ปีการศึกษา 2533

การสื่อสารข้อมูลโดยผ่านคลื่นความถี่วิทยุ

โดย

นาย สมนึก เพ็ชรพล 301293

นาย สมหมาย เหรียญรักวงศ์ 301301

นาย สุรกิจ เก่งระดมกิจ 301322

อาจารย์ที่ปรึกษา

อ.พลผดุง ผดุงกุล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2533

ภาควิชา อิเลคทรอนิคส์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การสื่อสารข้อมูลโดยผ่านคลื่นความถี่วิทยุ

DATA TRANSMISSION BY RADIO FREQUENCY

ผู้จัดทำ

1.นาย สมนึก เพ็ชรพล 301293

2.นาย สมหมาย เหรียญรักวงศ์ 301301

3.นาย สุรกิจ เก่งระดมกิจ 301322

.....อาจารย์ที่ปรึกษา

(อาจารย์ พลผดุง ผดุงกุล)

วพ.
ส 47
2533

028826

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

16 มี.ค. 2554

การสื่อสารข้อมูลโดยผ่านคลื่นความถี่วิทยุ

DATA TRANSMISSION BY RADIO FREQUENCY

สมนึก เพ็ชรพล 301293
สมหมาย เหรียญรักวงศ์ 301301
สุรกิจ เก่งระดมกิจ 301322
อาจารย์ที่ปรึกษา
อาจารย์ พลผดุง ผดุงกุล
ปีการศึกษา 2533

บทคัดย่อ

โครงงานนี้ทำขึ้นเพื่อใช้ในการติดต่อระหว่างระบบที่ควบคุมด้วยไมโครโปรเซสเซอร์ (microprocessor) 2 ระบบหรือมากกว่าโดยการติดต่อกันแบบอะซิงโครนัส (Asyncronus)โดยใช้ 8251 เป็นพอร์ทอนุกรม(serial part)แบบฮาร์ฟดูเพล็กซ์(half duplex)และอัตราโบด์(buad rate)เท่ากับ 9600 ใช้กับหน่วยประมวลผลกลางเบอร์ 8088 ในการติดต่อนี้ใช้ความถี่วิทยุ 300 Mhz เป็นคลื่นพาห์โดยเป็นการผสมคลื่นแบบพัลส์โคด (pulse code modulation) ประกอบไปด้วยตัวแม่ (master) จะเรียกเครื่องลูก (slave) โดยการเข้ารหัส (encoder) 8 bit ฉะนั้นจึงสามารถมีเครื่องลูกทั้งหมด 2^8 เครื่อง การติดต่อกันของแต่ละเครื่องสามารถทำได้ด้วยตัวแม่ (master)จะเรียกเครื่องลูก (slave) โดยส่งรหัสของตัวลูกมอดูเลท (modulate) ไปกับความถี่วิทยุผ่านความถี่ประมาณ 300 Mhz เมื่อตัวลูกรับสัญญาณก็จะทำการดีมอดูเลท (demodulate) แล้วนำรหัสที่ได้มาเปรียบเทียบกับรหัสของตัวเองที่เราตั้งไว้ที่ตัวถอดรหัสถ้าตรงกันจะมีเอาท์พุท (output) ออกไปทำการอินเตอร์รัพ (interrupt) หน่วยประมวลผลกลาง (CPU) ของระบบตัวลูกเครื่องนั้น แล้วตัวลูกก็จะส่งสัญญาณตอบรับการอินเทอร์รัพ กลับมาที่ตัวแม่จากนั้นตัวแม่ก็จะเริ่มติดต่อกับตัวลูกเครื่องนั้นโดยไม่เกี่ยวข้องกับเครื่องอื่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

028826

DATA TRANSMISSION BY RADIO FREQUENCY

Somnuek Petchpol 301293

Sommai Rienrakwong 301301

Surakit Kengradomkit 301322

Associate Professor Polphadung Phadungkul

1990

Abstract

This project is made for contacting between systems which are controlled by two systems or more microprofessor . The communication is asynchronous type by 8251 , serial port and communication is half duplex which has 9600 baud rate . CPU is 8088 . The frequency modulation is 300 MHz . In system is consist of master and slave . Address of slave used two states eight bits type encoder so we can use 2^8 slaves . The communication of each system working calls slave by send code and compare with their address . when slaves recieve this code , they will decode and compare with their address if it's in the same address , a decoder will send output , this output will contact with only this slave .

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

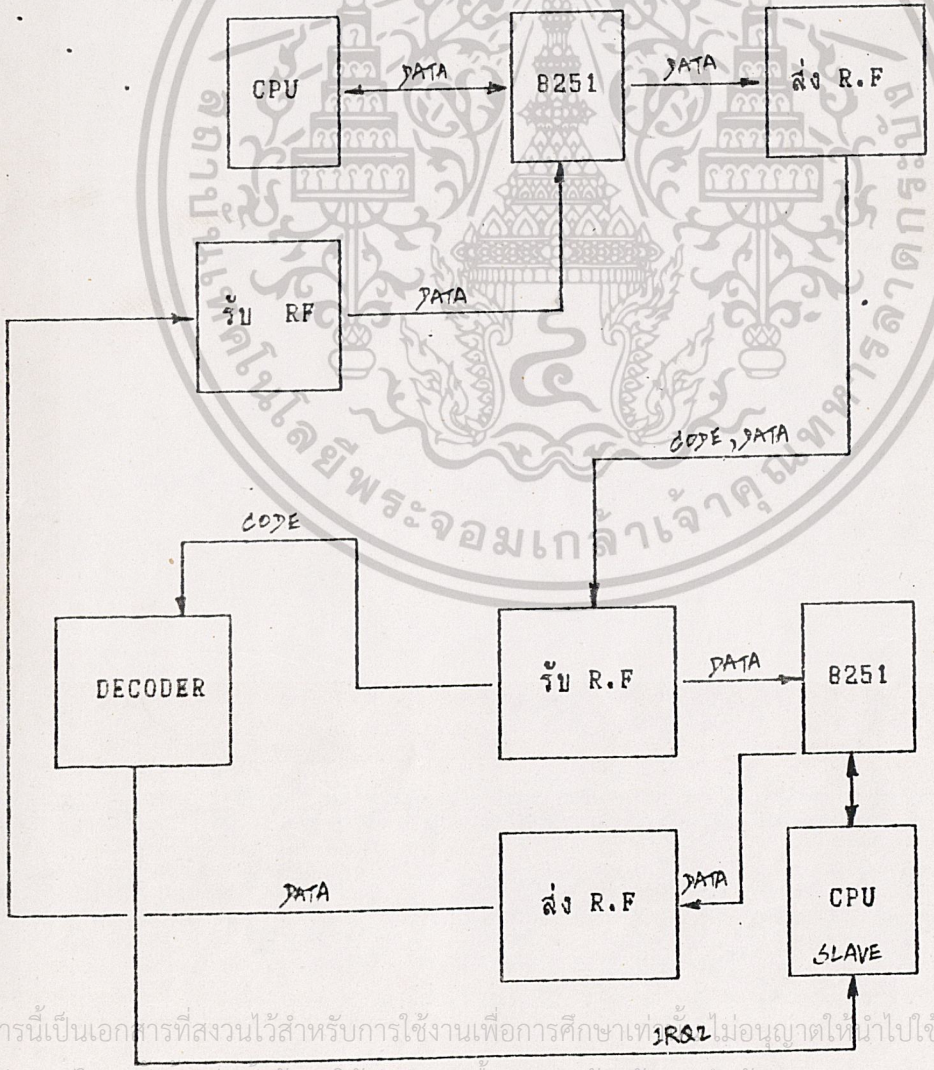
เรื่อง	หน้า
บทที่ 1 บทนำ	1
บทที่ 2 ทฤษฎีและหลักการ	3
-ฮาร์ดแวร์ที่เกี่ยวกับ 8251	5
-การอินเทอรัพท์	10
-การทำงานของวงจรส่วนภาคส่งและรับวิทยุ	13
บทที่ 3 การคำณวณและการสร้าง	15
-การทำงานของ MM53200	16
-วงจรที่ใช้ในการอินเตอร์เฟสของมาสเตอร์	18
-วงจรที่ใช้ในการอินเตอร์เฟสของสลาฟ	19
-เทคนิคในการควบคุมการรับส่งข้อมูล	20
-ลักษณะการทำงานของโปรแกรมรับและส่งข้อมูล	23
-ภาคส่งของเครื่องส่งความถี่วิทยุ	28
-ภาครับของเครื่องรับความถี่วิทยุ	39
-การพัฒนาวงจรส่วนภาครับ-ส่งวิทยุ	50
บทที่ 4 การทดลองและผลการทดลอง	76
บทที่ 5 บทที่วิจารณ์และสรุป	79
ภาคผนวก	
กิตติกรรมประกาศ	
หนังสืออ้างอิง	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1

บทนำ

ในปัจจุบันระบบการสื่อสารได้มีการพัฒนาขึ้นเป็นลำดับ โดยมีจุดเริ่มแลกจากโทรเลข วิทยุ โทรศัพท์ จนมาถึงยุคปัจจุบันที่คอมพิวเตอร์ได้เข้ามามีบทบาทสำคัญ ซึ่งมีส่วนให้ระบบการสื่อสารมีประสิทธิภาพสูงมากขึ้นซึ่งทำให้การสื่อสารมีความจำเป็น และมีความสำคัญมากขึ้นทุกขณะการพัฒนาระบบการสื่อสารข้อมูลเป็นไปด้วยความรวดเร็ว โครงงานนี้ทำขึ้นเพื่อศึกษาการสื่อสารระหว่างไมโครคอมพิวเตอร์ โดยมีเครื่องเป็นตัว master ส่วนเครื่องที่เหลือก็เป็นตัว slave ซึ่งมีลักษณะการทำงาน โดยใช้คลื่นความถี่วิทยุประมาณ 300 Mhz.



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เริ่มต้นจากการที่ CPU ของตัว MASTER ต้องการติดต่อกับเครื่องตัว SLAVE ตัวใดตัวหนึ่ง

CPU ก็จะทำการส่ง ADDRESS ของตัว SLAVE ที่ต้องการติดต่อด้วย (เลือกโดย software) ไปเข้ารหัสโดยใช้ ICเบอร์MM53200 จากนั้นก็จะนำสัญญาณจากตัวเข้ารหัสนี้ไป Modulate ก็คลื่นพาห์ที่ความถี่ถี่ 300 Mhz ส่งออกอากาศไป

ที่ตัว SLAVE ทุกตัวจะสามารถรับสัญญาณที่ส่งออกนี้ไปทำการ Demodulate แล้วนำสัญญาณนี้ไปถอดระหัสโดยใช้ IC เบอร์ MM53200 ซึ่งเราตั้งรหัสไว้ก่อนแล้วตัวถอดหัสก็จะทำการเปรียบเทียบรหัสของสัญญาณที่เข้ามากับรหัสที่ตั้งไว้ว่าตรงกันหรือไม่ ถ้าตรงกันตัวถอดรหัสก็จะส่ง output ออกมาค่าหนึ่งซึ่งเราจะนำเอา output นี้ไปใช้ในการ INTERRUPT CPU ของตัว SLAVE ตัวนั้นเมื่อ CPU ถูก INT แล้วก็จะส่งสัญญาณ CS (CHIP SELECTED)ออกไปให้ 8251 เมื่อที่จะส่งสัญญาณ ACK ตอบรับการติดต่อไปยังเครื่อง MASTER ในขณะนั้นเครื่อง SLAVE ตัวอื่นก็จะสามารถทำงานต่อไปได้โดยไม่เกี่ยวข้องกับการติดต่อนี้

การติดต่อระหว่างเครื่องทั้งสองต่อไปนี้ก็จะใช้ PROTOCOL แบบ XMODEM ซึ่งจะกล่าวรายละเอียด ในบทต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2

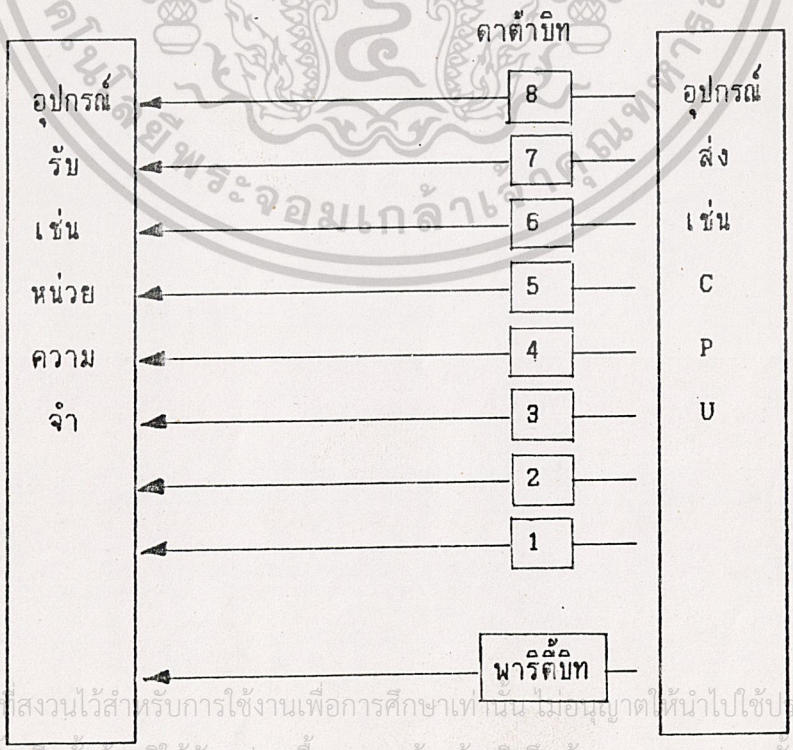
ทฤษฎี และ หลักการ

ก่อนที่เราจะกล่าวถึงการสื่อสารระหว่างเครื่อง เรามามองดูก่อนว่า สัญญาณที่ส่งออกมาจากเครื่องส่งและรับเข้าไปในเครื่องรับว่าเป็นอย่างไร

1.การถ่ายโอนข้อมูลแบบขนาน ลักษณะของการส่งข้อมูลแบบขนาน ทำได้โดยการส่งข้อมูลออกมาทีละ 1 ไบต์ คือ 8 บิท จากอุปกรณ์ส่งไปยังอุปกรณ์รับ ตัวกลางระหว่าง 2 เครื่องจะต้องมีช่องทางให้ข้อมูลเดินทางอย่างน้อย 8 ช่องทาง โดยมากจะเป็นสายขนานให้กระแสไฟฟ้าวิ่งมากกว่าจะเป็นตัวกลางชนิดอื่น เนื่องจากมีสัญญาณสูญหายไปกับความต้านทานของสาย ระยะทางระหว่าง 2 เครื่องไม่ควรเกิน 100 ฟุต ปัญหาที่เกิดขึ้นหากระยะทางของสายมากกว่านี้ก็คือ ระดับของกราวด์ในทางไฟฟ้าที่จุดรับผิดไปจากจุดส่ง ทำให้เกิดการผิดพลาดในการรับสัญญาณลอจิกทางฝ่ายรับ

นอกจากสายที่เป็นทางเดินของข้อมูลแล้วอาจจะมีทางเดินของสัญญาณควบคุมอื่นๆ เป็นต้นว่าบิตที่บอกพาริตี้ของสัญญาณ เพื่อเป็นการตรวจสอบความผิดพลาดของการรับสัญญาณที่ปลายทาง หรือสายที่ควบคุมการโต้ตอบ

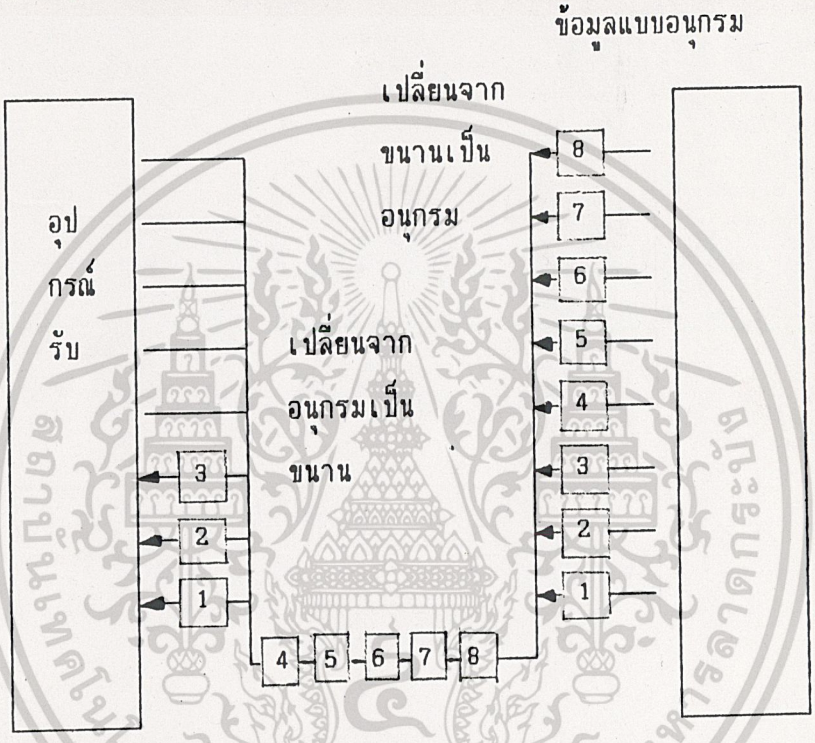
จะเห็นว่าการส่งแบบขนานส่วนมากจะทำในระยะใกล้ๆ เนื่องจากจะต้องมีช่องทางเดินของสัญญาณมากกว่า 8 สาย และอุปกรณ์ที่ติดต่อแบบขนานกับคอมพิวเตอร์ก็เห็นจะได้แก่ เครื่องพิมพ์ดังกล่าวมาแล้ว



รูปที่ 2.1 การส่งข้อมูลแบบขนาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.การถ่ายโอนข้อมูลแบบอนุกรม ในการถ่ายโอนข้อมูลแบบอนุกรม ข้อมูลถูกส่งออกมาทีละบิท ระหว่างจุดส่งและจุดรับ จะเห็นว่าการส่งข้อมูลแบบนี้จะช้ากว่าแบบขนานที่กล่าวมาแล้วแน่นอนแล้วทำไมต้องส่งแบบนี้ คำตอบก็คือ ตัวกลางการสื่อสารต้องการเพียงช่องเดียวหรือสายเพียงคู่เดียว ค่าใช้จ่ายในสื่อกลางจะต้องถูกกว่าแบบขนานอย่างแน่นอน สำหรับการส่งระยะทางไกลๆ โดยเฉพาะเมื่อเรามีระบบการสื่อสารทางโทรศัพท์ไว้ใช้งานอยู่แล้วย่อมจะเป็นการประหยัดกว่าที่จะทำการติดต่อสื่อสารทีละ 8 ช่อง เพื่อการถ่ายโอนข้อมูลแบบขนานอย่างแน่นอน



รูปที่ 2.2 การส่งข้อมูลแบบอนุกรม

รูปที่ 2.2 แสดงให้เห็นการส่งข้อมูลแบบอนุกรม ข้อมูลจากจุดส่งจะถูกเปลี่ยนให้เป็นอนุกรมเสียก่อนแล้วค่อยทยอยส่งออกทีละบิทไปยังจุดรับ ณ ที่จุดรับจะต้องมีกลไกในการเปลี่ยนข้อมูลที่ส่งมาทีละบิท ให้เป็นสัญญาณแบบขนานซึ่งลงตัวพอดีนั่นคือ บิท 1 ลงที่บัสข้อมูลเส้นที่ 1 พอดี การที่จะทำให้การแปลงสัญญาณจากอนุกรมทีละบิทให้ลงพอดีนั้นจำเป็นจะต้องมีกลไกที่เหมาะ เพื่อป้องกันการผิดพลาดในการรับ กลไกที่ว่านี้แบ่งเป็น 2 แบบ คือ

1.การส่งแบบ อะซิงโครนัส มีรูปแบบดังนี้คือ อักขระทุกๆตัวจะถูกนำด้วยบิทนำ (START BIT) โดยทั่วไปมีลอจิกเป็น "0" และปิดท้ายด้วยบิทตาม (STOP BIT) โดยทั่วไปมีลอจิกเป็น "1" ลักษณะการทำงานคือเมื่อยังไม่ส่งสัญญาณ สายส่งจะเป็น "H" คือ "1" เรียกว่า MARK เมื่อเริ่มส่งจะเป็นบิทนำคือ "0" อยู่หนึ่งบิท ต่อไปจะเป็นข้อมูลหนึ่งอักขระจะมีกี่บิทแล้วแต่จะเลือก ส่วนบิทใดจะเป็น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

"0" หรือ "1" ขึ้นอยู่กับรหัสของอักขระนั้นๆ เมื่อจบอักขระจะเป็นบิทตามคือ "1" อยู่หนึ่งหรือสองบิท หากต้องการส่งต่อไปก็จะเป็นบิทนำอีก ถ้าหยุดส่งจะเป็น MARK

2.การส่งแบบ ซิงโครนัส ไม่ใช้บิทนำและบิทตาม แต่จะส่งรหัสที่เรียกว่า SYNC CHARACTER

การติดต่อแบบอนุกรมอาจจะแบ่งตามรูปลักษณะได้ 3 แบบ คือ

1.แบบซิมเพล็กซ์ (SIMPLEX) ข้อมูลส่งได้ในทางเดียวเท่านั้น บางครั้งก็เรียกว่า การส่งทิศทางเดียว (UNDIRECTIONAL DATA BUS)

2.แบบฮาล์ฟดูเพล็กซ์ (HALF DUPLEX) ข้อมูลสามารถส่งได้ทั้งสองสถานี แต่จะต้องผลัดกันส่งผลัดกันรับ จะส่งและรับพร้อมกันไม่ได้

3.แบบฟูลดูเพล็กซ์ (FULL DUPLEX) ทั้งสองสถานีสามารถรับและส่งได้ในเวลาเดียวกัน

ในการทดลองนี้ เราใช้ 8251 2 ตัว ตัวหนึ่งเป็น MASTER อีกตัวหนึ่งเป็น SLAVE เพื่อใช้ในการสื่อสารข้อมูลอนุกรมแบบ HALF DUPLEX คือเป็นการผลัดกันรับส่งข้อมูลระหว่าง CPU 8088 2 ตัว (MASTER AND SLAVE) และเป็นการสื่อสารแบบอะซิงโครนัส รูปแบบโปรโตคอลคือ X-MODEM

สำหรับการทดลองในภาคเรียนนี้ จะให้ SLAVE รัน PROGRAM รอการ INTERRUPT จาก MASTER ก่อน แล้วจึงให้ MASTER ส่งสัญญาณ INTERRUPT (IRQ 2) ซึ่งจะได้สัญญาณนี้มาจากการถอดรหัส เพื่อเลือก ADDRESS ของเครื่อง SLAVE นี้ เมื่อ SLAVE ได้รับสัญญาณ IRQ 2 แล้ว จึงจะดำเนินการติดต่อรับส่งข้อมูลกับ MASTER ในแบบ X-MODEM โดยผ่านทางคลื่นความถี่วิทยุต่อไป

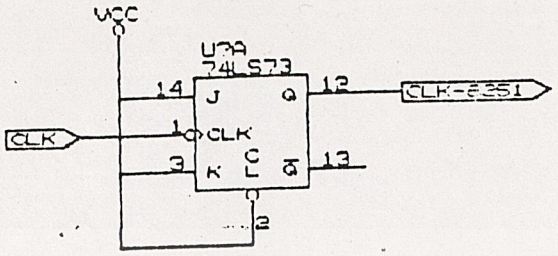
HARDWARE ที่เกี่ยวกับ 8251

1.CLOCK

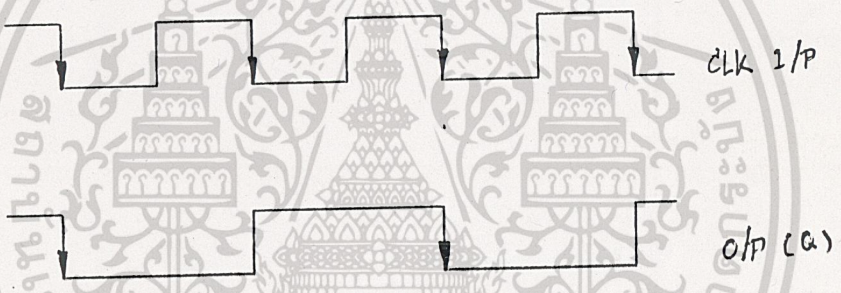
สำหรับ CLOCK ที่ 8251 ต้องการต้องไม่เกิน 3 MHz ซึ่ง CLOCK ที่มาจาก CPU 8088 มีค่าเท่ากับ 4.77 MHz ดังนั้นเราจึงต้องใช้ JK FLIP-FLOP มาเพื่อหาร CLOCK ของ CPU ลงให้เหลือครึ่งหนึ่งคือเหลือเท่ากับ 2.39 MHz ก่อนจึงจะนำมาจ่ายให้ 8251 ได้ ตามวงจรในรูปที่ 2.3 และผลลัพธ์ในรูปที่ 2.4

INPUT				OUTPUT
CLR	CLK	J	K	Q
H	↓	H	H	TOGGLE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



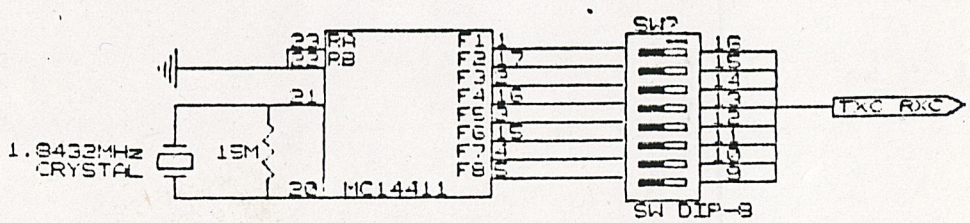
รูปที่ 2.3



รูปที่ 2.4

2.TxC และ RxC (TRANSMITTER CLOCK และ RECIEVE CLOCK)

ขานี้ใช้กำหนดอัตราโบด์ (BAUD RATE) ของการส่งและรับข้อมูลของ 8251 ทั้ง MASTER และ SLAVE ต้องใช้อัตราโบด์เดียวกัน สำหรับการทดลองนี้ใช้อัตราโบด์เท่ากับ 9600 และวงจรที่ใช้กำเนิดอัตราโบด์ทั้ง MASTER และ SLAVE ใช้วงจรเดียวกันคือ วงจรรูปที่ 2.5 และมีตารางแสดงการทำงานของ MC 14411 ดังตารางในหน้าต่อไป



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น รวมถึงการอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 2.5 วงจรที่ใช้กำเนิดอัตราโบด์

RATE SELECT		RATE
A	B	
0	0	*1
0	1	*8
1	0	*16
1	1	*64

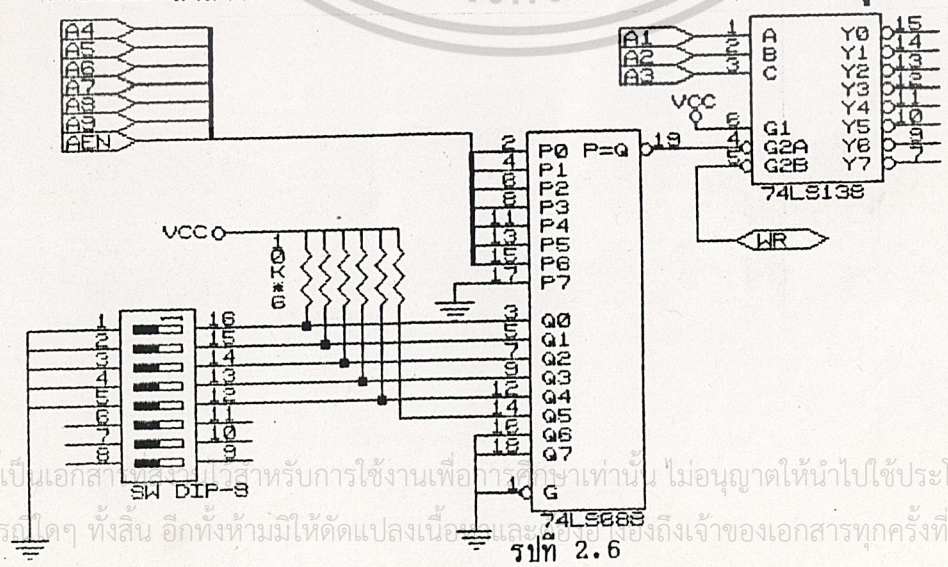
O/P NUMBER	O/P RATE *1
F1	9600
F2	7200
F3	3600
F4	2400
F5	1800
F6	1200
F7	600
F8	300

3.CS (CHIP SELECT)

เราใช้ 74LS138 ในการ DECODE เบอร์ PORT โดยเราเลือกใช้ PORT เบอร์ 0301H เป็น PORT CONTROL และ PORT เบอร์ 0300H เป็น PORT DATA

ในการเลือกเบอร์ PORT ที่จะเป็น CHIP SELECT นั้น เราจะต้อง A9 จาก 8088 เป็น 1 เสมอ ถ้าเราใช้อุปกรณ์ที่อยู่นอกเหนือจากอุปกรณ์ที่อยู่บน MAIN BOARD และให้ AEN (ADDRESS ENABLE) จะต้องเป็น 0 ตลอด ถ้าไม่เป็นการทำ DMA ส่วน A0 จะเป็นตัวเลือกว่าจะเป็น PORT CONTROL หรือ PORT DATA โดยต่อไว้กับขา C/D ของ 8251 ซึ่งจะกล่าวในตอนต่อไป

74LS138 สามารถ DECODE PORT เบอร์ดังต่อไปนี้ โดยใช้วงจรตามรูป 2.6 คือ



รูปที่ 2.6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Y0 = 0300H,0301H Y1 = 0302H,0303H

Y2 = 0304H,0305H Y3 = 0306H,0307H

Y4 = 0308H,0309H Y5 = 030AH,030BH

Y6 = 030CH,030DH Y7 = 030EH,030FH

4.C/D (CONTROL/DATA)

ใช้ A0 จาก CPU ต่อโดยตรงกับขานี้ของ 8251 เพื่อเป็นการเลือกว่าขณะนี้เป็นการให้ CPU ส่งรหัสควบคุมกับเลือก MODE และ CPU รับรหัสแสดงสถานะ ถ้า A0 เป็น 1 หรือในกรณีที่ทั้งรับทั้งส่งข้อมูลจะต้องให้ A0 เป็น 0 โดยขานี้ของ 8251 ต้องใช้คู่กับ RD และ WR

