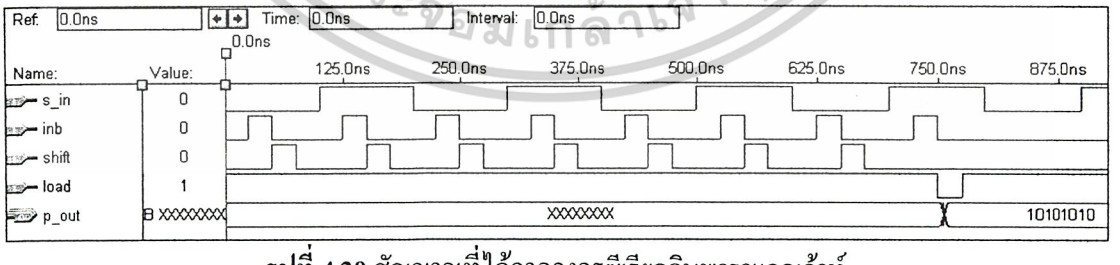
สัญญาณที่ได้จากวงจรบวก จะนำมาตัดสินว่าเป็นบิต '0' หรือบิต '1' ถ้าสัญญาณเป็น +8V (001000) จะได้ข้อมูลเป็นบิต '0' และถ้าสัญญาณเป็น -8V (111000)จะได้ข้อมูลเป็นบิต '1' ผลจากการจำลองการทำงาน จะได้ดังรูปที่ 4.19



รูปที่ 4.19 สัญญาณที่ได้จากวงจรตัดสินค่า

4.2.1.2.4 วงจรซีเรียลอินพาราแลล เอ้าท์ (SIPO)

สัญญาณอินพุต (s_in) จะนำมาเก็บไว้ที่บิต MSB เมื่อสัญญาณ inb เป็น '1' และจะถูกเลื่อนไปทางขวาทุกขอบขาขึ้นของสัญญาณ shift และจะถูกโหลดออกเมื่อครบ 8 บิตเมื่อสัญญาณ load เป็น '0' ผลจากการจำลองการทำงาน จะได้ดังรูปที่ 4.20

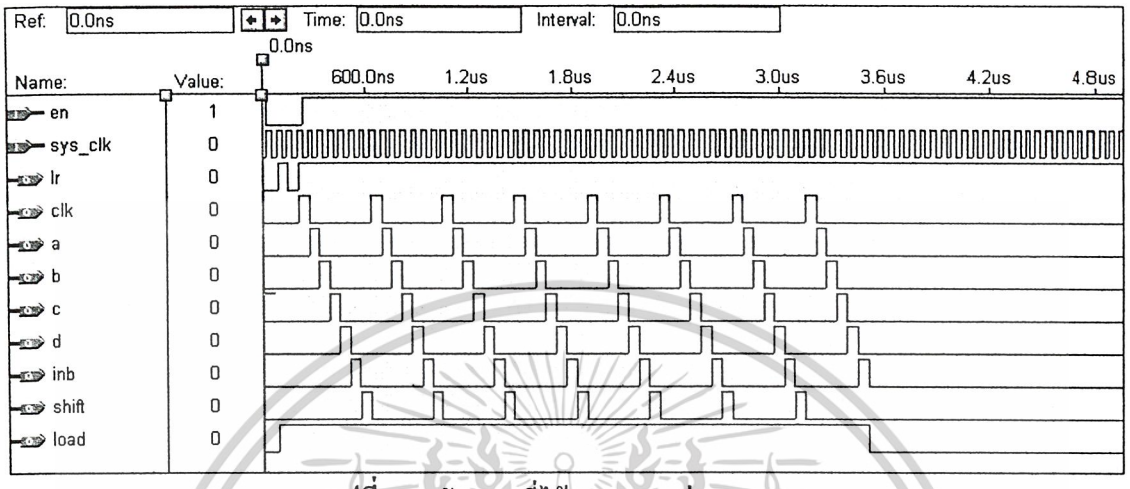


รูปที่ 4.20 สัญญาณที่ได้จากวงจรซีเรียลอินพาราแลลเอ้าท์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.1.2.5 วงจรหน่วยควบคุม

ความสัมพันธ์ระหว่างสัญญาณควบคุมทั้งหมดในวงจรเป็นไปตามที่เรากำหนดไว้ในวงจร หน่วยควบคุม ซึ่ง ผลการจำลองการทำงาน เป็นดังรูปที่ 4.21

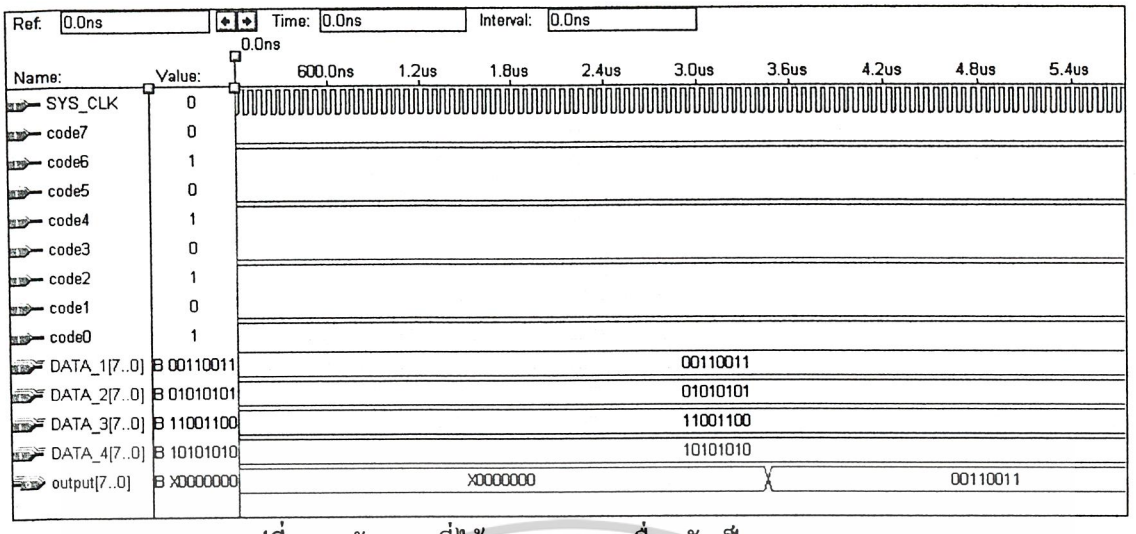


รูปที่ 4.21 สัญญาณที่ได้จากวงจรหน่วยควบคุม

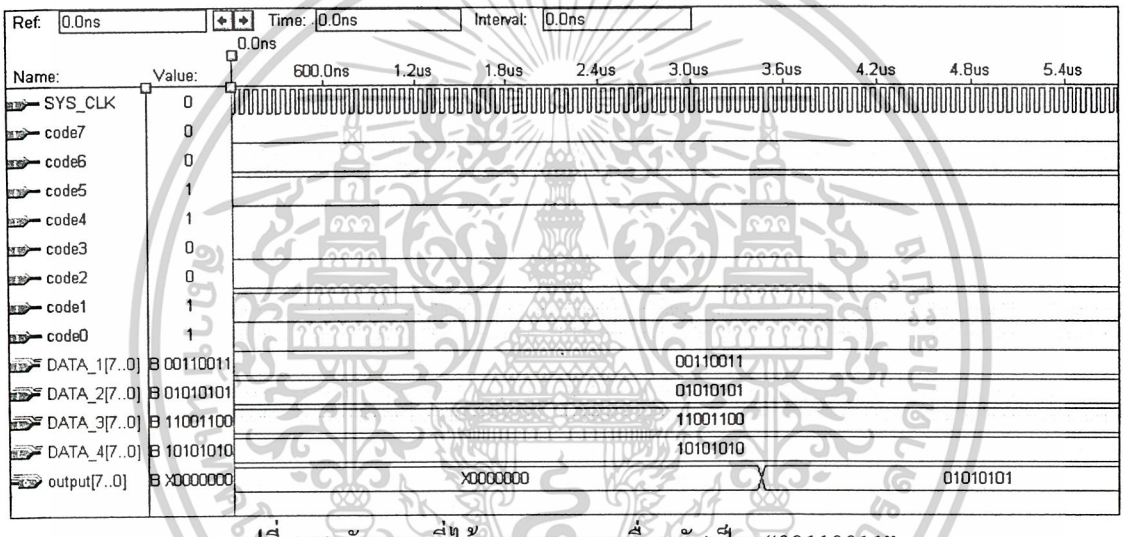
4.2.1.2.6 วงจรรวม

วงจรรวมทั้งหมดจะมีอินพุตเป็น 8 บิตที่จะเข้าวงจร PISO และเอาท์พุต 8 บิตที่ออกจากวงจร SIPO ซึ่งถ้าที่ด้านรับเราป้อนรหัสของผู้ใช้คนใด ก็จะได้ข้อมูลทั้ง 8 บิตของผู้ใช้คนนั้นกลับคืนมา ดังรูปที่ 4.22 ถ้ารหัสเป็นของผู้ใช้คนที่ 1 คือ “01010101” (โดยเรียงจาก code7, code6, code5, code4, code3, code2, code1, code0 ตามลำดับ) ข้อมูลทั้ง 8 บิตที่ได้ก็เป็นของผู้ใช้คนที่ 1(data_1), รูปที่ 4.23 เมื่อรหัสเป็นของผู้ใช้คนที่ 2 คือ “00110011” ข้อมูลทั้ง 8 บิตที่ได้ก็เป็นของผู้ใช้คนที่ 2(data_2), รูปที่ 4.24 เมื่อรหัสเป็นของผู้ใช้คนที่ 3 คือ “01011010” ข้อมูลทั้ง 8 บิตที่ได้ก็เป็นของผู้ใช้คนที่ 3(data_3), รูปที่ 4.25 เมื่อรหัสเป็นของผู้ใช้คนที่ 4 คือ “00111100” ข้อมูลทั้ง 8 บิตที่ได้ก็เป็นของผู้ใช้คนที่ 4 (data_4) และรูปที่ 4.26 เมื่อรหัสเป็นรหัสตั้งฉากอื่นๆ โดยในที่นี้คือ “01100110” ก็จะไม่ได้ข้อมูลใดๆออกมาคือจะได้ข้อมูล 8 บิตเป็น “00000000”

