สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ระบบโครงข่ายท้องถิ่นแบบไร้สาย

Wireless Local Area Network

T 046489

โดย

นายนิธิศ สุดแก้ว

นายนิษณะ พัฒนพนิชธำรงค์

นางสาวอัฏฐพร แก้วงาม

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

เลขหมู่.....
เลขทะเบียน 46489
วัน, เดือน, ปี - 2 เม.ย. 2546

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

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบโครงข่ายท้องถิ่นแบบไร้สาย

Wireless Local Area Network

โดย

นายนิธิศ สุดแก้ว 41014223

นายนิษณะ พัฒนพนิชธำรงค์ 41014229

นางสาวอัฏฐพร แก้วงาม 41014535

อาจารย์ที่ปรึกษา

รศ.ดร. ปราโมทย์ วาดเขียน

อ. นภัทร สระเอี่ยม

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2544

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบโครงข่ายท้องถิ่นแบบไร้สาย

Wireless Local Area Network

ผู้จัดทำ

1. นายนิธิศ สุดแก้ว 41014223

2. นายนิษณะ พัฒนพนิชธำรงค์ 41014229

3. นางสาวอัฏฐพร แก้วงาม 41014535

อาจารย์ที่ปรึกษา

(รศ.ดร. ปราโมทย์ วาดเขียน)

อาจารย์ที่ปรึกษา

(อ. นภัทร สระเอี่ยม)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบโครงข่ายท้องถิ่นแบบไร้สาย

Wireless Local Area Network

โดย นายนิธิศ สุดแก้ว 41014223

นายนิษณะ พัฒนพนิชธำรงค์ 41014229

นางสาวอัฏฐพร แก้วงาม 41014535

อาจารย์ที่ปรึกษา รศ.ดร. ปราโมทย์ วาดเขียน

อ. นภัทร สระเอี่ยม

บทคัดย่อ

ระบบการสื่อสารข้อมูลในปัจจุบันได้ถูกพัฒนารุดหน้าขึ้นอย่างรวดเร็ว โดยที่ในการสื่อสารข้อมูลระยะใกล้นั้น ได้มีการวางระบบโครงข่ายท้องถิ่น (Local Area Network) ขึ้น ซึ่งเป็นระบบที่ถูกใช้งานอย่างแพร่หลาย เนื่องจากสามารถส่งผ่านข้อมูลได้อย่างสะดวกและรวดเร็ว โดยทั่วไปแล้วระบบโครงข่ายท้องถิ่นจะใช้สายนำสัญญาณในการเชื่อมต่อ ซึ่งอาจก่อให้เกิดปัญหาด้านพื้นที่ในการวางสายขึ้น

ปริญญานิพนธ์ฉบับนี้ได้จัดทำขึ้นเพื่อศึกษาและพัฒนาระบบโครงข่ายท้องถิ่นให้ติดต่อสื่อสารกันแบบไร้สาย ซึ่งในที่นี้จะทำการทดลองสื่อสารระหว่างเครื่องคอมพิวเตอร์จำนวน 2 เครื่องก่อน โดยจะเขียนโปรแกรมเพื่อใช้ในการควบคุมรูปแบบการสื่อสารผ่านเครื่องส่งและเครื่องรับที่ใช้คลื่นวิทยุเป็นตัวนำส่งผ่านข้อมูล ซึ่งสามารถนำไปพัฒนาให้เป็นเครือข่ายที่สมบูรณ์ต่อไป

ABSTRACT

Nowadays, Data Communications System has been developed rapidly. The usage of Local Area Network (LAN) is widely spread because of convenient transmission. General LAN always use wire connection which causes area problem.

The purpose of this project is to study and improve wireless LAN connection between 2 computers. Programming is adopted to control pattern of communication using radio signal be a carrier of transceiver. This project can be developed as a perfect network later.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

	หน้า
1. บทนำ	1
2. ทฤษฎีและหลักการ	2
2.1 การสื่อสารข้อมูล	2
2.2 มาตรฐานพอร์ตอนุกรมแบบ RS232	6
2.3 ระบบโครงข่ายท้องถิ่น	9
2.4 เครื่องส่งและเครื่องรับสัญญาณวิทยุแบบไร้สาย	12
2.5 การมอดูเลตแบบฟรีเควนซีชิพท์คีย์อิ้ง (Frequency Shift Keying)	13
2.6 การมอดูเลตทางความถี่ (Frequency Modulation)	17
2.7 วงจรสังเคราะห์ความถี่ (Frequency Synthesizer)	25
2.8 วงจรขยายกำลัง (Power Amplifier)	40
2.9 แมตช์ชิ่งอิมพีแดนซ์ (Matching Impedance)	46
2.10 แอดเดรสของพอร์ตอนุกรม	50
2.11 คอนโทรล MSComm	52
3. การคำนวณและการสร้าง	62
3.1 การออกแบบระบบ	62
3.2 วงจรฟรีเควนซีชิพคีย์อิ้งมอดูเลชั่น	68
3.3 วงจรฟรีเควนซีชิพคีย์อิ้งดีมอดูเลชั่น	70
3.4 วงจรสังเคราะห์ความถี่	73
3.5 วงจรฟรีเควนซีดีมอดูเลเตอร์	78
3.6 วงจรขยายความถี่ช่วงกว้าง	80
3.7 วงจรแหล่งจ่ายกำลัง	82
4. การทดลองและผลการทดลอง	83
4.1 ผลการรันโปรแกรมทางด้านเครื่อง Server	83
4.2 ผลการรันโปรแกรมทางด้านเครื่อง Server	90
4.3 วงจรฟรีเควนซีชิพท์คีย์อิ้ง	96
4.4 ระบบสังเคราะห์ความถี่	98
4.5 วงจรฟรีเควนซีดีมอดูเลเตอร์	103
5. บทวิจารณ์และบทสรุป	105

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญภาพ

		หน้า
รูปที่ 2.1	โครงร่างของข่ายสายแบบ Point-to-point	3
รูปที่ 2.2	โครงร่างของข่ายสายแบบ Multipoint	3
รูปที่ 2.3	รูปแบบของการสื่อสารข้อมูล	3
รูปที่ 2.4	รูปแบบเฟรมข้อมูลของการสื่อสารข้อมูลแบบอนุกรม	4
รูปที่ 2.5	รูปแบบของข้อมูลอนุกรมแบบอะซิงโครนัส	5
รูปที่ 2.6	ระดับของแรงดันตามมาตรฐาน RS232	7
รูปที่ 2.7	รูปแบบของอุปกรณ์เชื่อมต่อข้อมูลกับวงจรข้อมูลปลายทาง	7
รูปที่ 2.8	รูปร่างและตำแหน่งขาของคอนเน็กเตอร์แบบ DB-25	8
รูปที่ 2.9	รูปร่างและตำแหน่งขาของคอนเน็กเตอร์แบบ DB-9	8
รูปที่ 2.10	ลักษณะการเชื่อมต่อพื้นฐานของระบบโครงข่ายท้องถิ่น	12
รูปที่ 2.11	บล็อกไดอะแกรมของเครื่องส่งและเครื่องรับสัญญาณวิทยุแบบไร้สาย	13
รูปที่ 2.12	ลักษณะของสัญญาณเอฟเอสเค	14
รูปที่ 2.13	อัตราการเปลี่ยนแปลงของสัญญาณอินพุต	15
รูปที่ 2.14	การเบี่ยงเบนความถี่ของสัญญาณเอฟเอสเค	15
รูปที่ 2.15	ลักษณะของสัญญาณเอฟเอ็ม	18
รูปที่ 2.16	การสร้างสัญญาณเอฟเอ็มโดยใช้เทคนิคเฟสล็อกลูป	18
รูปที่ 2.17	การเปลี่ยนแปลงเบสเซลฟังก์ชันอันดับ n	20
รูปที่ 2.18	การกระจายสเปกตรัมของสัญญาณเอฟเอ็ม	21
รูปที่ 2.19	บล็อกไดอะแกรมของวงจรสังเคราะห์ความถี่	27
รูปที่ 2.20	บล็อกไดอะแกรมของทรานส์เฟอร์ฟังก์ชันของเฟสล็อกลูป	27
รูปที่ 2.21	บล็อกไดอะแกรมของระบบป้อนกลับแบบบวก	29
รูปที่ 2.22	วงจรเรโซแนนซ์	30
รูปที่ 2.23	วงจรกำเนิดสัญญาณที่ใช้วงจรขยายแบบดิฟเฟอเรนเชียล	30
รูปที่ 2.24	วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์	31
รูปที่ 2.25	วงจรเฟตออสซิสเลเตอร์	31
รูปที่ 2.26	วงจรสังเคราะห์ความถี่เฟสล็อกลูปแบบพรีสเกลเลอร์สองโมดูลัส	33
รูปที่ 2.27	ผลต่างเฟสเมื่อความถี่อินพุทเท่ากัน	34
รูปที่ 2.28	ผลต่างเฟสเมื่อความถี่อินพุตไม่เท่ากัน	34

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญภาพ (ต่อ)

หน้า

รูปที่ 2.29 วงจรลูปฟิลเตอร์อันดับที่ 1 34
รูปที่ 2.30 วงจรลูปฟิลเตอร์อันดับที่ 2 35
รูปที่ 2.31 วงจรลูปฟิลเตอร์อันดับที่ 3 38
รูปที่ 2.32 Class A power amplifier 40
รูปที่ 2.33 Dual supply complementary push pull class B amplifier 41
รูปที่ 2.34 สัญญาณเอาท์พุตที่เกิด Cross over distortion 42
รูปที่ 2.35 Dual supply complementary class AB power amplifier 42
รูปที่ 2.36 กระแสเอาท์พุทของ Class C power amplifier 43
รูปที่ 2.37 วงจรขยายความถี่สัญญาณช่วงกว้างรูปแบบหนึ่ง 43
รูปที่ 2.38 วงจรสมมูลย์ของวงจรขยายความถี่สัญญาณช่วงกว้างในการหาค่าเอาท์พุตอิมพีแดนซ์ 44
รูปที่ 2.39 วงจรสมมูลย์ของวงจรขยายความถี่สัญญาณช่วงกว้างในการหาค่าอินพุตอิมพีแดนซ์ 45
รูปที่ 2.40 การแมตช์ชิ่งอิมพีแดนซ์ในวงจรไฟฟ้ากระแสตรง 46
รูปที่ 2.41 การแมตช์ชิ่งอิมพีแดนซ์ในวงจรไฟฟ้ากระแสสลับ 47
รูปที่ 2.42 รูปแบบการต่อโครงข่ายแบบรีเวอร์สแอล 47
รูปที่ 2.43 การออกแบบโครงข่ายแบบรีเวอร์สแอลในกรณีที่ $R_p > R_S$ 48
รูปที่ 2.44 โครงข่ายแบบพายและโครงข่ายแบบที 49
รูปที่ 2.45 การแมตช์ชิ่งโดยใช้โครงข่ายแบบพาย 49
รูปที่ 2.46 การแมตช์ชิ่งโดยใช้โครงข่ายแบบที 50
รูปที่ 2.47 ไดอะแกรมแสดงโครงสร้างทางฮาร์ดแวร์ของพอร์ตอนุกรม 50
รูปที่ 3.1 การเชื่อมต่อกันระหว่างเครื่องคอมพิวเตอร์ในระบบ 62
รูปที่ 3.2 ลำดับการทำการ Poll จากเครื่อง Server 63
รูปที่ 3.3 ลำดับการส่งข้อมูลจากเครื่องลูกข่าย 1 ไปยังเครื่องลูกข่าย 3 โดยผ่านเครื่อง Server 63
รูปที่ 3.4 ลักษณะ Frame ของสัญญาณคำสั่ง 64
รูปที่ 3.5 ลักษณะ Frame ของสัญญาณข้อมูล 66
รูปที่ 3.6 บล็อกไดอะแกรมของไอซีเบอร์ XR2206 68
รูปที่ 3.7 วงจรฟรีเควนซีชิพท์คีย์อิ้งมอดูเลเตอร์ 69
รูปที่ 3.8 บล็อกไดอะแกรมของไอซีเบอร์ XR2211 70
รูปที่ 3.9 วงจรฟรีเควนซีชิพท์คีย์อิ้งดีมอดูเลเตอร์ 72

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญภาพ (ต่อ)

		หน้า
รูปที่ 3.10	วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์	73
รูปที่ 3.11	วงจรพรีสเกลเลอร์แบบสองโมดูลัส	74
รูปที่ 3.12	วงจรเปรียบเทียบเฟสและการตั้งโปรแกรมหารความถี่	75
รูปที่ 3.13	วงจรลูปฟิลเตอร์	77
รูปที่ 3.14	บล็อกไดอะแกรมของไอซี MC3362	78
รูปที่ 3.15	วงจรฟรีเควนซีดีมอดูเลเตอร์	78
รูปที่ 3.16	วงจรสังเคราะห์ความถี่	79
รูปที่ 3.17	วงจรขยายแบบดาร์ลิงตัน	80
รูปที่ 3.18	วงจรขยายความถี่ช่วงกว้าง	81
รูปที่ 3.19	วงจรแหล่งจ่ายกำลัง DC	81
รูปที่ 3.20	หลักการทำงานของ Bridge Rectifier แบบไดโอด	82
รูปที่ 3.21	ตัวอย่างของวงจร Rectifier	82
รูปที่ 4.1	Server ทำการส่งสัญญาณ Poll ไปยังเครื่องลูกข่ายที่ 1	83
รูปที่ 4.2	Server ทำการ Poll ไปยังเครื่องถัดไป	84
รูปที่ 4.3	Server ได้รับข้อมูลจากเครื่องลูกข่าย 1 ครั้งที่ 1	85
รูปที่ 4.4	Server ได้รับข้อมูลจากเครื่องลูกข่าย 1 ครั้งที่ 2	86
รูปที่ 4.5	Server ส่งข้อมูลครั้งที่ 1 ไปยังเครื่องลูกข่าย 3	87
รูปที่ 4.6	Server ส่งข้อมูลครั้งที่ 2	88
รูปที่ 4.7	Server ได้รับสัญญาณจากเครื่องลูกข่าย 3	89
รูปที่ 4.8	โปรแกรมของเครื่องลูกข่าย	90
รูปที่ 4.9	เครื่องลูกข่าย 000001 รับสัญญาณ Poll จากเครื่องลูกข่าย 000010	90
รูปที่ 4.10	เครื่องลูกข่าย 000001 รับสัญญาณ Poll จากเครื่องลูกข่าย 000011	91
รูปที่ 4.11	การส่งสัญญาณNACK	91
รูปที่4.12	เครื่องลูกข่ายส่งสัญญาณ ACK1 ไปเครื่อง Server	92
รูปที่ 4.13	ผู้ใช้เลือกไฟล์ข้อมูลที่ต้องการจะส่ง	92
รูปที่ 4.14	การส่งข้อมูล 1,000 bytes แรก	93
รูปที่ 4.15	การส่งข้อมูล 1,000 bytes ชุดที่ 2	93
รูปที่ 4.16	การส่งข้อมูลชุดสุดท้ายของไฟล์ข้อมูล	94

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญภาพ (ต่อ)

หน้า

รูปที่ 4.17 เครื่อง Server ส่งสัญญาณ ACK2 94

รูปที่ 4.18 การบันทึกไฟล์ข้อมูล 95

รูปที่ 4.19 เครื่องลูกข่ายส่งสัญญาณ ACK2 ไปเครื่อง Server 95

รูปที่ 4.20 ระดับแรงดันของขา Tx ของพอร์ตอนุกรมเมื่อไม่มีการส่งข้อมูล 96

รูปที่ 4.21 สัญญาณฟรีเควนซีชิพท์คีย์อิ้งที่มีความถี่ 4800 Hz และ 9600 Hz 97

รูปที่ 4.22 สัญญาณที่ได้หลังจากการดีมอดูเลตฟรีเควนซีชิพท์คีย์อิ้ง 97

รูปที่ 4.23 สัญญาณไซน์ที่ผลิตได้ที่ความถี่กลางเท่ากับ 50 เมกะเฮิร์ตซ์ 98

รูปที่ 4.24 สเปกตรัมของสัญญาณที่ผลิตได้จากวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ 99

รูปที่ 4.25 Characteristic ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ 99

รูปที่ 4.26 สเปกตรัมของสัญญาณเอฟเอ็ม เมื่ออินพุตเป็นสัญญาณไซน์ความถี่ 10 kHz ขนาด 100 mV_{PP} 100

รูปที่ 4.27 สเปกตรัมของสัญญาณเอฟเอ็มเมื่ออินพุตเป็นสัญญาณไซน์ความถี่ 10 kHz ขนาด 200 mV_{PP} 100

รูปที่ 4.28 สเปกตรัมของสัญญาณเอฟเอ็มเมื่ออินพุตเป็นสัญญาณไซน์ความถี่ 20 kHz ขนาด 100 mV_{PP} 101

รูปที่ 4.29 สัญญาณที่ได้จากหารความถี่ไป 64 เท่า 102

รูปที่ 4.30 สัญญาณที่ขา Lock Detect เมื่อวงจรสังเคราะห์ความถี่อยู่ในสภาวะล็อก 102

รูปที่ 4.31 ระดับแรงดันที่ออกจากลูปฟิลเตอร์ 103

รูปที่ 4.32 CH1 แสดงสัญญาณฟรีเควนซีชิพท์คีย์อิ้งที่ป้อนให้กับวงจรมอดูเลเตอร์ 104
CH2 แสดงสัญญาณเอาท์พุตที่ได้จากไอซี MC3362

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

	หน้า
ตารางที่ 2.1 ตัวอย่างบิตพาริตี้ของข้อมูล	6
ตารางที่ 2.2 ตารางความสัมพันธ์ระหว่างการกระจายของสเปกตรัมกับมอดูเลชันอินเด็กซ์	22
ตารางที่ 2.3 ข้อมูลในแอดเดรส 0000:0411H ที่ใช้แจ้งจำนวนพอร์ตอนุกรม	51
ตารางที่ 2.4 ค่าคงที่สำหรับคุณสมบัติ Handshake	59
ตารางที่ 2.5 ค่าคงที่สำหรับคุณสมบัติ OnComm	60
ตารางที่ 2.6 ค่าคงที่สำหรับคุณสมบัติ Error	60
ตารางที่ 2.7 ค่าคงที่สำหรับคุณสมบัติ InputMode	61
ตารางที่ 3.1 การโปรแกรมค่าให้กับวงจรหารภายในไอซี MC145152-2	74

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

ปริญญานิพนธ์ระบบโครงข่ายท้องถิ่นแบบไร้สายนี้ เป็นการสร้างและออกแบบระบบการสื่อสารข้อมูลระหว่างคอมพิวเตอร์ ซึ่งในการเชื่อมต่อกันระหว่างเครื่องคอมพิวเตอร์จะใช้คลื่นวิทยุส่งผ่านตัวกลางที่เป็นอากาศ โดยเริ่มต้นทำการออกแบบระบบโครงข่ายท้องถิ่นแบบไร้สายที่มีเครื่องคอมพิวเตอร์ในระบบจำนวน 64 เครื่อง กำหนดให้มีคอมพิวเตอร์ที่ทำหน้าที่เป็นเครื่องควบคุม (Server) จำนวน 1 เครื่อง และคอมพิวเตอร์ที่ทำหน้าที่เป็นเครื่องลูกข่าย (Client) จำนวน 63 เครื่อง แต่ทั้งนี้ในการประยุกต์ใช้สามารถเพิ่มจำนวนของเครื่องคอมพิวเตอร์ในระบบได้อีกโดยทำการเปลี่ยนแปลงโปรแกรมควบคุมบางส่วน ระบบที่ออกแบบมีลักษณะของการเชื่อมต่อระหว่างคอมพิวเตอร์เป็นโครงข่ายรูปดาว ซึ่งการติดต่อกันระหว่างเครื่องคอมพิวเตอร์จะเป็นการสื่อสารกันผ่านทางพอร์ตอนุกรมของแต่ละเครื่องโดยใช้คลื่นวิทยุเป็นตัวนำข้อมูลข่าวสารไป

ปริญญานิพนธ์ชิ้นนี้ถูกสร้างและพัฒนาขึ้นมา เพื่อที่จะทำการศึกษาเกี่ยวกับการสื่อสารข้อมูลในระบบโครงข่ายท้องถิ่น โดยใช้พื้นฐานความรู้ของการสื่อสารข้อมูล ความรู้ทางด้านอิเล็กทรอนิกส์ในส่วนของเครื่องส่งและเครื่องรับคลื่นวิทยุ และความรู้ของการติดต่อสื่อสารแบบไร้สาย เนื่องจากผู้จัดทำได้เล็งเห็นถึงความสำคัญของเทคโนโลยีทางด้านนี้ ซึ่งนับวันจะยิ่งทวีความสำคัญในการสื่อสารโทรคมนาคมเพิ่มมากขึ้น ดังนั้นจึงได้สร้างโครงงานชิ้นนี้ขึ้นมาและได้ศึกษาในสาขาวิชาที่เกี่ยวข้อง เพื่อเป็นพื้นฐานความรู้ความเข้าใจและสามารถตอบสนองได้ทันกับการพัฒนาเทคโนโลยีทางด้านโทรคมนาคมในยุคปัจจุบันและอนาคตต่อไป

โดยเนื้อหาในปริญญานิพนธ์ฉบับนี้ จะประกอบด้วย

บทที่ 2 กล่าวถึงหลักการและทฤษฎีต่างๆ ที่เกี่ยวข้องและนำมาประยุกต์ใช้

บทที่ 3 กล่าวถึงวิธีการออกแบบและคำนวณตัวแปรต่างๆ ในวงจร

บทที่ 4 กล่าวถึงผลการทดลองที่ได้จากการออกแบบวงจร

บทที่ 5 กล่าวถึงปัญหาที่พบในชิ้นงาน และแนวทางการประยุกต์

ผู้จัดทำหวังเป็นอย่างยิ่งว่าปริญญานิพนธ์ฉบับนี้จะเป็นแนวทางสำหรับการพัฒนาระบบโครงข่ายท้องถิ่นแบบไร้สาย และให้ความรู้กับผู้จัดทำและผู้ที่มีความสนใจในเบื้องต้น โดยปริญญานิพนธ์นี้เป็นเพียงรูปแบบหนึ่งของการออกแบบ ซึ่งสามารถนำไปพัฒนาเป็นระบบอื่นๆ ได้ต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎีและหลักการ

2.1 การสื่อสารข้อมูล (Data Communication)

การสื่อสารข้อมูล หมายถึง การแลกเปลี่ยนข้อมูลระหว่างอุปกรณ์ 2 ชนิด ข้อมูลในที่นี้มักได้แก่ ข้อมูลดิจิตอลที่มีค่า “0” และ “1” องค์ประกอบหลักที่สำคัญในการสื่อสารข้อมูลคือ ผู้ส่งข่าวสาร (Sender) ผู้รับข่าวสาร (Receiver) และตัวกลางในการส่งผ่านข้อมูล (Medium) โดยช่องทางการสื่อสารข้อมูลระหว่างอุปกรณ์ 2 ตัว แบ่งออกได้เป็น 3 ประเภท

1. *การสื่อสารทางเดียว (Simplex)* เป็นการส่งแบบที่อุปกรณ์ปลายทางแต่ละด้านจะทำหน้าที่เป็นเฉพาะตัวส่งหรือตัวรับอย่างใดอย่างหนึ่งเท่านั้น เมื่ออุปกรณ์ชุดหนึ่งส่งข้อมูล อุปกรณ์อีกชุดจะต้องเป็นฝ่ายรับข้อมูลเสมอ จะไม่มีการเปลี่ยนทิศทางของข้อมูล

2. *การสื่อสารกึ่งสองทาง (Half Duplex)* จะมีการใช้อุปกรณ์ปลายทางที่พัฒนาขึ้นมาให้มีความสามารถทั้งรับและส่งได้ แต่ไม่สามารถรับและส่งพร้อมกันได้ ต้องผลัดกันเป็นตัวรับและตัวส่ง กล่าวคือ มีการเปลี่ยนเส้นทางในการส่งข้อมูลได้แต่คนละเวลา หรือข้อมูลจะไหลไปในทิศทางเดียว ณ เวลาใดๆ

3. *การสื่อสารสองทาง (Full Duplex)* ทั้งสองด้านสามารถส่งและรับข้อมูลได้ในเวลาเดียวกัน กล่าวคืออุปกรณ์ปลายทางเป็นผู้รับและผู้ส่งข้อมูลในเวลาเดียวกันได้

อุปกรณ์สื่อสารจะแบ่งลำดับความสำคัญในการรับส่งข้อมูลออกเป็น 2 ระดับ คือ

1. *สถานีต้นทาง (Primary Station)* คือ อุปกรณ์สื่อสารหรือเครื่องคอมพิวเตอร์ที่มีสิทธิในการรับส่งข้อมูลสูงกว่า กล่าวคือ มีสิทธิทำการร้องขอเพื่อที่จะทำการรับส่งข้อมูลได้

2. *สถานีปลายทาง (Secondary Station)* คือ อุปกรณ์สื่อสารหรือเครื่องคอมพิวเตอร์ที่ทำหน้าที่ในการรับคำสั่งและทำการตอบสนองต่อสถานีต้นทางเท่านั้น

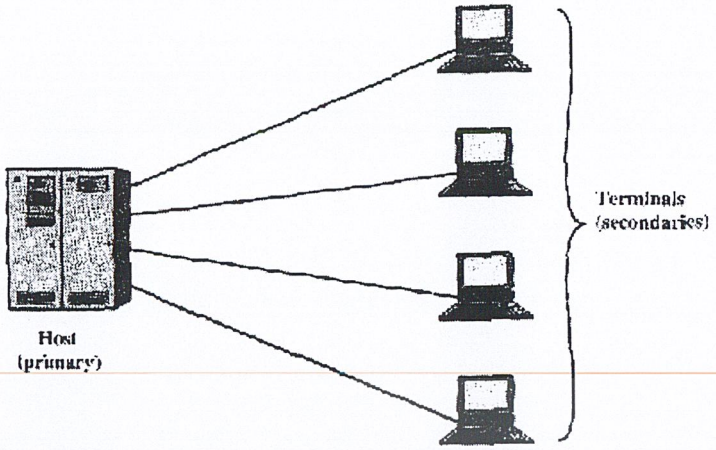
รูปแบบโครงร่างของข่ายสาย (Line Configurations) หรือลักษณะโครงสร้างในการเชื่อมต่อข้อมูลจากจุดหนึ่งไปยังอีกจุดหนึ่ง แบ่งออกเป็น 2 ลักษณะ คือ

1. *Point-to-point* เป็นกระบวนการเชื่อมต่ออุปกรณ์สื่อสารคู่ใดคู่หนึ่ง ซึ่งมีความเป็นอิสระต่อกันจากการเชื่อมต่อของอุปกรณ์สื่อสารคู่อื่นๆ

2. *Multipoint* เป็นกระบวนการเชื่อมต่อระหว่างอุปกรณ์สื่อสาร โดยที่อุปกรณ์สื่อสารทุกเครื่องมีการเชื่อมต่อถึงกันหมด

โดยรูปแบบโครงร่างของข่ายสายสามารถแสดงได้ดังรูปที่ 2.1 และ 2.2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.1 โครงร่างของข่ายสายแบบ Point-to-point

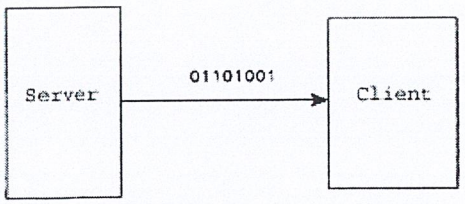


รูปที่ 2.2 โครงร่างของข่ายสายแบบ Multipoint

รูปแบบของการสื่อสารข้อมูล เป็นกระบวนการในการนำกลุ่มของบิตข้อมูลจากอุปกรณ์หนึ่งไปยังอีกอุปกรณ์หนึ่งหรือการส่งผ่านข้อมูลจากอุปกรณ์ส่งไปยังอุปกรณ์รับ แบ่งออกเป็น 2 แบบ คือ การสื่อสารข้อมูลแบบขนาน และการสื่อสารข้อมูลแบบอนุกรม ซึ่งมีลักษณะการส่งข้อมูลดังแสดงในรูปที่ 2.3



(ก) แบบขนาน



(ข) แบบอนุกรม

รูปที่ 2.3 รูปแบบของการสื่อสารข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

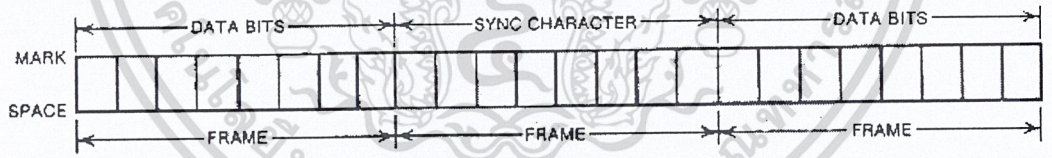
2.1.1 การสื่อสารข้อมูลแบบขนาน

การสื่อสารข้อมูลแบบขนาน เป็นการรับส่งข้อมูลทีละหลายๆ บิตออกจากอุปกรณ์ส่งไปยังอุปกรณ์รับพร้อมๆ กัน และช่องสัญญาณที่ใช้ในการรับส่งจะต้องมีอย่างน้อยเท่ากับจำนวนบิตที่ส่งออกไปพร้อมกันนั้น เพื่อใช้กับสัญญาณข้อมูลแต่ละบิตและสัญญาณควบคุมต่างๆ ที่ใช้ในการสื่อสาร ข้อดีของการสื่อสารแบบขนานคือ สามารถรับส่งข้อมูลได้อย่างรวดเร็วและเป็นจำนวนมาก ส่วนข้อเสียคือ ไม่เหมาะที่จะนำไปใช้ในการสื่อสารข้อมูลระยะไกล เนื่องจากค่าใช้จ่ายของสายนำสัญญาณมีราคาแพง และระยะทางที่มากอาจส่งผลให้เกิดความผิดพลาดของข้อมูลได้ อันมีสาเหตุมาจากการลดทอนภายในสายนำสัญญาณ ความผิดเพี้ยนของสัญญาณเนื่องจากสภาพความเป็นประจุภายในสาย สภาพความไม่สมบูรณ์ของตัวนำภายในสาย ระดับกราวนด์ (Ground) ทางไฟฟ้าที่อุปกรณ์รับผิดไปจากอุปกรณ์ส่ง เป็นต้น

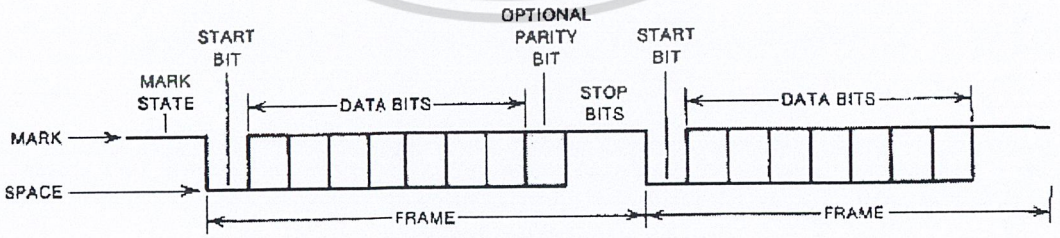
2.1.2 การสื่อสารข้อมูลแบบอนุกรม

การสื่อสารข้อมูลแบบอนุกรม ข้อมูลจะถูกส่งออกไปทีละบิตต่อเนื่องกันไป โดยการรับส่งจะใช้เพียงช่องสัญญาณเดียวเท่านั้น ทางด้านส่งจะส่งข้อมูลออกจากพอร์ตเรียงกันออกไปทีละบิต ส่วนทางด้านรับจะรับข้อมูลเข้ามาทีละบิตเช่นกันและทำการตรวจสอบบิตที่รับเข้ามาว่าบิตใดเป็นบิตเริ่มต้นบิตใดเป็นบิตสิ้นสุด ซึ่งมีข้อดีคือ ประหยัดสายนำสัญญาณ ส่วนข้อเสียคือ การส่งข้อมูลจะช้ากว่าการสื่อสารข้อมูลแบบขนาน เนื่องจากต้องส่งข้อมูลทีละบิตตามลำดับ

การสื่อสารข้อมูลแบบอนุกรมนั้นจะแบ่งออกได้เป็น 2 แบบ ได้แก่ แบบซิงโครนัส (Synchronous Transmission) และแบบอะซิงโครนัส (Asynchronous Transmission) ซึ่งมีรายละเอียดดังต่อไปนี้



(A) Synchronous Transmission



(B) Asynchronous Transmission

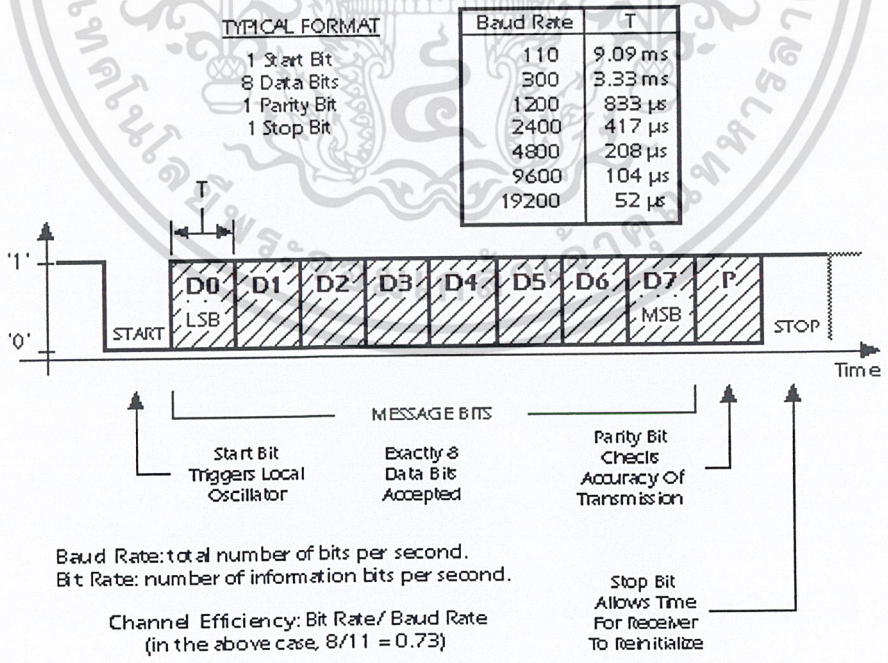
รูปที่ 2.4 รูปแบบเฟรมข้อมูลของการสื่อสารข้อมูลแบบอนุกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. *การสื่อสารข้อมูลอนุกรมแบบซิงโครนัส* จะไม่ใช้บิตเริ่มต้นและบิตสิ้นสุด และจะไม่มีการหยุดชั่วขณะระหว่างตัวอักษร (Character) แต่จะใช้วิธีให้จังหวะเวลาทั้งสองทางที่ติดต่อกัน มีอยู่สองวิธีที่ปฏิบัติกันคือ ใช้ตัวอักษรซิงค์ (Sync character) และใช้สัญญาณนาฬิกา (Clock signal) วิธีการใช้ตัวอักษรซิงค์จะใส่ไว้หน้าบล็อกของข้อมูล โดยตัวอักษรซิงค์นี้เป็นบิตจำนวนหนึ่งซึ่งทางอุปกรณ์เครื่องรับสามารถใช้ในการกำหนดอัตราเร็วของข้อมูลให้ตรงกับทางอุปกรณ์เครื่องส่ง ส่วนวิธีการใช้สัญญาณนาฬิกาจะใช้คนละสายหรือคนละช่องสัญญาณในการส่งข่าวสารเกี่ยวกับเวลาของข้อมูลที่จะส่ง ดังนั้นการติดต่อกันแบบนี้จะต้องใช้สายในการเชื่อมต่ออย่างน้อยที่สุด 3 เส้น คือ สายสัญญาณนาฬิกา, สายส่งข้อมูล และสายกราวนด์

2. *การสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัส* คือ การรับและส่งข้อมูลไปโดยไม่จำเป็นต้องมีสัญญาณนาฬิการ่วมด้วยเหมือนกับการรับส่งข้อมูลอนุกรมแบบซิงโครนัส แต่จะใช้การกำหนดค่าสัญญาณนาฬิกาทั้งภาครับและภาคส่งให้มีค่าเท่ากัน อัตราการถ่ายทอดข้อมูลของการสื่อสารแบบอนุกรมเรียกว่า บอดเรต (Baud Rate) มีหน่วยเป็นบิตต่อวินาที (Bit per second หรือ Bps) โดยรูปแบบของข้อมูลที่ใช้ในการรับส่งแบบอะซิงโครนัสประกอบด้วย 4 ส่วน ดังนี้

- บิตเริ่มต้น (Start Bit) ซึ่งจะมีขนาด 1 บิต
- บิตข้อมูลแบบอนุกรมจะมีขนาด 5,6,7 หรือ 8 บิต
- บิตตรวจสอบพาริตี้ (Parity Bit) จะมีขนาด 1 บิตหรือไม่มี
- บิตสิ้นสุด (Stop Bit) จะมีขนาด 1, 1.5 หรือ 2 บิต



รูปที่ 2.5 รูปแบบของข้อมูลอนุกรมแบบอะซิงโครนัส

เมื่อไม่มีข้อมูลจะส่ง ขาดาต้าจะมีสถานะลอจิก “1” ซึ่งจะเรียกสถานะนี้ว่าสถานะหยุดรอ (Waiting stage) การเริ่มต้นส่งข้อมูลจะเริ่มจากการให้ขาดาต้ามีลอจิก “0” ด้วยช่วงระยะเวลา 1 บิต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ซึ่งจะเรียกบิตนี้ว่าบิตเริ่มต้น จากนั้นบิตข้อมูลจะถูกส่งออกไป โดยเริ่มจากบิตที่มีนัยสำคัญต่ำสุด (LSB) ก่อน ซึ่งข้อมูลที่ส่งอาจจะมีจำนวนบิต 5,6,7 หรือ 8 บิตก็ได้ จากนั้นจะตามด้วยบิตพาริตี้ ซึ่งใช้เพื่อตรวจสอบความผิดพลาดที่เกิดขึ้นจากการส่งข้อมูล บิตสุดท้ายที่จะส่งคือบิตสิ้นสุด ซึ่งจะให้ขาดาต้ามีสถานะลอจิก "1" อีกครั้งด้วยระยะเวลาอย่างน้อย 1 บิต, 1.5 บิต หรือ 2 บิต เพื่อเป็นการแสดงว่าสิ้นสุดข้อมูลแล้ว โดยอัตราความเร็วในการรับและส่งข้อมูลของการรับส่งข้อมูลแบบอะซิงโครนัสหรือค่าบอดเรตมาตรฐานที่ใช้ได้แก่ 110, 150, 300, 600, 1200, 2400, 4800, 9600 และ 19200 บิตต่อวินาที

การตรวจสอบพาริตี้สามารถกำหนดให้เป็นแบบคี่ (odd), แบบคู่ (even) หรือไม่มีการตรวจสอบพาริตี้ (none) ก็ได้ การตรวจสอบพาริตี้เป็นการตรวจสอบจำนวนรวมของบิตที่เป็นลอจิก "1" ภายในข้อมูลที่ส่งไป 1 ไบต์ว่ามีจำนวนรวมเป็นเลขคู่หรือเลขคี่โดยต้องรวมบิตพาริตี้เข้าไปด้วย ในตารางที่ 2.1 แสดงตัวอย่างของบิตพาริตี้ในการรับส่งข้อมูลอนุกรม

ข้อมูล	บิตพาริตี้คู่	บิตพาริตี้คี่
00000000	0	1
00000001	1	0
00000010	1	0
00000011	0	1
00000100	1	0
11111110	0	1
11111111	1	0

ตารางที่ 2.1 ตัวอย่างบิตพาริตี้ของข้อมูล

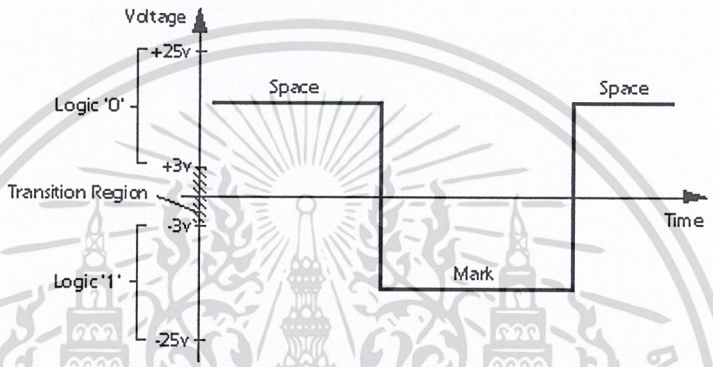
ส่วนทางภาครับจะต้องทำการกำหนดคุณสมบัติการตรวจสอบพาริตี้ให้ตรงกันว่าจะตรวจสอบพาริตี้คี่หรือพาริตี้คู่ เมื่อได้รับข้อมูลก็จะทำการตรวจสอบค่าพาริตี้ที่เกิดขึ้นว่าเป็นคู่หรือเป็นคี่ โดยการนับจำนวนลอจิก "1" ทั้งหมดรวมทั้งบิตพาริตี้ด้วย ถ้ากำหนดพาริตี้ไว้เป็นคู่แต่อ่านค่าตัวเลขในการนับออกมาได้ตัวเลขเป็นคี่ทางภาครับจะแสดงข้อมูลผิดพลาดออกมาให้ผู้ใช้ทราบ นับเป็นการตรวจสอบความผิดพลาดที่เกิดขึ้นในการถ่ายทอดข้อมูลที่ง่ายที่สุด แต่จะเชื่อถือได้เมื่อมีบิตข้อมูลที่ทำการส่งผิดพลาดเพียงบิตเดียวหรือผิดพลาดเป็นจำนวนคี่เท่านั้น ถ้าข้อมูลที่ทำการส่งมีบิตที่ผิดพลาดมากกว่า 1 บิต การตรวจสอบด้วยวิธีนี้จะไม่ได้ผล สำหรับการตั้งพาริตี้บิตเป็น none นั้นทั้งภาครับและภาคส่งจะไม่มีการตรวจสอบพาริตี้

2.2 มาตรฐานพอร์ตอนุกรมแบบ RS232

RS232 หรือ V.24 เป็นมาตรฐานการเชื่อมต่อระหว่างอุปกรณ์รับส่งปลายทาง (Data Terminal Equipment หรือ DTE) กับอุปกรณ์สื่อสารข้อมูล (Data Circuit Terminating Equipment หรือ DCE) เพื่อใช้ในการส่งข้อมูลไบนารีแบบอนุกรม โดยที่ RS232 เป็นมาตรฐานที่กำหนดขึ้นมาจากสมาคมอิเล็กทรอนิกส์

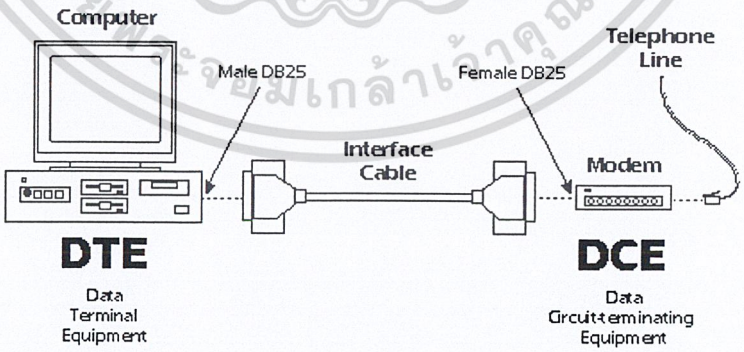
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อุตสาหกรรม (Electronic Industries Association หรือ EIA) ซึ่งขึ้นจากความร่วมมือกับบริษัท Bell ตัวอักษรภาษาอังกฤษของคำว่า "RS" ย่อมาจาก "Recommended Standard" แล้วตามด้วยเลข 3 ตัว และปิดท้ายด้วยตัวเลขทั้ง 3 ด้วยตัวอักษรภาษาอังกฤษ A, B, C, และ D เป็นการแสดงถึงรุ่นที่ได้รับการพัฒนามาตรฐาน RS232 ที่สร้างขึ้นโดยสมาคมอิเล็กทรอนิกส์อุตสาหกรรมจะเทียบเท่ากับมาตรฐาน V ที่สร้างขึ้นโดยองค์การสหภาพโทรคมนาคมระหว่างประเทศ (Consultative Committee on International Telephone and Telegraph หรือ CCITT) มาตรฐานนี้ในช่วงแรกจะใช้คอนเน็กเตอร์แบบ DB-25 โดยกำหนดความยาวสูงสุดของสายสัญญาณไว้ที่ 50 ฟุต มีระดับสัญญาณตั้งแต่ –3 ถึง – 12V แสดงว่ามีข้อมูล (Mark) และ +3 ถึง +12V แสดงว่าเป็นช่องว่าง (Space)



รูปที่ 2.6 ระดับของแรงดันตามมาตรฐาน RS232

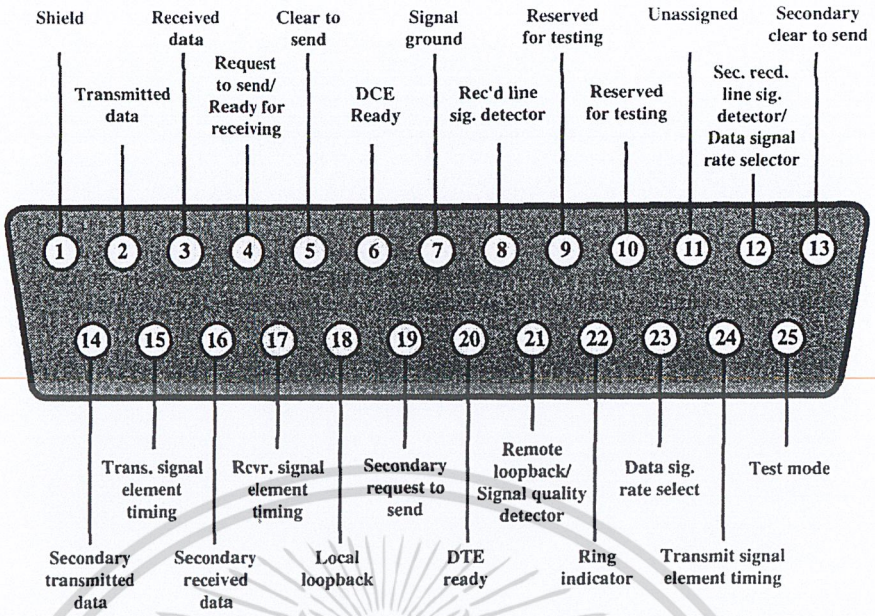
มาตรฐาน RS232 ได้กำหนดรูปแบบของอุปกรณ์รับส่งปลายทางกับอุปกรณ์สื่อสารข้อมูลไว้ว่าอุปกรณ์รับส่งปลายทางจะต้องเป็นอุปกรณ์ที่มีการประมวลผลในตัวเช่น ไมโครคอนโทรลเลอร์หรือไมโครคอมพิวเตอร์ ซึ่งมีความสามารถในการสร้างบิตข้อมูลแบบอนุกรมได้ ส่วนอุปกรณ์สื่อสารข้อมูลจะทำหน้าที่เป็นเพียงตัวรับข้อมูลที่ส่งมาจากอุปกรณ์รับส่งปลายทางเท่านั้น โดยการรับส่งข้อมูลระหว่างอุปกรณ์ทั้งสองจะกระทำผ่านมาตรฐาน RS232



รูปที่ 2.7 รูปแบบของอุปกรณ์เชื่อมต่อข้อมูลกับวงจรข้อมูลปลายทาง

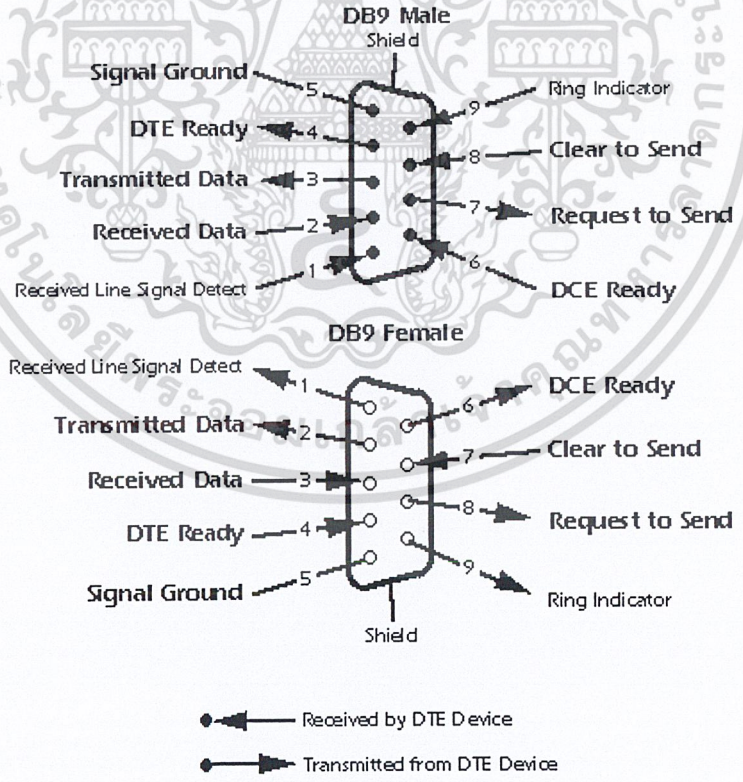
มาตรฐานการเชื่อมต่อแบบ RS232 จะใช้คอนเน็กเตอร์แบบ DB-25 หรือ DB-9 ซึ่งคอนเน็กเตอร์แบบ DB-25 จะมีขาต่อใช้งานเพียง 9 เส้นเช่นเดียวกับคอนเน็กเตอร์แบบ DB-9 เนื่องจากขาอื่นๆ ที่เคยใช้งานในอดีต ปัจจุบันมีการใช้งานไม่มากนักจึงถูกยกเลิกไป โดยรูปร่างและตำแหน่งขาของคอนเน็กเตอร์แบบ DB-25 เป็นดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.8 รูปร่างและตำแหน่งขาของคอนเน็กเตอร์แบบ DB-25

ส่วนรูปร่างและตำแหน่งขาของคอนเน็กเตอร์แบบ DB-9 โดยคอนเน็กเตอร์ตัวผู้สำหรับอุปกรณ์รับส่งปลายทาง และคอนเน็กเตอร์ตัวเมียสำหรับอุปกรณ์สื่อสารข้อมูล แสดงได้ดังรูปที่ 2.9



รูปที่ 2.9 รูปร่างและตำแหน่งขาของคอนเน็กเตอร์แบบ DB-9

สำหรับรายละเอียดหน้าที่การทำงานในแต่ละขาของพอร์ตอนุกรม RS-232 ที่ใช้คอนเน็กเตอร์แบบ DB-9 เป็นดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- *Data Carrier Detect หรือ Carrier Detect : DCD* ขานี้จะแอกตีฟเมื่อมีการส่งสัญญาณพาห์จากอุปกรณ์สื่อสารข้อมูล เช่น โมเด็ม สำหรับการใช้งานปกติ ขานี้จะไม่ได้ถูกใช้งานมากนัก

- *Received Data : RD หรือ RxD* ขานี้ใช้เพื่อรับสัญญาณอนุกรมเข้ามายังคอมพิวเตอร์โดยจะนำข้อมูลที่อ่านได้เก็บไว้ในรีจิสเตอร์บัฟเฟอร์

- *Transmitted Data : TD หรือ TxD* ขานี้ใช้เพื่อส่งข้อมูลออกจากคอมพิวเตอร์ โดยจะนำข้อมูลที่เก็บอยู่ในบัฟเฟอร์สำหรับส่งข้อมูลส่งออกไป

- *Data Terminal Ready : DTR* เป็นขาสัญญาณที่ส่งออกจากคอมพิวเตอร์เพื่อให้อุปกรณ์ปลายทางรับรู้ว่า ต้องการติดต่อด้วย โดยขา DTR นี้จะต้องเชื่อมต่อกับขา DSR ของอุปกรณ์ปลายทางและขา DTR ของอุปกรณ์ปลายทางจะต้องเชื่อมต่อกับขา DSR ของคอมพิวเตอร์ ถ้าใช้การเชื่อมต่อเป็นแบบ Null Modem ซึ่งใช้สายในการเชื่อมต่อเพียง 3 เส้น จะต้องต่อขา DTR และ DSR ของตัวมันเองเข้าด้วยกัน และต้องต่อขา DCD ด้วยในกรณีที่โปรแกรมสื่อสารที่ใช้มีการตรวจจับสัญญาณพาห์

- *Signal Ground : GND* เป็นขากราวด์ของระบบ

- *Data Set Ready : DSR* ขานี้จะใช้คู่กับขา DTR เพื่อตรวจสอบการเชื่อมต่อกันระหว่างคอมพิวเตอร์กับอุปกรณ์ปลายทาง ซึ่งขานี้จะเป็นขาที่รับข้อมูลจากภายนอกซึ่งถูกส่งมาจากขา DTR

- *Request To Send : RTS* เป็นขาสำหรับส่งสัญญาณร้องขอให้ทางอุปกรณ์ปลายทางส่งข้อมูลกลับมายังคอมพิวเตอร์ โดยขาที่รับสัญญาณ RTS ก็คือขา CTS ในกรณีที่ใช้การเชื่อมต่อแบบ Null modem 3 สาย จะต้องเชื่อมต่อขา RTS และ CTS ของตัวมันเองเข้าด้วยกัน เพื่อจะให้การรับและส่งข้อมูลสามารถเกิดขึ้นได้ตลอดเวลา

- *Clear To Send : CTS* ขานี้จะคอยรับสัญญาณจากขา RTS เมื่อรับสัญญาณได้ ข้อมูลที่ขา TxD จะถูกส่งออกไป เพื่อตรวจสอบอุปกรณ์ต่อพ่วงว่าพร้อมที่จะรับข้อมูลหรือไม่

- *Ring Indicator : RI* ใช้แสดงสถานะสัญญาณเรียกจากสายโทรศัพท์ โดยทั่วไปสายนี้จะไม่ถูกใช้งาน จะใช้งานก็ต่อเมื่อโปรแกรมมีการตรวจสอบสัญญาณนี้เท่านั้น

2.3 ระบบโครงข่ายท้องถิ่น

ระบบโครงข่ายท้องถิ่น หมายถึง โครงข่ายคอมพิวเตอร์ขนาดเล็กที่เป็นของกลุ่มผู้ใช้กลุ่มเล็กๆ กลุ่มหนึ่ง ปกติจะเป็นโครงข่ายที่มีขอบเขตอยู่ภายในอาคารเดียวกันหรือกลุ่มอาคารที่อยู่ติดกัน มีระยะทางไม่เกิน 2-3 กิโลเมตร เหมาะสำหรับการเชื่อมต่อเครื่องคอมพิวเตอร์ขนาดเล็กของพนักงานในองค์กรเข้าด้วยกัน โดยมีวัตถุประสงค์หลักคือ รวมผู้ใช้งานที่มีการทำงานในรูปแบบเดียวกันให้มาใช้ทรัพยากรหรืออุปกรณ์ส่วนกลางร่วมกัน เช่นเครื่องพิมพ์เลเซอร์สีขนาดใหญ่ เพื่อให้ไม่เกิดการสิ้นเปลืองและเกิดประโยชน์ในการใช้งานสูงสุด ลักษณะเด่นที่สำคัญของระบบโครงข่ายท้องถิ่น ได้แก่

1. ระยะทางของเส้นทางส่งที่ประกอบเป็นโครงข่ายนั้นจะสั้น ทำให้สัดส่วนของค่าใช้จ่ายสำหรับเส้นทางส่งระบบโครงข่ายท้องถิ่นเมื่อเปรียบเทียบกับระบบแล้วจะน้อย ดังนั้นสามารถเลือกใช้เส้นทางส่งที่มีความจุสูงและค่อนข้างแพงได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ค่าเวต (Weight) ของค่าใช้จ่ายในเส้นทางส่งของโครงข่ายโดยรวมจะสูงเมื่อเปรียบเทียบกับโครงข่ายสาธารณะที่เพิ่มประสิทธิภาพในการส่งด้วยการมัลติเพลกซ์แล้วสามารถใช้กระบวนการควบคุมสื่อสารอย่างง่ายๆ โดยที่ไม่ลดประสิทธิภาพในการส่งทำให้ระบบโครงข่ายท้องถิ่นโดยทั่วไปนั้นจะใช้เพียง 10%-50% ของความจุในการส่งสูงสุด