5.RD/WR (READ/WRITE)

สำหรับ 8088 จะมีขา IOR เป็น OUTPUT แอคทีฟที่ลอจิก 0 เพื่อเป็นการอ่านข้อมูลจาก PORT อยู่แล้วและยังมีขา IOW เป็น OUTPUT แอคทีฟที่ลอจิก 0 เพื่อเป็นการเขียนข้อมูลลงบน PORT ดังนั้น ขา RD ของ 8251 จึงสามารถต่อกับ IOR ของ 8088 ได้เลย และสำหรับ WR ของ 8251 ก็เช่นกัน สำหรับ WR ของ 8251 ก็เช่นกันสามารถต่อกับขา IOW ของ 8088 ได้โดยตรง

ตารางข้างล่างแสดงสถานะการทำงานร่วมกันระหว่างขา C/D , RD , WR และ CS ของ 8251

C/D	RD	WR	CS	ความหมาย
0	0	1	0	ข้อมูลจาก 8251 ถูกส่งไปยังบัสข้อมูล
0	1	0	0	ข้อมูลจากบัสข้อมูลไปยัง 8251
1	0	1	0	8251 อ่านรหัสแสดงสถานะจากบัสข้อมูล
x	x	x	1	บัสข้อมูล 8251 ลอยตัว

6.RESET

เนื่องจากการ RESET ของ 8251 จะต้อง RESET ตามการ RESET PROGRAM ที่เกี่ยวข้องกับ 8251 ของ CPU ดังนั้นการต่อขา RESET (ACTIVE ที่ 1) กับขา RESET DRIVER ของ CPU ซึ่งจะเป็นขาที่ ACTIVE ที่ 1 ในระยะแรกที่เริ่มเปิดเครื่อง และจะ ACTIVE ต่อไปจนกระทั่งระบบต่างๆของเครื่องพร้อมที่จะทำงานได้จึงจะตกเป็น 0 นั้น ไม่พอเพียงกับการ RESET 8251 เพราะการ RESET 8251 คือการเคลียร์ BUFFER ข้อมูลใน 8251 ให้ว่างก่อนเพื่อพร้อมที่จะดำเนินการส่งหรือรับข้อมูลต่อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไปทุกครั้งที่มีการ RUN PROGRAM เกี่ยวกับการส่งหรือรับข้อมูล ดังนั้นจึงต้องมีการทำ INTERNAL RESET 8251 ก่อนทุกครั้งด้วย SOFTWARE ซึ่งจะสั่ง INTERNAL RESET ได้โดยการกำหนดที่ COMMAND WORD ในส่วนโปรแกรม INITIAL ของการดำเนินงานกับ 8251

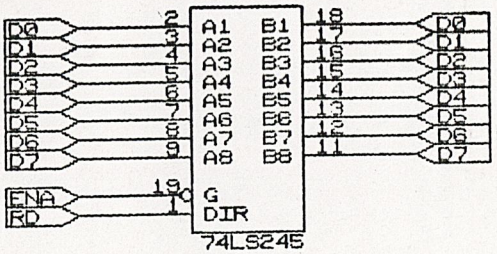
7.CTS (CLEAR TO SEND)

เป็นสัญญาณ INPUT จากภายนอก เมื่อ CTS เป็น 0 ตัว 8251 จึงจะสามารถส่งข้อมูลออกไปได้ (ถ้าสัญญาณ TxEN (TRANSMITTER ENABLE) ในรหัสควบคุมเป็น 1 ด้วย) ดังนั้นทั้ง MASTER และ SLAVE เราจะเอาขานี้ของ 8251 ต่อไว้กับ GND

8.D0-D7

เราควรต่อไว้ BUFFER 2 ทาง (74LS245) ระหว่างขา D0-D7 ของ CPU (ทั้ง MASTER และ SLAVE) กับขา D0-D7 ของ 8251 เพื่อที่จะได้กำหนดทิศทางการไหลของข้อมูลระหว่าง CPU กับ 8251 ได้ถูกต้องยิ่งขึ้น โดยที่เราสามารถกำหนดทิศทางการไหลของข้อมูลได้จากขา ENABLE G (ขา 19) กับขา 1 ของ 74LS245 คือเราให้ขา ENABLE G ต่อกับขา 19 (ENABLE) ของ 74LS688 และขา 1 ต่อกับขา RD คือเมื่อขา 1 เป็น "0" ข้อมูลจาก 8251 ก็จะไหลไปทาง CPU ได้ และถ้าขา 1 ของ 74LS245 "1" ข้อมูลจาก CPU ก็จะไหลไปให้ 8251 ได้ ส่วนลักษณะการต่อวงจรจะเป็นไปตามรูปที่ 2.7 และทิศทางการไหลของข้อมูลจะเป็นไปตามตารางข้างล่างนี้

DIR	DATA FLOW
1	CPU TO 8251
0	8251 TO CPU



รูปที่ 2.7

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
028820

การอินเทอรัพต์

สำหรับโครงงานนี้เราต้องการให้ MASTER เป็นตัวเลือกเครื่อง SLAVE โดยการส่งรหัสมายังเครื่อง SLAVE จะมีเครื่อง SLAVE เพียงเครื่องเดียวนั้นที่จะสามารถถอดรหัสได้ และสัญญาณที่ได้จากการถอดรหัสนี้ เราจะนำไปทริกขา IRQ2 8259 เพื่อจัด INTERRUPT CPU ของเครื่อง SLAVE

ขั้นตอนของการอินเทอรัพต์ 8259

1. มีสัญญาณขออินเทอรัพต์เข้ามาทางขา IRQ2

2. CPU จะเก็บ FLAG และ RETURN ADDRESS ลงบน STACK และทำให้ DATABUS อยู่ในอินพุทโมด

3. CPU ส่งสัญญาณ ACKNOWLEDGE โดยส่งมาเป็นพัลซ์ 2 ลูก

-ลูกแรกจะทำให้ 8259 ทำตาม MODE ที่สั่งไว้

-ลูกที่สอง 8259 ส่ง INTERRUPT VECTOR ไปให้ CPU

4. เมื่อ CPU ได้รับ INTERRUPT VECTOR ก็จะทำการหาตำแหน่ง ADDRESS เริ่มต้นของโปรแกรมทำงาน

จาก INTERRUPT VECTOR ที่กล่าวมาแล้วจะขึ้นอยู่กับ CHANNEL ของการ INTERRUPT และการ INITIAL 8259 เอง

สำหรับการจัดลำดับความสำคัญของการ INTERRUPT ขึ้นอยู่กับการเลือก MODE ตอนสั่ง INITIAL 8259 ซึ่งเราไม่จำเป็นจะต้องสั่งเองถ้าใช้เครื่องคอมพิวเตอร์ CPU 8088

การโปรแกรม 8259

ภายใน 8259 จะมีรีจิสเตอร์ ที่ใช้ในการเก็บคำสั่งต่างๆ ที่เรียกว่า COMMAND WORD อยู่ 2 ชนิด คือ

1. INITIAL COMMAND WORD (ICW) ซึ่งจะมีอยู่ 4 ชนิดคือ

1.1 ICW1

-D_0 เป็นการเลือกว่าจะต้องการสั่ง ICW4 หรือไม่ ถ้าต้องการ BIT นี้จะเป็น 1

-D_1 เป็นการเลือก MODE ว่าจะใช้ 8259 เพียงตัวเดียว หรืออยู่ใน CASCADE MODE ถ้า BIT นี้เป็น 1 จะใช้ 8259 เพียงตัวเดียว

-D_2 บิท นี้จะถูกใช้งานเฉพาะในกรณีที่ทำงานร่วมกับ CPU8080 เพราะฉนั้น BIT นี้จะถูก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

RESET เป็น 0

-D_3 เป็นการเลือกลักษณะสัญญาณที่จะมา TRIG IRQ2 ว่าจะเป็นการ LEVEL TRIGGER หรือ EDGE TRIGGER ถ้า บิทนี้เป็น 1 จะเป็นการเลือกแบบ LEVEL TRIGGER แต่สำหรับการ TRIG ที่ใช้ในโครงงานนี้จะเป็นแบบ EDGE TRIGGER

-D_5 D_6 D_7 บิททั้ง 3 นี้จะถูกใช้เฉพาะกับ CPU 8080,8085 เท่านั้นดังนั้นจึง RESET ข้อมูลใน BIT ทั้ง 3 นี้เป็น 0 ทั้งหมด

1.2 ICW2

-D_0 -D_2 ข้อมูลในทั้ง 3 บิทนี้จะถูกใช้งานในแต่เฉพาะ CPU 8080,8085 เท่านั้นดังนั้นจึงให้ข้อมูลทั้ง 3 ตัวนี้เป็น 0 ทั้งหมด

-D_3 -D_7 สำหรับ CPU 8088 จะถูกสั่งให้เป็นไปตามตารางข้างบนอยู่แล้ว ข้อมูลพวกนี้จะถูกใช้เพื่อประกอบการเปิดตาราง VECTOR ของ CPU

1.3 ICW3 สำหรับ COMMAND WORD ตัวนี้จะใช้กับระบบที่มี 8259 มากกว่า 1 ตัวดังนั้นในระบบนี้จึงไม่ต้องนำมาพิจารณา

1.4 ICW4

-D_0 สำหรับ CPU 8088 บิท นี้จะถูก SET ให้เป็น 1

-D_1 ถ้าเป็น 1 จะอยู่ใน MODE AUTO END OF INTERRUPT แต่ถ้าเป็น 0 จะอยู่ในโหมด NORMAL END OF INTERRUPT

-D_2 D_3 จะใช้ 2 บิทนี้ทำงานร่วมกันโดย D_2 จะใช้ในโหมด CASCADE

2.OPERATION COMMAND WORD (OCW) มีอยู่ 3 ชนิดคือ

2.1 OCW1

-D_0-D_7 ถ้าบิทไหนเป็น 1 จะเป็นการ MASK(การ DISABLE)สำหรับการขอ INTERRUPT ใน CHANNEL นั้นและสำหรับในโครงงานนี้จะเป็น COMMAND WORD เพียงตัวเดียวที่ต้อง SET ค่าให้มันใหม่โดยOCW1=FBH ซึ่งหมายความว่าเราให้บิท D_2 เป็นศูนย์แสดงการขอ INTERRUPT ทาง CHANNEL นี้สามารถกระทำได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2 OCW2

-D_0-D_2 ข้อมูลในบิททั้งสามนี้จะใช้สำหรับกำหนดว่าคำสั่งที่ส่งให้กับ OCW2 นี้จะมีผลต่อ -

CHANNEL ซึ่งจะเป็นรหัสเลขฐานสองของ CHANNEL ที่ต้องการ จะอย่างไรก็ตามบิททั้งสามนี้จะถูกใช้งานหรือไม่ขึ้นอยู่กับข้อมูลในบิท D_6

-D_5-D_7 บิททั้งสามนี้จะใช้งานร่วมกันเพื่อกำหนดโหมดการทำงานต่างๆของ 8259 โดยบิท D_5 จะใช้กำหนดว่าคำสั่งที่ส่งให้กับ 8259 นี้อยู่ในโหมด END OF INTERRUPT(EOI) หรือไม่บิท D_4 - จะใช้กำหนดว่าคำสั่งนี้ CHANNEL ใดของการ INTERRUPT (สั่งการ D_0-D_2)จะได้รับผลจากคำสั่งนี่หรือไม่บิท D_7 จะถูกใช้ในการกำหนดคำสั่งในโหมด ROTATION PRIORITY หรือไม่

2.3 OCW3

-D_0-D_1ข้อมูลในบิททั้ง 2 นี้จะถูกใช้ในการอ่านสถานะของ 8259 โดยข้อมูลในบิท D_0(RIS) จะใช้สำหรับเลือก REGISTER สถานะของ 8259 คือ INTERRUPT มาทาง CHANNEL ใดบ้าง เมื่อบิทนี้เป็น 1 และ IS(IN-SERVICE REGISTER) เป็น REGISTER ขนาด 8 บิทที่แสดงว่า CHANNEL ใดบ้างที่กำลังได้รับการตอบสนองอยู่ เมื่อบิทนี้เป็น 0 ซึ่ง 8259 จะส่งสถานะนี้ไปยัง DATABUS เมื่อพัลส์ของสัญญาณ READ ถูกส่งให้ 8259 แต่ทั้งนี้ต้องขึ้นอยู่กับ RR (D_1) ด้วยคือ RR ต้องเป็น 1จึงจะมีการส่งสถานะนี้ไปยัง DATABUS ได้

-D_2 ถ้าข้อมูลในบิทนี้ถูก SET เป็น 1 8259 จะทำงานใน MODE POLL .

-D_5 ถ้าบิทนี้เป็น 1 8259 จะทำงานใน MODE SPECIAL MASK แต่อย่างไรก็ตามผลข้อมูลใน BIT นี้ยังต้องขึ้นกับข้อมูลในบิท D_6 ด้วย

-D_4 ถ้าต้องการให้ 8259 ทำงานในโหมด MASK ต้องให้ข้อมูลในบิทนี้เป็น 1

สำหรับการโปรแกรมตัว 8259 ใน IBM/PC นั้นนอกจาก OCW1 แล้วตัว PC ทำการ SET 8259 เองอยู่แล้วเพราะฉนั้นไม่สมควรที่จะ SET ใหม่เพราะอาจทำให้ระบบทำงานผิดพลาดได้

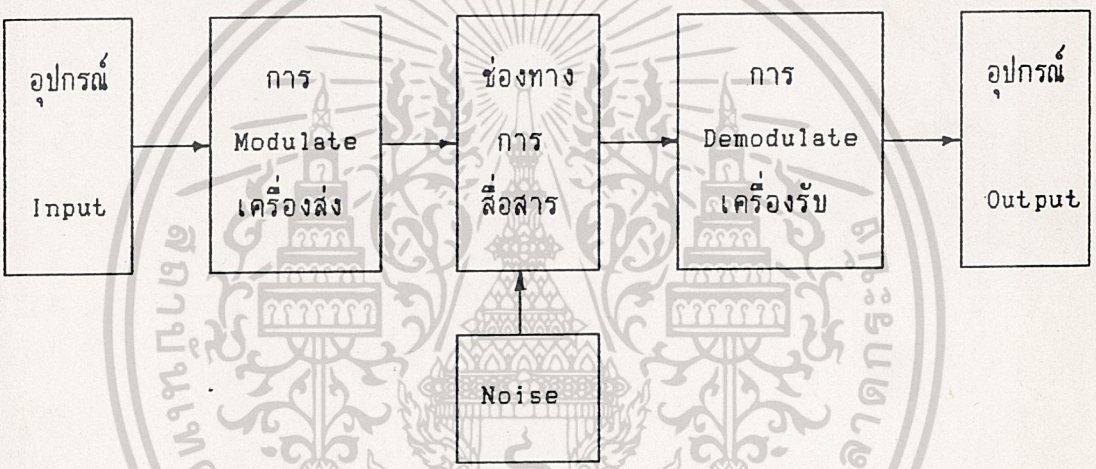
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การทำงานของวงจรส่วนภาคส่งและรับวิทยุ

แนวความคิดของการส่งผ่านข้อมูลนี้จะเป็นการส่งข้อมูลกระจายออกสู่อากาศโดยไม่ต้องสิ้นเปลืองกับสายส่ง และไม่ต้องยึดติดกับสถานที่ในการสื่อสารข้อมูล ดังนั้นส่วนสำคัญในการเชื่อมต่อ หรือติดต่อสื่อสารข้อมูลระหว่างเครื่อง Microcomputer นั้นก็คือ เครื่องรับ และ เครื่องส่งวิทยุ

ระบบการสื่อสารพื้นฐาน

ในระบบการสื่อสารไม่ว่าจะเป็นระบบใดจะมี Block Diagram เหมือนกัน ดังรูป



- อุปกรณ์ Input

  เป็นอุปกรณ์แปลงข่าวสารเป็นสัญญาณไฟฟ้า

- เครื่องส่ง

ทำหน้าที่รับสัญญาณไฟฟ้าจาก Input แล้ว Modulate ลงบนคลื่นพาหะความถี่สูง แล้วทำการขยายส่งออกอวกาศไปยังเครื่องรับ

วัตถุประสงค์ของการ Modulate สัญญาณ

1. เพื่อการใช้สายอากาศที่มีขนาดสั้นลง

   เนื่องจาก เราต้องใช้สายอากาศที่มีความยาว 1/4 λ , 1/2 λ

2. เพื่อการส่งสัญญาณได้หลายสถานีพร้อมกัน
3. เพื่อหลีกเลี่ยงการส่งที่ความถี่ต่ำเพราะว่าไปไม่ได้ไกล
4. การส่งสัญญาณข้อมูล DIGITAL โดยตรงออกสู่อากาศจะทำการออกแบบวงจรเครื่องรับ - ส่ง

เอกสารนี้เป็นกรรมสิทธิ์ของสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ทั้งนี้ต้องได้รับอนุญาตจากสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังก่อนเท่านั้น

เพื่อรักษาข้อมูลเดิมนั้นเป็นไปโดยลำบาก เพราะว่าสัญญาณ Pulse นั้นมี Bandwidth กว้างมาก

ส่วนประกอบของเครื่องส่ง

1.แหล่งกำเนิดสัญญาณความถี่สูง (Oscillator)

2.Modulator

3.วงจรขยาย

- ช่องทางสื่อสาร

สำหรับระบบสื่อสารเครื่องวิทยุนั้น จะเป็นบรรยากาศ (Free Space)

- Noise

เป็นสัญญาณรบกวน หรือ แทรกแซงที่เกิดขึ้นกับระบบสัญญาณทำให้สัญญาณที่ได้ผิดเพี้ยนไปจากเดิม

- เครื่องรับ

ทำหน้าที่รับ และ ขยายสัญญาณที่ต้องการจากสัญญาณที่มีกำลังอ่อน + Noise จากเครื่องส่ง สัญญาณที่ได้จะผ่านการ Demodulate เพื่อแปลงข่าวสารที่ Modulate จากภาคส่งกลับมา

ส่วนประกอบของเครื่องรับ

1.วงจรเลือกรับสัญญาณ (Detecter)

2.วงจรขยาย

3.Demodulator

- อุปกรณ์ Output

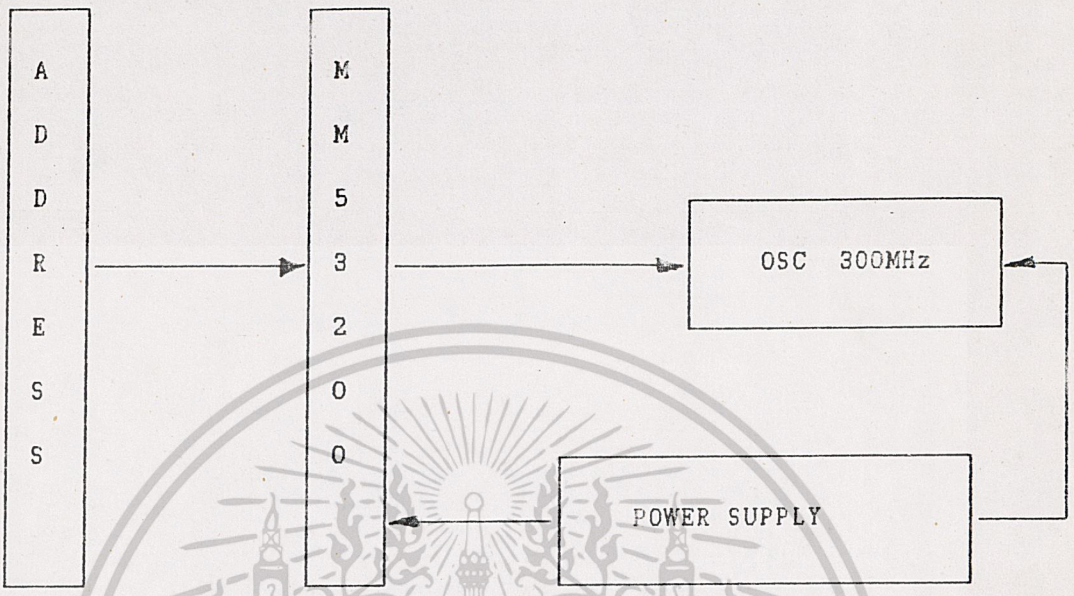
ทำหน้าที่แปลงสัญญาณไฟฟ้ากลับมาเป็นข่าวสาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การคำนวณและการสร้าง

ลักษณะการทำงานของวงจรส่วนภาคส่ง

สามารถอธิบายหลักการทำงานโดย BLOCK DIAGRAM ดังรูปที่ 2.8



รูปที่ 3.1

เมื่อ MASTER ต้องการติดต่อกับ SLAVE จะส่งข้อมูลจาก CPU ผ่านทาง PORT 8251 ส่วนของข้อมูลและส่วนของ ADDRESS ของ SLAVE ที่ต้องการจะติดต่อด้วย จะถูกเลือกโดย IC 4053

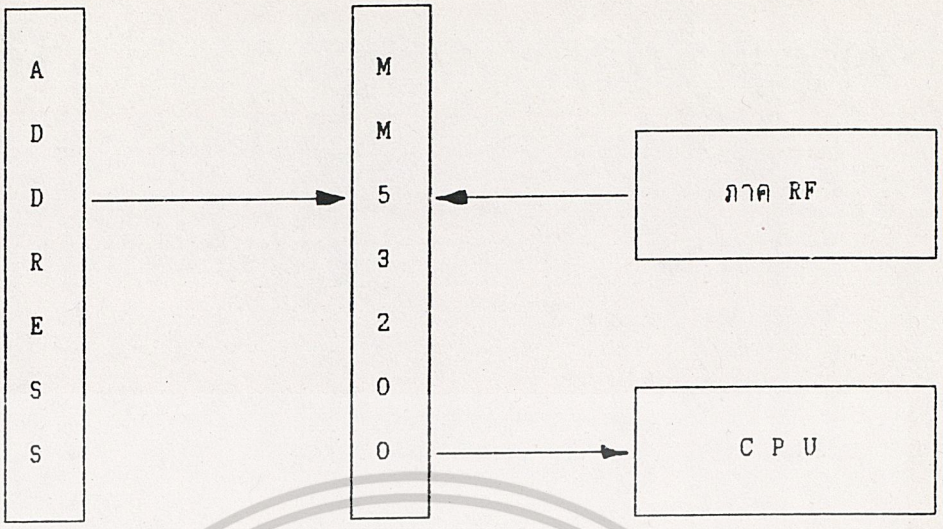
สำหรับตัว MM 53200 เป็นตัว ENCODER ซึ่งจะอธิบายถึงการทำงานของมันต่อไปในท้ายบทนี้ ในที่นี้ขอให้ทราบว่า MM 53200 เป็นตัวบังคับให้เกิดการออสซิลเลต คือเมื่อเอาท์พุทของ MM 53200 มีสถานะสูง จะมีผลทำให้วงจรออสซิลเลเตอร์ เกิดการออสซิลเลตที่ความถี่ในย่าน 300 MHz แต่ถ้าเอาพุทของ MM 53200 อยู่ในสถานะต่ำ จะมีผลทำให้วงจร OSC เกิดการออสซิเลตที่ความถี่นี้ คลื่นนี้จะเป็นคลื่นพาห์ของสัญญาณเอาท์พุทจาก MM 53200 เพื่อส่งออกอากาศต่อไป ลักษณะเช่นนี้อาจกล่าวได้ว่าเป็นการทำงานในแบบของ PULSE MODULATION

ลักษณะการทำงานของวงจรส่วนภาครับ

หลักการทำงานของวงจรส่วนภาครับ สามารถเขียนเป็น BLOCK DIAGRAM ดังรูปที่ 2.9 ในหน้าถัดไป

เมื่อข้อมูลถูกส่งผ่านเข้ามาจากทางภาค RF แล้ว สัญญาณที่ได้ก็จะนำไปป้อนให้กับ IC DECODER ซึ่งในที่นี้เราใช้ MM 53200 ต่ออีกแบบหนึ่ง เช่นในท้ายบทนี้ หรือให้กับ 8251 ตรงขา RxD นั่นก็คือจบวงจรภาครับส่งนั่นเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



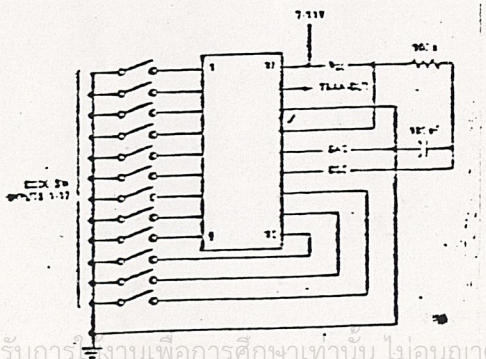
รูปที่ 2.1

การทำงานของ MM 53200

ไอซีเบอร์นี้เป็นไอซีสำหรับการเข้ารหัสและถอดรหัสดิจิตอลขนาด 12 บิท โดยมีการเว้นช่วงว่าง (BLANKING) และมีสตาร์ทบิทให้โดยอัตโนมัติ ดังรูปที่ 2.10

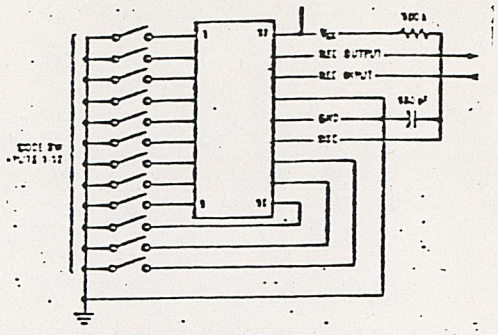
รูปที่ 2.10 แสดงสัญญาณที่ส่งออกจาก MM 53200 เมื่อใช้งานในโหมดการส่ง

การทำงานของไอซีเบอร์นี้แบ่งออกเป็นสองลักษณะคือ การต่อแบบโหมดการส่งและการต่อแบบโหมดการรับ การต่อทั้งสองโหมดนี้แสดงได้ดังรูปที่ 2.11 และรูปที่ 2.12



รูปที่ 2.3 การต่อใช้งานไอซี MM 53200 ในโหมดการส่งข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

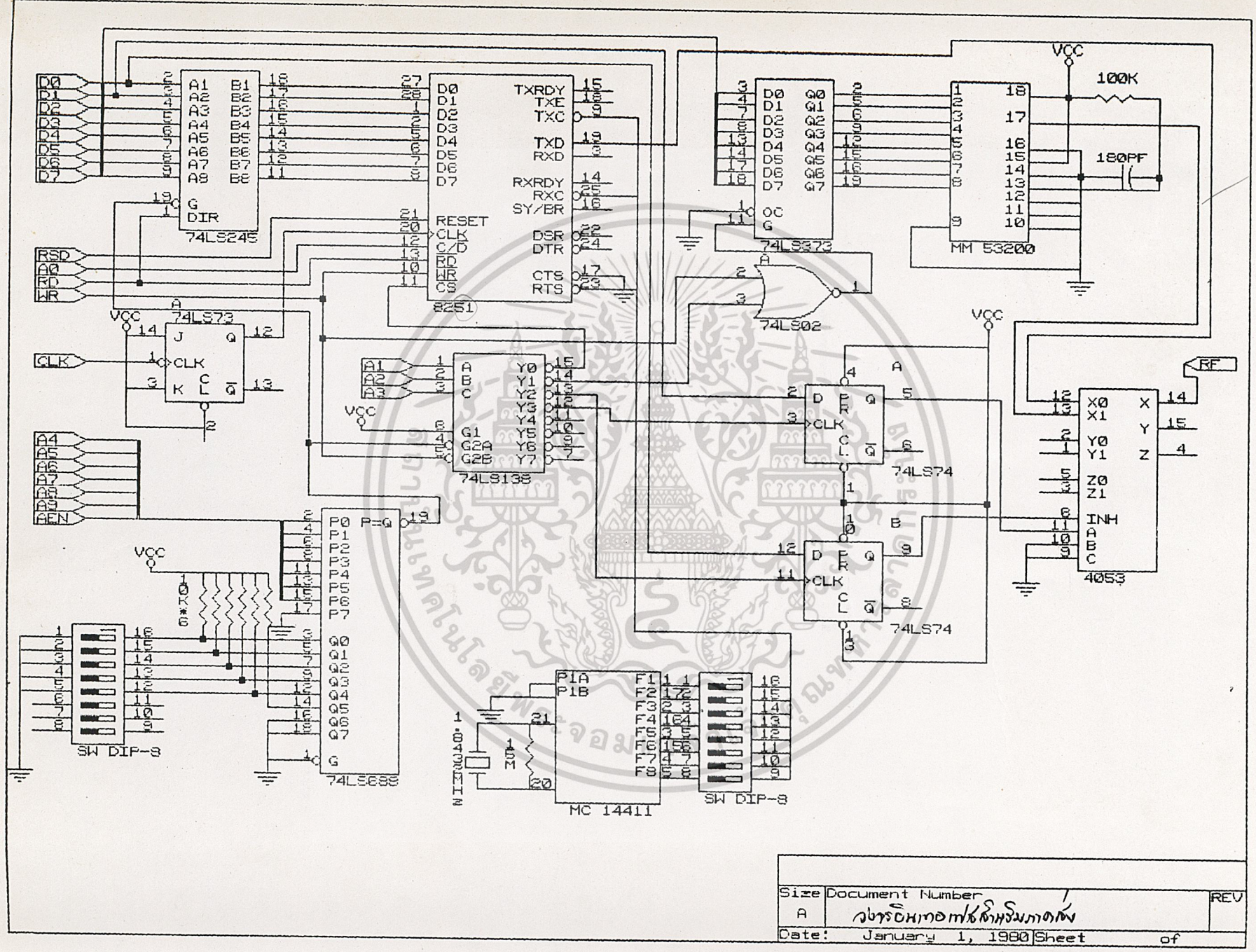


รูปที่ 3.4 การต่อใช้งานไอซี MM 53200 ในโหมดการรับข้อมูล

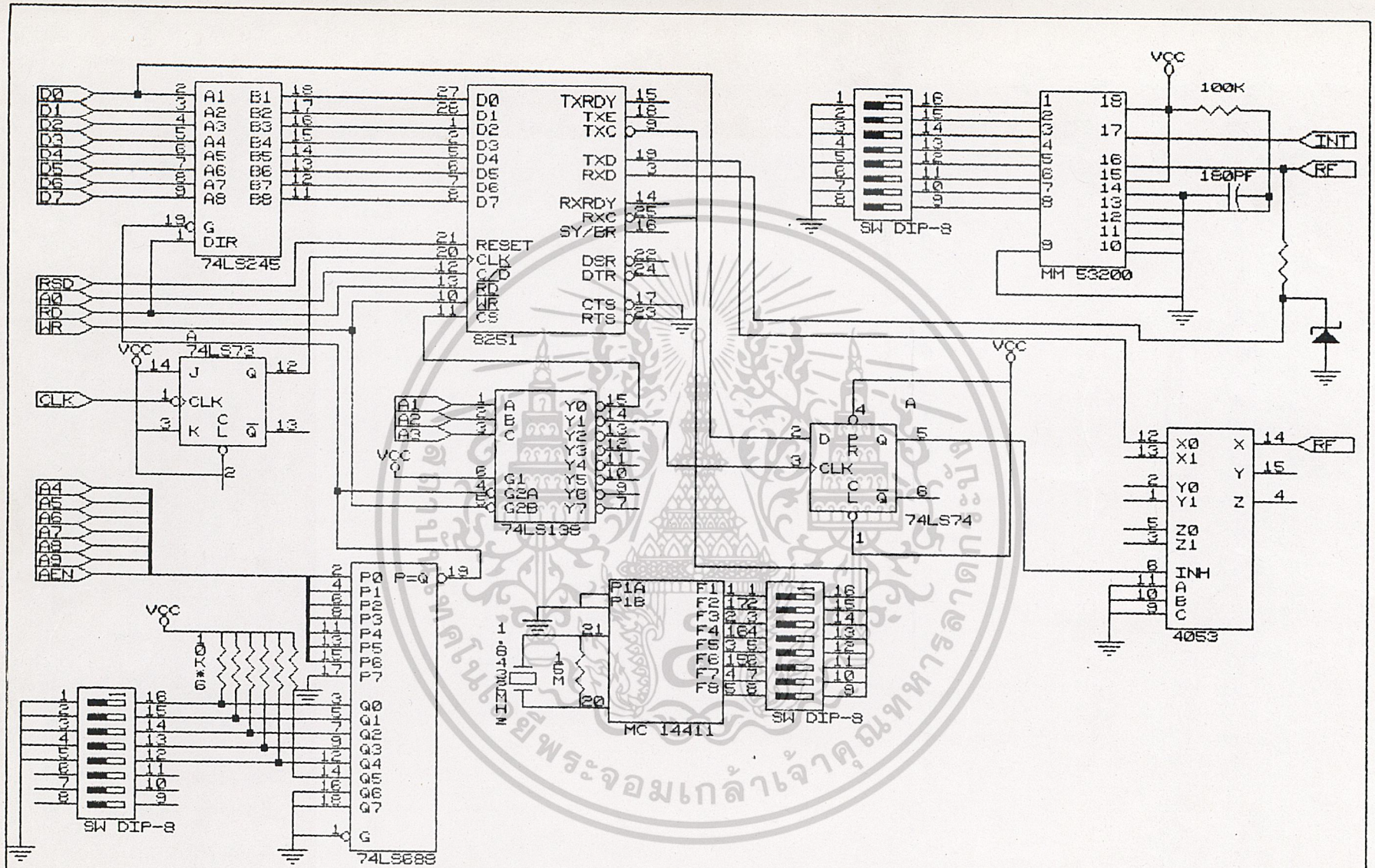
ในโหมดของส่งข้อมูลนั้น เมื่อเราต่อวงจรถูกต้องและจ่ายไฟเลี้ยงถูกต้องให้แก่ไอซี มันจะกำเนิดสัญญาณดิจิตอลขนาด 12 บิท ส่งออกมาทางขา 17 โดยเป็นสัญญาณแบบอนุกรม วนซ้ำอยู่ตลอดเวลาที่ไอซีทำงาน โดยมีลักษณะของสัญญาณเช่นเดียวกับในรูป