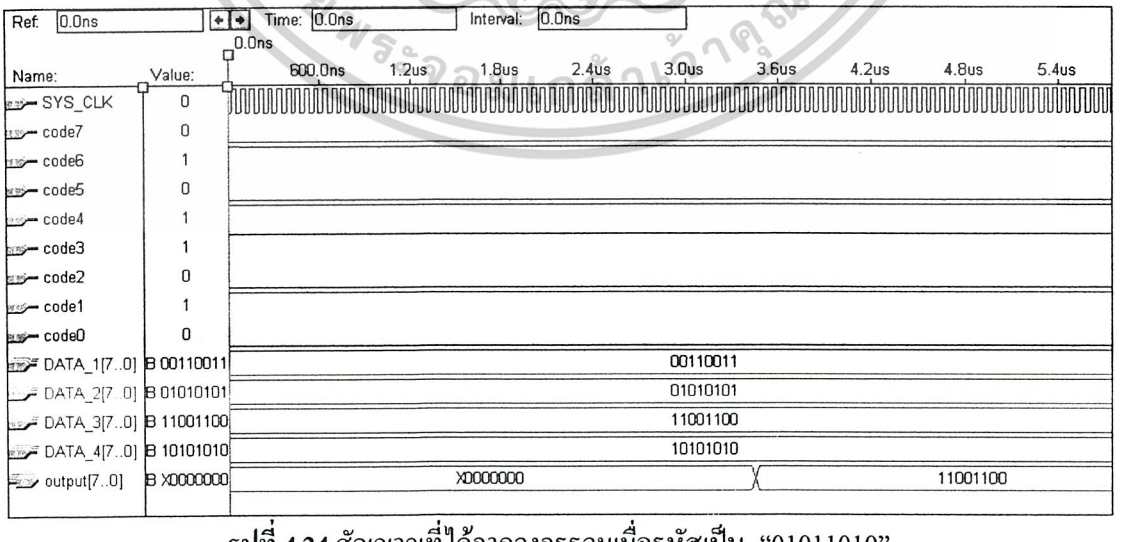
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.22 สัญญาณที่ได้จากวงจรรวมเมื่อรหัสเป็น "01010101"

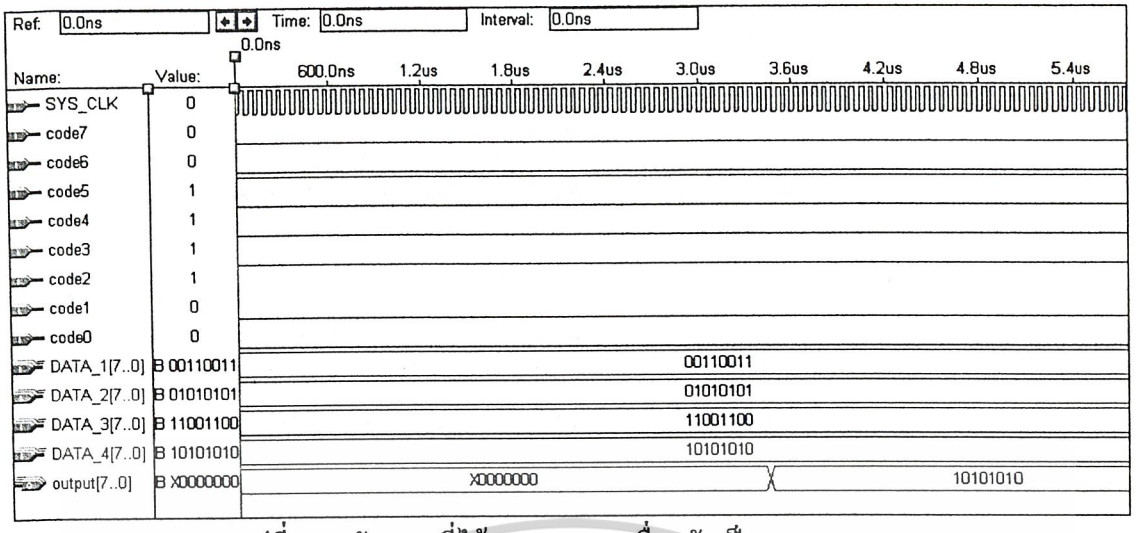


รูปที่ 4.23 สัญญาณที่ได้จากวงจรรวมเมื่อรหัสเป็น "00110011"

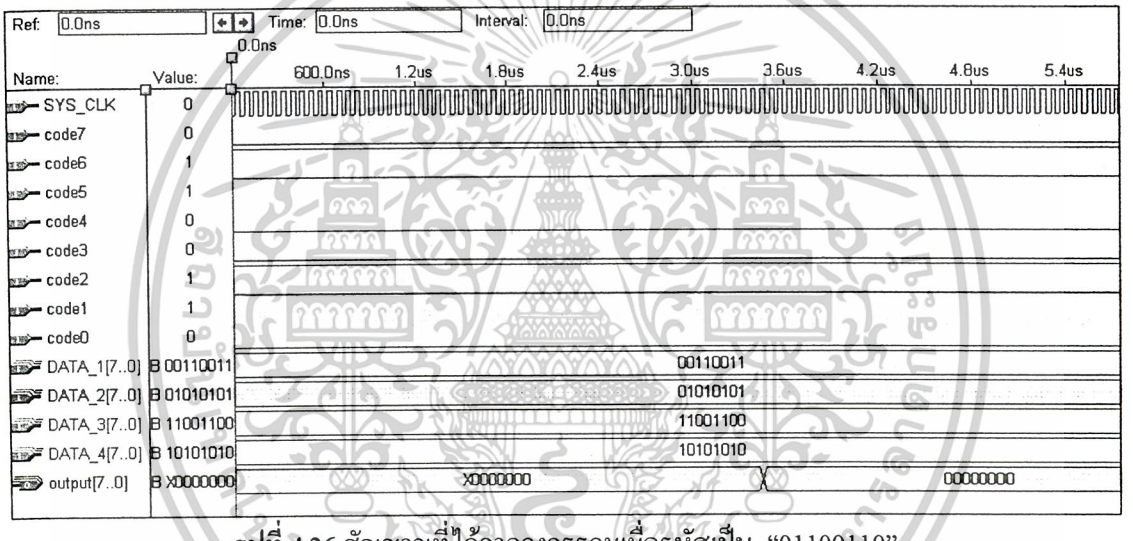


รูปที่ 4.24 สัญญาณที่ได้จากวงจรรวมเมื่อรหัสเป็น "01011010"

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.25 สัญญาณที่ได้จากวงจรรวมเมื่อรหัสเป็น "00111100"



รูปที่ 4.26 สัญญาณที่ได้จากวงจรรวมเมื่อรหัสเป็น "01100110"

4.2.1.2.7 สัญญาณอินพุตจากเครื่องกำเนิดความถี่

เมื่อป้อนสัญญาณซายน์ 4 ความถี่แทนข้อมูลของผู้ใช้งานทั้ง 4 คนเข้าสู่วงจร A/D แล้วเข้าสู่ชิป FPGA ที่สามารถเลือกรหัสของผู้ใช้งาน หลังจากนั้นสัญญาณที่ออกจากชิปจะนำเข้าสู่วงจร D/A เพื่อเปรียบเทียบกับข้อมูลของผู้ใช้งาน โดยผู้ใช้งานแต่ละคนมีความถี่ดังนี้

ผู้ใช้งานคนที่ 1 มีความถี่ 300 Hz

ผู้ใช้งานคนที่ 2 มีความถี่ 900 Hz

ผู้ใช้งานคนที่ 3 มีความถี่ 1 kHz

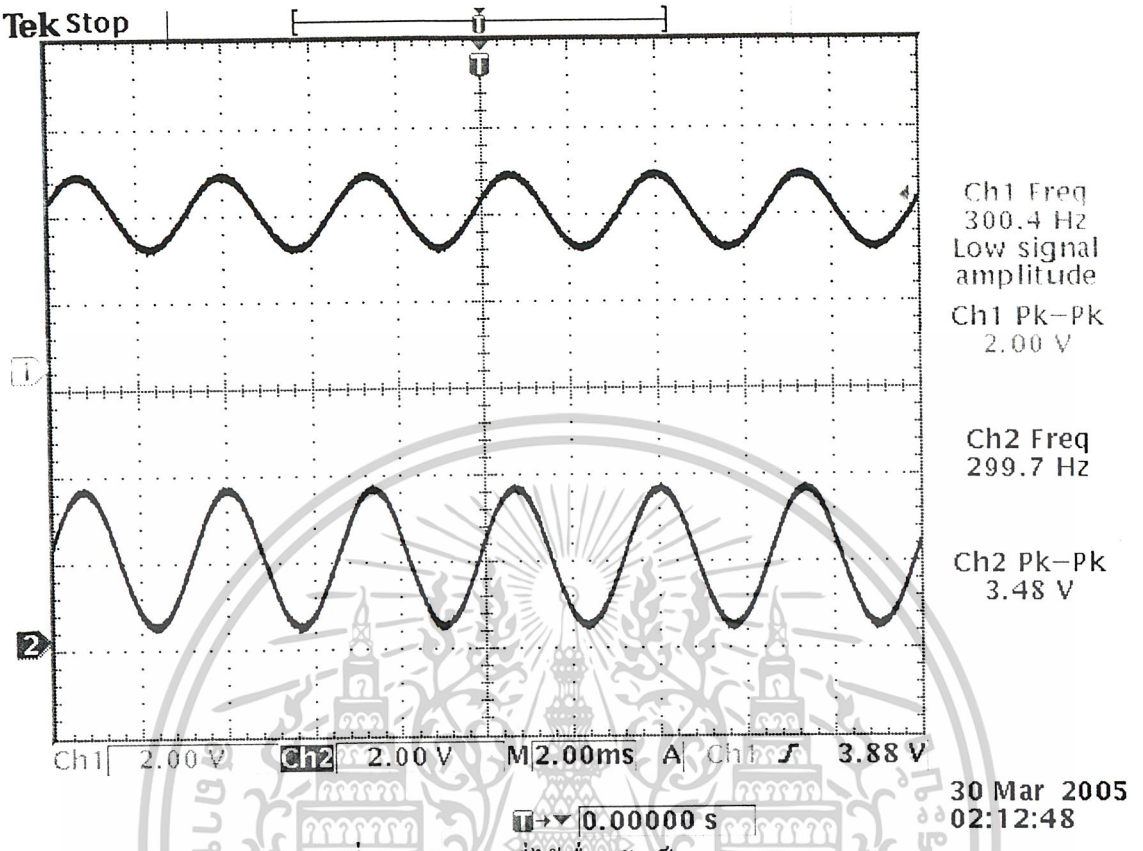
ผู้ใช้งานคนที่ 4 มีความถี่ 2 kHz

รูปที่ 4.27 เมื่อรหัสเป็นของผู้ใช้งานคนที่ 1 สัญญาณที่ได้ก็เป็นของผู้ใช้งานคนที่ 1 คือสัญญาณซายน์ที่มีความถี่ 300 Hz

Ch 1 คือ สัญญาณอินพุตของผู้ใช้งานคนที่ 1 ที่มีความถี่ 300 Hz, แอมปลิจูด 2V

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Ch 2 คือ สัญญาณเอาท์พุตที่ได้ โดยมีความถี่ประมาณ 300 Hz, แอมปลิจูดประมาณ 3.5V