3. เมื่อเปรียบเทียบกับโครงข่ายสาธารณะแล้วระยะทางในการส่งจะสั้นพร้อมกันนั้นสภาพแวดล้อมของการติดตั้งจะไม่เปลี่ยนแปลง ทำให้สามารถใช้ในการส่งข้อมูลด้วยความเร็วสูงและมีคุณภาพสูงที่จะเกิดความผิดพลาดในการส่งน้อย ซึ่งจะส่งผลให้สามารถลดการควบคุมสื่อสารที่ยุ่งยากเช่นการควบคุมความผิดพลาดลงได้

4. ระบบโครงข่ายท้องถิ่นโดยส่วนใหญ่แล้ว จะไม่ใช้การสื่อสารแบบคอนเนคชั่น คือจะไม่ใช้วิธีการส่งข้อมูลหลังจากที่มีการติดตั้งเส้นทางสื่อสารทางลอจิก (Data Link) ระหว่างคู่สื่อสาร แต่จะส่งข่าวสารในรูปแบบแพคเกต (Packet) ที่ประกอบด้วยแอดเดรสผู้รับรวมอยู่ไปยังทุกเทอร์มินัลที่เชื่อมต่อกับเนตเวอร์คและเทอร์มินัลจะรับเอาข่าวสารเฉพาะที่เป็นของตัวเองเท่านั้น วิธีการนี้เรียกว่าเป็นการสื่อสารแบบคอนเนคชั่นเลส กล่าวอีกนัยหนึ่งก็คือแต่ละเทอร์มินัลที่เชื่อมต่ออยู่กับ ระบบโครงข่ายท้องถิ่นนั้นจะมีตำแหน่งระบุซึ่งสามารถทำการสื่อสารในรูปแบบกระจายข่าวสารหรือการสื่อสารข่าวสารเดียวกันได้

5. เนื่องจากเป็นโครงข่ายที่มีขอบเขตปิดและเป็นอิสระดังนั้นจะไม่ได้รับข้อจำกัดทางเทคนิคหรือทางระบบเหมือนอย่างเช่นโครงข่ายสาธารณะ ดังนั้นจึงสามารถเสนอและออกแบบวิธีการแต่ละชนิดสำหรับการใช้งานได้อย่างอิสระ

ลักษณะการเชื่อมต่อพื้นฐานของระบบโครงข่ายท้องถิ่น แบ่งได้เป็น 3 ประเภท คือ

1. *โครงข่ายรูปดาว (Star Topology)* จะมีหน่วยประมวลผลหรือส่วนการควบคุมอยู่ตรงกลางเพื่อทำหน้าที่ตอบสนองการร้องขอของตัวลูกข่าย ส่วนตัวลูกข่ายก็จะต่อกระจายออกไปจากศูนย์ควบคุมกลางนี้ ซึ่งโครงข่ายรูปดาวนี้เป็นการทำงานแบบจุดต่อจุด คือ 1 พอร์ตต่อกับ 1 ตัวลูกข่ายด้วยช่องสัญญาณ 1 ช่อง โดยลักษณะพิเศษของโครงข่ายรูปดาวได้แก่

- เนื่องจากเป็นการควบคุมจากส่วนกลางด้วยอุปกรณ์ควบคุม ถ้าหากอุปกรณ์ควบคุมนั้นเกิดไม่ทำงาน โครงข่ายโดยรวมจะไม่สามารถสื่อสารกันได้ ดังนั้นเพื่อที่จะให้เกิดความน่าเชื่อถือมากขึ้นจำเป็นต้องมีการติดตั้งอุปกรณ์ควบคุมซ้อนขึ้นมา

- อุปกรณ์ควบคุมส่วนกลางกับสถานีจะเชื่อมต่อกันลักษณะ 1 ต่อ 1 ดังนั้นเมื่อเปรียบเทียบกับโครงข่ายอื่นๆแล้ว จะใช้เส้นทางส่งที่ยาวกว่า

2. *โครงข่ายแบบต้นไม้ (Bus/Tree Topology)* จะมีส่วนการควบและมีช่องสัญญาณที่ต่อกับส่วนการควบคุมนั้นเพียงเส้นเดียว โดยตัวลูกข่ายจะเชื่อมต่อเข้ากับช่องสัญญาณอันนั้น ดังนั้นจะมีตัวลูกข่ายเพียงตัวเดียวที่สามารถทำการเข้าถึงคอมพิวเตอร์แม่ได้เวลาหนึ่งๆ ส่วนตัวลูกข่ายอื่น ๆ จะไม่สามารถเข้าถึงคอมพิวเตอร์แม่ได้เลย จนกว่าช่วงเวลาดังกล่าวจะผ่านไป โดยลักษณะพิเศษของโครงข่ายแบบต้นไม้ได้แก่

- ฟังก์ชันควบคุมการสื่อสารนั้นจะกระจายไปอยู่แต่ละโหนด ทำให้ค่าใช้จ่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ของโครงข่ายจะเป็นสัดส่วนตามจำนวนโหนด ดังนั้นสำหรับระบบโครงข่ายท้องถิ่นที่มีขนาดเล็กและกลางนั้นจะค่อนข้างประหยัดค่าใช้จ่าย

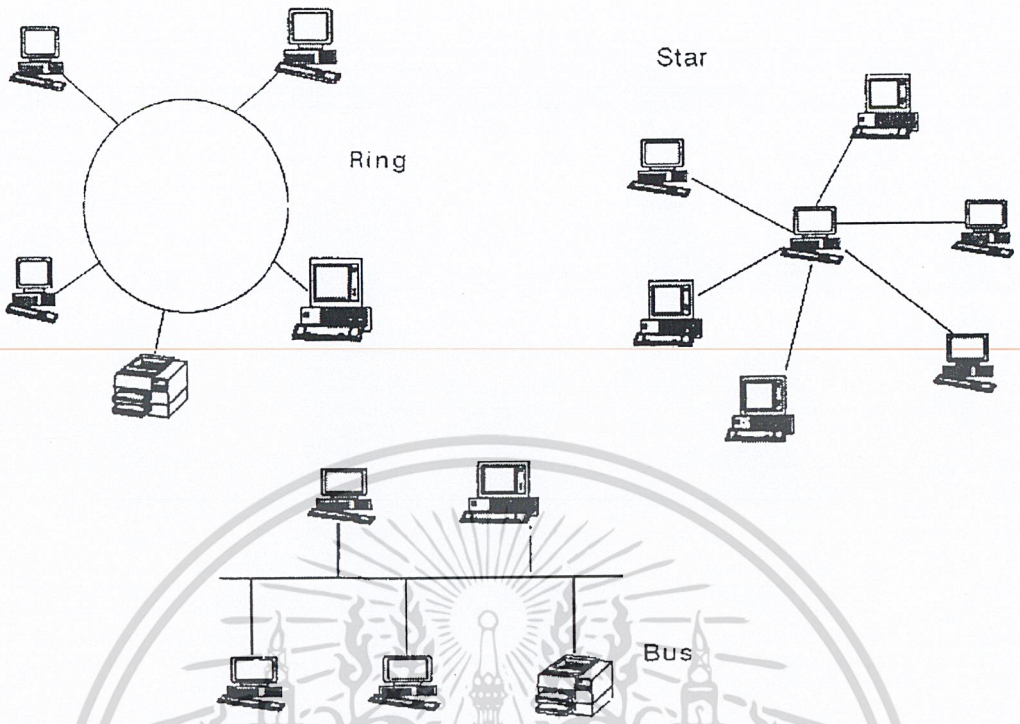
- เนื่องจากเป็นโครงข่ายในรูปแบบแพสซีพ (passive) ดังนั้นเมื่อมีโหนดใดๆ เสียหายก็ไม่มีอิทธิพลต่อการสื่อสารกับโหนดอื่นๆ
- สำหรับการเพิ่มเข้า ถอดออก หรือเคลื่อนย้ายโหนดนั้นจะไม่มีอิทธิพลต่อโหนดอื่น ดังนั้นการขยายโครงข่ายจะทำได้โดยมีประสิทธิภาพ

3. *โครงข่ายแบบวงแหวน (Ring Topology)* จะมีช่องสัญญาณเพียงช่องเดียว แต่ละตัวลูกข่ายต่อเข้ากับโครงข่ายซึ่งเป็นแบบวงปิด โดยทิศทางการไหลของข้อมูลจะเกิดขึ้นรอบๆวงแหวน ตัวลูกข่ายที่ต้องการส่งข้อมูลจะต้องรอจนกว่าโครงข่ายจะอยู่ในสภาวะว่างก่อนแล้วถึงจะทำการส่งข้อมูลได้ ซึ่งเครื่องรับก็จะตรวจสอบ Header และในขณะเดียวกันข้อมูลนั้นก็จะวนผ่านไปยังต้นทางเหมือนเดิมเพื่อรับข้อมูลต่อไป โดยลักษณะพิเศษของโครงข่ายแบบวงแหวนได้แก่

- ฟังก์ชันควบคุมการสื่อสารจะถูกกระจายไปแต่ละโหนดซึ่งทำให้ค่าใช้จ่ายในโครงข่ายจะเป็นสัดส่วนตามจำนวนโหนด ดังนั้นสำหรับระบบโครงข่ายท้องถิ่นที่มีขนาดเล็กและกลางนั้นจะค่อนข้างประหยัดค่าใช้จ่าย
- สิ่งที่แตกต่างจากโครงข่ายแบบต้นไม้นั้นก็คือ สัญญาณจะถูกสร้างขึ้นใหม่และทวนสัญญาณต่อ ดังนั้นสามารถใช้สำหรับสร้างระบบที่มีขนาดใหญ่กว่าโครงข่ายแบบตันไม้ แต่ในทางตรงข้ามจำเป็นที่จะต้องเพิ่มฟังก์ชันควบคุมการสื่อสาร ในกรณีที่สัญญาณที่ถูกสร้างจากโหนดใดๆได้รับความเสียหาย
- เนื่องจากเป็นโครงข่ายในลักษณะแอคทีพ (active) ดังนั้นถ้าโหนดใดๆเสียหายจะทำให้การสื่อสารทั้งระบบล้มเหลว เมื่อเป็นเช่นนี้ฟังก์ชั่นต่อผ่านหรือโครงข่ายซ้อนจะถูกนำมาใช้ในลักษณะของการสร้างเส้นทางกลับในกรณีที่มีความเสียหายที่ตำแหน่งใดๆ
- นอกจากนี้สำหรับโครงข่ายสื่อสารแบบวงแหวนที่มีโหนดใดโหนดหนึ่งควบคุมการสื่อสารของระบบโครงข่ายท้องถิ่นทั้งระบบจะถูกเรียกว่าลูป ทำนองเดียวกันโครงข่ายแบบลูปนั้นถ้าหากโหนดที่ปฏิบัติการควบคุมการสื่อสารเสียหายจะทำให้ระบบทั้งหมดไม่ทำงาน ดังนั้นจำเป็นที่จะต้องมีการสร้างโหนดควบคุมซ้อน

โดยลักษณะการเชื่อมต่อของระบบโครงข่ายท้องถิ่นทั้ง 3 ประเภท สามารถแสดงได้ดังรูปที่ 2.10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.10 ลักษณะการเชื่อมต่อพื้นฐานของระบบโครงข่ายท้องถิ่น

2.4 เครื่องส่งและเครื่องรับสัญญาณวิทยุแบบไร้สาย

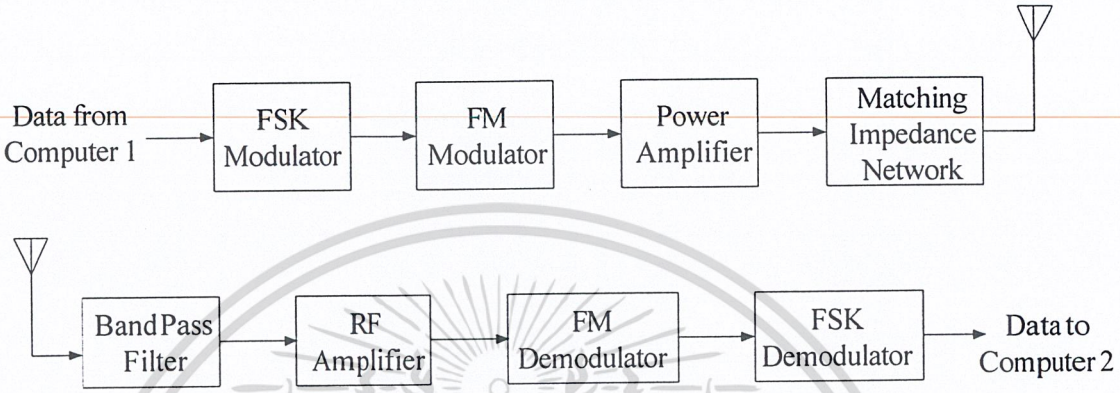
เครื่องส่งสัญญาณวิทยุที่ใช้ในโครงงานนี้เป็นแบบฟรีเควนซีมอดูเลชั่น ประกอบด้วยส่วนต่างๆ ดังนี้ วงจรฟรีเควนซีชิพท์คีย์อิ้งมอดูเลเตอร์ (Frequency Shift Keying Modulator) ซึ่งจะรับข้อมูลข่าวสารจากคอมพิวเตอร์ที่ถูกส่งออกมาทางพอร์ตอนุกรม และจะแปลงสัญญาณไบนารีนั้นให้เป็นสัญญาณอนาลอกที่มี 2 ความถี่ จากนั้นจะถูกส่งต่อไปยังวงจรฟรีเควนซีมอดูเลเตอร์ (Frequency Modulator) โดยสัญญาณที่ได้จะมีความถี่เปลี่ยนแปลงตามแอมปลิจูดของสัญญาณฟรีเควนซีชิพท์คีย์อิ้งที่เข้ามา ในที่นี้จะใช้วงจรสังเคราะห์ความถี่ในการกำเนิดคลื่นพาห์ แล้วส่งผ่านวงจรขยายกำลังเพื่อเพิ่มกำลังของสัญญาณข้อมูล เข้าสู่วงจรแมตช์ชิ่งอิมพีแดนซ์ทำให้สามารถถ่ายโอนกำลังงานของสัญญาณไปยังสายอากาศได้มากที่สุดและส่งออกสายอากาศ

ส่วนทางด้านเครื่องรับเมื่อสายอากาศรับสัญญาณที่ส่งมาจากเครื่องส่งแล้ว จะเข้าสู่วงจรกรองความถี่ผ่านเพื่อกรองเฉพาะสัญญาณช่วงความถี่ที่ต้องการให้สามารถผ่านออกมาได้ และขยายให้มีกำลังสูงขึ้นด้วยวงจรขยายสัญญาณ จากนั้นจะผ่านวงจรฟรีเควนซีดีมอดูเลเตอร์ (Frequency Demodulator) โดยจะลดความถี่ของสัญญาณลงมาอยู่ในย่านความถี่ที่เหมาะสม ซึ่งก็คือความถี่กลางหรือความถี่ไอเอฟ (Intermediate Frequency) และดีมอดูเลตสัญญาณนั้นเพื่อกำจัดความถี่กลางออกไป จะได้สัญญาณฟรีเควนซีชิพท์คีย์อิ้งออกมา จึงส่งต่อไปยังวงจรฟรีเควนซีชิพท์คีย์อิ้งดีมอดูเลเตอร์ (Frequency Shift Keying Demodulator) เพื่อนำสัญญาณข้อมูลไบนารีกลับมา และส่งเข้าไปยังพอร์ตอนุกรมของคอมพิวเตอร์อีกเครื่องหนึ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คอมพิวเตอร์แต่ละเครื่องจะมีโปรแกรมควบคุมการทำงานของระบบโครงข่าย ซึ่งทำหน้าที่ส่งและรับสัญญาณข้อมูลที่ต้องการ เครื่องส่งและเครื่องรับสัญญาณวิทยุเป็นเพียงอุปกรณ์ที่ทำหน้าที่แปลงสัญญาณจากคอมพิวเตอร์ให้สามารถส่งผ่านออกไปยังอากาศ และรับสัญญาณนั้นกลับคืนมาได้

บล็อกไดอะแกรมของเครื่องส่งและเครื่องรับสัญญาณวิทยุแบบไร้สายเป็นดังนี้



รูปที่ 2.11 บล็อกไดอะแกรมของเครื่องส่งและเครื่องรับสัญญาณวิทยุ

2.5 การมอดูเลตแบบฟรีเควนซีชิพท์คีย์อิ้ง (Frequency Shift Keying)

การมอดูเลตสัญญาณดิจิตอล (Digital Modulation) เข้ากับคลื่นพาห์ที่เป็นสัญญาณรูปไซน์ (Sinusoidal Wave) นั้น ทำเพื่อต้องการให้สัญญาณดิจิตอลเหล่านั้นสามารถส่งผ่านตัวกลางที่ออกแบบมาสำหรับสัญญาณอนาลอกได้ หรือทำการแปลงจากสัญญาณดิจิตอลให้เป็นสัญญาณอนาลอกนั่นเอง ซึ่งวิธีที่นิยมใช้มี 4 แบบ คือ

1. แอมปลิจูดชิพท์คีย์อิ้ง (Amplitude Shift Keying หรือ ASK)
2. ฟรีเควนซีชิพท์คีย์อิ้ง (Frequency Shift Keying หรือ FSK)
3. เฟสชิพท์คีย์อิ้ง (Phase Shift Keying หรือ PSK)
4. ควอดราเจอร์แอมปลิจูดมอดูเลชั่น (Quadrature Amplitude Modulation หรือ QAM)

การมอดูเลตแบบฟรีเควนซีชิพท์คีย์อิ้งหรือเอฟเอสเค เป็นการมอดูเลตที่รูปคลื่นของสัญญาณที่มีการเปลี่ยนแปลงความถี่ซึ่งถูกควบคุมโดยระดับแรงดันของสัญญาณข่าวสาร โดยเมื่อมีการเปลี่ยนแปลงระดับแรงดันของสัญญาณข่าวสารก็จะมีการเปลี่ยนแปลงความถี่ของสัญญาณคลื่นพาห์ ส่วนแอมปลิจูดและเฟสจะคงที่ตลอด การสร้างสัญญาณเอฟเอสเคสามารถทำได้ 2 แบบ คือ

1. *สร้างจากสวิตช์แบบดิจิตอล* สัญญาณดิจิตอลจะเลือกออสซิลเลเตอร์ที่มีความถี่ที่ต้องการเพื่อส่งสัญญาณออกไป โดยจำนวนออสซิลเลเตอร์ที่ใช้จะต้องมีจำนวนเท่ากับจำนวนระดับแรงดันของสัญญาณข่าวสาร จึงทำให้เฟสของสัญญาณแต่ละช่วงไม่มีความต่อเนื่องกัน

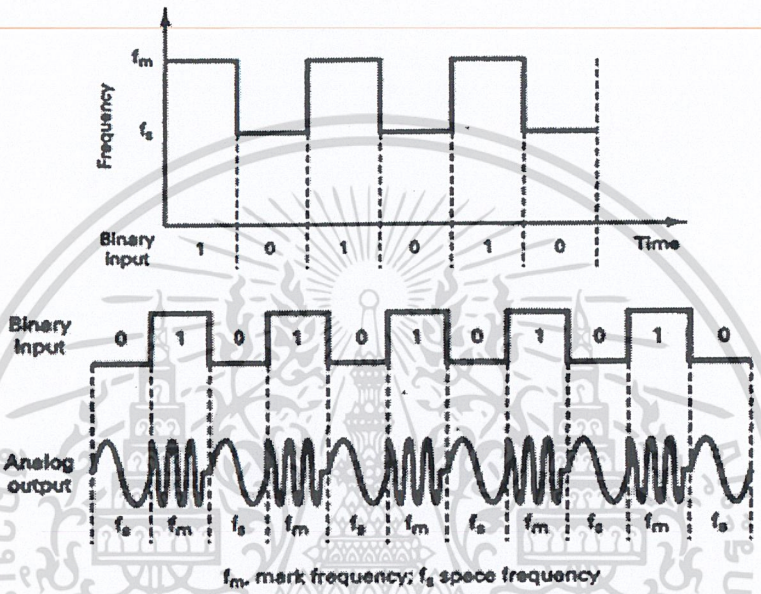
2. *สร้างโดยออสซิเลเตอร์ตัวเดียวกัน* โดยที่ออสซิลเลเตอร์นั้นสามารถเปลี่ยนแปลงความถี่ได้ สัญญาณที่ได้จากวิธีนี้จะมีความต่อเนื่องของสัญญาณ จึงเรียกว่า เอฟเอสเคเฟสต่อเนื่อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อพิจารณาถึงลักษณะของสัญญาณเอฟเอสเค จะเห็นว่าเป็นการเปลี่ยนสัญญาณดิจิตอลให้เป็นข้อมูลอนาลอกโดยเมื่อข้อมูล “0” เข้ามา จะถูกเปลี่ยนเป็นสัญญาณไซน์ที่มีความถี่ ω_1 และเมื่อข้อมูล “1” เข้ามา ก็จะถูกเปลี่ยนเป็นสัญญาณไซน์ที่มีความถี่ ω_2 ซึ่งสามารถเขียนเป็นสมการได้ดังนี้

$$s_1(t) = A\cos(\omega_1 t) \qquad \text{สำหรับสัญญาณไบนารี 0} \qquad (2.1)$$

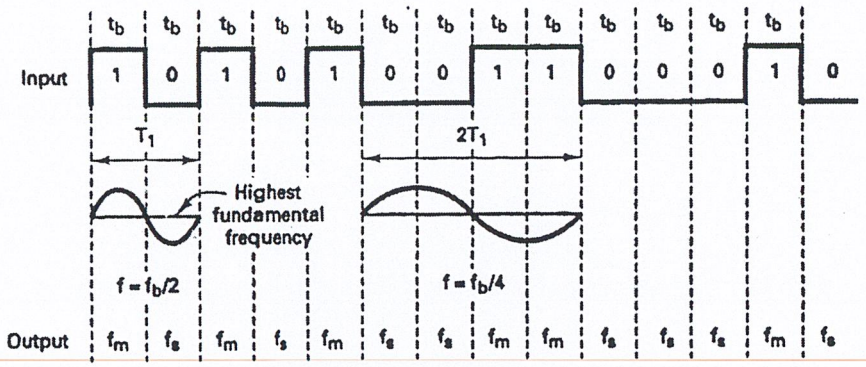
$$s_2(t) = A\cos(\omega_2 t) \qquad \text{สำหรับสัญญาณไบนารี 1} \qquad (2.2)$$



รูปที่ 2.12 ลักษณะของสัญญาณเอฟเอสเค

หลักการของเอฟเอสเค คือ เมื่อสัญญาณดิจิตอลที่มีลักษณะเป็นข้อมูลไบนารีถูกส่งเข้ามา จะทำให้ความถี่เลื่อนหรือเบี่ยงเบนไปตามการเปลี่ยนแปลงของข้อมูลไบนารีนั้น ดังนั้นสัญญาณทางเอาท์พุตของเอฟเอสเคมอดูเลเตอร์จะอยู่ในรูปของความถี่ที่มีการเปลี่ยนแปลงอย่างต่อเนื่อง เมื่อข้อมูลไบนารีด้านอินพุตเปลี่ยนแปลงจากสถานะลอจิก “1” เป็นลอจิก “0” (หรือในทางกลับกันคือ ลอจิก “0” เป็นลอจิก “1”) สัญญาณเอาท์พุตก็จะเลื่อนความถี่ระหว่าง 2 ความถี่ คือ ความถี่ที่ลอจิก “1” หรือ Mark frequency (f_m) และความถี่ที่ลอจิก “0” หรือ Space frequency (f_s) โดยการเปลี่ยนแปลงหรือการเลื่อนของความถี่แต่ละครั้งจะเกิดขึ้นเมื่อสถานะของลอจิกด้านสัญญาณเข้าเปลี่ยนแปลง นั่นคืออัตราการเปลี่ยนแปลงสัญญาณออกจะเท่ากับสัญญาณเข้า ซึ่งในดิจิตอลมอดูเลชั่นนั้นอัตราการเปลี่ยนแปลงของสัญญาณด้านอินพุตของเอฟเอสเคมอดูเลเตอร์จะเรียกว่า อัตราบิต (Bit Rate) มีหน่วยเป็นบิตต่อวินาที (bps) ส่วนอัตราการเปลี่ยนแปลงของสัญญาณด้านเอาท์พุตของของเอฟเอสเคมอดูเลเตอร์เรียกว่า อัตราบอด (Baud Rate) ดังนั้นในการส่งข้อมูลของเอฟเอสเคมอดูเลเตอร์ อัตราบิตจะเท่ากับอัตราบอดเสมอ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



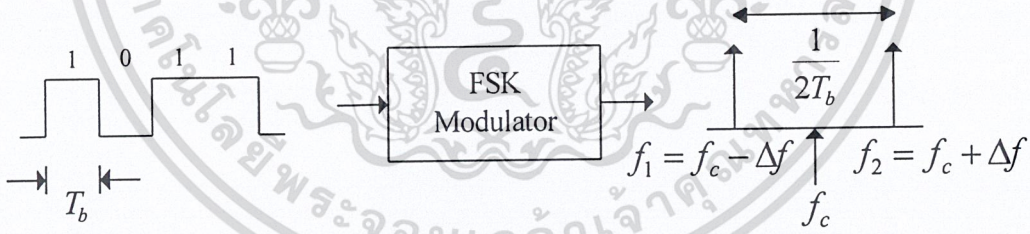
รูปที่ 2.13 อัตราการเปลี่ยนแปลงของสัญญาณอินพุต

จากรูปที่ 2.13 จะเห็นว่าอัตราการเปลี่ยนแปลงที่เร็วที่สุดของสัญญาณอินพุตจะเกิดขึ้นเมื่อไบนารีมีลักษณะเป็น 1 และ 0 สลับกันซึ่งก็คือ สัญญาณสี่เหลี่ยมนั่นเอง ซึ่งความถี่หลัก (Fundamental Frequency) ของคลื่นสี่เหลี่ยมจะเท่ากับครึ่งหนึ่งของอัตราการส่งข้อมูล ดังนั้นถ้าพิจารณาเฉพาะความถี่หลักเพียงอย่างเดียวแล้ว ความถี่สูงสุดของสัญญาณดิจิตอลที่ต้องการนำมามอดูเลตแบบเอฟเอสเคจะเท่ากับครึ่งหนึ่งของอัตราการส่งข้อมูล คือ

$$f_{a\max} = Bit\ Rate/2 \tag{2.3}$$

เมื่อ $f_{a\max}$ คือ ความถี่สูงสุดของสัญญาณดิจิตอลที่จะนำมามอดูเลต

ความถี่กลาง (Center Frequency หรือ f_0) ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ (Voltage Controlled Oscillator) จะอยู่ในตำแหน่งกลางระหว่าง f_m กับ f_s ดังแสดงในรูปที่ 2.14



รูปที่ 2.14 การเบี่ยงเบนความถี่ของสัญญาณเอฟเอสเค

เมื่อลอจิก "1" จากด้านอินพุตเข้ามา จะเลื่อนความถี่ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์จาก f_0 ไปเป็น f_s ซึ่งจะเห็นว่าการเปลี่ยนแปลงของข้อมูลไบนารีด้านอินพุตจาก "1" ไป "0" หรือ "0" ไป "1" จะทำให้ความถี่เอาท์พุตของเอฟเอสเคมอดูเลเตอร์เลื่อนหรือเบี่ยงเบนกลับไปมาระหว่าง f_m หรือ f_s เนื่องจากเอฟเอสเคนั้นก็คือรูปแบบหนึ่งของการมอดูเลตทางความถี่ ดังนั้นดัชนีการมอดูเลต (Modulate Index หรือ MI) ในเอฟเอสเคก็สามารถหาได้เช่นเดียวกับการมอดูเลตทางความถี่

$$MI = \Delta f / f_a \tag{2.4}$$

เมื่อ MI คือ ดัชนีการมอดูเลต

Δf คือ การเบี่ยงเบนของความถี่ใดๆ จากความถี่กลาง (Hz)

f_a คือ ความถี่ของสัญญาณที่นำมามอดูเลต (Hz)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ค่า MI สูงสุด คือ ค่า MI ที่ทำให้แบนวิดธ์กว้างที่สุด ซึ่งจะเกิดขึ้นจากการเบี่ยงเบนของความถี่ถูกมอดูเลตแล้ว และความถี่ของสัญญาณที่นำมามอดูเลตมีค่าสูงสุด

ในเอฟเอสเค ค่า Δf เป็นการเบี่ยงเบนของความถี่สูงสุด (Peak frequency deviation) ของสัญญาณที่ถูกมอดูเลตซึ่งมีค่าเท่ากับความแตกต่างระหว่าง f_m กับ f_0 หรือ f_0 กับ f_s ซึ่งก็คือ ครึ่งหนึ่งของความแตกต่างระหว่าง f_m กับ f_s นั่นคือ

$$\Delta f = (f_m - f_s)/2 \tag{2.5}$$

การเบี่ยงเบนของความถี่สูงสุดจะขึ้นอยู่กับขนาดหรือแอมปลิจูดของสัญญาณดิจิตอลที่นำมามอดูเลต เมื่อสถานะทางลอจิกเป็น "1" จะให้แรงดันออกมาค่าหนึ่งคงที่ตาม (เช่น 5 โวลต์) หรือ ถ้าในลอจิก "0" แรงดันออกมาคงที่ในระดับหนึ่งเช่นกัน (เช่น 0 โวลต์)

f_a เป็นความถี่ของข้อมูลไบนารีด้านอินพุต ซึ่งจะทำให้แบนวิดธ์กว้างที่สุดเมื่อ f_a = Bit Rate/2 เท่านั้น เพราะฉะนั้นเราสามารถหาค่า MI ได้จาก

$$MI = (f_s - f_m)/f_b \tag{2.6}$$

เมื่อ $f_s - f_m$ คือ ความถี่เบี่ยงเบนสูงสุด

f_b คือ อัตราบิตของไบนารีอินพุต

โดยทั่วไปความกว้างของแบนวิดท์จะแปรผันตรงกับค่า MI ซึ่งเช่นเดียวกับการมอดูเลตทางความถี่ที่ค่า MI โดยทั่วๆ ไปจะต้องมีค่าต่ำกว่า 1.0 เพื่อให้มีค่าแบนวิดท์ที่แคบที่สุดเรียกว่า Minimum Nyquist Bandwidth (f_n)

ส่วนทางด้านฟรีเควนซีชิพท์คีย์อิ้งดีมอดูเลชั่น จะเป็นตัวแยกสัญญาณไบนารีออกจากสัญญาณฟรีเควนซีชิพท์คีย์อิ้ง ซึ่งสามารถกระทำได้หลายแบบแต่ในที่นี้จะกล่าวถึง 3 แบบ คือ

1. *ใช้วงจร Match Filter* วงจรจะประกอบไปด้วยส่วนของวงจรกรองแถบความถี่ผ่าน (Bandpass filter) ซึ่งจะทำหน้าที่กรองความถี่ f_m และ f_s ซึ่งเราจะแทนด้วยลอจิก 1 และลอจิก 0 ตามลำดับ ต่อมานำเอาท์พุตของวงจรกรองแถบความถี่ผ่านมาเข้าวงจรตรวจจับกรอบสัญญาณ (Envelope Detector) ซึ่งจะทำการจับกรอบของสัญญาณที่ออกมาจากวงจรกรองแถบความถี่ผ่าน และนำมาเข้าวงจรเปรียบเทียบ (Comparator) ซึ่งจะได้สัญญาณไบนารีออกมา

2. *ใช้วงจรเฟสล็อกลูป (Phase Lock Loop)* มีความถี่ฟรีรันนิ่งเท่ากับ f_0 โดยในขณะที่ความถี่อินพุตของวงจรเฟสล็อคลูปเลื่อนไปมาระหว่าง f_m กับ f_s จะทำให้เกิดแรงดันคลาดเคลื่อนไฟตรง (DC Error Voltage) ซึ่งเป็นผลมาจากการเปรียบเทียบทางเฟส (Phase Comparator) ของสัญญาณอินพุต เนื่องจากความถี่อินพุตที่เข้ามายังวงจรเฟสล็อคลูปมีเพียง 2 ความถี่ คือ f_m และ f_s ดังนั้นค่าแรงดันดังกล่าวจึงมีเพียง 2 ระดับเท่านั้น ซึ่งสามารถแทนค่าลอจิก "1" และลอจิก "0" เมื่อความถี่ทางอินพุตเป็น f_m และ f_s ตามลำดับ เราจึงได้สัญญาณเอาท์พุตจากวงจรเฟสล็อคลูป กลับมาเป็นข้อมูลไบนารีเหมือนกับตอนแรกที่ส่งมาทุกประการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

3. *ใช้วงจรดิฟเฟอเรนชิเอเตอร์ (Differentiator)* คู่กับวงจรตรวจจับกรอบสัญญาณโดย วงจรดิฟเฟอเรนชิเอเตอร์จะทำการเปลี่ยนสัญญาณเอฟเอสเคให้คล้ายกับสัญญาณเอเอ็ม จากนั้นก็ใช้วงจรตรวจจับกรอบสัญญาณจับเอากรอบสัญญาณออกมา

2.6 การมอดูเลตทางความถี่ (Frequency Modulation)

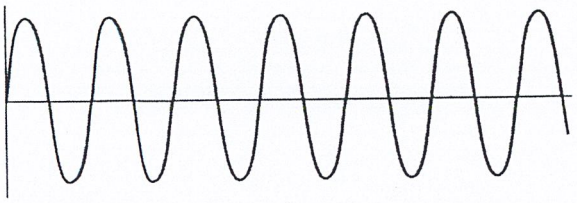
การมอดูเลตเป็นการผสมสัญญาณของข้อมูลเข้าไปกับสัญญาณอีกสัญญาณหนึ่งที่เรียกว่า คลื่นพาห์ (Carrier) ซึ่งสัญญาณนี้มีความถี่ที่เหมาะสมกับช่องสัญญาณนั้นๆ เพื่อให้ข้อมูลที่ส่งเข้าไปในช่องสัญญาณเดินทางได้ไกลมากขึ้น โดยมีกระบวนการดังนี้ เริ่มจากคลื่นพาห์ผสมกับสัญญาณข้อมูลที่ตัวมอดูเลต (Modulator) แล้วส่งไปที่เครื่องส่ง จากเครื่องส่งไปยังเครื่องรับจะเป็นช่องสัญญาณสำหรับลำเลียงสัญญาณผสมนี้ สัญญาณผสมจากเครื่องรับจะไปเข้าตัวดีมอดูเลต (Demodulator) เพื่อแยกสัญญาณข้อมูลออกมา การเลือกวิธีมอดูเลตขึ้นอยู่กับปัจจัยหลายประการ เช่น ชนิดของช่องสัญญาณ แบนด์วิดธ์ ประสิทธิภาพของระบบที่ต้องการ และความต้านทานต่อสัญญาณรบกวน เป็นต้น เมื่อมีการมอดูเลตที่ภาคส่งการที่จะรับข้อมูลได้นั้นต้องมีการดีมอดูเลตที่ภาครับด้วย

การมอดูเลตทางความถี่ (Frequency Modulation หรือ FM) เป็นการเปลี่ยนแปลงความถี่ของคลื่นพาห์ตามสัญญาณของข้อมูลที่เข้ามา ดังแสดงในรูปที่ 2.15 จะเห็นได้ว่าขนาดของคลื่นพาห์จะคงที่เสมอไม่ว่าสัญญาณข่าวสารจะมีการเปลี่ยนแปลงอย่างไร มีแต่ส่วนของความถี่เท่านั้นที่มีการเปลี่ยนแปลงตามขนาดความแรงของสัญญาณ การเปลี่ยนแปลงความถี่ตามขนาดความแรงของสัญญาณเรียกว่า การเบี่ยงเบนความถี่ (Frequency Deviation) โดยใน 1 รอบของสัญญาณข่าวสารนั้นสัญญาณเอฟเอ็มก็จะเปลี่ยนตามไปครบ 1 รอบด้วยเช่นกัน ดังนั้นถ้าสัญญาณข่าวสารมีความถี่มากขึ้นอัตราการเบี่ยงเบนของสัญญาณเอฟเอ็มก็จะมากขึ้นด้วยนั้นเอง

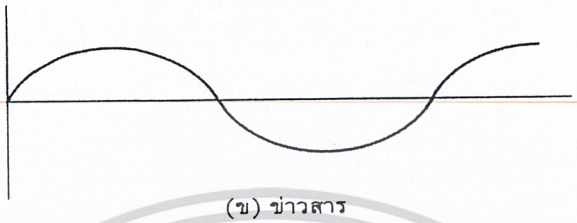
การมอดูเลตแบบเอฟเอ็มให้คุณภาพของสัญญาณดีกว่าการมอดูเลตแบบเอเอ็ม แต่ระบบจะซับซ้อนกว่า อย่างไรก็ตามจากความก้าวหน้าทางเทคโนโลยีในปัจจุบัน ทำให้การมอดูเลตแบบเอฟเอ็มสามารถทำได้ง่ายขึ้นโดยใช้เทคนิคเฟสล็อกลูป (Phase Lock Loop) ดังแสดงในรูปที่ 2.16 สัญญาณในการมอดูเลตแบบเอฟเอ็มจะได้จากสัญญาณขาออกของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ ซึ่งได้รับสัญญาณความผิดพลาดจากวงจรกรองผ่านความถี่ต่ำ (Low Pass Filter) เพื่อพยายามที่จะปรับความถี่และเฟสด้านขาออกกับค่าสัญญาณขาเข้าโดยการคูณกันเพื่อให้ได้ผลต่างระหว่างสัญญาณทั้งสองคงที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

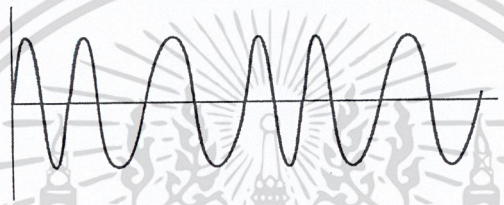
46489



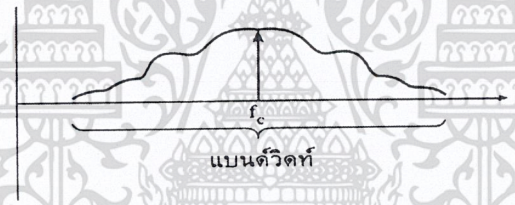
(ก) คลื่นพาห์



(ข) ข่าวสาร



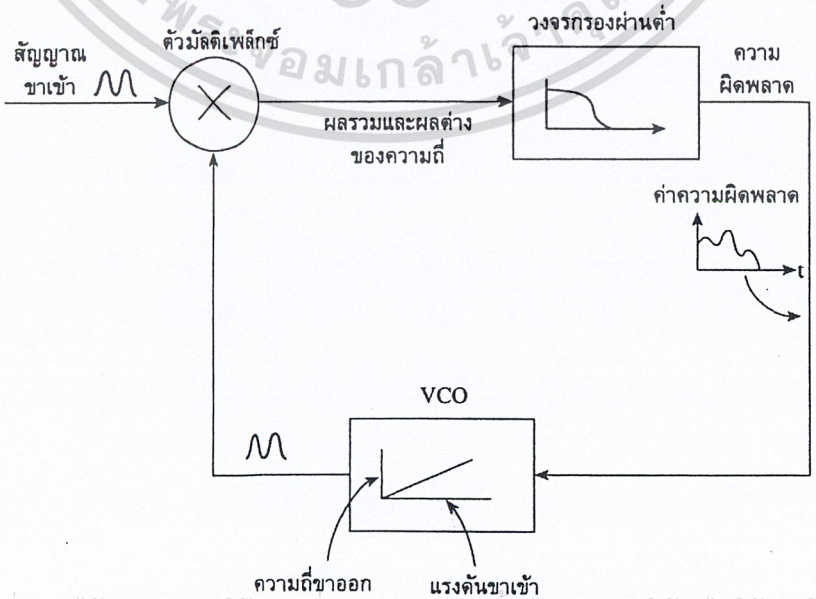
(ค) สัญญาณที่ถูกมอดูเลตแบบ FM



(ง) แบนด์วิดท์ของสัญญาณที่ถูกมอดูเลตแบบ FM

โดยที่ f_c เป็นความถี่ของคลื่นพาห์ (carrier frequency)

รูปที่ 2.15 ลักษณะของสัญญาณเอฟเอ็ม



รูปที่ 2.16 การสร้างสัญญาณเอฟเอ็มโดยใช้เทคนิคเฟสล็อกลูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อพิจารณาถึงสมการของสัญญาณ โดยกำหนดให้คลื่นพาห์ที่ยังไม่ได้มีการมอดูเลต คือ

$$\phi(t) = A_c \cos(\omega_c t + \theta) \tag{2.7}$$

ซึ่งการทำให้ค่า θ แปรผันตามค่าความแรงของสัญญาณข่าวสาร $m(t)$ จะทำให้ได้สัญญาณมอดูเลตเชิงมุม (Angle Modulated Signal) ออกมา ซึ่งค่า θ จะเรียกว่า เฟสของคลื่นรูปไซน์ $\phi(t)$ โดยอ้างอิงเทียบสัญญาณ $\cos(\omega_c t)$

$$\theta = k_p m(t) \tag{2.8}$$

อย่างไรก็ดีสามารถที่จะทำให้ค่า θ แปรผันตามค่าอินทิเกรตของสัญญาณข่าวสารได้

$$\theta = k_f \int_{-\infty}^{t} m(\tau) d\tau \tag{2.9}$$

จะเรียกสัญญาณที่เกิดขึ้นว่า สัญญาณมอดูเลตทางความถี่ (Frequency Modulated Signal) เนื่องจากค่าของเฟสชั่วขณะ (Instantaneous Phase) θ_i มีความสัมพันธ์กับค่าของความถี่ชั่วขณะ (Instantaneous Frequency) ω_i ดังนี้

$$\omega_i = \frac{d\theta_i(t)}{dt} \tag{2.10}$$

หรือ

$$\theta_i(t) = \int_{-\infty}^{t} \omega_i(\tau) d\tau \tag{2.11}$$

โดยจะพบว่าค่าความถี่ของคลื่นภายหลังจากการมอดูเลตความถี่แล้ว จะมีการเปลี่ยนแปลงความถี่ไปจาก ω_c เดิม ซึ่งอัตราการเบี่ยงเบนนั้นขึ้นกับค่าความแรงของสัญญาณมอดูเลต โดยสัญญาณข่าวสารที่เข้ามามอดูเลต $m(t)$ จะไปเบี่ยงเบนความถี่ของคลื่นพาห์ (ω_c) ทำให้ได้ความถี่คลื่นพาห์ที่เปลี่ยนไป และสามารถเขียนสมการรูปทั่วไปของสัญญาณเอฟเอ็มได้ คือ

$$\phi_{FM} = A_c \cos(\omega_c t + k_f \int_{-\infty}^{t} m(t) d\tau) \tag{2.12}$$

ถ้าสัญญาณข่าวสาร $m(t)$ ที่เข้ามาอยู่ในรูปของสัญญาณไซน์

$$m(t) = A_m \cos(\omega_m t) \tag{2.13}$$

จะสามารถเขียนสมการของเอฟเอ็มใหม่ได้เป็น

$$\phi_{FM} = A_c \cos[\omega_c t + k_f \int_{-\infty}^{t} A_m \cos(\omega_m t) d\tau] \tag{2.14}$$

$$\phi_{FM} = A_c \cos[\omega_c t + \frac{k_f A_m}{\omega_m} \sin(\omega_m t)] \tag{2.15}$$

ซึ่งค่า $\frac{k_f A_m}{\omega_m}$ ก็คือดัชนีการมอดูเลต (Modulation Index หรือ β) นั่นเอง

จากสมการที่ (2.15) สามารถกระจาย ได้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\phi_{FM} = A_c\{\cos(\beta\sin\omega_m t)\cos\omega_c t - \sin(\beta\sin\omega_m t)\sin\omega_c t\} \quad (2.16)$$

เมื่อทำการกระจายให้อยู่ในรูปอนุกรมฟูเรียร์ (Fourier Series) ดังนี้

$$\cos(\beta\sin\omega_m t) = J_0(\beta) + \sum_{\text{n : even}}^{\infty} 2J_n(\beta)\cos(n\omega_m t) \quad (2.17)$$

$$\sin(\beta\sin\omega_m t) = \sum_{\text{n : odd}}^{\infty} 2J_n(\beta)\sin(n\omega_m t) \quad (2.18)$$

โดยที่ n เป็นเลขจำนวนเต็มที่มีค่าบวก และ $J_n(\beta)$ คือ เบสเซลฟังก์ชันประเภทที่หนึ่งที่มีอันดับเป็น n ซึ่งเขียนได้ในรูปต่อไปนี้

$$J_n(\chi) = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{j(\beta\sin\alpha - n\alpha)}d\alpha \quad (2.19)$$

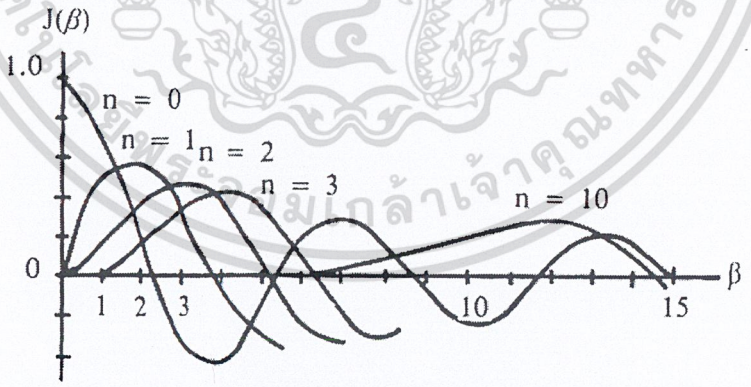
จะได้สัญญาณเอฟเอ็มที่กระจายอยู่ในรูปอนุกรมฟูเรียร์ดังต่อไปนี้

$$\phi_{FM} = A_c J_0(\beta)\cos\omega_c t + \sum_{n:odd}^{\infty} A_c J_n(\beta)\{\cos(\omega_c + n\omega_m)t - \cos(\omega_c - n\omega_m)t\}$$

$$+ \sum_{n:even}^{\infty} A_c J_n(\beta)\{\cos(\omega_c + n\omega_m)t - \cos(\omega_c - n\omega_m)t\} \quad (2.20)$$

และถ้าเราใช้คุณสมบัติของเบสเซลฟังก์ชันที่ $J_{-n}(\beta) = (-1)^n J_n(\beta)$ เราจะเขียนสมการได้ใหม่ดังนี้

$$\phi_{FM} = A_c\sum_{n=-\infty}^{\infty} J_n(\beta)\cos(\omega_c + n\omega_m)t \quad (2.21)$$



รูปที่ 2.17 การเปลี่ยนแปลงเบสเซลฟังก์ชันอันดับ n

เราพบว่าจะมีสเปกตรัมของสัญญาณอยู่ที่ความถี่ $f_r + nf_m$ โดยที่มีขนาดของสเปกตรัมเป็น $J_0(\beta)$ ดังนั้นเมื่อเราพิจารณาคุณสมบัติของ $J_0(\beta)$ ก็จะรู้ได้ว่าการกระจายของสเปกตรัมของสัญญาณเอฟเอ็มจะออกมาในรูปใด $J_0(\beta)$ นั้นเมื่อเขียนเป็นกราฟจะได้ตามที่แสดงไว้ในรูปที่ 2.17 ข้อสังเกตจากกราฟก็คือ บริเวณที่ $\beta \cong 0$ จะมี $J_0(\beta)$ ที่มีค่าสูงสุด และ $J_0(\beta)$ จะเล็กลงเมื่อ n มีค่าสูงขึ้น ข้อสังเกตอีกประการหนึ่งก็คือ เมื่อ β มีค่าใหญ่ขึ้นเอ็นเวลโลปของ $J_0(\beta)$ ที่แกว่งระหว่างค่าบวกกับลบจะมีขนาดเล็กลง โดยเฉพาะเมื่อ $n > \beta$ จะเล็กลงอย่างน่าสังเกต จากคุณสมบัติของเบสเซลฟังก์ชันดังกล่าวจะทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

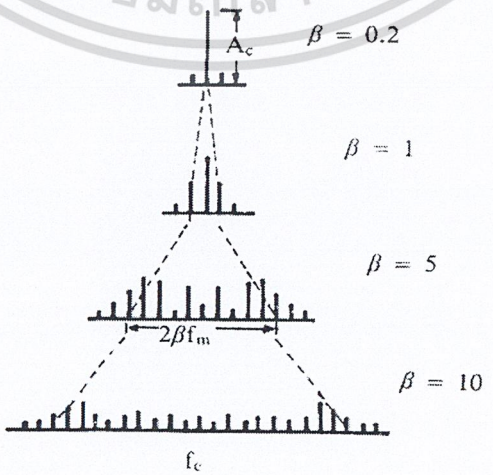
ให้ เราสามารถสรุปความสัมพันธ์ระหว่างการกระจายของสเปกตรัมกับ β ได้ดังนี้

1. $J_0(\beta)$ ซึ่งเป็นแอมปลิจูดของสเป็กตรัมของสัญญาณคลื่นพาห์จะมีค่าเปลี่ยนแปลงไปตาม β โดยที่จะมีค่าสูงที่สุด β = 0 และมีค่าเป็นศูนย์ที่ค่า β = 2.405, 5.520 เป็นต้น นั่นหมายความว่าสัญญาณคลื่นพาห์จะมีขนาดเปลี่ยนแปลงไปตามสัญญาณที่เข้ามามอดูเลต จึงมีข่าวสารของสัญญาณที่มามอดูเลตอยู่ด้วยบางส่วน ซึ่งจะแตกต่างกับกรณีของการมอดูเลตแบบเอเอ็มเพราะสัญญาณคลื่นพาห์ในระบบเอเอ็มจะไม่มีข่าวสารอยู่เลย

2. การกระจายของสเป็กตรัมจะขึ้นอยู่กับค่า β ถ้าค่า β มีขนาดเล็ก คือ $\beta << 1$ เราจะเห็นได้ว่า $J_0(\beta)$ กับ $J_1(\beta)$ เท่านั้นที่มีค่าสูงน่าสังเกต จึงทำให้มีสเป็กตรัมของความถี่ $f_r + f_m$ และ f_r เท่านั้นที่น่าสังเกต แต่ถ้า β ใหญ่ขึ้น โดยเฉพาะ $\beta >> 1$ การกระจายของสเป็กตรัมก็จะกว้างออกไป โดยที่ β สูงจะทำให้กระจายกว้างขึ้นซึ่งสอดคล้องกับการที่มีการเบี่ยงเบนความถี่สูง

ในกรณีที่ $\beta << 1$ ดังที่กล่าวข้างต้นนี้ เราจะเรียกสัญญาณเอฟเอ็มนั้นว่า สัญญาณเอฟเอ็มแบนด์วิดธ์แคบ (Narrow – band F.M. signal) ซึ่งสัญญาณนี้ถ้านำไปเขียนในรูปเฟสเซอร์เทียบเคียงกับสัญญาณเอเอ็มจะเป็นดังที่แสดงไว้ในรูปที่ 2.18 ซึ่งจะเห็นได้ว่ามีส่วนคล้ายคลึงกับสัญญาณเอเอ็มเพราะมีไซด์แบนด์ที่เท่ากับ $f_r + f_m$ ส่วนที่แตกต่างกันก็คือกรณีของสัญญาณเอฟเอ็มแบนด์วิดธ์แคบนั้น ผลรวมของไซด์แบนด์ทั้งคู่จะอยู่ในแนวตั้งฉากกับสัญญาณคลื่นพาห์ ซึ่งทำให้เฟสเซอร์ผลรวมมีเฟสเปลี่ยนแปลงตามเวลา ในขณะที่สัญญาณเอเอ็มนั้นผลรวมของไซด์แบนด์ทั้งคู่จะทำให้เฟสเซอร์ผลรวมมีขนาดเปลี่ยนแปลงตามเวลา

ดังนั้นสามารถสรุปได้ว่าสัญญาณเอฟเอ็มจะแบ่งออกได้เป็น 2 แบบ คือ เอฟเอ็มแบนด์แคบและเอฟเอ็มแบนด์กว้าง การที่เราจะดูว่าเป็นแบนด์แคบหรือแบนด์กว้างนั้นจะพิจารณาที่ค่าดัชนีการมอดูเลต (β) โดยที่ถ้าเป็นเอฟเอ็มแบนด์แคบในทางทฤษฎีนั้นค่า $\beta << \frac{1}{\sqrt{10}}$ แต่ในการใช้งานอาจมีค่าสูงถึง 0.5 ได้ และถ้า β มากกว่า 0.316 หรือ 0.5 จะเป็นเอฟเอ็มแบนด์กว้าง



รูปที่ 2.18 การกระจายสเปกตรัมของสัญญาณเอฟเอ็ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

β	พาหะ	ไซด์แบนด์คู่ที่							
		1	2	3	4	5	6	7	8
0.00	1.00	-	-	-	-	-	-	-	-
0.25	0.98	0.12	-	-	-	-	-	-	-
0.5	0.94	0.14	0.03	-	-	-	-	-	-
1.0	0.77	0.44	0.11	0.02	-	-	-	-	-
1.5	0.51	0.56	0.23	0.06	0.01	-	-	-	-
2.0	0.22	0.58	0.35	0.13	0.03	-	-	-	-
2.5	-0.05	0.50	0.45	0.22	0.07	0.02	-	-	-
3.0	-0.26	0.34	0.49	0.31	0.13	0.04	0.01	-	-
4.0	-0.40	-0.07	0.36	0.43	0.28	0.13	0.05	0.02	-
5.0	-0.18	-0.33	0.05	0.36	0.39	0.26	0.13	0.05	0.02
6.0	0.15	0.28	-0.24	0.11	0.36	0.36	0.25	0.13	0.06
7.0	0.30	0.00	-0.20	-0.17	0.16	0.35	0.34	0.23	0.13
8.0	0.17	0.23	-0.11	-0.29	-0.10	0.19	0.34	0.32	0.22
9.0	-0.09	0.24	0.14	-0.18	-0.27	-0.06	0.20	0.33	0.30
10.0	-0.25	0.04	0.25	0.06	-0.22	-0.23	-0.01	0.22	0.31

ตารางที่ 2.2 ตารางความสัมพันธ์ระหว่างการกระจายของสเปกตรัมกับมอดูเลชันอินเด็กซ์

ซึ่งจะเห็นว่าสเปกตรัมของ $\phi_{FM}(t)$ นั้นกระจายออกไปบนแกนความถี่ไม่มีสิ้นสุด ซึ่งหมายความว่าแบนด์วิดธ์ของสัญญาณเอฟเอ็มตามทฤษฎีนั้นมีค่าอนันต์ แต่อย่างไรก็ตามเมื่อพิจารณาให้ลึกลงไป จะพบว่าค่า $|J_n(\beta)|$ นี้จะลดลงหายไปอย่างรวดเร็ว เมื่อ $n > \beta$ โดยเฉพาะอย่างยิ่งเมื่อ β มีค่ามาก ทำให้เราสรุปได้ว่า ถ้าแถบข้างที่มีนัยสำคัญ คือ แถบข้างที่มีขนาดเกิน 1% ของขนาดของคลื่นพาห์ที่ยังไม่มีการมอดูเลตแล้ว จะมีไซด์แบนด์ตัวสุดท้าย คือ ตัวที่ $n \approx \beta$ เมื่อ β มีค่ามาก เพราะฉะนั้น ถ้าคิดโดยนัยสำคัญในทัศนะดังกล่าวแล้วจะพบว่าค่าแบนด์วิดธ์ของสัญญาณเอฟเอ็มตามนั้น จะมีค่าประมาณได้ดังต่อไปนี้ คือ

$$BW \approx 2n\omega_m \tag{2.22}$$

$$2\beta\omega_m = 2\Delta\omega \tag{2.23}$$

$$BW \approx 2\Delta\omega \qquad \text{สำหรับ } \beta \text{ ค่ามาก} \tag{2.24}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับเมื่อ $\frac{n}{\beta}$ มีค่าน้อยมาก ๆ นั้น ซึ่งเป็นกรณีของเอฟเอ็มแบนด์แคบ ค่าขนาดของ $J_n(\beta)$ที่มีนัยสำคัญจะมีเพียง $J_0(\beta)$ และ $J_1(\beta)$

$$BW \approx 2\omega_m \qquad \text{สำหรับ } \beta \text{ ค่าน้อย} \qquad (2.25)$$