ในโหมดการรับนั้น เราต้องต่อใช้งานไอซีดังในรูป เมื่อไอซีทำงานโดยรับข้อมูลเข้าทางขา 16 มันจะทำการเปรียบเทียบรหัสที่มันได้รับขนาด 12 บิท ว่ามีรหัสตรงกับที่ตั้งไว้ด้วยดิพสวิทช์ ที่ต่ออยู่กับตัวมันหรือไม่ โดยทำการเปรียบเทียบในแบบอนุกรม ถ้ารหัสที่รับได้ไม่ตรง ไอซีจะรีเซ็ทตัวเองแล้วรอเปรียบเทียบรหัสในชุดต่อไป ถ้ารหัสที่ได้รับตรงกับตัวมัน มันจะรอตรวจสอบว่าตรงกันต่อไปอีก 3 ครั้ง จึงจะทำให้เอาท์พุทที่ขา 17 เปลี่ยนสถานะจากสูงเป็นต่ำ หลังจากนั้นไอซีจะต้องได้รับรหัสที่ถูกต้องอีกภายในเวลา 128 มิลลิวินาที (ที่ความถี่สัญญาณนาฬิกาเท่ากับ 100 KHz) เอาท์พุทของไอซีจึงจะยังคงสถานะเป็นต่ำอยู่ต่อไป ถ้าไม่เช่นนั้นแล้ว เอาท์พุทของไอซีก็จะเปลี่ยนสถานะเป็นสูง แล้วรอทำการตรวจสอบรหัสอีก 4 ครั้งต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



Size	Document Number	REV
A	วงจรอินเทอร์เฟสสำหรับการสื่อ	
Date: January 1, 1980	Sheet of	



Size	Document Number	REV
A	วงจรอินเทอร์เฟสสำหรับงานทดลอง	
Date: January 1, 1980	Sheet of	

เทคนิคในการควบคุมการรับส่งข้อมูล

ในการรับส่งข้อมูล สิ่งที่หลีกเลี่ยงไม่ได้ก็คือ ความผิดพลาดในการรับส่งข้อมูล เพื่อที่จะลดความผิดพลาดในการรับส่งข้อมูล เราจึงต้องมีเทคนิคที่เหมือนกันทั้งฝ่ายรับและฝ่ายส่ง ในการควบคุมการรับส่งข้อมูล เทคนิคที่ว่านี้ก็คือ การส่งโปรโตคอล (PROTOCOL TRANSFER) โดยการใช้อักขระควบคุมในตารางของ ASCII สำหรับควบคุมการส่งข้อมูลออกมาเป็นกลุ่มที่มีขนาดคงที่

การใช้โปรโตคอลที่เป็นอักขระในโครงงานนี้ สำหรับควบคุมการส่งข้อมูลเป็นกลุ่มๆ มีดังต่อไปนี้

ACK (ACKNOWLEDGE) เป็นรหัสที่ใช้บอกตัวส่งว่าพร้อมแล้วต้องการจะติดต่อด้วยและเป็นสัญญาณที่ใช้บอกตัวส่งว่าข้อมูลถูกต้อง ให้ส่งข้อมูลบล็อกต่อไปได้ ซึ่งมีค่าเท่ากับ 06 ในรหัสแอสกี้

NAK (NEGATIVE ACKNOWLEDGE) เป็นรหัสที่ใช้บอกฝ่ายส่งว่าข้อมูลในบล็อกที่ผ่านมาผิดพลาด ให้ส่งข้อมูลบล็อกเดิมมาอีกครั้ง ซึ่งมีค่าเท่ากับ 21 ในรหัสแอสกี้

ETB (END OF TRANSMISSION BLOCK) เป็นการบอกฝ่ายรับว่าขณะนี้สิ้นสุดการส่งข้อมูลกลุ่มหนึ่งแล้ว ซึ่งมีค่าเท่ากับ 23 ในตารางแอสกี้

EOT (END OF TRANSMISSION) เป็นรหัสที่ใช้ระบุว่าสิ้นสุดการส่งผ่านข้อมูลหรือข้อมูลได้รับเรียบร้อยแล้วให้ยกเลิกการติดต่อ ซึ่งมีค่าเท่ากับ 04 ในรหัสแอสกี้

ENQ (ENQUIRY) เป็นอักขระที่ส่งมาจากฝ่ายรับ ขอให้ฝ่ายส่งส่งข้อมูลมา ซึ่งมีค่าเท่ากับ 05 ในตารางแอสกี้

SOH (START OF HEADING) เป็นส่วนที่เริ่มของส่วนหัว ซึ่งมีค่าเท่ากับ 01 ในตารางแอสกี้

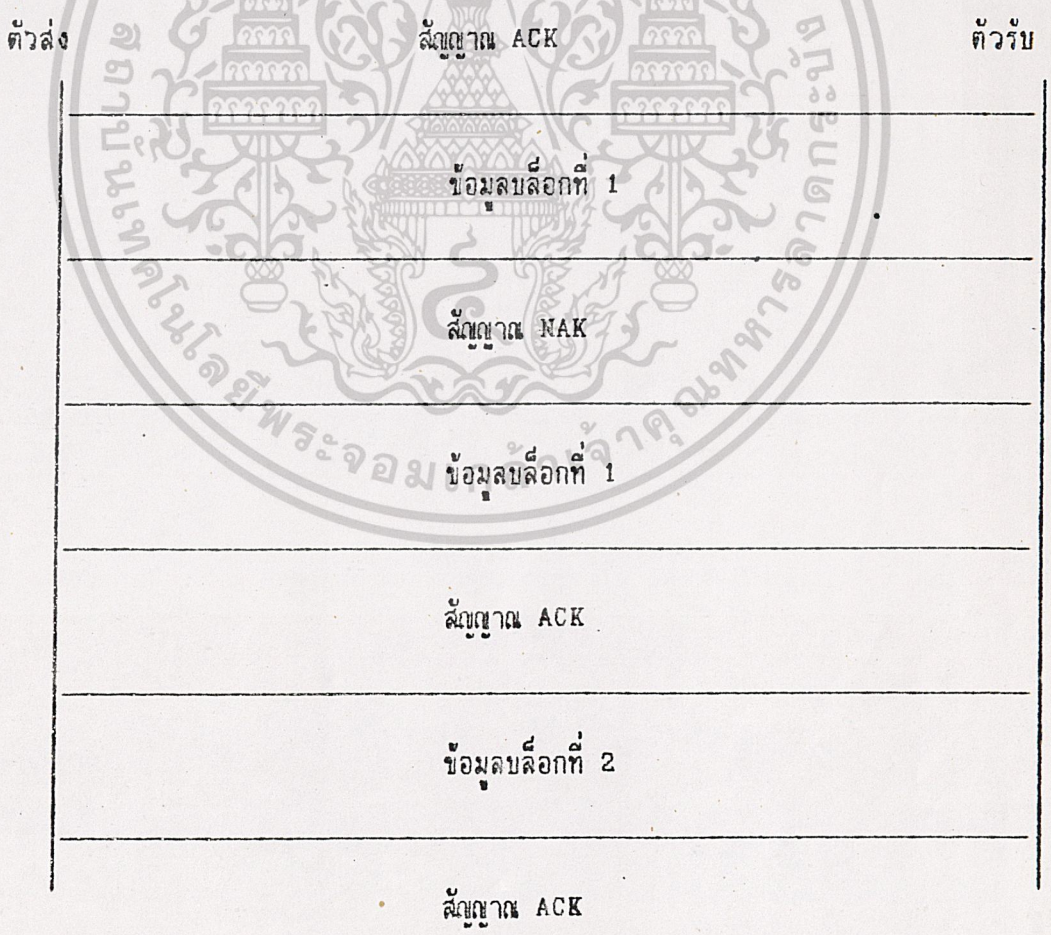
โปรโตคอลที่ใช้ใน IBM PC ส่วนมากจะเป็นโปรโตคอลที่มีชื่อว่า XMODEM ลักษณะการทำงานแสดงไว้ในหน้าถัดไป

ฝ่ายส่งจะยังไม่ส่งข้อมูลจนกว่าจะได้รับ ACK จากฝ่ายรับ ฝ่ายส่งจะส่งข้อมูลออกไปโดยมีรูปแบบเริ่มด้วย SOH ตามด้วยอักขระ 2 ตัว สำหรับบอกกลุ่มของข้อมูลที่ส่งและตามด้วยส่วนเติมเต็ม 1 (1's COMPLEMENT) ของกลุ่มต่อไปที่จะส่ง ต่อจากนั้นก็จะเป็นข้อมูล 128 ไบต์ ตามหลังด้วยการตรวจสอบข้อผิดพลาดโดยมีวิธีตรวจสอบผลบวก (CHECKSUM) การ CHECKSUM คำนวณมาจากการบวกค่า ASCII ของข้อมูลที่ส่งออกไปทั้งหมด 128 ไบต์ แล้วหารด้วย 255 เศษที่เหลือก็คือค่า CHECKSUM ข้างฝ่ายรับเมื่อแยกเอา SOH และหมายเลขบล็อก (BLOCK NUMBER) ทั้งสองออกไปแล้วก็จะเอาข้อมูลทั้ง 128 ไบต์มารวมกันมาเพื่อหาค่า CHECKSUM เอา CHECKSUM ที่หาได้เปรียบเทียบกับค่าที่ได้รับ หากตรงกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ก็ถือว่าข้อมูลที่ได้รับถูกต้องจึงส่งสัญญาณ ACK ไปให้ฝ่ายส่งได้รู้ว่าขณะนี้ได้รับข้อมูลไว้ถูกต้องแล้วส่งกลุ่มของข้อมูลต่อไปมาได้ ถ้าหากค่า CHECKSUM ไม่ถูกต้อง ฝ่ายรับก็จะส่ง NAK ให้ฝ่ายส่งเพื่อเป็นการบอกให้รู้ว่าข้อมูลที่รับได้ผิดพลาด ช่วยส่งกลุ่มของข้อมูลอันเก่ามาให้ ฝ่ายส่งก็จะส่งข้อกลุ่มเก่ามาให้ใหม่-การส่งใหม่จะดำเนินไป 9 ครั้ง หากยังคงได้รับสัญญาณ NAK ฝ่ายส่งจะหยุดการทำงานแสดงว่าตัวกลางการทำงานแสดงว่าตัวกลางการสื่อสารแย่มาก

การที่ XMODEM ใช้เลขบอกกลุ่ม (BLOCK NUMBER)2 ตัว (ตัวหนึ่งบอกกลุ่มที่ส่งขณะนี้ อีกตัวหนึ่งเป็นส่วนเติมเต็ม 1 ของกลุ่มต่อไป) เพื่อความแน่นอนว่ากลุ่มเดียวกันจะไม่ถูกส่งออกไปสองครั้ง ถ้าหากอักขระควบคุมการส่งเกิดสูญหายไประหว่างการส่ง ฝ่ายรับจะตรวจสอบดูว่ากลุ่มของข้อมูลที่ส่งมาเป็นกลุ่มที่ฝ่ายรับต้องการหรือไม่ ถ้าหากกลุ่มเก่าเกิดส่งมาใหม่อีกด้วยความผิดพลาดจาก ACK เป็น NAK ของฝ่ายส่ง ฝ่ายรับก็จะจับข้อมูลที่รับเข้ามาโยนทิ้งไปเมื่อทุกอย่างดำเนินไปอย่างเรียบร้อยจนสิ้นสุดแน่นอนที่จะส่ง ฝ่ายส่งก็จะส่ง ETX เป็นการบอกฝ่ายรับว่าหมดข้อความที่จะส่งแล้ว



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.5 แสดงการส่งข้อมูลซ้ำเมื่อมีการผิดพลาดเกิดขึ้น

XMODEM เหมาะสำหรับ IBM PC เหนือโปรโตคอลชนิดอื่น 3 ข้อ คือ

1. ใช้อักขระควบคุมที่มีอยู่แล้วใน ASCII

2. สามารถใช้ภาษาในระดับสูงควบคุมได้ เช่น ภาษาเบสิก ภาษาปาสคาล

3. ต้องการบัฟเฟอร์สื่อสารแค่ 256 ไบต์

4. ระบบบริการข่าวสารด้วยคอมพิวเตอร์ โดยทั่วไปใช้โปรโตคอล XMODEM

ข้อดีของระบบการควบคุมการรับส่งข้อมูลแบบอนุกรมโดยใช้ระบบการใช้โปรโตคอลก็คือ

1. โปรโตคอลบางชนิสามารถเลือกขนาดของกลุ่มข้อมูลได้

2. สามารถส่งข้อมูลที่ไม่ใช่ ASCII ได้ โดยไม่ต้องกลัวว่ารหัสนั้นจะไปทับรหัสควบคุมของ ASCII

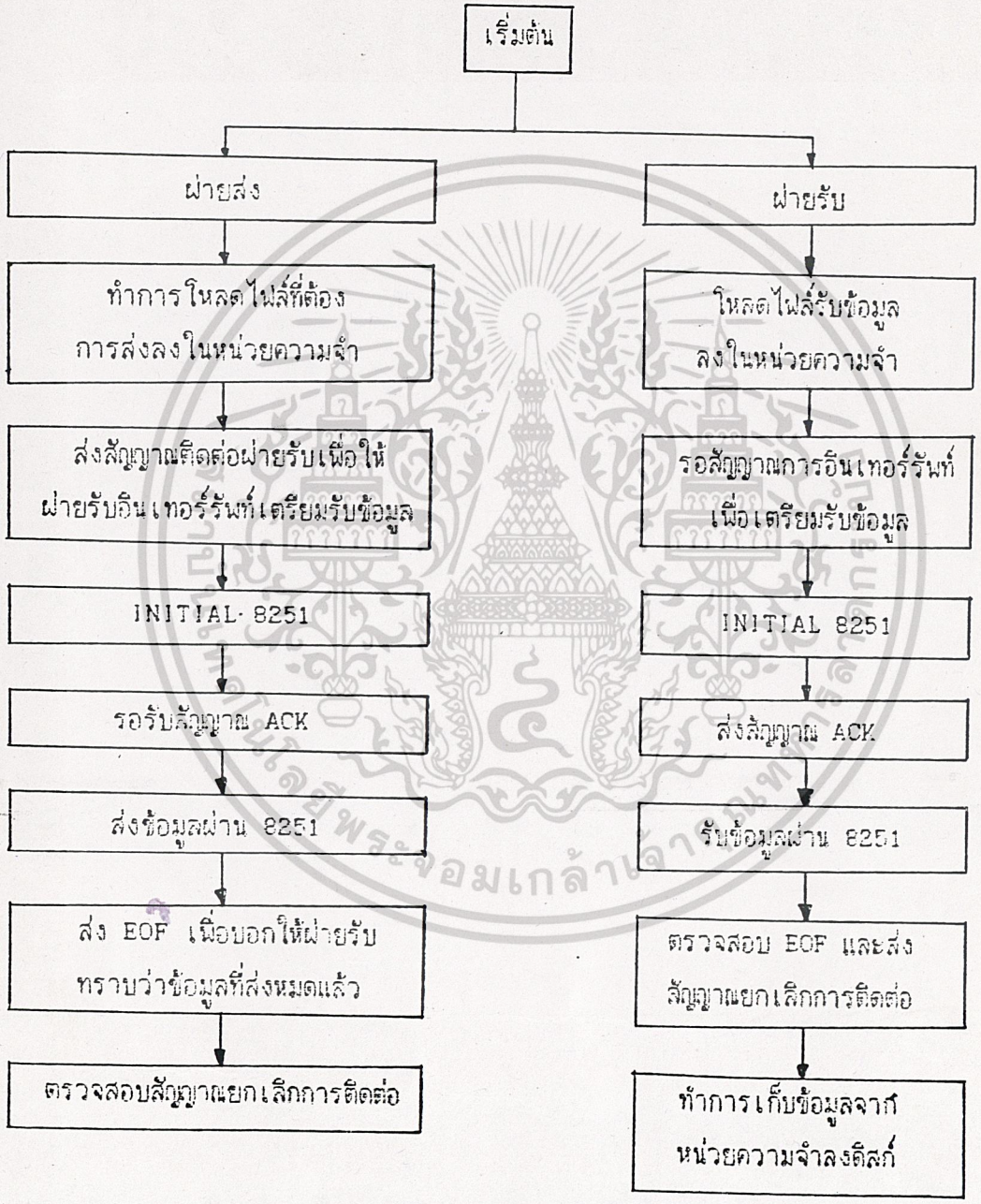
3. การตรวจสอบโดย CHECKSUM มีความสามารถตรวจสอบความผิดพลาดได้ดีกว่า ใช้บิทพาริตี้ในอยชิงโครนัสในขณะที่บิทพาริตี้สามารถให้ประสิทธิผลได้ 95 เปอร์เซ็นต์ หากพบการผิดพลาดด้วยบิทพาริตี้ ไม่ทำให้เกิดการส่งใหม่เกิดขึ้น แต่ข้อผิดพลาดจากการ CHECKSUM ทำให้เกิดการส่งข้อมูลมาใหม่

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลักษณะการทำงานของโปรแกรมรับและส่งข้อมูล

สำหรับการโปรแกรมการทำงานของระบบการรับส่งข้อมูลจะแสดงได้เป็นโฟล์วชาร์ต ดังรูปที่ 5.3



รูปที่ 3.5 แสดงโฟล์วชาร์ตของระบบการรับส่งข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับโปรแกรมการทำงานของตัวส่งหรือ MASTER แสดงได้ดังโฟล์วชาร์ตในรูปที่ 5.4 ซึ่งมีขั้นตอนการทำงานดังนี้

1. จะทำการโหลดไฟล์ที่ต้องการส่งไปให้ SLAVE เก็บไว้ในหน่วยความจำของเครื่องในที่นี้จะทำการจองเนื้อที่ในหน่วยความจำเอาไว้ 20 กิโลไบต์ ดังนั้นไฟล์ที่จะทำการส่งควรจะมีขนาดไม่เกิน 20 กิโลไบต์ แต่ทั้งนี้ถ้าต้องการส่งข้อมูลที่มีขนาดมากกว่า 20 กิโลไบต์ เราสามารถจองเนื้อที่ในหน่วยความจำให้มากกว่านี้ได้ แต่อาจจะมีปัญหาเนื่องจากจะทำให้โปรแกรมของเราใหญ่เกินไปเสียเวลาในการโหลดนาน หรืออาจจะใช้วิธีที่ 2 คือพัฒนาโปรแกรมใหม่ให้สามารถโหลดไฟล์ข้อมูลไม่ต้องมากนักคือประมาณ 1 กิโลไบต์แต่โหลดบ่อยครั้งกว่าเดิม ทั้งนี้จะทำให้ทางฝ่ายรับเมื่อรับข้อมูลแล้วก็จะทำการเก็บข้อมูลจากหน่วยความจำลงดิสก์ ซึ่งเป็นขั้นตอนที่ช้าทำให้ทางฝ่ายส่งต้องรอส่งข้อมูลบล็อคต่อไปนานสาเหตุที่เราให้ฝ่ายรับทำการโหลดไฟล์ข้อมูลก่อนเนื่องจากในขั้นตอนนี้ต้องใช้เวลามาก ดังนั้นถ้าฝ่ายรับทำงานเร็วกว่าคืออาจจะส่งสัญญาณ ACK มาในขณะที่ฝ่ายส่งยังทำการโหลดข้อมูลอยู่อาจจะทำให้สัญญาณ ACK สูญหายได้

2. ส่งสัญญาณการอินเทอร์รัพท์ไปให้ฝ่ายรับ เพื่อให้มีการรันโปรแกรมเพื่อทำการรับข้อมูลที่จะส่งไปให้

3. ทำการ INITIAL 8251 ให้พร้อมที่จะทำงาน

4. รอรับสัญญาณ ACK

5. ขั้นตอนการส่งข้อมูล เมื่อรับสัญญาณ ACK ได้แล้ว จะทำการส่งข้อมูลออกไปทีละไบต์ ในขณะเดียวกันก็จะทำการบวกค่ารหัสแอสกี้ของตัวอักษรนั้น รวมทั้งทำการเช็คหาไบต์สุดท้าย (1AH) ในกรณีที่บล็อคของข้อมูลนี้น้อยกว่า 256 ไบต์ และทำการส่ง EOF ออกไปเพื่อให้ฝ่ายรับรู้ว่าเป็นข้อมูลตัวสุดท้าย หรือเช็คจำนวนข้อมูลจำนวน 256 ไบต์ ในกรณีที่ไฟล์ข้อมูลมีขนาดใหญ่ และยังไม่พบ EOF (1AH) จากนั้นส่ง ETB ออกไปเพื่อให้ฝ่ายรับรู้ว่าเป็นการสิ้นสุดบล็อคข้อมูลนั้นๆ

สำหรับในกรณีที่ไม่สามารถรับสัญญาณ ACK ได้ จะทำการยกเลิกการติดต่อ ทั้งนี้อาจจะพัฒนาโปรแกรมให้ตรวจเช็คสัญญาณนี้ หลายๆ รอบก่อนก็ได้โดยทำการส่งสัญญาณขอให้ตอบไปยังฝ่ายรับ ให้ทำการส่งสัญญาณเดิมกลับมาใหม่ แต่ถ้ายังไม่สามารถรับสัญญาณ ACK ได้แสดงว่าตัวกลางของการสื่อสารนี้เลวร้ายมากก็ควรยกเลิกการติดต่อ

6. ทำการส่งผลรวมของข้อมูล (SUM OF DATA) เพื่อให้ฝ่ายรับทำการเช็คดูว่าข้อมูลที่ส่งไปกับข้อมูลที่รับได้ตรงกันหรือไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7. รอรับสัญญาณการตอบกลับมาของฝ่ายรับ ซึ่งสัญญาณที่ตอบกลับมาอาจจะได้แก่สัญญาณ ACK สัญญาณ EOT และสัญญาณ NAK

โดย ถ้าเป็นสัญญาณ ACK แสดงว่าข้อมูลที่ส่งไปถูกต้องให้ส่งข้อมูลบล็อคต่อไป ไปได้

ถ้าเป็นสัญญาณ NAK แสดงว่าข้อมูลที่ส่งไปผิดพลาดให้ส่งข้อมูลบล็อคเดิมกลับไป

ถ้าเป็นสัญญาณ EOT แสดงว่าข้อมูลที่ส่งไปซึ่งเป็นข้อมูลบล็อคสุดท้ายถูกต้องให้ยกเลิกการติดต่อ

8. ในกรณีที่ข้อมูลที่ส่งไปยังไม่หมดก็จะกลับไปทำตามขั้นตอนการส่งข้อมูลดังที่กล่าวมาแล้วตามลำดับจนกว่าจะเจอ EOF

สำหรับโปรแกรมการทำงานของตัวรับหรือ SLAVE แสดงได้ดังโฟล์วชาร์ตในรูปที่ 5.5 ซึ่งมีขั้นตอนการทำงานดังนี้

1. จะรอรับสัญญาณการเรียกจากตัวส่ง หรือสัญญาณการอินเทอร์รัพท์

2. ทำการ INITIAL 8251 เพื่อให้พร้อมที่จะทำงาน

3. ส่งสัญญาณ ACK

4. ขั้นตอนการรอรับข้อมูล เมื่อมีข้อมูลเข้ามาทางฝ่ายรับจะทำการเช็คว่าเป็น EOF หรือ ETB หรือไม่ ถ้าไม่ใช่ทั้ง 2 อย่างก็จะทำการบวกค่าของรหัสแอสกี้ของข้อมูลนั้นๆ ไปเรื่อยๆ จนกว่าจะเจอ EOF ซึ่งแสดงว่าเป็นข้อมูลไบต์สุดท้าย หรือเจอ ETB ซึ่งแสดงว่าเป็นการสิ้นสุดของข้อมูลบล็อคนั้นแล้ว

5. รับผลรวมของข้อมูลที่ส่งมาจากตัวส่ง

6. ทำการเช็คผลรวมที่ได้ในขั้นตอนที่ 4 และที่ 5 ว่าตรงกันหรือไม่

- ถ้าตรงกันจะทำการส่งสัญญาณ ACK กลับไปและรอรับข้อมูลบล็อคใหม่
- ถ้าไม่ตรงกันจะทำการส่งสัญญาณ NAK กลับไปและรอรับข้อมูลบล็อคเดิม
- ในกรณีที่เจอ EOF และตรวจสอบผลรวมของข้อมูลแล้วถูกต้องก็จะทำการส่งสัญญาณ EOT ออกไปให้ตัวส่งเพื่อเป็นการบอกว่าข้อมูลได้รับเรียบร้อยแล้วให้ทำการยกเลิกการติดต่อ

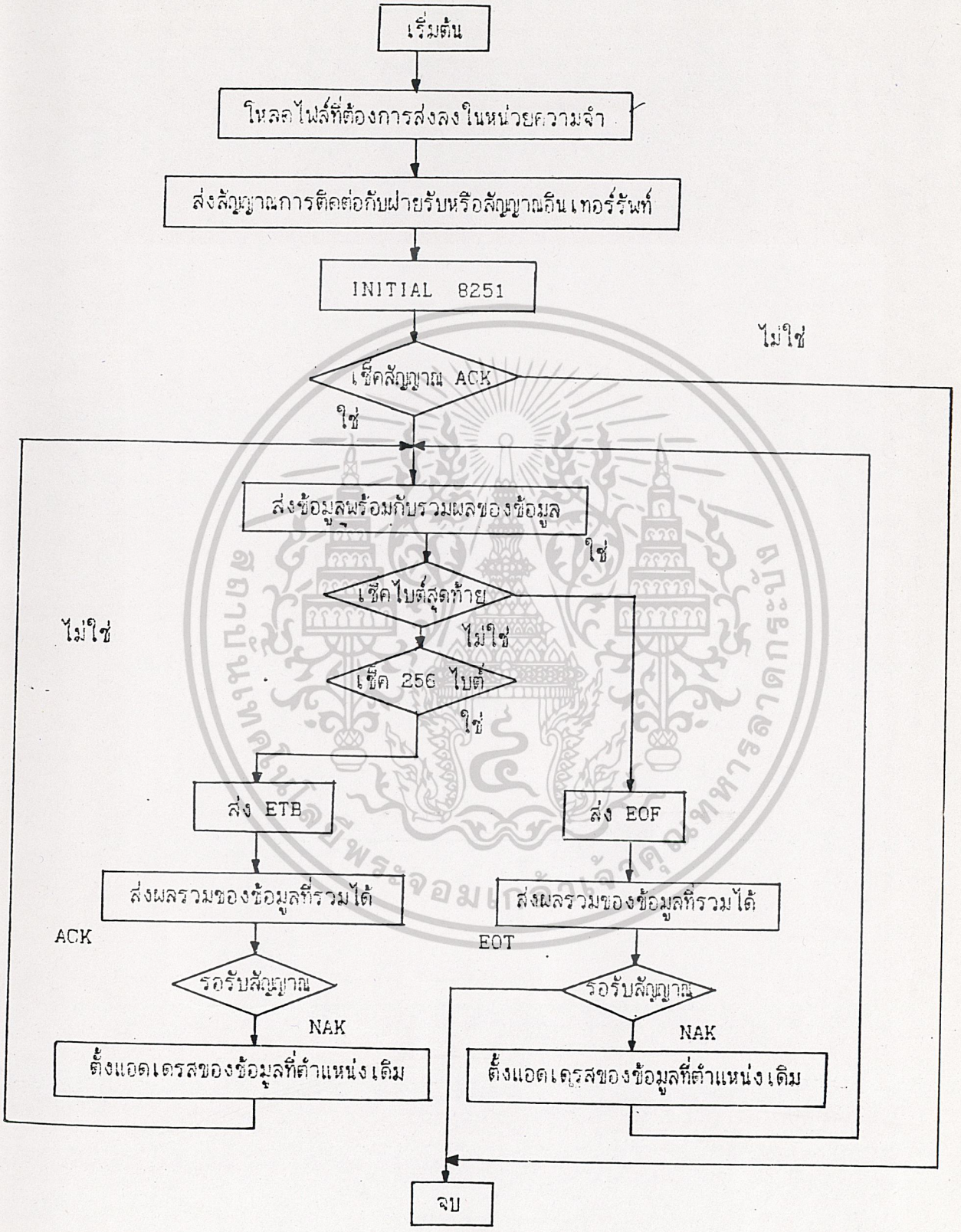
7. ในกรณีที่ยังไม่เจอ EOF จะทำขั้นตอนการรับข้อมูลตามลำดับจนกว่าจะเจอ EOF