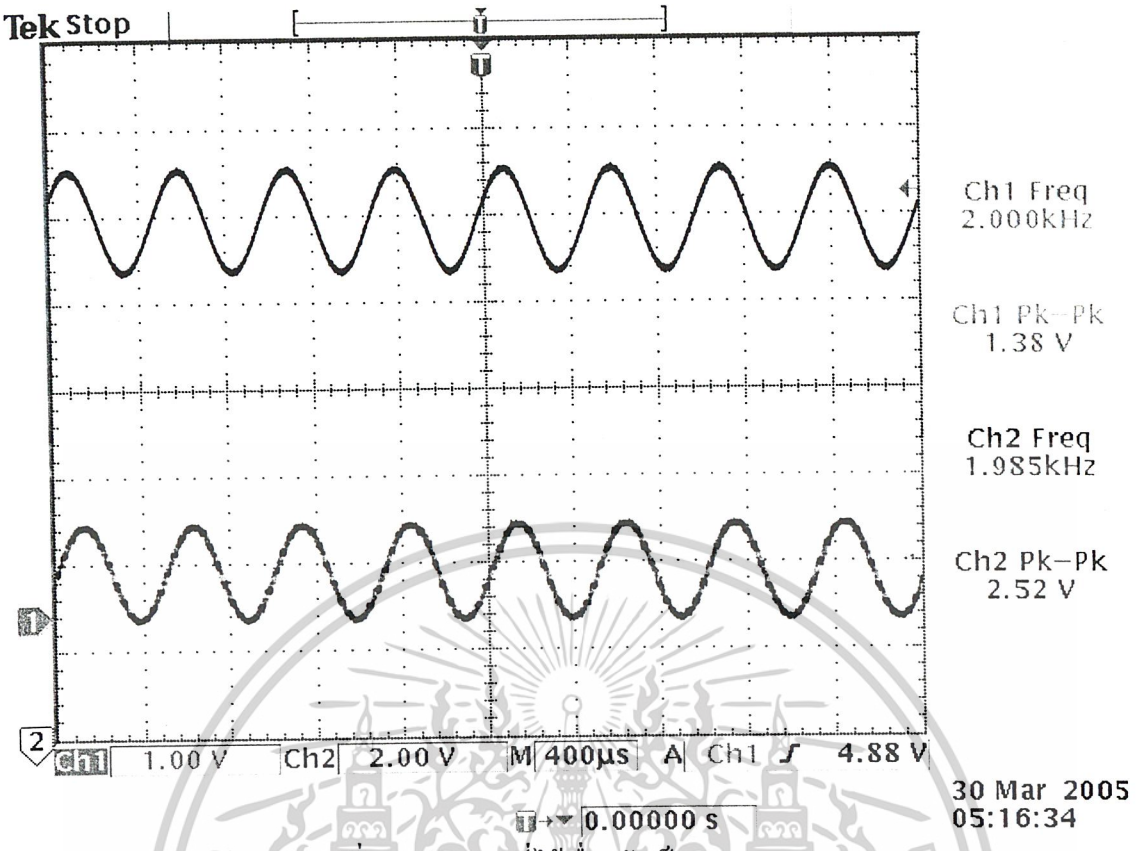
รูปที่ 4.27 สัญญาณที่ได้เมื่อรหัสเป็น "01010101"

รูปที่ 4.28 เมื่อรหัสเป็นของผู้ใช้งานคนที่ 4 สัญญาณที่ได้ก็เป็นของผู้ใช้งานคนที่ 4 คือสัญญาณซายน์ที่มีความถี่ 2 kHz

Ch 1 คือ สัญญาณอินพุตของผู้ใช้งานคนที่ 4 ที่มีความถี่ 2 kHz, แอมปลิจูด 1.4V

Ch 2 คือ สัญญาณเอาท์พุตที่ได้ โดยมีความถี่ประมาณ 2 kHz, แอมปลิจูดประมาณ 2.5V

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.28 สัญญาณที่ได้เมื่อรหัสเป็น "00111100"

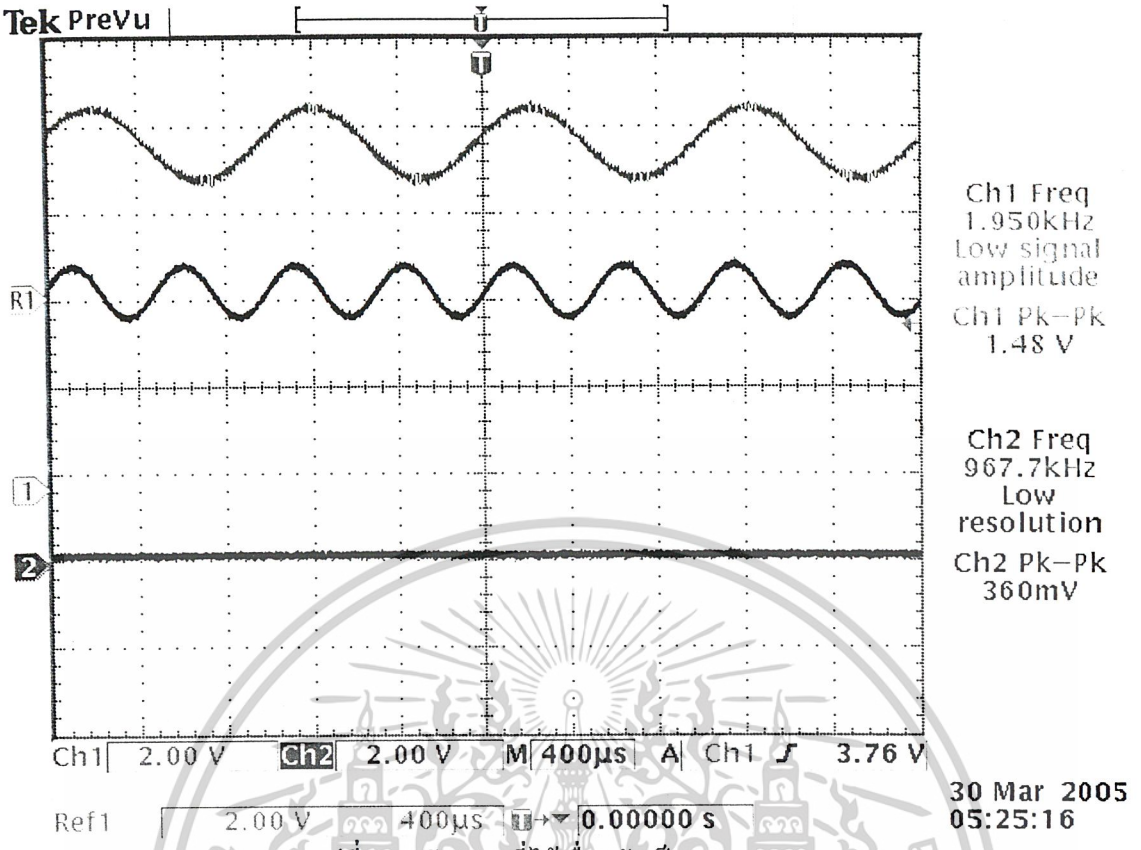
และรูปที่ 4.29 เมื่อรหัสเป็นรหัสตั้งฉากอื่นๆ โดยในที่นี้ใช้รหัส "01100110" ก็จะไม่ได้สัญญาณใดๆ ออกมา

Ref1 คือ สัญญาณอินพุตของผู้ใช้งานคนที่ 1 ที่มีความถี่ 300 Hz, แอมปลิจูด 2 V

Ch 1 คือ สัญญาณอินพุตของผู้ใช้งานคนที่ 4 ที่มีความถี่ 2 kHz, แอมปลิจูด 1.5 V

Ch 2 คือ สัญญาณเอาท์พุตที่ได้ซึ่งจะไม่ได้สัญญาณใดๆออกมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

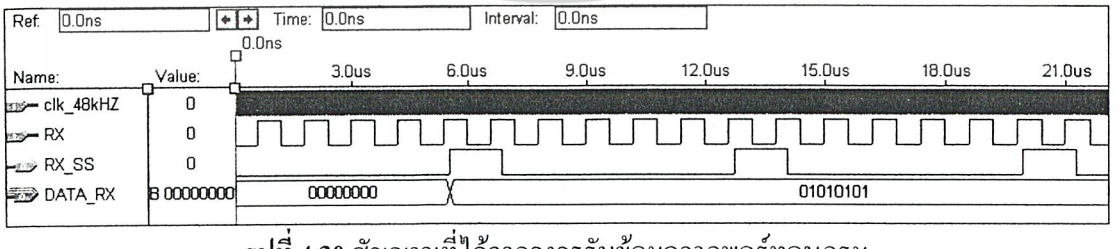


รูปที่ 4.29 สัญญาณที่ได้เมื่อรหัสเป็น "01100110"

4.2.2 ผลการทดลองระบบแบบหลายผู้ใช้ผ่านพอร์ทอนุกรม

4.2.2.1 วงจรรับข้อมูลจากพอร์ทอนุกรม

วงจรรับข้อมูลจากพอร์ทอนุกรมนี้จะทำการรับข้อมูลที่มาจากพอร์ทอนุกรม โดยข้อมูลที่รับมาได้จะแบ่งเป็นสตาร์ทบิต 1 บิต ซึ่งเป็นบิต '0', ข้อมูลที่เราต้องการ 8 บิต(data_rx) และสต๊อปบิตอีก 1 บิต ซึ่งเป็นบิต '1' แต่จะนำเอาเฉพาะข้อมูลที่เราต้องการ 8 บิตไปใช้เท่านั้น และเมื่อรับข้อมูลมาครบทั้ง 10 บิต แล้วจะส่งสัญญาณอีเนเบิล (rx_ss) ออกมาควบคุมส่วนควบคุม เพื่อให้วงจรทั้งหมดทำงานได้ก็ต่อเมื่อได้รับข้อมูลมาครบแล้ว โดยผลจากการจำลองการทำงาน จะได้ดังรูปที่ 4.30

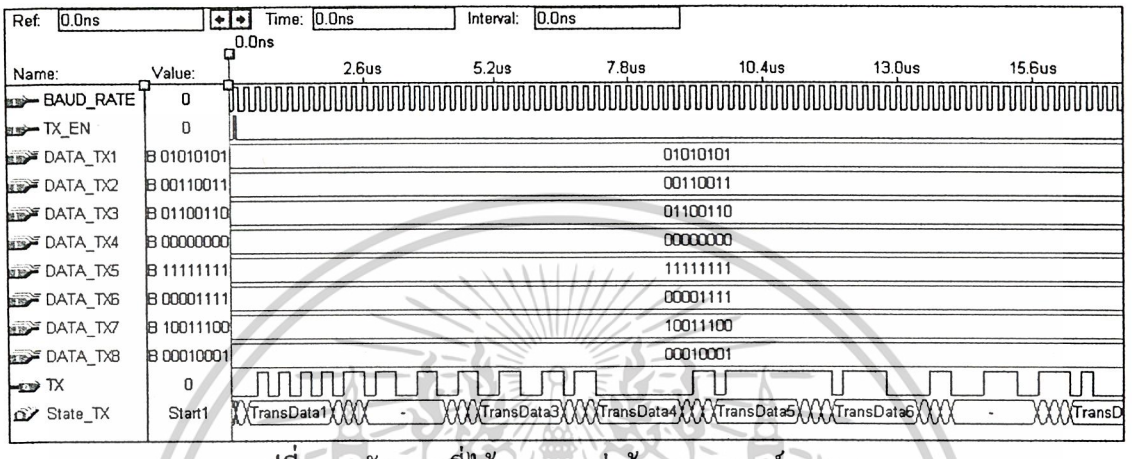


รูปที่ 4.30 สัญญาณที่ได้จากวงจรรับข้อมูลจากพอร์ทอนุกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.2.2 วงจรส่งข้อมูลออกพอร์ทอนุกรม