ในการหาแบนด์วิดธ์ของสัญญาณเอฟเอ็ม สำหรับการมอดูเลตสัญญาณรูปไซน์ความถี่เดียว หรือที่นิยมเรียกกันว่า โทนมอดูเลชัน (Tone modulation) นั้น เจอาร์ คาร์สัน (J.R. Carson) ได้เสนอสูตร

$$BW \approx 2(\Delta\omega + \omega_m) \qquad (2.26)$$

หรือเขียนในรูปใหม่ได้เป็น

$$BW \approx 2\omega_m(\beta + 1) \qquad (2.27)$$

สมการ (2.26) และ (2.27) รู้จักกันดีในชื่อของกฎของคาร์สัน (Carson's rule) อย่างไรก็ดีสำหรับสัญญาณข่าวสารทั่วไป กฎนี้จะใช้ได้ดี เมื่อ $\Delta\omega << \omega_m$ หรือเมื่อเงื่อนไข $\Delta\omega_m << \omega_m$ ไม่เป็นจริง ดังนั้นในกรณีของเอฟเอ็มแบนด์กว้างหรือแบนด์ปานกลาง ควรใช้ความสัมพันธ์

$$BW \approx 2(\Delta\omega + 2\omega_m) \qquad (2.28)$$

$$BW \approx 2\omega_m(\beta + 2) \qquad (2.29)$$

จะให้ผลลัพธ์ที่มีประสิทธิผลกว่ากฎของคาร์สัน ดังนั้นจึงสรุปเป็นสูตรแบนด์วิดธ์ได้ว่า

$$BW \approx 2\omega_m(\beta + k) \qquad \text{เรเดียน/วินาที} \qquad (2.30)$$

โดย k = 1 เมื่อ β มีค่าน้อย และ k = 2 เมื่อ β มีค่ามาก และถ้าเราคิดว่า คือ ค่าแบนด์วิดธ์ของสัญญาณข่าวสาร $m(t)$ ซึ่งเท่ากับ $2\pi\beta$ เรเดียน/วินาที เราจะพบว่า ค่าแบนด์วิดธ์ตามสมการ (2.30) จะแปลงเป็นค่าแบนด์วิดธ์ในหน่วยของเฮิร์ตซ์ ได้ดังนี้ คือ

$$BW \approx 2f_m(\beta + k) \qquad (2.31)$$

การสร้างสัญญาณเอฟเอ็ม นั้นสามารถแบ่งได้กว้างๆ เป็น 2 แบบด้วยกันคือ แบบทางตรงกับแบบทางอ้อม แบบทางตรงนั้นจะหมายถึง การนำสัญญาณที่เข้ามามอดูเลตไปเปลี่ยนความถี่ของคลื่นพาห์โดยตรง สำหรับแบบทางอ้อมนั้นก็จะเป็นวิธีการที่อาศัยวงจรหลายๆ วงจรประกอบกัน

1. *การสร้างสัญญาณเอฟเอ็มแบบทางตรง* เป็นการนำเอาสัญญาณที่จะเข้ามามอดูเลตไปเปลี่ยนความถี่ของคลื่นพาห์โดยตรง ซึ่งออสซิลเลเตอร์ที่ทำงานในลักษณะนี้ได้ก็มีรีเฟลกซ์ไคลสตรอน (Reflex klystron) และกันน์ไดโอด (Gunn diode) ที่ใช้ในย่านความถี่ไมโครเวฟ เป็นต้น ออสซิลเลเตอร์ทั้งสองแบบนี้ ความถี่ที่ออสซิลเลตออกมาจะเปลี่ยนไปตามแรงดันไบอัสในรูปเชิงเส้น ดังนั้นเมื่อนำสัญญาณที่เข้ามามอดูเลตไปเปลี่ยนแรงดันไบอัสก็จะทำให้ได้สัญญาณเอฟเอ็มออกมาโดยตรง สำหรับความถี่ในย่านต่ำกว่าไมโครเวฟลงมานั้น เรามักจะใช้ชิ้นส่วนอิเล็กทรอนิกส์ที่เรียกว่าวาแรคเตอร์ (Varactor หรือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Variable reactor) ชิ้นส่วนนี้จะมีคุณสมบัติเฉพาะคือ ค่าคาปาซิแตนซ์จะเปลี่ยนแปลงไปตามระดับของแรงดันที่คร่อมอยู่ ดังนั้นถ้าเราใช้วาแรคเตอร์นี้เป็นส่วนหนึ่งของวงจรเรโซแนนท์ที่ใช้ในวงจรออสซิลเลเตอร์ เราก็จะสามารถสร้างสัญญาณเอฟเอ็มได้ โดยป้อนสัญญาณที่เข้ามามอดูเลตไปเปลี่ยนค่าคาปาซิแตนซ์ของวาแรกเตอร์ ข้อดีของวงจรนี้คือทำได้ง่ายและค่าเบี่ยงเบนความถี่ที่ได้จะสูงพอสมควร แต่ก็มีข้อเสียที่สำคัญคือ ค่า f_c จะดริฟท์ (drift) ได้ง่าย จึงจำเป็นต้องมีวงจรชดเชยอุณหภูมิและวงจรควบคุมความถี่ f_c ให้คงที่ซึ่งก็จะเพิ่มความยุ่งยากของวงจรให้สูงขึ้น

2. *การสร้างสัญญาณเอฟเอ็มแบบทางอ้อม* จะใช้วงจรอินทิเกรเตอร์ในการอินทิเกรตสัญญาณที่เข้ามามอดูเลต และส่วนที่เป็นบาลานซ์มอดูเลเตอร์ เฟสชิพท์เตอร์และวงจรบวกจะทำหน้าที่เป็นเฟสมอดูเลเตอร์ ซึ่งจะทำให้สัญญาณที่ผ่านวงจรส่วนนี้จะเป็นสัญญาณเอฟเอ็มที่มีค่าเบี่ยงเบนความถี่ต่ำ ซึ่งหมายถึงสัญญาณที่ได้ในขั้นนี้ก็จะเป็นสัญญาณเอฟเอ็มแบบแบนด์วิดธ์แคบ วิธีสร้างสัญญาณแบบทางอ้อมที่กล่าวมานี้มีชื่อเรียกว่า วิธีมอดูเลตแบบอาร์มสตรอง (Armstrong's modulation method) กรณีที่ต้องการสัญญาณเอฟเอ็มแบบแบนด์วิดธ์กว้างก็จะต้องมีการออกแบบตั้งแต่ต้น คือ ใช้วงจรคูณความถี่เข้าช่วยจะทำให้ความถี่ของคลื่นพาห์สูงขึ้นพร้อม ๆ กับทำให้ค่าเบี่ยงเบนความถี่สูงขึ้น ข้อสังเกตที่สำคัญของการคูณความถี่ที่กล่าวมานี้ก็คือ ค่ามอดูเลชันอินเด็กซ์สูงขึ้น ซึ่งผลที่ตามมาก็คือค่าเบี่ยงเบนความถี่ หรือ Δf จะสูงขึ้น นั่นหมายถึงความว่าถ้าสัญญาณเบสแบนด์ที่เข้ามามอดูเลตมีความถี่เป็นเอฟเอ็ม ขนาดของไซด์แบนด์ก็จะกว้างออกไป โดยที่ช่วงห่างระหว่างสเปกตรัมที่อยู่ติดกันในไซด์แบนด์ก็ยังเป็นเอฟเอ็มเหมือนเดิม แต่สิ่งที่แตกต่างไปจากเดิมก็คือ ขนาดของสเปกตรัมจะเปลี่ยนแปลงไปจากเดิม

ส่วนการดีมอดูเลตสัญญาณเอฟเอ็มนั้นจะทำได้หลายแบบด้วยกัน แบบที่ใช้อย่างกว้างขวางมี 3 แบบ คือ

1. *การดีมอดูเลตสัญญาณเอฟเอ็มโดยการแปลงเป็นสัญญาณเอเอ็ม* เป้าหมายของการแปลงสัญญาณเอฟเอ็มเป็นสัญญาณเอเอ็มนั้นก็เพื่อให้สามารถดีเทกต์สัญญาณเบสแบนด์กลับออกมาได้โดยวิธีตรวจจับกรอบสัญญาณ การแปลงสัญญาณเอฟเอ็มเป็นสัญญาณเอเอ็มจะใช้วงจรที่สามารถแปลงการเบี่ยงเบนความถี่ชั่วขณะให้เป็นการเปลี่ยนแปลงของขนาดได้ วงจรที่ทำหน้าที่นี้ได้ดีคือ วงจรดิฟเฟอเรนชิเอเตอร์ ทั้งนี้เนื่องจากเมื่อเราดิฟเฟอร์เรนชิเอตสัญญาณเอฟเอ็ม สัญญาณที่ได้จะเป็นสัญญาณที่มีขนาดเปลี่ยนแปลงเหมือนกับสัญญาณเอเอ็ม ส่วนความถี่จะเปลี่ยนแปลงเหมือนเอฟเอ็ม จากนั้นเรานำเอาท์พุตของวงจรดิฟเฟอเรนชิเอเตอร์ไปเข้าวงจรเอนเวโลปดีเทกชันก็จะได้สัญญาณข้อมูลข่าวสารกลับมาเหมือนเดิม

2. *ควอดราเจอร์ดีเทกชัน (Quadrature detection)* การดีมอดูเลตสัญญาณเอฟเอ็มแบบควอดราเจอร์ดีเทคชันนั้นเป็นการใช้วงจรแยกแยะเฟส (Phase discriminator) ในการดึงเอาสัญญาณเบสแบนด์ออกมาจากสัญญาณเอฟเอ็ม ซึ่งวงจรแยกแยะเฟสจะอาศัยหลักการพื้นฐานของการดิฟเฟอเรนชิเอตควอดราเจอร์ดีเทกชันนั้นเป็นชื่อเรียกระบบดีเทกชันที่ตั้งการหน่วงเวลาของวงจรหน่วงเวลาให้มีค่า τ ที่ทำให้ $\omega_c \tau = \pi/2$

3. *เฟสล็อกลูป* การใช้เฟสล็อกลูปในการดีมอดูเลตสัญญาณเอฟเอ็มนั้นจะอาศัยหลักการที่วงจรเฟสล็อกลูปพยายามที่จะล็อกหรือซิงโครไนซ์ มุมชั่วขณะของสัญญาณจากวงจรโวลต์เตจคอน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โทรลออสซิลเลเตอร์เข้ากับมุมชั่วขณะของสัญญาณอินพุตที่เข้ามา การล็อกมุมชั่วขณะนี้จะหมายถึงความถี่และเฟสของสัญญาณทั้งสองตรงกันพอดี ส่วนประกอบพื้นฐานของวงจรเฟสล็อกลูปจะประกอบด้วยวงจรเปรียบเทียบเฟส และวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ซึ่งเป็นวงจรออสซิลเลเตอร์ที่ความถี่ออสซิลเลตถูกควบคุมด้วยแรงดันจากสัญญาณภายนอก จึงมีลักษณะเหมือนวงจรมอดูเลเตอร์ของสัญญาณเอฟเอ็ม เฟสล็อกลูปโดยทั่วไปจะสามารถจับความถี่ได้ในช่วงหนึ่ง กล่าวคือ ในขณะที่เฟสล็อกลูปทำงานอย่างอิสระ (Free running) แล้วเกิดมีสัญญาณความถี่อื่นเข้ามาเฟสล็อกลูปจะพยายามติดตามเพื่อจับความถี่นั้น ช่วงความถี่ที่เฟสล็อกลูปสามารถติดตามจับได้นี้เรียกว่าช่วงความถี่ตรวจจับ (Capture range) และเมื่อเฟสล็อกลูปติดตามจับความถี่ได้แล้ว ช่วงความถี่ที่เฟสล็อกลูปจะติดตามจับได้ตลอดเวลาจะเรียกว่าช่วงความถี่ล็อก (Lock range) โดยทั่วไปช่วงความถี่ล็อกจะกว้างกว่าช่วงความถี่ตรวจจับ ขณะที่เฟสล็อกลูปล็อกแล้วเอาท์พุตของลูปฟิลเตอร์ก็คือสัญญาณข้อมูลข่าวสารนั่นเอง

2.7 วงจรสังเคราะห์ความถี่ (Frequency Synthesizer)

วงจรสังเคราะห์ความถี่เป็นวงจรที่ทำหน้าที่ผลิตสัญญาณความถี่ขนาดพอเหมาะตามที่เรากำหนด โดยช่วงความถี่ใช้งานของวงจรสังเคราะห์ความถี่จะจำกัดอยู่ในช่วงความถี่ที่แน่นอน วิธีสังเคราะห์ความถี่แบ่งออกเป็น 2 วิธี คือ

1. *วิธีสังเคราะห์โดยตรง (Direct Synthesis)* ทำการสังเคราะห์ความถี่โดยนำความถี่หลายค่ามาผสมกันเพื่อให้ได้ความถี่ที่ต้องการ

2. *วิธีสังเคราะห์โดยอ้อม (Indirect Synthesis)* ทำการสังเคราะห์ความถี่โดยใช้หลักการของเฟสล็อกลูป (Phase Lock Loop) ในการผลิตความถี่ขึ้นมา

ซึ่งในที่นี้เราจะใช้วิธีการสังเคราะห์ความถี่โดยอ้อม โดยอาศัยหลักการทำงานของเฟสล็อกลูปที่ว่า ระบบเฟสล็อกลูปเป็นระบบควบคุมความถี่ที่ใช้วิธีเปรียบเทียบเฟสของความถี่ทางด้านเอาท์พุตกับเฟสของความถี่อ้างอิงซึ่งถูกป้อนเข้าทางด้านอินพุตของระบบ โดยสามารถให้สัญญาณออกที่มีความถี่เท่ากับความถี่ของสัญญาณเข้าได้ในเวลาอันสั้น และเมื่อเกิดการล็อกขึ้นแล้ว ถ้าความถี่ของสัญญาณเข้ามีการเปลี่ยนแปลง ระบบเฟสล็อกลูปยังคงรักษาสภาพการล็อกนั้นอยู่ ซึ่งมีผลให้สัญญาณออกยังคงมีความถี่เท่ากับสัญญาณเข้าที่มีการเปลี่ยนแปลงความถี่ไปแล้ว

ระบบเฟสล็อกลูปประกอบด้วยส่วนสำคัญ 3 ส่วน ดังนี้

1. *เฟสดีเทคเตอร์* ทำหน้าที่เปรียบเทียบเฟสของอินพุตซึ่งจะมีผลของเฟสที่ต่างกันเรียกว่าความผิดพลาดเฟส (Phase Error) ความผิดพลาดเฟสนี้จะมีค่าน้อยที่สุดเป็นศูนย์ และจะมีค่ามากที่สุดเป็น $\pi/2$ เฟสดีเทคเตอร์จะทำการเปลี่ยนความผิดพลาดเฟสนี้ให้กลายเป็นระดับแรงดันด้วยค่าคอนเวอร์ชันเกน (Conversion gain)

1.1 *เฟสดีเทคเตอร์แบบที่ 1* จะถูกออกแบบมาเพื่อทำการตรวจจับคลื่นสี่เหลี่ยมต่างๆ ทั้งทางอนาลอกหรือทางดิจิตอล ซึ่งเฟสดีเทคเตอร์แบบที่ 1 จะมีความสัมพันธ์เป็นเชิงเส้นสำหรับเฟสอินพุตในช่วงหนึ่งตั้งแต่ 0 - π

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.2 *เฟสดีเทคเตอร์แบบที่ 2* จะทำการตรวจการเปลี่ยนแปลงสถานะหรือขอบสัญญาณโดยวงจรเฟสดีเทคเตอร์ชนิดนี้จะผลิตพัลส์บวกหรือลบขึ้นอยู่กับการนำหน้า (lead) หรือตามหลัง (lag) ของสัญญาณจากวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ เมื่อเปรียบเทียบกับสัญญาณอ้างอิง ความกว้างของพัลส์เหล่านี้จะเท่ากับช่วงกว้างระหว่างขอบของสัญญาณอินพุตทั้งสองของเฟสดีเทคเตอร์และจะเป็นตัวกำหนดขนาดของแรงดันที่ป้อนให้วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ ข้อดีของเฟสดีเทคเตอร์ชนิดนี้คือ การมีเอาท์พุตที่ไม่ขึ้นกับดิวตีไซเคิล (duty cycle) ของสัญญาณอินพุตและการไม่มีการกระเพื่อม (Ripple) ที่เอาท์พุตเหมือน ในกรณีของเฟสดีเทกเตอร์แบบที่ 1

2. *ลูปฟิลเตอร์* ทำหน้าที่กรองสัญญาณความถี่สูงที่ออกมาจากเฟสดีเทกเตอร์ เนื่องจากเฟสดีเทคเตอร์ให้เอาท์พุตเป็นสัญญาณไฟฟ้ากระแสตรงที่มีแรงดันไฟฟ้ากระแสสลับร่วมกรองเอาสัญญาณความถี่ที่ได้เกิดจากความต่างเฟสมากออก ทำให้ระบบสามารถจับ (Capture) สัญญาณได้ในช่วงหนึ่งและช่วยให้ระบบรักษาการล็อกไว้ได้อีกด้วย และใช้ปรับขนาดสัญญาณไฟตรงเพื่อให้การควบคุมดีขึ้น เอาท์พุตของวงจรนี้จะป้อนให้แก่วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์

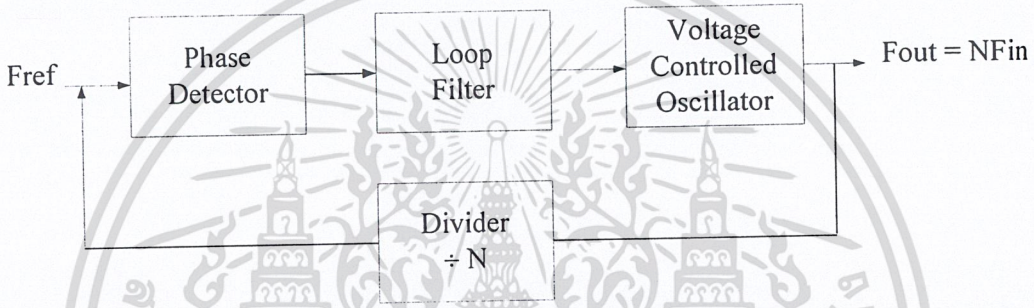
3. *วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์* ทำหน้าที่ผลิตความถี่สัญญาณโดยการควบคุมระดับแรงดันเอาท์พุตด้วยคอนเวอร์ชันเกน ระดับของแรงดันนี้จะได้จากเอาท์พุตของลูปฟิลเตอร์ ความถี่ที่ล็อกมาจากลูปฟิลเตอร์จะมีผลทำให้เอาท์พุตของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์เปลี่ยนความถี่ด้วยเช่นกัน เนื่องจากวงจรออสซิลเลเตอร์เป็นวงจรผลิตความถี่สัญญาณจึงเป็นส่วนที่จำเป็นสำหรับวงจรเฟสล็อกลูป วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ไม่จำเป็นต้องมีความสัมพันธ์เป็นเชิงเส้นกับแรงดันไฟฟ้าที่มาควบคุมมากนักในกรณีทั่วไป แต่ถ้าไม่เป็นเชิงเส้นมากเกินไปแล้วอัตราขยายรอบก็จะเปลี่ยนแปลงไปตามความถี่ของสัญญาณด้วย ฉะนั้นจึงต้องคำนึงถึงความเสถียรของลูปด้วย

การทำงานของเฟสล็อกลูป ในขณะเริ่มแรกที่ยังไม่มีสัญญาณเข้าไปในระบบ แรงดันควบคุม (Control Voltage) จะเท่ากับ 0 วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์จะทำงานโดยตั้งความถี่ไว้ที่ f_r ซึ่งเรียกว่า ความถี่อิสระ (Free running Frequency) เมื่อมีสัญญาณเข้าไปในระบบ วงจรเปรียบเทียบเฟสจะทำการเปรียบเทียบเฟสและความถี่ของสัญญาณอินพุตกับความถี่ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์และผลิตแรงดันคลาดเคลื่อน (Error Voltage) ซึ่งสัมพันธ์กับความแตกต่างของเฟสและความถี่ระหว่างสัญญาณทั้งสอง แรงดันคลาดเคลื่อนนี้จะถูกกรองแล้วส่งไปยังขาควบคุมของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ ในการนี้แรงดันควบคุมจะไปบังคับความถี่ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ให้เปลี่ยนไปในทิศทางที่จะลดความถี่ที่แตกต่างระหว่างแรงดันขาออก (Output Voltage) กับสัญญาณที่เข้ามา ถ้าความถี่ของสัญญาณที่เข้าใกล้เคียงกับแรงดันขาออกจากการป้อนกลับของเฟสล็อกลูป จะทำให้ล็อกกับสัญญาณที่เข้ามา ขณะที่ทำการล็อกนั้นความถี่ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์จะเท่ากับสัญญาณอินพุตแต่เฟสยังต่างกันอยู่ มีความจำเป็นต่อการผลิตแรงดันคลาดเคลื่อนที่จะไปคอยปรับความถี่ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์จากค่าความถี่อิสระให้เท่ากับความถี่ที่เข้ามา ดังนั้นเฟสล็อกลูปจะยังคงรักษาสภาพการล็อก การที่ระบบสามารถที่จะปรับตัวเองได้ ทำให้เฟสล็อกลูปสามารถติดตามการล็อกกับสัญญาณที่เข้าไปให้อยู่ในสภาพล็อกเช่นเดิม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

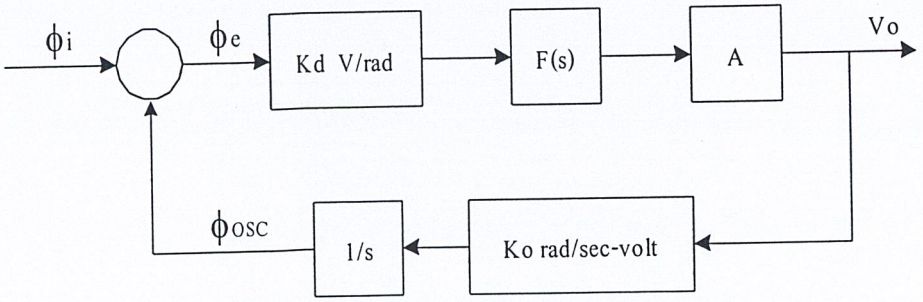
เฟสล็อกลูปโดยทั่วไปจะสามารถจับความถี่ได้ในช่วงหนึ่ง กล่าวคือในขณะที่เฟสล็อกลูปทำงานอย่างอิสระ (Free running) แล้วเกิดมีสัญญาณความถี่อื่นเข้ามา เฟสล็อกลูปก็จะพยายามติดตามเพื่อจับความถี่นั้น ช่วงความถี่ที่เฟสล็อกลูปสามารถติดตามจับได้ตลอดเวลาเรียกว่า ช่วงความถี่ตรวจจับ (Capture Range) และเมื่อเฟสล็อกลูปติดตามจับความถี่ได้แล้วช่วงที่เฟสล็อกลูปจะติดตามจับได้ตลอดเวลาเรียกว่าช่วงความถี่ล็อก (Lock Range) ซึ่งโดยทั่วไปแล้วช่วงความถี่ล็อกจะกว้างกว่าช่วงความถี่ตรวจจับ

วงจรสังเคราะห์ความถี่ เป็นแหล่งกำเนิดความถี่ที่สัญญาณเอาท์พุตสามารถให้ความถี่เป็นทวีคูณ (Multiple) ของสัญญาณความถี่อ้างอิงได้โดยการติดตั้งวงจรหารความถี่เพิ่มเข้าไปในส่วนป้อนกลับ ทำให้เราได้ความถี่เอาท์พุตมีขนาดเป็นความถี่ N เท่าของความถี่อ้างอิง เมื่อเฟสล็อกลูปอยู่ในสภาวะล็อก ซึ่งแสดงบล็อกไดอะแกรมได้เป็นดังรูปที่ 2.19



รูปที่ 2.19 บล็อกไดอะแกรมของวงจรสังเคราะห์ความถี่

วงจรสังเคราะห์ความถี่โดยใช้เฟสล็อกลูปมีหลักการทำงานคือ วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์จะผลิตสัญญาณที่มีความถี่ค่าหนึ่งออกมา ผ่านวงจรหารความถี่หรือพรีสเกลเลอร์ (Prescaler) จากนั้นสัญญาณที่ได้จะถูกทำการเปรียบเทียบเฟสกับสัญญาณอ้างอิง (Reference frequency) โดยเฟสดีเทกเตอร์จะทำการเปรียบเทียบเฟส และทำการแปลงความต่างเฟสให้กลายเป็นแรงดันไฟตรงค่าหนึ่งซึ่งมีการกระเพื่อม (Ripple) จากนั้นลูปฟิลเตอร์จะทำการกรองแรงดันไฟตรงดังกล่าวให้ได้เป็นแรงดันไฟตรงที่ปราศจากการกระเพื่อม เพื่อใช้เป็นแรงดันควบคุมวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ให้ผลิตสัญญาณที่มีความถี่ค่าใหม่ออกมา และจะเป็นเช่นนี้เรื่อยไปจนกระทั่งความต่างเฟสที่ได้เป็นศูนย์ ซึ่งจะทำให้ลูปฟิลเตอร์ปล่อยแรงดันไฟตรงคงที่ค่าหนึ่งออกมาเป็นผลทำให้วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ผลิตสัญญาณที่มีความถี่คงที่ออกมาค่าหนึ่งตามที่ต้องการ ซึ่งการหาทรานส์เฟอร์ฟังก์ชันของระบบเฟสล็อกลูปสามารถทำได้ดังต่อไปนี้



รูปที่ 2.20 บล็อกไดอะแกรมของทรานส์เฟอร์ฟังก์ชันของเฟสล็อกลูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดย K_d = ทรานส์เฟอร์ฟังก์ชันของเฟสดีเทกเตอร์

$F(s)$ = ทรานส์เฟอร์ฟังก์ชันของลูปฟิลเตอร์

A = อัตราการขยายของวงจรขยายสัญญาณ

K_0 = ทรานส์เฟอร์ฟังก์ชันของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์

$$\omega = \frac{d\theta}{dt} \tag{2.32}$$

จาก $$\omega = s\theta \tag{2.33}$$

$$\theta = \int \omega \ dt \tag{2.34}$$

และ $$\theta = \frac{\omega}{s} \tag{2.35}$$

เนื่องจากเอาท์พุตของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์จะเป็นค่าความถี่ที่ขึ้นอยู่กับอินพุตที่เป็นสัญญาณไฟตรง แต่เราต้องการค่า ϕ_{OSC} ฉะนั้นจะต้องใส่บล็อก $1/s$ เพื่อเปลี่ยน ω_{OSC} ให้เป็น ϕ_i

จากทรานส์เฟอร์ฟังก์ชัน

$$\left(\frac{V_o}{\phi_i}\right) = \frac{A}{1+AF} \tag{2.36}$$

เมื่อ A = อัตราการขยายไปข้างหน้า

B = อัตราการขยายป้อนกลับ

$$\frac{V_o}{\phi_i} = \frac{KdF(s)A}{1+K_dF(s)A\frac{K_o}{s}} \tag{2.37}$$

$$= \frac{sK_dF(s)A}{s+K_dF(s)K_o} \tag{2.38}$$

โดยปกติเรามักจะใช้ตัวแปรอินพุตเป็นความถี่มากกว่าเฟส ดังนั้นทรานส์เฟอร์ฟังก์ชัน คือ

$$H(s) = \frac{V_o}{\omega_i} = \frac{V_o}{s\phi_i} \tag{2.39}$$

$$= \frac{K_dF(s)A}{s+K_dK_oAF(s)} \tag{2.40}$$

วงจรเฟสล็อกลูปที่ใช้ในระบบสื่อสารทั่วไป จะต้องให้มีพิสัยการล็อกกว้างเพื่อที่จะได้สามารถติดตามการเปลี่ยนแปลงความถี่ของสัญญาณอินพุตได้ในช่วงกว้าง นอกจากนั้นยังต้องการให้ระบบมีแบนด์วิดธ์ของลูปแคบ ๆ เพื่อกำจัดสัญญาณที่อยู่นอกแบนด์

พิกัดความถี่ตลอดช่วงที่ระบบเฟสล็อกลูปสามารถดำรงการล็อกไว้ได้กับสัญญาณอินพุต เราเรียกว่าพิสัยการล็อก เราสามารถหาพิสัยการล็อกได้โดยพิจารณาจากสมการของความผิดพลาดเฟส ซึ่งเราพบว่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความผิดพลาดเฟสจะมีค่ามากที่สุดเท่ากับ π น้อยที่สุดเท่ากับศูนย์ จะได้พิสัยการล็อก $\Delta\omega = \omega_i - \omega_o$ เท่ากับ

$$\Delta\omega = (\theta_e - \pi/2)K_o K_d F(s) \tag{2.41}$$

แทนค่า $\theta_{e\max} = \pi$ และ $\theta_{e\min} = 0$

$$\Delta\omega = (\pm\pi/2)K_o K_d F(s) \tag{2.42}$$

$$2\Delta\omega_L = \pm\pi K_o K_d F(s) \tag{2.43}$$

เมื่อ $\Delta\omega_L$ คือ พิสัยการล็อก

เราจะสังเกตได้ว่า ถ้า ω_i เบี่ยงเบนไปจากความถี่ศูนย์กลาง ω_o มาก สภาวะสมดุลของการซิงโครนัสจะไม่สามารถดำรงอยู่ต่อไปได้ และลูปจะหลุดออกไปจากการล็อกกับความถี่ของสัญญาณอินพุต

การวิเคราะห์ถึงพิสัยการล็อกของระบบต้องมีสภาวะเดิมล็อกอยู่กับความถี่อินพุตแล้ว แต่มีอีกอย่างที่เราควรพิจารณาเมื่ออยู่ในสภาวะที่ระบบยังไม่ล็อกกับสัญญาณอินพุต เพื่อกำหนดช่วงความถี่ ซึ่งระบบเฟสล็อกลูปสามารถล็อกกับสัญญาณอินพุตในช่วงความถี่ดังกล่าว ความถี่ช่วงนี้เราเรียกว่า พิสัยแคปเจอร์ ซึ่งจะสัมพันธ์กับค่าความถี่คัตออฟของลูปฟิลเตอร์ ω_i และ พิสัยการล็อก $\Delta\omega_L$

ในกรณีทั่ว ๆ ไป $\Delta\omega_L >> \omega_i$

$$\Delta\omega_c = \pm\sqrt{\omega_i \Delta\omega_L} \tag{2.44}$$

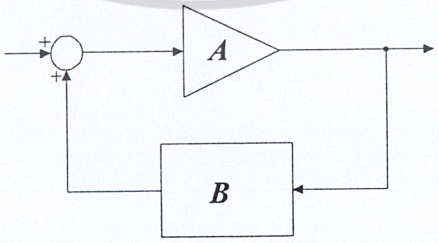
$$2\Delta\omega_c = 2\sqrt{\omega_i \Delta\omega_L} \tag{2.45}$$

เมื่อ $\Delta\omega_c$ คือ พิสัยแคปเจอร์

รายละเอียดของส่วนประกอบต่างๆ ของวงจรสังเคราะห์ความถี่ แสดงได้ดังต่อไปนี้

2.7.1 วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ (Voltage Controlled Oscillator)

การออกแบบวงจรกำเนิดสัญญาณอาศัยหลักการป้อนกลับแบบบวก (Positive feedback) ตามบล็อกไดอะแกรมในรูปที่ 2.21



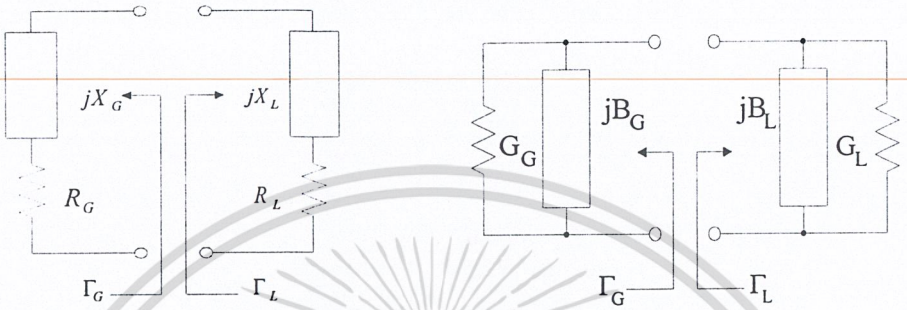
รูปที่ 2.21 บล็อกไดอะแกรมของระบบป้อนกลับแบบบวก

โดย A คือ อัตราขยายของวงจรขยายในส่วนของฟอร์เวิร์ดพาธ (Forward Path)

B คือ อัตราส่วนการป้อนกลับในส่วนของฟีดแบคพาธ (Feedback Path)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วงจรจะออสซิลเลตได้ต่อเมื่อ $|AB|$ = 1 (Barkhausen Criterion) และ $\angle AB$ = 360° ดังนั้นจากเงื่อนไขข้างต้น $A_{total} = \frac{A}{1-AB}$ จะมีค่า = ∞ เมื่อเกิดการออสซิลเลต วงจรขยาย(A)จะทำการขยายสัญญาณรบกวนซึ่งเป็นการเริ่มต้นให้วงจรเกิดการออสซิลเลต หลังจากนั้นความถี่ที่ถูกออสซิลเลตจะถูกเลือกโดยวงจรเรโซแนนซ์ซึ่งแบ่งออกได้เป็น 2 รูปแบบ คือ



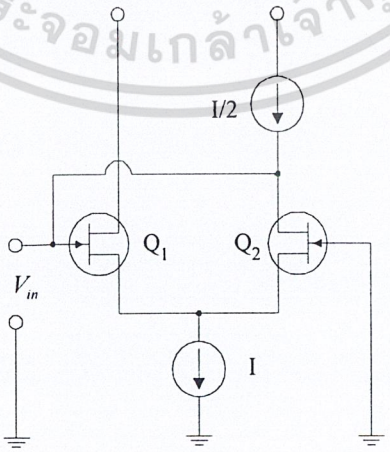
(A) วงจรเรโซแนนซ์แบบอนุกรม (B) วงจรเรโซแนนซ์แบบขนาน

รูปที่ 2.22 วงจรเรโซแนนซ์

สำหรับสภาวะเริ่มต้นการออสซิลเลต จะได้ $|R_G| > R_L$ เมื่อ R_G มีค่าความต้านทานเป็นลบและสำหรับสภาวะ steady state จะได้ $R_G + R_L = 0$ และ $X_G + X_L = 0$

สำหรับสภาวะเริ่มต้นการออสซิลเลตจะได้ $|G_G| > G_L$ เมื่อ G_G มีค่าความนำไฟฟ้าเป็นลบและสำหรับสภาวะ steady state จะได้ $G_G + G_L = 0$ และ $B_G + B_L = 0$

การออกแบบวงจรกำเนิดสัญญาณอาศัยหลักการของความต้านทานเชิงลบ (Negative resistance) มาใช้อธิบายเพราะง่ายกว่า ตัวอย่างของวงจรที่จะยกมาอธิบายอาศัยการอธิบายในเชิงความต้านทานเชิงลบเช่นเดียวกันโดยจากรูปที่ 2.23 แสดงวงจรกำเนิดสัญญาณโดยหากต้องการหาเงื่อนไขที่แสดงการออสซิลเลตสามารถคำนวณเริ่มต้นจาก



รูปที่ 2.23 วงจรกำเนิดสัญญาณที่ใช้วงจรขยายแบบดิฟเฟอเรนเชียล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เริ่มต้นคำนวณหาค่าความต้านทานด้านอินพุต (Input resistance)

$$i_o = g_m \frac{V_i}{2} \tag{2.46}$$

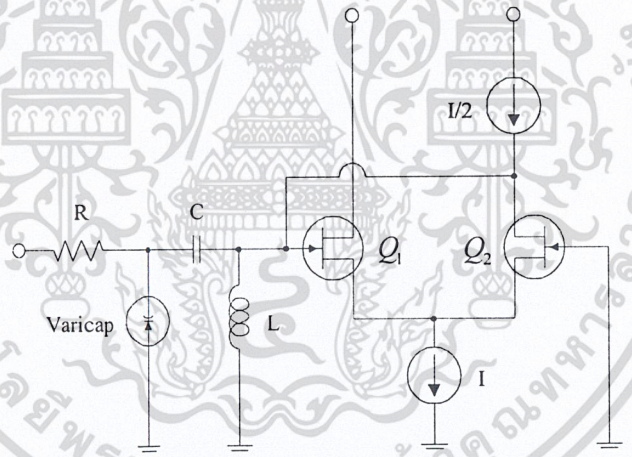
เมื่อสมมติให้ความต้านทานอินพุตของเฟต (FET) และแหล่งจ่ายกระแสคงที่มีค่ามาก ๆ ดังนั้น แรงดันที่ตกคร่อมขาเกต (Gate) จะเป็น

$$V_{gs} \approx \frac{V_{in}}{2} \tag{2.47}$$

$$\frac{V_{in}}{I_{in}} = \frac{-2}{g_m} \tag{2.48}$$

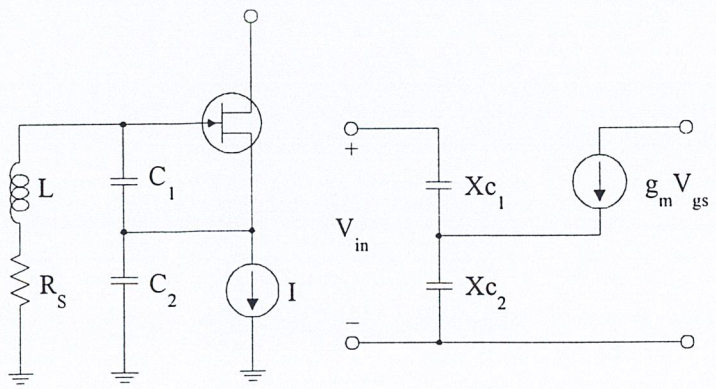
$$R_{in} = \frac{-2}{g_m} \tag{2.49}$$

ดังนั้นหากทำการใช้เรโซเนเตอร์ทางด้านอินพุตของออสซิลเลเตอร์ (V_{in}) เพื่อเลือกความถี่และใช้วาริแคปแทนตัวเก็บประจุจะได้วงจรออสซิลเลเตอร์ควบคุมได้ด้วยแรงดัน ตามรูปที่ 2.24 โดยไอซีสำเร็จรูปที่อาศัยองค์ประกอบภายในเป็นออสซิลเลเตอร์ประเภทนี้ เช่น MC1648 ทำงานได้ดีที่ความถี่ที่ต่ำกว่า 500 เมกะเฮิรตซ์



รูปที่ 2.24 วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์

วงจรเฟตออสซิลเลเตอร์ (FET Oscillator) สามารถแสดงได้ดังนี้



รูปที่ 2.25 วงจรเฟตออสซิลเลเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$V_{in} = i_{in}\left(X_{C_1} + X_{C_2}\right) + g_m V_{gs} X_{C_2} \tag{2.50}$$

$$V_{gs} = V_{xc1} = i_{in} X_{c1} \tag{2.51}$$

$$V_{in} = i_{in}\left(X_{C_1} + X_{C_2}\right) + g_m i_{in} X_{C_1} X_{C_2} \tag{2.52}$$

$$\frac{V_{in}}{I_{in}} = X_{C_1} + X_{C_2} + g_m X_{C_1} X_{C_2} \tag{2.53}$$

$$Z_{in} = \frac{-g_m}{\omega^2 C_1 C_2} + \frac{1}{j\omega[C_1 C_2 /(C_1 + C_2)]} \tag{2.54}$$

ซึ่งจะเห็นว่า ความต้านทานขาเข้าจะมีความต้านทานที่เป็นลบ

$$R_{in} = \frac{-g_m}{\omega^2 C_1 C_2} \tag{2.55}$$

ต่ออนุกรมอยู่กับ

$$C_{in} = \frac{C_1 C_2}{(C_1 + C_2)} \tag{2.56}$$

นำตัวเหนี่ยวนำ (L) มาต่อเข้ากับวงจรนี้โดย R_s คือความต้านทานแฝงที่อนุกรมอยู่กับตัวเหนี่ยวนำจะได้ว่าวงจรจะเริ่มทำงานได้โดยกำเนิดความถี่ที่

$$f_0 = \frac{1}{2\pi\sqrt{L\{C_1 C_2 /(C_1 + C_2)\}}} \tag{2.57}$$

ผลรวมของ C_1 และ C_2 ในแบบอนุกรมจะมีค่ามากที่สุดเมื่อ $C_1 = C_2 = C_{in}$ ดังนั้น

$$\sqrt{\frac{R_S}{g_m}} < \frac{1}{\omega C_m} \tag{2.58}$$

2.7.2 วงจรหารความถี่หรือพรีสเกลเลอร์ (Prescaler)

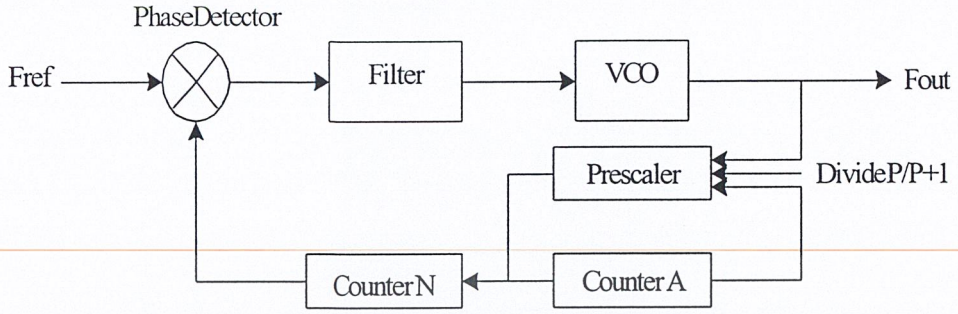
วงจรสังเคราะห์ความถี่มีอยู่หลายแบบ ตัวอย่างที่จะกล่าวถึงต่อไปนี้เป็นวงจรสังเคราะห์ความถี่เฟสล็อกลูปแบบพรีสเกลเลอร์สองโมดูลัส ซึ่งมีขั้นตอนการตั้งความถี่ขั้นละ f_R เท่ากับความถี่อ้างอิง วงจรเฟสล็อกลูปในรูปที่ 2.26 ใช้พรีสเกลเลอร์ซึ่งเป็นวงจรนับที่ตัวหารเปลี่ยนค่าได้ระหว่าง P กับ $P+1$ เราเรียกพรีสเกลเลอร์แบบนี้ว่า “พรีสเกลเลอร์สองโมดูลัส” วงจรนับหาร N_t ซึ่งโปรแกรมตัวหารได้นั้นทำงานที่ความถี่ต่ำ

เหตุผลที่ใช้พรีสเกลเลอร์แบบสองโมดูลัสก็เพื่อลดทอนความถี่ลงให้ใช้กับวงจรหาร N ตระกูล TTL และ CMOS ได้ ทำให้สามารถสังเคราะห์ความถี่ไปถึงย่าน VHF , UHF ได้

ส่วนสำคัญของวงจรคือพรีสเกลเลอร์แบบสองโมดูลัส ซึ่งเป็นไอซีตระกูล ECL มีความสามารถในการทำงานที่ความถี่สูง จะทำการหารล่วงหน้า (prescale) ก่อน และทำงานในลักษณะที่หารได้ 2 ค่า สลับกันในไอซีตัวเดียว โดยที่ตัวหารความถี่มีค่าต่างกันอยู่หนึ่ง คือ P และ $P+1$ เช่นหาร 64/65

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอาท์พุตของพรีสเกลเลอร์จะป้อนไปให้แก่วงจรเคาน์เตอร์ตระกูล TTL 2 ตัว ตัวหนึ่งเป็นเคาน์เตอร์หลัก (main counter) ส่วนอีกตัวหนึ่งเป็นเคาน์เตอร์เสริม



รูปที่ 2.26 วงจรสังเคราะห์ความถี่เฟสล็อกลูปแบบพรีสเกลเลอร์สองโมดูลัส

เคาน์เตอร์เสริมจะเป็นตัวบังคับให้พรีสเกลเลอร์หารด้วย P หรือ $P+1$ สมมุติว่าป้อนข้อมูลหรือพรีเซตตัวเลขให้เคาน์เตอร์เสริมและในขณะนี้พรีสเกลเลอร์ให้ $P+1$ เป็นตัวหาร เคาน์เตอร์เสริมจะนับก่อนหลังลงไปเรื่อย ๆ เคาน์เตอร์หลักจะนับไปพร้อม ๆ กับเคาน์เตอร์เสริม เมื่อเคาน์เตอร์เสริมหยุดนับจะส่งสัญญาณบังคับให้พรีสเกลเลอร์เปลี่ยนตัวหารด้วย P และ เคาน์เตอร์หลักจะนับถอยหลังต่อไปจนถึงศูนย์ เมื่อเคาน์เตอร์หลักและเคาน์เตอร์เสริมนับถึงศูนย์เมื่อใด ทั้งคู่จะถูกพรีเซตด้วยตัวเลขข้อมูลใหม่อีกครั้ง ดังนั้นตัวเลขที่พรีเซตให้เคาน์เตอร์เสริมจะต้องน้อยกว่าตัวเลขที่พรีเซตให้เคาน์เตอร์หลัก

สมมุติตัวเลขที่พรีเซตให้ตัวเคาน์เตอร์หลักเป็น N และเคาน์เตอร์เสริมเป็น A เริ่มแรกให้พรีสเกลเลอร์อยู่ในสภาวะหาร $P+1$ ซึ่งจะยังคงหารด้วย $P+1$ ไปจนกว่าเคาน์เตอร์เสริมจะนับลงเป็นศูนย์ นั่นคือเวลาที่ใช้ในการนับของเคาน์เตอร์เสริมคิดเป็นจำนวนไซเคิลของวงจรออสซิลเลเตอร์ควบคุมโดยแรงดันผ่านไปเท่ากับ $P+1$ คูณด้วย A ไซเคิล

หลังจากนั้นพรีสเกลเลอร์จะถูกบังคับให้เปลี่ยนตัวหารเป็น P โดยสัญญาณบังคับจากเคาน์เตอร์ A ยังเหลืออยู่ $N-A$ นั่นคือจะต้องใช้เวลาในการนับเคาน์เตอร์หลักคิดเป็นจำนวนไซเคิลของวงจรออสซิลเลเตอร์ควบคุมโดยแรงดันที่ผ่านไปเท่ากับ P คูณด้วย $N-A$

ดังนั้นรวมค่าตัวหารทั้งหมดจะได้เท่ากับ

$$Nt = (P+1)A + P(N-A) \quad (2.59)$$

$$= PN + A \quad (2.60)$$

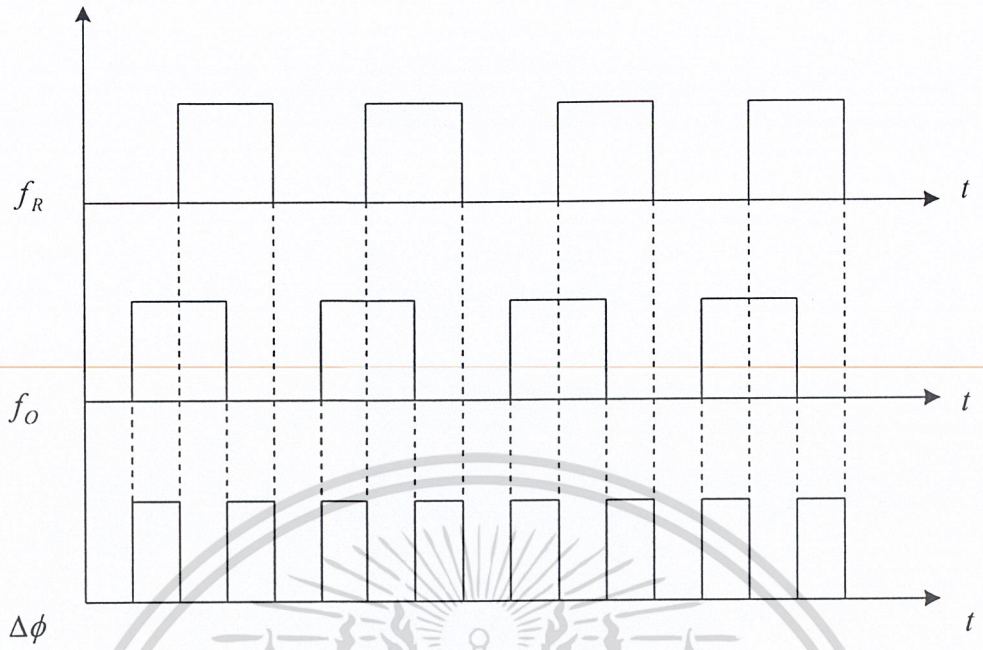
ความถี่ออสซิลเลเตอร์ควบคุมโดยแรงดันจะเท่ากับ $PN+A$ เท่าของความถี่อ้างอิง

$$F_{out} = F_{ref}\ (PN+A) \quad (2.61)$$

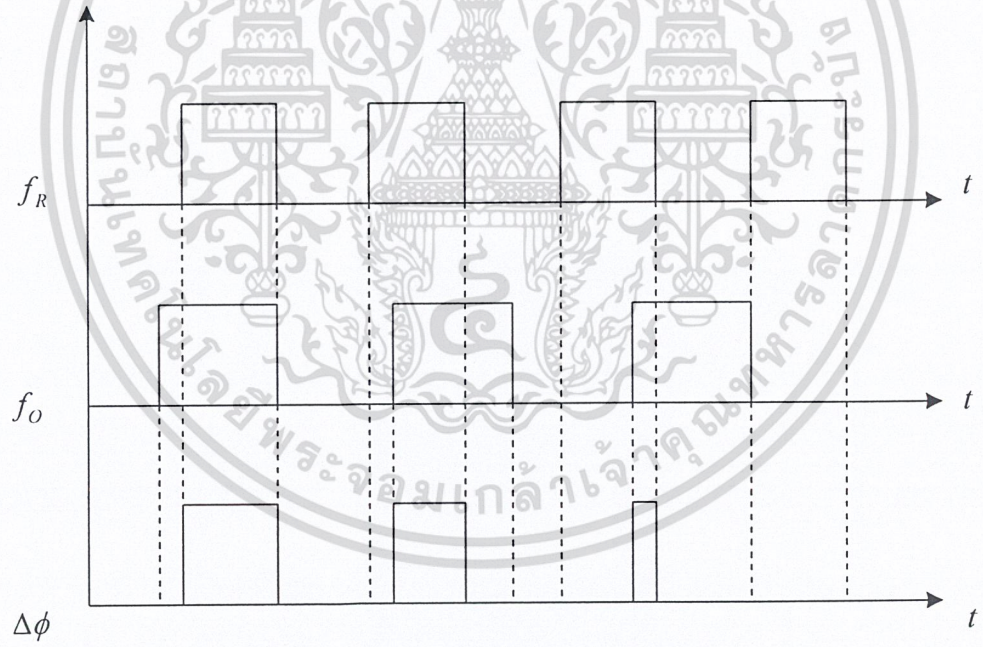
2.7.3 เฟสดีเทกเตอร์ (Phase Detector)

เฟสดีเทกเตอร์ ทำหน้าที่เปรียบเทียบเฟสของอินพุตซึ่งจะมีผลที่แตกต่างกันเรียกว่าความผิดพลาดทางเฟส (Phase Error) ความผิดพลาดทางเฟสนี้จะมีค่าน้อยสุดเป็นศูนย์ และจะมีค่ามากที่สุดเป็น $\pi/2$ ลักษณะการเปรียบเทียบเฟสของอินพุตทั้งสองของเฟสดีเทกเตอร์จะแสดงดังรูปที่ 2.27

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



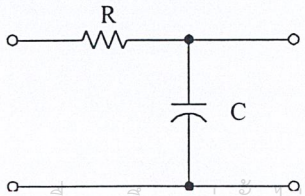
รูปที่ 2.27 ผลต่างเฟสเมื่อความถี่อินพุตเท่ากัน



รูปที่ 2.28 ผลต่างเฟสเมื่อความถี่อินพุตไม่เท่ากัน

2.7.4 ลูปฟิลเตอร์ (Loop Filter)

การออกแบบวงจรลูปฟิลเตอร์อันดับที่ 1



รูปที่ 2.29 วงจรลูปฟิลเตอร์อันดับที่ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$F(S) = \frac{1}{S\tau + 1} \tag{2.62}$$

จะได้ทรานส์เฟอร์ฟังก์ชันแบบลูปปิด (Closed Loop Transfer Function) คือ

$$T(S) = \frac{K_V (1/S\tau + 1)}{S + K_V (1/S\tau + 1)} \tag{2.63}$$

$$= \left(\frac{K_V}{S(S\tau + 1) + K_V}\right) \tag{2.64}$$

$$= \frac{K_V/\tau}{S^2 + S/\tau + K_V/\tau} \tag{2.65}$$

เทียบกับทรานส์เฟอร์ฟังก์ชันมาตรฐานของวงจรกรองความถี่ต่ำผ่าน

$$T(S) = \frac{\omega_n^2}{S^2 + 2\xi\omega_n S + \omega_n^2} \tag{2.66}$$

จะได้ว่า

$$\omega_n^2 = \frac{K_V}{\tau} \tag{2.67}$$

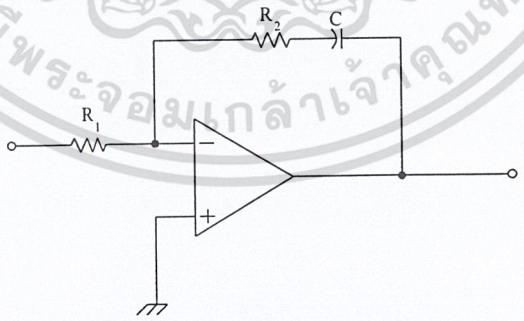
$$2\xi\omega_n = \frac{1}{\tau} \tag{2.68}$$

$$\xi = \frac{1}{2\sqrt{\tau K_V}} \tag{2.69}$$

เมื่อ ω_n = Natural Angular Frequency

ξ = Damping Factor

การออกแบบวงจรลูปฟิลเตอร์อันดับที่ 2



รูปที่ 2.30 วงจรลูปฟิลเตอร์อันดับที่ 2

$$F(S) = \frac{R_2 + \frac{1}{SC}}{R_1} \tag{2.70}$$

$$= \frac{SR_2C + 1}{SR_1C} \tag{2.71}$$

$$= \frac{S\tau_2 + 1}{S\tau_1} \tag{2.72}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$T(S) = \frac{K_V F(S)}{S + K_V F(S)} \tag{2.73}$$

$$= \frac{K_V (S\tau_2 + 1)\ S\tau_1}{S + K_V (S\tau_2 + 1)\ S\tau_1} \tag{2.74}$$

$$= \frac{K(S\tau + 1)/\tau_1}{S^2 + K_V \frac{\tau_2}{\tau_1} S + \frac{K_V}{\tau_1}} \tag{2.75}$$

$$T(S) = \frac{2\xi\omega_n S + \omega_n^2}{S^2 + 2\xi\omega_n S + \omega_n^2} \tag{2.76}$$

จะได้ว่า

$$\omega_n^2 = \frac{K_V}{\tau_1} \tag{2.77}$$

$$2\xi\omega_n = K_v \frac{\tau_2}{\tau_1} \tag{2.78}$$

$$\xi = \frac{\tau_2}{2}\sqrt{\frac{K_v}{\tau_1}} = \frac{\tau_2}{2}\omega_n \tag{2.79}$$