8. ปิดท้ายข้อมูลที่รับมาได้เรียบร้อยแล้วด้วย 1AH

9. ทำการเก็บข้อมูลที่รับได้ จากหน่วยความจำลงดิสก์

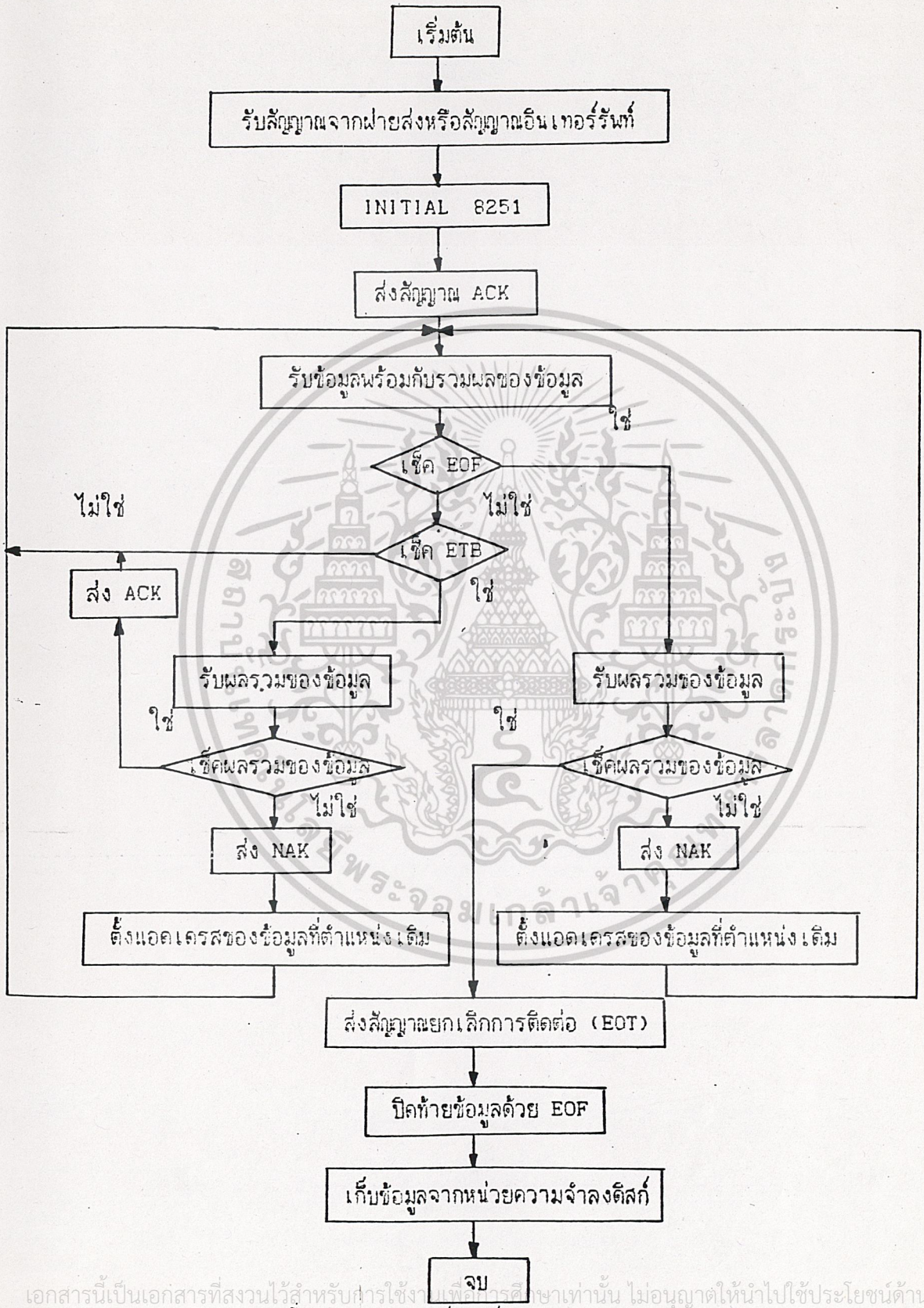
สำหรับซอร์สโปรแกรมของทั้งตัวรับและตัวส่งได้แสดงเอาไว้ในภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.7 แสดงโฟลว์ชาร์ตของการส่งข้อมูลของ MASTER

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

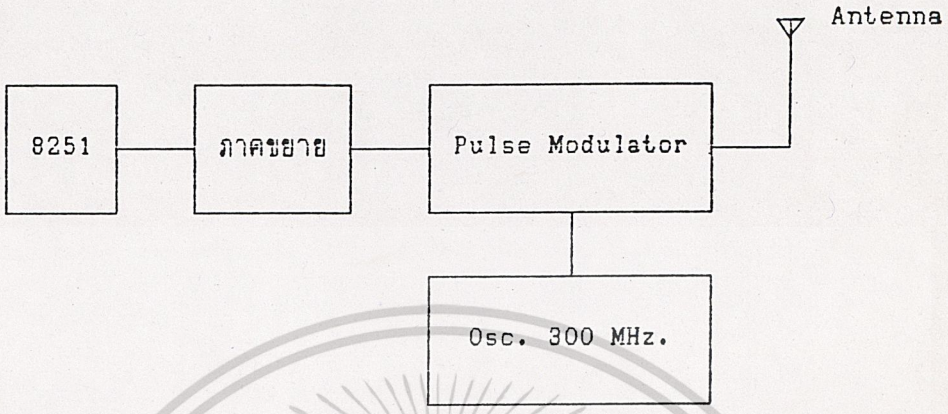


รูปที่ 3.8 แสดงโฟล์วชาร์ตของการรับข้อมูลของ SLAVE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคส่งของเครื่องส่งความถี่วิทยุ

หลักการทำงานของภาคเครื่องส่งวิทยุสามารถเขียนเป็น Block Diagram ได้ ดังรูป

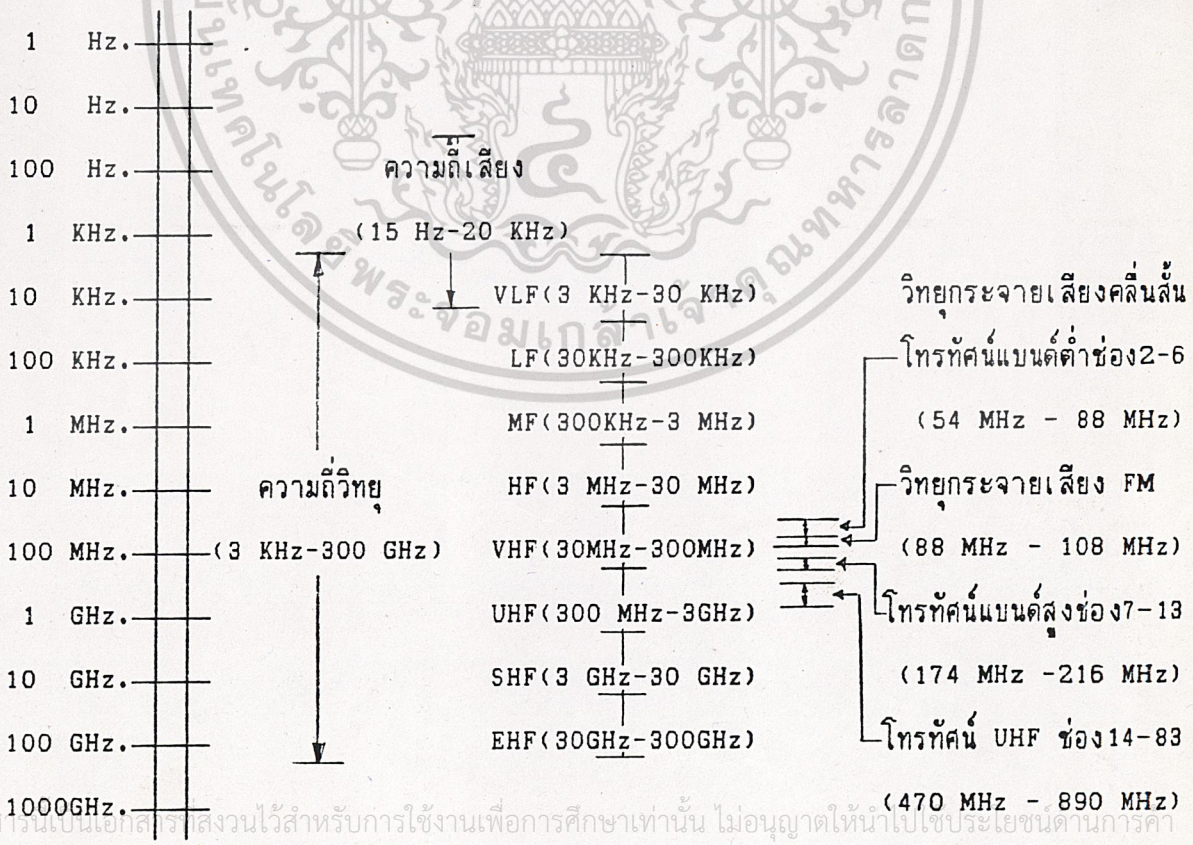


รูปที่ 3.8

- วงจร Oscillator

การเลือกความถี่ใช้งาน

ตารางแสดง Spectrum ความถี่ และการใช้งานแต่ละช่วงความถี่



เอกสารนี้เป็นเอกสารสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะเห็นได้ว่า ในช่วงความถี่ 216 MHz - 470 MHz นั้นไม่มีการใช้งานในการสื่อสาร ดังนั้นจึงเลือกใช้ความถี่ของวงจร Oscillator (หรือความถี่ของคลื่นพาหะที่ใช้ในการผสมสัญญาณ)ที่ 300 MHz ซึ่งจะเกิดสัญญาณรบกวนระหว่างความถี่ข้างเคียงน้อยมาก

ชนิดของวงจร Oscillator

1.Oscillator ที่ใช้ RC

- ใช้งานในช่วงความถี่ต่ำ
- ความถี่สามารถเปลี่ยนแปลงได้
- เสถียรภาพของวงจรไม่ดี

2.Oscillator ที่ใช้ LC

- ความถี่เปลี่ยนแปลงได้ง่าย
- โดยทั่วไปเสถียรภาพของวงจรไม่ดี
- ใช้งานย่านความถี่สูง

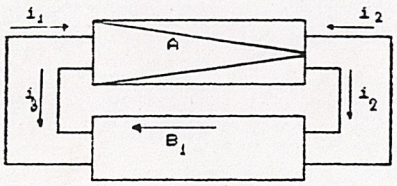
3.Oscillator ที่ใช้ Crystal

- เสถียรภาพของวงจรดีเป็นพิเศษ
- การทำให้ความถี่เปลี่ยนแปลงทำได้ยาก
- ใช้งานย่านความถี่สูง (แต่ไม่สูงมากนัก)

จากคุณสมบัติของวงจร Oscillator ทั้ง 3 แบบเราจึงเลือกใช้วงจร Oscillator ที่ใช้ LC

หลักการของวงจร Oscillator

อุปกรณ์ที่ใช้จะเป็น Transistor ซึ่งลักษณะของสัญญาณจะเป็นกระแส และมีการป้อนกลับของสัญญาณเข้าทาง Input ในรูปของกระแสเช่นกัน โดยมีโครงสร้างของวงจรดังรูป



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 2.10

จากรูปค่า A_1 และ β_1 จะเท่ากับ

$$A_1 = i_2 / i_1$$

$$\beta_1 = i_3 / i_2$$

การ Oscillate จะเริ่มเมื่อ $i_1 < i_3$

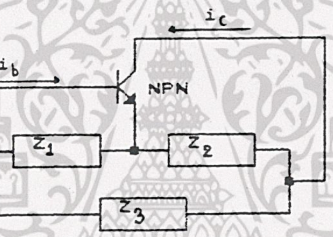
การ Oscillate จะเริ่มมีค่าคงที่เมื่อ $i_1 = i_3$

$A_1 * \beta_1 = i_3 / i_1 > 1$ (เริ่มการ Oscillate)

$A_1 * \beta_1 = i_3 / i_1 = 1$ (Oscillate แบบต่อเนื่อง)

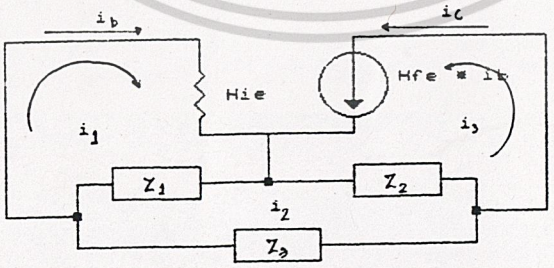
LC Oscillator

ลักษณะพื้นฐานของวงจร LC Oscillator แสดงได้ดังรูป



รูปที่ 3.11

เราสามารถเขียนเป็นวงจรสมมูลได้ดังรูป



รูปที่ 3.12

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปจะได้ว่า

$$(h_{ie} + Z_1)i_1 - (Z_1 * i_2) = 0$$

$$(-Z_1 * i_1) + (Z_1 + Z_2 + Z_3)i_2 + (Z_2 * i_3) = 0 \qquad (1)$$

เพราะว่า $i_1 = i_b$, $i_3 = i_c = h_{fe} * i_b$ ดังนั้นจากสมการ (1) จะได้เป็น

$$(h_{ie} + Z_1)i_b - (Z_1 * i_2) = 0$$

$$[(h_{fe} * Z_1) - Z_1]i_b + (Z_1 + Z_2 + Z_3)i_2 = 0 \qquad (2)$$

จาก สมการ (2) เราจะได้คำตอบก็ต่อเมื่อ $i_b = 0$ และ $i_2 = 0$ ดังนั้นสัมประสิทธิ์ Determinant จะต้องเป็น 0

$$\begin{bmatrix} h_{ie} + Z_1 & -Z_1 \\ (h_{fe} * Z_2) - Z_1 & Z_1 + Z_2 + Z_3 \end{bmatrix} \begin{bmatrix} i_b \\ i_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{vmatrix} h_{ie} + Z_1 & -Z_1 \\ (h_{fe} * Z_2) - Z_1 & Z_1 + Z_2 + Z_3 \end{vmatrix} = 0$$

$$(h_{ie} + Z_1)(Z_1 + Z_2 + Z_3) + Z_1 [(h_{fe} * Z_2) - Z_1] = 0$$

$$h_{ie} (Z_1 + Z_2 + Z_3) + Z_1 (Z_2 (1 + h_{fe}) + Z_3) = 0 \qquad (3)$$

ถ้าค่า Z_1 , Z_2 , Z_3 เป็นค่า Reactance จริงๆแล้วจะเท่ากับ jX_1 , jX_2 , jX_3 ตามลำดับ จากสมการที่ (3) จะมีค่าเป็น

$$jh_{ie} (X_1 + X_2 + X_3) - [X_1 * X_2 (1 + h_{fe})] - (X_1 * X_3) = 0$$

จาก Imaginary Part และ Real Part ของสมการบนจะได้ว่า

$$X_1 + X_2 + X_3 = 0 \qquad (4)$$

$$X_2 (1 + h_{fe}) + X_3 = 0 \qquad (5)$$

สมการที่ (4) จะแสดงข้อกำหนดของความถี่ของสัญญาณ

สมการที่ (5) จะแสดงข้อกำหนดศักดาของสัญญาณ

จาก สมการที่ (5) เราจะได้ว่า

$$X_2 = - X_3 / (1 + h_{fe}) \qquad (6)$$

จากสมการ (6) จะเห็นได้ว่าค่า X_2 และ X_3 จะเป็นค่า Reactance ที่ต่างชนิดกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

แทนสมการ (6) ลงในสมการ (4)

$$X_1 = -(h_{fe} * X_3) / (1 + h_{fe}) \quad (7)$$

จากสมการ (7) จะเห็นได้ว่าค่า X_1 และ X_3 จะเป็นค่า Reactance ที่ต่างชนิดกัน

จากสมการ (6) และ (7) จะได้ว่า

$$h_{fe} = X_1 / X_2$$

ดังนั้นผลจากสมการเหล่านี้สามารถสรุปได้ว่า

1. ค่า Reactance ของ X_1 , X_2 จะต้องเป็นคนละชนิดกับ X_3

   ดังนั้น ถ้า X_1 , X_2 เป็น L X_3 เป็น C เราจะเรียก Hartley Oscillator

   ถ้า X_1 , X_2 เป็น C X_3 เป็น L เราจะเรียก Colpits Oscillator

2. ค่า Reactance ต่างๆควรจะเป็นดังนี้

   $$|X_2| < |X_3| \quad , \quad |X_1| < |X_3|$$

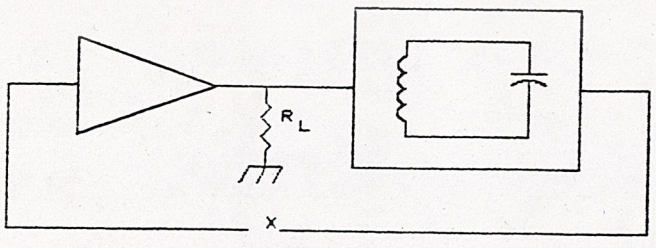
3. h_{fe} ควรมีค่าเป็น $h_{fe} = X_1 / X_2$

   ค่า h_{fe} นี้เป็นค่าที่การ Oscillate เป็นแบบอิ่มตัวแล้ว ถ้าในกรณีเริ่มการ Oscillate ค่า h_{fe} ควรจะมากกว่า X_1 / X_2

ส่วนประกอบที่จำเป็น สำหรับวงจร Oscillator

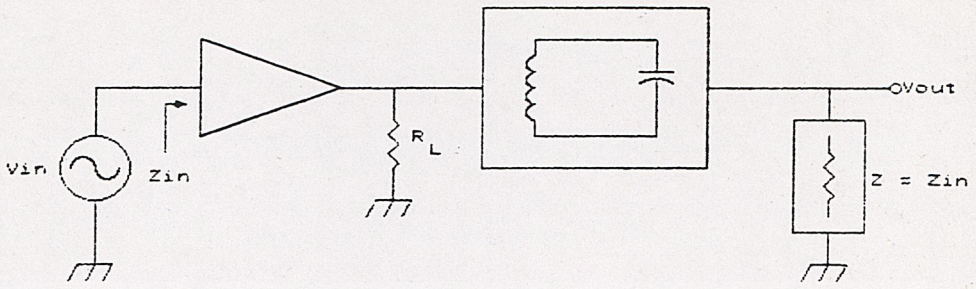
1. วงจรขยายสัญญาณ
2. วงจรเรโซเนเตอร์
3. ความต้านทานเอ้าท์พุท

ดังแสดงในรูป



(a)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(b)

รูปที่ 3.13 (a) แสดงส่วนประกอบที่จำเป็นของวงจร Oscillator

(b) แสดง รูปแบบวงจร Oscillator เมื่อทำการตัดลูปการป้อนกลับ

วงจรขยายสัญญาณ จะทำหน้าที่ขยายสัญญาณความถี่ที่ต้องการ

วงจรเรโซเนเตอร์ จะทำหน้าที่กำหนดความถี่ และยังเป็นวงจรป้อนกลับอีกด้วย

ถ้ามีสัญญาณรบกวนจากวงจรรีโซเนเตอร์ ซึ่งเกิดจากส่วนความต้านทาน สัญญาณรบกวนจะถูกขยายโดยบางส่วนจะปรากฏที่ความต้านทานโหลด และบางส่วนจะเข้าไปสู่วงจรรีโซเนเตอร์ซึ่งทำหน้าที่เป็นวงจรฟิลเตอร์ โดยสัญญาณรบกวนที่มีความถี่อยู่ในช่วงความถี่ผ่านของวงจงฟิลเตอร์จะสามารถผ่านออกไปได้ ส่วนสัญญาณรบกวนที่มีความถี่อื่นจะถูกลดทอนลงไป

สัญญาณ Output จากวงจรรีโซเนเตอร์จะถูกขยายโดยวงจรขยายสัญญาณ ถ้าสัญญาณมีขนาดมากกว่าระดับของสัญญาณเดิมและมีเฟสตรงกัน สัญญาณนี้จะถูกขยายไปเรื่อยๆจนเกิดการ Oscillate ถ้าสัญญาณมีเฟสไม่ตรงกัน สัญญาณเหล่านี้ก็ไม่สามารถเพิ่มขนาดได้ และจะไม่เกิดการ Oscillate

จากรูป 3.4 (b) โวลท์เตจที่ Output จะปรากฏที่ Input (Z_{in}) จะพิจารณาโดยแบ่งโวลท์เตจออกเป็น 2 ส่วน

-เฟสตรงกับ Input จะเปรียบเทียบกับ Input ถ้า อัตราขยายโวลท์เตจของลูปทั้งหมดเป็นบวกและมีค่ามากกว่า 1 การ Oscillate จะเกิดขึ้นได้ที่ความถี่เท่ากับความถี่ Input ถ้า อัตราขยายโวลท์เตจของลูปทั้งหมดน้อยกว่า 1 การ Oscillate จะไม่สามารถเกิดขึ้นได้ที่ความถี่นี้

-เฟสต่างกับ Input +90 องศา

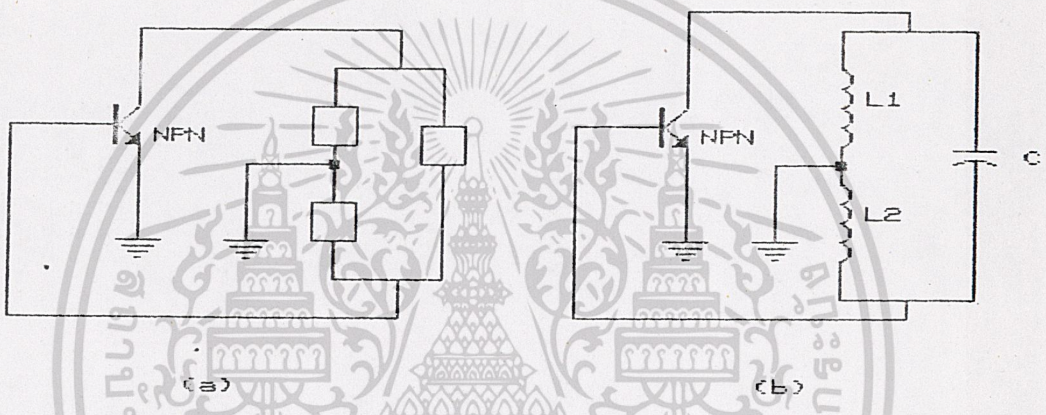
จากรูป 3.4 (a) ถ้ามีสัญญาณรบกวนเข้ามาที่ Input จะถูกขยาย และ ฟิลเตอร์ แล้วส่งกลับไปที่ Input อีก จะเห็นว่าขนาดของสัญญาณจะเพิ่มขึ้นไปเรื่อยๆถ้าวงจรขยายสัญญาณเป็นเชิงเส้น แต่ว่า

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และไม่อนุญาตให้ดัดแปลงเนื้อหา

มันเป็นไปไม่ได้ เพราะพลังงานนั้นจะต้องคงที่ ดังนั้นขบวนการที่จะทำให้ขนาดของสัญญาณคงที่วงจรขยายจะต้องเป็นอุปกรณ์ที่ไม่เป็นเชิงเส้น คืออัตราขยายของมันจะลดลงเมื่อขนาดของ สัญญาณInput มีค่ามากขึ้น และขนาดของสัญญาณ Output จะคงที่เมื่ออัตราการขยายของลูปมีค่าเท่ากับ 1

Hartley Oscillator แบบ Common Emitter

วงจรพื้นฐานของ Hartley Oscillator แสดงได้ดังรูป 3.5 ซึ่งทั้ง Hartley และColpits Oscillator นั้นจะมีหลักการเหมือนกัน เพียงแต่สลับที่กันระหว่าง L กับ C ซึ่งสำหรับ Common emitter นั้นจะขอกล่าวเพียง Hartley Oscillator เท่านั้น



รูปที่ 3.14 (a) แสดงวงจร Oscillator แบบ Common emitter

(b) แสดงวงจร Oscillator แบบ Hartley Common emitter

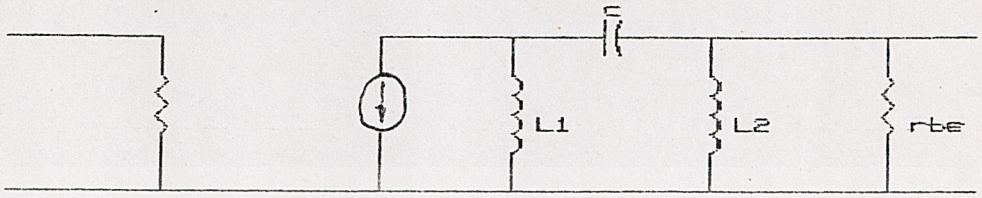
จากรูปที่ 3.1 นั้น สำหรับวงจร Oscillator นั้น จะเป็น Positive Feedback ซึ่งมีอัตราการขยายเป็น

$$A = A_1 / (1-A_1F)$$

จะเห็นได้ว่ามีเพียงเทอม A_1F เท่านั้นที่มี Phase Shift

จากรูปที่ 3.5 (b) นั้นเราสามารถเขียนวงจรสมมูลได้ดังรูป 3.6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.15 แสดงวงจรสมมูลของ รูปที่ 3.5 (b)

จากรูปที่ 3.6 สำหรับวงจร Common emitter Fully Bypass จะได้อัตราการขยายเป็น

$$A_i = \beta Z_L / r_{be} \quad (1)$$

$$Z_L = X_{L1} // (X_{c1} + Z) \quad (2)$$

$$Z = (r_{be} // X_{L2}) \quad (3)$$

นำสมการ (2) แทนลงในสมการ (1)

$$A_i = \frac{\beta}{r_{be}} \left| \frac{X_{L1} (X_c + Z)}{X_{L1} + (X_c + Z)} \right| \quad (4)$$

จาก Feedback Network

$$V_f = FV_o \quad (5)$$

จาก Voltage Devider

$$V_f = \frac{V_o Z}{Z + X_c} \quad (6)$$

จาก (5) = (6) เราจะได้ค่า F โดยที่

$$F = \frac{Z}{Z + X_c} \quad (7)$$

ดังนั้น จากสมการ (4) และ (7)

$$A_i F = \frac{\beta}{r_{be}} \left| \frac{X_{L1} * Z}{X_{L1} + X_c + Z} \right|$$

เอกสารนี้เป็นเอกสารสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$A_1F = \frac{\beta}{r_{be}} \left| \frac{jwL_1 * Z}{jwL_1 + (1/jwC) + Z} \right| \quad (8)$$

$$Z = \frac{r_{be} * jwL_2}{r_{be} + jwL_2} \quad (9)$$

จาก (9) และ (8) เราจะได้

$$A_1F = \frac{\beta}{r_{be}} \left| \frac{jwL_1 * (r_{be} * jwL_2 / r_{be} + jwL_2)}{jwL_1 + (1/jwC) + (r_{be} * jwL_2 / r_{be} + jwL_2)} \right|$$

$$A_1F = \frac{\beta}{1- w^2\{ [(L_2)^2 - (L_1)^2]/2 \} + (wr_{be}L_1C + wr_{be}L_2C + w^3r_{be})j} \quad (10)$$

Phase Shift ของ Loop = 0 ที่ความถี่ที่ทำให้เกิดการ Oscillate $w = W_o$ ดังนั้นในสมการ (10) j term = 0 นั่นคือมุมของ $A_1F = 0$

มุมของ $A_1F = \tan^{-1} \frac{\text{imaginary part}}{\text{real part}} = 0$

$$(wr_{be}L_1C + wr_{be}L_2C + w^3r_{be})j = 0$$

$$W_o = 1 / \sqrt{LC} \quad (11)$$

$$L = L_1 + L_2$$

จากสมการ (10) ที่ความถี่ที่ทำให้เกิดการ Oscillate

$$A_1F = \frac{\beta}{1- (W_o)^2\{ [(L_2)^2 - (L_1)^2]/2 \}}$$

แทนค่าด้วยสมการ (11)

$$\beta_{(wo)} \geq L_1 / L_2 \quad (12)$$

ผลจากสมการเหล่านี้สรุปได้ว่า

1. เราสามารถกำหนดความถี่ของการ Oscillate ได้โดยการเปลี่ยนค่า L , C จากสมการ (11)