วงจรส่งข้อมูลออกพอร์ทอนุกรมจะทำการส่งข้อมูล (TX) ออกทางพอร์ทอนุกรมโดยจะส่งข้อมูลออกทีละ 10 บิต ซึ่งแบ่งเป็นสตาร์ทบิต 1 บิต ซึ่งเป็นบิต '0', ข้อมูลที่เราต้องการ 8 บิต และสต๊อปบิตอีก 1 บิต ซึ่งเป็นบิต '1' และจะทำการส่งชุดต่อไปอีกจนครบ 8 ชุดโดยเรียงจาก data_tx1, data_tx2, data_tx3, data_tx4, data_tx5, data_tx6, data_tx7, data_tx8 ผลจากการจำลองการทำงาน จะได้ดังรูปที่ 4.31



รูปที่ 4.31 สัญญาณที่ได้จากวงจรส่งข้อมูลออกพอร์ทอนุกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
สรุปผลการวิเคราะห์

วิเคราะห์ผลการทดลอง

ระบบ CDMA เป็นระบบที่มีคุณสมบัติในการเลือกการส่งและการับ การดักฟังทำได้ยาก สามารถป้องกันและแยกสัญญาณรบกวนออกไปได้ ป้องกันการแทรกสอดสัญญาณได้ดี มีความปลอดภัยของข้อมูลสูง เป็นต้น รายงานฉบับนี้ได้นำวิธีการมอดูเลตแบบสเปกตรัมแผ่ที่ใช้คือแบบวิธีจัดลำดับโดยตรง ซึ่งระบบ DS-CDMA ที่ทำการศึกษาจะประกอบไปด้วย เครื่องส่ง ช่องสัญญาณ และเครื่องรับ โดยเครื่องส่งจะประกอบไปด้วยลำดับข้อมูลซึ่งจะถูกคูณกับรหัสการแผ่สเปกตรัมที่มีรูปแบบเป็นเลขฐานสอง จากนั้นสัญญาณจะถูกส่งผ่านช่องสัญญาณ ซึ่งจะเกิดการเฟดของสัญญาณทำให้ขนาดของสัญญาณมีการผิดเพี้ยนไป โดยกำหนดให้ช่องสัญญาณดังกล่าว มีสัญญาณรบกวน ซึ่งในที่นี้กำหนดให้เป็นสัญญาณรบกวนเกาส์ขาวแบบบวก จากนั้นสัญญาณจะส่งไปที่เครื่องรับโดยสัญญาณดังกล่าวจะถูกคูณกับรหัสการแผ่สเปกตรัมซึ่งจะต้องมีรหัสตรงกับทางด้านส่ง จากนั้นจะผ่านส่วนของตัวรับสัญญาณ ทำการวิเคราะห์สมรรถภาพของระบบ DS-CDMA ที่มีสัญญาณรบกวนอันเนื่องมาจากสัญญาณรบกวนเกาส์ขาวแบบบวก จากนั้นนำผลที่ได้มาจำลองการทำงานโดยใช้โปรแกรม MATLAB

จากผลการวิเคราะห์เชิงเลขแสดงให้เห็นตัวแปรที่มีผลต่อสมรรถนะของระบบคือ ค่า SNR ที่ใช้ในระบบ โดยจากการวิเคราะห์เชิงเลขแสดงให้เห็นว่าค่า SNR ที่เพิ่มขึ้นจะส่งผลให้ค่าอัตราความผิดพลาดบิตของข้อมูลลดลง และจากผลการทดลองทั้ง 2 แบบจะสังเกตได้ว่าค่าความผิดพลาดของบิตข้อมูลนั้นมีค่าต่ำกว่าค่าความผิดพลาดของบิตสัญญาณ (BER(data) <BER(signal))ในขณะที่มีระดับของอัตราค่าพลังงานของสัญญาณต่อสัญญาณรบกวน (SNR) ซึ่งแสดงให้เห็นถึงคุณสมบัติของระบบ DS-CDMA ที่แม้จะมีการผิดพลาดของสัญญาณมากบ้างแต่เมื่อถอดรหัสกลับมาเป็นสัญญาณข้อมูลแล้วจะมีความผิดพลาดที่น้อยลง แต่สิ่งที่ต้องแลกมาก็คือจะต้องใช้สัญญาณในการส่งมากขึ้น

เมื่อทำการทดลองเปรียบเทียบสมถรรภาพของรบบ DS-CDMA เมื่อมีการเปลี่ยนแปลงความยาวของรหัสวอล์ช จะเห็นว่าเมื่อความยาวของ รหัสวอล์ชเพิ่มมากขึ้น ส่งผลให้ระบบมีสมรรถภาพเพิ่มมากขึ้น ส่งผล คือที่ระดับอัตราส่วนระหว่างสัญญาณและสัญญาณรบกวน (SNR) เดียวกัน ความผิดพลาดบิตข้อมูล (BER) ของระบบจะลดลงตามลำดับ หรืออีกนัยคือระบบมีความสามารถในการป้องกันสัญญาณรบกวนได้ดีขึ้น และสามารถถอดรหัสข้อมูลให้ถูกต้องได้ดีกว่าเดิม

ในการทดลองซิงโครไนเซชั่น โดยใช้คุณสมบัติอัตสหสัมพันธ์ (auto-correlation) ของวอล์ช จะพบว่าระบบสามารถแยกแยะค่าอัตสหสัมพันธ์ แล้วทำการปรับเปลี่ยนระบบเพื่อที่จะให้ซิงโครไนเซชั่นกับสัญญาณที่รับได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วิจารณ์ผลการทดลอง

ในการทดลองเริ่มต้นด้วยการศึกษาข้อมูลของส่วนการเข้ารหัสและถอดรหัสข้อมูล การเข้ารหัสในแต่ละส่วนของระบบ DS-CDMA เช่น การเข้ารหัสวอล์ช การรวมกันของสัญญาณ การถอดรหัสวอล์ช โดยการอินทริเกรด และการเปรียบเทียบข้อมูล จุดที่เป็นปัญหาอย่างมากคือ เมื่อแบ่งการทำงานในกลุ่มเพื่อออกแบบส่วนของการจำลองการทำงานในแต่ละส่วน เมื่อนำมาต่อร่วมกันไม่สามารถนำมาต่อร่วมกันได้เลย จำเป็นที่จะต้องมีการแปลงส่วนเชื่อมต่อ เพื่อนำมาทำงานต่อเนื่องกัน ทำให้ต้องทำส่วนเชื่อมต่อในหลายๆจุด ซึ่งต้องใช้เวลาเป็นอย่างมาก จึงเป็นประสบการณ์ในการวางแผนงาน และปรับปรุงการทำงานต่อไป

การออกแบบวงจรดิจิตอลนั้นเมื่อนำเอาข้อมูลหลายๆข้อมูลมาบวกกัน จะทำให้ได้ข้อมูลหลายระดับแต่ระบบดิจิตอลนั้นมีแค่บิต “0” กับบิต“1” เท่านั้น จึงจำเป็นต้องใช้ระดับโวลท์เตจเสมือน

การทดลองระบบผู้ใช้หลายคนผ่านพอร์ทอนุกรมนั้นไม่สามารถกำหนดความเร็วบอดเรตได้สูงๆ เพราะระบบจะทำงานไม่ทัน ดังนั้นจึงกำหนดความเร็วบอดเรตได้เพียง 4800

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก

ระบบการเข้าช่องสัญญาณ (Multiple access system)

1. การเข้าถึงหลายทางแบบแบ่งความถี่ (Frequency Division Multiple Access : FDMA)

FDMA เป็นระบบการเข้าถึงข้อมูลที่ใช้ในระบบอนาล็อกแบบแบ่งความถี่และใช้กันอย่างกว้างขวาง ในระบบดาวเทียมสื่อสารทั่วโลกหลายระบบด้วยกันรวมทั้งระบบอินเทลแซต โดยใช้การทำงานของโครงข่ายไมโครเวฟ ภาคพื้นดินแม้ว่าประสิทธิภาพการใช้วงจรจะต่ำก็ตาม จากการเข้าถึงหลายทางในระบบดาวเทียมอินเทลแซต IV-A ปรากฏว่าประสิทธิภาพ การใช้งานความถี่ของระบบการเข้าถึงหลายทางแบบแบ่งความถี่ผันแปรตามขนาดของคลื่นพาห์เช่น 9.6 ช่อง/MHz สำหรับตัวพาห์คลื่นเดียวที่มีความกว้างแถบ2.5 MHz และ 31.7 ช่อง/MHz ที่มีความกว้างแถบ 25 MHz ปัจจุบันระบบอินเทลแซต ใช้ระบบคลื่นพาห์ปลายทาง (multi-destination carrier) ซึ่งสัญญาณเบสแบนด์หลายๆ สัญญาณที่จะส่งไปตามสถานีคมนาคมภาคพื้นดินจำนวนหนึ่งจะมอดูเลตคลื่นพาห์ร่วมกัน (common carrier) แล้วส่งไปโดยวิธีการของระบบนี้ อย่างไรก็ตามประสิทธิภาพของการใช้วงจรจะลดลงถ้าจำนวนสถานีคมนาคมภาคพื้นดินเพิ่มขึ้น ในกรณีของการสื่อสารที่ต้องการความจุต่ำ (low capacity traffic) เรานิยมใช้ระบบ FDMA ชนิด เอสซีพีซี (SCPC) ที่ใช้การกำหนดล่วงหน้าหรือ สเปด (Spade) ที่ใช้กำหนดตามคำขอเป็นเทคนิคการเข้าถึงแบบหลายทางที่นับได้ว่าเป็นมาตรฐานของระบบโทรศัพท์เคลื่อนที่ชนิดอนาล็อก เทคนิค FDMA กำหนดให้ผู้ใช้บริการรายหนึ่งครอบครองช่องสื่อสารวิทยุหนึ่งช่องไปตลอดเวลาที่มีการใช้

2. การเข้าถึงช่องสัญญาณแบบแบ่งช่วงเวลา (Time Division Multiple Access : TDMA)

ระบบ TDMA เป็นระบบที่พัฒนาขึ้นมาเพื่อแก้ปัญหาที่เกิดจากการใช้ระบบ FDMA ในระบบนี้สัญญาณดิจิตอลที่เกิดจากการมัลติเพล็กซ์ โดยการแบ่งทางเวลาจะมอดูเลตตัวคลื่นพาห์คลื่นเดียวส่งและรับโดยสถานีคมนาคมภาคพื้นดินจำนวนหนึ่ง โดยผ่านทางทรานสปอนเดอร์ของดาวเทียมสื่อสารร่วมกัน แม้ว่าระบบจะต้องการเทคนิคการซิงโครไนซ์ และความกว้างแถบที่ใช้ความถี่กำหนดมาสามารถใช้อย่างมีประสิทธิภาพสูงได้เนื่องจากสามารถใช้เทคนิค DSI (Digital Speech Interpolation) ได้ง่ายกว่า ระบบนี้ให้ความคล่องตัวในการปฏิบัติการสื่อสาร เทคนิคการเข้าถึงแบบ TDMA เป็นระบบ IS-54 การแบ่งช่องสื่อสารวิทยุแต่ละช่องของเทคนิค TDMA กำหนดให้มีแบนด์วิดท์ช่องละ 30 kHz แต่จุดเด่นที่ระบบ IS-54 เหนือกว่าแบบ FDMA ก็คือหนึ่งช่องสื่อสารวิทยุจะมีการแบ่งช่วงเวลาในการใช้งานออกเป็น 3 ช่วงสลับกันไป ทำให้สามารถรองรับผู้ใช้บริการได้ถึง 3 รายต่อหนึ่งช่องความถี่ การใช้งานของผู้ใช้บริการแต่ละรายจะเป็นไปตามช่วงเวลาที่ถูกกำหนดไว้เรียกว่าช่วงเวลา (time slot) หากพิจารณาให้รายละเอียดจะพบว่าระบบโทรศัพท์เคลื่อนที่ซึ่งใช้เทคนิค TDMA ทุกระบบต่างก็ใช้เทคนิคของ FDMA เป็นพื้นฐาน เพราะยังมีการแบ่งช่องสื่อสารวิทยุออกเป็นช่องๆ ตามช่องความถี่ แล้วจึงนำแต่ละช่องมาแบ่งตามช่องเวลา