เพราะฉะนั้นสามารถปรับค่า ω_n และ ξ เป็นอิสระจากกันได้ ทำการตรวจสอบเสถียรภาพโดยใช้วิธีโบดจากฟังก์ชันการถ่ายโอนลูปแบบเปิด (Open Loop Transfer Function)

$$T'(S) = \frac{K_V F(S)}{S} \tag{2.80}$$

$$= \frac{K_V (S\tau_2 + 1)}{S^2 \tau_1} \tag{2.81}$$

แทน $S = j\omega$ จะได้

$$T'(S) = \frac{2\xi \frac{S}{\omega_N} + 1}{\left(\frac{S}{\omega_n}\right)^2} \tag{2.82}$$

เมื่อ $\tau_1 = \frac{K_V}{\omega_n^2}$ และ $\tau_2 = \frac{2\xi}{\omega_n}$

$$\left|T'(j\omega)\right| = \frac{K_v}{\tau_1 \omega^2}\sqrt{1 + \omega^2 \tau_2^2} \tag{2.83}$$

$$\left|T'(\frac{j\omega}{\omega_n})\right| = \left(\frac{\omega_n}{\omega}\right)^2 \sqrt{1 + 4\xi^2 \left(\frac{\omega_n}{\omega}\right)^2} \tag{2.84}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

พิจารณาเฟสจะได้ว่า

$$\phi\left(\frac{j\omega}{\omega_n}\right) = \tan^{-1}\left(2\xi\frac{\omega}{\omega_n}\right) - 180° \tag{2.85}$$

กำหนดให้ $X = \left(\frac{\omega}{\omega_n}\right)$

$$|T'(jX)| = \frac{1}{X^2}\sqrt{(1+4\xi^2X^2)} \tag{2.86}$$

$$\phi(jX) = \tan^{-1}(2\xi X) - 180° \tag{2.87}$$

คำนวณหาค่า X ในขณะที่

$$\left|T'\left(\frac{j\omega_1}{\omega_n}\right)\right| = 1 \tag{2.88}$$

$$1 = \frac{1}{X^2}\sqrt{(1+4\xi^2X^2)} \tag{2.89}$$

ยกกำลังสองแล้วจัดรูปใหม่

$$X^4 - 4\xi^2X^2 - 1 = 0 \tag{2.90}$$

หารากคำตอบที่เป็นบวก

$$X = \frac{\omega_1}{\omega_n} = \sqrt{2\xi^2 + \sqrt{(4\xi^4+1)}} \tag{2.91}$$

แทน X ลงในสมการ

$$\theta(jX) = \tan^{-1}\left(2\xi\sqrt{2\xi^2 + \sqrt{4\xi^4+1}}\right) \tag{2.92}$$

ได้ Phase margin

$$\begin{aligned}\theta_M(jX) &= \theta(jX) + 180° \\ &= \tan^{-1}\left(2\xi\sqrt{2\xi^2 + \sqrt{(4\xi^4+1)}}\right)\end{aligned} \tag{2.93}$$

แบนด์วิดธ์ –3 dB

$$|T(j\omega)| = \frac{2\xi\omega_n S + \omega_n^2}{S^2 + 2\xi\omega_n S + \omega_n^2} \tag{2.94}$$

$$|T(j\omega)| = \frac{\sqrt{(2\xi\omega_n)^2\omega^2 + \omega_n^4}}{\sqrt{(\omega_n^2 - \omega^2)^2 + (2\xi\omega_n)^2\omega^2}} \tag{2.95}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อ $|T_B(j\omega)| = \frac{1}{\sqrt{2}}$

$$|T(j\omega)| = \frac{\sqrt{(2\xi\omega_n)^2\omega_B^2 + \omega_n^4}}{\sqrt{(\omega_n^2 - \omega_B^2) + (2\xi\omega_n)^2\omega_B^2}} \quad (2.96)$$

$$(\omega_n^2 - \omega_B^2)^2 + (2\xi\omega_n)^2\omega_B^2 = 2(2\xi\omega_n)^2\omega_B^2 + 2\omega_n^4 \quad (2.97)$$

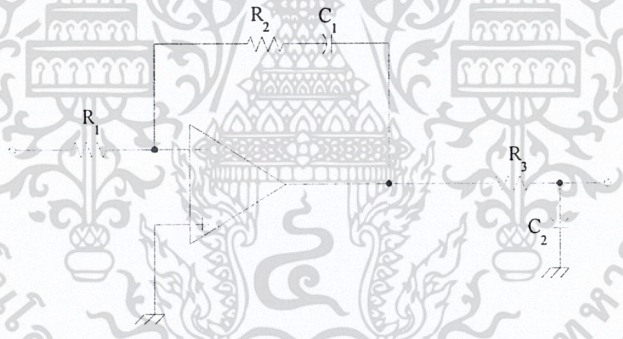
$$\omega_B^4 - (2\omega_n^2)\omega_B^2 + \omega_n^4 + (4\xi^2\omega_n^2)\omega_B^2 = 2(4\xi^2\omega_n^2)\omega_B^2 + 2\omega_n^4 \quad (2.98)$$

$$\omega_B^4 - (\omega_n^2)(2 + 4\xi^2)\omega_n^2 - \omega_n^4 = 0 \quad (2.99)$$

$$\omega_B^2 = \frac{(2 + 4\xi^2)(\omega_n^2) \pm \sqrt{\omega_n^2(2 + 4\xi^2)^2 + 4\omega_n^4}}{2} \quad (2.100)$$

$$\omega_{bandwidth} = \omega_n\sqrt{(1 + 2\xi^2) \pm \sqrt{2 + 4\xi^2 + 4\xi^4}} \quad (2.101)$$

การออกแบบวงจรลูปฟิลเตอร์อันดับที่ 3



รูปที่ 2.31 วงจรลูปฟิลเตอร์อันดับที่ 3

$$F(S) = \frac{S\tau_2 + 1}{S\tau_1}\frac{1}{S\tau_3 + 1} \quad (2.102)$$

เมื่อ $\tau_1 = R_1C_1$, $\tau_2 = R_2C_2$, $\tau_3 = R_3C_3$

จะได้ทรานส์เฟอร์ฟังก์ชันแบบลูปปิด

$$T(S) = \frac{K_V F(S)}{S + K_V F(S)} \quad (2.103)$$

$$= \frac{\frac{K_V(S\tau_2 + 1)}{S\tau_1(S\tau_3 + 1)}}{S + \frac{K_V(S\tau_2 + 1)}{S\tau_1(S\tau_3 + 1)}} \quad (2.104)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$= \frac{K_V(S\tau_2+1)}{S(S\tau_1(S\tau_3+1))+K_V(S\tau_2+1)} \quad (2.105)$$

$$= \frac{K_V(S\tau_2+1)}{S^3\tau_1\tau_3+S^2\tau_1+SK_V\tau_2+K_V} \quad (2.106)$$

$$= K_V\left(\frac{\tau_2}{\tau_2\tau_3}\right)\left(\frac{S+\frac{1}{\tau_2}}{S^3+\frac{1}{\tau_3}S^2+\frac{K_V\tau_2}{\tau_1\tau_3}S+\frac{K_V}{\tau_1\tau_3}}\right) \quad (2.107)$$

ทรานส์เฟอร์ฟังก์ชันแบบลูปเปิดคือ

$$T'(S) = \frac{K_V F(S)}{S} \quad (2.108)$$

$$= \frac{K_V F(S\tau_2+1)}{S\cdot S\tau_1(S\tau_3+1)} \quad (2.109)$$

$$\phi(j\omega) = \tan^{-1}(\tau_2\omega) - \tan^{-1}(\tau_3\omega) - 180° \quad (2.110)$$

การหาเสถียรภาพโดยวิธีโบด จะได้ส่วนเผื่อเฟส (Phase Margin)

$$\phi_M(j\omega) = \phi(j\omega) + 180° \quad (2.111)$$

$$= \tan^{-1}(\tau_2\omega) - \tan^{-1}(\tau_3\omega) \quad (2.112)$$

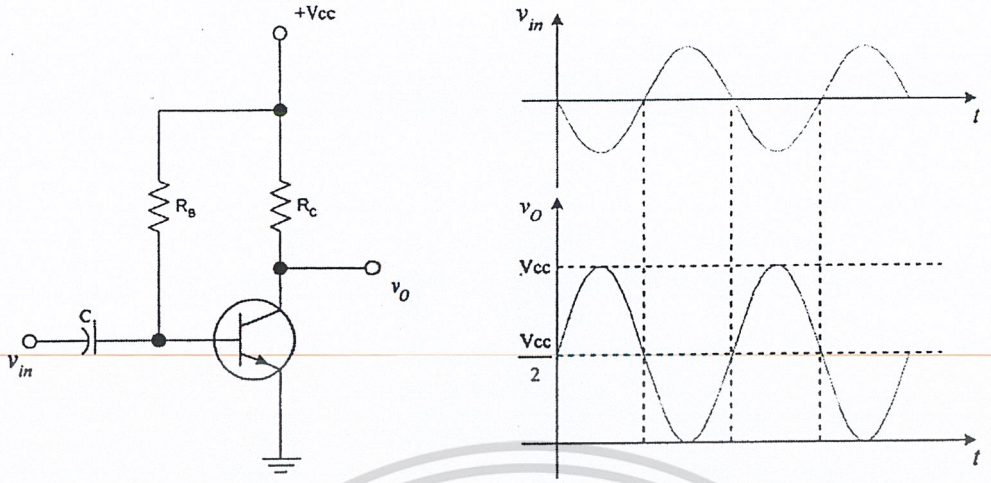
2.8 วงจรขยายกำลัง (Power Amplifier)

วงจรขยายกำลังสามารถจะแบ่งตามชนิดของการไบอัสหรืออาจเรียกอีกอย่างว่าเป็นคลาส (Class) ซึ่งมีด้วยกันหลายคลาส การเลือกชนิดของคลาสจะขึ้นอยู่กับชนิดการใช้งาน วงจรขยายกำลังที่ใช้กันอยู่ทั่วไปคือ วงจรขยายคลาสเอ วงจรขยายคลาสบี และ วงจรขยายคลาสซี

2.8.1 วงจรขยายคลาสเอ

เป็นวงจรขยายที่มีจุดการไบอัสจุดกึ่งกลางของดีซีโหลดลาย (DC Load Line) และจำกัดการสวิงของสัญญาณไว้ให้ไม่เกินจุดคัตออฟ (Cut off) หรือจุดอิ่มตัว (Saturated) ทำให้มีกระแสไหลตลอดเวลา หรืออาจกล่าวได้ว่าวงจรขยายคลาสเอนั้น มีการทำงานตลอด 360 องศาหรือเต็มลูกคลื่น ด้วยเหตุนี้จึงทำให้วงจรขยายสัญญาณคลาสเอนั้นมีการสูญเสียกำลังงานมากกว่าคลาสอื่น และวงจรขยายคลาสเอเป็นวงจรขยายที่มีความเป็นเชิงเส้นมากที่สุดในบรรดาวงจรขยายกำลัง ตามรูปที่ 2.32

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่2.32 Class A Power Amplifier

หาค่าประสิทธิภาพ (efficiency: η) ได้เป็น

$$\frac{P_L}{P_S} = \eta = \frac{\text{ค่าพลังงานเฉลี่ยของสัญญาณที่ปลดปล่อยไปให้โหลด}}{\text{ค่าพลังงานเฉลี่ยที่ดึงมาจากแหล่งจ่ายไปตรง}} \tag{2.113}$$

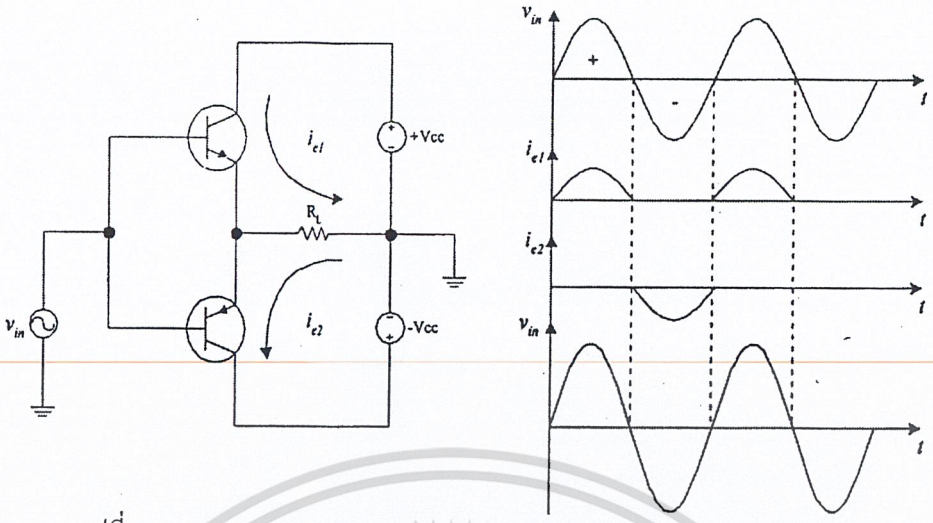
โดยที่ $$P = V_{rms} I_{rms} = \frac{V_p}{\sqrt{2}} . \frac{I_p}{\sqrt{2}} = \frac{V_p I_p}{2} = \frac{I_p^2 R}{2} = \frac{V_p^2}{2} \tag{2.114}$$

โดยทั่วไปค่า η ของ Class A Power Amplifier มีค่าไม่เกิน 50%

2.8.2 วงจรขยายคลาสบี

เป็นวงจรขยายที่ทำงานเพียงแค่ 180 องศาเท่านั้น หรือทำงานเพียงแค่ครึ่งคาบของสัญญาณอินพุตซึ่งอาจจะเป็นด้านบวกหรือด้านลบก็ได้ และการพิจารณาว่าเป็นคลาสบีนั้นก็ขึ้นอยู่กับการไบอัสเป็นตัวกำหนด วงจรขยายคลาสบีนั้นจะมีประสิทธิภาพมากกว่าคลาสเอ (คลาสเอ 50% ส่วนคลาสบี 70%) แต่อย่างไรก็ตามคลาสบีจะมีความเป็นเชิงเส้นน้อยกว่าคลาสเอ เนื่องจากคลาสบีมีการทำงานเพียง 180องศา เมื่อต้องการให้คลาสบีขยายสัญญาณได้เต็ม 360 องศา หรือเต็มไซเคิลของสัญญาณก็ต้องใช้คลาสบี 2 ชุด ดังรูปที่ 2.33

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2 .33 (ก) Dual supply complementary push pull class B power amplifier

(ข) รูปคลื่นแสดงการทำงานของวงจรในรูปที่ 2.33 (ก)

วงจรขยายกำลัง Complementary push pull Class B power amplifier จะใช้ทรานซิสเตอร์ 2 ตัว แบบ NPN และ PNP ต่อกันแสดงดังรูปที่ 2.33 โดยใช้แรงดัน 2 ชุดต่อกัน ชุดหนึ่งมีค่า $+V_{CC}$ และอีกชุดหนึ่งมีค่า $-V_{CC}$

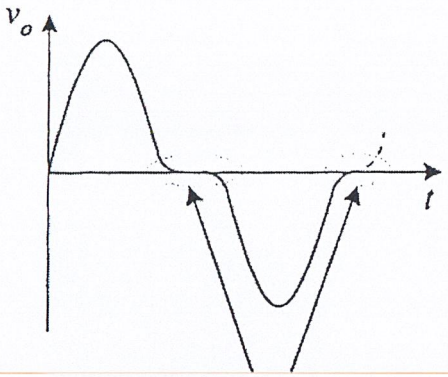
พิจารณาสัญญาณอินพุต V_{in} ด้านซีกบวกจะทำให้ทรานซิสเตอร์ Q1 NPN ทำงานและทรานซิสเตอร์ Q2 PNP ไม่ทำงาน รูปคลื่นการทำงานของวงจรในรูปที่ 2.33 (ก) แสดงในรูปที่2.23 (ข) ทำให้เกิดการไหลของกระแสอีมิตเตอร์-คอลเล็คเตอร์: I_{C1} ดังรูปที่ 2.33 (ก)

เมื่อครึ่งซีกลบของสัญญาณอินพุตเข้ามา จะทำให้ทรานซิสเตอร์ Q2 PNP ทำงานและทรานซิสเตอร์ Q1 NPN ไม่ทำงาน ทำให้เกิดกระแส I_{C2} ไหลตามรูปที่ 2.33 (ก) ซึ่งมีทิศตรงข้ามกับ I_{C1} ผลก็คือ แรงดันเอาท์พุตที่คร่อม V_L จะแสดงในรูปกราฟล่างสุด

จากการทำงานในลักษณะคลาสบี เมื่อสัญญาณอินพุตมีค่าน้อยๆ (โดยประมาณน้อยกว่า 0.6 V) จะทำให้ทรานซิสเตอร์ทั้งสองไม่ทำงาน (Cut off) ดังนั้นสัญญาณเอาท์พุต Vo จะมีค่าเท่ากับศูนย์ในช่วงเวลาหนึ่ง ดังแสดงในรูปที่ 2.34 เรียกปรากฏการณ์นี้ว่า เกิด Cross over distortion จะทำให้เอาท์พุตออกมาเกิดความผิดเพี้ยน

ดังนั้นเพื่อให้ผลของ Cross over distortion นั้นหายไป เราจึงต้องทำการไบอัสขาเบสของทรานซิสเตอร์ จึงทำให้เกิดมีวงจรอีกลักษณะหนึ่งคือวงจร Complementary push pull classAB power amplifier โดยที่วงจรแสดงไว้ในรูปที่ 2.35 โดยมีค่าความต้านทาน R_1 และ R_2 เป็นตัวแบ่งแรงดัน เพื่อให้ทรานซิสเตอร์ Q1 ทำงาน และในทำนองเดียวกัน ตัวต้านทาน R_3 และ R_4 จะเป็นตัวแบ่งแรงดันเพื่อให้ทรานซิสเตอร์ Q2 ทำงาน ดังนั้นปัญหาของ Cross over distortion จะหมดไป

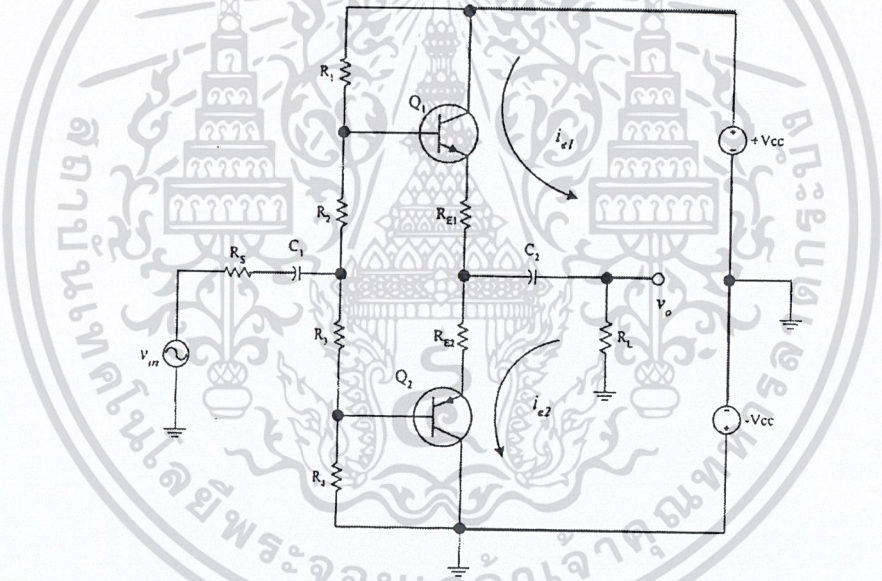
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



Crossover Distortion

รูปที่2.34 สัญญาณเอาท์พุตที่เกิด Cross over distortion

หลักการทำงานของวงจรในรูปที่ 2.35 จะเหมือนกับการอธิบายการทำงานของวงจร Complementary push pull Class B power amplifier ดังที่กล่าวในตอนต้น โดยค่าประสิทธิภาพ (η) ปกติมีค่าระหว่าง 50-75%



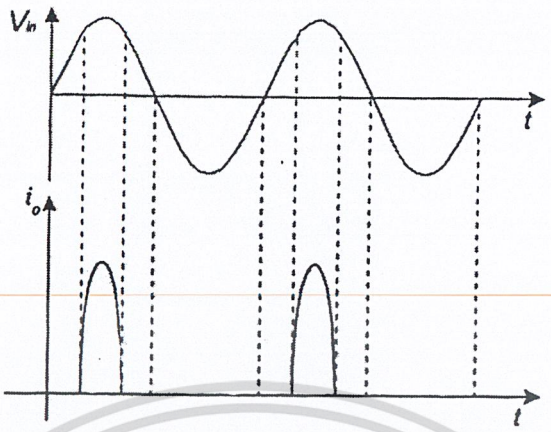
รูปที่2.35 Dual supply complementary class AB power amplifier

2.8.3 วงจรขยายคลาสซี

เป็นวงจรขยายที่มีการทำงานน้อยกว่า 180 องศา หรือน้อยกว่าครึ่งคาบของสัญญาณอินพุต โดยไบอัสไว้ ณ จุดสภาวะปกติ เมื่อไม่มีอินพุตเข้ามาจะไม่มีกระแสคอลเลเตอร์ไหลหรืออยู่ในสภาวะคัตออฟ และวงจรขยายคลาสซีนี้เป็นวงจรขยายที่มีความเป็นเชิงเส้นต่ำที่สุด แต่ก็เป็นคลาสที่มีประสิทธิภาพสูงสุด (80-90%) หากต้องการไบอัสทรานซิสเตอร์ให้เป็นคลาสซีจำเป็นต้องทำการไบอัสย้อนกลับให้แก่รอยต่อของเบส-อิมิตเตอร์ การไบอัสจากภายนอกไม่จำเป็นเนื่องจากขนาดของสัญญาณอินพุตจะเป็นตัวจัดการไบอัสให้แก่วงจรเองอยู่แล้ว ดังนั้น จึงไม่นิยมประยุกต์ใช้งานเป็นตัวขยายเสียง แต่มักใช้กับวงจรความถี่สูง เช่น เครื่องรับวิทยุเอเอ็ม หรือวงจรจูนความถี่ เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

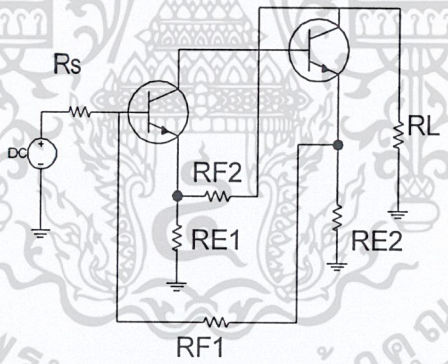
โดยปกติค่าประสิทธิภาพ(η) ของ Class C จะมีค่าสูงมากประมาณ 90%



รูปที่2.36 กระแสเอาท์พุตของ Class C power amplifier

2.8.4 วงจรขยายความถี่สัญญาณช่วงกว้าง (Wideband Amplifier)

วงจรขยายความถี่สัญญาณช่วงกว้าง มีคุณสมบัติคือ มีอัตราขยายคงที่ในช่วงความถี่กว้างๆ อินพุตและเอาท์พุตอิมพีแดนซ์คงที่ และมีสัญญาณรบกวนต่ำ โดยจะตอบสนองความถี่ได้ตั้งแต่ความถี่ต่ำๆ จนถึงสูงระดับกิกะเฮิรตซ์



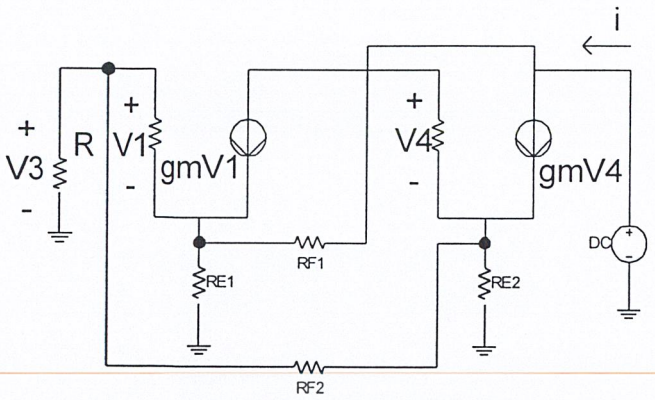
รูปที่ 2.37 วงจรขยายความถี่สัญญาณช่วงกว้างรูปแบบหนึ่ง

มีหลักการการออกแบบเป็นวงจรขยาย 2 วงจรต่อแบบคาสเคด และมีการป้อนกลับ 2 ลูป ผ่านทาง RF1 และ RF2 เพื่อให้วงจรมีแบนด์วิดธ์กว้าง และมีอัตราขยายสูง โดยอินพุตอิมพีแดนซ์และเอาท์พุตอิมพีแดนซ์ เท่ากับ 50 โอห์ม จากทฤษฎีของ Negative Feedback ดังรูป จะมีผลต่ออัตราขยายของวงจรและทำให้อินพุตอิมพีแดนซ์ของวงจรเดิมที่มีค่าสูงให้ต่ำลงเป็น 50 โอห์ม ในทำนองเดียวกัน RF2 ก็จะลดค่าเอาท์พุตอิมพีแดนซ์จากเดิมให้เหลือ 50 โอห์ม ได้เช่นกัน ซึ่งความสัมพันธ์ระหว่างอัตราขยาย และแบนด์วิดธ์จะมีค่าแปรผกผันกัน โดยจะมีค่าอัตราขยายเป็นดังสมการ

$$A_v = (RF1 + RF2)/RE1 \tag{2.115}$$

ในการคำนวณจะทำการแปลงเป็นวงจรสมมูลย์ดังรูปที่ 2.38

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.38 วงจรสมมูลย์ของวงจรขยายความถี่สัญญาณช่วงกว้างในการหาค่าเอาท์พุตอิมพีแดนซ์

สามารถหาค่าเอาท์พุตอิมพีแดนซ์ R_0 โดยใส่โหลด R ซึ่งมีค่า Output Resistance 50 โอห์ม ที่ด้านอินพุตแล้วหาโวลต์เตจและกระแสที่ตกคร่อมที่เอาท์พุตได้ดังนี้

$$V_2 = \frac{R_{E1}}{R_{E1}+R_{F1}}V \tag{2.116}$$

$$V_3 = g_{m2}V_4\frac{R_{E2}(R+R_{F2})}{R_{E2}+R+R_{F2}}\times\frac{R}{R+R_{F2}} \tag{2.117}$$

$$= g_{m2}V_4\frac{R_{E2}R}{R_{E2}+R_{F2}+R} \tag{2.118}$$

ถ้า V_1 น้อยมากจะได้

$$V_2 = V_3 \tag{2.119}$$

$$\frac{R_{E1}}{R_{E1}+R_{F1}}V = g_{m2}V_4\frac{R_{E2}R}{R_{E2}+R_{F2}+R} \tag{2.120}$$

$$g_{m2}V_4 = \frac{R_{E1}}{R_{E1}+R_{F1}}\times\frac{R_{E2}+R+R_{F2}}{R_{E2}R}\times V \tag{2.121}$$

$$\frac{1}{R_0} = \frac{i}{V} \tag{2.122}$$

$$\frac{1}{R_0} = \frac{1}{V}\left(\frac{V}{R_{E1}+R_{F1}} + g_{m2}V_4\right) \tag{2.123}$$

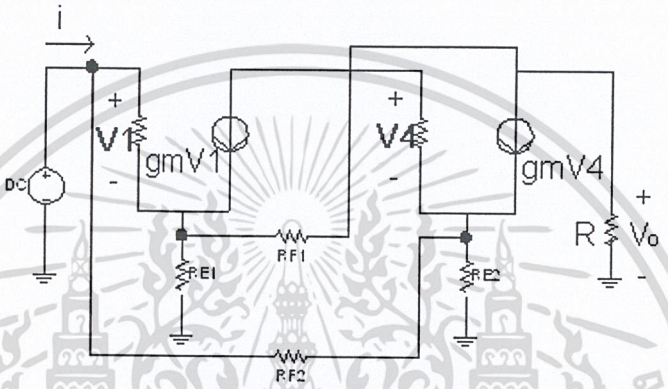
$$\frac{1}{R_0} = \frac{1}{R_{E1}+R_{F1}} + \frac{R_{E2}+R+R_{F2}}{R_{E2}R}\times\frac{R_{E1}}{R_{E1}+R_{F1}} \tag{2.124}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$R_0 = \frac{R_{E2}R(R_{E1}+R_{F1})}{R_{E2}R+(R_{E2}+R+R_{F2})R_{E1}} \tag{2.125}$$

$$R_0 = \frac{(R_{E1}+R_{F1})R_{E2}R}{R_{E1}(R_{E2}+R+R_{F2})} \;||\; (R_{E1}+R_{F1}) \tag{2.126}$$

ส่วนอินพุตอิมพีแดนซ์ และอัตราการขยายสามารถหาได้โดยทำการใส่โหลด 50 โอห์ม ที่ด้านเอาท์พุตแล้วหาโวลต์เตจและกระแสที่ตกคร่อมอินพุต



รูปที่ 2.39 วงจรสมมูลย์ของวงจรขยายความถี่สัญญาณช่วงกว้างในการหาค่าอินพุตอิมพีแดนซ์

$$V_2 = g_{m2}V_4 \times \frac{(R_{E1}+R_{F1})R}{R_{E1}+R+R_{F1}} \times \frac{R_{E1}}{(R_{E1}+R_{F1})} \tag{2.127}$$

$$V_2 = V \tag{2.128}$$

$$i = g_{m2}V_4 \frac{R_{E2}}{(R_{E2}+R_{F2})} \tag{2.129}$$

$$i = V \frac{R_{E1}+R+R_{F1}}{(R_{E1}+R_{F1})R} \times \frac{R_{E1}}{(R_{E1}+R_{F1})} \times \frac{R_{E2}}{(R_{E2}+R_{F2})} \tag{2.130}$$

$$R_i = \frac{V}{i} \tag{2.131}$$

$$R_i = \frac{RR_{E1}}{R_{E1}+R+R_{F1}} \times \frac{R_{E2}+R_{F2}}{R_{E2}} \tag{2.132}$$

และมีกระแสอีกส่วนหนึ่ง ดังนั้น

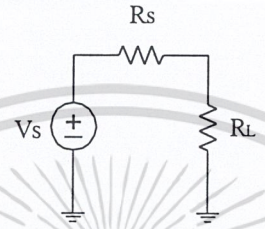
$$R_i = \frac{RR_{E1}}{R_{E1}+R+R_{F1}} \times \frac{R_{E2}+R_{F2}}{R_{E2}} \;||\; R_{E2}+R_{F2} \tag{2.133}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.9 แมตช์ชิ่งอิมพีแดนซ์ (Matching Impedance)

การแมตชิ่งอิมพีแดนซ์นั้นมีความจำเป็นอย่างมากในการออกแบบวงจร เพื่อให้สามารถส่งผ่านกำลังงาน (Power) จากแหล่งกำเนิด (Source) ไปยังโหลด (Load) ได้มากที่สุด (Maximum power transfer)

ในระบบที่เป็นวงจรไฟฟ้ากระแสตรง (DC circuit) จะเห็นว่ากำลังงานจะส่งจากแหล่งกำเนิดไปยังโหลดมากที่สุดเมื่อค่าความต้านทานภายในของแหล่งกำเนิด (Source Resistance) มีค่าเท่ากับความต้านทานของโหลด (Load Resistance)



รูปที่ 2.40 การแมตช์ชิ่งอิมพีแดนซ์ในวงจรไฟฟ้ากระแสตรง

$$I = \frac{V_S}{R_S + R_L} \tag{2.134}$$

$$P_L = I^2 R_L = \frac{V_S{}^2 R_L}{(R_S + R_L)^2} \tag{2.135}$$

$$\frac{dP_L}{dR_L} = \frac{(R_S + R_L)^2 V_S{}^2 - V_S{}^2 R_L - 2(R_S + R_L)}{(R_S + R_L)^4} \tag{2.136}$$

ค่า P_L มีค่ามากที่สุด (Maximum) เมื่อ $\frac{dP_L}{dR_L} = 0$

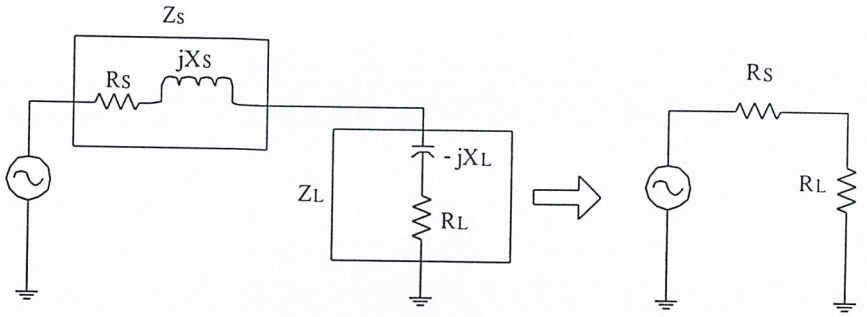
$$(R_S + R_L)^2 = 2(R_S R_L + R_L{}^2) \tag{2.137}$$

$$R_S{}^2 + 2R_S R_L + R_L{}^2 = R_L{}^2 + 2R_S R_L + R_L{}^2 \tag{2.138}$$

$$R_S = R_L \tag{2.139}$$

ในทำนองเดียวกันระบบที่เป็นวงจรไฟฟ้ากระแสสลับ (AC Circuit) จะมีการส่งผ่านกำลังงานจากแหล่งกำเนิดไปยังโหลดมากที่สุด เมื่อ ความต้านทานเชิงซ้อนของโหลด (Load Impedance : Z_L) มีค่าเท่ากับคอมเพล็กซ์คอนจูเกต (Complex conjugate) ของความต้านทานเชิงซ้อนของแหล่งกำเนิด (Source Impedance : Z_S)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

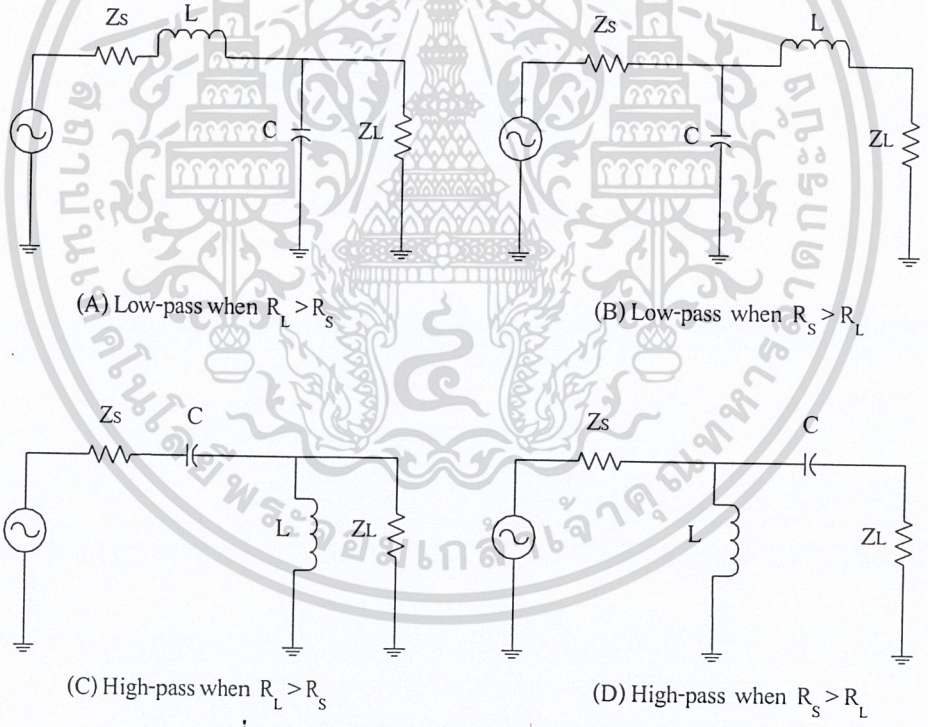


รูปที่ 2.41 การแมตช์ชิ่งอิมพีแดนซ์ในวงจรไฟฟ้ากระแสสลับ

จะเห็นว่าค่าของ X_S และ X_L นั้นขึ้นอยู่กับค่าความถี่ ดังนั้นจะเกิดการแมตช์ชิ่งอิมพีแดนซ์อย่างสมบูรณ์ได้ที่ความถี่เดียวเท่านั้น ซึ่งจะเป็นปัญหาหากเราต้องการแบนวิดธ์ที่กว้าง

2.9.1 โครงข่ายแบบรีเวอร์สแอล (Reverse L type network)

วงจรแมตช์ชิ่งอิมพีแดนซ์ที่ง่ายและใช้กันมากคือใช้โครงข่ายแบบรีเวอร์สแอล ซึ่งแบ่งได้เป็น 4 รูปแบบดังรูป



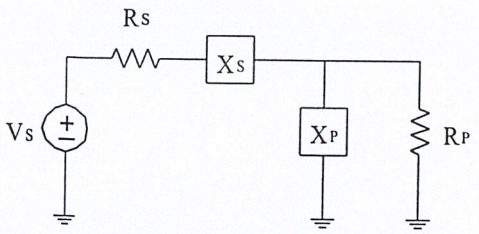
รูปที่ 2.42 รูปแบบการต่อโครงข่ายแบบรีเวอร์สแอล

เราจะใช้ Shunt Component ต่อกับด้านที่มีค่าความต้านทานมาก เพื่อทำให้เกิดความต้านทานเชิงซ้อนค่าหนึ่ง โดยมีส่วนจริงเท่ากับค่าความต้านทานของอีกด้านหนึ่งซึ่งมีค่าน้อยกว่า และใช้ Series Component ในการหักล้างส่วนจินตภาพของความต้านทานเชิงซ้อนดังกล่าวให้หมดไป

ดังนั้นในรูปที่ 2.42 หาก $R_S > R_L$ ก็จะต่อวงจรหรือโครงข่ายแบบรีเวอร์สแอลได้ตามรูปที่ 2.42 (B) หรือ 2.42 (D) และในทำนองเดียวกันหาก $R_L > R_S$ ก็จะสามารถต่อวงจรได้ตามรูปที่ 2.42 (A) หรือ 2.42 (C)

เอกสารนี้ใช้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การออกแบบโครงข่ายแบบรีเวอร์สแอล



รูปที่ 2.43 การออกแบบโครงข่ายแบบรีเวอร์สแอลในกรณีที่ $R_P > R_S$

$$Q_S = Q_P = \sqrt{\frac{R_P}{R_S} - 1} \tag{2.140}$$

$$Q_S = \frac{X_S}{R_S} \tag{2.141}$$

$$Q_P = \frac{R_P}{X_P} \tag{2.142}$$

โดย Q_S = The Q of the series leg

Q_P = The Q of the shunt leg

R_S = The series resistance

R_P = The shunt resistance

X_S = The series reactance

X_P = The shunt reactance

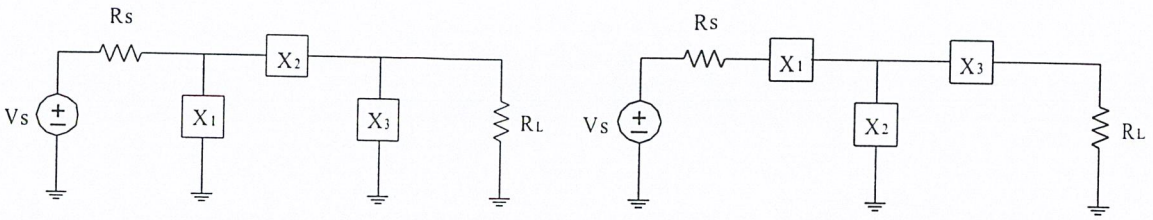
โดยค่าของ X_S และ X_P เป็นไปได้ทั้ง Capacitance และ Inductance แต่จะต้องไม่เหมือนกัน

ข้อเสียของโครงข่ายแบบรีเวอร์สแอล คือ ไม่สามารถปรับค่า Q ได้ตามต้องการเพราะค่า Q ขึ้นกับ R_S และ R_P ในการแมตช์ซึ่งอิมพีแดนซ์โดยใช้โครงข่ายแบบรีเวอร์สแอลของโหลดเชิงซ้อน (Complex Load) ในทางปฏิบัติจริง Z_S และ Z_L จะเป็นค่าเชิงซ้อนมากกว่ามากกว่าค่าที่มีแต่ส่วนจริง เนื่องจากในภาคอินพุต และเอาท์พุต ของแต่ละ stage ที่นำมาต่อกันนั้นจะมีส่วนของรีแอคแตนซ์อยู่ด้วย ดังนั้นในการออกแบบจะใช้วิธีเดียวกันกับ การออกแบบที่กล่าวมาแล้ว และจะใช้วิธีการหักล้างค่ารีแอคแตนซ์เข้าช่วย

2.9.2 โครงข่ายแบบสามองค์ประกอบ (Three element Network)

จากโครงข่ายแบบรีเวอร์สแอล จะเห็นว่าค่า Q ถูกกำหนดโดยค่า R_S และ R_L ซึ่งผู้ออกแบบไม่สามารถออกแบบค่า Q ตามต้องการได้ ซึ่งถ้าค่า Q มีค่ามากจะได้แบนด์วิดธ์แคบ และถ้าค่า Q มีค่าน้อยจะได้แบนด์วิดธ์กว้าง โดยในการออกแบบโครงข่ายแบบสามองค์ประกอบจะสามารถเลือกค่า Q ได้แบ่งออกได้เป็น โครงข่ายแบบพาย (Pi network) และโครงข่ายแบบที (T network)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

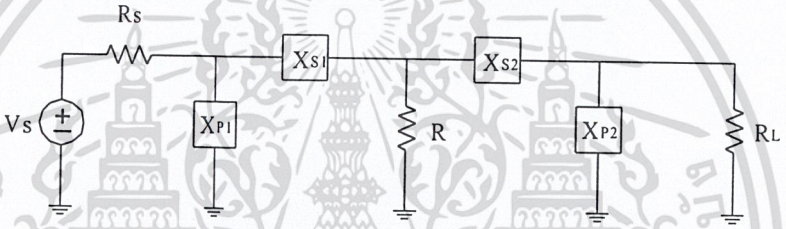


(A) โครงข่ายแบบพาย (B) โครงข่ายแบบที

รูปที่ 2.44 โครงข่ายแบบพายและโครงข่ายแบบที

โครงข่ายแบบพาย (Pi network)

โครงข่ายแบบพายจะพิจารณาเป็น Back to Back Reverse L network ซึ่ง Reverse L network ทั้งสองนี้จะทำหน้าที่แมตช์ชิ่ง (Matching) ค่าความต้านทานของโหลดและค่าความต้านทานของแหล่งกำเนิดไปยังค่า Virtual Resistance (R) ซึ่งอยู่ระหว่าง L network ทั้งสอง



รูปที่ 2.45 การแมตช์ชิ่งโดยใช้โครงข่ายแบบพาย

การออกแบบแต่ละส่วนจะทำเหมือนกับโครงข่ายแบบรีเวอร์สแอล โดยค่า Virtual Resistance (R) จะต้องมีค่าน้อยกว่าทั้ง R_S และ R_L เพราะว่า Virtual Resistance ต่ออยู่กับแขนอนุกรม (Series Arm) ของโครงข่ายแบบรีเวอร์สแอล

โดยที่ค่า Q ของโครงข่ายแบบพาย มีค่าดังสมการ

$$Q_t = \sqrt{\frac{R_{HIGHER}}{R} - 1} \qquad (2.143)$$

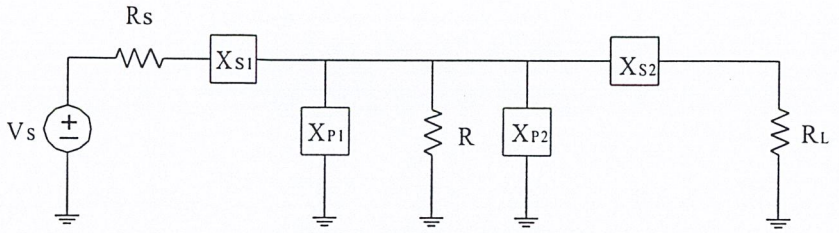
$R_{H\,IGHER}$ = ค่าที่มากที่สุดระหว่าง R_S และ R_L

R = Virtual Resistance

โครงข่ายแบบที (T network)

การออกแบบจะเหมือนกับโครงข่ายแบบพาย โดย Virtual Resistance จะต้องมากกว่าทั้ง R_S และ R_L เพราะว่า Virtual Resistance ต่ออยู่กับแขนขนาน (Shunt Arm) ของโครงข่ายแบบรีเวอร์สแอล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.46 การแมตช์ชิ่งโดยใช้โครงข่ายแบบที

โครงข่ายแบบทีจะเหมาะสมกับการแมตช์ชิ่งค่า R_S และ R_L ที่มีค่าต่ำ และต้องการค่า Q สูงโดยที่ค่า Q ของโครงข่ายแบบที มีค่าดังสมการ

$$Q_t = \sqrt{\frac{R_{SMALLER}}{R} - 1} \tag{2.144}$$

$R_{SMALLER}$ = ค่าน้อยที่สุดระหว่าง R_S และ R_L

R = Virtual Resistance

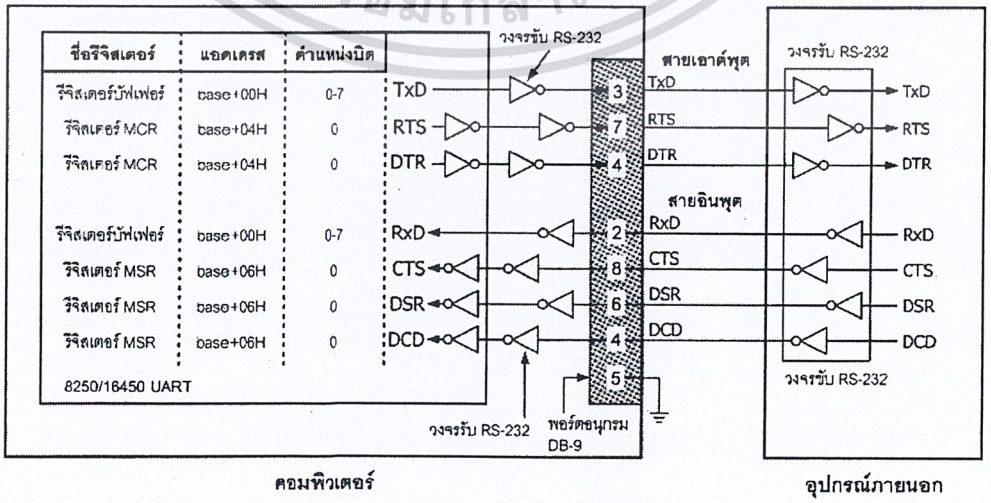
จะเห็นว่าสำหรับการแมตช์ชิ่งโดยใช้โครงข่ายแบบสามองค์ประกอบที่กล่าวมาเป็นเฉพาะความต้านทานของแหล่งกำเนิด และความต้านทานของโหลด แต่ถ้าเป็นความต้านทานเชิงซ้อน ก็จะใช้วิธีหักล้างค่ารีแอคแตนซ์ เหมือนกับที่ได้กล่าวมาแล้วในโครงข่ายแบบรีเวอร์สแอลในกรณีที่โหลดเป็นความต้านทานเชิงซ้อน

2.10 แอดเดรสของพอร์ตอนุกรม

แอดเดรสพื้นฐานของพอร์ตอนุกรมมี 4 ตำแหน่งดังนี้ คือ

COM1 : 3F8H COM2 : 2F8H

COM3 : 3E8H COM4 : 2E8H



รูปที่ 2.47 ไดอะแกรมแสดงโครงสร้างทางฮาร์ดแวร์ของพอร์ตอนุกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อเริ่มเปิดเครื่องเพื่อใช้งานคอมพิวเตอร์ ไบออสภายในคอมพิวเตอร์จะทำการตรวจสอบแอดเดรสของพอร์ตอนุกรมทั้งหมด ถ้าไบออสตรวจสอบพบแอดเดรสของพอร์ตอนุกรม ไบออสจะนำแอดเดรสที่ตรวจพบไปเก็บไว้ในหน่วยความจำขนาด 2 ไบต์ สำหรับพอร์ตอนุกรม COM1 จะเก็บไว้ที่แอดเดรส 0000:0400H และ 0000:0401H ส่วนตำแหน่งอื่น ๆ มีรายละเอียดดังนี้

COM1 = 0000:0402H – 0000:0403H

COM2 = 0000:0404H – 0000:0405H

COM3 = 0000:0406H – 0000:0407H

นอกจากนี้ที่หน่วยความจำแอดเดรส 0000:0411H ยังใช้สำหรับแสดงจำนวนของพอร์ตอนุกรมที่มีใช้อยู่ในคอมพิวเตอร์อีกด้วย โดยมีรายละเอียดดังแสดงในตารางที่ 2.3

บิต 3	บิต 2	บิต 1	จำนวนพอร์ต
0	0	0	ไม่มีพอร์ตอนุกรม
0	0	1	มีพอร์ตอนุกรม 1 พอร์ต
0	1	0	มีพอร์ตอนุกรม 2 พอร์ต
0	1	1	มีพอร์ตอนุกรม 3 พอร์ต
1	0	0	มีพอร์ตอนุกรม 4 พอร์ต

ตารางที่ 2.3 ข้อมูลในแอดเดรส 0000:0411H ที่ใช้แจ้งจำนวนพอร์ตอนุกรม

ในการรับและส่งข้อมูลแบบอนุกรมผ่านพอร์ตอนุกรม RS232 เช่น ใช้คำสั่งพิมพ์ออกทางเครื่องพิมพ์ เรียกอินเตอร์รัปต์ของไบออสหรือดอส การเขียนหรืออ่านไปยังแอดเดรสพอร์ตโดยตรง วิธีสุดท้ายเป็นวิธีที่มีความยืดหยุ่นในการใช้งานที่สุด ยกตัวอย่าง ถ้าต้องการส่งข้อมูลไปยังพอร์ตอนุกรม COM1 โดยใช้คำสั่งภาษา QBASIC ง่าย ๆ ดังนี้

```
OUT &H3F8,X
```

ค่า X ในที่นี้ หมายถึง ข้อมูลที่ต้องการส่ง มีขนาด 8 บิต

สำหรับการอ่านข้อมูลจากพอร์ตอนุกรม จะเป็นการอ่านข้อมูลมาจากรีจิสเตอร์บัฟเฟอร์สำหรับรับข้อมูล (แอดเดรส 3F8H เช่นเดียวกัน) ซึ่งสามารถเขียนโปรแกรมง่าย ๆ ได้ดังนี้

```
Y = INP (&H3F8)
```

ค่า Y ในที่นี้ คือค่าที่อ่านได้จากรีจิสเตอร์บัฟเฟอร์สำหรับรับข้อมูล โดยมีขนาด 8 บิต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แต่เมื่อใช้คำสั่งนี้ในขณะที่โปรแกรมทำงานผ่านระบบปฏิบัติการวินโดวส์ จะไม่สามารถใช้งานได้เนื่องจากระบบปฏิบัติการวินโดวส์ ได้เข้าฝังตัวพอร์ตอนุกรมเข้าเป็นส่วนหนึ่งของระบบปฏิบัติการแล้ว ดังนั้นการเรียกใช้งานจึงจำเป็นต้องเรียกผ่านเครื่องมือที่ติดต่อผ่านระบบปฏิบัติการ เช่น การใช้คอนโทรล MSCOMM32.OCX ของโปรแกรม Visual BASIC

2.11 คอนโทรล MSComm

สำหรับการใช้งาน Visual BASIC ตั้งแต่เวอร์ชัน 2 เป็นต้นมา ใน Visual BASIC จะมีคัสตอมคอนโทรลสำหรับการสื่อสารอนุกรมผ่านทางพอร์ตอนุกรมของคอมพิวเตอร์มาให้ โดยใน Visual BASIC เวอร์ชัน 2 และเวอร์ชัน 3 จะใช้ชื่อว่า MSCOMM.VBX ส่วนเวอร์ชัน 4 ใช้ชื่อว่า MSCOMM16.OCX สำหรับการทำงานกับระบบปฏิบัติการ 16 บิต และ MSCOMM32.OCX สำหรับการทำงานกับระบบปฏิบัติการ 32 บิต สำหรับใน Visual BASIC เวอร์ชัน 5 จะมีเพียง MSCOMM32.OCX เท่านั้น เพราะถูกออกแบบมาให้ใช้งานกับระบบปฏิบัติการ 32 บิต

MSComm จัดเตรียมทางเลือกเอาไว้ 2 ทางเพื่อความสะดวกในการสื่อสารข้อมูล ทางแรก คือ การสื่อสารข้อมูลที่กระตุ้นด้วยเหตุการณ์ (Event-driver communications) เป็นรูปแบบการใช้งานที่มีประสิทธิภาพมากสำหรับการตอบสนองแบบทันทีทันใด เช่น เมื่อตัวอักษรถูกส่งมาที่พอร์ตอนุกรมหรือเกิดการเปลี่ยนแปลงที่ขา Data Carrier Detect (DCD) หรือขา Request To Send (RTS) เหตุการณ์ Oncomm ของ MSComm จะสามารถตรวจจับสัญญาณนั้นได้ทันที ซึ่งจะกล่าวถึงรายละเอียดในหัวข้อคุณสมบัติ CommEvent ต่อไป ส่วนทางเลือกที่สองเป็นการคอยตรวจสอบค่าเหตุการณ์และความผิดพลาดที่เกิดขึ้นด้วยการดูค่าที่เปลี่ยนแปลงในคุณสมบัติ CommEvent หลังจากให้โปรแกรมทำงานในฟังก์ชันต่าง ๆ ไปเรียบร้อยแล้ว ซึ่งวิธีนี้ใช้งานได้ดีในกรณีที่โปรแกรมมีขนาดเล็ก

คอนโทรล MSComm 1 ตัว สามารถควบคุมการทำงานของพอร์ตอนุกรมได้ 1 พอร์ต ถ้าในโปรแกรมที่ใช้งานต้องการติดต่อกับพอร์ตอนุกรมมากกว่า 1 พอร์ต จะต้องใช้คอนโทรล MSComm มากกว่า 1 ตัวเพื่อควบคุมพอร์ตอนุกรมในแต่ละพอร์ต แอดเดรสของพอร์ตอนุกรมและแอดเดรสของการเกิดอินเตอร์รัปต์สามารถเปลี่ยนแปลงได้จากการแก้ไขค่าที่ Control Panel

ถึงแม้ว่า คอนโทรล MSComm จะมีคุณสมบัติมากมาย แต่สามารถทำความเข้าใจได้ไม่ยากดังนี้

2.11.1 CommPort

ใช้ในการกำหนดและอ่านค่าพอร์ตอนุกรมที่ติดต่ออยู่ (COM1, COM2, COM3, COM4)

รูปแบบการใช้งาน : object.CommPort[=value]

โดย Value เป็นค่าของพอร์ตอนุกรม ชนิดของข้อมูลเป็น Integer ค่า Value สามารถกำหนดได้ในช่วง 1-16 (ค่าเริ่มต้นกำหนดไว้ที่ 1) เมื่อมีการกำหนดค่าแล้วทำการเปิดพอร์ตโดยใช้คุณสมบัติ PortOpen แต่ว่าพอร์ตนั้นไม่มีอยู่ในระบบ MSComm จะสร้างสัญญาณแสดงข้อผิดพลาด error 68

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขึ้นมา ซึ่งหมายถึง อุปกรณ์ตัวนี้ไม่มีอยู่ในระบบ ดังนั้นการเขียนโปรแกรมจึงจำเป็นต้องกำหนดตำแหน่งของพอร์ตอนุกรมก่อนที่ใช้คำสั่ง OpenPort

2.11.2 Setting

ใช้ในการกำหนดและอ่านค่าอัตราบอด, พาริตี้, จำนวนของบิตข้อมูล และจำนวนของบิตสิ้นสุด

รูปแบบการใช้งาน : `object.Settings[=value]`

ค่า Value มีชนิดข้อมูลเป็นแบบสตริง มีรูปแบบเป็น "BBBB,P,D,S" โดย BBBB เป็นอัตราบอด, P เป็นค่าพาริตี้, D เป็นจำนวนของบิตข้อมูล และ S เป็นจำนวนของบิตปิดท้าย ปกติแล้วค่านี้ถูกกำหนดไว้เป็น "9600, N, 8, 1"