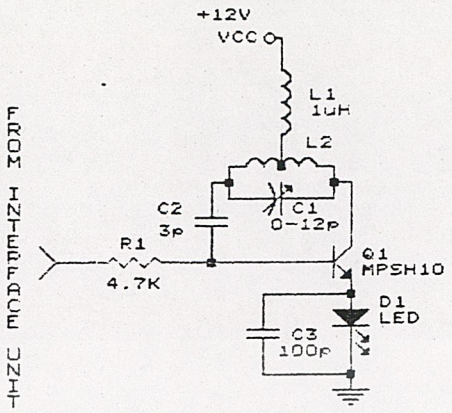
2. อัตราการขยายต่ำสุดของการ Oscillate หาได้จากสมการ (12)

สำหรับวงจร Colpits Oscillator นั้นจะได้สมการ (11),(12) เหมือนกัน โดยที่

$$C = C_1C_2 / C_1 + C_2$$

$$\beta_{(wo)} \geq C_1 / C_2$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.16 แสดงรูปแบบของวงจร Oscillator ที่ใช้

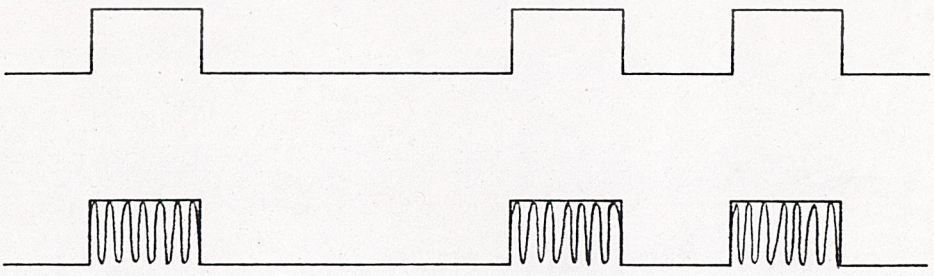
- Pulse Modulator

จากวงจรรูปที่ 3.7 จะเห็นว่า

-เมื่อมีสัญญาณ (Input logic 1) เข้าสู่วงจร Oscillator Oscillator ก็จะ Oscillate ที่ความถี่ 300 MHz

-เมื่อไม่มีสัญญาณ (Input logic 0) เข้าสู่วงจร Oscillator Oscillator ก็จะหยุดการทำงาน

จากการทำงานของวงจร Oscillator ที่กล่าวมานั้นคือหลักการของ Pulse Modulator โดยจะมีตัวอย่างรูปแบบของสัญญาณดังรูป



รูปที่ 3.17 (a) แสดงรูปแบบของสัญญาณ Input ของวงจร Oscillator
(b) แสดงรูปแบบของสัญญาณ Output ของวงจร Oscillator

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น

- Antenna

ในที่นี้ใช้การ Radiate ออกจากวงจร Oscillator เลยเพื่อให้เกิดความง่าย และ ได้วงจรขนาดเล็ก

จากรูปที่ 3.7 L_{RFC} (L_1) ในวงจรใส่ไว้เพื่อ

1. เพื่อให้กระแสไหลผ่านสู่วงจรได้สะดวก เพราะว่าความต้านทานของ L จะต่ำมากเมื่อความถี่ต่ำ

2. สัญญาณจากวงจรจะไม่สามารถไหลผ่าน L ขึ้นไปได้ เพราะว่าความต้านทานของ L จะสูงมากเมื่อความถี่สูง

3. เอาท์พุทโวลเตจจะ Swing ได้เป็น 2 เท่าของไฟเลี้ยง เพราะว่าตัว L มี Storage Energy

L_{RFC} จะไม่มีผลต่อวงจร เพราะว่าค่า L_{RFC} ที่ใช้ในวงจร ใหญ่กว่า L ที่ทำให้เกิดการ Oscillate มาก

C_2 ในวงจรใส่ไว้เพื่อ กันไฟเลี้ยงรั่วผ่าน L กลับมา Biass Transistor ทำให้การ Biass ผิด

C_3 C Bypass ที่ขา Emitter ต่อไว้เพื่อใช้ในการเพิ่ม Gain ที่ความถี่สูง

การคำนวณหาค่า L,C ที่ทำให้ได้ความถี่ที่ต้องการ

กำหนดค่า C เป็นแบบปรับค่าได้ 3 - 12 PF เพื่อให้ง่ายต่อการปรับหาความถี่ที่ต้องการ

จากสมการ (11)

$$f = 1 / 2\pi\sqrt{LC}$$

$$300 * 10^{6} = 1 / 2\pi\sqrt{L * (8 * 10^{-12})}$$

$$L = 18.76 \text{ nH}$$

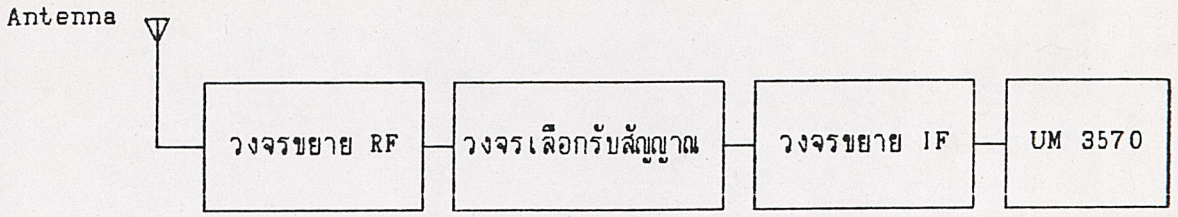
จะเห็นได้ว่าค่า L ที่ได้เป็นค่าที่น้อยมากไม่สามารถสร้างขึ้นจากขดลวดได้ จึงต้องทำ L บนลายทองแดง

การออกแบบในทางปฏิบัติที่กล่าวมาทั้งหมดนั้นเป็นการประมาณค่า ดังนั้นในการใช้งานจริงจะต้องมีการปรับแต่งวงจรเพื่อให้ได้ผลที่ต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<u>ภาครับของเครื่องรับความถี่วิทยุ</u>

หลักการทำงานพื้นฐานของเครื่องรับวิทยุ สามารถเขียนเป็น Block diagram ได้ดังรูป



- Antenna

เป็นส่วนสำคัญของทั้งเครื่องรับ และเครื่องส่ง ทำหน้าที่แผ่คลื่นจากเครื่องส่งออกสู่อากาศ และรับคลื่นเข้าสู่เครื่องรับ

- วงจรขยาย RF

ทำหน้าที่ขยายสัญญาณความถี่สูง จากสายอากาศ ให้มีขนาดของสัญญาณสูงพอ

- วงจรเลือกรับสัญญาณ

เป็นส่วนเลือกรับสัญญาณที่ได้กำหนดไว้

- วงจรขยาย IF

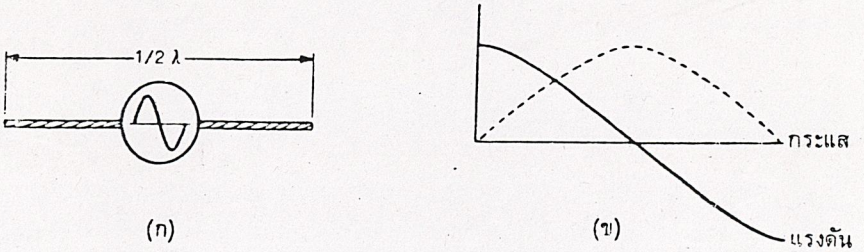
ทำหน้าที่ขยายสัญญาณ จากวงจรเลือกรับสัญญาณ ให้มีขนาดของสัญญาณสูงพอ

- Antenna

หลักการของสายอากาศ

สายอากาศเสมือนกับวงจรไฟฟ้า ที่ประกอบด้วย L และ C

ในทางทฤษฎีแล้วสายอากาศ และ สายอากาศรับ จะมีคุณสมบัติเหมือนกัน ดังนั้นเราสามารถใช้สายอากาศทำหน้าที่ได้ทั้งส่ง และ รับ



รูปที่ 3.18 (a) แสดงสายอากาศไดโพลชนิด Half Wave

(b) แสดงแรงดัน และกระแส บนสายอากาศไดโพลชนิด Half Wave

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูป (b) จะเห็นว่าปลายทั้งสองของสายอากาศเสมือนเปิดวงจรแรงดันจึงมากที่สุด และที่จุดกึ่งกลางของสายอากาศ (Feed Point)จะมีกระแสมากที่สุด และแรงดันต่ำสุด ดังนั้น Impedance ของไดโพลชนิด Half Wave น่าจะมีค่าเป็น 0 แต่ในความจริงแล้ว Impedance จะมีค่าคงที่ค่าหนึ่ง เนื่องจากพลังงานบางส่วนสูญเสียในการแผ่คลื่นออกไปและไม่สะท้อนกลับ

- เนื่องจากปลายสายอากาศทั้งสองข้างมีแรงดันสูง จึงทำให้เกิดสนามไฟฟ้าที่ปลายทั้งสอง

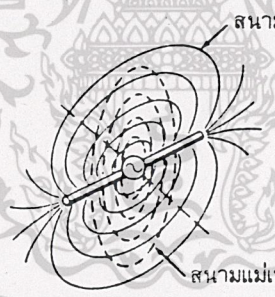
- กระแสที่ไหลในสายอากาศทำให้เกิดสนามแม่เหล็กขึ้น ณ.บริเวณ Feed Point สนามแม่เหล็กจะแรงที่สุด

โดยที่คลื่นดังกล่าวจะเดินทางไปเรื่อยๆ แม้ว่ากระแส และ แรงดัน จะหยุดป้อนให้สายอากาศแล้วก็ตาม

ความแรงของสนามจะขึ้นอยู่กับระยะห่างจากเครื่องส่ง และ กำลังของเครื่องส่ง

- ความแรงสนามเป็นค่าแรงดันที่เหนี่ยวนำบนสายอากาศ แต่เนื่องจาก $P = V^2 / R$ ดังนั้น ความแรงของสนาม α รากที่ 2 ของกำลังของเครื่องส่ง

- ความแรงของสนามจะลดลงครึ่งหนึ่ง เมื่อเพิ่มระยะทางขึ้น 2 เท่า



รูปที่ 3.19 สนามแม่เหล็ก และ สนามไฟฟ้า ที่ปรากฏรอบ สายอากาศไดโพล ชนิด Half Wave

จากรูป ทิศทางของสนามไฟฟ้าจะเป็นตัวกำหนด ทิศทางของ Polalization ซึ่งจะเป็นตัวกำหนด ทิศทางของสายอากาศตัวรับ

ความต้านทานการแพร่คลื่น

ความต้านทานในสายอากาศไม่สามารถหาได้โดยใช้ โอห์มมิเตอร์วัดคร่อม เนื่องจากโอห์มมิเตอร์นั้นจะวัดค่าความต้านทาน DC

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น

ความต้านทานการแพร่คลื่นของสายอากาศเป็นความต้านทานสมมติ โดยคิดจาก กำลังส่งที่สายอา-

กาศสามารถเปลี่ยนเป็นคลื่นส่งออกไป

$$R = P / I^2$$

สายอากาศ ชนิด Quater Wave

การวางสายอากาศที่มีความยาว $\lambda /4$ ในแนวดิ่ง บนระนาบตัวนำที่เป็นกราวด์ ผลลัพธ์ที่ได้จะเปรียบเสมือนกับใช้สายอากาศไดโพลชนิด Half Wave ตั้งในแนวดิ่ง เพราะว่าระนาบกราวด์จะเปรียบเสมือนกระจกที่ทำให้เกิดสายอากาศอีกเส้นหนึ่งยาว $\lambda/4$ เราจึงสามารถลดความยาวของสายอากาศได้ และจะมีค่า Input Impedance ประมาณครึ่งหนึ่งของสายอากาศไดโพลชนิด Half Wave

การคำนวณความยาวของสายอากาศไดโพล ชนิด Half Wave

คำนวณได้จากสูตร

$$\lambda = 3 * 10^8 / f$$

สำหรับกรณีของสายอากาศ ปลายของเส้นลวดเสมือนมีความยาวทางไฟฟ้ามากกว่าปกติ เนื่องจากเสมือนมีตัวเก็บประจุต่อที่ปลายสายดังนั้น สูตรการคำนวณจึงต้องแก้ไขเป็น

$$\lambda/2 = 142.5 / f \text{ (MHz)} \quad \text{หน่วยเมตร}$$

การคำนวณความยาวของสายอากาศของวงจรจริง

จากสมการ

$$\lambda/2 = 142.5 / 300 = 0.475 \text{ เมตร}$$

เนื่องจากความยาวของสายอากาศที่ได้มากเกินไป เราจึงใช้สายอากาศชนิด Quater Wave โดยใช้ระนาบกราวด์ ดังนั้น ความยาวของสายอากาศของวงจรจริง

$$\lambda/4 = 0.475/2 = 23.75 \text{ Cm}$$

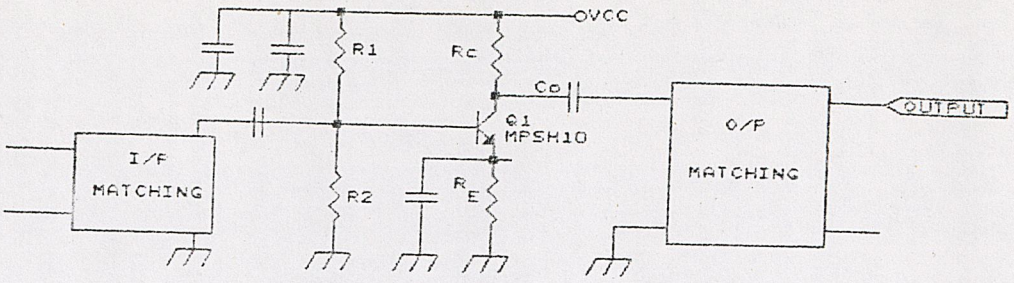
- วงจรขยายสัญญาณ RF

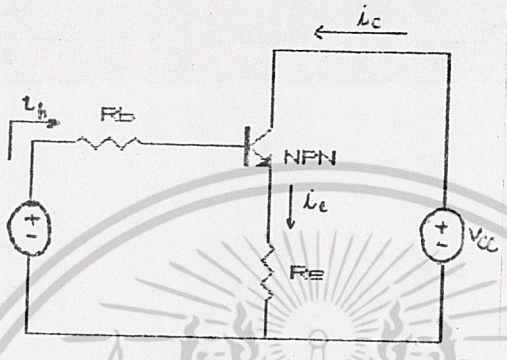
การคำนวณ DC BIAS ของวงจรขยายสัญญาณความถี่สูง

การคำนวณ DC BIAS ให้กับวงจรขยายสัญญาณความถี่สูงจำเป็นต้องคำนึงถึงเสถียรภาพของวงจรอย่างมาก โดยเฉพาะค่า R_E ในวงจร เพราะอาจทำให้วงจรเกิดการไม่เสถียรภาพขึ้นได้ หรือทำให้เกิดสัญญาณรบกวนในวงจรมากขึ้น ในการออกแบบจึงต้องพิจารณาช่วงอุณหภูมิที่ใช้งาน วงจรขยายสัญญาณความถี่สูงเป็นหลัก เพราะอุณหภูมิจะมีผลต่อพารามิเตอร์ต่างๆของทรานซิสเตอร์ เช่น ค่า β, V_{BE} และ I_{CBO}

วงจรที่นำมาใช้งานในที่นี้มีลักษณะดังรูปต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้





รูปที่ 3.20

จากวงจรซึ่งเป็นวงจรขยายสัญญาณ Class A จะได้

$$I_C = \beta I_b + (1+\beta) I_{CO}$$

ดังนั้น

$$I_B = \frac{I_C}{\beta} - \frac{(1+\beta) I_{CO}}{\beta} \quad (1)$$

และ

$$V_{BB} = I_C R_E + I_B (R_B + R_E) + V_{BE} \quad (2)$$

แทนค่า I_B ลงใน (2) แล้วได้

$$I_C = \frac{\beta(V_{BB} - V_{BE}) + (1+\beta)(R_B + R_C) I_{CO}}{[R_B + (1+\beta)R_E][R_B + (1+\beta)R_E]} \quad (3)$$

จากสมการจะเห็นว่า เสถียรภาพของการไบอัสขึ้นอยู่กับค่า β ของทรานซิสเตอร์ โดยที่ค่า β มีค่าไม่คงที่ ขึ้นอยู่กับอุณหภูมิ และ โครงสร้างของทรานซิสเตอร์แต่ละตัว ดังนั้นถ้าต้องการให้วงจรมีเสถียรภาพดีก็ต้องทำให้การไบอัสวงจรขึ้นกับค่า β น้อยที่สุด ซึ่งจากสมการ (3) จะเห็นได้ว่าทำได้โดยให้ R_E มีค่ามากๆ และ R_b มีค่าน้อยๆ จึงจะได้ $(1+\beta)R_E >> R_B$ และให้ $V_{BB} >> V_{BE}$ เพื่อให้กระแสคอลเลคเตอร์ไม่ขึ้นกับค่า V_{BE} ของทรานซิสเตอร์สมการ (3) จึงเขียนใหม่ได้เป็น

$$I_C = \frac{\beta(V_{BB}) + R_B + R_E I_{CO}}{(1+\beta)R_E \quad R_B}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพื่อให้ง่ายแก่การคำนวณจึงให้ $I_{co} = 0$ เพราะค่า I_{co} มีค่าน้อยมากและค่าพารามิเตอร์ของทรานซิสเตอร์ที่อุณหภูมิต่างๆสามารถหาค่าได้ดังนี้

พารามิเตอร์	ซิลิกอนทรานซิสเตอร์	เจอรมันเนียมทรานซิสเตอร์
$\beta(25 + \Delta T)$	$\beta(25)(1+\Delta T/80\ C)$	$\beta(25)(1+\Delta T/60\ C)$
$V_{BE}(25 + \Delta T)$	$(0.60+0.002\Delta T)$	$(0.20 + 0.002\Delta T)$

โดยให้ $\Delta T = T_2 - T_1$ และค่ากระแสคอลเลคเตอร์มีการเปลี่ยนแปลงในช่วง I_{c1} ถึง I_{c2} โดยมีค่าเท่ากับ I_{c1} ที่อุณหภูมิ T_1 และมีค่า I_{c2} ที่อุณหภูมิ T_2 และให้ $T_2 > T_1$

ค่า I_c สามารถกำหนดได้โดยดูจาก Data Sheet ของทรานซิสเตอร์ โดยพิจารณาจากค่าสูงสุดของกระแสคอลเลคเตอร์ และค่า Dissipation Rating ของทรานซิสเตอร์ และถ้าต้องการให้มี Noise Figure ต่ำๆ ก็พิจารณาได้จากกราฟระหว่างอินพุทอิมพีแดนซ์ของวงจร และ ค่ากระแสคอลเลคเตอร์ที่ Noise figure ค่าต่างๆ ส่วนค่า I_{c2} ก็กำหนดได้โดยให้มีค่าต่างจาก I_{c1} ประมาณ 10 - 20% ของ I_{c1} ขึ้นว่ากับว่าเราต้องการให้วงจรมีเสถียรภาพมากน้อยแค่ไหน ส่วนค่า $\beta(25)$ เป็นค่าβที่อุณหภูมิ 25 C ซึ่งสามารถดูได้จาก Data Sheet ของทรานซิสเตอร์เช่นกัน

จากค่า I_{c1} และ I_{c2} นำมาหาค่าเฉลี่ยของกระแสคอลเลคเตอร์ (I_{cn}) และค่าเปอร์-เซนต์ การเปลี่ยนแปลงของกระแสคอลเลคเตอร์ (P) โดย

$$I_{cn} = (I_{c1} + I_{c2})/2 \qquad (4)$$

$$I_c = I_{cn} + P\%/I_{cn}/100 \qquad (5)$$

เมื่อ $I_{c1} > I_{c2}$ จะได้

$$I_{c1} = I_{cn}(1-P)$$

และ $$I_{c2} = I_{cn}(1+P)$$

ให้ $$I_{c1}/I_{c2} = N$$

จะได้ $$N = \frac{1 + P}{1 - P}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และ $P = \frac{N-1}{N+1}$

จากสมการ (3) จะได้

$$V_{BB} = V_{BE} + \frac{I_C}{\beta}[R_B + (1+\beta)R_E]$$

ที่อุณหภูมิสูงสุด T_2 จะได้

$$V_{BB} = V_{BE2} + \frac{I_{C2}}{\beta}[R_B + (1+\beta_2)R_E] \quad (6)$$

และที่อุณหภูมิต่ำสุด T_1 จะได้

$$V_{BB} = V_{BE1} + \frac{I_{C1}}{\beta}[R_B + (1+\beta_2)R_E] \quad (7)$$

(7)-(6) และหาค่า R_B จะได้

$$R_B = \frac{R_E\{[(1+\beta_2)/\beta_2]I_{C2} - [(1-\beta_1)/\beta_1]I_{C1}\} - (V_{BE1} - V_{BE2})}{I_{C1}/\beta_1 - I_{C2}/\beta_2} \quad (8)$$

จากสมการจัดให้อยู่ในรูปอย่างง่ายจะได้

$$R_B = AR_E - B \quad (9)$$

โดยที่

$$A = \frac{[(1+\beta_2)/\beta_2]I_{C2} - [(1-\beta_1)/\beta_1]I_{C1}}{I_{C1}/\beta_1 - I_{C2}/\beta_2} \quad (10)$$

$$B = \frac{V_{BE1} - V_{BE2}}{I_{C1}/\beta_1 - I_{C2}/\beta_2} \quad (11)$$

หรือ

$$A = \frac{N(1+\beta_2) - (\beta_2/\beta_1)(1+\beta_1)}{\beta_2/\beta_1 - N}$$

$$B = \frac{\beta_2 (V_{BE1} - V_{BE2})}{I_{C1}(\beta_2/\beta_1 - N)}$$

แทนค่าสมการ (9) - (10) ลงในสมการ (6) หรือ (7) จะได้

$$V_{BB} = F + GR_E \quad (14)$$

$$F = \frac{I_{C1}V_{BE2}/\beta_1 - I_{C2}V_{BE1}/\beta_2}{I_{C1}/\beta_1 - I_{C2}/\beta_2} \quad (15)$$

$$G = \frac{I_{C1}I_{C2}(\beta_2 - \beta_1)}{} \quad (16)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\beta_1\beta_2 \left(I_{C1}/\beta_1 - I_{C2}/\beta_2\right)$$

เสถียรภาพของวงจรจะดีที่สุดเมื่อ $R_{Emin} = B/A$ มีค่าเป็นบวก

จากสมการ (8) แทนค่า $I_{C2} = NI_{C1}$ จะได้

$$N = \frac{[R_B + (1 + \beta_1)R_E]/\beta_1 + (V_{BE1} - V_{BE2})/I_{C1}}{[R_B + (1+\beta_2)R_E]/\beta_2} \qquad (17)$$

จากสมการ (17) ทำให้เราทราบว่าเสถียรภาพของวงจรเปลี่ยนไปอย่างไรเมื่อเราใช้ R_E และ R_B ค่าต่างๆ เมื่อ I_{C1} คงที่ โดยถ้า $N > N_{min}$ ก็แสดงว่าวงจรจะมีเสถียรภาพได้

โดยค่า N_{min} สามารถหาได้เมื่อ $R_B = 0$ และให้ $(V_{BE1} - V_{BE2})/I_{C1}$ มีค่าเข้าใกล้ 0 จะได้

$$N_{min} = \frac{\beta_2 (1 + \beta_1)}{\beta_1 (1 + \beta_2)} \qquad (18)$$

$$P_{min} = \frac{N_{min} - 1}{N_{min} + 1}$$

จากที่กล่าวมาทั้งหมดสามารถสรุปเป็นขั้นตอนการออกแบบได้ดังนี้

1. กำหนดอุณหภูมิใช้งานต่ำสุดเป็น T_1 และอุณหภูมิสูงสุดเป็น T_2 และหาค่า β และค่า V_{BE} ที่อุณหภูมิ T_1 และ T_2

2. กำหนดค่ากระแสคอลเลคเตอร์ I_{C2} และ ค่ากระแสคอลเลคเตอร์ที่สามารถเปลี่ยนแปลงได้ I_{C2} มีค่ามากกว่า I_{C1} 10 - 20 %

3. หาค่า N_{min} จากสมการ (18) และหาค่า N จากสมการ $N = I_{C1}/I_{C2}$ ถ้า N มีค่ามากกว่า N_{min} ก็แสดงว่าวงจรอาจมีเสถียรภาพได้ และถ้าต้องการทราบว่าวงจรมีเสถียรภาพกี่เปอร์เซนต์ก็หาได้จากค่า P

4. ใช้สมการ (9) - (11) หาความสัมพันธ์ของ $R_B = AR_E - B$ และหาค่า R_{Emin} โดยให้ $R_B = 0$ จะได้ $R_{Emin} = B/A$

5. เลือกค่า R_E ให้มีค่ามากกว่า R_{Emin} แล้วหาค่า R_B จากสมการ (9)

6. หาค่า V_{BB} จากสมการ (14) - (16)

7. จากค่า R_{BB} หาค่า R_1 และ R_2 ได้ดังต่อไปนี้

$$I_{BB} = V_{BB} / R_2$$

$$(I_{BB} + I_B)R_1 = V_{CC} - V_{BB}$$

แทนค่า I_{BB}

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$[(V_{BB} / R_2) + I_B] R_1 = V_{CC} - V_{BB}$$

$$R_1 = \frac{(V_{CC} - V_{BB})R_2}{V_{BB} + I_B R_2}$$

$$R_{BB} = R_1 // R_2 = (R_1 R_2)/(R_1 + R_2)$$

$$R_1 = R_{BB} (1 + R_1/R_2)$$

แทนค่า R_1

$$\frac{(V_{CC} - V_{BB})R_2}{V_{BB} + I_B R_2} = \frac{R_{BB} [1 + (V_{CC} - V_{BB})R_2]}{(V_{BB} + I_B R_2)R_2}$$

$$(V_{CC} - V_{BB})R_2 = (V_{BB} + I_B R_2)R_{BB} + R_{BB}(V_{CC} - V_{BB})$$

$$= (V_{CC} + I_B R_2)R_{BB}$$

$$(V_{CC} - V_{BB} - I_B R_{BB})R_2 = R_{BB} V_{CC}$$

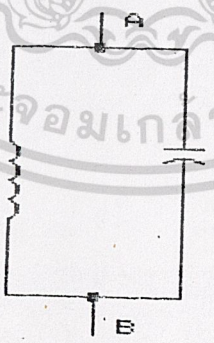
ดังนั้น

$$R_2 = \frac{R_{BB} V_{CC}}{V_{CC} - V_{BB} - I_B R_{BB}} \qquad (19)$$

$$R_1 = R_{BB} R_2 / (R_2 - R_{BB}) \qquad (20)$$

- วงจรเลือกรับสัญญาณ , Demodulate

วงจรนี้ประกอบด้วย Tank Circuit ซึ่งประกอบด้วย ตัวเก็บประจุ และ ตัวเหนี่ยวนำ ดังรูป



รูปที่ 3.21

จากวงจร ที่ความถี่ Resonance จะทำให้ $X_C = X_L$ ซึ่งเราสามารถหาความถี่นี้ได้จากสมการ

$$W_o = 1/\sqrt{LC}$$

ณ.ที่ความถี่นี้จะทำให้ Impedance ระหว่างจุด A กับ B มีค่าสูงสุด ทำให้กระแสไหลผ่านน้อยที่สุด

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การคำนวณวงจรเลือกความถี่

จากสมการ

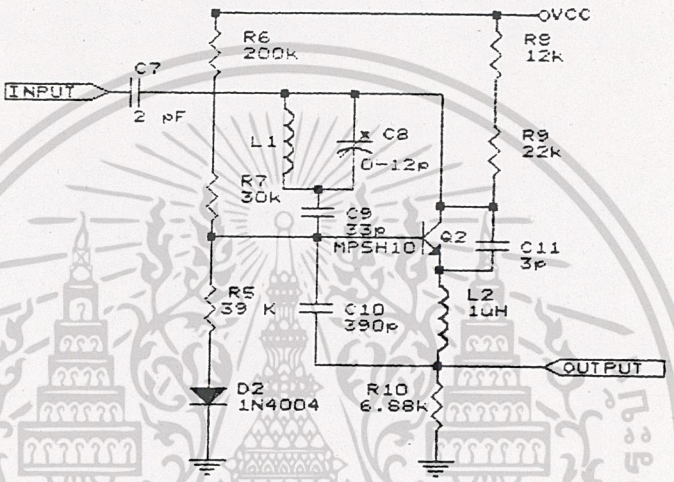
$$W_o = 1/\sqrt{LC}$$

เราสามารถคำนวณหาค่า L ได้โดย

$$L = 1/ (4 * 10^{-12})(2 * 300 * 10^{6})^2$$

$$= 70 \text{ nH}$$

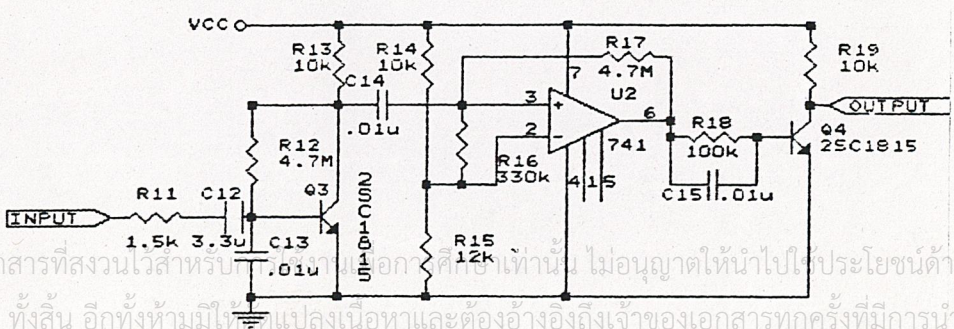
ในวงจรจริงเราใช้ แกนเฟอไรท์พันลวดไว้ประมาณ 1 รอบครึ่ง



รูปที่ 3.22 แสดงวงจรเลือกรับสัญญาณ และ Demodulate จริงที่ใช้งาน

จากวงจร จะเห็นได้ว่า Tank Circuit จะเป็นตัวเลือกความถี่ และ กำหนดกระแสเข้าสู่ Transistor Q_2