ภายในช่องสื่อสารวิทยุของระบบ IS-54 ซึ่งได้ประกอบไปด้วยเวลา 3 ช่องจะมีการแบ่งแต่ละช่องเวลาให้ผู้ใช้บริการแต่ละรายที่กำหนดโทรศัพท์ผู้ใช้แต่ละคนจะสามารถรับหรือส่งข้อความได้เฉพาะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ช่วงเวลาที่ตนได้รับสิทธิเท่านั้น ช่วงเวลาอื่นๆ จะถูกใช้งานโดยผู้ใช้บริการรายอื่น แต่เนื่องจากช่องเวลาแต่ละช่องกินเวลาระยะสั้นมากจนประสาทของมนุษย์ไม่สามารถแยกแยะได้ว่ามีการแบ่งเวลาออกเป็นช่วงให้ใช้งาน ทำให้ผู้ใช้โทรศัพท์สามารถใช้โทรศัพท์พูดคุยได้ตลอดเวลา ไม่มีการติดขัดแต่อย่างใด

ระบบโทรศัพท์เคลื่อนที่ GSM เป็นอีกระบบหนึ่งที่ใช้เทคนิค TDMA เพียงแต่แบนด์วิดท์ของช่องสัญญาณวิทยุแต่ละช่องมีขนาดเป็น 200 kHz และมีการแบ่งช่องสื่อสารวิทยุออกเป็น 8 ช่องเวลาเมื่อพิจารณาดูแล้วจะเห็นว่าระบบ GSM ก็มิได้เพิ่มประสิทธิภาพในการใช้ย่านความถี่ให้เหนือระบบอนาล็อก แต่ประการใดทั้งนี้เนื่องจากการใช้ย่านความถี่ความถี่กว้าง 200 kHz สำหรับรองรับผู้ใช้บริการ 8 ราย ซึ่งก็เป็นตัวเลขเดียวกับระบบอนาล็อกซึ่งผู้ใช้บริการแต่ละรายใช้ช่องสื่อสารวิทยุขนาด 25 kHz รวม 8 ราย ก็ยังคงใช้แบนด์วิดท์ขนาด 200 Hz อยู่เช่นเดียวกัน

3. การเข้าถึงหลายแบบการแบ่งรหัส (CDMA)

ระบบ CDMA เป็นระบบที่มีคุณสมบัติของการเข่าถึงหลายทางโดยการเข้ารหัสโดยแต่ละผู้ใช้งานจะมีรหัสเฉพาะที่แตกต่างกันในแต่ละผู้ใช้งาน เพื่อใช้ในการเข้ารหัสข้อมูลที่ต้องการส่ง โดยที่เครื่องรับจะรู้รหัสของผู้ใช้งานและจะทำการถอดรหัสหลังจากรับสัญญาณได้และนำไปแปลงเพื่อให้ได้สัญญาณเดิมกลับมา เมื่อแบนด์วิดท์ของสัญญาณรหัสมากกว่าแบนด์วิดท์ของข้อมูลที่ต้องการส่ง ซึ่งการเข้ารหัสจะเป็นการแผ่สเปกตรัมของสัญญาณและเรียกว่า การมอดูเลตแบบสเปกตรัมแผ่

ข้อกำหนดในการบ่งบอกว่าเป็น การมอดูเลตแบบสเปกตรัมแผ่มี 2 ข้อ คือ

1) แบนด์วิดท์ของสัญญาณที่ส่งออกไปต้องมากกว่าแบนด์วิดท์ของข้อมูลที่ยังไม่สเปกตรัมแผ่

2) แบนด์วิดท์ของสัญญาณที่ส่งออกไปไม่ขึ้นอยู่กับข้อมูลที่ยังไม่เข้ารหัสสเปกตรัมแผ่

นั่นคือการมอดูเลตแบบสเปกตรัมแผ่จะแปลงสัญญาณข้อมูลข่าวสารไปเป็นสัญญาณส่งออกอากาศที่มีแบนด์วิดท์มากขึ้น ซึ่งทำได้โดยการเข้ารหัสสัญญาณข้อมูลโดยสัญญาณรหัสนี้จะไม่ขึ้นอยู่กับข้อมูลและมีความกว้างของสเปกตรัมมากกว่าสัญญาณข้อมูลและจะกระจายกำลังของสัญญาณตลอดช่วงแบนด์วิดท์กว้างทำให้ มีความหนาแน่นกำลังที่ต่ำ อัตราส่วนของแบนด์วิดท์ที่ทำการส่งกับแบนด์วิดท์ของข้อมูลรียกว่า อัตราขยายประมวลผล (Processing Gain : Gp) ของระบบสเปกตรัมแผ่

$$G_p = B_t/B_i$$

โดย B_t คือแบนด์วิดท์ที่ทำการส่งหลังการเข้ารหัส และ B_i คือแบนด์วิดท์ของข้อมูลก่อนการเข้ารหัส

ที่เครื่องรับจะเปรียบเทียบสัญญาณที่รับได้กับสัญญาณรหัสเพื่อให้ได้ข้อมูลเดิมก่อนการเข้ารหัสออกมานั่นคือเครื่องรับจะต้องรู้รหัสที่ใช้ในการมอดูเลตก่อน สัญญาณสเปกตรัมแผ่จะมีคุณสมบัติที่แตกต่างจากสัญญาณแถบแคบดังนี้

1) ความสามารถของการเข้าถึงหลายทาง

ถ้ามีการส่งสัญญาณที่มีการแผ่สเปกตรัม เมื่อมีผู้ใช้งานหลายคนในเวลาเดียวกัน ทางเครื่องรับก็จะสามารถแยกผู้ใช้งานแต่ละรายได้โดยใช้รหัสที่แตกต่างกันของแต่ละผู้ใช้งานซึ่งรหัสดังกล่าวต้องมีค่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สหสัมพันธ์ข้าม ที่ต่ำพอ การหาค่าสหสัมพันธ์ของสัญญาณที่รับได้ด้วยรหัสสเปกตรัมแผ่จะดีสเปสสัญญาณของผู้ใช้งานนั้นแต่จะทำการแผ่สเปกตรัมของสัญญาณของผู้ใช้รายอื่นแทน นั่นคือกำลังของสัญญาณในแบนด์วิดท์ของข้อมูลจะมากกว่ากำลังของสัญญาณการแทรกสอดซึ่งทำให้แยกสัญญาณที่ต้องการได้ โดยความสามารถของการเข้าถึงหลายทาง

2) การป้องกันการแทรกสอดแบบหลายวิถี

ในช่องสัญญาณวิทยุสัญญาณระหว่างเครื่องส่ง-รับ จะมีจำนวนเส้นทางมากกว่า 1 วิถี อันเนื่องมาจากการสะท้อนและการหักเห ซึ่งสัญญาณในวิถีต่างๆ คือสัญญาณที่ถูกส่งมาต่างกันที่ขนาดแอมพลิจูดและเฟสเมื่อรวมสัญญาณดังกล่าวที่เครื่องรับก็จะเกิดการเสริมกันบางความถี่และหักล้างกันบางความถี่ ซึ่งการมอดูเลตแบบสเปกตรัมแผ่สามารถแก้ปัญหาดังกล่าวได้ โดยขึ้นกับชนิดของวิธีการมอดูเลตที่ใช้

3) เป็นระบบส่วนบุคคล

นั่นคือเครื่องรับจะดีสเปสได้ต้องรู้รหัสของสัญญาณที่ต้องการเท่านั้น

4) ลดผลจากการสอดแทรก

สหสัมพันธ์ข้ามของสัญญาณรหัสกับสัญญาณแถบแคบจะเป็นการกระจายกำลังงานของสัญญาณแถบแคบนั่นคือ ไปลดกำลังของสัญญาณแทรกสอดในแบนด์วิดท์ของข้อมูล 2 โดยสัญญาณที่มีการแผ่กระจายสเปกตรัมกับสัญญาณการแทรกสอดแถบแคบ ที่เครื่องรับ สัญญาณที่มีการแผ่สเปกตรัมจะดีสเปสในขณะที่สัญญาณแทรกสอดจะถูกแผ่สเปกตรัมกลายเป็นสัญญาณรบกวนพื้นหลังเมื่อเทียบกับสัญญาณที่ดีสเปส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข

การมอดูเลตสเปกตรัมแผ่ (Spread Spectrum Modulation : SS Modulate)

1. การจัดลำดับโดยตรง (Direct Sequence : DS)

ใน DS-CDMA สัญญาณข้อมูลจะถูกมอดูเลตโดยตรงกับสัญญาณรหัสดิจิตอล โดยสัญญาณข้อมูล อาจจะเป็นสัญญาณดิจิตอลหรืออนาล็อกก็ได้ สัญญาณข้อมูลจะถูกคูณกับสัญญาณรหัสและได้สัญญาณที่มอดูเลตเป็นคลื่นพาห์แบบแถบกว้างซึ่งเป็นเหตุผลที่เรียกวิธีนี้ว่า DS-CDMA

คุณสมบัติของระบบ DS-CDMA

1) การเข้าถึงหลายทาง

เมื่อมีผู้ใช้งานหลายคนใช้งานช่องสัญญาณพร้อมกันในเวลาเดียวกัน สัญญาณ DS จะเกิดการซ้อนทับกันในเวลาและความถี่ ที่เครื่องรับแบบร่วมนัยที่ถูกใช้ในการมอดูเลตส่วนของรหัสออกไป เมื่อพิจารณากำลังงานของผู้ใช้งานที่ต้องการในแบนด์วิดท์ของข้อมูล ถ้าสัมพันธ์ข้ามระหว่างรหัสของผู้ใช้งานที่ต้องการกับรหัสสัญญาณแทรกสอดมีค่าน้อย การดีเทคแบบร่วมนัยจะทำให้เกิดกำลังงานของสัญญาณแทรกสอดในแบนด์วิดท์ของข้อมูลในขนาดที่ต่ำ

2) การแทรกสอดหลายวิถี

ถ้าลำดับรหัสมีคุณสมบัติอัตสหสัมพันธ์ เป็นแบบอุดมคติ ซึ่งจะมีค่าเป็นเมื่ออยู่นอก $[-T_c T_c]$ โดย T_c คือค่าความกว้างของชิปนั่นคือถ้าสัญญาณที่ต้องการมีค่าประวิงเวลามากกว่า $2T_c$ การดีมอดูเลตแบบร่วมนัยจะทำให้สัญญาณดังกล่าวเป็นสัญญาณ การแทรกสอด ซึ่งทำให้กำลังงานในแบนด์วิดท์ของข้อมูลมีขนาดต่ำ