สำหรับค่ามาตรฐานในการกำหนดค่าพาริตี้มีดังนี้

สัญลักษณ์	รายละเอียด
E	พาริตี้คู่ (Even)
M	ลอจิก "1" (MARK)
N	ไม่ใช้ (ค่าปกติ)
O	พาริตี้คี่ (Odd)
S	ลอจิก "0" (Space)

ค่าที่ใช้ในการกำหนดจำนวนบิตมี 4 ค่า คือ 5,6,7 และ 8

ค่าที่ระบุจำนวนบิตปิดท้ายมี 3 ค่า คือ 1, 1.5 และ 2

ตัวอย่างการใช้งานคำสั่ง Settings โดยจะเป็นการกำหนดค่าบอดเรตเท่ากับ 9600 ไม่มีพาริตี้ จำนวนบิตข้อมูล 8 บิต และบิตปิดท้าย 1 บิต สามารถเขียนโปรแกรมได้ดังนี้

```
MSComm1.Settings = "9600,N,8,1"
```

หมายเหตุ สาเหตุที่ค่าที่กำหนดจะต้องอยู่ภายในเครื่องหมายคำพูด "" เนื่องจาก ค่าที่กำหนดนี้อยู่ในรูปตัวแปรสตริง

2.11.3 PortOpen

ใช้ในการกำหนดและอ่านค่าสถานะของพอร์ตอนุกรม เพื่อเปิดและปิดพอร์ตอนุกรม

รูปแบบการใช้งาน : `object.PortOpen[=value]`

ค่า Value มีชนิดข้อมูลเป็นแบบบูลีน คือ True กับ False โดย True หมายถึง การเปิดพอร์ตอนุกรมและ False หมายถึง การปิดพอร์ตอนุกรม สำหรับการปิดพอร์ตนั้นจะมีการเคลียร์บัฟเฟอร์รับข้อมูลและบัฟเฟอร์ส่งข้อมูลด้วย คอนโทรล MSComm จะปิดพอร์ตอนุกรมโดยอัตโนมัติเมื่อออกจากโปรแกรม ก่อนที่จะใช้คุณสมบัติ PortOpen ต้องตรวจสอบให้แน่ใจก่อนว่าคุณสมบัติ CommPort นั้นได้ทำการกำหนดตำแหน่งของพอร์ตอนุกรมไว้ถูกต้องหรือไม่ มิเช่นนั้น MSComm จะแสดงข้อผิดพลาด Error 98 แจ้งแก่ผู้ใช้งาน หรือถ้าพอร์ตอนุกรมนั้นถูกเปิดเอาไว้แล้ว โปรแกรมก็จะแจ้งข้อผิดพลาดออกมาเช่นเดียวกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถ้าคุณสมบัติ DTREnable หรือ RTSEnable ถูกกำหนดให้เป็น True ก่อนที่จะทำการเปิดพอร์ต ค่าคุณสมบัติของ DTREnable หรือ RTSEnable จะถูกเซตเป็น False หลังจากปิดพอร์ต แต่ถ้าเซตเป็น False หลังจากเปิดโปรแกรมแล้ว ค่าที่กำหนดไว้จะเป็นค่าเดิม

ตัวอย่างการใช้คำสั่งเปิดพอร์ต เพื่อติดต่อสื่อสารกับพอร์ตอนุกรม COM1 และมีบอดเรต 9,600 บิตต่อวินาที ไม่มีพาริตี้ จำนวนบิตข้อมูล 8 บิต และบิตปิดท้าย 1 บิต มีดังนี้

```
MSComm1.Settings = "9600, n, 8, 1"
MSComm1.CommPort = 1
MSComm1.PortOpen = True
```

2.11.4 Input

ใช้ในการอ่านค่าและลบค่าขบวนข้อมูลจากบัฟเฟอร์ภาครับ

รูปแบบการใช้งาน : object.Input

คุณสมบัติ InputLen เป็นตัวกำหนดจำนวนของตัวอักษรที่จะอ่านโดยคุณสมบัติ Input การกำหนดค่าให้ InputLen เท่ากับ 0 เป็นการกำหนดให้คุณสมบัติ Input ทำการอ่านค่าข้อมูลในบัฟเฟอร์รับข้อมูลทั้งหมด

คุณสมบัติ InputMode เป็นตัวกำหนดชนิดของข้อมูลที่คุณสมบัติ Input รับเข้ามา ถ้า InputMode ถูกกำหนดเป็น comInputMode Text คุณสมบัติ Input จะส่งค่าข้อมูลกลับมาในรูปแบบของข้อความชนิดข้อมูลเป็นแบบ Variant ถ้า InputMode กำหนดเป็น comInputModeBinary คุณสมบัติ Input จะส่งข้อมูลกลับมาในรูปของไบนารีและชนิดข้อมูลเป็นแบบ Variant

2.11.5 InBufferCount

ใช้ในการส่งค่าจำนวนของตัวอักษรที่อยู่ในบัฟเฟอร์ภาครับ

รูปแบบการใช้งานคำสั่ง : object.InBufferSize[=value]

คำสั่ง InBufferCount จะแสดงค่าจำนวนของตัวอักษร ซึ่งรับมาจากภายนอกและยังเก็บอยู่ในบัฟเฟอร์ภาครับ เพื่อให้ผู้ใช้งานอ่านค่าออกไป สำหรับการเคลียร์ค่าบัฟเฟอร์ภาครับทำได้โดยกำหนดให้ InBufferCount มีค่าเป็น 0

2.11.6 InBufferSize

ใช้ในการกำหนดและคืนค่าขนาดของบัฟเฟอร์ภาครับในหน่วยเป็นไบต์

รูปแบบการใช้งานคำสั่ง : object.InBufferSize[=value]

คำสั่ง InBufferSize ใช้เพื่อกำหนดขนาดของบัฟเฟอร์ภาครับโดยค่าเริ่มต้นถูกกำหนดไว้ที่ 1,024 ไบต์

หมายเหตุ การกำหนดค่าบัฟเฟอร์ภาครับขนาดใหญ่จะทำให้หน่วยความจำที่เหลือสำหรับการใช้งานส่วนอื่น ๆ จะเหลือน้อย อย่างไรก็ตามการกำหนดค่าบัฟเฟอร์ภาครับที่น้อยเกินไปจะทำให้เกิดการโอเวอร์โฟลว์หรือข้อมูลล้นบัฟเฟอร์เว้นแต่จะมีการใช้แฮนด์เช็ก ดังนั้นค่าปานกลางที่เหมาะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สมก็คือค่า 1,024 ซึ่งเป็นค่าเริ่มต้นนั่นเอง แต่ถ้าโปรแกรมมีการเกิดโอเวอร์โฟลว์แล้วจึงค่อยปรับเพิ่มค่าขนาดของบัฟเฟอร์ให้มีค่ามากขึ้น

2.11.7 InputLen

ใช้ในการกำหนดค่าและคืนจำนวนของตัวอักษรที่อ่านจากบัฟเฟอร์ภาครับ

รูปแบบการใช้งานคำสั่ง : object.InputLen[=value]

ค่าเริ่มต้นของคุณสมบัติ InputLen มีค่าเท่ากับ "0" การกำหนดค่าเท่ากับ "0" จะทำให้คำสั่ง Input ของ MSComm อ่านค่าข้อมูลที่อยู่ภายในบัฟเฟอร์ภาครับทั้งหมด

ถ้าไม่มีข้อมูลอยู่ในบัฟเฟอร์ภาครับมากเท่ากับจำนวน InputLen คำสั่ง Input จะส่งค่าว่าง ("") กลับออกมา ผู้ใช้งานสามารถตรวจสอบข้อมูลในบัฟเฟอร์ภาครับได้โดยใช้คุณสมบัติ InBufferCount โดยกำหนดให้มีข้อมูลอยู่ในบัฟเฟอร์ภาครับได้โดยใช้คุณสมบัติ InBufferCount โดยกำหนดให้มีข้อมูลอยู่ในบัฟเฟอร์ภาครับก่อนแล้วจึงค่อยอ่านข้อมูลจากบัฟเฟอร์ภาครับ

คุณสมบัตินี้มักใช้กับการอ่านค่าข้อมูลจากเครื่องมือหรือเครื่องจักรที่มีการกำหนดค่าขนาดความยาวของข้อมูลเอาไว้แล้ว

2.11.8 InputMode

ใช้ในการกำหนดค่าและคืนค่าชนิดของข้อมูลที่รับโดยคำสั่ง Input

รูปแบบการใช้งานคำสั่ง : object.InputMode[=value]

คุณสมบัติ InputMode ใช้กำหนดว่าข้อมูลชนิดไหนที่รับเข้ามาผ่านคำสั่ง Input โดยข้อมูลจะเลือกได้ 2 ประเภท คือ

- comInputModeText สำหรับข้อมูลที่อยู่ในรูปข้อความตัวอักษรตามมาตรฐาน ANSI โดยจำต้องกำหนดค่าเป็น "0" และค่าเริ่มต้นของการรับค่าข้อมูลก็จะเป็นค่านี้
- comInputModeBinary สำหรับข้อมูลอื่น ซึ่งจะเก็บในรูปไบนารีรวมกันอยู่เป็นไบต์ข้อมูล

2.11.9 Output

ใช้ในการส่งขบวนของข้อมูลไปยังบัฟเฟอร์ส่งข้อมูล

รูปแบบการใช้งาน : object.Output[=value]

ค่า Value เป็นค่าของตัวอักษรที่เขียนไปยังบัฟเฟอร์ส่งข้อมูล คุณสมบัติ Output สามารถใช้ในการส่งข้อมูลตัวอักษรหรือข้อมูลไบนารีก็ได้ โดยการส่งข้อมูลเป็นรูปแบบตัวอักษรจะต้องกำหนดข้อมูลเป็นแบบ Variant และมีข้อมูลภายในเป็นแบบสตริง สำหรับการส่งข้อมูลไบนารีจะต้องกำหนดชนิดของข้อมูลเป็นแบบ Variant และมีข้อมูลภายในเป็นแบบ Byte

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.11.10 OutBufferCount

ใช้ในการคืนค่าจำนวนของข้อมูลตัวอักษรที่เก็บอยู่ในบัฟเฟอร์ภาคส่ง และสามารถใช้คำสั่งนี้เพื่อเคลียร์บัฟเฟอร์ภาคส่งได้ด้วย

รูปแบบการใช้งานคำสั่ง : object.OutBufferCount[=Value]

ผู้ใช้งานสามารถเคลียร์บัฟเฟอร์ภาคส่งได้โดยการกำหนดค่า OutBufferCount เป็น "0"

2.11.11 OutBufferSize

ใช้ในการกำหนดค่าและคืนค่าขนาดของบัฟเฟอร์ภาคส่ง ชนิดตัวแปรเป็นแบบไบต์

รูปแบบการใช้งานคำสั่ง : Object.OutBufferSize[=object]

คุณสมบัติ OutBufferSize ใช้สำหรับกำหนดขนาดของบัฟเฟอร์ภาคส่ง โดยค่าปกติที่ใช้งานจะมีค่าเท่ากับ 512 ไบต์

หมายเหตุ การกำหนดค่าบัฟเฟอร์ภาคส่งที่มากเกินไปจะทำให้ มีหน่วยความจำเหลือให้ใช้งานน้อย แต่อย่างไรก็ตามถ้ากำหนดค่าน้อยเกินไป จะทำให้เกิดข้อมูลล้นบัฟเฟอร์ขึ้นได้ ยกเว้นจะมีการใช้ แฮนด์เช็ค วิธีการที่ถูกต้องในการกำหนดค่าคือ ทดลองใช้ค่าเริ่มต้นคือค่า 512 ไบต์ดูก่อน ถ้าโปรแกรมทำงานแล้วเกิดการล้นของข้อมูลค่อยเพิ่มค่าของ OutBufferSize ให้มากขึ้น

2.11.12 ParityReplace

ใช้ในการกำหนดและคืนค่าตัวอักษรที่ไปวางในตำแหน่งที่เกิดข้อผิดพลาดจากพาริตี้

รูปแบบการใช้งานคำสั่ง : object.ParityRePlace[=Value]

บิตพาริตี้ เป็นบิตที่ทางภาคส่งข้อมูลทำการส่งมาพร้อมกับข้อมูล เพื่อตรวจสอบข้อผิดพลาดของข้อมูล โดยเมื่อมีการใช้พาริตี้ คอนโทรล MSComm จะทำการบวกบิตทุกบิตที่มีค่าลอจิก "1" ในแต่ละไบต์ และทำการตรวจสอบผลลัพธ์ว่าบิตที่อ่านได้นั้นมีจำนวนลอจิก "1" เป็นเลขคู่หรือคี่ และตรงกับค่าที่กำหนดไว้แต่ต้นหรือไม่ ถ้าค่าที่นำมาบวกแล้วมีพาริตี้ไม่ตรงแสดงว่าการรับส่งข้อมูลผิดพลาด

การกำหนดค่า เริ่มต้นให้กับ ParityReplace นั้นกำหนดให้ใช้เครื่องหมาย (?) ไปวางไว้ที่ตำแหน่งที่เกิดพาริตี้ผิดพลาด ถ้ากำหนดค่า ParityReplace ให้เป็นค่าว่าง ("") จะเป็นการยกเลิกการใช้งาน ParityReplace และไม่มีการป้อนข้อมูลแทนเมื่อตรวจพบข้อผิดพลาด

ParityReplace ใช้ชนิดข้อมูลเป็นแบบสตริง แต่จะการกำหนด จะกำหนดได้เพียงไบต์เดียวเท่านั้น ซึ่งสามารถใช้ค่าใด ๆ ก็ได้ที่เป็นโค้ด ANSI มีค่าระหว่าง 0-255

2.11.13 DTREnable

ใช้ในการกำหนดสถานะลอจิกของขา Data Terminal Ready (DTR) โดยสัญญาณของเขา DTR จะส่งจากคอมพิวเตอร์ไปยังโมเด็มเพื่อแสดงว่าคอมพิวเตอร์พร้อมที่จะรับข้อมูลแล้ว ชนิดของข้อมูลเป็นแบบบูลีน

รูปแบบการใช้งาน : object.DTREnable[=value]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ค่า Value เป็นค่าสถานะ True หรือ False เพื่อกำหนดลอจิกของขา DTR ให้เป็น “0” หรือ “1” โดย True หมายถึง ให้ขา DTR มีลอจิก “1” และ False หมายถึงให้ขา DTR มีลอจิก “0”

หมายเหตุ เมื่อขา DTR ถูกกำหนดสถานะให้เป็น True ที่ขา DTR จะมีสถานะลอจิก “1” เมื่อทำการเปิดพอร์ตและจะมีสถานะเป็น “0” เมื่อมีการปิดพอร์ต เมื่อขา DTR ถูกกำหนดสถานะเป็น False ที่ขา DTR จะมีสถานะลอจิก “0” ตลอดเวลาไม่ว่าจะใช้คำสั่งเปิดพอร์ตหรือปิดพอร์ต

สำหรับการใช้งานกับโมเด็ม การทำให้ขา DTR เป็นลอจิก “0” จะเป็นการวางหูโทรศัพท์หรือยกเลิกการติดต่อ

2.11.14 RTSEnable

ใช้เพื่อกำหนดสถานะลอจิกให้ขา Request To Send (RTS) โดยขา RTS จะเป็นสัญญาณที่ส่งจากคอมพิวเตอร์ไปยังโมเด็มเพื่อร้องขอส่งข้อมูล ชนิดของข้อมูลเป็นแบบบูลีน

รูปแบบการใช้งาน : object.RTSEnable[=value]

ค่า Value เป็นค่าสถานะ True หรือ False เพื่อกำหนดลอจิก “0” หรือ “1” ให้ขา RTS โดย True หมายถึง ให้ขา RTS มีลอจิก “1” และ False หมายถึง ให้ขา RTS มีลอจิก “0”

หมายเหตุ เมื่อขา RTSEnable ถูกกำหนดให้เป็น True ขา RTS จะมีสถานะลอจิก “1” เมื่อเปิดพอร์ตและมีสถานะลอจิก “0” เมื่อปิดพอร์ต

2.11.15 EOFEnable

เป็นการกำหนดให้ MSComm รอสัญลักษณ์แสดงส่วนท้ายสุดของไฟล์ (End of file : EOF) ระหว่างการรับอินพุตเข้ามา ถ้าพบสัญลักษณ์ EOF ภาคอินพุตจะหยุดรับข้อมูล และเหตุการณ์ OnComm จะถูกกระตุ้นให้ทำงาน คุณสมบัติ CommEvent จะมีค่าเท่ากับ 7 หรือ ComEvEOF

รูปแบบการใช้งาน : object.EOFEnable[=value]

โดย value เป็นค่าสถานะ True หรือ False เพื่ออีนาเบิลหรือดิสเอเบิลการทำงานของเหตุการณ์ OnComm เมื่อตรวจพบสัญลักษณ์ EOF โดย

- True หมายถึง เหตุการณ์ OnComm จะถูกกระตุ้นให้ทำงานด้วย EOF
- False หมายถึง เหตุการณ์ OnComm จำไม่ถูกกระตุ้นให้ทำงานด้วย EOF

เมื่อ EOFEnable กำหนดให้เป็น False ส่วนควาบคุมจะไม่มีการตรวจสอบ

2.11.16 CTSHolding

ผู้ใช้งานสามารถตรวจสอบการทำงานของเขา Clear To Send (CTS) ได้ว่ามีสถานะลอจิก “0” หรือ “1”โดยค่าที่อ่านได้จะเป็นบูลีน True และ False ถ้าค่า CTSHolding เป็น True ขา CTS จะมีสถานะลอจิกเป็น “1” ถ้าค่า CTSHolding เป็น Fales ขา CTS จะมีสถานะลอจิกเป็น “0”

รูปแบบการใช้งาน : object.CTSHolding

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อขา CTS เป็นลอจิก “0” (CTSHolding = False) และเกิดไทม์เอาต์ คอนโทรล MSComm จะกำหนดให้คุณสมบัติ CommEvent มีค่าเป็น comEventCTSTO (Clear To Send Timeout) และกระตุ้นให้เกิดเหตุการณ์ OnComm

2.11.17 CDHolding

ผู้ใช้งานสามารถตรวจสอบการทำงานของขา Data Carrier Detect (DCD) ได้ว่ามีสถานะลอจิกเป็น “1” หรือ “0” โดยค่าที่อ่านได้จะเป็นบูลีน True และ False ถ้าค่า CDHolding เป็น True ขา DCD จะมีสถานะลอจิก “1” ถ้าค่า CDHolding เป็น False ขา DCD จะมีสถานะลอจิก “0”

รูปแบบการใช้งาน : object.CDHolding

เมื่อขา DCD มีลอจิก “1” (CDHolding = True) และเกิดไทม์เอาต์ คอนโทรล MSComm จะกำหนดให้คุณสมบัติ CommEvent มีค่าเป็น comEvent CTSTO (Clear To Send Timeout) และกระตุ้นให้เกิดเหตุการณ์ OnComm

2.11.18 DSRHolding

ผู้ใช้งานสามารถตรวจสอบการทำงานของขา Data Set Ready (DSR) ได้ว่ามีสถานะลอจิก “1” หรือ “0” โดยค่าที่อ่านได้จะเป็นบูลีน True และ False ค่า DSRHolding เป็น True ขา DSR จะมีสถานะลอจิก “1” ถ้าค่า DSRHolding เป็น False ขา DSR จะมีสถานะลอจิก “0”

รูปแบบการใช้งาน : object. DSRHolding

เมื่อขา DSR เป็นลอจิก “1” (DSRHolding = True) และเกิดไทม์เอาต์ คอนโทรล MSComm จะกำหนดให้คุณสมบัติ CommEvent มีค่าเป็น comEverntDSRTO (Data Set Ret Ready Timeout) และกระตุ้นให้เกิดเหตุการณ์ OnComm

2.11.19 Handshaking

ใช้ในการกำหนดคุณสมบัติและคืนค่ารูปแบบแฮนด์เช็กทางฮาร์ดแวร์

รูปแบบการใช้งานคำสั่ง : object.Handshaking[=value]

ค่าตัวแปร Value ที่ใช้กำหนดค่ากำหนดได้ 4 รูปแบบด้วยกันคือ

1. comNone ค่าที่กำหนดคือ 0 เป็นการกำหนดให้ไม่มีการแฮนด์เช็ก (เป็นค่าเริ่มต้น)
2. comXOnXOff ค่าที่กำหนดคือ 1 เป็นการกำหนดให้ใช้แฮนด์เช็กแบบ XON/XOFF
3. comRTS ค่าที่กำหนดคือ 2 เป็นการกำหนดให้ใช้ขา RTS/CTS (Request To Send/Clear To Send)
4. comRTSXOnXOff ค่าที่กำหนดคือ 3 เป็นการกำหนดให้ใช้ทั้งแบบ Request To Send และ XON/XOFF

คุณสมบัติ Handshaking ใช้เพื่อกำหนดรูปแบบการสื่อสารภายใน ระหว่างที่ข้อมูลถูกส่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ในเชิงพาณิชย์ ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไปยังบัฟเฟอร์ภาครับ เมื่อข้อมูลตัวอักษรถูกส่งมาถึงพอร์ตอนุกรม อุปกรณ์สื่อสารข้อมูลจะทำการย้ายข้อมูลไปยังบัฟเฟอร์ภาครับ เพื่อที่จะให้โปรแกรมสามารถอ่านค่าไปใช้งานได้ ถ้าไม่มีบัฟเฟอร์ภาครับ โปรแกรมที่ใช้งานจะต้องทำการอ่านค่าข้อมูลโดยตรงจากฮาร์ดแวร์ของพอร์ตอนุกรม ซึ่งผู้ใช้งานจะเกิดปัญหาข้อมูลสูญหายได้ เนื่องจากว่าการเปลี่ยนแปลงของข้อมูลที่ส่งเข้ามามีการเปลี่ยนแปลงอย่างรวดเร็ว

คุณสมบัติ Handshaking ช่วยให้ผู้ใช้งานแน่ใจได้ว่าข้อมูลที่ได้รับมานั้นไม่มีการสูญหาย เมื่อบัฟเฟอร์ภาครับที่รับข้อมูลนั้นเกิดข้อมูลล้นหรือโอเวอร์โฟลว์ (overflow) โดยใช้วิธีการตรวจสอบความพร้อมของบัฟเฟอร์ว่าพร้อมรับข้อมูลหรือไม่ก่อนที่จะส่งข้อมูลมาให้

2.11.20 Break

ใช้ในการเซตและเคลียร์ค่าสัญญาณ Break ชนิดของข้อมูลเป็นแบบบูลีน

รูปแบบการใช้งาน : object.Break[=value]

โดย Value เป็นค่าบูลีน ถ้า Value = True หมายถึง การส่งสัญญาณ Break ออกไป Value = False หมายถึงการเคลียร์สัญญาณ Break

เมื่อกำหนดให้สัญญาณ Break เป็น True จะเป็นการหยุดการส่งข้อมูลชั่วคราวจนกว่าการสั่งให้สัญญาณ Break เป็น False

2.11.21 เหตุการณ์ OnComm

เหตุการณ์ OnComm จะถูกสร้างขึ้นเมื่อค่าของคุณสมบัติ CommEvent มีการเปลี่ยนแปลงเพื่อแสดงผลการเปลี่ยนแปลงเหล่านั้นแบบทันทีทันใดหรือแสดงข้อผิดพลาดที่เกิดขึ้น

2.11.22 ค่าคงที่คุณสมบัติของคอนโทรล MSComm

ค่าคงที่	ค่า	รายละเอียด
ComNone	0	ไม่ใช้การตรวจสอบแฮนด์เช็ก
ComXonXoff	1	ใช้การตรวจสอบแฮนด์เช็กแบบ XOn/Xoff
ComRTS	2	ใช้การตรวจสอบแฮนด์เช็กผ่านทางขา RTS และ CTS
ComRTSXOnXOff	3	กำหนดการตรวจสอบแฮนด์เช็กทั้งแบบ RTS, CTS และ XOn/XOff

ตารางที่ 2.4 ค่าคงที่สำหรับคุณสมบัติ Handshake

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ค่าคงที่	ค่า	รายละเอียด
ComEvSend	1	ส่งค่าเหตุการณ์ (send event)
ComEvReceive	2	รับค่าเหตุการณ์ (receive event)
ComEvCTS	3	มีการเปลี่ยนแปลงที่ขา CTS
ComEvDSR	4	มีการเปลี่ยนแปลงที่ขา DSR
ComEvCD	5	มีการเปลี่ยนแปลงที่ขา DCD
ComEvRing	6	ตรวจสอบสัญญาณกระดิ่งของโทรศัพท์
ComEvEOF	7	ตรวจพบตำแหน่งท้ายสุดของไฟล์ (End of file)

ตารางที่ 2.5 ค่าคงที่สำหรับคุณสมบัติ OnComm

ค่าคงที่	ค่า	รายละเอียด
ComEventBreak	1001	ได้รับสัญญาณ Break
ComEventCTSTO	1002	ขา CTS เกิดไทม์เอาต์
ComEventDSRTO	1003	ขา DSR เกิดไทม์เอาต์
ComEventFrame	1004	เกิดข้อผิดพลาดที่เฟรมข้อมูล (Framing error)
ComEventOverrun	1006	พอร์ตอนุกรมเกิดโอเวอร์รัน (Port overrun)
ComEventCDTO	1007	ขา DCD เกิดไทม์เอาต์
ComEventRxOver	1008	บัฟเฟอร์รับข้อมูลเกิดโอเวอร์โฟลว
ComEventRxParity	1009	เกิดข้อผิดพลาดที่พาริตี้ (Parity error)
ComEventTxFull	1010	บัฟเฟอร์ส่งข้อมูลเต็ม

ตารางที่ 2.6 ค่าคงที่สำหรับคุณสมบัติ Error

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ค่าคงที่	ค่า	รายละเอียด
comInputModeText	0	ข้อมูลที่รับมีคุณสมบัติเป็นข้อมูล (ค่าปกติ)
comInputModeBinary	1	ข้อมูลที่รับเข้ามาเป็นข้อมูลไบนารี

ตารางที่ 2.7 ค่าคงที่สำหรับคุณสมบัติ InputMode

2.11.23 การใช้ MSComm เพื่อการติดต่อฮาร์ดแวร์

จากรายละเอียดของ MSComm ที่กล่าวไปในตอนต้น จะเห็นได้ว่าวิธีการที่จะอ่านค่าหรือเขียนค่าไปยังขาสถานะและขาควบคุมของพอร์ตอนุกรมสามารถทำได้โดยใช้คำสั่งเหล่านี้

DTREnable สำหรับสั่งให้ขา DTR มีลอจิก “0” หรือ “1”

RTSEnable สำหรับสั่งให้ขา RTS มีลอจิก “0” หรือ “1”

CTSHolding สำหรับอ่านค่าสถานะจากขา CTS ว่ามีลอจิก “0” หรือ “1”

CDHolding สำหรับอ่านค่าสถานะจากขา DCD ว่ามีลอจิก “0” หรือ “1”

DSRHolding สำหรับอ่านค่าสถานะจากขา DSR ว่ามีลอจิก “0” หรือ “1”

Break สำหรับการสั่งให้ขา Txd ว่ามีลอจิก “0” หรือ “1”

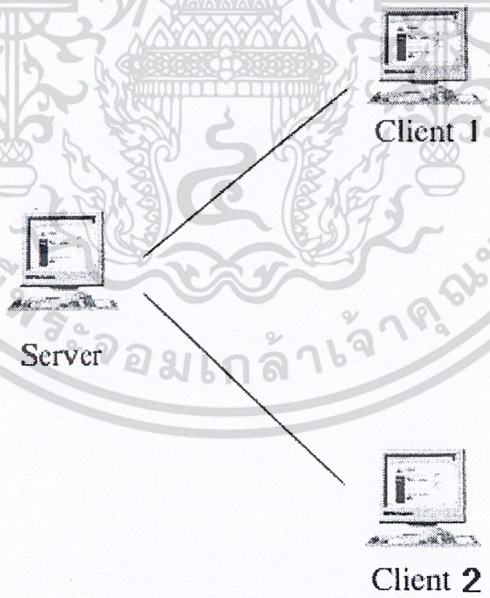
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
การคำนวณและการสร้าง

3.1 การออกแบบระบบโครงข่าย

โครงงานชิ้นนี้เป็นการจำลองระบบเครือข่ายคอมพิวเตอร์ ซึ่งได้ทำการออกแบบให้มีเครื่องคอมพิวเตอร์ในระบบจำนวนทั้งหมด 64 เครื่อง โดยกำหนดให้มีเครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็นเครื่อง Server จำนวน 1 เครื่อง และมีเครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็นเครื่องลูกข่ายจำนวน 63 เครื่อง

ระบบที่ได้เริ่มทำการศึกษาจะมีเครื่อง Server จำนวน 1 เครื่อง และเครื่องลูกข่ายจำนวน 3 เครื่อง ในระบบที่ได้ทำการออกแบบจะมีลักษณะโครงสร้างการเชื่อมต่อข้อมูลระหว่างเครื่องคอมพิวเตอร์ในระบบจะเป็นแบบโครงข่ายรูปดาว โดยเครื่องคอมพิวเตอร์จะมีการเชื่อมต่อระหว่างกันในแต่ละเครื่องจะเป็นแบบ Point-to-Point โดยมีเครื่อง Server ทำหน้าที่เป็น Primary Station ซึ่งจะมีสิทธิในการรับ-ส่งข้อมูลสูงกว่าเครื่องคอมพิวเตอร์ลูกข่าย กล่าวคือ เครื่อง Server จะสามารถที่จะทำการร้องขอเพื่อที่จะรับข้อมูลหรือส่งผ่านข้อมูลได้ โดยเครื่อง Server จะทำการควบคุมการเชื่อมต่อกันของเครื่องคอมพิวเตอร์ลูกข่ายทั้ง 3 เครื่อง ที่ทำหน้าที่เป็น Secondary Station ซึ่งจะทำการรับคำสั่งจาก Primary Station และตอบสนองเพียงอย่างเดียว อีกทั้งยังทำการควบคุมการเชื่อมต่อกันระหว่างเครื่องลูกข่ายกับตัวเครื่อง Server เองด้วย



รูปที่ 3.1 การเชื่อมต่อกันระหว่างเครื่องคอมพิวเตอร์ในระบบ

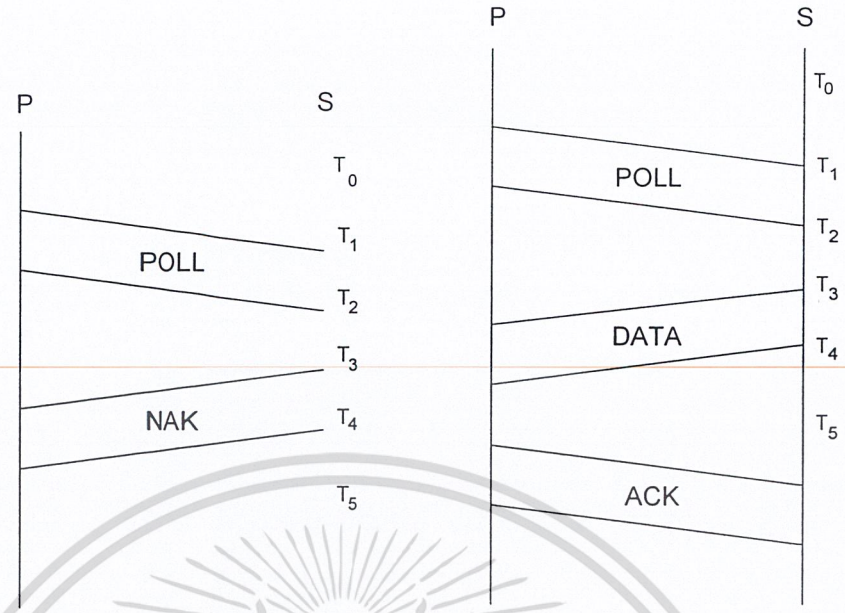
ในการเชื่อมต่อกันระหว่างเครื่องคอมพิวเตอร์นี้ จะทำการเชื่อมต่อผ่านทางพอร์ตอนุกรมของเครื่องคอมพิวเตอร์แต่ละเครื่อง โดยเครื่อง Server จะทำการเชื่อมต่อกับเครื่องคอมพิวเตอร์ลูกข่ายทั้งหมดผ่านทางพอร์ตอนุกรมเพียงพอร์ตเดียว ในการเชื่อมต่อกันโดยใช้พอร์ตอนุกรมเพียงพอร์ตเดียวอาจจะเกิดปัญหาในการชนกันของข้อมูล (Collision) ในกรณีที่เครื่องคอมพิวเตอร์แต่ละเครื่องมีความต้องการที่จะส่งข้อมูลในเวลาเดียวกัน ดังนั้นในระบบนี้จึงได้ทำการออกแบบให้มีโปรโตคอลในการควบคุมการรับ-ส่งข้อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มูลเพื่อป้องกันการชนกันของข้อมูล โดยใช้วิธีการในการติดต่อกันระหว่างเครื่องคอมพิวเตอร์แบบ Poll / Select

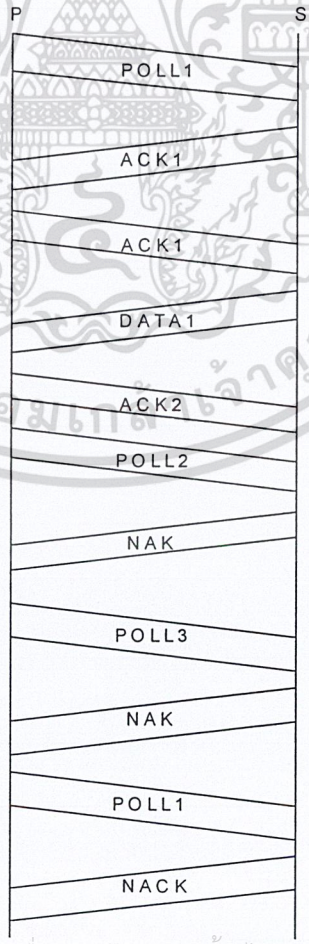
ลักษณะการทำงานของระบบโดยใช้การ Poll / Select นั้น จะเริ่มจากเครื่อง Server ทำการส่งสัญญาณ Poll ไปยังเครื่องลูกข่าย เพื่อเป็นการสอบถามเครื่องลูกข่ายแต่ละเครื่องว่าต้องการที่จะส่งข้อมูลไปยังเครื่องคอมพิวเตอร์เครื่องอื่นหรือไม่ โดยในการ Poll จะทำการ Poll เครื่องลูกข่ายทีละเครื่องสลับกันไป (เครื่องลูกข่าย1, เครื่องลูกข่าย2, เครื่องลูกข่าย3, เครื่องลูกข่าย1, เครื่องลูกข่าย2, เครื่องลูกข่าย3,...) โดยจะทำการแบ่งเวลาสำหรับในการทำการ Poll เครื่องลูกข่ายแต่ละเครื่อง ถ้าหากเครื่องลูกข่ายไม่ต้องการส่งข้อมูลก็จะส่งสัญญาณ NAK ไปยังเครื่อง Server ซึ่งจะทำการตีความหมายว่าเครื่องลูกข่ายไม่มีความต้องการที่จะติดต่อเครื่องคอมพิวเตอร์เครื่องอื่นๆ จากนั้นเครื่อง Server จะทำการเปลี่ยนไปทำการ Poll เครื่องลูกข่ายเครื่องถัดไป แต่ถ้าเครื่องลูกข่ายที่ได้รับสัญญาณ Poll จากเครื่อง Server ต้องการที่จะส่งข้อมูลไปยังเครื่องคอมพิวเตอร์ลูกข่ายเครื่องอื่นๆภายในระบบ ซึ่งเครื่องลูกข่ายเครื่องนั้นจะทำการส่งข้อมูลพร้อมทั้งหมายเลขเครื่องของเครื่องคอมพิวเตอร์ (IDRR) ที่ต้องการจะทำการส่งข้อมูลส่งไปยังเครื่อง Server โดยข้อมูลทั้งหมด(Data+IDRR) ที่เครื่องลูกข่ายได้ทำการส่งไปยังเครื่อง Server จะถูกแบ่งออกให้ข้อมูลมีขนาดในการส่งครั้งละ 1,000 ไบต์ เมื่อเครื่อง Server รับข้อมูลจำนวน 1,000 ไบต์แล้ว จะทำการส่งสัญญาณ ACK1 ไปยังเครื่องลูกข่าย และจะทำการ Poll ไปยังเครื่องลูกข่ายเครื่องอื่นๆและวนกลับมารับข้อมูลจากเครื่องลูกข่ายเครื่องเดิม และจะทำในลักษณะเช่นเดียวกันนี้ไปเรื่อยๆจนข้อมูลจากเครื่องลูกข่ายเครื่องนั้นหมด โดยจะทำการตรวจสอบเครื่องหมาย ##### เพื่อเป็นสัญลักษณ์ในการจบไฟล์ของข้อมูลที่ส่ง เมื่อเครื่อง Server รับข้อมูลมาจนครบไฟล์แล้ว จะทำการส่งสัญญาณACK2 ไปยังเครื่องลูกข่ายที่ส่งข้อมูลมา และจะทำการส่งสัญญาณ Select เพื่อเป็นการบอกว่าต้องการที่จะทำการส่งข้อมูลให้เครื่องลูกข่ายเตรียมตัวที่จะรับข้อมูล ถ้าหากว่าเครื่องลูกข่ายเครื่องนั้นพร้อมที่จะทำการรับข้อมูลก็จะทำการส่งสัญญาณ ACK ตอบกลับไปยังเครื่อง Server หลังจากนั้นเครื่อง Server จะทำการส่งข้อมูลที่เก็บเอาไว้ไปยังเครื่องลูกข่าย IDRR (เครื่องลูกข่ายเครื่องที่เครื่องลูกข่ายเครื่องที่กล่าวมาข้างต้นต้องการส่งข้อมูลไป) ครั้งละ 1,000 ไบต์ จนข้อมูลที่อยู่ในไฟล์ที่ส่งหมด โดยส่งในลักษณะเดียวกันกับการส่งข้อมูลจากเครื่องลูกข่ายไปยังเครื่อง Server ที่ได้กล่าวมาแล้วข้างต้น และเครื่อง Server ก็จะทำการ Poll ต่อไปยังเครื่องลูกข่ายอื่นเรื่อยๆจนกว่าจะมีเครื่องคอมพิวเตอร์ต้องการที่จะทำการส่งข้อมูลอีก ก็จะทำตามกระบวนการตามที่กล่าวมาข้างต้นอีก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(A) Poll แล้วไม่มีการส่งผ่านข้อมูล (B) Poll แล้วมีการส่งผ่านข้อมูล

รูปที่ 3.2 ลำดับการทำการ Poll จากเครื่อง Server



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.3 ลำดับการส่งข้อมูลจากเครื่องลูกข่าย 1 ไปยังเครื่องลูกข่าย 3 โดยผ่านเครื่อง Server

โดยข้อมูลที่จะทำการส่งในโครงงานชิ้นนี้ จะทำการศึกษาและพัฒนาจากข้อมูลที่เป็นรูปแบบของไฟล์ข้อมูล โดยเริ่มจากไฟล์ข้อมูลที่มีนามสกุล TXT ก่อน หลังจากนั้นจึงค่อยทำการพัฒนาเป็นไฟล์ข้อมูลนามสกุล DOC และนามสกุลอื่นๆต่อไป

สัญญาณที่ได้ออกแบบไว้สำหรับการสื่อสารในการส่งผ่านข้อมูลในระบบของโครงงานชิ้นนี้ ได้ทำการออกแบบให้สัญญาณมีลักษณะเป็น Frame ของ Block ข้อมูลต่างๆ โดยข้อมูลที่อยู่ในแต่ละ Block จะประกอบไปด้วยข้อมูลที่สำคัญในการสื่อสารระหว่างเครื่องคอมพิวเตอร์ โดยสัญญาณที่ใช้จะแบ่งออกเป็น 2 ประเภทด้วยกัน คือ สัญญาณคำสั่งและสัญญาณข้อมูล

1. *สัญญาณคำสั่ง* สัญญาณคำสั่งของระบบในโครงงานชิ้นนี้ ได้แก่ สัญญาณ Poll, Select, ACK และ NAK โดยลักษณะ Frame ของสัญญาณคำสั่ง จะเป็นดังรูป

START	IDR	IDT	FUNCTION	STOP
4 bits	6 bits	6 bits	4 bits	4 bits

รูปที่ 3.4 ลักษณะ Frame ของสัญญาณคำสั่ง

Frame ของสัญญาณคำสั่งมีทั้งหมด 24 bits ซึ่งประกอบไปด้วย Block ต่างๆ ดังนี้

- Block START
  - เป็นตัวบอกว่าเป็นส่วนแรกของ Frame
  - มีข้อมูล 1111
  - มีขนาด 4 bits
- Block IDR
  - เป็นตัวบอกหมายเลขของเครื่องคอมพิวเตอร์ของเครื่องรับ
  - ข้อมูลจะขึ้นอยู่กับหมายเลขของเครื่องคอมพิวเตอร์ของเครื่องรับ โดยหมายเลขของเครื่องคอมพิวเตอร์ภายในระบบจะแบ่งออกเป็น
    1. เครื่อง Server มีหมายเลข 000000
    2. เครื่องลูกข่าย 1 มีหมายเลข 000001
    3. เครื่องลูกข่าย 2 มีหมายเลข 000010

    ต่อไปเรื่อยๆ จนถึงเครื่องลูกข่ายเครื่องที่ 63 มีหมายเลข 111111
  - มีขนาด 6 bit (คอมพิวเตอร์ภายในระบบของโครงงานนี้มี 3 เครื่อง)
- Block IDT
  - เป็นตัวบอกหมายเลขของเครื่องคอมพิวเตอร์ของเครื่องส่ง
  - ข้อมูลจะขึ้นอยู่กับหมายเลขของเครื่องคอมพิวเตอร์ของเครื่องส่ง โดยหมายเลขของเครื่องคอมพิวเตอร์ภายจะแบ่งออกเป็น
    1. เครื่อง Server มีหมายเลข 000000

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. เครื่องลูกข่าย 1 มีหมายเลข 000001

3. เครื่องลูกข่าย 2 มีหมายเลข 000010

ต่อไปเรื่อยๆ จนถึงเครื่องลูกข่ายเครื่องที่ 63 มีหมายเลข 111111

– มีขนาด 6 bit (คอมพิวเตอร์ภายในระบบของโครงงานนี้มี 3 เครื่อง)

- Block Function
  - เป็นตัวบอกหน้าที่ของ Frame ว่าเป็น Frame สำหรับทำหน้าที่อะไร
  - มีขนาด 4 bits แบ่งออกเป็น

1. Poll กำหนดให้มีข้อมูลเป็น 1100 ทำหน้าที่ในการ Poll
2. Select กำหนดให้มีข้อมูลเป็น 0011 ทำหน้าที่ในการ Select
3. ACK1 กำหนดให้มีข้อมูลเป็น 1110 ทำหน้าที่ในการ ACK ตอบรับการส่งสัญญาณคำสั่ง และสัญญาณข้อมูลทุกเฟรมที่ไม่ใช่เฟรมสุดท้ายของสัญญาณข้อมูล
4. ACK2 กำหนดให้มีข้อมูลเป็น 1111 ทำหน้าที่ในการ ACK ตอบรับการส่งสัญญาณเฟรมสุดท้ายของสัญญาณข้อมูล
5. NAK กำหนดให้มีข้อมูลเป็น 0000 ทำหน้าที่ในการ NAK

- Block STOP
  - เป็นตัวบอกว่าเป็นส่วนท้ายของ Frame
  - มีข้อมูล 0000
  - มีขนาด 4 bits

2. *สัญญาณข้อมูล* สัญญาณข้อมูลจะเป็นข้อมูลของไฟล์ที่จะทำการส่งโดยลักษณะ Frame ของสัญญาณข้อมูลจะเป็นดังรูป

START	IDR	IDT	IDRR	DATA	STOP
4 bits	2 bits	2 bits	2 bits	1,000 bytes	4 bits

รูปที่ 3.5 ลักษณะ Frame ของสัญญาณข้อมูล

Frame สัญญาณข้อมูลจะมีขนาด 1,000 bytes ซึ่งประกอบไปด้วย Block ต่างๆ ดังนี้

- Block START
  - เป็นตัวบอกว่าเป็นส่วนแรกของ Frame
  - มีข้อมูล 1111
  - มีขนาด 4 bits
- Block IDR
  - เป็นตัวบอกหมายเลขของเครื่องคอมพิวเตอร์ของเครื่องรับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

— ข้อมูลจะขึ้นอยู่กับหมายเลขของเครื่องคอมพิวเตอร์ของเครื่องรับ โดยหมายเลขของเครื่องคอมพิวเตอร์ภายในระบบจะแบ่งออกเป็น

1. เครื่อง Server มีหมายเลข 000000
2. เครื่องลูกข่าย 1 มีหมายเลข 000001
3. เครื่องลูกข่าย 2 มีหมายเลข 000010

ต่อไปเรื่อยๆ จนถึงเครื่องลูกข่ายเครื่องที่ 63 มีหมายเลข 111111

— มีขนาด 6 bit (คอมพิวเตอร์ภายในระบบของโครงงานนี้มี 3 เครื่อง)

- Block IDT
  - เป็นตัวบอกหมายเลขของเครื่องคอมพิวเตอร์ของเครื่องส่ง
  - ข้อมูลจะขึ้นอยู่กับหมายเลขของเครื่องคอมพิวเตอร์ของเครื่องส่ง โดยหมายเลขของเครื่องคอมพิวเตอร์ภายในระบบจะแบ่งออกเป็น
    1. เครื่อง Server มีหมายเลข 000000
    2. เครื่องลูกข่าย 1 มีหมายเลข 000001
    3. เครื่องลูกข่าย 2 มีหมายเลข 000010

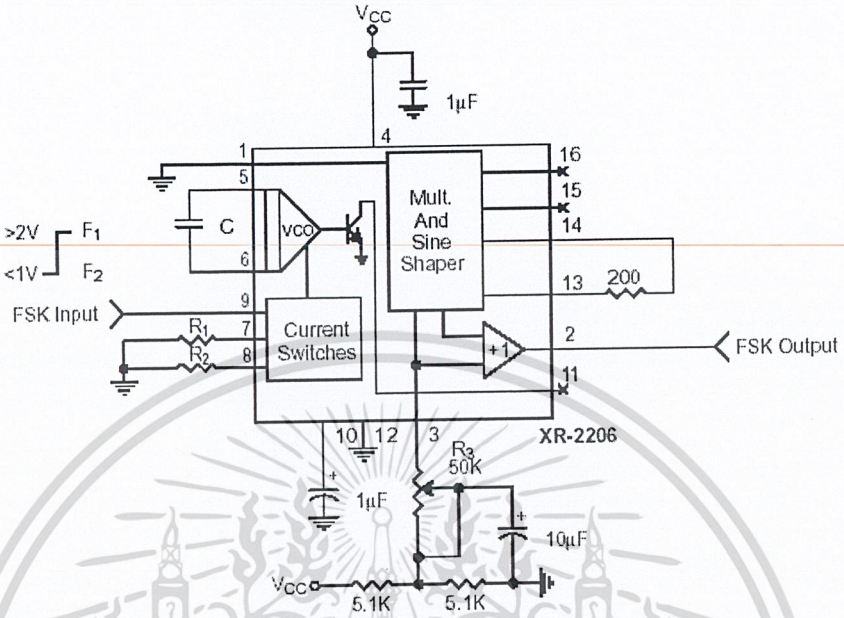
    ต่อไปเรื่อยๆ จนถึงเครื่องลูกข่ายเครื่องที่ 63 มีหมายเลข 111111
  - มีขนาด 6 bit (คอมพิวเตอร์ภายในระบบของโครงงานนี้มี 3 เครื่อง)
- Block IDRR
  - เป็นตัวบอกหมายเลขของเครื่องคอมพิวเตอร์ที่ต้องการจะส่งข้อมูล
  - มีขนาด 6 bits
- Block DATA
  - เป็นส่วนของข้อมูลที่อยู่ในไฟล์ข้อมูลที่จะทำการส่ง มีขนาด 1,000 bytes
- Block STOP
  - เป็นตัวบอกว่าเป็นส่วนท้ายของ Frame
  - มีข้อมูล 0000
  - มีขนาด 4 bits

3.2 วงจรฟรีเควนซีชิพท์คีย์อิ้งมอดูเลเตอร์ (Frequency Shift Keying Modulator)

ในที่นี้เลือกใช้ไอซีเบอร์ XR2206 ซึ่งเป็นอุปกรณ์โมโนลิทิคฟังก์ชันเจนเนอเรเตอร์ (Monolithic function generator) ใช้งานสำหรับสร้างสัญญาณซายน์ (sine), สามเหลี่ยม (triangle), แรมพ์ (ramp), พัลส์ (pulse) ที่มีคุณภาพ เสถียรภาพและความเที่ยงตรงสูง รูปคลื่นเอาท์พุตสามารถมอดูเลตได้ทั้งทางขนาดและทางความถี่ ค่าแรงดันภายนอกโดยมีช่วงความถี่ใช้งานตั้งแต่ 0.01 เฮิรตซ์ ถึง 1 เมกะเฮิรตซ์ ไอซีเบอร์นี้เหมาะสำหรับการใช้งานในระบบสื่อสารในระบบเครื่องมือวัด และระบบที่ต้องการแหล่งกำเนิดสัญญาณซายน์ เอเอ็ม เอฟเอ็ม หรือสัญญาณที่ผ่านการเข้ารหัสแบบเลื่อนความถี่มีค่าเฉลี่ยเบี่ยงเบนของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อุณหภูมิเป็น 20 ppm/องศาเซลเซียล ความถี่ออสซิลเลเตอร์สามารถกวาดเป็นเชิงเส้นได้สูงกว่า 2000 : 1 ช่วงความถี่ โดยการควบคุมของแรงดันภายนอกในขณะที่มีความเพี้ยนต่ำ



รูปที่ 3.6 บล็อกไดอะแกรมของไอซีเบอร์ XR2206

XR2206 ประกอบด้วย 4 ภาคการทำงาน ได้แก่ ส่วนแรงดันควบคุมการทำงานของออสซิลเลเตอร์ (Voltage Controlled oscillator) วงจรคูณอนาลอก วงจรปรับสัญญาณสวิตซ์กระแส ส่วนแรงดันควบคุมการทำงานของออสซิลเลเตอร์จะผลิตความถี่ที่เป็นสัดส่วนกับกระแสอินพุตซึ่งสามารถกำหนดได้ โดยตัวต้านทานที่ต่อเข้าที่ขาที่เกี่ยวกับเวลาลงกราวด์ เนื่องจากมี 2 ขาที่เกี่ยวกับเวลา ดังนั้นจึงสามารถสร้างสัญญาณได้ 2 ความถี่ สำหรับการกำเนิดสัญญาณเอฟเอสเคจะใช้ขาควบคุมสัญญาณเอฟเอสเค (FSK Input Control) ค่านี้จะไปควบคุมวงจรภายในส่วนสวิตช์กระแสเพื่อให้เกิดการเลือกขาที่ต่อตัวต้านทานสำหรับวงจรในส่วนแรงดันควบคุมการทำงานของออสซิลเลเตอร์

การเข้ารหัสแบบเอฟเอสเค XR2206 สามารถใช้งานโดยตัวต้านทาน (Timing resistor) 2 ตัวแยกกัน (R1, R2) ที่ขา 7 และขา 8 ของ XR2206 ดังรูปที่ 3.6 โดยตัวต้านทานแต่ละตัวจะถูกใช้นั้นขึ้นอยู่กับสัญญาณลอจิกที่เข้าที่ขา 9 ของ XR2206 หากขา 9 เปิดวงจรหรือต่ออยู่กับสัญญาณที่แรงดันสูงกว่าหรือเท่ากับ 2 โวลต์ แล้ว R1 จะเป็นตัวต้านทานกำหนดเวลา ทำนองเดียวกันหากขา 9 ต่ออยู่กับสัญญาณที่แรงดันต่ำกว่าหรือเท่ากับ 1 โวลต์ แล้ว R2 จะเป็นตัวต้านทานกำหนดเวลา ดังนั้นความถี่ของสัญญาณเอาท์พุตสามารถเข้ารหัสเป็น 2 ระดับที่มีความถี่เป็น f_1 และ f_2 โดยที่

$$f_1 = 1/R_1C \tag{3.1}$$

$$f_2 = 1/R_2C \tag{3.2}$$

ระดับแรงดันไฟตรงของเอาท์พุตที่ขา 2 ประมาณเท่ากับแรงดันไฟตรงที่ไบอัสเข้าที่ขา 3 ของ ไอซี ซึ่งจากรูปที่ 3.6 ขา 3 ถูกไบอัสด้วยครึ่งหนึ่งของค่าแรงดัน V+ เทียบกับกราวด์ ดังนั้นจะได้ค่าแรงดันไฟตรงที่เอาท์พุตเท่ากับ V/2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ฮาร์โมนิกที่เกิดขึ้นของคลื่นซายน์สามารถลดลงเหลือ 0.5% โดยการเพิ่มตัวต้านทานค่าประมาณ 200 โอห์ม ที่ขา 13 และขา 14 ของไอซีโดยแท็บ (Tap) ตรงกลางของตัวต้านทานปรับค่าได้ค่าประมาณ 500 โอห์ม แล้วทำการปรับความสมมาตรที่ขา 15, 16 ก่อน จากนั้นจึงปรับความเพี้ยน (Distortion) ที่ขา 13, 14

สำหรับโครงงานนี้ใช้ค่าความถี่สเปซ 4800 เฮิร์ตซ และความถี่มาร์ค 9600 เฮิร์ตซ ดังนั้นจะได้ค่าอุปกรณ์ต่างๆ ดังนี้

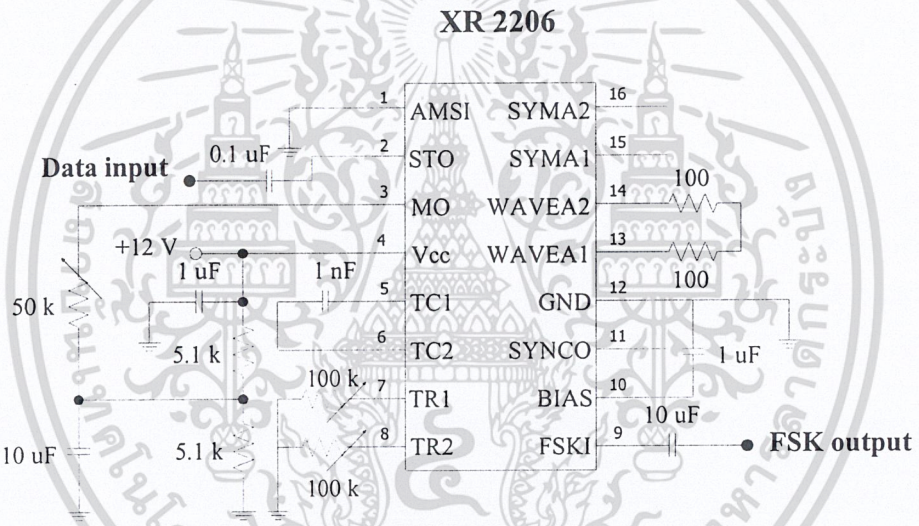
$$R_1 = 1 / f_1 C \tag{3.3}$$

โดย f_1 = 4800 เฮิร์ตซ และเลือกใช้ C = 1 nF ดังนั้นคำนวณค่า R1 ได้เท่ากับ 208.33 kΩ

$$R_2 = 1 / f_2 C \tag{3.4}$$

โดย f_2 = 9600 เฮิร์ตซ และเลือกใช้ C = 1 nF ดังนั้นคำนวณค่า R2 ได้เท่ากับ 104.166 kΩ