- วงจรขยายสัญญาณ IF



รูปที่ 3.23

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

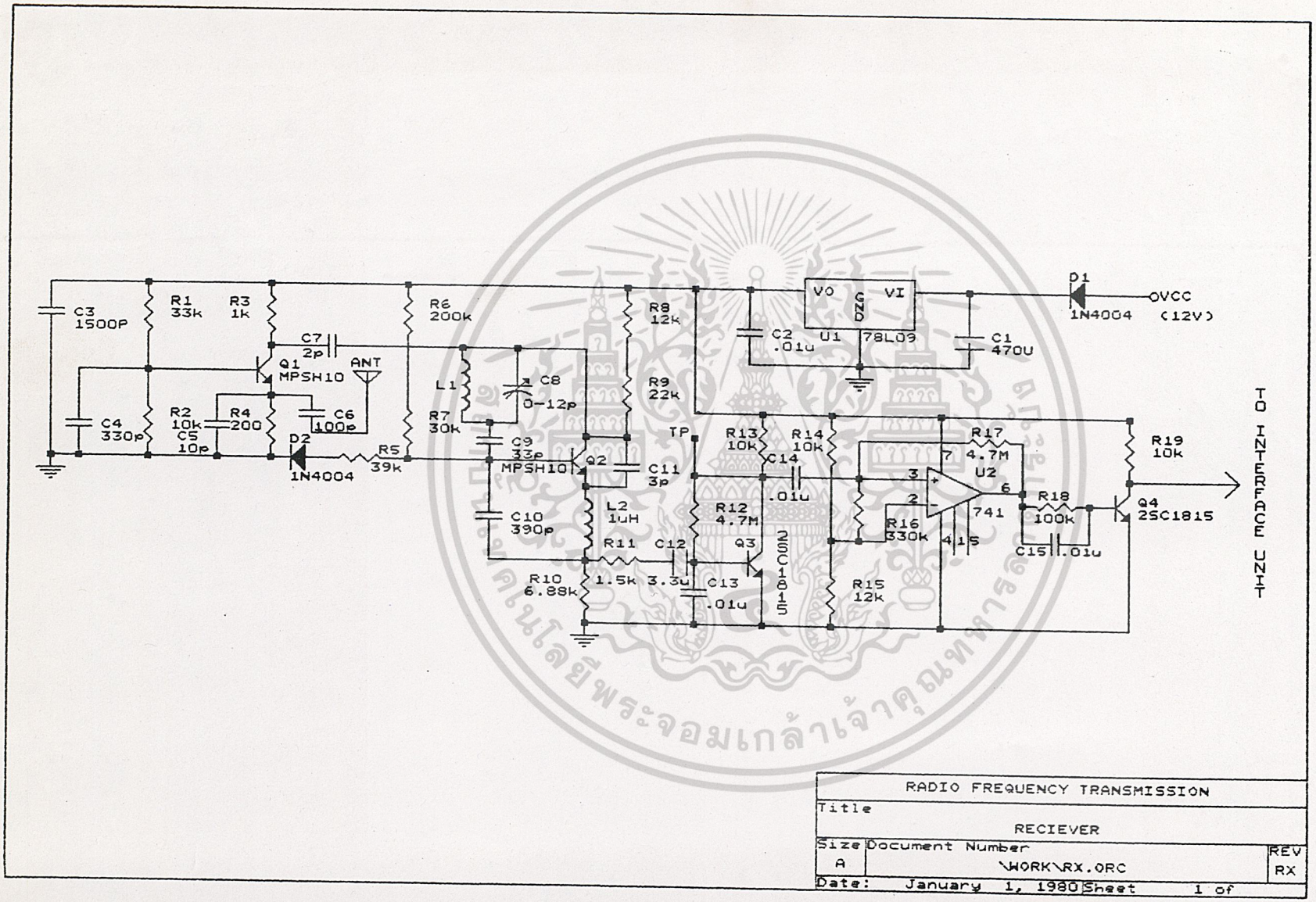
จากวงจรประกอบด้วย

-วงจรขยายสัญญาณความถี่ต่ำ ทำหน้าที่ขยายสัญญาณที่ได้จากวงจรเลือกรับสัญญาณ Demodulate ที่มีความถี่ต่ำ ให้มีขนาดของสัญญาณสูงขึ้น

-วงจร Comparator ทำหน้าที่ในการเปรียบเทียบแรงดันจาก วงจรขยายสัญญาณความถี่ต่ำ กับแรงดันอ้างอิง เพื่อให้สัญญาณเป็นรูป Pulse มากขึ้น หรือกำจัด Noise ที่ปนมากับสัญญาณข้อมูล

-วงจรกลับเฟส และ ขยายสัญญาณ ทำหน้าที่กลับเฟสของสัญญาณให้ตรงกับสัญญาณข้อมูล หรือ ให้ได้เฟสของสัญญาณที่ต้องการนำไปใช้ในวงจรต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



RADIO FREQUENCY TRANSMISSION		
Title: RECIEVER		
Size: A	Document Number: \WORK\RX.ORC	REV: RX
Date: January 1, 1980	Sheet 1 of	

การพัฒนาวงจรส่วนภาครับ - ส่งวิทยุ (Transciever)

จากการปรับปรุงวงจรในส่วนภาครับ-ส่งวิทยุที่ได้พัฒนาไว้ในเทอมแรกนั้น ประสพกับปัญหาดังนี้ คือ

- วงจร Oscillator ของส่วนภาคส่ง มีกำลังในการส่งน้อยมาก การพัฒนาวงจรในส่วน Power Amplifier ในย่านความถี่สูงถึง UHF ทำได้ยาก เนื่องจากอุปกรณ์ที่ใช้หาได้ยาก และมีราคาแพงมาก

- การออกแบบวงจรในย่านความถี่สูงถึง UHF Transmission line ต้องใช้ Microstrip Technics ซึ่งขึ้นอยู่กับคุณสมบัติทางกายภาพของ PCB อยู่มาก ซึ่งการออกแบบ และ การควบคุมลายวงจรบน PCB ทำได้ยาก

- การส่งสัญญาณข้อมูลแบบ Pulse Modulation ออกสู่อากาศโดยตรงจะทำให้เกิดความผิดพลาดของข้อมูลสูง และ มีสัญญาณรบกวนสูงด้วย

จากเหตุผลดังกล่าวในการแก้ปัญหาที่เกิดขึ้น จึงได้ออกแบบวงจรส่วนภาครับ-ส่งวิทยุใหม่

ภาคส่ง Modulate สัญญาณข้อมูลออกเป็นสัญญาณเสียง 2 ความถี่

. Modulate สัญญาณเสียงแบบ FM ย่านความถี่แคบ

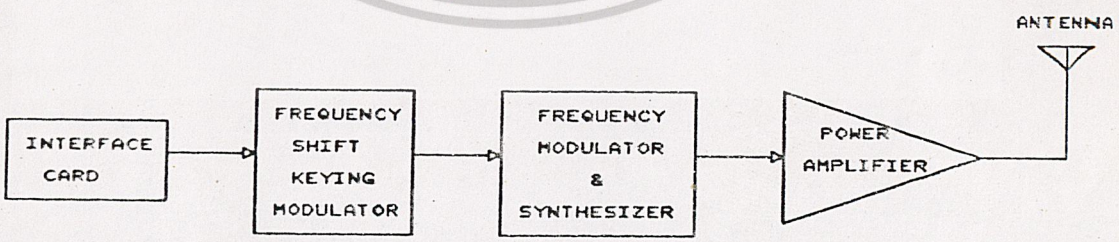
. ใช้ Phase Lock Loop สังเคราะห์ความถี่คลื่นพาหะในช่วงความถี่ VHF ขนาด 88-112 MHz เปลี่ยนช่วงความถี่ได้ โดยมีความกว้างของแต่ละช่องเท่ากับ 200 KHz

ใช้วงจรขยายสัญญาณ Class A และ Class C กำลัง 2 Watts

ภาครับ เป็นเครื่องรับวิทยุแบบ FM

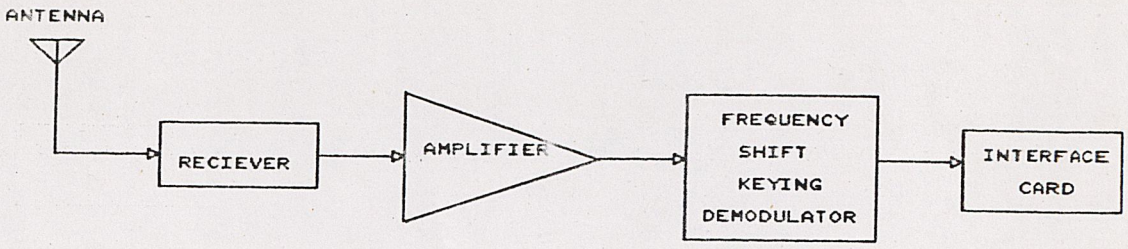
Demodulate สัญญาณเสียงออกเป็น สัญญาณข้อมูล 2 ระดับ

ซึ่งวงจรพื้นฐานของเครื่องรับ-ส่งวิทยุ แสดงให้เห็นโดย Block diagram ดังนี้



รูปที่ 3.24 แสดงโครงสร้างของเครื่องส่งวิทยุ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.25 แสดงโครงสร้างของเครื่องรับวิทยุ

วงจร FSK Modulator

ใช้ XR-2206 เป็นตัว Modulate สัญญาณแบบ FSK โดยใช้ช่วง Bandwidth เท่าที่จำเป็นเท่านั้น จากวงจรดังรูป 3.26

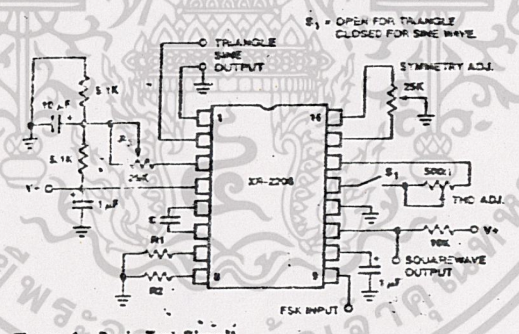


Figure 1. Basic Test Circuit.

รูปที่ 3.26 แสดงวงจรพื้นฐานของ XR-2206 FSK Modulator

- ความถี่ของ Upper และ Lower Frequency

เนื่องจากเราต้องการช่วง Bandwidth ที่น้อยที่สุด Lower Frequency จะต้องมีค่าอย่างน้อย 55% ของ Upper Frequency

สำหรับ Buad Rate ที่กำหนดมาให้นั้นสามารถหา ความถี่ Upper และ Lower โดยอาศัยความสัมพันธ์จาก

$$\frac{\text{Upper - Lower Frequency Difference (Hz)}}{\text{Maximum Data Rate (Buad)}} >= 83\ \%$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การหาค่าของอุปกรณ์ในวงจร ในรูป 3.26

ความถี่ของ Upper และ Lower Frequency สามารถกำหนดได้โดย

Upper Frequency = $1/R_1C$

Lower Frequency = $1/R_2C$

จากรูปที่ 3.27 R_1 R_2 ที่เหมาะสมอยู่ในช่วง 10 - 100 K กำหนดค่า C หาค่า R_1 R_2 ที่เหมาะสม

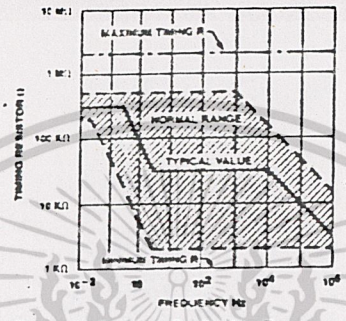


Figure 4. R versus Oscillation Frequency.

รูปที่ 3.27 กราฟแสดงค่า R_1 R_2 และความถี่ ที่สามารถปรับได้

วงจร FSK Demodulator

ใช้ XR-2211 เป็นตัว Demodulate สัญญาณแบบ FSK จากวงจรดังรูป 3.28

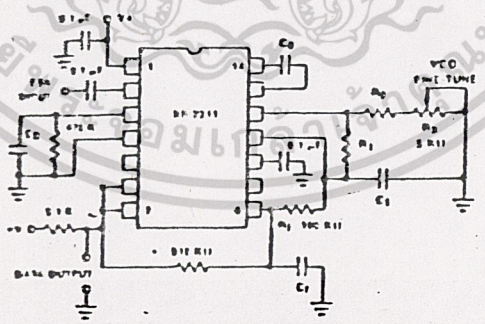


Figure 10. External Connections for FSK Demodulation with Carrier Detect Capability

รูปที่ 3.28 แสดงวงจรพื้นฐานของ XR-2211 FSK Demodulator

- ค่าความถี่กลางของ Upper และ Lower Frequency

สามารถกำหนดโดย

$f_o = (f_L + f_H)/2 = [1/(\quad)C_o]$

เอกสารนี้จัดทำขึ้นไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากตาราง Electrical Characteristics ของ XR-2211 ค่า $R_o + R_x$ ที่ใช้อยู่ในช่วง 15 - 100 K

- Tracking Range (ช่วงของความถี่ที่สัญญาณ Input กวาดไป โดยที่ PLL สามารถเก็บความถี่นั้นได้) กำหนดได้โดย

$$\Delta f = [(R_o + R_x) f_o / R_1]$$

- Loop Damping Factor (ตัวกำหนดจำนวนของ Overshoot , Undershoot หรือ สัญญาณ Ringing จากการตอบสนองของ Step Input ในด้านความถี่ของ PLL) กำหนดได้โดย

$$\zeta = 1/4 \sqrt{C_o / C_1}$$

โดยทั่วไปจะเลือกค่า $= 1/2$

- Time Constant ของ FSK Output Filter กำหนดได้โดย

$$\tau_f = R_f C_f$$

โดยการประมาณค่า

$$\tau_f = 0.3 / \text{Buad Rate}$$

- Capture Range (ช่วงของความถี่ที่ PLL สามารถ Lock ความถี่นั้นได้) กำหนดได้โดย

$$\Delta f_c = (80 - 90 \%) \Delta f$$

โดยที่ช่วงของ Capture Range จะถูกกำหนดโดย C_1

- Lock-Detect Filter ทำหน้าที่กำจัดสัญญาณรบกวนจาก Lock-Detect Output

สำหรับค่า $R_D = 470$ K ค่าอย่างต่ำของ C_D กำหนดได้โดย

$$C_D (\mu F) = 16 / \text{Capture Range (Hz)}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

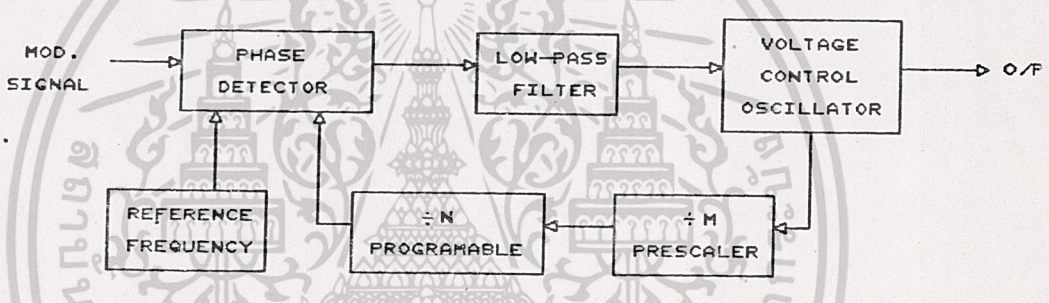
วงจร Frequency Synthesizer

วงจรสังเคราะห์ความถี่ หรือ Frequency Synthesizer เป็นวงจรที่ให้ Output เป็นสัญญาณความถี่ที่มีเสถียรภาพสูงจำนวนมากจากสัญญาณอ้างอิงภายในวงจรสัญญาณเดียว สามารถแบ่งออกเป็น 2 ลักษณะคือ

1.การสังเคราะห์ความถี่โดยตรง(Direct Frequency Synthesis) ใช้ Crystal-Controled Oscillator หลาย ๆ ชุดทำการ ทวีความถี่ ผสมความถี่ หารความถี่ จนได้ความถี่สุดท้ายที่ต้องการ

2.การสังเคราะห์ความถี่ทางอ้อม (Indirect,Coherent Synthesis) ใช้ Phase-Lock Loop ในการทำงาน

ในโครงงานนี้ใช้ระบบ PLL ในการสังเคราะห์ความถี่ ซึ่งมีระบบพื้นฐานแสดงดังรูป 3.29



รูปที่ 3.29 แสดงโครงสร้างของระบบ Phase-Lock Loop

การสังเคราะห์ความถี่ด้วย PLL ทำได้โดย ให้ความถี่ Input เป็นความถี่อ้างอิงมาตรฐานที่สร้างจากวงจรผลิตความถี่คุณภาพสูงที่ควบคุมด้วยCrystal จากนั้นสร้าง VCO ให้ทำงานที่ความถี่ทวีคูณของความถี่อ้างอิง ทำการหารความถี่ด้วยเลขนั้นแล้วป้อนเข้า PD ความถี่ของ VCO ก็จะเปลี่ยนไปตามตัวหารนั้น

$$f_o = N f_r$$

โดยมีสมการพื้นฐานของ Loop (Closed-Loop Transferfunction) เป็น

$$\phi(s) = \frac{K_o K_d F(s)}{S + K_o K_d F(s)/K_N} = \frac{K_o K_d F(s)/S}{1 + K_o K_d F(s)/K_N S}$$

ซึ่ง Forward Gain ของระบบคือ

$$G(s) = K_o K_d F(s)/S$$

และ Open-Loop Gain คือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$G(s)H(s) = K_o K_d F(s)/K_N S$$

โดยที่ K_d คือ Gain Factor หรือ Conversion Gain หน่วย Volt/Radian (ของ PD)

K_o คือ Gain Factor หรือ Conversion Gain หน่วย Radian/Sec/Volt (ของ VCO

F(s) คือ Transferfunction ของ Loop Filter

K_N คือ ตัวหารความถี่

จากสมการพื้นฐานของ Loop สามารถนำไปประยุกต์เป็นวงจรสังเคราะห์ความถี่ได้ โดยนำParameter ต่างๆ ในวงจร เปลี่ยนเป็นวงจรจริงแล้วนำมาต่อกันเป็นระบบ

- วงจร Phase Detector

เป็นวงจรที่ให้ Input เป็นแรงดัน (Analog) หรือเป็นความกว้างของ Pulse (Digital) ซึ่งขึ้นอยู่กับความต่าง Phase ของ Input ทั้งสอง

วงจรที่สามารถทำหน้าที่เป็น Phase Detector ได้คือ

1.Analog Phase Detector สำหรับวงจรที่มีขนาดของสัญญาณที่เปลี่ยนแปลงแบบไม่จำกัด ซึ่งมีวงจร Mixer และ วงจร Sampling Detector

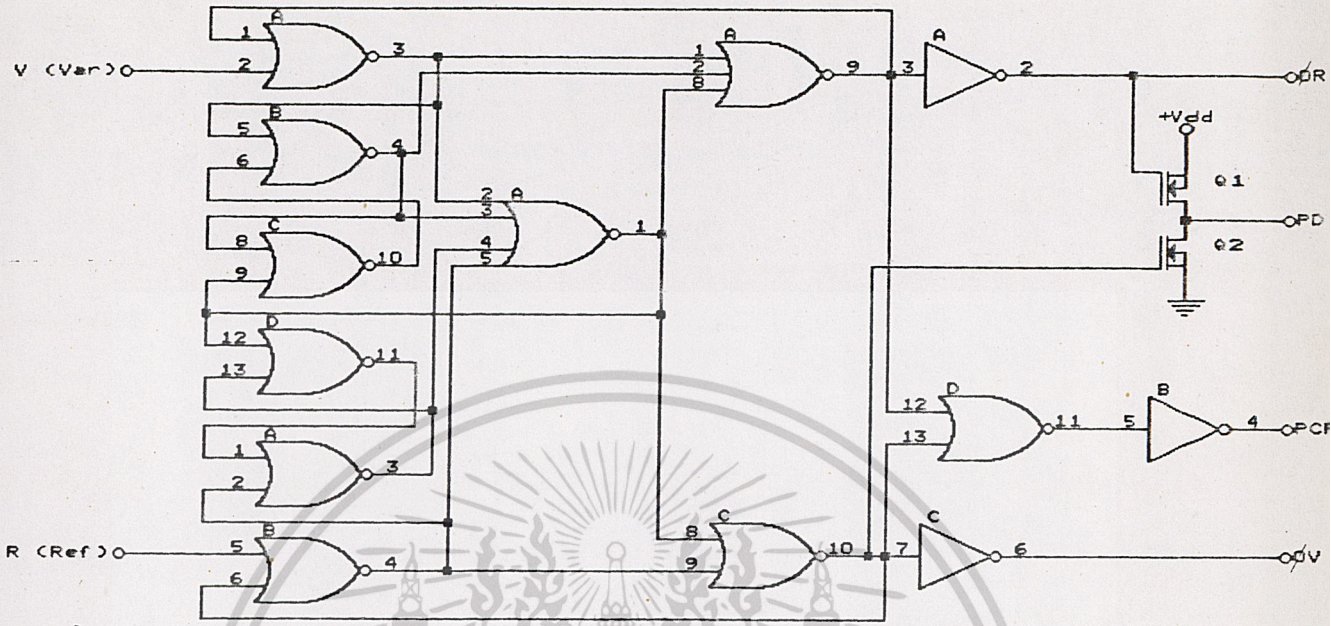
2.Digital Phase Detector สำหรับวงจรที่มีขนาดของสัญญาณที่เปลี่ยนแปลงแบบจำกัด ซึ่งมีวงจร Exclusive-OR Phase Detector , วงจร Flip-Flop Detector และ วงจร Phase-Frequency Detector

สำหรับในโครงงานนี้ใช้วงจรPhase-Frequency Detector จึงขอกล่าวเฉพาะส่วนของวงจรนี้เท่านั้น

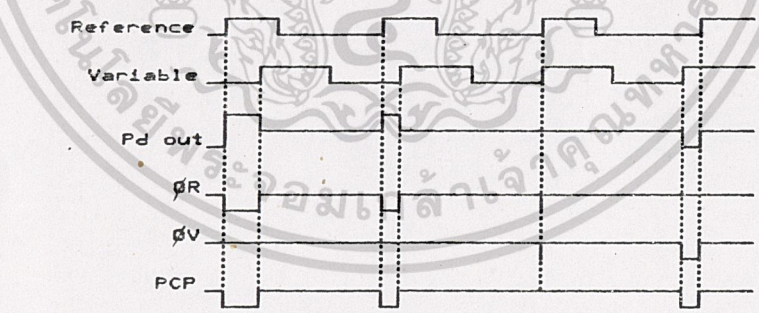
- วงจร Phase-Frequency Detector

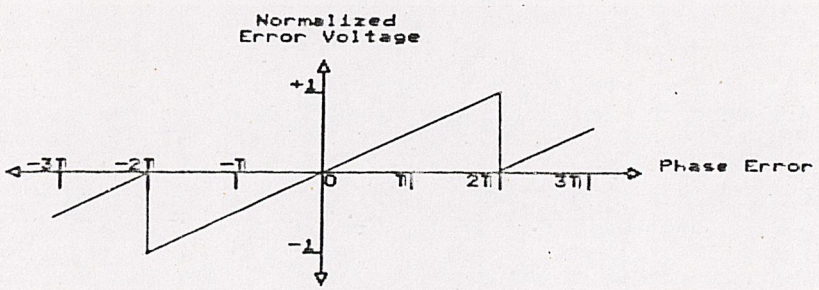
เป็นวงจรที่ให้ผลตอบสนองเร็วกว่าใน วงจร PLL เมื่อ Input ความถี่ต่างกัน , ไม่จำเป็นต้องมีวงจร Filter ที่ดี เมื่อเทียบกับวงจรอีกสองแบบ และใช้กับระบบ PLL ที่ต้องการการตอบสนองในย่านกว้าง โดยมีรูปวงจรดังแสดงในรูปที่ 3.29 และ แสดงรูปคลื่นของวงจร Phase Detector ดังรูป 3.30 a

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.29 แสดงวงจร Phase-Frequency Detector ชนิดทำงานที่ขอบขาขึ้น





รูปที่ 3.30 แสดงรูปคลื่นของวงจร Phase Detector และ
ความสัมพันธ์ระหว่าง ความต่าง Phase และแรงดัน Output

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วงจร Phase-Frequency Detector ให้ Output 2 แบบคือ

1.Tri-State Output (ขา P_D)

ถ้าความถี่ $f_v > f_r$ จะเป็น Pulse ลบ

ถ้าความถี่ $f_v < f_r$ จะเป็น Pulse บวก

ถ้าความถี่ $f_v = f_r$ Output จะ High Impedance

โดยจะมี Convertion Gain = $V_{DD} / 4\pi$

สำหรับความสัมพันธ์ระหว่าง ความต่าง Phase และแรงดัน Output แสดงดังในรูป 3.30 b

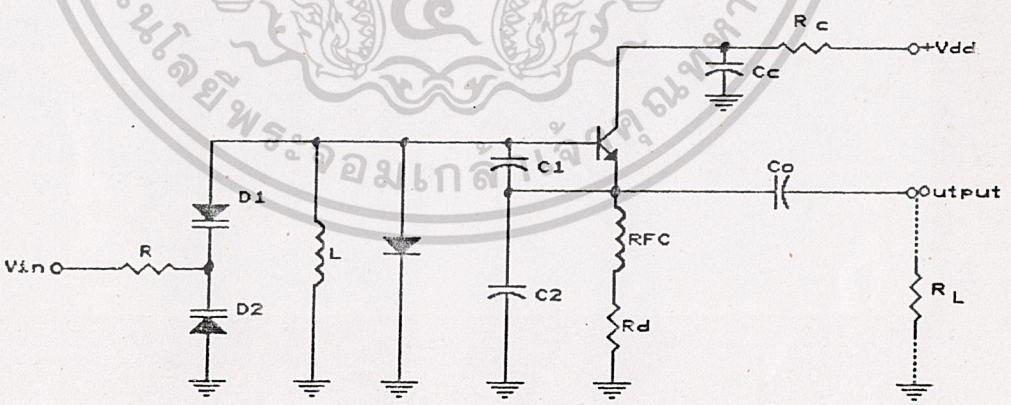
2.Phase Pulse Output

ให้ Logic 0 ทุกครั้งที่เกิดความต่าง Phase ของ f_v และ f_r นอกนั้นเป็น 1 ซึ่งจะใช้แสดงภาวะการ Lock ของ Loop

จากวงจรดังกล่าวจะรวมอยู่ใน IC ชนิด CMOS ตัวเดียวกันคือเบอร์ 4046

- วงจร Voltage-Control Oscillator

เป็นวงจร Oscillator ที่ใช้ LC โดยที่ ความถี่ Output จะแปรตามแรงดัน Input ที่ควบคุมโดยการเปลี่ยนแรงดันไบอัสให้ Varicap โดยใช้วงจรดังรูป



รูปที่ 3.32 แสดงวงจร Oscillator ที่ใช้ Varicap

จากรูป

C_c จะมีค่า X_c ต่ำที่ความถี่ใช้งาน โดยจะทำหน้าที่แยกแรงดัน DC ออกจากวงจร Resonance

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น สงวนลิขสิทธิ์ตามกฎหมาย

D_1 D_2 ต่อกลับกันเพื่อแก้ปัญหาการ Rectify แต่ค่า Capacitance รวมจะลดลงครึ่งหนึ่ง และ D_1D_2 ต้องมีคุณสมบัติที่ใกล้เคียงกันมากที่สุด

จากวงจร Oscillator ดังรูปต้องนำค่า Capacitance ของ Varicap ไปรวมกับ C_1 C_2 ดังนั้นจะได้สมการความถี่เป็น

$$f_o = [\, 2\pi (LC_T)^{1/2} \,]^{-1}$$

โดยที่ $$C_T = C_D + \left[\frac{C_1 C_2}{C_1 + C_2} \right]$$

เมื่อ C_D คือ ค่า Capacitance ของ Varicap ที่แรงดัน Input ใด ๆ

เมื่อกำหนดค่า r = 1/2 (ขึ้นอยู่กับการ Dope สารของ Varicap Diode) เราสามารถกำหนด Gain Factor ของ VCO ได้จากสมการ

$$K_o = \frac{\Delta f_o}{\Delta V} = \frac{C_{v1} \sqrt{\phi + V_1}}{8\pi \sqrt{L} \, [\, C_T (\phi + V_2) \,]^{3/2}}$$

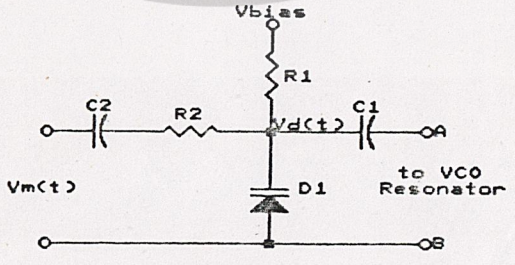
เมื่อ ϕ คือ ค่าแรงดันรอยต่อของ Diode (=~ 0.7 V)

V_1 คือ แรงดันที่กำหนดค่า Capacitance ต่ำสุดของ Varicap

V_2 คือ แรงดันที่ไบอัส Varicap

- วงจร Frequency Modulator

เราสามารถ Modulate สัญญาณ Base-Band กับ f_o ของ VCO ในระบบ PLL โดยการใช้วงจร FM Modulator ดังรูป 3.33



รูปที่ 3.33 แสดงวงจร FM Modulator

จากรูป วงจรจะทำหน้าที่แยกแหล่งกำเนิดสัญญาณ Base-Band ซึ่งอาจมี Impedance ต่ำซึ่งอาจจะไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รบกวนการทำงานของ Loop Filter

ที่ขั้ว AB จะนำไปต่อคร่อมกับขดลวดของวงจร Oscillator

C_1 มีค่า X_c สูงมากเมื่อเทียบกับ D_1 ที่ ความถี่ Carrier สำหรับความถี่ Carrier ระดับ VHF ขณะที่ยังไม่มีการ Modulate จะมีค่าประมาณ 0.5 - 2 PF

C_2 กันแรงดัน DC จากแหล่งกำเนิดสัญญาณ Base-Band

V_{bias} เป็นตัวกำหนด ค่าความเก็บประจุของ D_1

Modulation Index ของ FM กำหนดโดย

$$mf = \Delta f / f_m$$

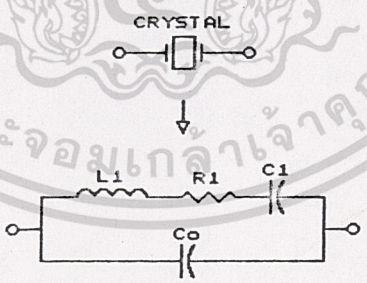
โดย $\Delta f = K E_m \sin W_m t$

เมื่อ K คือ ค่า Frequency Deviation Constant ซึ่งขึ้นอยู่กับคุณสมบัติของ D_1

จากสมการจะเห็นได้ว่าความสัมพันธ์ของ Δf_c กับ $V_m(t)$ จะไม่เป็นเชิงเส้นทีเดียวนักเนื่องจากคุณสมบัติของ D_1 และค่า Modulation Index จะขึ้นอยู่กับ ค่า K และ แรงดันสูงสุด (E_m)