3) การแทรกสอดแบบแถบแคบ

การดีเทคแบบร่วมนัยที่เครื่องรับรวมถึงการคูณสัญญาณที่รับได้กับลำดับของรหัสที่สร้างขึ้นใหม่โดยแหล่งกำเนิดรหัส อย่างไรก็ดีที่เครื่องส่งการคูณสัญญาณแถบแคบด้วยรหัสแถบกว้างจะทำการแผ่สเปกตรัมสัญญาณแถบแคบ ดังนั้นกำลังงานในแบนด์วิดท์ของข้อมูลจะลดลงโดยตัวประกอบที่มีค่าเท่ากับอัตราขยายการประมวลผล

จากคุณสมบัติดังกล่าวสามารถสรุปถึงข้อดี และข้อเสีย ของ DS-CDMA ได้ดังนี้

ข้อดี

1) สัญญาณรหัสสร้างได้ง่ายโดยการคูณธรรมดา
2) ใช้ความถี่คลื่นพาห์เพียงความถี่เดียวทำให้การสังเคราะห์ความถี่ทำได้ง่าย
3) สามารถใช้ในการดีมอดูเลตแบบสมนัยได้

ข้อเสีย

1) การค้นเป้า (Acquire)และการรักษาสภาพการซิงโครไนซ์ของสัญญาณที่รับได้และสัญญาณรหัสทำได้ยาก เนื่องจากต้องทำในขนาดความกว้างชิปซึ่งมีขนาดเล็ก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2) เนื่องจากการซิงโครไนซ์ของสัญญาณที่รับได้กับสัญญาณรหัสที่สร้างใหม่ทำได้ยากทำให้แบนด์วิดท์ทั้งหมดที่กระจายถูกกำจัดในช่วง 10 ถึง 20 MHz

3) กำลังงานที่รับได้จาก ผู้ใช้งานที่ใกล้สถานีฐาน (base station) มีค่ามากกว่าที่รับได้จากที่ที่ไกลกว่าทำให้ผู้ใช้งานที่ใกล้สถานีฐานจะสร้างสัญญาณแทรกสอดให้กับผู้ใช้งานที่อยู่ใกล้ออกไปทำให้รับสัญญาณได้ยากขึ้นซึ่งผลของความใกล้-ไกล (near-far effect) แก้ได้โดยการควบคุมกำลังงานเพื่อให้ทุกสัญญาณจากทุกผู้ใช้งานมีกำลังงานเฉลี่ยที่เท่ากัน ที่สถานีฐานแต่อย่างไรก็ดี การควบคุมดังกล่าวทำได้ยาก

2. การกระโดดทางความที่ (Frequency Hopping : FH)

ในวิธีการนี้ความถี่คลื่นพาห์ของสัญญาณข่าวสารที่ถูกมอดูเลตจะไม่คงที่แต่จะเปลี่ยนไปในลักษณะที่มีคาบในช่วงเวลา T ความถี่คลื่นพาห์จะคงที่แต่จะเปลี่ยนไปในแต่ละช่วงเวลา ซึ่งรูปแบบการกระโดดนี้จะเป็นไปตามสัญญาณรหัสซึ่งเซตของความถี่ที่เป็นไปได้จะเรียกว่าเซตของการกระโดด (Hop Set) ซึ่งการครอบครองความถี่จะแตกต่างกับกรณีของวิธีการจัดลำดับโดยตรง โดยวิธีดังกล่าวจะใช้แบนด์วิดท์ทั้งหมดเพื่อใช้ในการส่งข้อมูลในขณะที่วิธีการกระโดดความถี่จะใช้แบนด์วิดท์เพียงส่วนเล็กๆ เพื่อใช้ในการส่งแต่ความถี่จะเปลี่ยนไปในเวลาที่ต่างกัน

สมมุติว่าวิธีการกระโดดความถี่ส่งในความถี่ที่ 2 ในช่วงคาบเวลาแรกดังรูปที่ 2.7 ระบบ DS จะส่งในช่วงเวลาเดียวกัน ตลอดช่วงความถี่ ดังนั้นค่ากำลังงานที่ส่งในช่วงความถี่ 1 จะน้อยกว่าในระบบ FH อย่างไรก็ดีระบบ DS ส่งความถี่ 1 ตลอดช่วงเวลา ในขณะที่ระบบ FH จะส่งในช่วงความถี่นี้ เฉพาะในช่วงเวลาเท่านั้น ในระบบทั้งสองจะมีค่ากำลังงานในช่วงความถี่เท่ากัน

สัญญาณข้อมูลจะถูกมอดูเลตกับคลื่นพาห์ซึ่งนิยมใช้วิธีการมอดูเลตแบบ FM กับสัญญาณอนาล็อก และ GSK ในการมอดูเลตแบบดิจิตอลและใช้การสังเคราะห์ความถี่ที่ความเร็วสูงที่ถูกควบคุมโดยสัญญาณรหัสและความถี่คลื่นพาห์จะถูกแปลงความถี่กลายเป็นความถี่ที่ใช้ในการส่ง

ที่เครื่องรับโดยใช้แหล่งกำเนิดลำดับรหัส สัญญาณที่รับได้จะถูกแปลงความถี่เป็นสัญญาณเบสแบนด์ที่ถูกมอดูเลตกับคลื่นพาห์ ซึ่งข้อมูลดังกล่าวจะได้กลับมาหลังจากการดีมอดูเลต ส่วนของวงจรซิงโครไนซ์/ติดตาม (Synchronize/tracking) ทำให้แน่ใจว่าการกระโดดที่เกิดจากแหล่งกำเนิดคลื่นพาห์จะซิงโครไนซ์กับรูปแบบการกระโดดของคลื่นพาห์ที่รับได้เพื่อให้เกิดการดีสเปสที่ถูกต้อง

ในระบบกระโดดความถี่ของ CDMA ยังสามารถแบ่งได้ 2 ชนิด คือ ถ้าจำนวนของการกระโดดมากกว่าอัตราข้อมูล กล่าวได้ว่าเป็นการกระโดดแบบเร็ว (Fast Frequency Hop CDMA : F-FH CDMA) ในกรณีนี้ความถี่คลื่นพาห์จะเปลี่ยนหลายครั้งตลอดการส่ง 1 บิต นั่นคือข้อมูล 1 บิต จะถูกส่งในความถี่ที่ต่างกัน ถ้าจำนวนการกระโดดน้อยกว่าอีตราข้อมูลกล่าวได้ว่าเป็นการกระโดดแบบช้า (Slow Frequency Hop CDMA : S-FH CDMA) ในกรณีนี้ข้อมูลหลายบิตจะถูกส่งที่ความถี่เดียวกัน แบนด์วิดท์ ของสัญญาณในการกระโดดแต่ละครั้งจะขึ้นอยู่กับ แบนด์วิดท์ของสัญญาณข้อมูล และรูปร่างของ สัญญาณการกระโดดและความถี่การกระโดดถ้ากรณีของการกระโดดแบบช้า แบนด์วิดท์ของสัญญาณ จะเป็นตัวแปรสำคัญในการกำหนดขนาดของแบนด์วิดท์ที่ใช้งาน ถ้ากรณีการกระโดดแบบเร็วรูปร่างของสัญญาณการกระโดด จะเป็นตัวแปรสำคัญในการกำหนดขนาดของ แบนด์วิดท์ที่ใช้งานในการกระโดดแต่ละครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คุณสมบัติของระบบ FH-CDMA

1) การเข้าถึงหลายทาง

ในกรณีของการกระโดดแบบเร็วข้อมูลจำนวน 1 บิตจะถูกส่งในความถี่ที่แตกต่างกันหลายความถี่ ถ้าแถบความถี่ส่วนใหญ่ถูกใช้งานโดยผู้ใช้งานเดียวกำลังงานของสัญญาณที่รับได้จะมากกว่ากำลังงานของสัญญาณแทรกสอดและจะรับสัญญาณได้ถูกต้อง

ในกรณีของการกระโดดแบบช้าข้อมูลหลายบิตจะถูกส่งในความถี่เดียว ถ้าความน่าจะเป็นของผู้ใช้งานอื่นที่ส่งในแถบความถี่เดียวกันต่ำพอ สัญญาณของผู้ใช้งานที่ต้องการจะรับได้อย่างถูกต้องหากเกิดมีสัญญาณแทรกสอดของผู้ใช้งานที่ส่งในช่วงเวลาดังกล่าว การใช้รหัสแก้ไขความผิดพลาด (Error-correcting code) จะทำให้ได้ข้อมูลที่ต้องการกลับคืนมา

2) การแทรกสอดแบบหลายวิถี

ในกรณีของการกระโดดแบบเร็ว ความถี่คลื่นพาห์จะเปลี่ยนหลายครั้งในช่วง 1 บิต ดังนั้นสัญญาณจะถูกมอดูเลตและถูกส่งไปบนความถี่คลื่นพาห์หลายความถี่ ผลของการแทรกสอดหลายวิถีจะแตกต่างในความถี่คลื่นพาห์ที่แตกต่างกัน นั่นคือสัญญาณจะถูกขยายที่ความถี่หนึ่งแต่จะถูกลดทอนที่อีกความถี่หนึ่ง ที่เครื่องรับผลตอบสนองที่ความถี่ที่แตกต่างกันจะถูกเฉลี่ยทำให้ลดผลของการแทรกสอดหลายวิถี ซึ่งในระบบ DS-CDMA จะลดส่วนการแทรกสอดหลายวิถีได้ดีกว่า

3) สัญญาณการแทรกสอดแบบแถบแคบ

สมมุติให้เกิดการแทรกสอดแบบแถบแคบที่ความถี่หนึ่งในความถี่ของการกระโดด ถ้าให้ G_p คือความถี่ของการกระโดด (G_p คือ อัตราขยายการประมวลผล) ผู้ใช้งานที่ต้องการส่งจะใช้ความถี่การกระโดด โดยการแทรกสอดจะมีโอกาสเกิด $1/G_p$ ที่เวลาใดๆ ซึ่งการแทรกสอดดังกล่าว จะลดลงด้วยตัวประกอบ G_p

4) ความน่าจะเป็นของการถูกดักจับของสัญญาณมีค่าต่ำ (LPI)

ความยากในการรับสัญญาณของระบบ FH ไม่ได้ขึ้นอยู่กับการมีกำลังส่งที่ต่ำเหมือนในระบบ DS-CDMA แต่อาศัยความถี่ที่สัญญาณทำการส่งซึ่งต้องรู้ล่วงหน้าและช่วงเวลาในการส่งแต่ละความถี่มีค่าน้อย

จากคุณสมบัติดังกล่าวสามารถสรุปถึงข้อดีและข้อเสียของ FH-CDMA ได้ดังนี้

ข้อดี

1) การซิงโครไนซ์ในระบบ FH-CDMA จะง่ายกว่าในระบบ DS-CDMA เนื่องจากในระบบ FH-CDMA การซิงโครไนซ์จะขึ้นอยู่กับช่วงเวลาในการกระโดดเมื่อการกระจายทางสเปกตรัมไม่ขึ้นอยู่กับการกระโดดที่ความถี่สูงแต่ขึ้นอยู่กับเซตของการกระโดดที่มาก ซึ่งช่วงเวลาการกระโดดจะมีค่ามากกว่าช่วงเวลาของชิปของระบบ DS-CDMA ดังนั้นระบบ FH-CDMA จึงยอมให้มีการผิดพลาดของการซิงโครไนซ์ได้มากกว่า