ดังนั้นรูปวงจรที่ได้จะเป็น



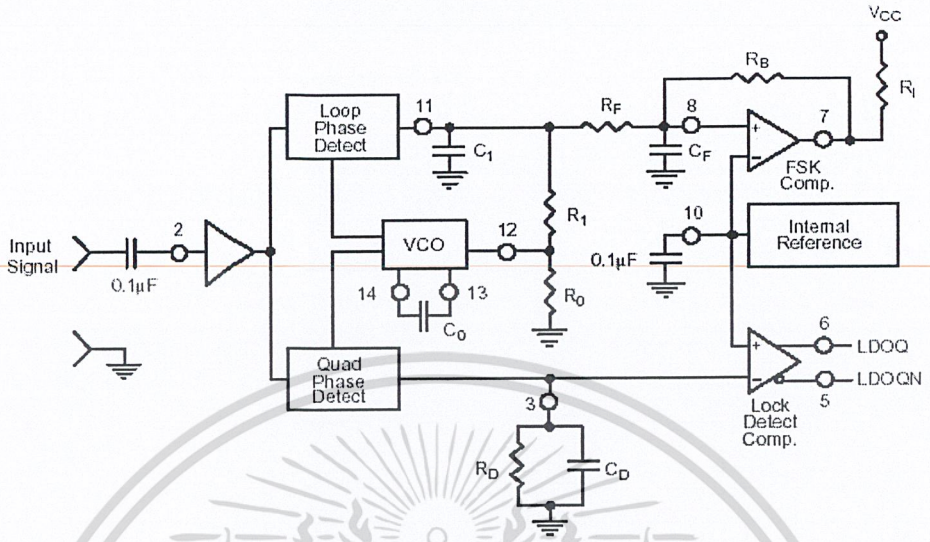
รูปที่ 3.7 วงจรฟรีเควนซีชิพท์คีย์อิ้งมอดูเลเตอร์

3.3 วงจรฟรีเควนซีชิพท์คีย์อิ้งดีมอดูเลเตอร์ (Frequency Shift Keying Demodulator)

XR2211 เป็นวงจรเฟสล็อคลูปแบบโมโนลิธิค (Monolithic) ซึ่งออกแบบมาสำหรับการใช้งานทำงานด้านการสื่อสารข้อมูล โดยเฉพาะลักษณะพิเศษของไอซีเบอร์นี้เหมาะสมสำหรับการใช้งานเป็นตัวถอดรหัสแบบฟรีเควนซี่ชิฟคีย์อิ้ง ไอซีทำงานในช่วงกว้างของไฟเลี้ยง คือ 4.5 โวลต์ ถึง 20 โวลต์ และมีช่วงความถี่ที่กว้างโดยอยู่ในช่วง 0.01 เฮิรตซ์ ถึง 300 กิโลเฮิรตซ์ สามารถใช้สัญญาณอนาลอกได้ในช่วง 2 มิลลิโวลต์ ถึง 3 โวลต์ อีกทั้งยังสามารถอินเทอร์เฟสได้กับทั้งวงจรลอจิกตระกูลดีทีแอล (DTL), ทีทีแอล (TTL) และอีซีแอล (ECL) วงจรภายในประกอบไปด้วยวงจรเฟสล็อคลูป สำหรับติดตามสัญญาณอินพุตในช่วงแถบผ่าน, วงจรควอดเดรเจอร์เฟสดีเท็คเตอร์ (Quadrature Phase Detector) ใช้สำหรับตรวจสอบสัญญาณพาหะ และตัวเปรียบเทียบแรงดันฟรีเควนซี่ชิฟท์คีย์อิ้ง (FSK voltage comparator) ใช้สำหรับดีมอดูเลตสัญญาณฟรีเควนซี่ชิฟคีย์อิ้งกับอุปกรณ์ที่ต่อภายนอกกำหนดได้อย่างอิสระ แรงดันอ้างอิงภายใน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นสัดส่วนกับไฟเลี้ยงวงจร ดังนั้นเหมาะสำหรับระบบที่มีการเปลี่ยนแปลงสมรรถภาพต่ำเมื่อไฟเลี้ยงวงจรเปลี่ยนแปลง



รูปที่ 3.8 บล็อกไดอะแกรมของไอซีเบอร์ XR2211

เอาท์พุตของเฟสดีเท็คเตอร์ให้สัญญาณผลบวก และผลต่างความถี่ของสัญญาณอินพุตกับสัญญาณอินพุตกับสัญญาณจากวงจรแรงดันควบคุมออสซิลเลเตอร์นั้นคือ ความถี่เอาท์พุตจะเป็น $f_{in} + f_{vco}$ และ $f_{in} - f_{vco}$ ดังนั้นในขณะล็อคความถี่ของสัญญาณเอาท์พุตจะเป็น $2f_{in}$ และ 0 เฮิรตซ์ โดยการต่อตัวเก็บประจุคร่อมสัญญาณเอาท์ของเฟสดีเท็คเตอร์ คือ ใส่วงจรกรองความถี่ต่ำนั้นเอง จะทำให้สัญญาณความถี่สูง $f_{in} + f_{vco}$ ถูกลดทอนไปทำให้เหลือเพียงแรงดันกระแสตรงที่เกิดจากความต่างเฟสของความถี่ทั้งสองเกิดขึ้นเช่นนี้เรื่อยๆ ทำให้วงจรแรงดันควบคุมออสซิลเลเตอร์ ติดตามความถี่ของสัญญาณอินพุตได้

ส่วนที่เหลือของ XR2211 ทำงานดังนี้คือ หากวงจรแรงดันควบคุมออสซิลเลเตอร์ ถูกขับด้วยความถี่สูงกว่าหรือต่ำกว่าความถี่ศูนย์กลาง แล้ววงจรเปรียมเทียบแรงดันจะสร้างสัญญาณเอาท์พุตลอจิกสูงและสัญญาณเอาท์พุตลอจิกต่ำ เพื่อเฟสล็อกลูปอยู่ในช่วงล็อค (quadrature phase detector and lock detector comparator) สัญญาณเอาท์พุตลอจิกต่ำ เมื่อเฟลล็อกลูปอยู่ในช่วงล็อค (quadrature phase detector and lock detector comparator)

การถอดรหัสสัญญาณที่เข้ารหัสแบบเลื่อนความถี่เอฟเอสเค รูปที่ 3.8 แสดงการต่อพื้นฐานของวงจรถอดรหัสแบบเอฟเอสเคโดยอ้างอิงจากรูปที่ 3.8 และรูปที่ 3.9 หน้าที่ของอุปกรณ์ภายนอกแต่ละตัวเป็นดังนี้ R_0 และ C_0 กำหนดความถี่ศูนย์กลางของเฟสล็อกลูป R_1 กำหนดแถบความถี่ของระบบ C_1 ใช้กำหนดค่าเวลาในวงจรกรองความถี่แบบวนรอบ (Loop Damping) สำหรับข้อมูลเอาท์พุตของสัญญาณเอฟเอสเค (R_B = 510 kΩ) จากขา 7 ไปขา 8 มีไว้เพื่อเป็นการป้อนกลับแบบบวกสำหรับตัวเปรียบเทียบแรงดันของสัญญาณเอฟเอสเค เพื่อให้เกิดความรวดเร็วในการเปลี่ยนสถานะลอจิก

วงจรตามรูปที่ 3.8 สามารถใช้ได้กับวงจรถอดรหัสเอฟเอสเค ทั่วๆ ไป โดยการเลือกอุปกรณ์ต่อภายนอกสำหรับการกำหนดความถี่มาร์ค และความถี่สเปช (f_1, f_2) ค่าพารามิเตอร์สามารถคำนวณได้จาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำนวณความถี่ศูนย์กลางของเฟสล็อกลูป : f_0

$$f_o = \sqrt{f_1 f_2} \tag{3.5}$$

โดย f_1 หรือความถี่มาร์คมีค่า 9600 Hz และ f_2 หรือความถี่สเปซมีค่า 4800 Hz

$$f_0 = 6788.2251$$

เลือกค่าของตัวต้านทานกำหนดเวลา : R_0 ให้อยู่ในช่วง 10 กิโลโอห์ม ถึง 100 กิโลโอห์ม โดยสามารถเลือกได้ตามใจชอบ สำหรับค่าอ้างอิงของ R_0 เป็น 20 กิโลโอห์ม ซึ่งต่อความต้านทานปรับค่าได้ไว้สำหรับปรับค่าละเอียด (Rx) 20 กิโลโอห์ม

$$R_T = R_0 + \frac{R_x}{2} \tag{3.6}$$

$$R_T = 20k + \frac{20k}{2} = 30k$$

คำนวณค่าของ C_0 จากสมการ

$$C_0 = \frac{1}{R_T f_0} \tag{3.7}$$

$$C_0 = 4.91nF$$

คำนวณค่า R_1 เพื่อกำหนด ความเบี่ยงเบนของความถี่มาร์ค และความถี่สเปซ

$$R_1 = \frac{R_T f_0}{f_1 - f_2} x\,2 \tag{3.8}$$

$$R_1 = 84.852k$$

คำนวณค่า C_1 เพื่อกำหนดการวนรอบ โดยค่าการรอบมีค่าอ้างอิงเท่ากับ 1/2 แล้ว

$$C_1 = \frac{1250 C_0}{R_1 \varsigma^2} \tag{3.9}$$

$$C_1 = 289.33pF$$

คำนวณค่า R_f

$$R_f = 5R_1 \tag{3.10}$$

$$R_f = 424.260k$$

คำนวณค่า R_B

$$R_B = 5R_f \tag{3.11}$$

$$R_B = 2.121M$$

และคำนวณค่า R_{sum} ได้จาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$R_{SUM} = \frac{\left(R_f + R_1\right) \times R_B}{\left(R_f + R_1 + R_B\right)} \tag{3.12}$$

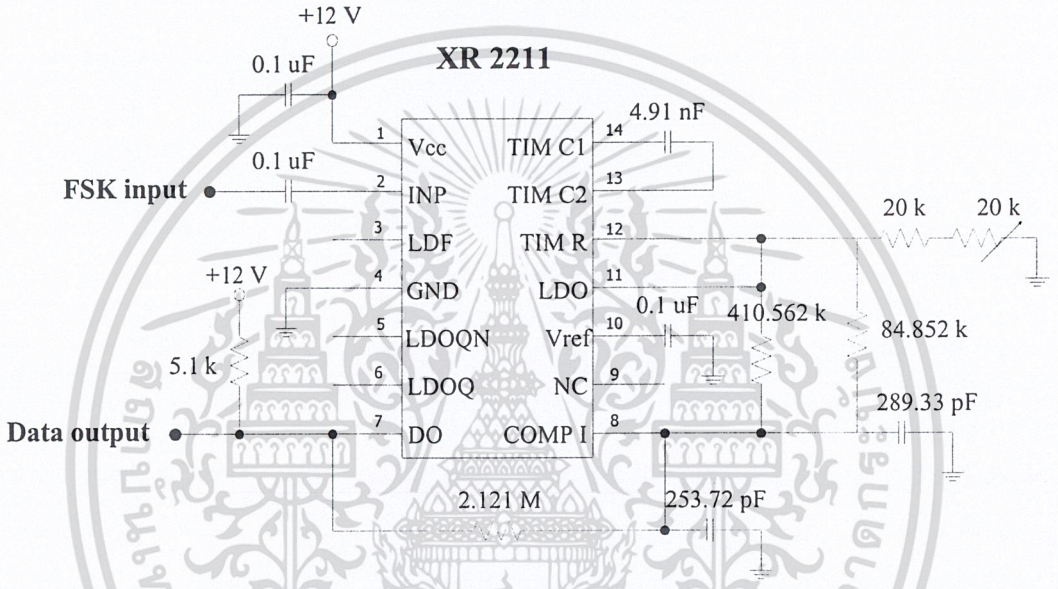
$$R_{sum} = 410.562k$$

ดังนั้นสามารถคำนวณค่า C_f

$$C_f = \frac{0.25}{R_{sum} \times BaudRate} \tag{3.13}$$

$$C_f = 253.72pF$$

วงจรที่ได้จะเป็นดังรูป



รูปที่ 3.9 วงจรฟรีเควนซีชิพท์คีย์อิ้งดีมอดูเลเตอร์

3.4 วงจรสังเคราะห์ความถี่ (Frequency Synthesizer)

วงจรสังเคราะห์ความถี่ที่ใช้ในโครงงานนี้ จะอยู่ในส่วนของวงจรฟรีเควนซีมอดูเลเตอร์ และวงจรฟรีเควนซีดีมอดูเลเตอร์ ซึ่งในโครงงานนี้ได้ทำการส่งข้อมูลจากเครื่อง Server ไปยังเครื่องลูกข่ายที่ความถี่ 50 เมกะเฮิร์ตซ ดังนั้นที่ลูกข่ายทั้ง 2 ตัว จะต้องมีวงจรสังเคราะห์ความถี่ที่ให้กำเนิดความถี่ 60.7 เมกะเฮิร์ตซ ป้อนให้กับวงจรฟรีเควนซีดีมอดูเลเตอร์ เพื่อให้สามารถแปลงความถี่เอฟเอ็มให้กลายเป็นความถี่ไอเอฟได้ ส่วนการส่งข้อมูลจากเครื่องลูกข่ายไปยังเครื่อง Server จะใช้ความถี่ 27 เมกะเฮิร์ตซ ดังนั้นที่เครื่อง Server ก็จะต้องมีวงจรสังเคราะห์ความถี่ที่ให้กำเนิดความถี่ 37.7 เมกะเฮิร์ตซเช่นกัน ในที่นี้จะแสดงการคำนวณเฉพาะวงจรสังเคราะห์ความถี่ที่ให้กำเนิดความถี่ 50 เมกะเฮิร์ตซ ส่วนที่ความถี่อื่นๆ ก็สามารถคำนวณได้ในทำนองเดียวกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.4.1 วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์

ออกแบบวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ แบบโคลพิตต์ออสซิลเลเตอร์โดยใช้ JFET เบอร์ J310 ความสัมพันธ์ระหว่างกระแสเอาต์พุต (I_D) กับแรงดันที่ใช้ควบคุมการเปลี่ยนแปลง (V_{GS}) เป็นไปตามสมการกำลังสอง

$$I_{DS} = I_{DSS}\left[1-\left(\frac{V_{GS}}{V_P}\right)\right]^2 \tag{3.14}$$

การคำนวณหาจุด Q ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ จะใช้สมการกฎกำลังสองและสมการโหลดไลน์ (Load Line) ของไบอัสตัวเอง (self-bias) คือ

$$V_{GS} = -I_D R_S \tag{3.15}$$

แทนค่าพารามิเตอร์ต่าง ๆ ของ J310 จากดาต้าชีตดังนี้

$$I_{DSS} = 40mA$$

$$V_P = -4V$$

$$R_S = 100\Omega$$

ทำการแก้สมการเพื่อหาจุด Q ดังนั้นจะได้ค่าต่าง ๆ ดังนี้

$$I_{DQ} = 15.278mA$$

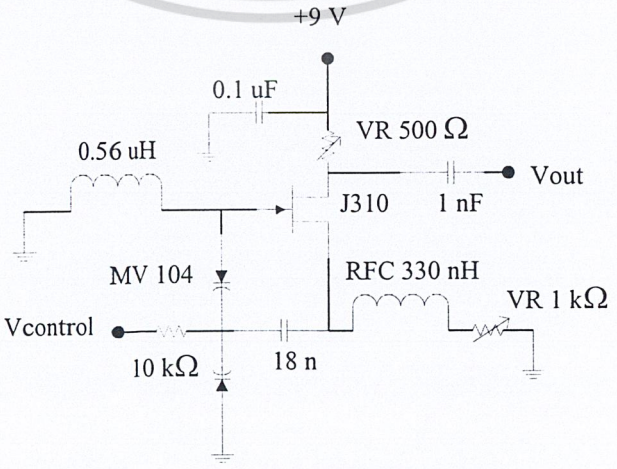
$$V_{GSQ} = -1.5278V$$

และคำนวณหาค่า L, C_1 และ C_2 ได้จากสมการ

$$f = \frac{1}{2\pi\sqrt{LC_T}} \tag{3.16}$$

$$C_T = \frac{C_1 C_2}{C_1 + C_2} \tag{3.17}$$

จะได้วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ ดังรูปที่ 3.10

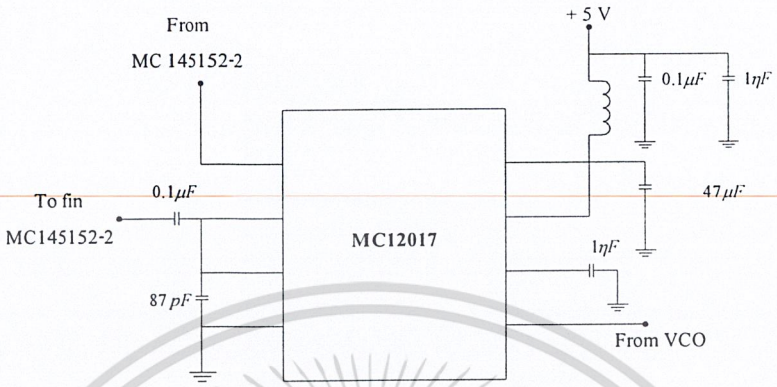


รูปที่ 3.10 วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.4.2 วงจรพรีสเกลเลอร์แบบสองโมดูลัส

พรีสเกลเลอร์เป็นตัวหารความถี่ที่เข้ามาให้ต่ำลง เพื่อที่จะป้อนให้กับวงจรเฟสดีเทกเตอร์ ในที่นี้ใช้ ไอซีเบอร์ MC12017 ซึ่งจะถูกตั้งหารเท่ากับ 64/65 โดยทำการต่อวงจรดังรูปที่ 3.11



รูปที่ 3.11 วงจรพรีสเกลเลอร์แบบสองโมดูลัส

3.4.3 วงจรเปรียบเทียบเฟสและการตั้งโปรแกรมหารความถี่

ไอซีเบอร์ MC145152-2 ประกอบด้วยส่วนที่เป็นการโปรแกรมเพื่อหารความถี่ที่เข้ามา ส่วนของความถี่อ้างอิงและส่วนของเฟสดีเทกเตอร์ดังรูปที่ 3.12 ซึ่งไอซีนี้สามารถตั้งโปรแกรมเพื่อหารความถี่ที่เข้ามาให้ได้ 16 บิต แบบขนาน โดย 10 บิต สำหรับตัวหาร N (10 bit/N Counter) และ 6 บิต สำหรับตัวหาร A (6 bit/A Counter) และมีโมดูลัสคอนโทรล (Modulus Control:MC) เพื่อส่งสัญญาณไปควบคุมการหารของไอซี พรีสเกลเลอร์ นอกจากนี้ยังมีล็อกดีเทกต์ (Lock Detect:LD) เพื่อใช้บอกสถานะการล็อกของเฟสล็อกลูปอีกด้วย

ในส่วนความถี่อ้างอิงจะใช้คริสตอล 10.24 MHz เป็นตัวอ้างอิง และสามารถหารความถี่ลงมาโดยกำหนดค่า RA_0 RA_1 และ RA_2 ซึ่งดูจากตาราง ที่ 3.1

Reference Address Code			Total Divide Value
RA1	RA1	RA0	
0	0	0	8
0	0	1	64
0	1	0	128
0	1	1	256
1	0	0	512
1	0	1	1024
1	1	0	1160
1	1	1	2048

ตารางที่ 3.1 การโปรแกรมค่าให้กับวงจรหารภายในไอซี MC145152-2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในการทดลองนี้กำหนด RA_0 RA_1 และ RA_2 เป็น "1 0 1" ซึ่งเท่ากับการหารด้วย 1024 ฉะนั้นจะได้ความถี่อ้างอิงภายในตัวไอซี คือ

$$f_{ref} = \frac{10.24x10^6}{1024} = 10kHz$$

ในการตั้งโปรแกรมหารความถี่มีวิธีการคำนวณ ดังนี้

1. ทำการหาค่าอัตราส่วนการหารทั้งหมด (N_t) โดยคิดจากอัตราส่วนระหว่างค่าความถี่ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ (f_{VCO}) กับค่าความถี่อ้างอิง (f_{ref}) ได้เป็นดังนี้

$$N_t = \frac{f_{VCO}}{f_{ref}} = \frac{50MHz}{10kHz} = 5000$$

2. เปรียบเทียบค่าที่ได้กับสมการ $N_t = NP + A$ จากดาต้าชีตของไอซี MC145152-2 โดยที่ N คือ ตัวโปรแกรม $N_0 - N_9$ = 10 บิต มีค่าตั้งแต่ 0 - 1023

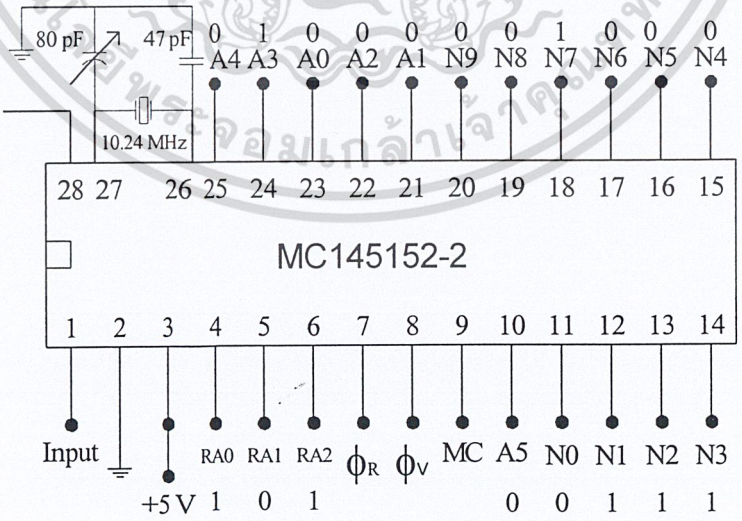
A คือ ตัวโปรแกรม $A_0 - A_5$ = 6 บิต มีค่าตั้งแต่ 0 – 63

P คือ ตัวหารของพรีสเกลเลอร์ หาร 64

แทนค่าสมการได้ดังนี้ N = 78 แปลงเป็นเลขฐาน 2 ได้เป็น (MSB) 0001001110 (LSB)

A = 8 แปลงเป็นเลขฐาน 2 ได้เป็น (MSB) 001000 (LSB)

สังเกตว่าถ้าตัวหารมีค่ามากขึ้นจะทำให้ความถี่ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์มีค่ามากขึ้น และถ้าตัวหารน้อยลงก็จะทำให้ความถี่ของวงออสซิลเลเตอร์ควบคุมด้วยแรงดัน มีค่าลดลงตามไปด้วย ซึ่งค่าความถี่ที่เปลี่ยนแปลงไปนี้จะมีค่าเท่ากับความถี่อ้างอิง 10 kHz ต่อ 1 บิต ของ A และจะเปลี่ยนแปลงไป 640 kHz (64 x 10 kHz) ต่อ 1 บิต ของ N ซึ่งจากการคำนวณข้างต้นจะสามารถนำมาสร้างเป็นวงจรได้ดังรูปที่ 3.12



รูปที่ 3.12 วงจรเปรียบเทียบเฟสและการตั้งโปรแกรมหารความถี่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.4.4 วงจรลูปฟิลเตอร์ (Loop Filter)

การออกแบบวงจรลูปฟิลเตอร์ ออกแบบโดยใช้ไอซีเบอร์ LM358 ซึ่งมีการออกแบบตามคาต้าชีตของไอซีเบอร์ MC145152-2 โดย

f_0 = Local oscillator frequency (3.18)

f_s = Channel spacing (3.19)

f_b = $0.01\, f_s$ (3.20)

f_{rc} = $20\, f_b$ (3.21)

δ = 0.8 (3.22)

V_{DD} = Supply voltage (3.23)

$$K_{VCO} = 2\pi \frac{\Delta f_{VCO}}{\Delta V_{VCO}} \quad (3.24)$$

C_1 = Fix (3.25)

$$K_\phi = \frac{V_{DD}}{2\pi} \quad (3.26)$$

$$\omega_n = \frac{2\pi f_b}{\sqrt{2\delta^2+1+\sqrt{\left(2\delta^2+1\right)^2+1}}} \quad (3.27)$$

$$N_t = \frac{f_0}{f_s} \quad (3.28)$$

$$R_1 = \frac{K_\phi K_{VCO}}{C_1 \omega_n^2 N_t} \quad (3.29)$$

$$R_2 = \frac{2\delta}{\omega_n C_1} \quad (3.30)$$

$$C_c = \frac{4}{2R_1 f_{rc}} \quad (3.31)$$

จากสมการด้านบนส่วนที่จะหาค่าลงในวงจร คือ R_1, R_2 และ C_c เพราะฉะนั้นแทนค่าลงในสมการ คือ

$$f_0 = 50MHz$$

$$f_s = 10kHz$$

$$f_b = 0.01 \times 10kHz = 100Hz$$

$$f_{rc} = 20 \times 100 = 2Hz$$

$$V_{DD} = 5V$$

$$K_{VCO} = 2\pi \frac{(51.7-48.7)\times 10^6}{(5-3.5)} = 12.5664 \times 10^6 rad/s$$

$$C_1 = 0.1\mu F$$

$$K_\phi = \frac{5}{2\pi} = 0.7958$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\omega_n = \frac{2\pi(100)}{\sqrt{2(0.8)^2+1+\sqrt{(2(0.8)^2+1)^2+1}}} = 287.6976 rad/s$$

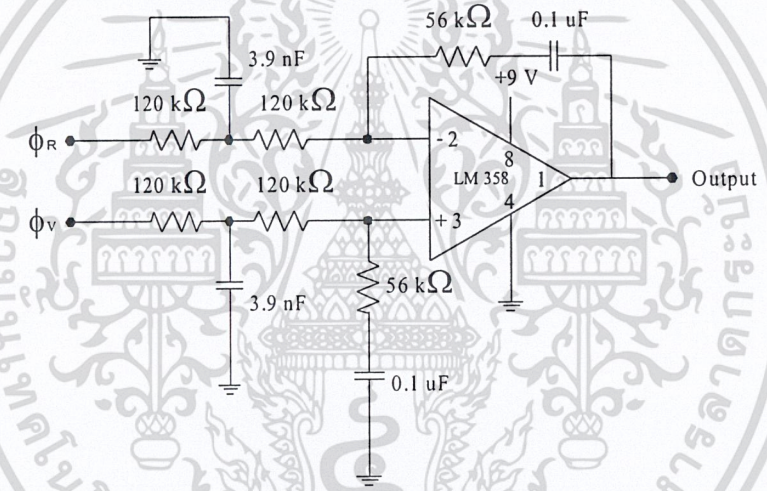
$$N_t = \frac{50MHz}{10kHz} = 5000$$

$$R_1 = \frac{(0.7958)(12.5664\times10^6)}{(0.1\times10^{-6})(287.6976)^2(5000)} = 241.64k\Omega$$

$$R_2 = \frac{2(0.8)}{(287.6976)(0.1\times10^{-6})} = 55.6k\Omega$$

$$C_c = \frac{4}{2(241.64\times10^3)(2\times10^3)} = 4.138nF$$

หลังจากนั้นทำการแบ่ง R_1 ออกเป็น 2 ตัว จึงได้ค่าประมาณ คือ 120 kHz และ 120 kHz ตามลำดับ R_2 ประมาณ 56 kHz นำมาสร้างเป็นวงจรลูปฟิลเตอร์ได้ดังวงจรในรูปที่ 3.13



รูปที่ 3.13 วงจรลูปฟิลเตอร์

3.5 วงจรฟรีเควนซีดีมอดูเลเตอร์ (Frequency Demodulator)

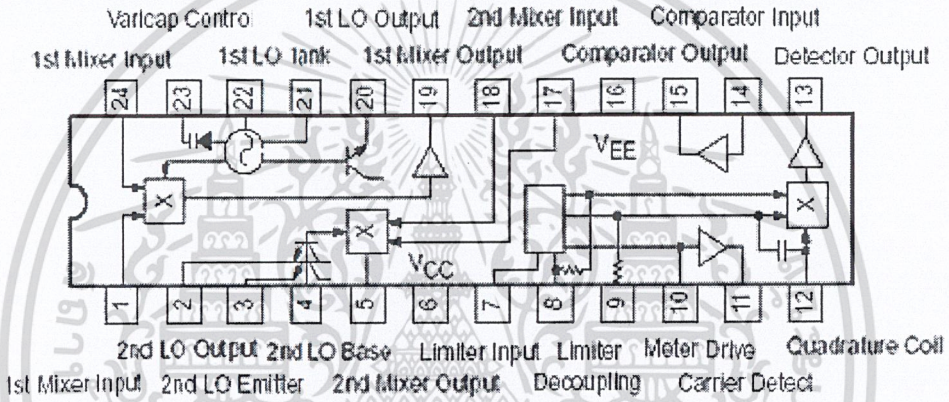
วงจรภาคไอเอฟและดีมอดูเลเตอร์ในที่นี้ใช้ไอซีเบอร์ MC 3362 ซึ่งประกอบด้วยส่วนต่างๆดังนี้

- วงจรผสมสัญญาณ(Mixer) ซึ่งจะทำหน้าที่เลื่อนความถี่ของสัญญาณไปไว้ในช่วงความถี่กลางซึ่งทำการเลื่อนความถี่ 2 ต่อ (Double Conversion) โดยครั้งแรกเลื่อนความถี่ไปที่ 10.7 เมกะเฮิรตซ์ ก่อนแล้วจึงเลื่อนความถี่ของสัญญาณอีกครั้งหนึ่งไปที่ 455 กิโลเฮิรตซ์
- เซรามิคฟิลเตอร์(Ceramic Conversion) ทำหน้าที่เป็นวงจรกรองช่วงความถี่ผ่าน 10.7 เมกะเฮิรตซ์ เพื่อให้ได้ความถี่กลาง 10.7 เมกะเฮิรตซ์ออกมา
- เซรามิคฟิลเตอร์ 455 กิโลเฮิรตซ์ ทำหน้าที่เป็นวงจรกรองช่วงความถี่ผ่าน 455 กิโลเฮิรตซ์ เพื่อให้ได้ความถี่กลาง 455 เมกะเฮิรตซ์ออกมาที่จะส่งเข้าไปขยายที่วงจรลิมิตเตอร์ในไอซี MC 3362 อีกครั้ง

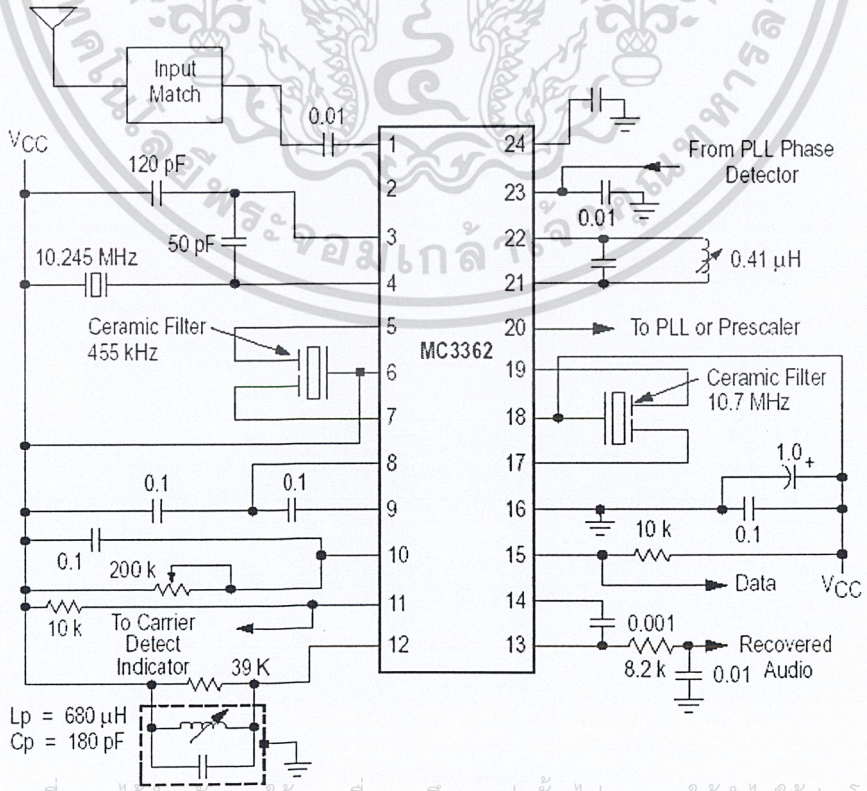
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- วงจรขยายลิมิตเตอร์ (Limitter) เป็นวงจรขยายสัญญาณที่ได้จากเซรามิคฟิลเตอร์ให้แรงขึ้นและให้ได้ความถี่ขนาด 455 กิโลเฮิรตซ์จริงๆ โดยการขลิบสัญญาณที่เกินออกเพื่อให้เอาท์พุตมีระดับสัญญาณที่สม่ำเสมอกันเพื่อลดความผิดเพี้ยนของสัญญาณก่อนที่จะส่งไปยังวงจรดีเทคเตอร์

- วงจรดีเทคเตอร์ จะทำการดีเทคสัญญาณที่รับมาจากวงจรลิมิตเตอร์ ให้เป็นสัญญาณข่าวสารดั้งเดิมที่ทำการส่งมา โดยที่มีหลักการทำงาน คือ ลดความถี่จากสัญญาณเอฟเอ็มที่รับมาได้เป็น 10.7 MHz โดยใช้หลักการผสมสัญญาณ (Mixer) ในที่นี้ความถี่ของสัญญาณเอฟเอ็ม คือ 50 MHz เพราะฉะนั้นสัญญาณที่นำมาผสมกับ 50 MHz คือ 60.7 MHz จากนั้นนำสัญญาณเอฟเอ็มที่ความถี่ 10.7 MHz ไปผสมกับความถี่ 10.245 MHz ที่มาจากคริสตอลออสซิลเลเตอร์ ได้สัญญาณเอฟเอ็มความถี่ 455 kHz จากนั้นนำสัญญาณเอฟเอ็มความถี่ 455 kHz ไปดีมอดูเลตโดยใช้ Quadrature discriminator ได้สัญญาณข่าวสารที่ต้องการ

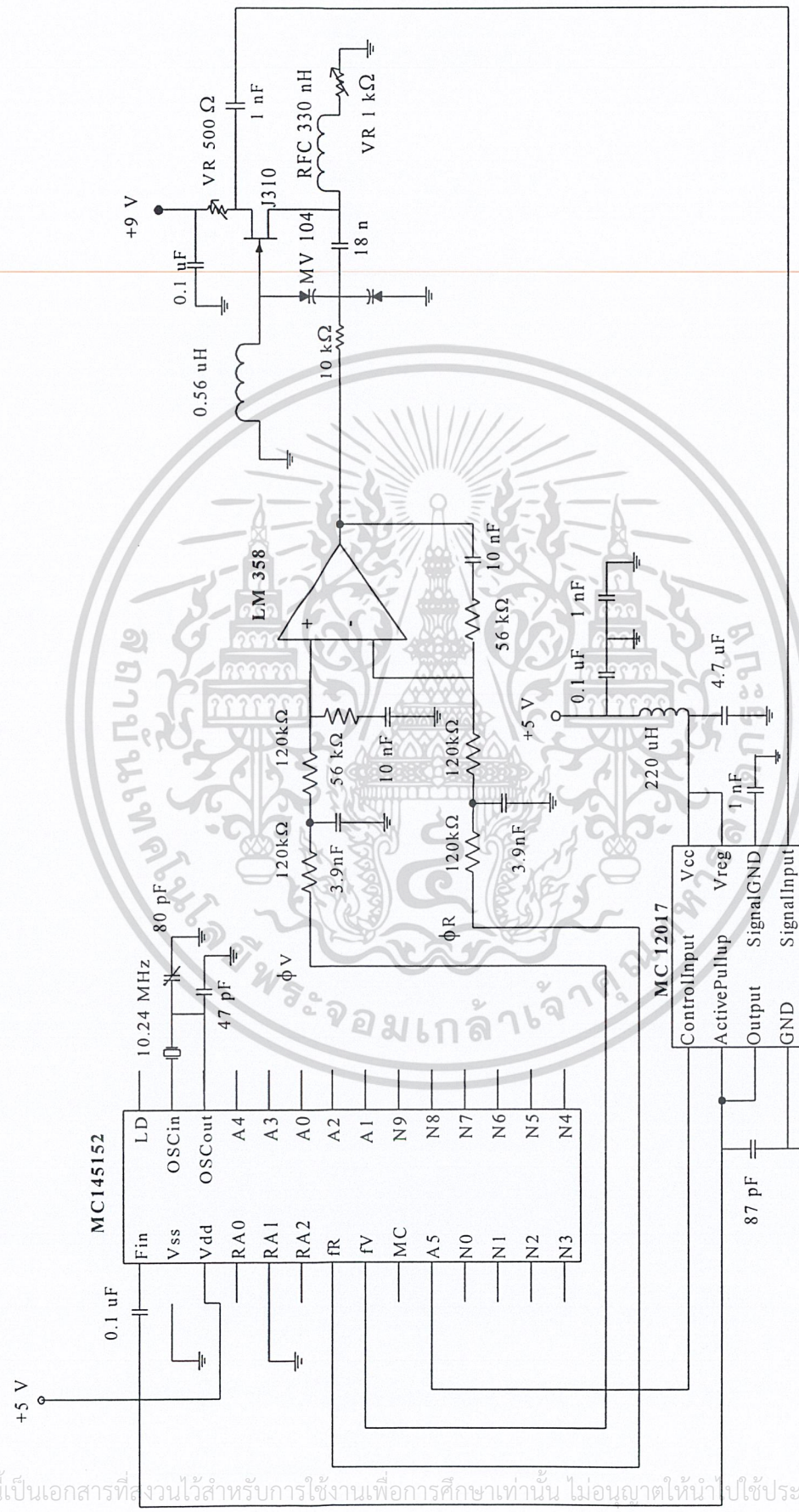


รูปที่ 3.14 บล็อกไดอะแกรมของไอซี MC3362



รูปที่ 3.15 วงจรฟรีเควนซีดีมอดูเลเตอร์

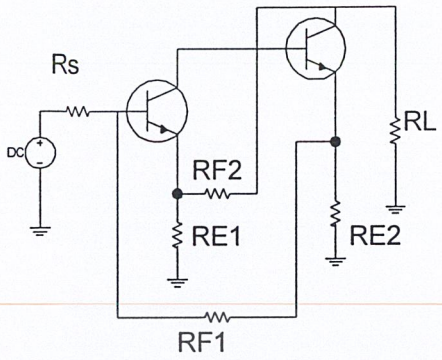
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.16 วงจรสังเคราะห์ความถี่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.6 วงจรขยายความถี่ช่วงกว้าง (Wideband Ampilfier)



รูปที่ 3.17 วงจรขยายแบบดาร์ลิงตั้น

มีหลักการการออกแบบเป็นวงจรขยาย 2 วงจรต่อแบบคาสเคด และมีการป้อนกลับ 2 ลูป ผ่านทาง RF1 และ RF2 เพื่อให้วงจรมีแบนด์วิธกว้าง และมีอัตราขยายสูง โดยอินพุตอิมพีแดนซ์ และเอาท์พุตอิมพีแดนซ์ เท่ากับ 50 โอห์ม จากทฤษฎีของ negative Feedback ดังรูป จะมีผลต่ออัตราขยายของวงจรและทำให้อินพุตอิมพีแดนซ์ของวงจรเดิมที่มีค่าสูงให้ต่ำลงเป็น 50 โอห์ม ในทำนองเดียวกัน RF2 ก็จะลดค่าเอาท์พุตอิมพีแดนซ์จากเดิมให้เหลือ 50 โอห์ม ได้เช่นกัน ซึ่งความสัมพันธ์ระหว่างอัตราขยาย และแบนด์วิดท์จะมีค่าแปรผกผันกัน โดยจะมีค่าอัตราขยายเป็นดังสมการ

$$A_v = (R_{F1} + R_{F2}) / R_{E1} \tag{3.32}$$

ในที่นี้เลือกใช้วงจรที่สร้างจากทรานซิสเตอร์เบอร์ 2N3904 โดยทำการต่อวงจรดังรูป โดยวงจรนี้มีการปรับปรุงวงจรให้สามารถรับอินพุตได้สูงขึ้น โดยใช้ทรานซิสเตอร์จำนวน 4 ตัว โดย Q1 เป็นตัวขยายสัญญาณส่วนแรก Q2 กับ Q3 ใช้เพิ่มอัตราขยายโดยต่อเป็นวงจรดาร์ลิงตัน ส่วน Q4 ใช้ต่อเป็นแบบไดโอดเพื่อทำให้แรงดันที่ตกคร่อม Q3 มีค่าเป็น 0.7 โวลต์ ซึ่งจะทำให้ภาคเอาต์พุตมีกำลังสูงขึ้นจากกระแสที่เพิ่มขึ้น โดยค่าอัตราขยายจะอยู่ในระดับประมาณ +10 dBm และเพื่อป้องกันไม่ให้เกิดการอิ่มตัว(Saturate) จึงทำการต่อวงจรนี้ 2 ชุดเข้าด้วยกันโดยมีวงจรลดทอนสัญญาณแบบ พายโมเดลเชื่อมระหว่าง 2 ชุดเพื่อลดทอนให้อยู่ในระดับประมาณ +14 dBm และยังเป็นการแยกสัญญาณเพื่อให้มองเห็นโหลดได้ชัดเจนอีกด้วย

ซึ่งมีวิธีการออกแบบใช้งานดังนี้

1. เลือกทรานซิสเตอร์ที่มีค่า f_T มากกว่าความถี่ที่ใช้งานประมาณ 10 เท่า
2. เลือกจุดทำงาน Q-point โดยกำหนด I_{C1}, I_{C2}, V_{CE1} และ V_{CE2} ที่ต้องการ
3. กำหนด V_{CC} และเลือก V_{E1}, V_{E2} จะได้ค่า R ไบอัสที่จุดต่างๆดังนี้

$$R_{E1} = V_{E1} / I_{C1} \text{ R}_{\text{E1}} \text{ และ } R_{E2} = V_{E2} / I_{C2}$$

$$R_{C1} = \{V_{CC} - (V_{CE1} + V_{E1})\} / I_{C1}$$

$$R_{C2} = \{V_{CC} - (V_{CE2} + V_{E2})\} / I_{C2\,2}$$

$$R_{f2} = \beta\{V_{E2} - (V_{E1} + V_{EE})\} / I_{C1}$$

4. เลือกค่า A_v ที่ต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. กำหนดค่า R_{E1}, R_{E2}, R_{F1}, R_{F2} ให้เหมาะสม เพื่อให้ได้ค่า R_0 และ R_i มีค่าประมาณ 50 โอห์ม

ดังนั้น ค่าที่เลือกใช้ในโครงงานนี้คือ

$f_T = 200MHz$

$V_{CC} = 5V$

$I_{C1} = 10mA$ $I_{C2} = 20mA$

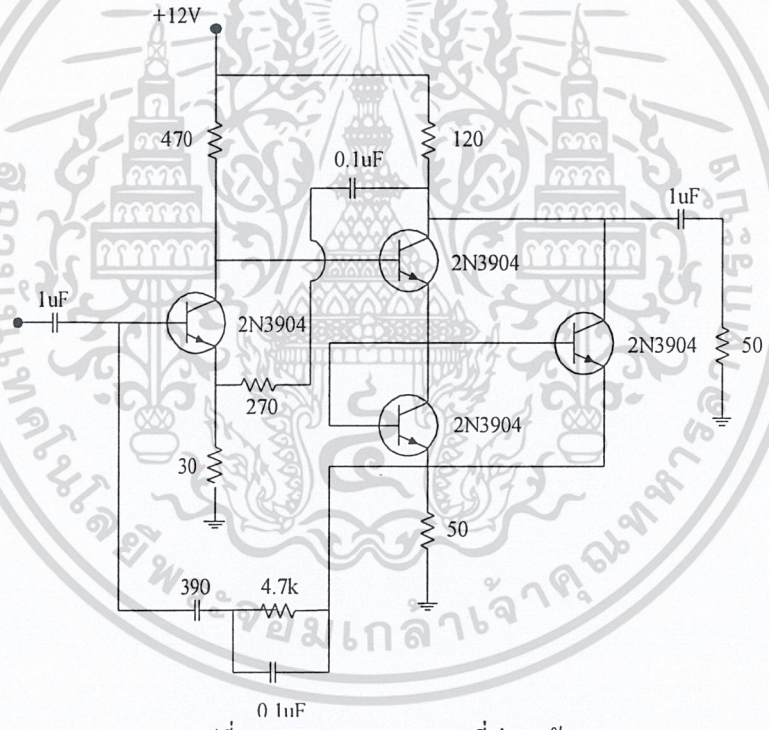
$R_{E1} = 30\Omega$ $R_{E2} = 50\Omega$

$R_{C1} = 300\Omega$ $R_{C2} 120\Omega$

$R_{F1} = 270\Omega$ $R_{F2} = 470\Omega$

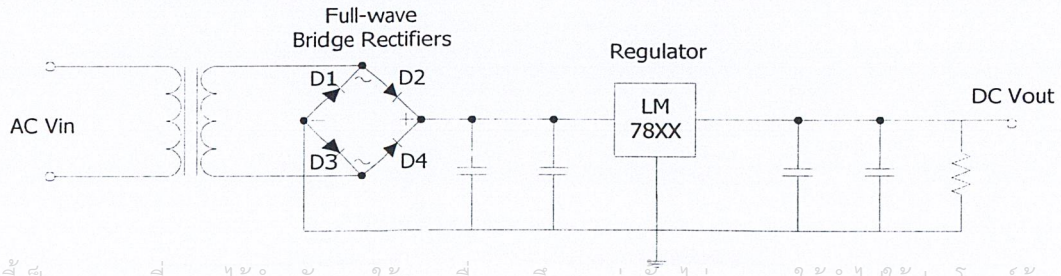
$A_v = 20$ dBm

$R_i = 38.5\Omega$ $R_0 = 120\Omega$



รูปที่ 3.18 วงจรขยายความถี่ช่วงกว้าง

3.7 วงจรแหล่งจ่ายกำลัง (Power Supply)

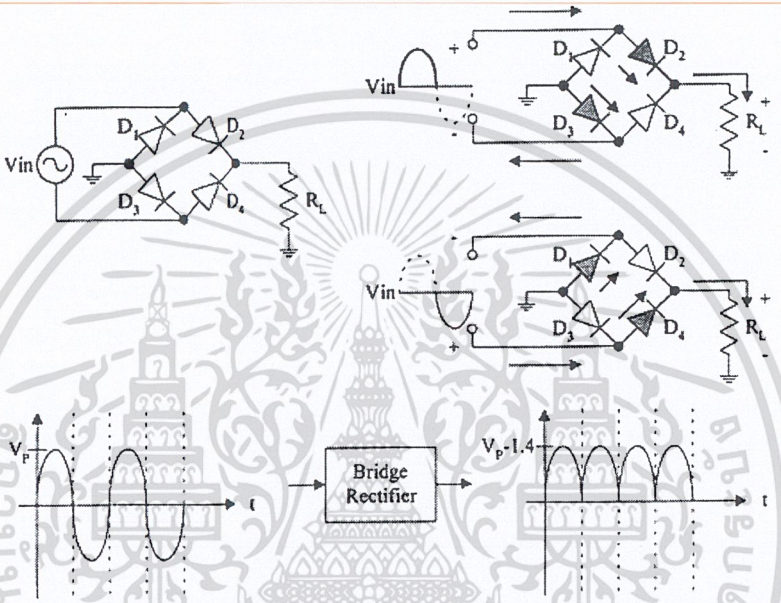


รูปที่ 3.19 วงจรแหล่งจ่ายกำลัง DC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แหล่งจ่ายกำลังเป็นอุปกรณ์ที่ใช้แปลงแรงดันแบบ AC ให้กลายเป็นแรงดันแบบ DC ประกอบด้วยอุปกรณ์พื้นฐานดังแสดงในรูปที่ 3.18 คือ

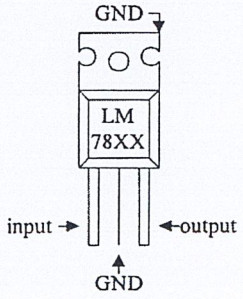
1. *Rectifiers* เป็นอุปกรณ์ที่ใช้แปลงไฟ AC เป็นไฟ DC ซึ่งจะยังคงทำให้สัญญาณไฟ DC มีการกระเพื่อมอยู่ (AC-to-DC Converter) โดยในที่นี้เลือกใช้ Full-Wave Bridge Rectifiers เป็นวงจรแปลงกระแสแบบเรียงเต็มลูกคลื่น จะมีทั้งแบบที่ใช้ไดโอด 4 ตัวต่อกันแบบ bridge และแบบที่เป็นอุปกรณ์สำเร็จรูป 4 ขา โดย Rectifier แบบไดโอดที่ต่อกันแบบ bridge มีหลักการทำงานดังแสดงในรูปที่ 3.19



รูปที่ 3.20 หลักการทำงานของ Bridge Rectifier แบบไดโอด

2. *Low pass Filter* ทำให้ไฟ DC ที่ได้จาก Rectifier เรียบมากขึ้น โดยใช้หลักการ Charge และ Discharge ของตัวเก็บประจุ ซึ่งสามารถเลือกใช้ได้หลายแบบอาจเป็นแอกทีฟฟิลเตอร์หรือพาสซีฟฟิลเตอร์ก็ได้

3. *Regulator* เป็นอุปกรณ์ที่ให้ไฟ DC ค่าต่างๆ โดยการแปลงจากแหล่งกำเนิดไฟ DC (DC-to-DC Converter) โดย Regulator เป็นอุปกรณ์สารกึ่งตัวนำ 3 ขา มีหน้าที่แปลงระดับแรงดัน DC ไปเป็นแรงดัน DC อีกระดับหนึ่ง มีขา 3 ขา คือ ขา input, ground และ output มีลักษณะแสดงในรูปที่ 3.20



รูปที่ 3.21 ตัวอย่างของวงจร Rectifier

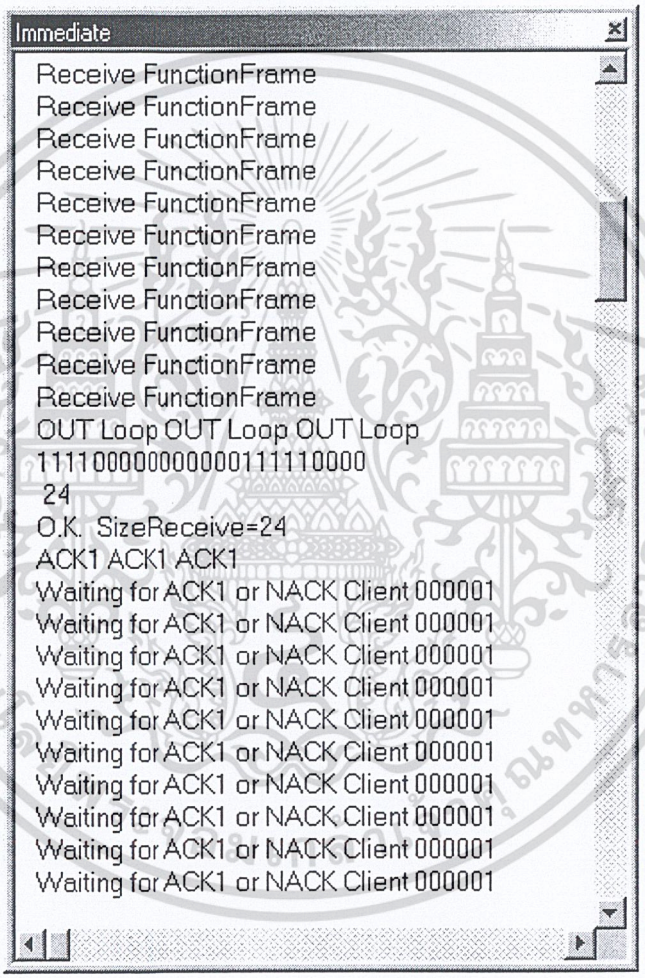
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

การทดลองและผลการทดลอง

4.1 ผลการรันโปรแกรมทางด้านเครื่อง Server

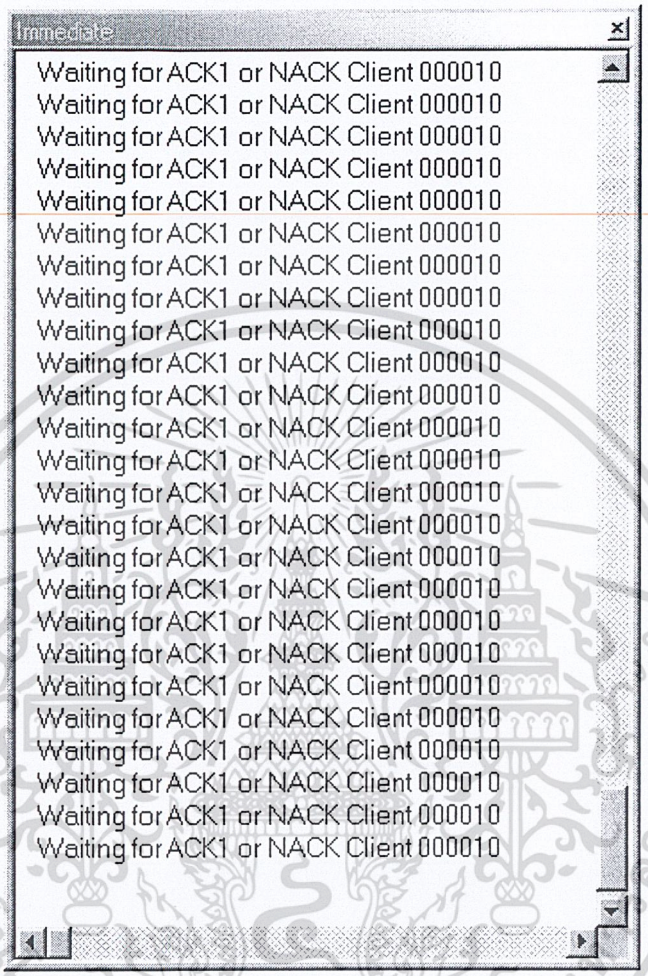
เมื่อเราเริ่มรันโปรแกรม โปรแกรมจะทำการส่งสัญญาณ Poll ไปยังเครื่องลูกข่ายซึ่งจะเรียงลำดับจากเครื่องที่ 1, เครื่องที่ 2, เครื่องที่ 3 , ... ดังรูปที่ 4.1



รูปที่ 4.1 Server ทำการส่งสัญญาณ Poll ไปยังเครื่องลูกข่ายที่ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

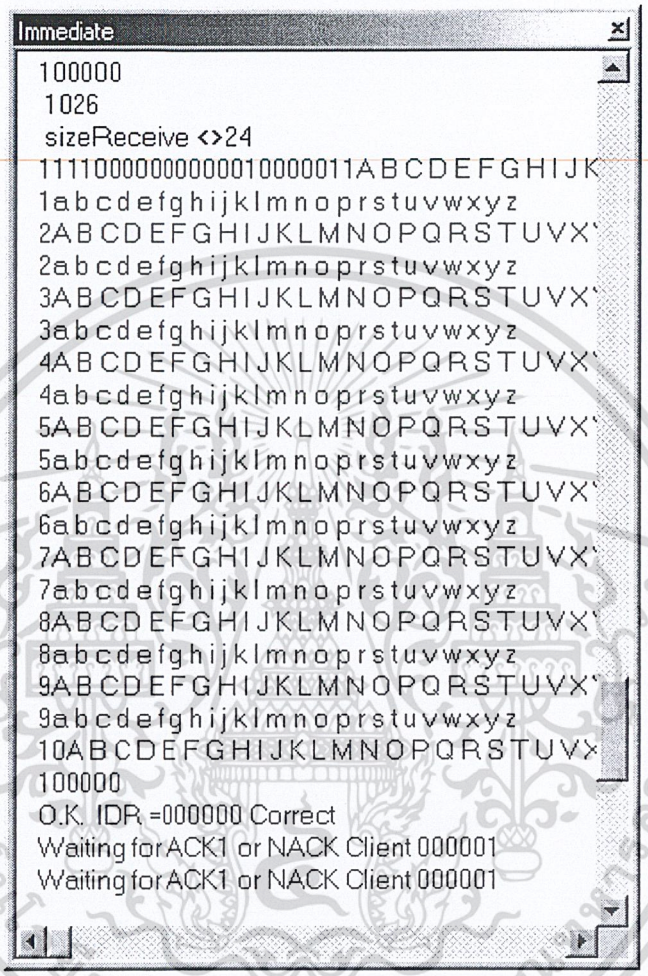
หลังจากนั้น Server จะรอรับสัญญาณ ACK1 จากเครื่องลูกข่าย หลังจากนั้นจะทำการ Poll ไปยังเครื่องลูกข่ายถัดไปดังรูปที่ 4.2