- วงจรผลิตความถี่อ้างอิง

ใช้วงจร Oscillator ชนิดควบคุมด้วย Crystal เนื่องจากมีเสถียรภาพระยะสั้นและยาวดีมาก โดยวงจรสมมูลของ Crystal แสดงดังรูป 3.34



รูปที่ 3.34 แสดงวงจรสมมูลของ Crystal

L_1 R_1 C_1 เป็นวงจร Resonance อนุกรม แทนการสั่นของ Crystal

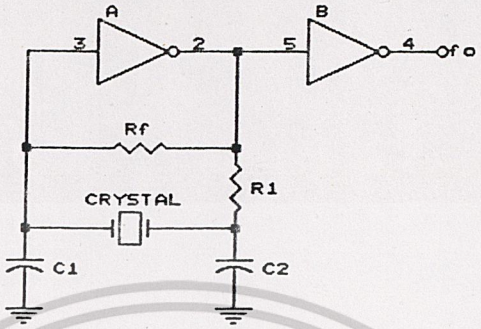
C_o เกิดจาก Holder ของ Crystal

ถ้าเป็นการสั่นที่ Fundamental Mode ของ Crystal แบบ AT Cut ความถี่ของการ Resonance สามารถกำหนดได้จาก

เอกสารนี้ ... ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

f_o (MHz) =~ 1675 / ความหนา (μm)

วงจร Oscillator ในโครงงานนี้สร้างจาก CMOS IC ที่เป็น Inverter ดังวงจรรูปที่ 3.35



รูปที่ 3.35 แสดงวงจร Crystal Oscillator ที่สร้างจาก Inverter

วงจรดังรูปจะทำงานได้อย่างถูกต้องถ้าทราบค่า C_L (Loading Capacitance) ของ Crystal ซึ่งกำหนดโดยผู้ผลิต โดย

$$C_L = \frac{C_{in} C_{out}}{C_{in} + C_{out}} + C_a + C_o + \frac{C_1 C_2}{C_1 + C_2}$$

โดย C_{in} และ C_{out} คือ Input และ Output Capacitance ของ Inverter

C_a คือ Capacitance ระหว่างขา Input และ Output ของ Inverter

ปกติที่แรงดัน 5 V C_{in} = 5 PF C_{out} = 6 PF C_a = 5 PF

ส่วน R_f =~ 5-20 M Ohm

R_1 จำกัดแรงดันไบอัส Crystal โดยจะมีค่าประมาณ 0-10 K Ohm

C_1 C_2 ที่ความถี่ต่ำกว่า 13 MHz จะมีค่าไม่เกิน 35 PF และ ความถี่ Output สามารถปรับแต่งได้เล็กน้อยโดยการปรับค่า C_1

ความถี่ของ Crystal มีค่าสูงเป็น MHz แต่ความถี่อ้างอิงมีค่าเป็น KHz จึงต้องใช้วงจรหารความถี่ ซึ่งเราใช้ IC 4060 ที่มี Asyncronous Counter และ Inverter อยู่ภายในตัวเอง

- วงจรหารความถี่

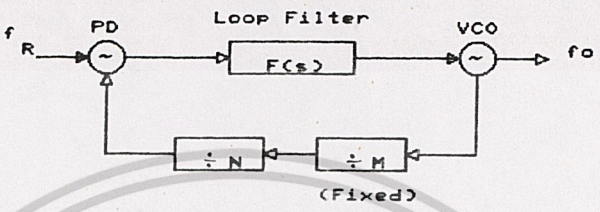
จะทำหน้าที่หารความถี่ของ VCO ในวงจรสังเคราะห์ความถี่ จึงต้องเป็นชนิดเปลี่ยนความถี่ได้อย่างอิสระ ซึ่งจะเป็นวงจร Syncronous Counter ต่อคาสเคดกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แต่ถ้าในกรณีที่ความถี่ของ VCO สูงเป็น VHF หรือ UHF วงจรหารทั่วไปจะไม่ทำงาน จำเป็นต้องใช้วิธี

Heterodyne หรือ Prescale ลดความถี่ของ VCO ก่อนที่จะป้อนเข้าสู่วงจรหาร

ในที่นี้จะกล่าวถึงวิธี Prescale ที่ใช้ในโครงงานนี้เท่านั้น โดยมี Block diagram แสดงวิธี Prescale ดังรูปที่ 3.36



รูปที่ 3.36 แสดงวิธีหารความถี่ โดยวิธี Prescale

ใช้ IC ชนิด ECL (Emitter Couple Logic) ที่สามารถทำงานที่ความถี่ VHF หรือ UHF เป็นตัวหารความถี่คงที่ก่อนป้อนเข้าสู่วงจรหาร โดยวิธีนี้จะได้

$$f_o = Mf_R * N$$

จากสมการจะเห็นได้ว่าข้อเสียของวิธี Prescale คือ ความกว้างของ Channel จะเป็น M เท่าของ f_R ซึ่งจะมีค่ามากทำให้ Loop มีคุณสมบัติเลวลง แต่มีข้อดีคือสะดวกกว่าวิธี Heterodyne

วงจร R.F. Power Amplifier

อาจแบ่งตามลักษณะการทำงานได้ 2 ลักษณะคือ

1.วงจรขยายแบบ Linear โดยใช้วงจร Class A , B และ AB วงจรแบบนี้ใช้กับ Single Sideband Transmitter และ Wide Band Transmitter

2.วงจรขยายแบบ Nonlinear โดยใช้วงจร Class C ,D และ E วงจรแบบนี้ใช้กับ Narrow Band Transmitter

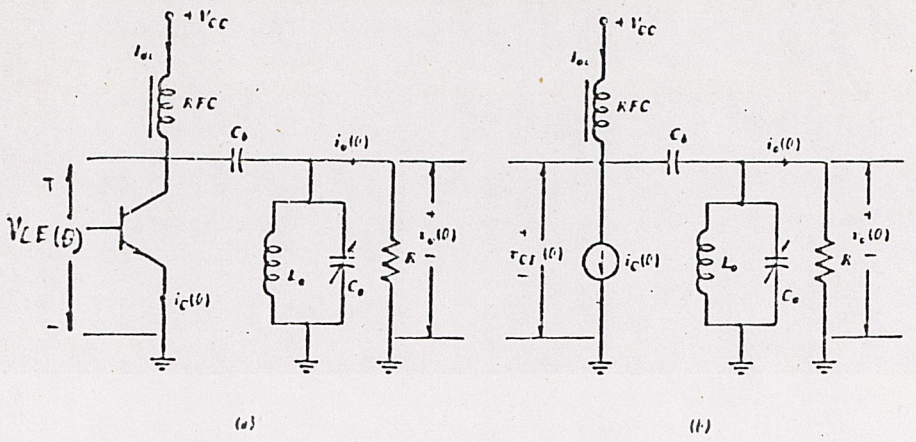
โดยในที่นี้จะขอกล่าวถึงวงจร Class A และ C ที่ใช้ในโครงงานนี้เท่านั้น

- วงจรขยายกำลัง Class A

ทำหน้าที่ขยายสัญญาณทั้ง Cycle ของ Input Output ที่ได้จะมีลักษณะเหมือน Input ทุกประการ ซึ่งลักษณะของวงจรแสดงได้ดังรูปที่ 3.37

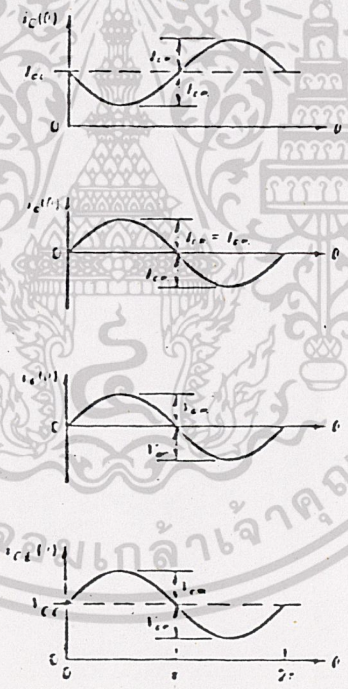
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Class A amplifier. (a) Circuit; (b) equivalent circuit; and (c) waveforms.



รูปที่ 3.37 แสดงวงจรขยายกำลัง Class A

และ รูปคลื่นที่ได้จากวงจรขยายกำลัง Class A แสดงได้ดังรูปที่ 3.38



รูปที่ 3.38 แสดงรูปคลื่นในวงจรขยายกำลัง Class A

สมมติให้ แรงดัน Bias และ สัญญาณ Input ทำให้เกิด กระแส Collector($i_c(\theta)$) มีค่าเท่ากับ

$$i_c(\theta) = I_{cq} - I_{cm} \sin\theta$$

I_{cq} คือส่วนประกอบที่เป็น DC

$I_{cm} \sin\theta$ คือส่วนประกอบที่เป็น AC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนประกอบที่เป็นสัญญาณ AC สามารถผ่าน C_b ไปยัง Output ทำให้เกิดแรงดัน Output มีค่าเท่ากับ

$$v_o(\theta) = I_{cm} R \sin\theta = V_{om} \sin\theta$$

แรงดัน Collector ($v_c(\theta)$) ประกอบด้วย แรงดัน DC เท่ากับแหล่งจ่าย และ แรงดัน AC เท่ากับ แรงดัน Output ดังนั้น

$$v_c(\theta) = v_{cc} + v_{om} \sin\theta = v_{cm} \sin\theta$$

เนื่องจากวงจรทำงานเป็น Current Souce ดังนั้น $v_c(\theta)$ ต้องมีค่าเป็นบวก จึงต้องจำกัดให้ $v_{om} < V_{cc}^2$ และ $I_{om} = V_{om} / R <= V_{cc} / R$

ในการหาประสิทธิภาพของวงจร ขยาย Class A สามารถหาได้จาก

กำลังงาน DC ที่วงจรดึงจากแหล่งจ่าย

$$P_i = V_{cc} I_{cq} = V_{cc}^2 / R$$

กำลังงาน Output

$$P_o = V_{om}^2 / 2R = V_{cc}^2 / 2R$$

ประสิทธิภาพของวงจรคือ

$$\eta = P_o / P_i = V_{om}^2 / 2 V_{cc}^2 <= 1/2$$

กำลังสูญเสียใน Transistor คือ

$$P_d = P_i - P_o$$

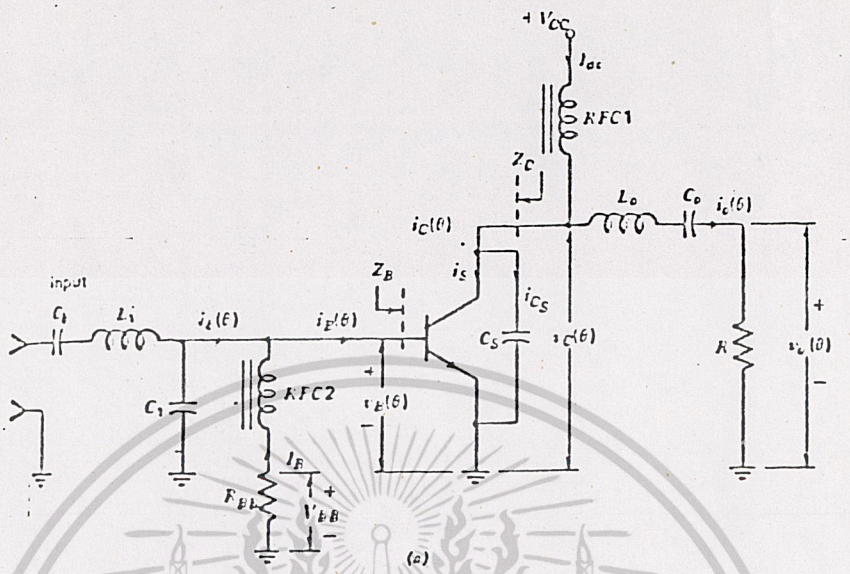
จะเห็นได้ว่า วงจรขยาย Class A มีประสิทธิภาพต่ำจึงมักจะใช้เป็นวงจรขับ .มากกว่าใช้เป็นวงจรขยายกำลังภาคสุดท้าย

- วงจรขยายกำลัง Class C

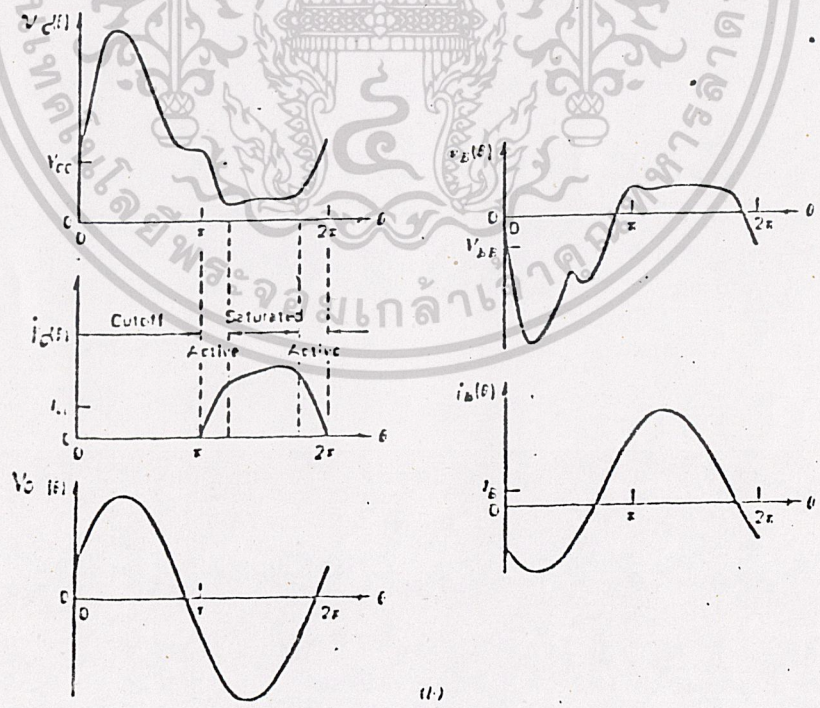
การวิเคราะห์การทำงานมาตรฐานไม่สามารถทำได้ แบบจำลองแสดงการทำงานก็จะมีข้อสมมติที่เป็นไปได้ยาก และ Parameterหลายตัวก็ไม่สามารถกำหนดได้ล่วงหน้า ดังนั้นการวิเคราะห์การทำงานในลักษณะต่าง ๆ สามารถดูรายละเอียดเพิ่มเติมได้จากหนังสืออ้างอิง

ลักษณะวงจรขยายกำลัง Class C ดังรูป 3.39 และ รูปคลื่นดังรูป 3.40

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.39 แสดงวงจรขยายกำลัง Class C



รูปที่ 3.40 แสดงรูปคลื่นในวงจรขยายกำลัง Class C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- จากวงจร ขยายกำลัง Class C

RFC 1 จะมีค่าความต้านทานสูงมากที่ความถี่ใช้งาน มีเพียงกระแส DC เท่านั้นที่ไหลผ่าน

L_o C_o จะต้องมีค่า Q สูงพอที่จะทำให้ กระแสที่ความถี่ Input ไหลเท่านั้น

วงจร Matching ทาง Input และ Output อาจจะเป็นวงจรอื่นขึ้นกับ Z_B Z_C ของ Transistor

R_{BB} RFC2 ในบางกรณีอาจไม่ต้องมีโดยต่อขา Base ลง Ground หรือ อาจใช้เป็นส่วนหนึ่งของวงจร Matching Network

RFC1 อาจใช้เป็นส่วนหนึ่งของ Matching Network

R_{BB} มีค่า ไม่กี่ Ohm - ไม่กี่ร้อย Ohm R_{BB} ที่ไม่เหมาะสมอาจทำให้ Transistor เสียหายเนื่องจากได้รับแรงดัน Reverse Bias มากเกินไป

RFC_1 RFC_2 จะต้องมี Impedance มากพอที่ความถี่ใช้งาน ประมาณ 4 -10 เท่า ของ Resistor ที่ขา Collector และ Base

RFC_1 RFC_2 จะต้องมีค่า Q น้อย ๆ มิฉะนั้นวงจรอาจจะ Oscillate ได้ที่ความถี่ต่ำ (ต่ำกว่าความถี่ใช้งาน)

RFC1 ต้องมีความต้านทาน DC ต่ำพอ ในกรณีที่เป็น Power Amp. กำลังสูง ๆ

V_{cc} จะต้องมีการ Bypass ความถี่ที่ใช้งานดีพอ

Transistor ต้องมีการระบายความร้อนอย่างดีพอ

- จากรูปคลื่นของส่วนต่าง ๆ ในวงจร

ขนาดของรูปคลื่นจะขึ้นอยู่กับ Transistor , มุมในการนำกระแส , ค่า Q ของวงจร Matching Input และ Output , แรงดัน DC ที่จ่ายให้วงจร , กำลังงานที่ Input ซึ่งเราสามารถคุมได้แค่เพียงแรงดัน DC และกำลังงานที่ Input เท่านั้น ดังนั้นการหาสมการทั่วไปเพื่อกำหนดลักษณะของรูปคลื่นจึงทำได้ยาก และวงจรจะทำงานได้อย่างถูกต้องก็ต่อเมื่อวงจรมีการ Matching โดยสมบูรณ์เท่านั้น โดยที่ขนาด และกำลังของ Output ประมาณได้จาก

$$P_o \sim 0.625\, V_{eff}^2 / R_1 = 0.625\, (V_{cc} - V_{sat})^2 / R_1$$

- ในการวิเคราะห์การทำงานของวงจร

สำหรับ Input รูป Sine

ขณะ Active หรือ Saturate ขณะที่ $i_b(\theta)$ เป็นบวก B-E จะนำกระแส $V_b(\theta)$ ~ แรงดัน Forward Transistor ก็จะ Active หรือ Saturate (ขึ้นกับ $V_c(\theta)$) กระแสไหลผ่าน RFC2 ค่อนข้างคงที่ตลอด Cycle ของ Input กระแส AC จะไหลผ่าน L_1 Transistor จะนำกระแสจนกระทั่ง $i_b(\theta)$ เริ่มกลับขั้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขณะ Cut off $i_b(\theta)$ กลับขั้ว Transistor คายประจุที่ขา Base B-E จะเป็น Reverse Bias กระแสใน L_1 และ RFC2 จะไหลผ่าน C_1 ค่า C ที่ขา B-E จะทำให้ แรงดันเป็นลบในช่วงแรก และจะกลับมาเป็นบวก Transistor จะ Cut off จนกระทั่งแรงดันที่ขา Base = V_r (ซึ่งจะเป็นตัวกำหนดมุมในการนำกระแสของ Transistor) เมื่อ แรงดันที่ขา Base = V_r Base จะดึงกระแส DC ที่ Input ให้ผ่าน RFC2 และ R_{bb} ซึ่งจะทำให้เกิด $-V_{bb}$ ไป Bias Transistor ในช่วง Cut off

สัญญาณInput จะขับ Transistor ให้ทำงานแบบ Cut off และ Saturate วงจรจึงมีประสิทธิภาพสูงกว่า Class A โดยประสิทธิภาพของวงจรหาได้จาก

$$\eta_{max} = \frac{2y - \sin 2y}{4 (\sin y - y \cos y)}$$

โดย 2y คือ มุมในการนำกระแสของ Transistor

การทำงานของ Transistor ในลักษณะ Cut off และ Saturate ดังนั้นกระแส Collector จะประกอบด้วย Harmonic ของ สัญญาณ Input ด้วย

วงจร Matching Network

การ Matching ระหว่าง แหล่งกำเนิดสัญญาณ - Transistor - Load (50 Ohm) ทำได้หลายวิธีขึ้นกับ Bandwidthที่ต้องการ , ความถี่ที่ใช้งาน , Power ของวงจร และ Input/Output Impedance ของวงจร

วงจร Matching Network เองก็มีหลายลักษณะ แต่จะขอกล่าวเฉพาะชนิด Lump Constant ดังรูป 3.41 ส่วนที่มาของวงจรให้ดูจากหนังสืออ้างอิง

วิธีการคำนวณได้มีการทำเป็นตาราง Computer เรียบร้อยแล้ว

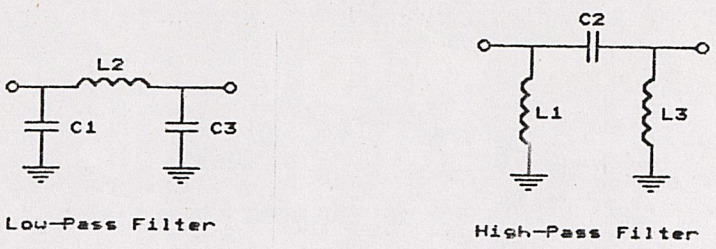
วงจร Filter

จะขอกล่าวถึงเฉพาะวงจร Chebychev Filter ที่ใช้ในวงจรภาคขยายเท่านั้น ส่วน Filter พื้นฐานอื่น ๆ ขอให้ไปศึกษาจากหนังสืออ้างอิง

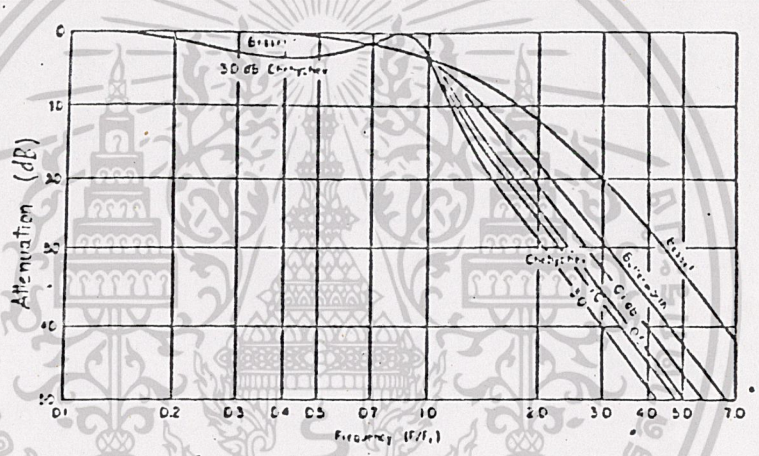
- Chebychev Filter

วงจร Chebychev Filter ดังแสดงดังรูป 3.42 สามารถลดทอนสัญญาณนอก Pass Band ได้ดีกว่าดังแสดงดังรูป 3.43 แต่มีข้อเสียคือ Pass Band จะเป็น Ripple และ มีการ Delay ที่แย่กว่าแบบอื่น ดังแสดงดังรูป 3.44

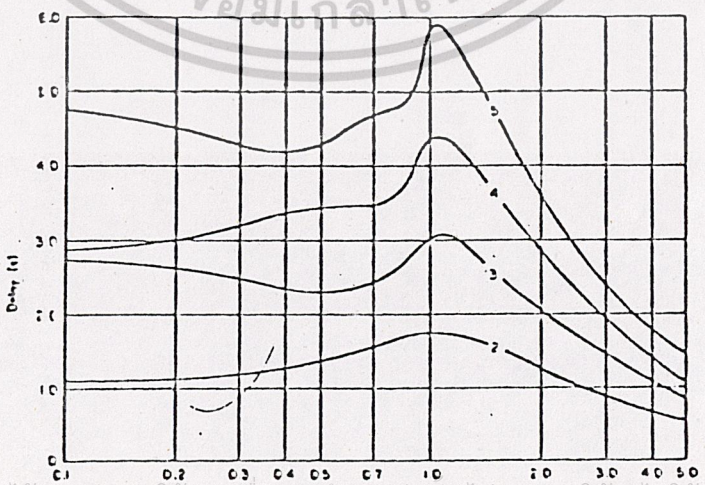
เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.42 แสดงวงจร Filter



รูปที่ 3.43 แสดงการลดทอนของ Filter แบบต่าง ๆ ที่ความถี่ Cut off และ Pole เท่ากัน



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.44 แสดงการ Delay ของ Chebychev Filter

ค่าของอุปกรณ์ในวงจร CHebychev Filter จาก 2-7 , Pass Band Ripple 0.1 ,0.5 ,1 แล 3 dB , Load Resistance 1 Ohm , Cut off Frequency 1 rad/sec แสดงได้ดังตาราง

ORDER	RIPPLE (dB)	C_1	L_2	C_3	L_4	C_5	L_6	C_7	R_{out}
2	0.1		1.211	1.643					0.738
	0.5		0.982	1.916					0.501
	1.0		0.835	2.219					0.376
	3.0		0.535	3.109					0.172
3	0.1	1.438	1.589	1.438					1.000
	0.5	1.862	1.280	1.862					1.000
	1.0	2.215	1.088	2.215					1.000
	3.0	3.349	0.712	3.349					1.000
4	0.1		0.981	2.225	1.642	1.330			0.738
	0.5		0.920	2.585	1.303	1.824			0.501
	1.0		0.831	2.991	1.121	2.212			0.376
	3.0		0.592	4.349	0.748	3.441			0.172
5	0.1	1.300	1.553	2.238	1.553	1.300			1.000
	0.5	1.809	1.298	2.682	1.298	1.809			1.000
	1.0	2.216	1.132	3.115	1.132	2.216			1.000
	3.0	3.481	0.762	4.538	0.762	3.481			1.000
6	0.1		0.970	2.220	1.550	2.350	1.450	1.350	0.738
	0.5		0.905	2.575	1.307	2.710	1.298	1.794	0.504
	1.0		0.836	3.031	1.190	3.161	1.140	2.224	0.376
	3.0		0.603	4.464	0.793	4.606	0.769	3.505	0.172
7	0.1	1.198	1.444	2.128	1.596	2.128	1.444	1.198	1.000
	0.5	1.737	1.271	2.664	1.357	2.664	1.271	1.737	1.000
	1.0	2.184	1.121	3.118	1.163	3.118	1.121	2.184	1.000
	3.0	3.519	0.772	4.639	0.801	4.639	0.772	3.519	1.000

ตาราง แสดงค่าของอุปกรณ์ในวงจร Chebychev Filter

จากค่าที่ได้ในตารางเราสามารถทำการ Scaling หาค่าของอุปกรณ์ที่ใช้งานจริงได้โดย

$$C = C_n / 2\pi F_c R$$

$$L = RL_n / 2\pi F_c$$

เมื่อ C_n L_n คือค่า Normalize จากตาราง

R คือ ค่าของ Load ที่ใช้งาน

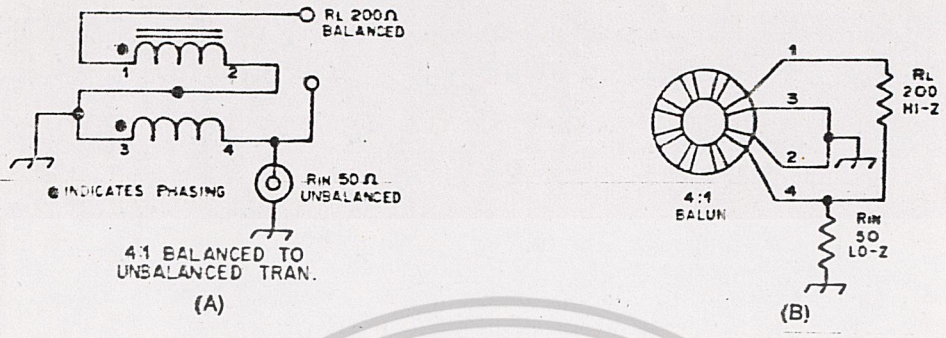
F_c คือ ค่าความถี่ Cut off 3 dB ที่ต้องการ

สายอากาศ

สายอากาศที่ใช้เป็นสายอากาศ Dipole ชนิด Half Wave ซึ่งรายละเอียดได้เคยกล่าวมาแล้ว สำหรับตัวสายอากาศใช้สายนำสัญญาณชนิด Twin Lead 300 Ohm ที่ใช้เป็นสายนำสัญญาณสำหรับเสาอากาศโทรทัศน์ เพื่อง่ายต่อการปรับแต่งค่า VSWR และทำการ Matching Impedance ระหว่าง เครื่องส่ง - สายอากาศ

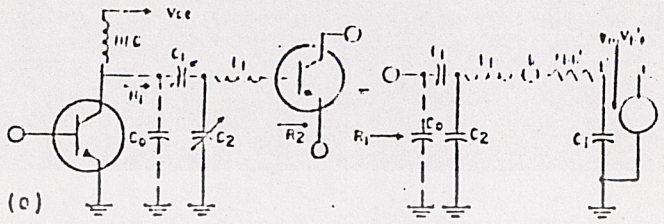
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยการใช้ 4Impedance Balance ที่ได้จากการพันขดลวดบนแกน Ferrite ดังรูป 3.45



รูปที่ 3.45 แสดงการพัน 4Impedance Balance

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



FOR $X_{L_1} >> X_{c_1}; R_1 > R_2 > r_{bb'}$

(1) $X_{L_1} = Q_L R_2 + Q_L r_{bb'}$

(2) $X_{c_1} = X_{c_0}\left[\sqrt{\frac{(Q_L^2+1)\, r_{bb'}}{R_1}} - 1\right]$

(3) $X_{c_2} = \frac{r_{bb'}(Q_L^2+1)}{Q_L} \cdot \frac{1}{\left[1-\sqrt{\frac{R_1 r_{bb'}(Q_L^2+1)}{X_{c_0}^2 Q_L^2}}\right]}$

FOR $X_{c_2} >> X_{c_1}; R_1 > R_2 > r_{bb'}$

(1) $X_{L_1} = \frac{R_2(Q_L^2+1)}{Q_L} \cdot \frac{1}{\left[1+\sqrt{\frac{R_1 R_2}{X_{c_0}^2} \cdot \frac{(Q_L^2+1)}{Q_L^2}}\right]} \cdot \frac{r_{bb'}(Q_L^2+1)}{Q_L} \cdot \frac{1}{\left[1+\sqrt{\frac{R_1 r_{bb'}}{X_{c_0}^2} \cdot \frac{(Q_L^2+1)}{Q_L^2}}\right]}$