2) เนื่องจาก FH-CDMA มีความน่าจะเป็นที่ผู้ใช้งานต่างกันจะใช้ช่วงความถี่เดียวกันในเวลาเดียวกันมีค่าต่ำ นั่นคือถ้าผู้ใช้งานที่ไกลจากสถานีฐานส่งข้อมูลมาและที่สถานีฐานได้รับพร้อมกันสัญญาณของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผู้ใช้งานที่ใกล้กับสถานีฐานส่งมา แต่โอกาสที่จะเกิดในช่วงความถี่เดียวกันและในเวลาเดียวกันต่ำ นั่นคือจะลดผลใกล้-ไกล ได้

ข้อเสีย

1) ต้องใช้การสังเคราะห์ความถี่มีความซับซ้อนสูง

2) การใช้การดีมอดูเลตแบบโคฮีเรนท์ทำได้ยากเนื่องจากมีปัญหาที่ต้องรักษาความสัมพันธ์ของเฟสในช่วงการกระโดด

3. การกระโดดทางเวลา (Time Hopping: TH)

ในระบบนี้สัญญาณข้อมูลจะถูกส่งในช่วงเวลาที่กำหนดโดยรหัสของผู้ใช้งานแต่ละคนแกนเวลาจะถูกแบ่งเป็นเฟรมแต่ละเฟรมจะถูกแบ่งเป็น M ไทม์สล็อตผู้ใช้งานแต่ละคนจะส่งเฟรมใดใน M ไทม์สล็อต จะขึ้นอยู่กับรหัสที่กำหนดให้ในแต่ละผู้ใช้งาน

คุณสมบัติของระบบ TH-CDMA

1) การเข้าถึงหลายทาง

โดยกำหนดให้ความน่าจะเป็นของการส่งในช่วงเวลาเดียวกันของผู้ใช้งานมีค่าต่ำ โดยการกำหนดรหัสที่แตกต่างกันในแต่ละผู้ใช้งาน หากเกิดการส่งพร้อมกันเกิดขึ้นการใช้รหัสแก้ความผิดพลาด (Error-detecting feedback) จะทำให้ได้สัญญาณเดิมกลับคืนมา

2) การแทรกสอดแบบแถบแคบ

สัญญาณ TH-CDMA จะถูกส่งโดยใช้เวลาที่น้อยลง ซึ่งจะลดลงเท่ากับ $1/G_p$ โดย G_p คือ อัตราขยายการประมวลผล ที่เครื่องรับจะรับสัญญาณการแทรกสอด ในช่วงที่รับสัญญาณที่ต้องการเท่านั้น ดังนั้นจะรับสัญญาณการแทรกสอดในช่วงเวลาที่เป็น $1/G_p$ เปอร์เซ็นต์ของเวลา ซึ่งลดลงกำลังของการแทรกสอดด้วยตัวแปรเสริม G_p

3) ความน่าจะเป็นของการถูกดักจับของสัญญาณมีค่าต่ำ (LPI)

ในระบบ TH-CDMA ความถี่ของแต่ละผู้ใช้งานที่ใช้ส่งจะคงที่ แต่จะแตกต่างกันที่เวลาและช่วงเวลาที่ใช้ในการส่งยากที่จะรู้ว่าผู้ใช้งานไหนส่งมา

จากคุณสมบัติดังกล่าวสามารถแสดงข้อดีและข้อเสีย ของระบบ TH-CDMA ได้

ข้อดี

1) ง่ายในการสร้างกว่าระบบ FH-CDMA

2) มีประโยชน์เมื่อข้อมูลถูกส่งในกำลังงานเฉลี่ยแบบจำกัดแต่ไม่จำกัด กำลังค่าสูงสุด (peak power) เมื่อข้อมูลถูกส่งแบบเบิรสต์สั้นๆ ที่กำลังสูงๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3) ระบบ TH-CDMA ก็ลดผลของความใกล้ไกล เช่นเดียวกับระบบ FH-CDMA เนื่องจากความน่าจะเป็นที่ผู้ใช้งานที่ต่างกัน จะใช้ช่วงเวลาเดียวกันมีค่าต่ำนั่นคือผู้ใช้งานที่ใกล้กับสถานีและไกลจากสถานีฐานจะมีรูปแบบของการกระโดดทางเวลาที่แตกต่างกัน จึงลดผลของความใกล้ไกลได้

ข้อเสีย

1) หากมีการส่งพร้อมกันเกิดขึ้นจะเกิดการสูญเสียข้อเสียข้อมูลมากจึงต้องใช้ รหัสแก้ไขความผิดพลาด และการวางสลับ (Interleave) ที่ดี

4. ระบบไฮบริด (Hybrid System)

ระบบไฮบริดจะเป็นการรวมเทคนิคการมอดูเลตแบบสเปกตรัมแผ่ 2 เทคนิคหรือมากกว่าเข้าด้วยกัน หากพิจารณาแค่ DS-CDMA FH-CDMA และ TH-CDMA สามารถมีไฮบริดได้ถึง 4 แบบคือ DS/FH-CDMA DS/TH-CDMA FH/TH-CDMA และ DS/FH/TH-CDMA

ซึ่งไฮบริดเป็นการนำข้อดีของระบบที่จะนำมารวมกันเข้าด้วยกันเช่น DS/FH-CDMA ได้คุณสมบัติส่วนที่ดีของ DS-CDMA คือ การลดผลของช่องสัญญาณแบบหลายวิถี และ FH-CDMA คือการลดผลของความใกล้ไกล แต่ความซับซ้อนของเครื่องส่งและเครื่องรับจะเพิ่มขึ้น

โดยสัญญาณข้อมูลจะถูกแผ่สเปกตรัมก่อนโดยใช้ รหัสของส่วนของ DS และถูกมอดูเลตกับคลื่นพาห์ที่เกิดจากการกระโดดทางความถี่ที่สอดคล้องกับรหัส การใช้รหัสสัญญาณนาฬิกาจะช่วยให้ความสัมพันธ์รหัสทั้ง 2 คงที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ค.

โปรแกรม MATLAB ที่เขียน

```
clear all
BlockNum = 1; % ====== !!!warning!!!
Ima1 = imread('cat.jpg');
Ima2 = imread('dog.jpg');
figure, imshow(Ima1);
figure, imshow(Ima2);
w=[1 -1  1 -1  1 -1  1 -1;
   1  1 -1 -1  1  1 -1 -1];
alp = [-1,1];
Eb = mean(sum(alp.^2));
SNR = [1:10]; %dB ===== !!!warning!!!
sig = sqrt(Eb./(10.^(SNR/10))); %Standard deviation
for SNR_loop = 1:length(SNR)
    tot_err = 0;
    bit_err = 0;
    for loop_num = 1:BlockNum
%====== for image loop
=====================================================
        for color = 1:3
            for row = 1:200
                for column = 1:250
                    DouIma1(row,column,color) =
double(Ima1(row,column,color));
                    BinIma1 = dec2bin(DouIma1(row,column,color),8);
                    DouIma2(row,column,color) =
double(Ima2(row,column,color));
                    BinIma2 = dec2bin(DouIma2(row,column,color),8);
%== CDMA spread <START> ==
                    ref = dec2bin(0,8);
                    for j = [1:8]
                    if BinIma1(1,j) == ref(1,j)
                        data(1,j) = -1;
                    else
                        data(1,j) = 1;
                    end
                    if BinIma2(1,j) == ref(1,j)
                        data(2,j) = -1;
                    else
                        data(2,j) = 1;
                    end
                end
        for i=1:2 % ================image pixel
==========================
            for j=1:8 % ===========image pixel
==========================
                n = sig(SNR_loop)*randn(1,8); %Variance = sig^2
                for a=1:2
                    for b=1:8
                        if data(a,j)==w(a,b)
                            t(a,b)=1;
                        else
                            t(a,b)=-1;
                        end
                    end
                end
                for b=1:8
%=======%===========Send==========% 3 level : 1  0 -1
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    Tx(1,b)=(t(1,b)+t(2,b))/2;
%=======%============Transmit==========%
                    Rx(1,b)=Tx(1,b)+n(1,b);
%=======%============Receive===========% approximate value
                    if (Rx(1,b))>0.5
                        Rx_hat(1,b)=1;
                    else
                        if Rx(1,b)<=-0.5
                            Rx_hat(1,b)=-1;
                        else
                            Rx_hat(1,b)=0;
                        end
                    end
                end
%=======%==============================%=======%
                bit_err1 = sum(Tx~=Rx_hat);      %Transmit bit
                bit_err = bit_err + bit_err1;
%=======%============decode===========%=======%
                decode=0;
                for b=1:8
                    detect=Rx_hat(1,b)*w(i,b); %loop w(i,b)???
                    decode=decode+detect;
                end
                if decode>=0
                    BinX(i,j)=1;          %change interface
                else
                    BinX(i,j)=-1;         %change interface
                end
            end % for j
        end     % for i
int = [1 1 1 1 1 1 1 1];
for j = [1:8]
   if BinX(1,j) == int(1,j)
       BinIR1(1,j) = 1;
   else
       BinIR1(1,j) = -1;
   end
   if BinX(2,j) == int(1,j)
       BinIR2(1,j) = 1;
   else
       BinIR2(1,j) = -1;
   end
end
%ImaRecovery   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%ImageEncode == 0
if BinIR1 == [-1 -1 -1 -1 -1 -1 -1 -1];
    BinR1 = '00000000';
    DouY1 = bin2dec(BinR1);
    JPG1(row,column,color) = uint8(DouY1);
end
if BinIR2 == [-1 -1 -1 -1 -1 -1 -1 -1];
    BinR2 = '00000000';
    DouY2 = bin2dec(BinR2);
    JPG2(row,column,color) = uint8(DouY2);
end
%ImageEncode == 1
if BinIR1 == [-1 -1 -1 -1 -1 -1 -1  1];
    BinR1 = '00000001';
    DouY1 = bin2dec(BinR1);
    JPG1(row,column,color) = uint8(DouY1);
end
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
if BinIR2 == [-1 -1 -1 -1 -1 -1 -1  1];
    BinR2 = '00000001';
    DouY2 = bin2dec(BinR2);
    JPG2(row,column,color) = uint8(DouY2);
end
                    :
                    :
                    :
                    :
%ImageEncode == 254
if BinIR1 == [ 1  1  1  1  1  1  1 -1];
    BinR1 = '11111110';
    DouY1 = bin2dec(BinR1);
    JPG1(row,column,color) = uint8(DouY1);
end
if BinIR2 == [ 1  1  1  1  1  1  1 -1];
    BinR2 = '11111110';
    DouY2 = bin2dec(BinR2);
    JPG2(row,column,color) = uint8(DouY2);
end
%ImageEncode == 255
if BinIR1 == [ 1  1  1  1  1  1  1  1];
    BinR1 = '11111111';
    DouY1 = bin2dec(BinR1);
    JPG1(row,column,color) = uint8(DouY1);
end
if BinIR2 == [ 1  1  1  1  1  1  1  1];
    BinR2 = '11111111';
    DouY2 = bin2dec(BinR2);
    JPG2(row,column,color) = uint8(DouY2);
end
%ImaRecovery  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%== CDMA spread <END> ==
%== DATA BER counter ====================%========%
                err = sum(data~=BinX);
                sum_err=0;
                for j=1:8
                    sum_err=sum_err+err(1,j);
                end
                tot_err=tot_err+sum_err;
%====== end image loop
====================================================
                end % column
            end      % row
        end          % color
%====== end image loop
====================================================
    end % End of all blocks
figure, imshow(JPG1);
figure, imshow(JPG2);
    dat_BER(SNR_loop)=tot_err/(16*3*200*250*BlockNum);
    Tx_BER(SNR_loop)=bit_err/(16*8*3*200*250*BlockNum);
end% End of SNR loop
SNR
dat_BER
Tx_BER
figure,semilogy(SNR,dat_BER,'rx-'),xlabel('SNR(dB)'),ylabel('data
BER');
figure,semilogy(SNR,Tx_BER,'rx-'),xlabel('SNR(dB)'),ylabel('Tx BER');
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ง

วงจรหน่วยควบคุมที่อธิบายด้วยภาษา VHDL