รูปที่ 4.2 Server ทำการ Poll ไปยังเครื่องถัดไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

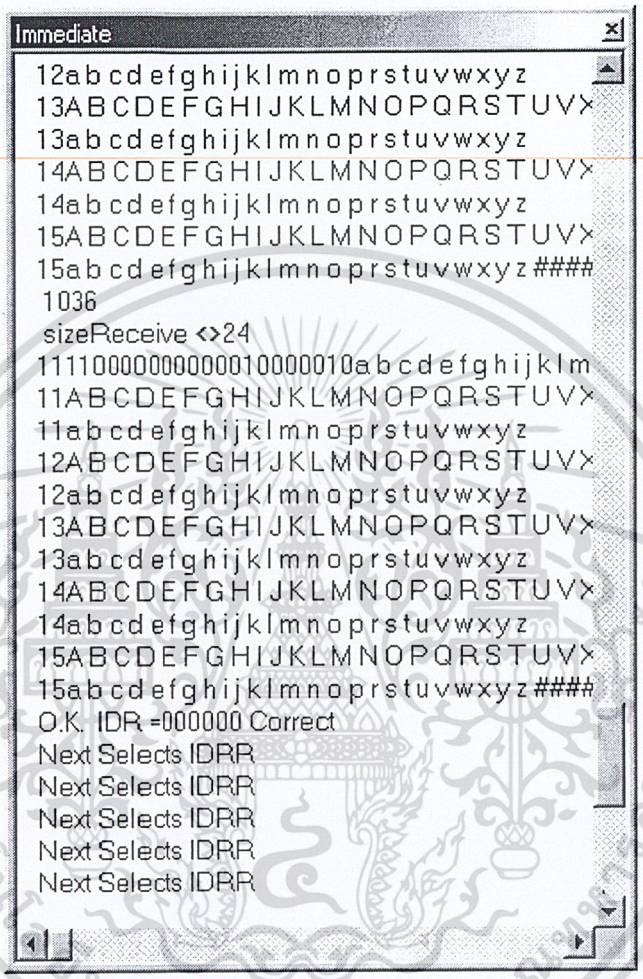
ถ้าเราสมมติว่าเครื่องลูกข่าย 1 จะส่งข้อมูลที่มีขนาด 1500 Byte ไปยังเครื่องลูกข่าย 2 เครื่องลูกข่าย 1 หลังจากที่ได้รับสัญญาณ Poll แล้วจะต้องตอบสัญญาณ ACK1 ไปบอกให้ Server ทราบหลังจากนั้นเครื่องลูกข่ายจะส่งข้อมูลมาให้เครื่อง Server เมื่อ Server ได้รับข้อมูลจะมีหน้าตาดังรูปที่ 4.3



รูปที่ 4.3 Server ได้รับข้อมูลจากเครื่องลูกข่าย 1 ครั้งที่ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

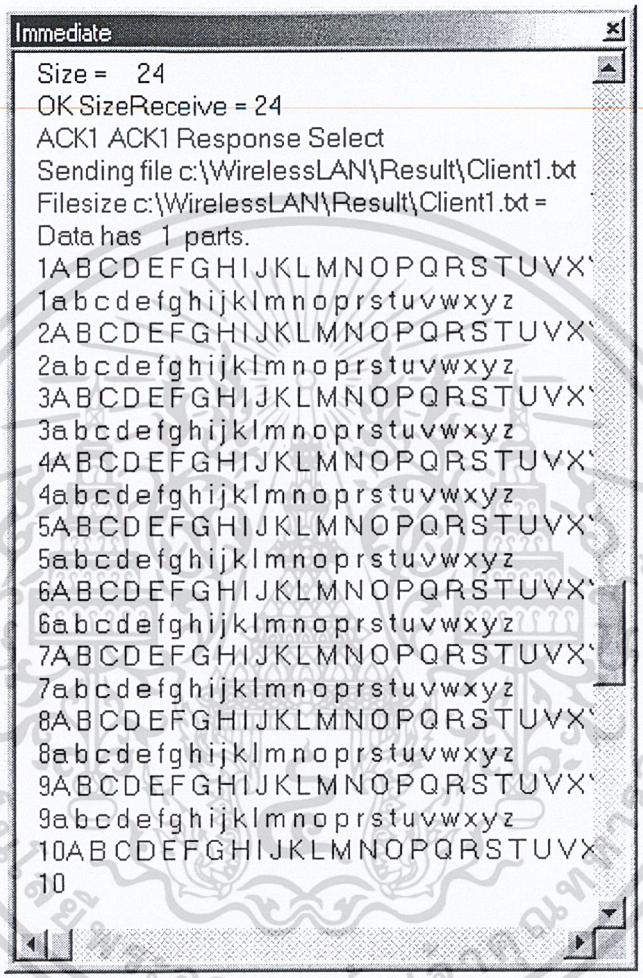
หลังจากได้รับข้อมูลชุดแรกแล้วเครื่อง Server จะทำการ Poll ไปยังเครื่องถัดไปจนกระทั่งกลับมา Poll ยังเครื่องลูกข่าย 1 อีกครั้งจึงจะได้รับสัญญาณ ACK1แล้วจึงรอรับข้อมูล ซึ่งข้อมูลครั้งที่ 2 ที่รับมาได้มีหน้าตาดังรูปที่ 4.4



รูปที่ 4.4 Server ได้รับข้อมูลจากเครื่องลูกข่าย 1 ครั้งที่ 2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

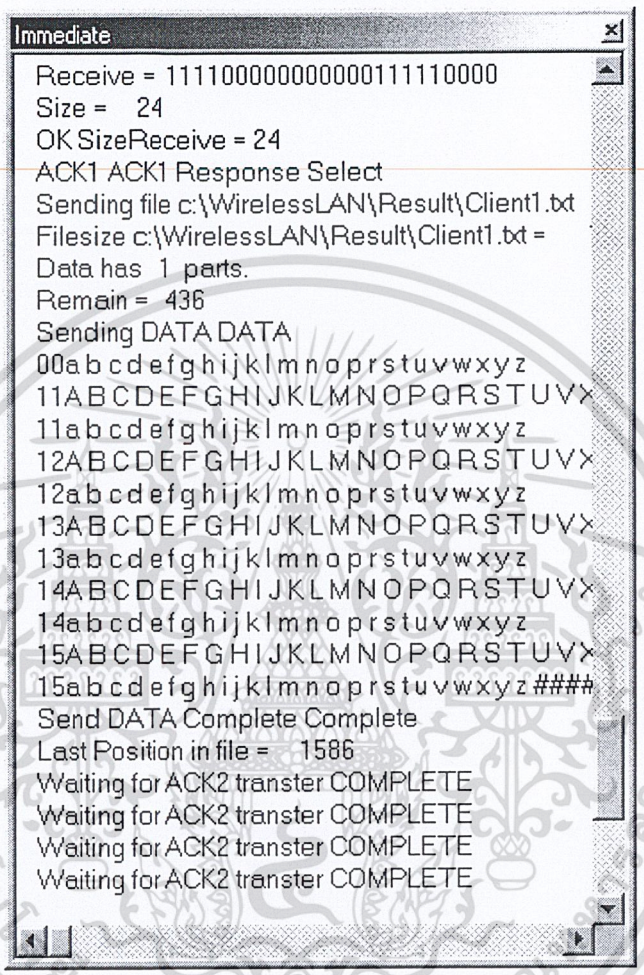
หลังจากที่ Server ได้รับข้อมูลครบแล้ว server จะทำการส่งสัญญาณ Select ไปยังเครื่องลูกข่ายที่ 3 เพื่อทำการส่งข้อมูลที่เครื่องลูกข่ายที่ 1 ส่งมาเมื่อเครื่องลูกข่าย 2 ได้รับสัญญาณ Select ก็จะส่งสัญญาณ ACK1 ตอบกลับมาหลังจากนั้น Server ก็จะทำการส่งข้อมูลชุดแรกไปยังเครื่องลูกข่าย 2 ซึ่งมีผลการส่งเป็นดังรูปที่ 4.5



รูปที่ 4.5 Server ส่งข้อมูลครั้งที่ 1 ไปยังเครื่องลูกข่าย 3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

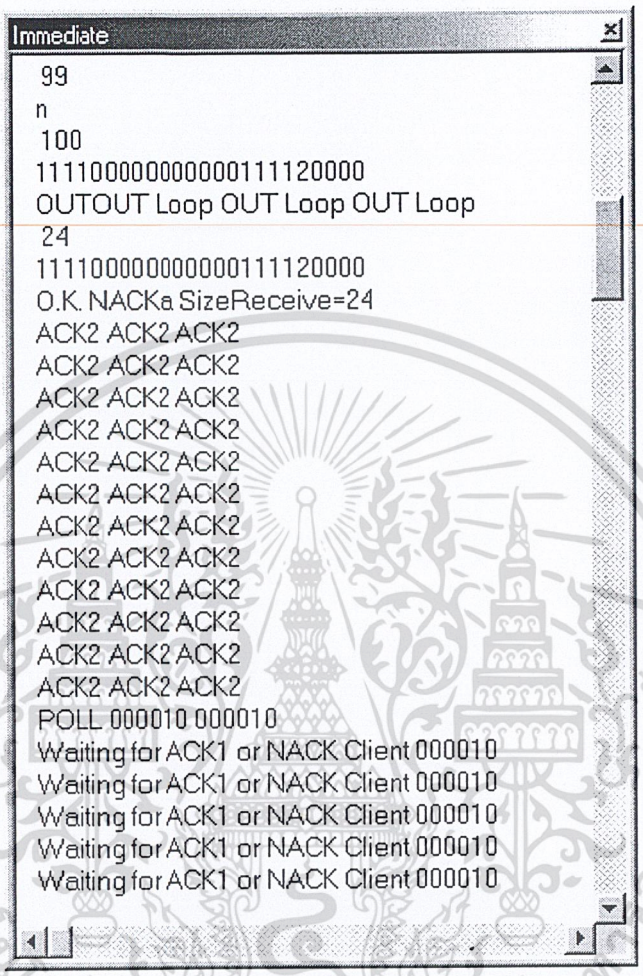
หลังจากส่งข้อมูลครั้งที่ 1 แล้ว Server จะทำการ Poll เครื่องถัดไปจนวนมาถึงเครื่องลูกข่าย 3 Server จึงส่งสัญญาณ Select อีกครั้งหนึ่งเพื่อที่จะส่งข้อมูลครั้งที่ 2 โดยผลการส่งข้อมูลครั้งที่ 2 เป็นดังรูปที่ 4.6



รูปที่ 4.6 Server ส่งข้อมูลครั้งที่ 2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หลังจากที่ Server ส่งข้อมูลครบแล้วก็จะรอสัญญาณ ACK2 จากเครื่องลูกข่าย 2 เพื่อเป็นการบอกว่าเครื่องลูกข่าย 3 ได้รับข้อมูลครบแล้ว Server จึงทำการ Poll เครื่องถัดไป

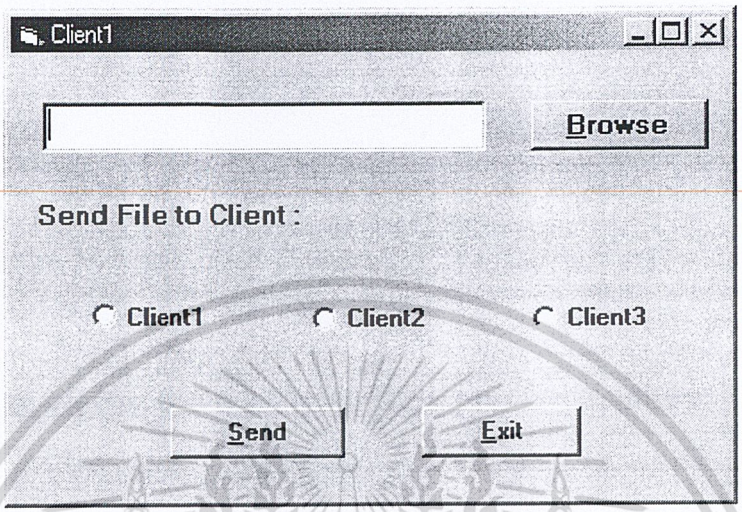


รูปที่ 4.7 Server ได้รับสัญญาณจากเครื่องลูกข่าย 3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

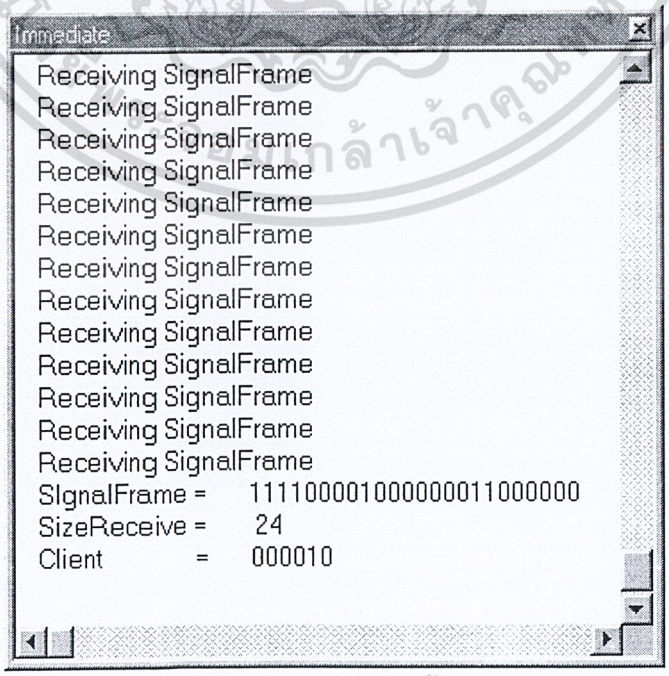
4.2 ผลการรันโปรแกรมทางด้านเครื่องลูกข่าย

เมื่อเราทำการรันโปรแกรมจะได้โปรแกรมดังรูปที่ 4.8



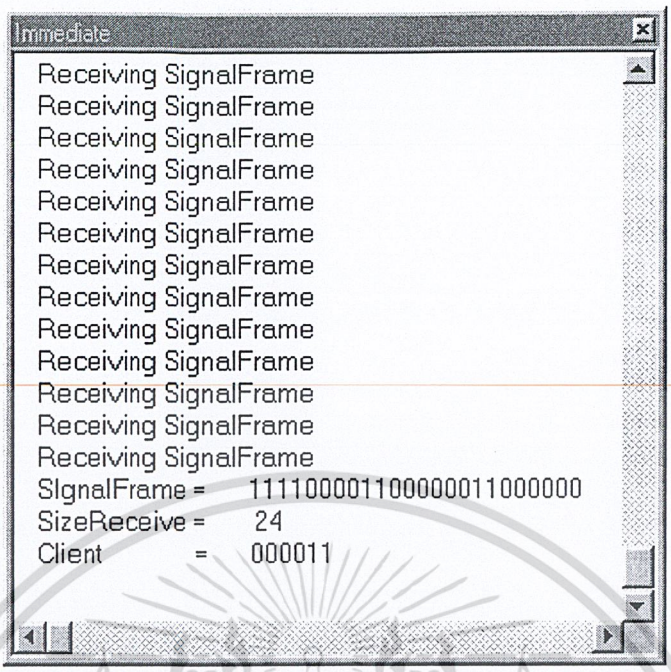
รูปที่ 4.8 โปรแกรมของเครื่องลูกข่าย

เมื่อทำการรันโปรแกรม เครื่องลูกข่ายทุกเครื่องในระบบจะได้รับการ Poll จากเครื่อง Server เครื่องลูกข่ายจะทำการรับสัญญาณ Poll พร้อมทั้งตรวจสอบว่าสัญญาณ Poll ที่ได้รับนั้นเป็นของตัวเองหรือไม่ โดยการตรวจสอบ IDR ในเฟรมคำสั่ง Poll ถ้าหากใช่ เครื่องลูกข่ายจะทำการประมวลผล แต่ถ้าไม่ใช่ เครื่องลูกข่ายเครื่องนั้นก็จะไม่ทำอะไร เช่น ในกรณีที่เครื่องลูกข่าย 000001 ได้รับสัญญาณ Poll ของเครื่องลูกข่าย 000010 และเครื่องลูกข่าย 000011 เครื่องลูกข่าย 000001 ก็จะไม่ทำการประมวลผล ดังรูป 4.9



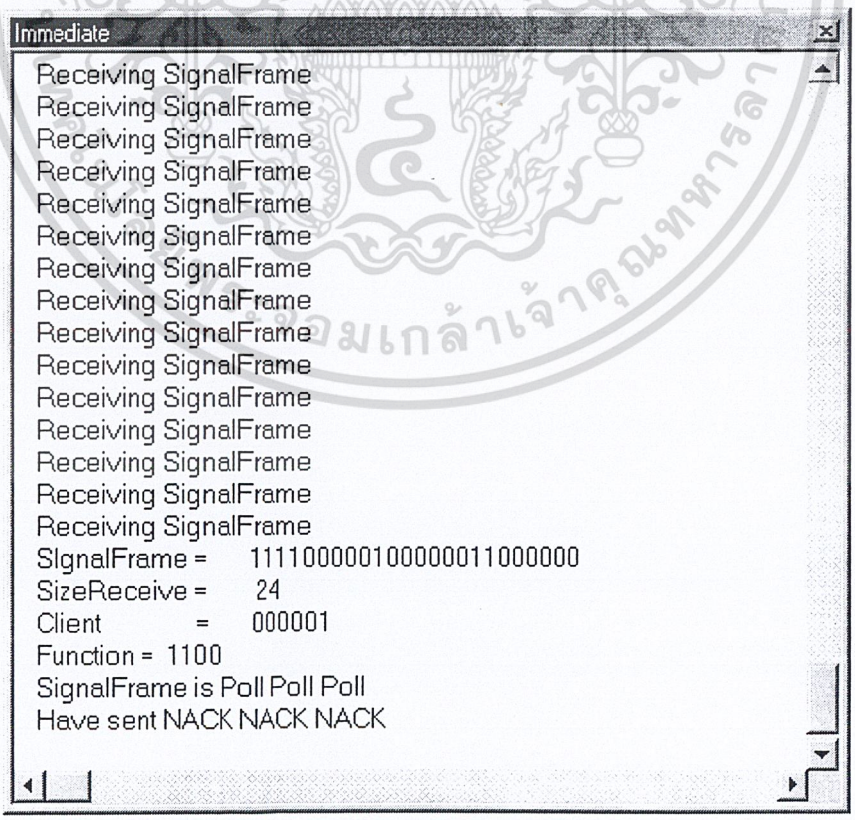
รูปที่ 4.9 เครื่องลูกข่าย 000001 รับสัญญาณ Poll จากเครื่องลูกข่าย 000010

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.10 เครื่องลูกข่าย 000001 รับสัญญาณ Poll จากเครื่องลูกข่าย 000011

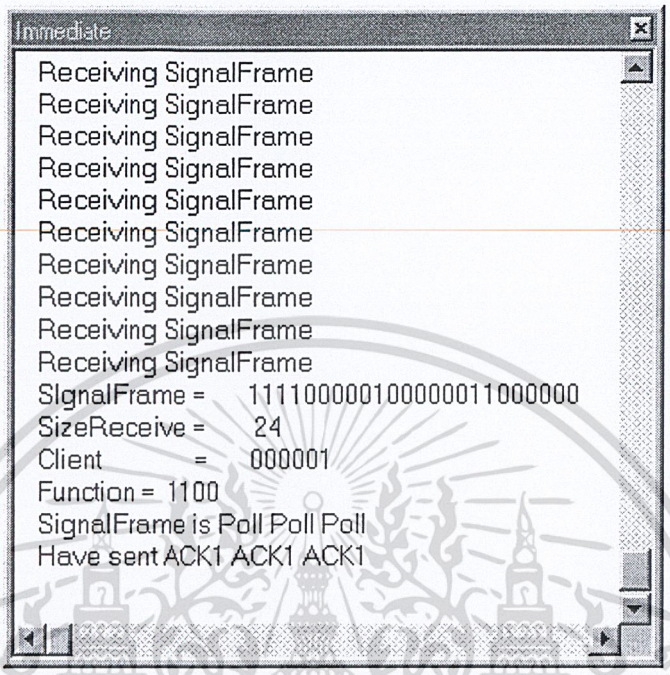
ถ้าผู้ใช้ไม่ต้องการส่งข้อมูลไปยังเครื่องลูกข่ายเครื่องอื่น เมื่อได้รับสัญญาณ Poll จากเครื่อง Server ก็จะส่งสัญญาณ NACK ไปบอกให้ Server ทราบ ดังรูป



รูปที่ 4.11 การส่งสัญญาณNACK

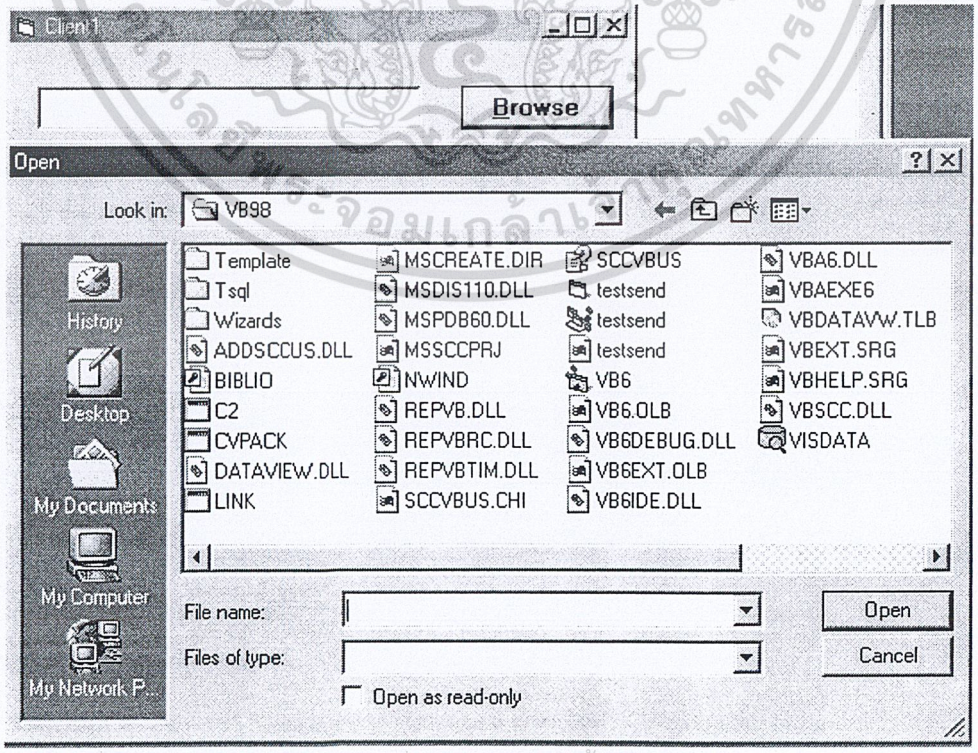
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แต่ถ้าเครื่องลูกข่ายต้องการส่งข้อมูลไปยังเครื่องลูกข่ายตัวอื่น เครื่องลูกข่ายจะส่งสัญญาณ ACK1 ไปยังเครื่อง Server



รูปที่4.12 เครื่องลูกข่ายส่งสัญญาณ ACK1 ไปเครื่อง Server

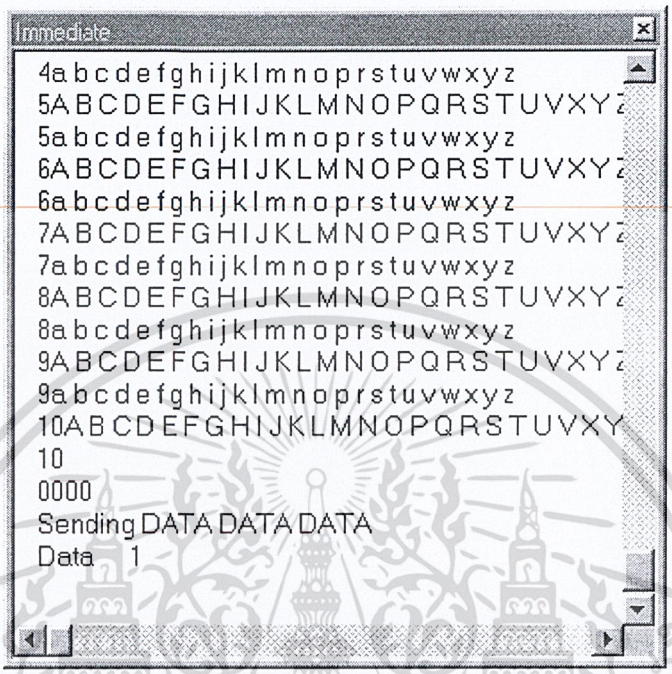
แล้วจึงทำการเลือกข้อมูลที่จะส่งมีหน้าตาดังรูปที่ 4.13



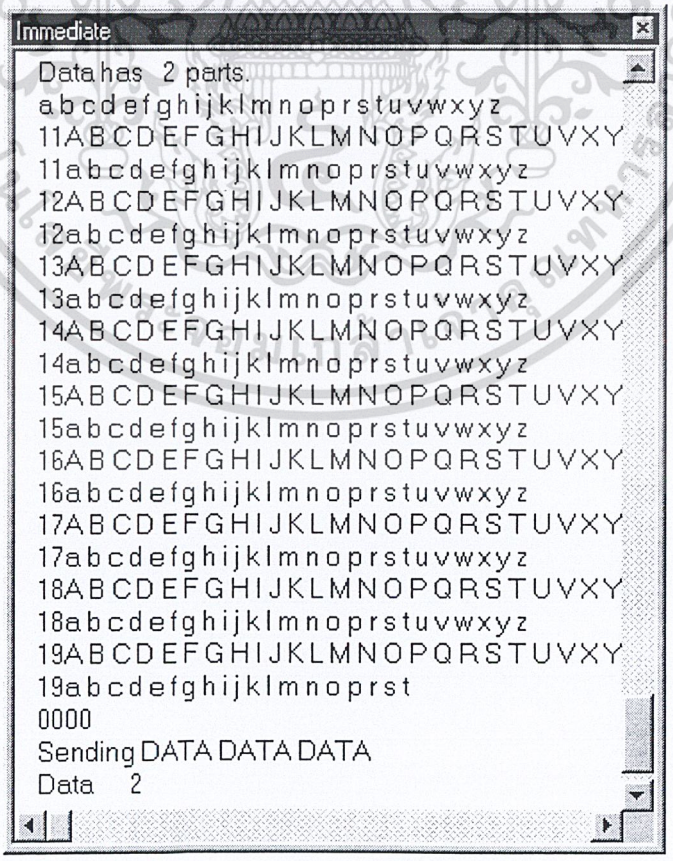
รูปที่ 4.13 ผู้ใช้เลือกไฟล์ข้อมูลที่ต้องการจะส่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในการส่งข้อมูลจากเครื่องลูกข่าย จะทำการแบ่งข้อมูลในการส่งไฟล์ ให้มีขนาดของข้อมูลครั้งละ 1,000 bytes แล้วรอสัญญาณ Poll ครั้งต่อไป จึงค่อยทำการส่งไฟล์อีก 1,000 bytes ถัดไป ทำการส่งในลักษณะนี้ไปเรื่อยๆจนหมดไฟล์ ตามรูป

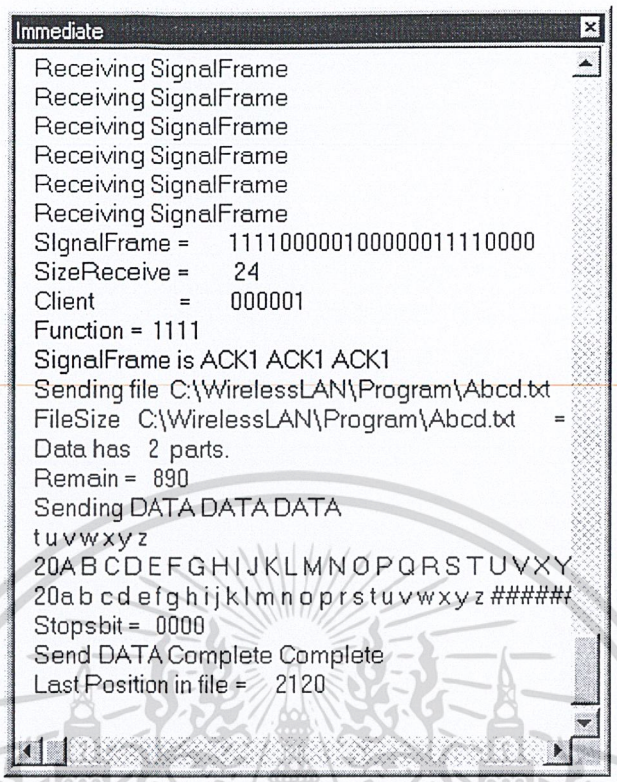


รูปที่ 4.14 การส่งข้อมูล 1,000 bytes แรก



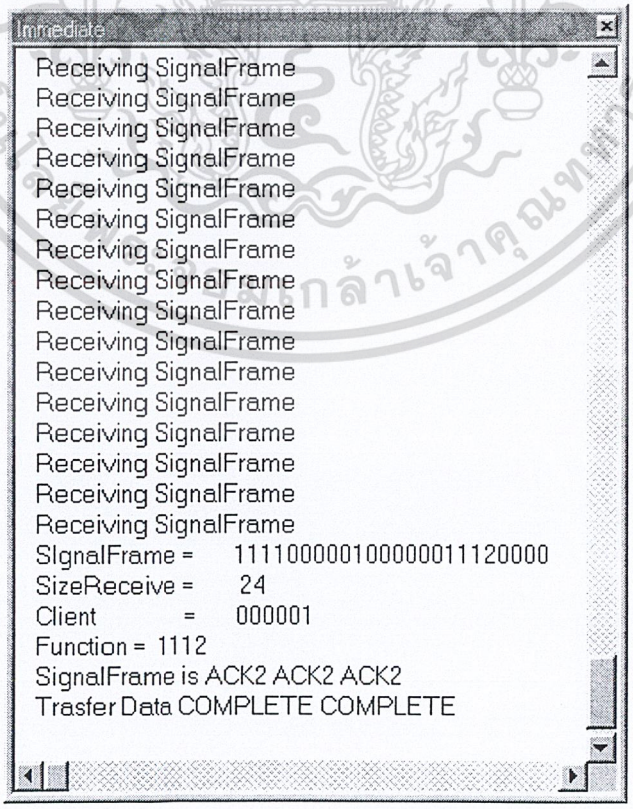
รูปที่ 4.15 การส่งข้อมูล 1,000 bytes ชุดที่2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.16 การส่งข้อมูลชุดสุดท้ายของไฟล์ข้อมูล

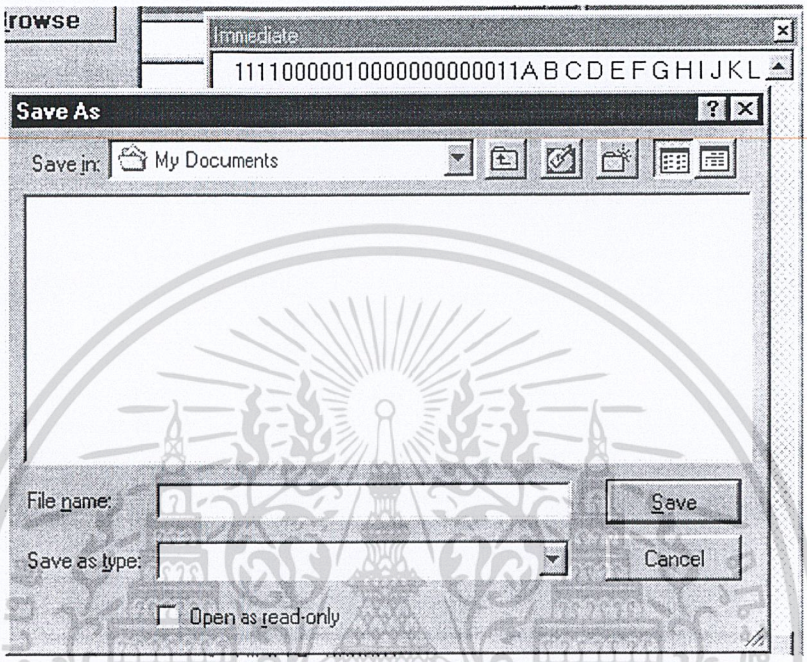
เมื่อเครื่อง Server ทำการรับข้อมูลเข้ามาจนหมด ก็จะทำการส่งสัญญาณ ACK2 เพื่อเป็นการบอกให้เครื่องลูกข่ายทราบว่าได้รับข้อมูลเข้ามาครบแล้ว ดังรูป 4.17



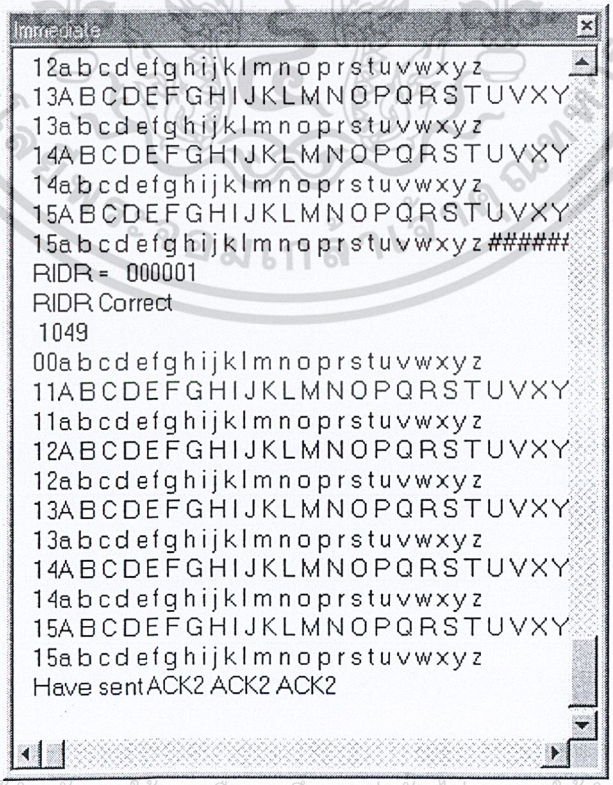
รูปที่ 4.17 เครื่อง Server ส่งสัญญาณ ACK2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เครื่อง Server จะทำการ Select เครื่องลูกข่ายที่เครื่องลูกข่ายที่ข้างต้นต้องการส่งข้อมูล พร้อมทั้งส่งข้อมูล โดยเมื่อเครื่องลูกข่ายรับข้อมูลได้หมดแล้ว เครื่องลูกข่ายจะทำการบันทึกไฟล์ข้อมูลเก็บไว้ในเครื่องคอมพิวเตอร์ แล้วทำการส่งสัญญาณ ACK2 ไปยังเครื่อง Server เพื่อบอกให้รู้ว่าได้รับข้อมูลครบหมดแล้ว ซึ่งเป็นการจบกระบวนการ การรับ-ส่งข้อมูล



รูปที่ 4.18 การบันทึกไฟล์ข้อมูล

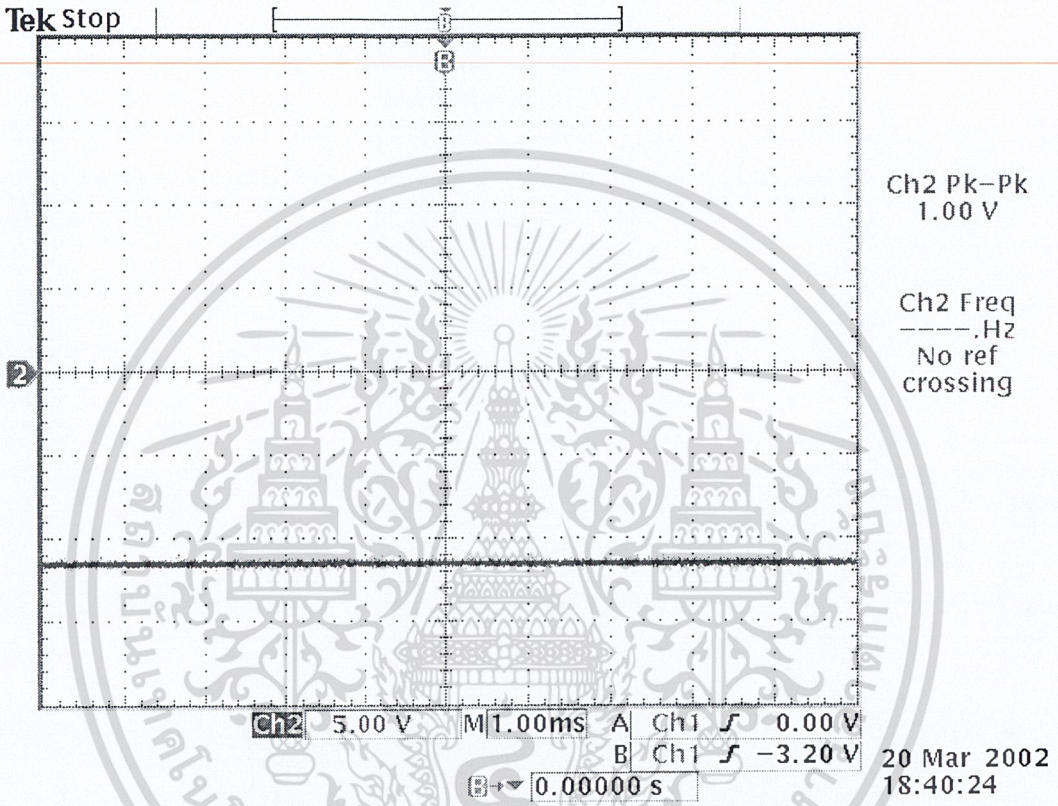


รูปที่ 4.19 เครื่องลูกข่ายส่งสัญญาณ ACK2 ไปเครื่อง Server

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3 วงจรฟรีเควนซีชิพท์คีย์อิ้ง

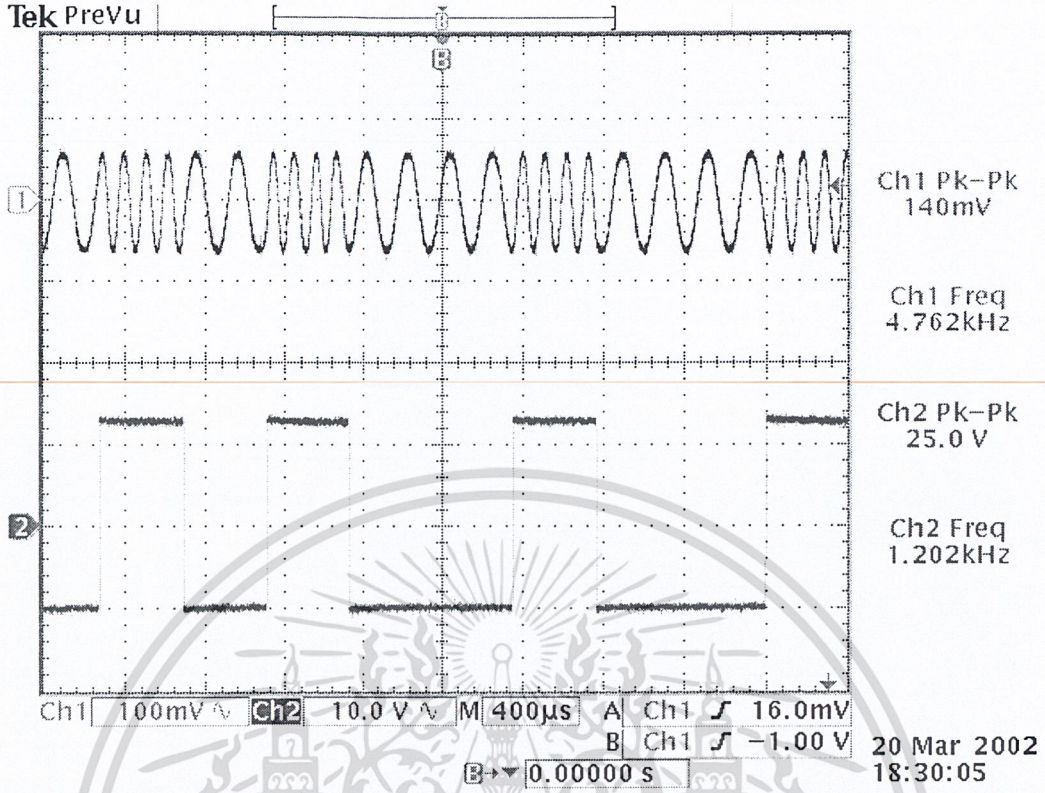
วงจรฟรีเควนซีชิพท์คีย์อิ้งทำหน้าที่แปลงสัญญาณดิจิตอลที่รับมาจากพอร์ตอนุกรมของเครื่องคอมพิวเตอร์ให้เป็นสัญญาณไซน์เฟสต่อเนื่องที่มีความถี่แตกต่างกัน 2 ความถี่ โดยเริ่มแรกเมื่อไม่มีข้อมูลส่งออกมาจากพอร์ต ระดับแรงดันจะมีค่าเป็นลบคงที่ค่าหนึ่ง ซึ่งเมื่อเทียบกับมาตรฐาน RS232 พบว่า ณ เวลานี้คอมพิวเตอร์ส่งลอจิก "1" ออกมาอย่างต่อเนื่อง หรือที่เรียกว่า Idle line นั่นเอง



รูปที่ 4.20 ระดับแรงดันของขา Tx ของพอร์ตอนุกรมเมื่อไม่มีการส่งข้อมูล

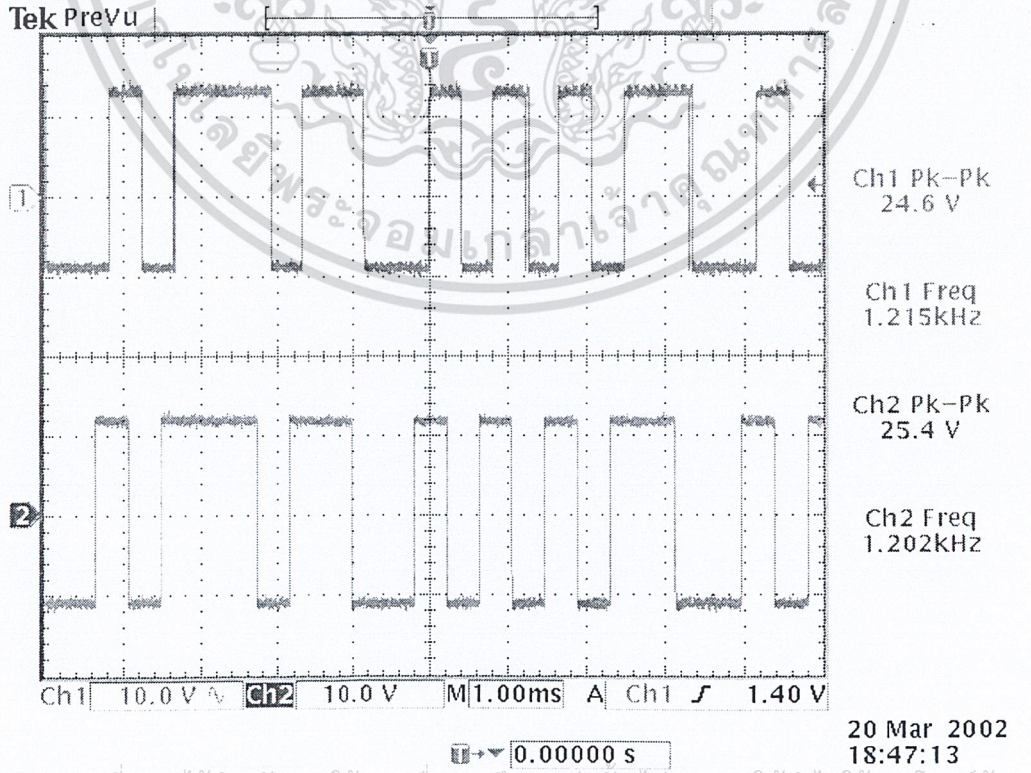
ซึ่งในเวลานี้วงจรฟรีเควนซีชิพท์คีย์อิ้งจะให้กำเนิดสัญญาณไซน์ที่มีความถี่คงที่ค่าหนึ่งออกมา เนื่องจากระดับแรงดันของ Idle line ไม่มีการเปลี่ยนแปลง เมื่อมีการส่งข้อมูลออกมา ระดับแรงดันจะมีการเปลี่ยนแปลงเป็นสัญญาณดิจิตอลที่มีระดับแรงดัน 2 ระดับ อยู่ในช่วง +3 ถึง +12 กับ –3 ถึง –12 แต่จากการทดลองพบว่าไอซี XR2206 ที่ใช้สร้างสัญญาณฟรีเควนซีชิพท์คีย์อิ้งจะให้สัญญาณที่ไม่ผิดเพี้ยนได้สูงสุดที่ประมาณ 8 โวลต์สูงสุด จึงทำการแบ่งแรงดัน (devide) ก่อน วงจรฟรีเควนซีชิพท์คีย์อิ้งจะให้กำเนิดสัญญาณไซน์ออกมา 2 ความถี่ โดยในที่นี้กำหนดให้ลอจิก "1" ให้กำเนิดความถี่ 9600 Hz และ ลอจิก "0" ให้กำเนิดความถี่ 4800 ดังแสดงในรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.21 สัญญาณฟรีเควนซีชิพท์คีย์อิ้งที่มีความถี่ 4800 Hz และ 9600 Hz

ในส่วนของด้านรับทำการทดสอบได้โดยป้อนเอาท์พุตของวงจรฟรีเควนซีชิพท์คีย์อิ้งให้กับวงจรฟรีเควนซีชิพท์คีย์อิ้งดีมอดูเลเตอร์ ผลที่ได้จะเป็นดังนี้



รูปที่ 4.22 สัญญาณที่ได้หลังจากการดีมอดูเลตฟรีเควนซีชิพท์คีย์อิ้ง

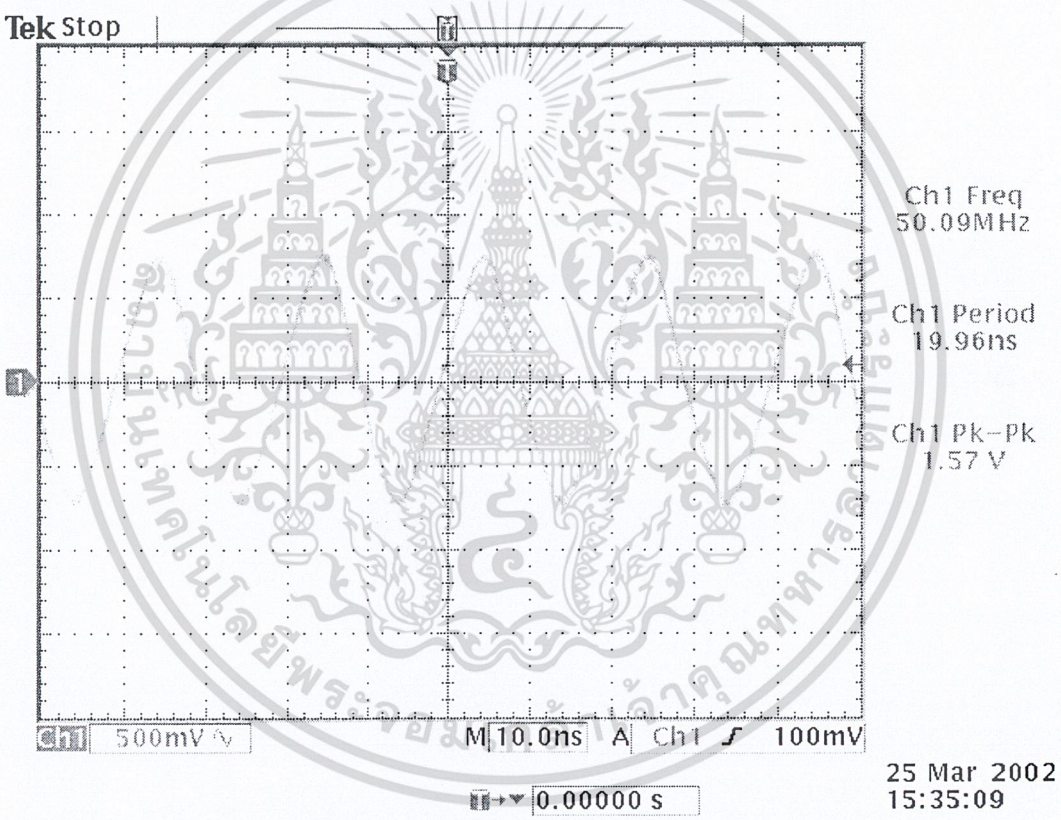
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ซึ่งได้ทำการทดสอบการทำงานโดยเขียนโปรแกรมขึ้นมาโปรแกรมหนึ่งให้ทำการส่งข้อมูลผ่านทางพอร์ตอนุกรม และโปรแกรมรับข้อมูลที่คอมพิวเตอร์อีกเครื่องหนึ่ง จากนั้นทำการส่งข้อมูลออกไปผ่านวงจรฟรีเควนซีชิพท์คีย์อิ้งซึ่งข้อมูลที่รับเข้ามาได้ก็คือข้อมูลที่ส่งมานั่นเอง จึงสรุปได้ว่าวงจรฟรีเควนซีชิพท์คีย์อิ้งสามารถทำงานได้ โดยไม่เกิดความผิดพลาด

4.4 ระบบสังเคราะห์ความถี่

4.4.1 วงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์

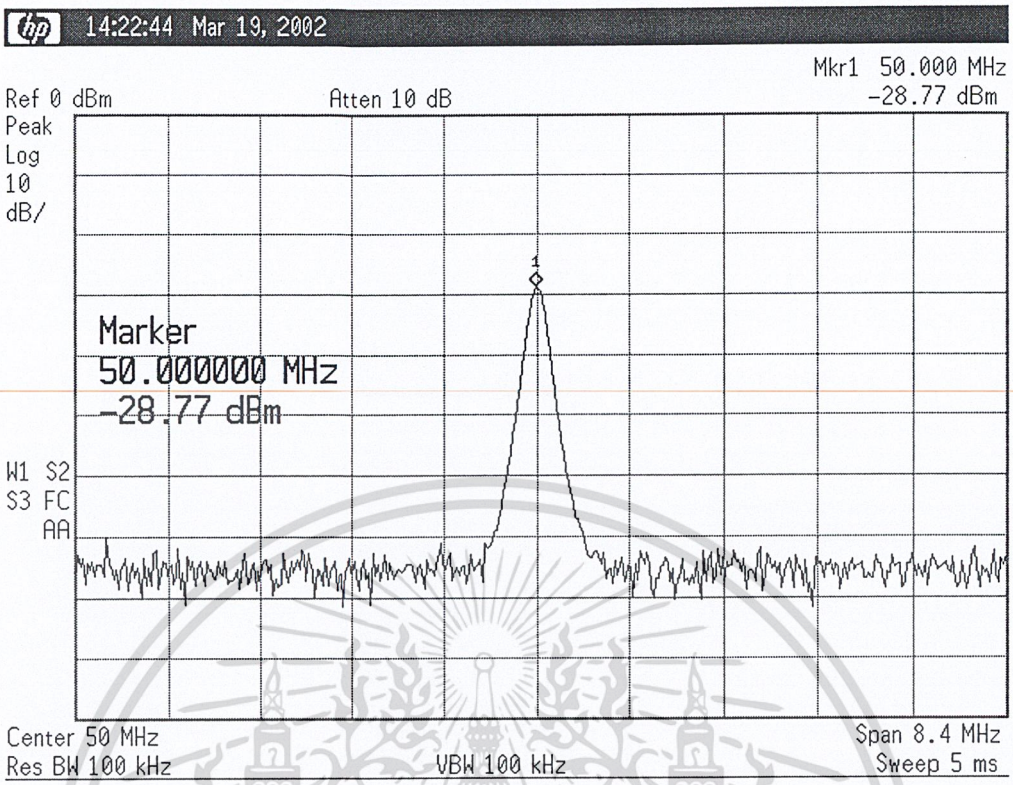
ผลการทดลองของการต่อวงจรคอลพิตต์ออสซิลเลเตอร์เป็นวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ โดยกำหนดความถี่กลางไว้ที่ 50 เมกะเฮิร์ตซ์ ได้รูปสัญญาณออกมาเป็นดังนี้



รูปที่ 4.23 สัญญาณไซน์ที่ผลิตได้ที่ความถี่กลางเท่ากับ 50 เมกะเฮิร์ตซ์

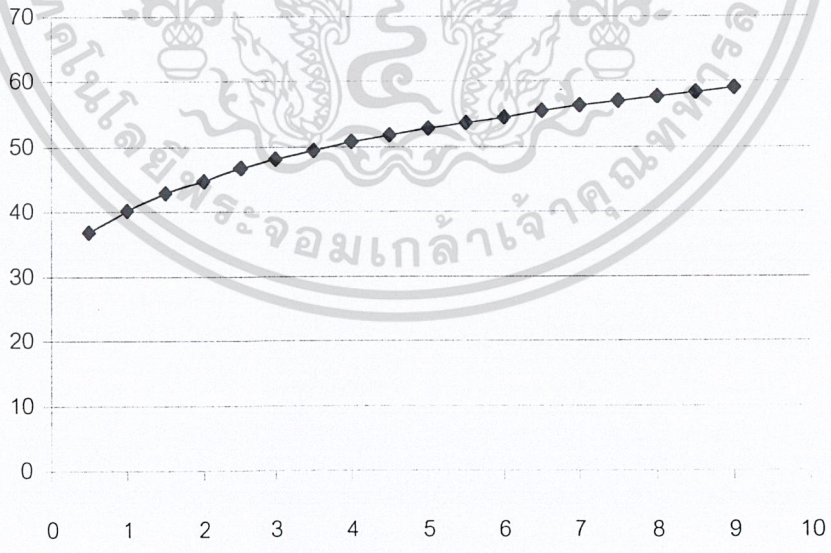
เมื่อพิจารณาในแกนความถี่ จะได้สเปกตรัมของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ที่ 50 เมกะเฮิรตซ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.24 สเปกตรัมของสัญญาณที่ผลิตได้จากวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์

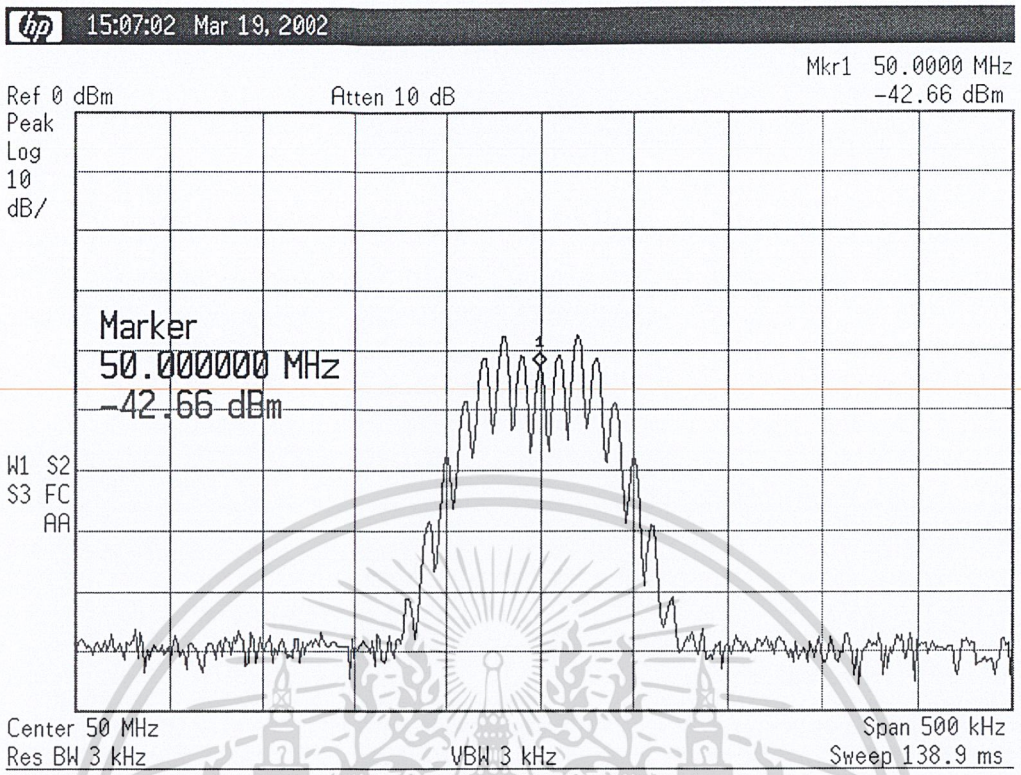
และเมื่อทำการหาค่า characteristic ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ จะได้กราฟเป็นดังรูป



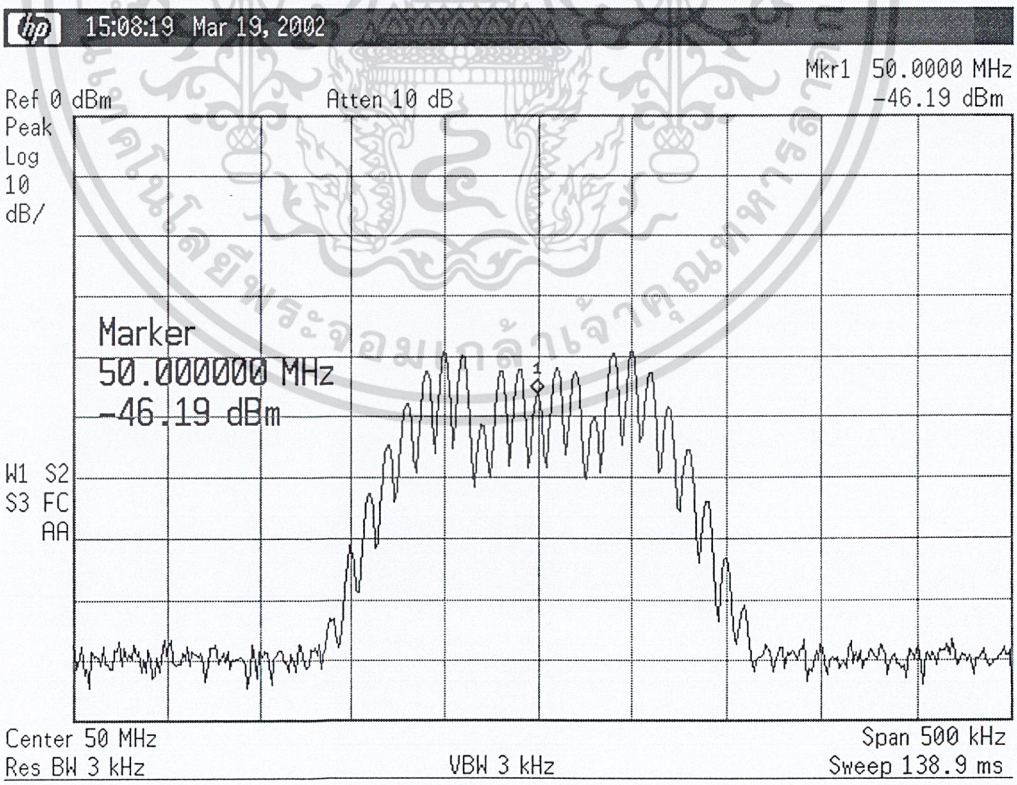
รูปที่ 4.25 Characteristic ของวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์

เมื่อทดลองป้อนสัญญาณไซน์ที่สมมติว่าเป็นสัญญาณข่าวสารที่จะมอดูเลตเข้าไป สเปกตรัมของสัญญาณจะเป็นดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

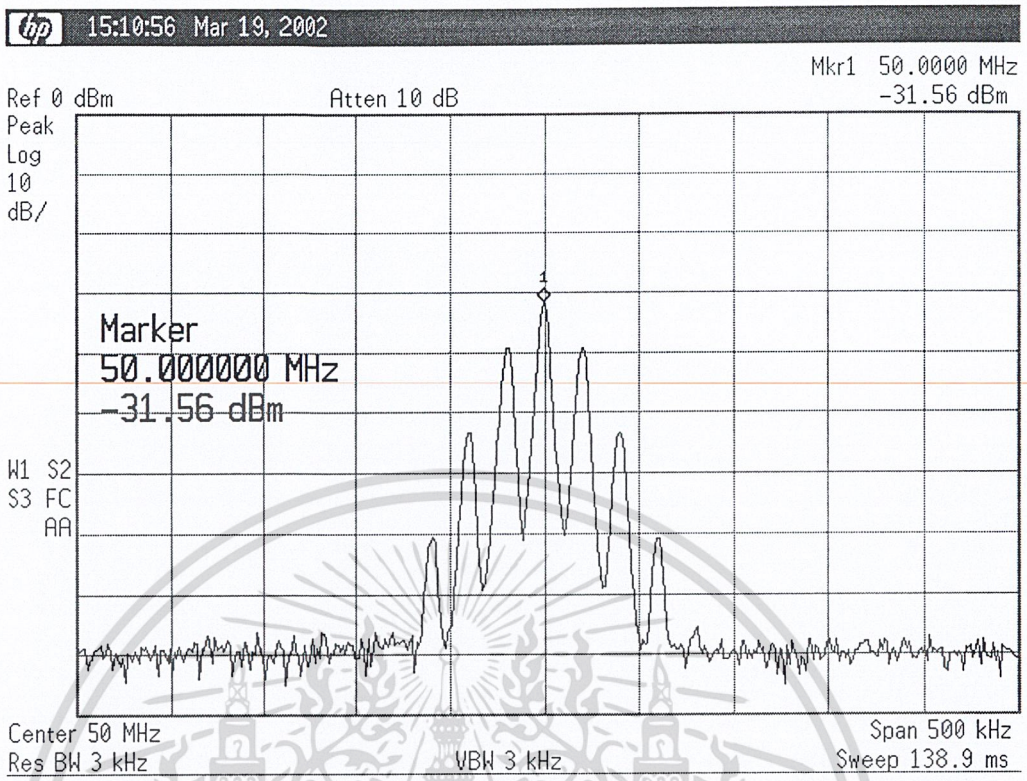


รูปที่ 4.26 สเปกตรัมของสัญญาณเอฟเอ็ม เมื่ออินพุตเป็นสัญญาณไซน์ความถี่ 10 kHz ขนาด 100 mV_{PP}



รูปที่ 4.27 สเปกตรัมของสัญญาณเอฟเอ็มเมื่ออินพุตเป็นสัญญาณไซน์ความถี่ 10 kHz ขนาด 200 mV_{PP}

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



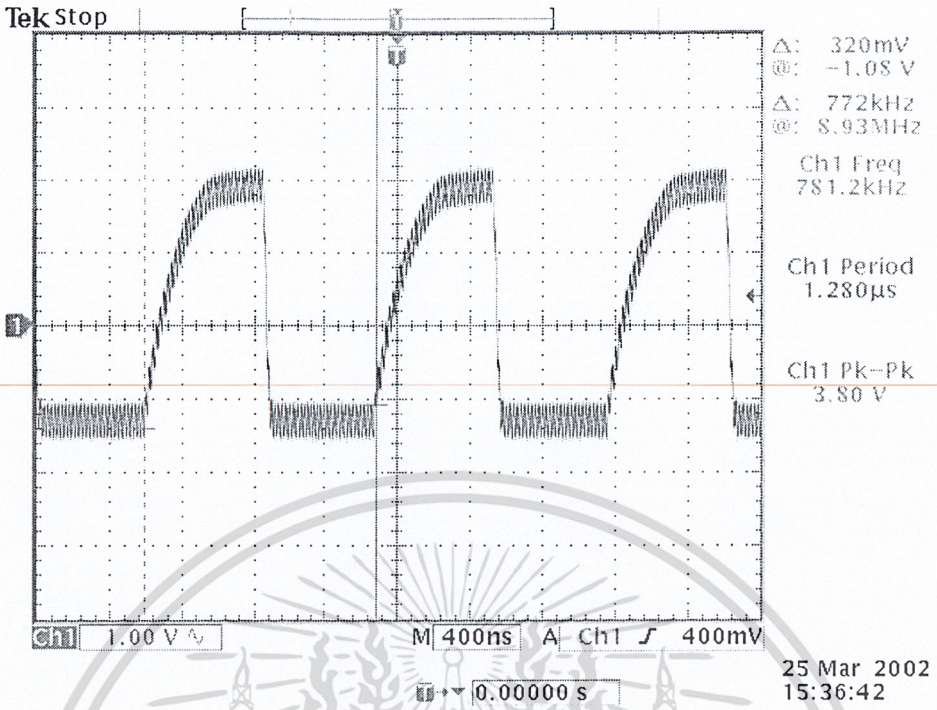
รูปที่ 4.28 สเปกตรัมของสัญญาณเอฟเอ็มเมื่ออินพุตเป็นสัญญาณไซน์ความถี่ 20 kHz ขนาด 100 mV_{PP}

จากผลการทดลองจะพบว่าเมื่อสัญญาณที่นำมามอดูเลตมีขนาดแอมปลิจูดมากขึ้น การกระจายของสเปกตรัม (span) ก็จะมีค่ามากขึ้น แบนด์วิดธ์ของสัญญาณเอฟเอ็มก็มีค่ามากขึ้น ซึ่งก็ตรงกับทฤษฎีของการมอดูเลตทางความถี่ที่ได้กล่าวมาในบทที่ 2 ส่วนเมื่อสัญญาณที่นำมามอดูเลตมีความถี่เปลี่ยนแปลงระยะห่างของแท่งสเปกตรัมก็จะมีการเปลี่ยนแปลง แต่แบนด์วิดธ์ของสัญญาณเอฟเอ็มจะไม่เปลี่ยนแปลงมากนัก ซึ่งก็ตรงกับในทฤษฎีเช่นกัน

4.4.2 วงจรพรีสเกลเลอร์แบบสองโมดูลัส

เป็นการทำงานร่วมกันไอซี MC12017 และ MC145152 โดยที่ไอซี MC12017 จะทำการหารความถี่ของวงจรเอาต์พุตจากวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์ ด้วยค่า 64/65 ก่อนเข้าไอซี MC145152 ซึ่งผลที่ได้จากการป้อนสัญญาณที่ได้จากวงจรโวลต์เตจคอนโทรลออสซิลเลเตอร์เข้าไปที่ไอซี MC12017 เป็นดังนี้

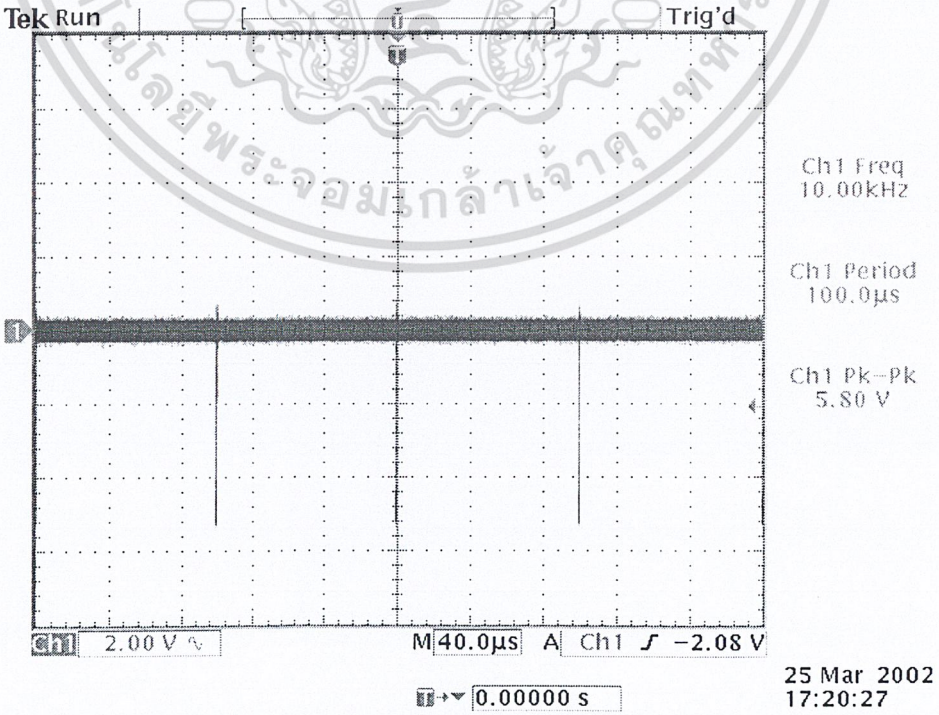
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.29 สัญญาณที่ได้จากหารความถี่ไป 64 เท่า

4.4.3 วงจรเฟสดีเท็กเตอร์และวงจรหารความถี่

ไอซี MC145152 ซึ่งเป็นวงจรสังเคราะห์ความถี่ จะตรวจสอบว่าอยู่ในสภาวะล็อกหรือไม่นั้นโดยถ้าระบบอยู่ในสภาวะ ล็อกแล้ว เฟสของสัญญาณอ้างอิง และเฟสของสัญญาณวีซีโอจะต้องตรงกัน โดยดูได้จากขาสัญญาณ LD, ϕ_R และ ϕ_V จะต้องอยู่ในสภาวะ High แต่จะมีช่วง Low ในระยะเวลาสั้น ๆ

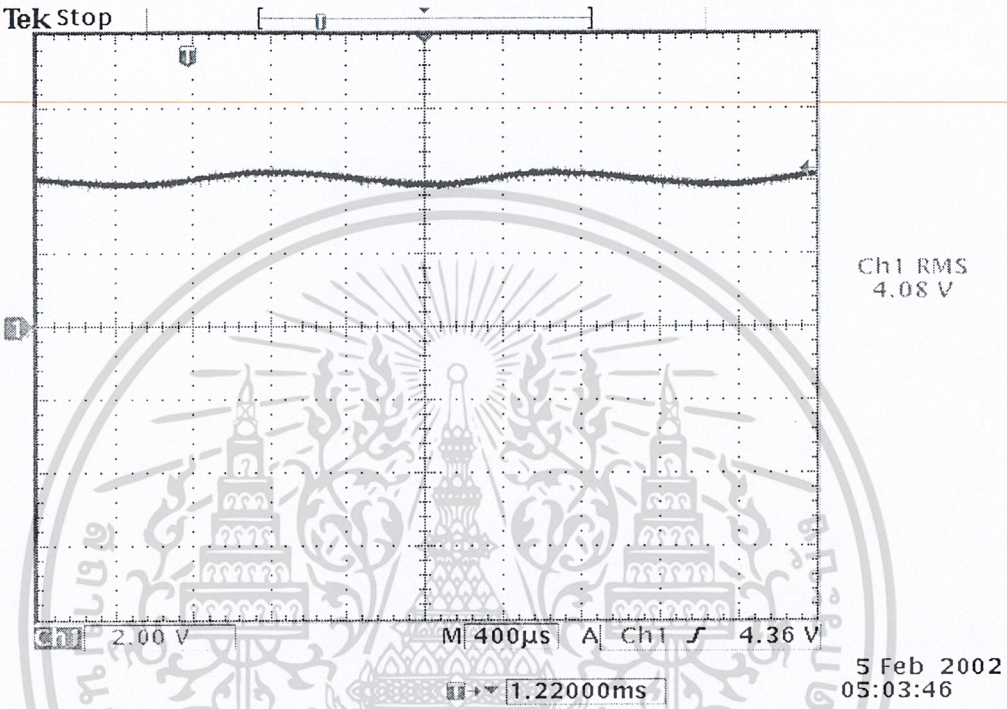


รูปที่ 4.30 สัญญาณที่ขา Lock Detect เมื่อวงจรสังเคราะห์ความถี่อยู่ในสภาวะล็อก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4.4 วงจรลูปฟิลเตอร์

วงจรลูปฟิลเตอร์จะทำหน้าที่แยกสัญญาณความถี่ต่ำออกจากสัญญาณเอาต์พุตของเฟสดีเทคเตอร์เพื่อเอาไปใช้ควบคุมออสซิเลเตอร์เพื่อคงค่าความถี่ที่เราต้องการไว้ได้ ซึ่งผลที่ได้จากการวัดขาเอาท์พุตของไอซี LM358 แสดงดังรูป

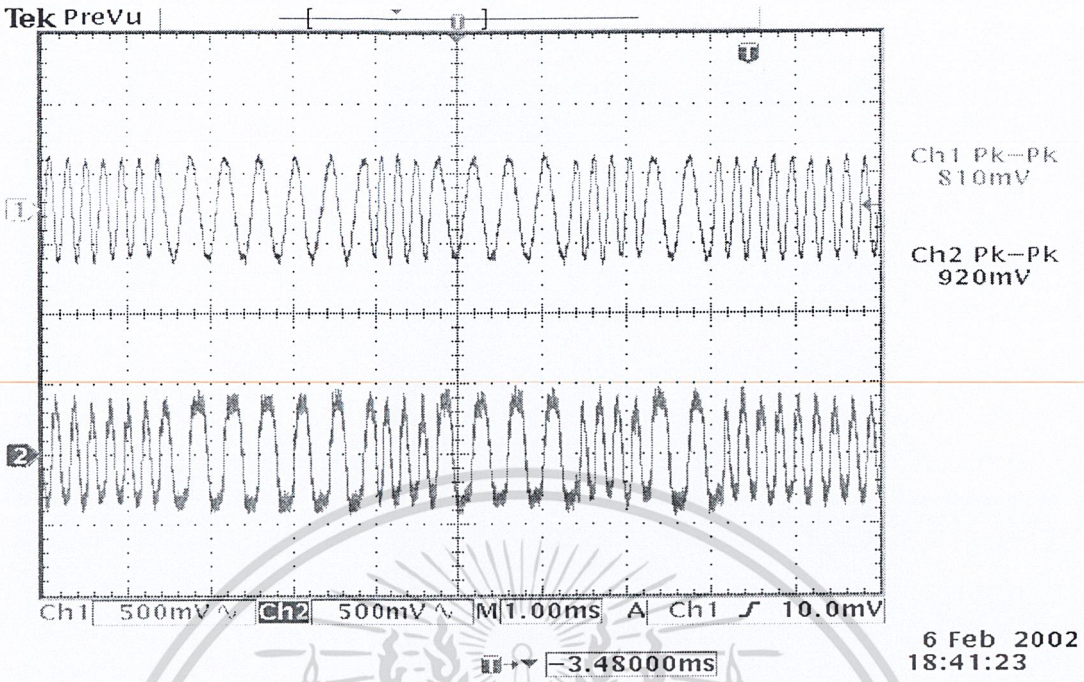


รูปที่ 4.31 ระดับแรงดันที่ออกจากลูปฟิลเตอร์

4.5 วงจรฟรีเควนซีดีมอดูเลเตอร์

วงจร MC3362 เป็นวงจรตรวจจับสัญญาณเอฟเอ็ม โดยใช้หลักการของ Double Down Convertor คือทำการลดความถี่ของสัญญาณเอฟเอ็มลงสองครั้ง คือลดลงมาที่ 10.7 MHZ และที่ 455 KHz ตามลำดับ จากนั้นก็จะทำการตรวจจับสัญญาณเอฟเอ็มที่ความถี่ 455 KHz

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.32 CH1 แสดงสัญญาณฟรีเควนซีชิพท์คีย์อิ้งที่ป้อนให้กับวงจรมอดูเลเตอร์
CH2 แสดงสัญญาณที่เอาท์พุตที่ได้จากไอซี MC3362

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
บทวิจารณ์และบทสรุป

ในโครงงานนี้ได้ทำการออกแบบระบบโครงข่ายข้อมูล โดยเริ่มแรกกำหนดให้มีจำนวนคอมพิวเตอร์ 3 เครื่อง ซึ่งแต่ละเครื่องจะมีเครื่องส่งและเครื่องรับสัญญาณวิทยุแบบไร้สาย 1 ชุด เพื่อทำการส่งสัญญาณข้อมูลและสัญญาณควบคุม แล้วออกแบบโปรแกรมรับส่งไฟล์ข้อมูลผ่านพอร์ตอนุกรมด้วยโปรแกรม Visual Basic ส่วนของเครื่องส่งและเครื่องรับจะใช้การมอดูเลตแบบฟรีเควนซีมอดูเลชันที่มีความถี่ของสัญญาณพาห์เป็น 50 เมกะเฮิรตซ์ ส่งผ่านออกอากาศไปยังเครื่องรับที่อยู่ที่คอมพิวเตอร์อีกเครื่องหนึ่ง ซึ่งผลการทดสอบปรากฏว่าสามารถส่งผ่านข้อมูลออกไปและรับกลับมาได้ และในเฉพาะส่วนของโปรแกรมซึ่งได้ทำการทดสอบผ่านสาย สามารถทำการส่งสัญญาณตรวจสอบสถานะเครื่องคอมพิวเตอร์ภายในระบบและเครื่องคอมพิวเตอร์ภายในระบบสามารถทำการส่งไฟล์ข้อมูลระหว่างเครื่อง 2 เครื่องได้

ปัญหาที่พบในการทดสอบโครงงานนี้คือ การใช้สัญญาณคลื่นวิทยุนั้นจะมีการรบกวนของความถี่ที่ไม่ต้องการซึ่งอาจทำให้ข้อมูลเกิดความผิดพลาดขึ้นได้ ส่วนข้อจำกัดของโครงงานนี้ก็คือ ระบบที่ทำการออกแบบไม่ได้ใช้โปรโตคอลมาตรฐานที่ใช้กันในระบบการสื่อสารข้อมูลทั่วไป ดังนั้นจึงไม่สามารถนำโครงงานชิ้นนี้ ไปใช้กับร่วมกับระบบโครงข่ายข้อมูลอื่นๆได้ โครงงานนี้จึงเป็นเพียงแนวทางในการศึกษาและพัฒนาเกี่ยวกับระบบการสื่อสารข้อมูลแบบไร้สายต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมทางด้านเครื่องเซิร์ฟเวอร์

```
Option Explicit
Public a As Integer, n As Integer, i As Integer, ACK2 As String, b As
Integer, Main As Integer
Public Starts As String, TIDR As String, TIDT As String, TIDRR As
String, Stops As String
Public RIDR As String, RIDT As String, RIDRR As String, iFileNum As
Integer
Public Poll As String, Selects As String, ACK1 As String, NACK As
String, z As Integer
Public Skip1 As Boolean, Skip2 As Boolean, Skip3 As Boolean
Public doing000001 As String, doing000010 As String, doing000011 As
String
Public Sub Form_Load()
MSComm1.CommPort = 2
MSComm1.PortOpen = True
MSComm1.Settings = "2400,n,8,1"
MSComm1.InputLen = 0
MSComm1.InBufferCount = 0
Dim FrameFunct(5) As String
 z = 0
 b = 1
 Main = 1
 Skip1 = False
 Skip2 = False
 Skip3 = False
 iFileNum = FreeFile
 Starts = "1111"
 Stops = "0000"
 Poll = "1100"
 Selects = "0011"
 ACK1 = "1111"
 ACK2 = "1112"
 NACK = "0000"
 TIDT = "000000" 'server
 FrameFunct(1) = Starts
 FrameFunct(3) = TIDT 'server
 FrameFunct(4) = Poll
 FrameFunct(5) = Stops
 doing000001 = "Poll"
 doing000010 = "Poll"
 doing000011 = "Poll"
 Do While Main = 1

line10:  TIDR = "000001"
 If doing000001 = "Poll" Then
  FrameFunct(2) = TIDR
  For i = 1 To 5
  MSComm1.Output = FrameFunct(i)
  Next i
  Debug.Print "POLL 000001 000001"
line5:  For b = 1 To 2000
    Debug.Print "Waiting for ACK1 or NACK Client 000001"
    If MSComm1.InBufferCount Then
     Call CheckSignal(TIDR)
     If Skip1 = True Then
      Skip1 = False
      GoTo line8
     Else
      GoTo line5
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
     End If
    End If
   Next b
 Else
  Call SendSelect(RIDRR)
 End If

line8:  TIDR = "000010"
 If doing000010 = "Poll" Then
  FrameFunct(2) = TIDR
  For i = 1 To 5
  MSComm1.Output = FrameFunct(i)
  Next i
  Debug.Print "POLL 000010 000010"
line6:  For b = 1 To 2000
    Debug.Print "Waiting for ACK1 or NACK Client 000010"
    If MSComm1.InBufferCount Then
      Call CheckSignal(TIDR)
      If Skip2 = True Then
      Skip2 = False
      GoTo line9
     Else
      GoTo line6
     End If
    End If
   Next b
 Else
  Call SendSelect(RIDRR)
 End If

line9:  TIDR = "000011"
 If doing000011 = "Poll" Then
  FrameFunct(2) = TIDR
  For i = 1 To 5
  MSComm1.Output = FrameFunct(i)
  Next i
  Debug.Print "POLL 000011 000011"
line7:  For b = 1 To 2000
    Debug.Print "Waiting for ACK1 or NACK Client 000011"
    If MSComm1.InBufferCount Then
     Call CheckSignal(TIDR)
     If Skip3 = True Then
      Skip3 = False
      GoTo line10
     Else
      GoTo line7
     End If
    End If
   Next b
 Else
  Call SendSelect(RIDRR)
 End If
 Loop
End Sub

Sub CheckSignal(TIDR As String)
Dim FunctionFrame As String, DataFrame As String, Data As String,
Receive As String, IDR As String, IDT As String, IDRR As String, Funct
As String, SizeReceive As Integer, Receive1 As String
Dim i As Integer, part As String, test As String, test1 As Integer,
zloop As Integer, Buf As String
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Dim FrameFunct(5) As String, StopSendFile As String, MaxSize As
Integer, SizetoCut As Integer
Dim cut As Integer, Data1 As String, Data2 As String, PositionSharp As
Integer
line1:
 Starts = "1111"
 Stops = "0000"
 Poll = "1100"
 Selects = "0011"
 ACK1 = "1111"
 ACK2 = "1112"
 NACK = "0000"
 TIDT = "000000" 'server
 Receive = ""
 Receive1 = ""
 n = 1
 b = 1
   Do While MSComm1.InBufferCount <> 0
     For n = 1 To 1000
     Debug.Print "Receive FunctionFrame"
     Next n
     Receive1 = MSComm1.Input
     Receive = Receive + Receive1
   Loop
   MSComm1.InBufferCount = 0
   Debug.Print "OUT Loop OUT Loop OUT Loop"
   SizeReceive = Len(Receive)
   Debug.Print Receive
   Debug.Print SizeReceive
   If SizeReceive = 24 Then
     Debug.Print "O.K. SizeReceive=24"
     FunctionFrame = Receive
     RIDR = Mid(FunctionFrame, 5, 6)
      If RIDR = "000000" Then
        Funct = Mid(FunctionFrame, 17, 4)
        Select Case Funct
         Case "1111"
          Debug.Print "ACK1 ACK1 ACK1"
          Call SendACK1
         Case "0000"
          Debug.Print "NACK NACK NACK"
          GoTo line3

        End Select
      Else
        Debug.Print RIDT
        GoTo line1
      End If
     RIDT = Mid(FunctionFrame, 11, 6)
     TIDR = RIDT

   Else
     Debug.Print " sizeReceive <>24"
     DataFrame = Receive
     Debug.Print Receive
     RIDR = Mid(DataFrame, 5, 6)
      If RIDR = "000000" Then
        Debug.Print "O.K. IDR =000000 Correct"
      Else
        Debug.Print RIDR
        GoTo line1
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        End If
      RIDT = Mid(DataFrame, 11, 6)
      RIDRR = Mid(DataFrame, 17, 6)
      n = SizeReceive - 26
      Data1 = Mid(DataFrame, 23, n)
      PositionSharp = (InStr(1, Data1, "#"))
      Select Case TIDR
          If PositionSharp <> 0 Then
        Data2 = Mid(Data1, 1, (PositionSharp - 1))
        Open "d:\WirelessLAN\Result\Client1.txt" For Binary As
#iFileNum
        Put #iFileNum, LOF(iFileNum) + 1, Data2
        Close #iFileNum
        Else
        Open "d:\WirelessLAN\Result\Client1.txt" For Binary As
#iFileNum
        Put #iFileNum, LOF(iFileNum) + 1, Data1
        Close #iFileNum
        End If
        Open "d:\WirelessLAN\Result\dataframe.txt" For Binary As
#iFileNum
        Put #iFileNum, LOF(iFileNum) + 1, DataFrame
        Close #iFileNum
        StopSendFile = Mid(DataFrame, 1017, 5)
        If StopSendFile = "#####" Then
         Call SendACK2
         Open "d:\WirelessLAN\Result\idrr1.txt" For Binary As
#iFileNum
         Put #iFileNum, , RIDRR
         Close #iFileNum
         Call SendSelect(RIDRR)
         Skip1 = True
        Else
         Open "d:\WirelessLAN\Result\idrr1.txt" For Binary As
#iFileNum
           Put #iFileNum, , RIDRR
         Close #iFileNum
         GoTo line3
        End If
      Case "000010"
        If PositionSharp <> 0 Then
        Data2 = Mid(Data1, 1, (PositionSharp - 1))
        Open "d:\WirelessLAN\Result\Client2.txt" For Binary As
#iFileNum
        Put #iFileNum, LOF(iFileNum) + 1, Data2
        Close #iFileNum
        Else
        Open "d:\WirelessLAN\Result\Client2.txt" For Binary As
#iFileNum
        Put #iFileNum, LOF(iFileNum) + 1, Data1
        Close #iFileNum
        End If
        Open "d:\WirelessLAN\Result\dataframe.txt" For Binary As
#iFileNum
        Put #iFileNum, LOF(iFileNum) + 1, DataFrame
        Close #iFileNum
        StopSendFile = Mid(DataFrame, 1017, 5)
        If StopSendFile = "#####" Then
         Call SendACK2
         Open "d:\WirelessLAN\Result\idrr2.txt" For Binary As
#iFileNum
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
          Put #iFileNum, , RIDRR
          Close #iFileNum
          Call SendSelect(RIDRR)
          Skip1 = True
         Else
          Open "d:\WirelessLAN\Result\idrr2.txt" For Binary As
#iFileNum
             Put #iFileNum, , RIDRR
          Close #iFileNum
          GoTo line3
         End If
       Case "000011"
         If PositionSharp <> 0 Then
         Data2 = Mid(Data1, 1, (PositionSharp - 1))
         Open "d:\WirelessLAN\Result\Client3.txt" For Binary As
#iFileNum
         Put #iFileNum, LOF(iFileNum) + 1, Data2
         Close #iFileNum
         Else
         Open "d:\WirelessLAN\Result\Client3.txt" For Binary As
#iFileNum
         Put #iFileNum, LOF(iFileNum) + 1, Data1
         Close #iFileNum
         End If
         Open "d:\WirelessLAN\Result\dataframe.txt" For Binary As
#iFileNum
         Put #iFileNum, LOF(iFileNum) + 1, DataFrame
         Close #iFileNum
         StopSendFile = Mid(DataFrame, 1017, 5)
         If StopSendFile = "#####" Then
          Call SendACK2
          Open "d:\WirelessLAN\Result\idrr3.txt" For Binary As
#iFileNum
          Put #iFileNum, , RIDRR
          Close #iFileNum
          Call SendSelect(RIDRR)
          Skip1 = True
         Else
          Open "d:\WirelessLAN\Result\idrr3.txt" For Binary As
#iFileNum
             Put #iFileNum, , RIDRR
          Close #iFileNum
          GoTo line3
         End If
      End Select

   End If
line3: End Sub

Sub SendACK1()
Dim FrameFunct(5) As String, i As Integer
FrameFunct(1) = Starts
FrameFunct(2) = TIDR
FrameFunct(3) = TIDT
FrameFunct(4) = ACK1
FrameFunct(5) = Stops
For i = 1 To 5
 MSComm1.Output = FrameFunct(i)
Next i
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Sub SendACK2()
Dim FrameFunct(5) As String, i As Integer, c As Integer
FrameFunct(1) = Starts
FrameFunct(2) = TIDR
FrameFunct(3) = TIDT
FrameFunct(4) = ACK2
FrameFunct(5) = Stops
For i = 1 To 5
 MSComm1.Output = FrameFunct(i)
Next i
For c = 1 To 2000
 Debug.Print "Next Selects IDRR"
Next c
End Sub
Sub SendSelect(TIDR As String)
 Dim Fsize As Integer, l As Integer, CurrentPosition As Integer,
SizeReceive As Integer
 Dim Datavar As String, Data As String, Receive As String, Receive1 As
String
 Dim FunctionFrame As String, Funct As String, Num As Integer, Remain
As Integer, Rema As Integer
 Dim c As Integer, FrameFunct(5) As String, DataFrame(6) As String
 Dim framefunction1 As String
 Dim Myfos As New FileSystemObject
 Dim Tmpfil1
 c = 1
 b = 1
 a = 1
 n = 1
 Starts = "1111"
 Stops = "0000"
 Poll = "1100"
 Selects = "0011"
 ACK1 = "1111"
 ACK2 = "1112"
 NACK = "0000"
 TIDT = "000000"
 DataFrame(1) = Starts
 DataFrame(2) = TIDR
 DataFrame(3) = TIDT
 DataFrame(6) = Stops
 FrameFunct(1) = Starts
 FrameFunct(2) = TIDR
 FrameFunct(3) = TIDT
 FrameFunct(4) = Selects
 FrameFunct(5) = Stops
  For i = 1 To 5
  MSComm1.Output = FrameFunct(i)
  Next i
  Debug.Print "Send Selects Send Selects "
  Debug.Print "MSComm1.InBufferCount1=", MSComm1.InBufferCount
  MSComm1.InBufferCount = 0
  For b = 1 To 2000
  Debug.Print "Waiting For ACK1"
   If MSComm1.InBufferCount Then
   Debug.Print "MSComm1.InBufferCount2=", MSComm1.InBufferCount
   Do While MSComm1.InBufferCount <> 0
     For n = 1 To 500
      Debug.Print "Receiving ACK1"
     Next n
     Receive = ""
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      Receive1 = " "
      Receive1 = MSComm1.Input
      Receive = Receive + Receive1
    Loop
    MSComm1.InBufferCount = 0
    Debug.Print "OUT Loop OUT Loop OUT Loop"
    SizeReceive = Len(Receive)
    Debug.Print "Receive =", Receive
    Debug.Print "Size =", SizeReceive
    If SizeReceive = 24 Then
      Debug.Print "OK SizeReceive = 24"
      FunctionFrame = Receive
      RIDR = Mid(FunctionFrame, 5, 6)
       If RIDR = "000000" Then
       Funct = Mid(FunctionFrame, 17, 4)
       RIDT = Mid(FunctionFrame, 11, 6)
       TIDR = RIDT
       TIDRR = RIDT
       Select Case Funct
        Case "1111"
         Debug.Print "ACK1 ACK1 Response Select"
          DataFrame(2) = TIDR
          DataFrame(4) = TIDRR
         Select Case TIDR
         Case "000001"
          If z = 0 Then
          Open "d:\WirelessLAN\Result\Client1.txt" For Binary As
#iFileNum
          End If
          Debug.Print "Sending file d:\WirelessLAN\Result\Client1.txt"
          Fsize = LOF(iFileNum)
          Num = Fsize \ 1000
          Debug.Print "Filesize d:\WirelessLAN\Result\Client1.txt =",
Fsize
          Debug.Print "Data has", Num, "parts."
          If z < Num Then
           Data = ""
           For CurrentPosition = (1 + (1000 * (z))) To (1000 + (1000
* (z)))
           Datavar = " "
           Get #iFileNum, CurrentPosition, Datavar
           Data = Data + Datavar
           Next CurrentPosition
           DataFrame(5) = Data
           For i = 1 To 6
           MSComm1.Output = DataFrame(i)
           Next i
           Debug.Print Data
           z = z + 1
           Debug.Print "Data", z
           doing000001 = "Select"
           For i = 1 To 2000
            Debug.Print "Next Poll Client 000010"
           Next i
           GoTo Line2

         Else
           Data = ""
           Remain = 1022 - (Fsize - (1000 * Num))
           Debug.Print "Remain =", Remain
           For CurrentPosition = (1000 * Num) To Fsize
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            Datavar = " "
            Get #iFileNum, CurrentPosition, Datavar
            Data = Data + Datavar
            Next CurrentPosition
            For Rema = 1 To Remain
            Data = Data + "#"
            Next Rema
            DataFrame(5) = Data
            Debug.Print "Sending DATA DATA"
            For i = 1 To 6
            MSComm1.Output = DataFrame(i)
            Next i
            Debug.Print Data
            Debug.Print "Send DATA Complete Complete"
            Debug.Print "Last Position in file =", CurrentPosition - 1
          Close #iFileNum
            doing000001 = "Poll"
            CurrentPosition = 0
            Set Tmpfil1 =
Myfos.GetFile("d:\WirelessLAN\Result\Client1.txt")
            Tmpfil1.Delete
          End If

        Case "000010"
          If z = 0 Then
          Open "d:\WirelessLAN\Result\Client2.txt" For Binary As
#iFileNum
          End If
          Debug.Print "Sending file d:\WirelessLAN\Result\Client2.txt"
          Fsize = LOF(iFileNum)
          Num = Fsize \ 1000
          Debug.Print "Filesize d:\WirelessLAN\Result\Client2.txt =",
Fsize
          Debug.Print "Data has", Num, "parts."
          If z < Num Then
            Data = ""
            For CurrentPosition = (1 + (1000 * (z))) To (1000 + (1000
* (z)))
            Datavar = " "
            Get #iFileNum, CurrentPosition, Datavar
            Data = Data + Datavar
            Next CurrentPosition
            DataFrame(5) = Data
            Debug.Print Data
            For i = 1 To 6
             MSComm1.Output = DataFrame(i)
            Next i
            Debug.Print "Sending DATA DATA"
            z = z + 1
            Debug.Print "Data", z
            If z + 1 <> Num Then
            doing000010 = "Select"
            GoTo Line2
            End If
          Else
            Data = ""
            Remain = 1022 - (Fsize - (1000 * Num))
            Debug.Print "Remain =", Remain
            For CurrentPosition = (1000 * Num) To Fsize
            Datavar = " "
            Get #iFileNum, CurrentPosition, Datavar
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                Data = Data + Datavar
                Next CurrentPosition
                For Rema = 1 To Remain
                Data = Data + "#"
                Next Rema
                DataFrame(5) = Data
                Debug.Print "Sending DATA DATA"
                For i = 1 To 6
                MSComm1.Output = DataFrame(i)
                Next i
                Debug.Print Data
                Debug.Print "Send DATA Complete Complete"
                Debug.Print "Last Position in file =", CurrentPosition - 1
              Close #iFileNum
                doing000010 = "Poll"
                CurrentPosition = 0
                Set Tmpfil1 =
Myfos.GetFile("d:\WirelessLAN\Result\Client2.txt")
                Tmpfil1.Delete
              End If
            Case "000011"
              If z = 0 Then
              Open "d:\WirelessLAN\Result\Client3.txt" For Binary As
#iFileNum
              End If
              Open "d:\WirelessLAN\Result\Client3.txt" For Binary As
#iFileNum
              Debug.Print "Sending file d:\WirelessLAN\Result\Client3.txt"
              Fsize = LOF(iFileNum)
              Num = Fsize \ 1000
              Debug.Print "Filesize d:\WirelessLAN\Result\Client3.txt =",
Fsize
              Debug.Print "Data has", Num, "parts."
              If z < Num Then
               Data = ""
               For CurrentPosition = (1 + (1000 * (z))) To (1000 + (1000 *
(z)))
                Datavar = " "
                Get #iFileNum, CurrentPosition, Datavar
                Data = Data + Datavar
              Next CurrentPosition
              DataFrame(5) = Data
              Debug.Print Data
              For i = 1 To 6
               MSComm1.Output = DataFrame(i)
              Next i
              Debug.Print "Sending DATA DATA"
              z = z + 1
              Debug.Print "Data", z
              If z + 1 <> Num Then
               doing000011 = "Select"
               GoTo Line2
              End If
             Else
              Data = ""
              Remain = 1022 - (Fsize - (1000 * Num))
              Debug.Print "Remain =", Remain
              For CurrentPosition = (1000 * Num) To Fsize
               Datavar = " "
               Get #iFileNum, CurrentPosition, Datavar
               Data = Data + Datavar
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            Next CurrentPosition
            For Rema = 1 To Remain
             Data = Data + "#"
            Next Rema
            DataFrame(5) = Data
            Debug.Print "Sending DATA DATA"
            For i = 1 To 6
             MSComm1.Output = DataFrame(i)
            Next i
            Debug.Print Data
            Debug.Print "Send DATA Complete Complete"
            Debug.Print "Last Position in file =", CurrentPosition - 1
           Close #iFileNum
            doing000011 = "Poll"
            CurrentPosition = 0
            Set Tmpfil1 =
Myfos.GetFile("d:\WirelessLAN\Result\Client3.txt")
            Tmpfil1.Delete
           End If
         End Select

        Case "0000"
         Debug.Print "NACK NACK NACK"
         MSComm1.Output = "Response NACK"
       End Select
       Else
       End If
    Else
      Debug.Print "OK NACKa sizeReceive <>24 Invalidity"
    End If
   For c = 1 To 2000
    Debug.Print "Waiting for ACK2 transter COMPLETE"

    If MSComm1.InBufferCount Then
    Do While MSComm1.InBufferCount <> 0
      For n = 1 To 100
      Debug.Print "n"
      Debug.Print n
      Next n
      Receive = ""
      Receive1 = " "
      Receive1 = MSComm1.Input
      Receive = Receive + Receive1
       Debug.Print Receive1
    Loop
    MSComm1.InBufferCount = 0
    Debug.Print "OUTOUT Loop OUT Loop OUT Loop"
    SizeReceive = Len(Receive)
    Debug.Print SizeReceive
    Debug.Print Receive
     If SizeReceive = 24 Then
      Debug.Print "O.K. NACKa SizeReceive=24"
      FunctionFrame = Receive
      RIDR = Mid(FunctionFrame, 5, 6)
       If RIDR = "000000" Then
       Funct = Mid(FunctionFrame, 17, 4)
       RIDT = Mid(FunctionFrame, 11, 6)
       TIDR = RIDT
       TIDRR = RIDT
       Select Case Funct
        Case "1112"
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            Debug.Print "ACK2 ACK2 ACK2"
            GoTo Line2
          Case "0000"
            Debug.Print "NACK NACK NACK"
            MSComm1.Output = "Response NACK"
        End Select
        Else
        Debug.Print "Incorrect"
        End If
    Else
      Debug.Print "OK NACKa sizeReceive <>24 Invalidity"
    End If
    End If
   Next c
   End If
  Next b

Line2:
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมทางด้านเครื่องลูกข่าย

```
Option Explicit
Public a As Integer, n As Integer, ACK1 As String, NACK As String,
ACK2 As String
Public Starts As String, Stops As String, Poll As String, Selects As
String
Public i As Integer, IDR As String, IDT As String, FileName As
String, FileSave As String, Bug As Boolean
Public b As Integer, Send As Boolean, IDRR As String, iFileNum As
Integer, Num As Integer

Private Sub Command1_Click()
Send = True
End Sub

Private Sub Command2_Click()
End
End Sub

Private Sub Command3_Click()
browse.ShowOpen
FileName = browse.FileName
Text1.Text = FileName
End Sub

Public Sub Form_Load()
MSComm1.CommPort = 1
MSComm1.PortOpen = True
MSComm1.Settings = "2400,n,8,1"
MSComm1.InputLen = 0
Dim FrameFunct(5) As String
iFileNum = FreeFile
 n = 0
 a = 1
 b = 0
 Starts = "1111"
 Stops = "0000"
 Poll = "1100"
 Selects = "0011"
 ACK1 = "1111"
 ACK2 = "1112"
 NACK = "0000"
 IDT = "000001"
 IDR = "00000"
 Send = False
 Bug = True
 End Sub

Private Sub Timer1_Timer()
Dim FunctionFrame As String, DataFrame As String, Funct As String,
Data As String, Receive As String
Dim SizeReceive As Integer, Receive1 As String, RIDR As String, RIDT
As String, RIDRR As String
Dim i As Integer, n As Integer, test As String, test1 As Integer,
Currentposition As Integer
 Starts = "1111"
 IDR = "000000" 'Server Only
 IDT = "000010" 'Clent1
 Stops = "0000"
 Poll = "1100"
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 Selects = "0011"
 ACK1 = "1111"
 NACK = "0000"

line1:
 Receive = ""
 Receive1 = ""
 If MSComm1.InBufferCount <> 0 Then
   Do While MSComm1.InBufferCount <> 0
     MSComm1.InputLen = 0
     Receive1 = MSComm1.Input
     Receive = Receive + Receive1
     For n = 1 To 800
     Debug.Print "Receiving SignalFrame"
     Next n
   Loop
   MSComm1.InBufferCount = 0
   Debug.Print "SIgnalFrame =", Receive
   SizeReceive = Len(Receive)
   Debug.Print "SizeReceive =", SizeReceive
   If SizeReceive = 24 Then
    FunctionFrame = Receive
     'Open "d:\WirelessLAN\SaveData\Functionframe.txt" For Binary As
#iFileNum + 1
     'Put #iFileNum + 1, , FunctionFrame
     'Close #iFileNum + 1
    RIDR = Mid(FunctionFrame, 5, 6) 'Cut IDR
    Debug.Print "Client   =", RIDR
     If RIDR = IDT Then 'Check IDR -Is correct?
     Funct = Mid(FunctionFrame, 17, 4)
     Debug.Print "Function =", Funct
      Select Case Funct
       Case "1111"    'ACK1
         Debug.Print "SignalFrame is ACK1 ACK1 ACK1"
       If (Send = True) And (Bug = True) Then
        Call ResponseACK1
        Bug = False
       Else
        If b < Num Then
         If Send = True Then
          Call ResponseACK1
         End If
        Else
         If Send = True Then
          Send = False
          Call ResponseACK1
         End If
        End If
       End If
      Case "1112"    'ACK2
        Debug.Print "SignalFrame is ACK2 ACK2 ACK2 "
        Debug.Print "Trasfer Data COMPLETE COMPLETE"
      Case "0000"    'NAK
        Debug.Print "SignalFrame is NACK NACK NACK"
      Case "1100"    'Poll
        Debug.Print "SignalFrame is Poll Poll Poll"
        If Send = True Then
        Call SendACK1
        Else
        Call SendNACK
        End If
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        Case "0011"    'Select
          Debug.Print "SignalFrame is Select Select Select"
          Call SendACK1
          Debug.Print "For response Select Select"
          Call ResponseSelects
        End Select
      Else 'IDR is not correct.
        GoTo line1 'Return loop.
      End If
    RIDT = Mid(FunctionFrame, 11, 6)
  End If
 End If
End Sub
Sub ResponseACK1()
NACK = "0000"
'Dim b As Integer
Dim teststop As String
Dim i As Integer, Fsize As Integer, Datashow As String, Datavar As
String
Dim DataResponsePoll As String, Currentposition As Integer
Dim FrameData(6) As String, iFileNum As Integer, Remain As Integer,
Rema As Integer
If Option1.Value = True Then
  IDRR = "000001"
ElseIf Option2.Value = True Then
  IDRR = "000010"
ElseIf Option3.Value = True Then
  IDRR = "000011"
End If
 iFileNum = FreeFile
 FrameData(1) = Starts
 FrameData(2) = "000000" ' Client1 send to Server(000000) only.
 FrameData(3) = IDT
 '====================DataFrame(4) = IDRR
 FrameData(4) = IDRR
 '====================DataFrame(5) =Data
 FrameData(6) = Stops
 Open FileName For Binary As #iFileNum
 Debug.Print "Sending file ", FileName
 Fsize = LOF(iFileNum)
 Num = Fsize \ 1000
 Debug.Print "FileSize", FileName, " =", Fsize
 Debug.Print "Data has", Num, "parts."
  'For b = 1 To Num
  'Do While b < Num
  If b < Num Then
   Datashow = ""
    For Currentposition = (1 + (1000 * (b))) To (1000 + (1000 * (b)))
    Datavar = " "
    Get #iFileNum, Currentposition, Datavar
    Datashow = Datashow + Datavar
    Next Currentposition
     ' Datashow = Datashow + "####"
    FrameData(5) = Datashow
    Debug.Print Datashow
    For i = 1 To 6
     MSComm1.Output = FrameData(i)
    Next i
    teststop = FrameData(6)
    Debug.Print teststop
    Debug.Print "Sending DATA DATA DATA"
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      b = b + 1
      Debug.Print "Data", b
      If b <> Num Then
      GoTo line2
      End If
    Else
    'Loop
    'Next b
    Datashow = ""
    Remain = 1010 - (Fsize - (1000 * Num))
    Debug.Print "Remain =", Remain
    For Currentposition = (1000 * Num) To Fsize
     Datavar = " "
     Get #iFileNum, Currentposition, Datavar
     Datashow = Datashow + Datavar
    Next Currentposition
    For Rema = 1 To (Remain - 1)
     Datashow = Datashow + "#"
    Next Rema
     FrameData(5) = Datashow
     Debug.Print "Sending DATA DATA DATA"
    For i = 1 To 6
     MSComm1.Output = FrameData(i)
    Next i
    Debug.Print Datashow
    teststop = FrameData(6)
     Debug.Print "Stopsbit = ", teststop
     Debug.Print "Send DATA Complete Complete"
     Debug.Print "Last Position in file =", Currentposition - 1
     Currentposition = 0
    End If
  Close #1
  line2: End Sub
  Sub SendACK1()
  Dim FrameFunct(5) As String
  FrameFunct(1) = Starts
  FrameFunct(2) = IDR
  FrameFunct(3) = IDT
  FrameFunct(4) = ACK1
  FrameFunct(5) = Stops
  For i = 1 To 5
  MSComm1.Output = FrameFunct(i)
  Next i
  Debug.Print "Have sent ACK1 ACK1 ACK1"
  End Sub

  Sub SendACK2()
  Dim FrameFunct(5) As String
  FrameFunct(1) = Starts
  FrameFunct(2) = IDR
  FrameFunct(3) = IDT
  FrameFunct(4) = ACK2
  FrameFunct(5) = Stops
  For i = 1 To 5
  MSComm1.Output = FrameFunct(i)
  Next i
  Debug.Print "Have sent ACK2 ACK2 ACK2"
  End Sub

  Sub SendNACK()
  Dim FrameFunct(5) As String
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
FrameFunct(1) = Starts
FrameFunct(2) = IDR
FrameFunct(3) = IDT
FrameFunct(4) = NACK
FrameFunct(5) = Stops
For i = 1 To 5
MSComm1.Output = FrameFunct(i)
Next i
Debug.Print "Have sent NACK NACK NACK"
End Sub
Sub ResponseSelects()
Dim FunctionFrame As String, DataFrame As String, Funct As String,
Data As Variant, Receive As String
Dim SizeReceive As Integer, Receive1 As String, RIDR As String, RIDT
As String, RIDRR As String
Dim i As Integer, n As Integer, test As String, test1 As Integer, lo
As Integer, lo1 As Integer
Dim asd As Integer, StopSendFile As String, Data1 As String, Data2 As
String, PositionSharp As Integer
 Starts = "1111"
 IDR = "000000" 'Server Only
 IDT = "000010" 'Client1
 Stops = "0000"
 Poll = "1100"
 Selects = "0011"
 ACK1 = "1111"
 NACK = "0000"
 MSComm1.InBufferCount = 0
 For lo = 1 To 4000
  Debug.Print lo, "Waiting for Data as IDRR"
  If MSComm1.InBufferCount <> 0 Then
   Do While MSComm1.InBufferCount <> 0
   'Debug.Print "Selected Data Selected Data Selected Data Selected
Data"
     For n = 1 To 1000
     Debug.Print n, "Loop"
     Next n
     MSComm1.InputLen = 0
     Receive = ""
     Receive1 = " "
     Receive1 = MSComm1.Input
     Receive = Receive + Receive1
     Debug.Print Receive1
   Loop
   MSComm1.InBufferCount = 0
   SizeReceive = Len(Receive)
   Open "d:\WirelessLAN\Result\receive.txt" For Binary As #iFileNum
   Put #iFileNum, LOF(iFileNum) + 1, Receive 'Data
   'SizeReceive = LOF(iFileNum)
   Close #iFileNum
    DataFrame = Receive
    RIDR = Mid(DataFrame, 5, 6) 'Cut IDR
    Debug.Print "RIDR =", RIDR
      If RIDR = IDT Then 'Check IDR -Is correct?
        Debug.Print "RIDR Correct"
        RIDT = Mid(DataFrame, 11, 6)
        RIDRR = Mid(DataFrame, 17, 6)
        n = SizeReceive - 26
        Data1 = Mid(DataFrame, 23, n)
        PositionSharp = (InStr(1, Data1, "#"))
        If PositionSharp <> 0 Then
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            Debug.Print SizeReceive
            Data2 = Mid(Data1, 1, (PositionSharp - 1))
            Debug.Print Data2
            Open "d:\WirelessLAN\Result\selectresult.txt" For Binary As
#iFileNum
            Put #iFileNum, LOF(iFileNum) + 1, Data2 'Data
            Close #iFileNum
            Call SendACK2
            GoTo line3
          Else
            Debug.Print Data1
            Open "d:\WirelessLAN\Result\selectresult.txt" For Binary As
#iFileNum
            Put #iFileNum, LOF(iFileNum) + 1, Data1 'Data
            Close #iFileNum
            For lo1 = 1 To 500
            Debug.Print "Waiting For ACK1."
            Next lo1
            Debug.Print "Receive ACK1 ACK1"
            Debug.Print "Receive ACK1 ACK1"
            Debug.Print "Receive ACK1 ACK1"
            GoTo line3
          End If
        Else 'IDR is not correct.
          Debug.Print "RIDR Not Correct"
          'GoTo line1 Return loop.
          Debug.Print "RIDR Not Correct"
          Debug.Print "RIDR Not Correct"
          Debug.Print "RIDR Not Correct"
          Debug.Print "RIDR Not Correct"
          Debug.Print "RIDR Not Correct"
          GoTo line3
        End If

    End If

Next lo
line3: End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

กฤษดา ใจเย็น, ชัยวัฒน์ ลิ้มพรจิตรวิไล, อรรถพล บุญยะโภคา. เรียนรู้และปฏิบัติการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอกผ่านพอร์ตอนุกรม PC-Serial port Interfacing text-lab manual. บริษัท อินโนเวตีฟ เอ็กเพอริเมนต์ จำกัด

จิรศักดิ์ เหลืองอุไร, ผู้แปล. ระบบเครือข่าย LAN สำหรับผู้เริ่มต้น. บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน), 2538

ประสิทธิ์ ทีฆพุฒิ. การสื่อสารโทรคมนาคม (Telecommunication) ภาคพื้นฐาน. บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน), 2539

ปราโมทย์ วาดเขียน, วิวัฒน์ กิรานนท์. พื้นฐานการสื่อสารข้อมูล. กรุงเทพฯ : คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2536

ฟ้าฟื้น เบญจดล. พื้นฐานโทรคมนาคม. ปทุมธานี : สำนักพิมพ์มหาวิทยาลัยรังสิต

รัชนัย อินทุใส. การสื่อสารโทรคมนาคม. หจก.สำนักพิมพ์ฟิสิกส์เซ็นเตอร์

วิวัฒน์ กิรานนท์. วิศวกรรมสื่อสาร Communication Engineering. กรุงเทพฯ : คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2540

สัลยุทธ์ สว่างวรรณ, ผู้แปล . Computer Networks: เครือข่ายคอมพิวเตอร์. บริษัทเพียร์สัน เอ็ดดูเคชั่น อินโดไชน่า จำกัด, 2542

A. Santamaria and F.J. Lopez-Hernandez. Wireless LAN Systems. 1st Ed. London : Artech House Inc., c1994

Stallings, William. Data and Computer Communications. 6th Ed. Upper Saddle River, N.J. : Prentice Hall, c2000

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้