```
Library ieee;
Use ieee.std_logic_1164.ALL;
Use ieee.std_logic_Unsigned.ALL;

Entity control_unit is

Port (   sys_clk,en                          : in  std_logic;
         lr,clk,a,b,c,d,inb,shift,load : out std_logic);
End ;

Architecture behavior  of control_unit is
    SIGNAL state  : integer range 0 to 57 :=0;

Begin
      PROCESS (sys_clk)
      BEGIN
      if sys_clk'event and sys_clk='1' then
      CASE state IS
            WHEN 0 =>   if en = '0' then
                        state<=1;
                        a<='0';
                        b<='0';
                        c<='0';
                        d<='0';
                        lr<='1';
                        clk<='0';
                        inb<='0';
                        shift<='0';
                        load<='1';
                        end if;
           WHEN 1 =>    state<=2;
                        a<='0';
                        b<='0';
                        c<='0';
                        d<='0';
                        lr<='0';
                        clk<='0';
                        inb<='0';
                        shift<='0';
                        load<='1';
          WHEN 2 =>     state<=3;
                        a<='0';
                        b<='0';
                        c<='0';
                        d<='0';
                        lr<='1';
                        clk<='1';
                        inb<='0';
                        shift<='0';
                        load<='1';
          WHEN 3 =>     state<=4;
                        a<='1';
                        b<='0';
                        c<='0';
                        d<='0';
                        lr<='1';
                        clk<='0';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                inb<='0';
                shift<='0';
                load<='1';
        WHEN 4 =>   state<=5;
                a<='0';
                b<='1';
                c<='0';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
        WHEN 5 =>   state<=6;
                a<='0';
                b<='0';
                c<='1';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
        WHEN 6 =>   state<=7;
                a<='0';
                b<='0';
                c<='0';
                d<='1';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
        WHEN 7 =>   state<=8;
                a<='0';
                b<='0';
                c<='0';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='1';
                shift<='0';
                load<='1';
        WHEN 8 =>   state<=9;
                a<='0';
                b<='0';
                c<='0';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='1';
                load<='1';
        ------------------------------------------------------------
----------bit 1
        WHEN 9 =>   state<=10;
                a<='0';
                b<='0';
                c<='0';
                d<='0';
                lr<='1';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                clk<='1';
                inb<='0';
                shift<='0';
                load<='1';
WHEN 10 =>      state<=11;
                a<='1';
                b<='0';
                c<='0';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
WHEN 11 =>      state<=12;
                a<='0';
                b<='1';
                c<='0';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
WHEN 12 =>      state<=13;
                a<='0';
                b<='0';
                c<='1';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
WHEN 13 =>      state<=14;
                a<='0';
                b<='0';
                c<='0';
                d<='1';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
WHEN 14 =>      state<=15;
                a<='0';
                b<='0';
                c<='0';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='1';
                shift<='0';
                load<='1';
WHEN 15 =>      state<=16;
                a<='0';
                b<='0';
                c<='0';
                d<='0';
                lr<='1';
                clk<='0';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        inb<='0';
                        shift<='1';
                        load<='1';
        ------------------------------------------------------------------
-----bit 2
            WHEN 16 =>  state<=17;
                        a<='0';
                        b<='0';
                        c<='0';
                        d<='0';
                        lr<='1';
                        clk<='1';
                        inb<='0';
                        shift<='0';
                        load<='1';
            WHEN 17 =>  state<=18;
                        a<='1';
                        b<='0';
                        c<='0';
                        d<='0';
                        lr<='1';
                        clk<='0';
                        inb<='0';
                        shift<='0';
                        load<='1';
            WHEN 18 =>  state<=19;
                        a<='0';
                        b<='1';
                        c<='0';
                        d<='0';
                        lr<='1';
                        clk<='0';
                        inb<='0';
                        shift<='0';
                        load<='1';
            WHEN 19 =>  state<=20;
                        a<='0';
                        b<='0';
                        c<='1';
                        d<='0';
                        lr<='1';
                        clk<='0';
                        inb<='0';
                        shift<='0';
                        load<='1';
            WHEN 20 =>  state<=21;
                        a<='0';
                        b<='0';
                        c<='0';
                        d<='1';
                        lr<='1';
                        clk<='0';
                        inb<='0';
                        shift<='0';
                        load<='1';
            WHEN 21 =>  state<=22;
                        a<='0';
                        b<='0';
                        c<='0';
                        d<='0';
                        lr<='1';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    clk<='0';
                    inb<='1';
                    shift<='0';
                    load<='1';
            WHEN 22 =>  state<=23;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='1';
                    load<='1';
      ------------------------------------------------------------
-----bit 3
            WHEN 23 =>  state<=24;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='1';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 24 =>  state<=25;
                    a<='1';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 25 =>  state<=26;
                    a<='0';
                    b<='1';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 26 =>  state<=27;
                    a<='0';
                    b<='0';
                    c<='1';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 27 =>  state<=28;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='1';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 28 =>  state<=29;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='1';
                    shift<='0';
                    load<='1';
            WHEN 29 =>  state<=30;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='1';
                    load<='1';
    ------------------------------------------------------------------
-----bit 4
            WHEN 30 =>  state<=31;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='1';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 31 =>  state<=32;
                    a<='1';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 32 =>  state<=33;
                    a<='0';
                    b<='1';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 33 =>  state<=34;
                    a<='0';
                    b<='0';
                    c<='1';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 34 =>  state<=35;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='1';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 35 =>  state<=36;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='1';
                    shift<='0';
                    load<='1';
            WHEN 36 =>  state<=37;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='1';
                    load<='1';
    ------------------------------------------------------------------
-----bit 5
            WHEN 37 =>  state<=38;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='1';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 38 =>  state<=39;
                    a<='1';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 39 =>  state<=40;
                    a<='0';
                    b<='1';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 40=>   state<=41;
                    a<='0';
                    b<='0';
                    c<='1';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 41 =>  state<=42;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='1';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 42 =>  state<=43;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='1';
                    shift<='0';
                    load<='1';
            WHEN 43 =>  state<=44;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='1';
                    load<='1';
    ------------------------------------------------------------------
-----bit 6
            WHEN 44 =>  state<=45;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='1';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 45 =>  state<=46;
                    a<='1';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 46 =>  state<=47;
                    a<='0';
                    b<='1';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 47 =>  state<=48;
                    a<='0';
                    b<='0';
                    c<='1';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
             WHEN 48 =>  state<=49;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='1';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='0';
                    load<='1';
            WHEN 49 =>  state<=50;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='1';
                    shift<='0';
                    load<='1';
            WHEN 50 =>  state<=51;
                    a<='0';
                    b<='0';
                    c<='0';
                    d<='0';
                    lr<='1';
                    clk<='0';
                    inb<='0';
                    shift<='1';
                    load<='1';
        ------------------------------------------------------------
    -----bit 7
            WHEN 51=>   state<=52;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                a<='0';
                b<='0';
                c<='0';
                d<='0';
                lr<='1';
                clk<='1';
                inb<='0';
                shift<='0';
                load<='1';
WHEN 52 =>      state<=53;
                a<='1';
                b<='0';
                c<='0';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
WHEN 53 =>      state<=54;
                a<='0';
                b<='1';
                c<='0';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
WHEN 54 =>      state<=55;
                a<='0';
                b<='0';
                c<='1';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
WHEN 55 =>      state<=56;
                a<='0';
                b<='0';
                c<='0';
                d<='1';
                lr<='1';
                clk<='0';
                inb<='0';
                shift<='0';
                load<='1';
WHEN 56 =>      state<=57;
                a<='0';
                b<='0';
                c<='0';
                d<='0';
                lr<='1';
                clk<='0';
                inb<='1';
                shift<='0';
                load<='1';
WHEN 57 =>      state<=0;
                a<='0';
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        b<='0';
                        c<='0';
                        d<='0';
                        lr<='1';
                        clk<='0';
                        inb<='0';
                        shift<='0';
                        load<='0';
-------------------------------------------------------------------
bit 8
            WHEN others => state<=0;
                        a<='0';
                        b<='0';
                        c<='0';
                        d<='0';
                        lr<='1';
                        clk<='0';
                        inb<='0';
                        shift<='0';
                        load<='1';
                    END CASE;
            END IF;
      END PROCESS ;
END;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ปริญญานิพนธ์นี้สำเร็จลุล่วงได้เนื่องจากความมุ่งมั่น ตั้งใจจริงในการทำงานของเพื่อนร่วมงาน และที่สำคัญได้รับการช่วยเหลือจากบุคลหลายบุคลซึ่งขอขอบคุณ อ.พรชัย ทรัพย์นิธิ ที่ให้คำแนะนำที่ดีตลอดมา อ.นิภา ลีลารุจิ ที่ให้ยืมใช้ห้องโปรเจคและอุปกรณ์ อ.ศรวัฒน์ ชิวปรีชา ที่ช่วยแนะแนวทาง นาย จารุวัตร เจริญลาภกิจ น.ส. จุฑามาศ ทองทวีผล ที่คอยช่วยเหลือ นส.พฤษชาติ น้อยกลาง ที่มีอุปกรณ์ให้ยืมเสมอ รวมทั้งผู้ที่ไม่ได้กล่าวถึงในที่นี้ ขอขอบคุณจากใจ

นายปุ๊ฉน ศิริมงคลลาวัลย์
นายพีรพงศ์ นิติเขตต์ปรีชา
นายพีระพล สุวรรณโชติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

1. WCDMA and cdma2000 for 3G Mobile network by Karim,M.R.New York:Mcgraw-Hill,2002

2. ToskataWCDMA for UMTS by Harri Holma and Antti Toskata.Chichester by John Willey,2000

3.Wideband Wireless digital communication.Andress by F.Molisch,editor. Upper saddle River,NJ:Printice Hall DTR,2001

4. CDMA Cellular Mobile Communications and Network Security by Dr. Man Young Rhee

5. ภาษา VHDL สำหรับการออกแบบวงจรดิจิตอล โดย ชำนาญ ปัญญาใส,วัชรากร หนูทอง,2547

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้