(2) $X_{c_1} = X_{c_0}\left[\sqrt{\frac{r_{bb'}(Q_L^2+1)}{R_1}} - 1\right]$

(3) $X_{c_2} = Q_L R_2 + Q_L r_{bb'}$

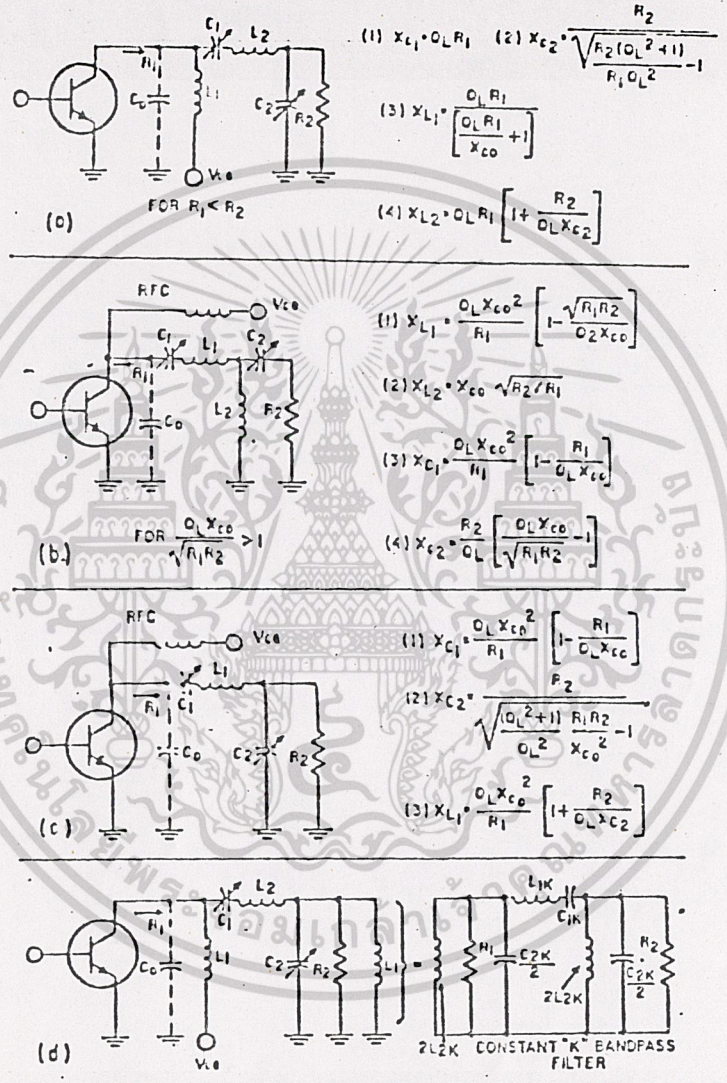
FOR $R_1 > R_2 > r_{bb'}; X_{L_2} > X_{c_1}$

(1) $X_{L_1} = \frac{R_1}{Q_L}$

(2) $X_{L_2} = \frac{R_2}{Q_L} \cdot \frac{\left[\sqrt{\frac{R_1}{R_2}} - 1\right]}{\left[1 - \frac{R_1}{Q_L X_{c_0}}\right]}$ (3) $X_{c_1} = \frac{R_1}{Q_L} \cdot \frac{\left[1-\sqrt{\frac{R_2}{R_1}}\right]}{1 - \frac{R_1}{Q_L X_{c_0}}}$ (4) $X_{c_2} = \frac{R_1}{Q_L} \cdot \frac{\sqrt{\frac{R_2}{R_1}}}{\left[1 - \frac{R_1}{Q_L X_{c_0}}\right]}$

Transistor input-circuit coupling networks.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



LET $C_{2K} = 2\,C_{OUT}$, R_1, R_2; f_1 = LOW FREQ. CUTOFF, f_2 = HI-FREQ. CUTOFF

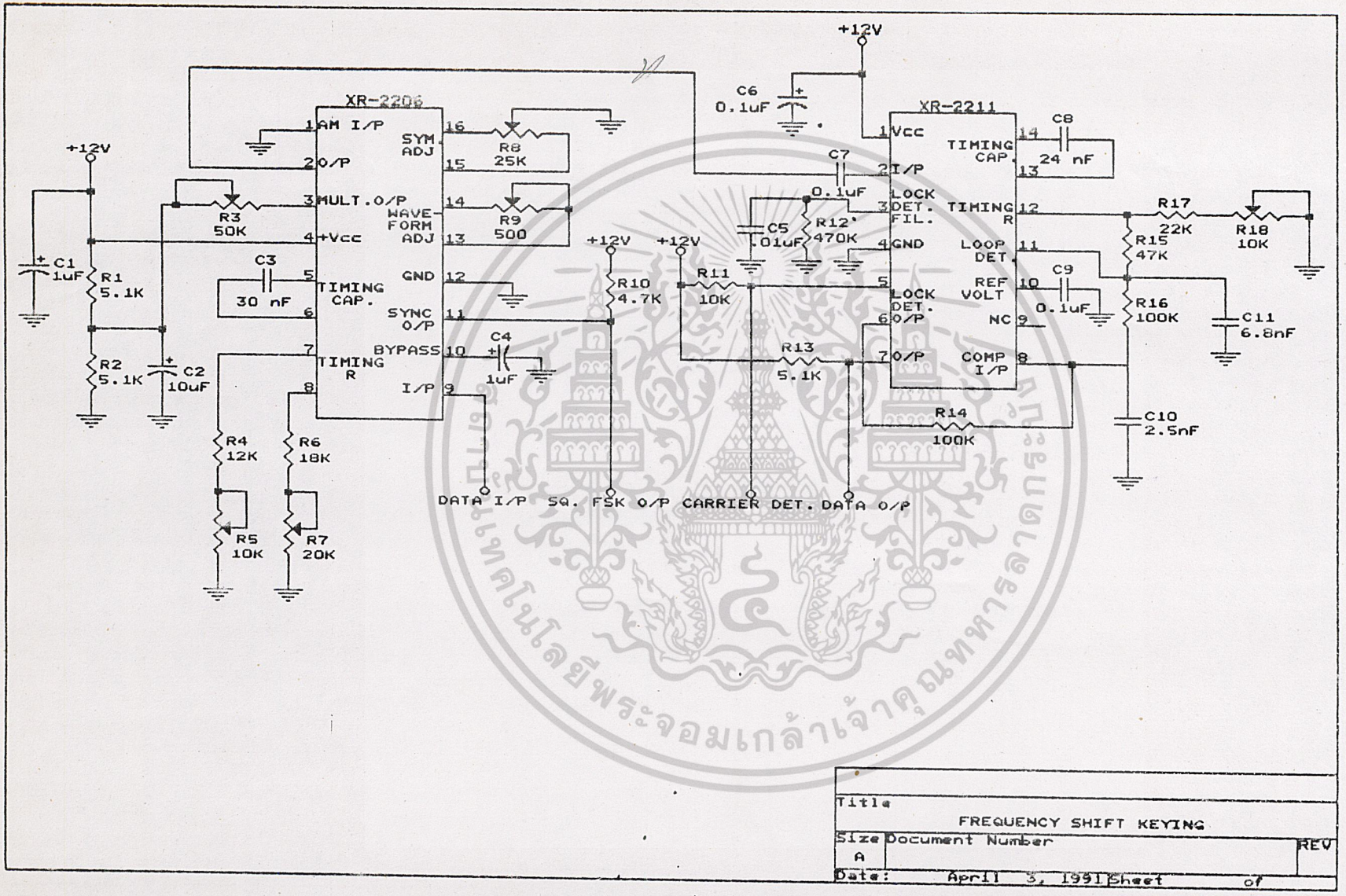
(1) $(f_2-f_1) = \frac{1}{2\pi C_0 R_L}$ (2) $L_2 = L_{1K} \frac{R_1}{\pi(f_2-f_1)}$ (3) $L_1 = 2L_{2K} = \frac{(f_2-f_1)R_1}{2\pi f_1 f_2}$

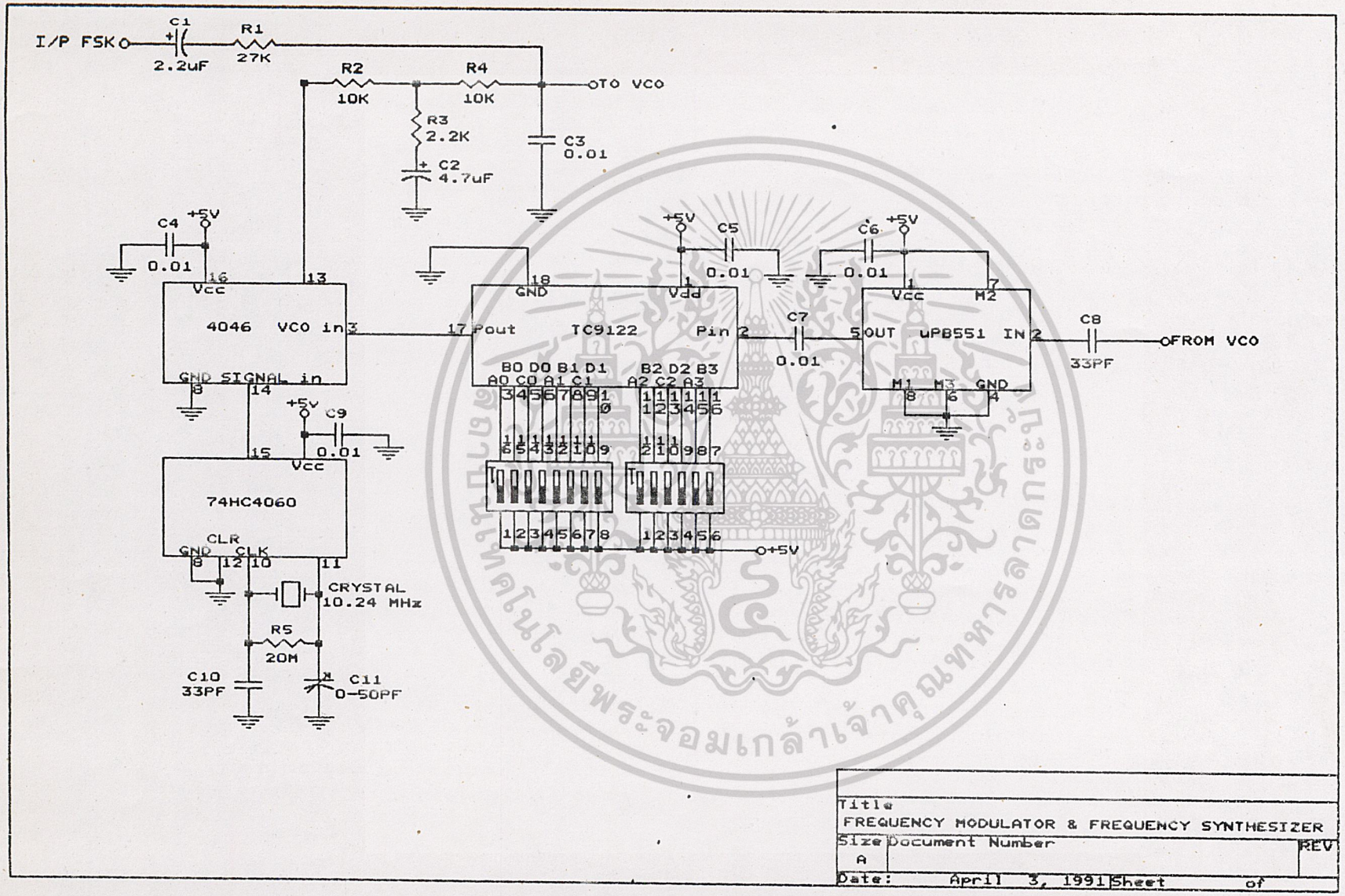
(4) $C_1 = C_{1K} = \frac{f_2-f_1}{4\pi f_1 f_2 R_1}$ (5) $C_2 = C_0 = \frac{C_{2K}}{2}$

Additional transistor output-coupling networks including transistor output capacitance.

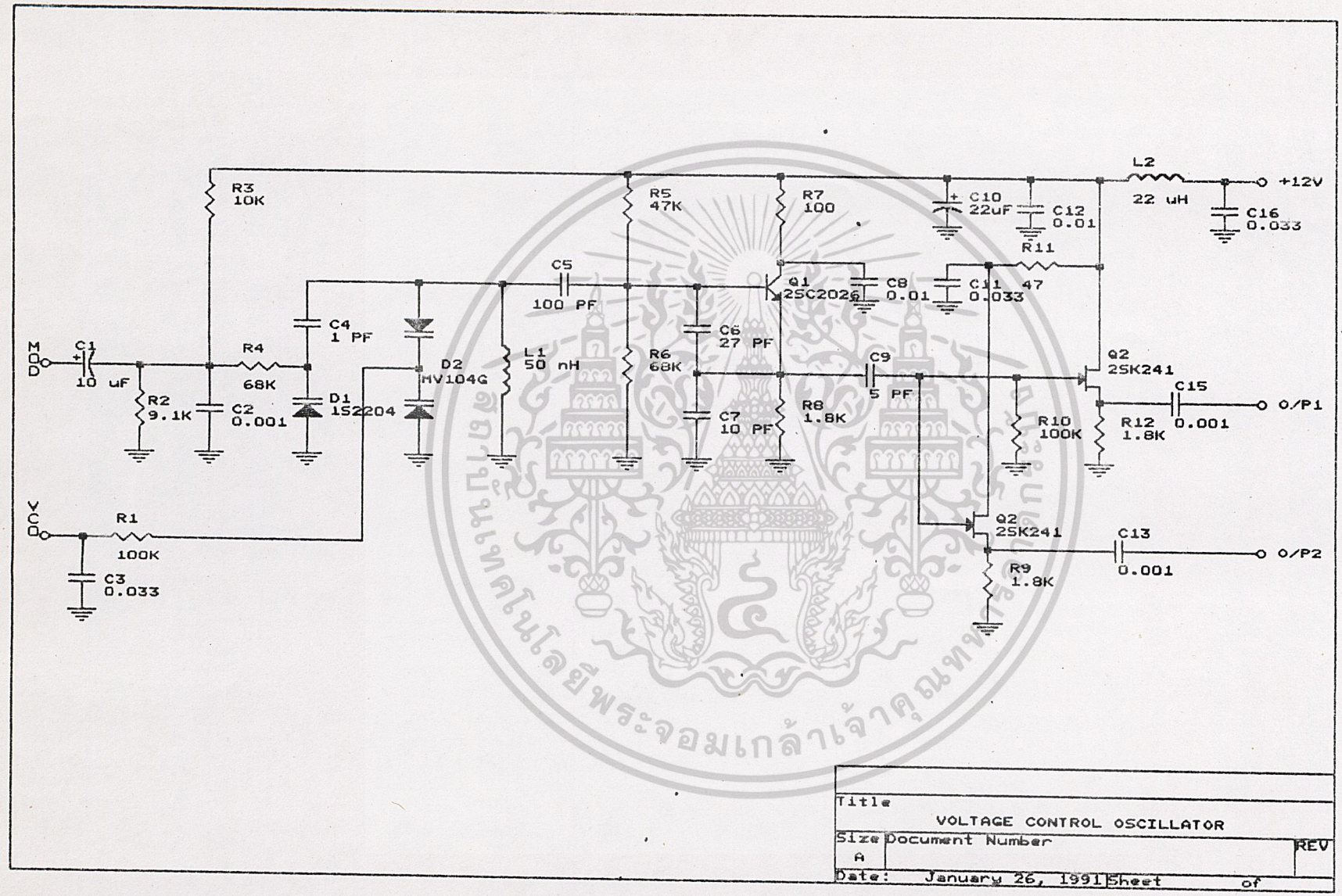
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของลิขสิทธิ์ทุกครั้งที่มีการนำไปใช้

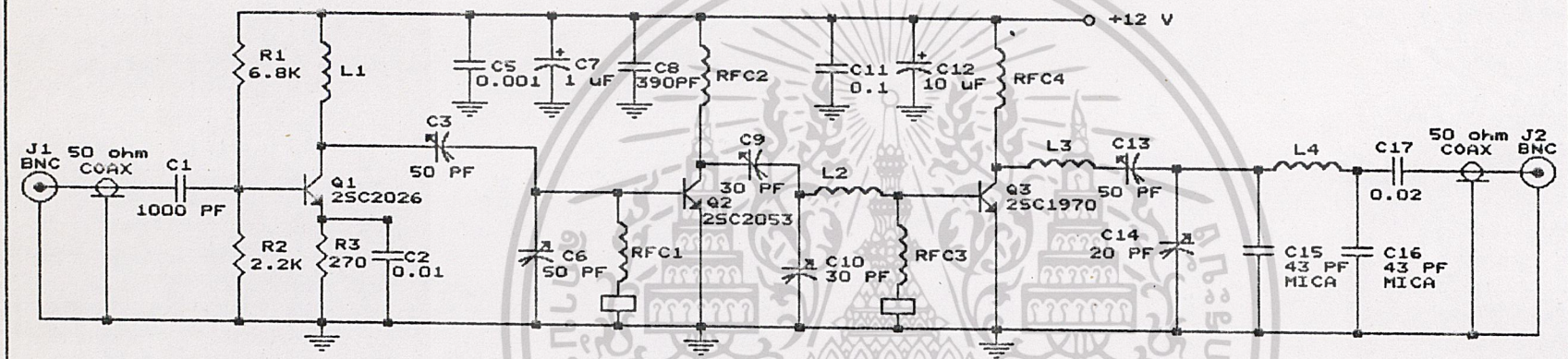
รูปที่ 3.41 แสดงตัวอย่างของวงจร Matching Network


XR-2206
XR-2211
+12V
AM I/P
SYM ADJ
MULT.O/P
WAVE-FORM ADJ
+Vcc
GND
TIMING CAP.
SYNC O/P
BYPASS
TIMING R
I/P
Vcc
I/P
LOCK DET. FIL.
GND
LOCK DET. O/P
O/P
TIMING CAP.
TIMING R
LOOP DET.
REF VOLT
NC
COMP I/P
C1 1uF
R1 5.1K
R2 5.1K
C2 10uF
R3 50K
C3 30 nF
R4 12K
R5 10K
R6 18K
R7 20K
R8 25K
R9 500
C4 1uF
C5 .01uF
R10 4.7K
R11 10K
R12 470K
C6 0.1uF
C7 0.1uF
R13 5.1K
R14 100K
C8 24 nF
C9 0.1uF
C10 2.5nF
R15 47K
R16 100K
R17 22K
R18 10K
C11 6.8nF
DATA I/P
SQ. FSK O/P
CARRIER DET.
DATA O/P
Title
FREQUENCY SHIFT KEYING
Size A
Document Number
REV
Date: April 3, 1991 Sheet of



Title: FREQUENCY MODULATOR & FREQUENCY SYNTHESIZER		
Size: A	Document Number:	REV:
Date: April 3, 1991	Sheet:	of:


MOD
C1
10 uF
R2
9.1K
R4
68K
C2
0.001
C4
1 PF
D1
1S2204
D2
MV104G
R3
10K
C5
100 PF
L1
50 nH
R5
47K
R6
68K
C6
27 PF
C7
10 PF
R7
100
Q1
2SC2026
C8
0.01
R8
1.8K
C9
5 PF
C10
22uF
C12
0.01
R11
C11
0.033
47
L2
22 uH
+12V
C16
0.033
Q2
2SK241
C15
0.001
O/P1
R10
100K
R12
1.8K
Q2
2SK241
C13
0.001
O/P2
R9
1.8K
VCO
R1
100K
C3
0.033
Title
VOLTAGE CONTROL OSCILLATOR
Size
A
Document Number
REV
Date: January 26, 1991
Sheet
of



L1 19SWG ø1/4 4TURNS

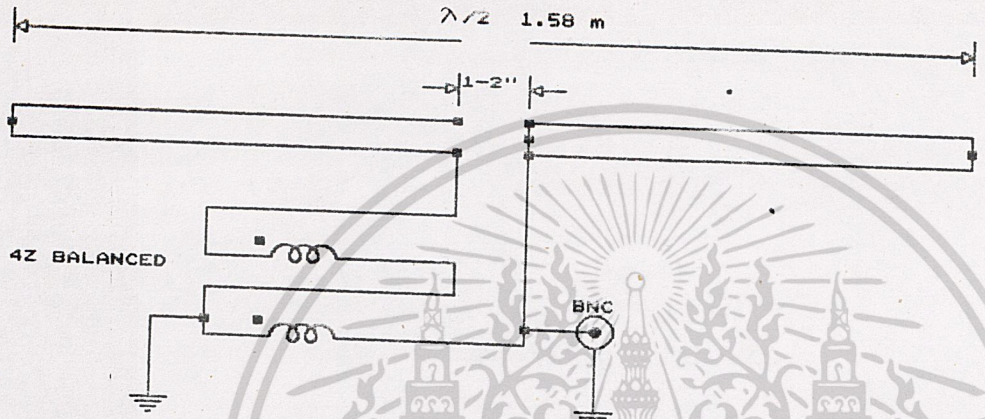
L2 18SWG ø1/4 3TURNS

L3 20SWG ø1/4 7TURNS

L4 19SWG ø1/4 6TURNS

Title	POWER AMPLIFIER	
Size A	Document Number	REV
Date: April 3, 1991	Sheet	of

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง



สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

Title: DIPOLE ANTENNA		
Size: A	Document Number:	REV
Date: April 3, 1991	Sheet	of

บทที่ 4

การทดลองและผลการทดลอง

ส่วนที่ได้ทำไปแล้วในเทอมนี้คือ

-ส่วน INTERFACE และ SOFTWARE เบื้องต้นในการส่งรับข้อมูล

-ส่วน R.F

การทดลองแบ่งออกเป็น 2 ส่วน

ส่วน INTERFACE

-สื่อสารข้อมูลระหว่าง IBM-PC กับ IBM-PC โดยผ่านส่วนอินเตอร์เฟส (INTERFACE) ด้วยอัตราเร็วในการส่งข้อมูล 9600 baud ผ่านสายส่งโดยไม่มีความผิดพลาดของข้อมูลโดยเฉพาะกับไฟล์ขนาดที่ไม่ยาวมากนัก แต่ถ้าความยาวของข้อมูลมากอาจจะทำให้ความยาวช่วงท้ายของไฟล์หายไปได้มีสาเหตุมาจากสัญญาณที่ส่งเกิดผิดพลาดขึ้นเพราะ สันนิษฐานว่าบัฟเฟอร์ที่เชื่อมต่อระหว่างบัสข้อมูลกับ INPUT PARALLEL ไม่ได้ PULL UP ไว้เพราะฉะนั้นการรับส่งข้อมูลนานทำให้ขนาดของสัญญาณตกลงมาได้

ส่วน R.F

-ทดลองส่งสัญญาณความถี่ 4800 Hz หรืออัตราเร็วในการส่งข้อมูล 9,600 bit\sec- ที่ระยะทาง 100 m . สัญญาณที่เอาท์พุท (output) หลังจากผ่านคอมพาราเตอร์ (comperater) ได้สัญญาณรูปเดียวกับสัญญาณที่ส่งมา เมื่อทดลองเปลี่ยนอัตราเร็วในการส่งสามารถส่งรับด้วยอัตราเร็วสูงสุดของการส่งข้อมูล 15,000 bit/sec

-เมื่อต่อส่วน R.F และส่วน INTERFACE เข้าด้วยกันเนื่องจากมีสัญญาณเนื่องมาจากเซมิคอนดักเตอร์นอยส์ผ่านทางชัพพลายจึงทำให้การส่งข้อมูลผิดพลาด ซึ่งได้ทำการแก้ไขโดยการ Isolator ออกจากกันโดยใช้ OPTO-COUPLE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วน R.F. ที่ได้พัฒนาในส่วนหลัง

1.Loop สามารถ Lock และให้ความถี่ Output ได้ในช่วง

N_{min}	= 442	N_{max}	= 501
VCO_{min}	= 0.610 V	VCO_{max}	= 4.84 V
$f_{o\ min}$	= 88.4 MHz	$f_{o\ max}$	= 100.2 MHz

จะเห็นได้ว่าวงจร VCO สามารถ Oscillate ความถี่ได้ในช่วง 87 - 112 MHz โดยใช้ระดับแรงดัน VCO สูงกว่า 12 V โดยมร Conversion Gain =~ 2 MHz / V แต่ในส่วนอื่นของ PLL ใช้ระดับแรงดันเพียงแค่ 5 V (เนื่องจากข้อกำหนดของ TC9122P ที่สามารถใช้ระดับแรงดันไฟเลี้ยงสูงสุดเพียงแค่ 8.5 V เท่านั้น) ดังนั้นระดับแรงดันที่ได้จากตัว PD ที่จะไปสู่ VCO จึงไม่เกิน 5 V ซึ่งสามารถ Lock ความถี่ได้ไม่เกิน100.2 MHz เท่านั้น

2.Output ที่ได้จาก PLL มีสัญญาณรบกวนมาก โดยสัญญาณรบกวนจะมาจากวงจรผลิตความถี่อ้างอิงและวงจรหารความถี่มากที่สุด ซึ่งได้แก้ไขดังนี้

- วงจร VCO จะมีความไวต่อสัญญาณรบกวนมากจึงต้องมีการ Shield อย่างดี เพื่อป้องกันการเหนี่ยวแม่เหล็กไฟฟ้าของสัญญาณรบกวน ถ้าไม่มีการ Shield ที่ดีพอ Output ของ VCO จะเกิด Sideband จำนวนมาก เครื่องรับจะได้ยินเสียงความถี่ f_R

- การเชื่อมต่อวงจรในแต่ละส่วนของ PLL ต้องใช้สาย Coaxial เพื่อป้องกันสัญญาณรบกวน

3.เมื่อเกิดมีการสั่นสะเทือนกับวงจร จะเกิดการ Modulate ความถี่ของการสั่นสะเทือนเข้ากับสัญญาณ Carrier ของ VCO ซึ่งแก้ไขได้โดย

- ติดตั้งวงจร VCO ให้มั่นคงที่สุด

- ออกแบบให้ L_1 มีขนาดเล็กที่สุด , ใช้ลวดเส้นใหญ่ ๆ ในการพัน L และ ทายาง Silicone หุ้มขดลวดไว้เพื่อป้องกันการสั่นสะเทือน (หรืออาจใช้การสร้าง L_1 จาก Transmittion Line ซึ่งไม่ได้ทำในที่นี้)

4.วงจร PLL สังเคราะห์ความถี่ได้ไม่เท่ากันตลอดย่านความถี่ เนื่องจากวงจรหารจะมีความสามารถเปลี่ยนไปเมื่อความถี่เพิ่มขึ้น แต่ค่าที่ผิดพลาดเป็นค่าที่น้อยสามารถยอมกับได้ การแก้ไขทำได้โดยการจูน C_1 L_1 ให้ผิดพลาดน้อยที่สุดในช่วงแคบ ๆ ที่ต้องการเท่านั้นไม่สามารถทำได้ตลอดช่วง

5.Output ที่ได้จาก VCO มีค่าประมาณ -5 dBm

6.อัตราการขยายกำลังของ วงจร Power Amp.ประมาณ 36 dB Bandwidth อยู่ในช่วง 87 - 110 MHz

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใด ๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7.วงจร Power Amplifier ให้ผลในการทำงานไม่เท่ากันตลอดย่านความถี่ โดยจะให้กำลังสูงสุดที่

ความถี่ประมาณ 100 MHz ประมาณ 2 Watts และ ประมาณ 1 Watt ที่ความถี่ประมาณ 88 MHz แก้ไขโดยการปรับวงจร Matching ในแต่ละส่วนของวงจร Power Amp. ให้ได้กำลังสูงสุดที่ความถี่ที่ใช้งาน

8.ประสิทธิภาพรวมของวงจร ประมาณ 50%

9.วงจร Power Amp. Q_3 จะมีอุณหภูมิสูงเมื่อทำงานที่ Output สูงสุดเป็นเวลานาน กำลังของวงจร Power Amp. จะตกลงตามอุณหภูมิที่สูงขึ้น โดยจะมีกำลังประมาณ 1.5 Watt เมื่ออุณหภูมิสูงกว่า 60 °C (จากการประมาณโดยการสัมผัส) แก้ไขโดยการใช้แผ่นระบายความร้อนที่ใหญ่ขึ้น

10.สามารถส่งข้อมูลด้วยอัตราเร็วสูงสุดประมาณ 4800 Bit/Sec เนื่องจาก Upper Frequency จะถูกจำกัดโดย Loop Filterใน PLL ที่เป็น Low-Pass Filter

11.ระยะทางในการส่งข้อมูล สามารถส่งข้อมูลในอัตราเร็ว 1200 Bit/Sec ในระยะที่มากกว่า 2 Km ได้โดยไม่เกิดการผิดพลาด

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

บทวิจารณ์และสรุป

สรุปผลการทดลอง ในโครงงานนี้ในส่วนของการ INTERFACE กับ PC นั้นเราสามารถรับส่งข้อมูลได้อย่างถูกต้องด้วยอัตราเร็วในการส่งข้อมูล 9600 baud โดยใช้ PROTOCAL แบบ XMODEM ส่งผ่านคลื่นความถี่วิทยุ 300 MHz ได้เป็นผลสำเร็จ และปัญหาที่พบคือ เนื่องจากสัญญาณรบกวนจากคอมพิวเตอร์ผ่านทางขั้วพลายเข้ามารบกวนเครื่องส่งวิทยุ ได้แก้ไขโดยใช้ OPTO-ISOLATOR แยกแหล่งจ่ายไฟออกจากกัน ทำให้สามารถติดต่อข้อมูลผ่านคลื่นความถี่วิทยุได้ สำหรับเครื่องรับส่งวิทยุชุดที่ใช้ในโครงงานนี้เมื่อปรับได้ที่แล้ว สามารถรับส่งได้ระยะทางประมาณ 100 เมตร โดยรับส่งข้อมูลได้ถูกต้อง

ในโครงงานนี้ใช้วงจรเครื่องรับความถี่วิทยุ ในการรับส่ง 2 ชุด ซึ่งเป็นการติดต่อแบบฮาล์ฟดูเพลก การติดต่อจะทำเนียงด้านใดด้านหนึ่งเท่านั้น แต่เพื่อลดการรบกวนซึ่งกันและกันจึงให้ความถี่ในของทั้ง 2 ชุดแตกต่างกัน

สำหรับในการพัฒนาขั้นตอนไป ควรจะพัฒนา SOFTWARE ให้มีความอ่อนตัวมากกว่านี้เพื่อนำไปใช้งานได้หลายด้าน อีกทั้งควรทำให้เป็น Resident Program

---

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมสำหรับเครื่องแม่(MASTER)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
CGROUP  GROUP   CODE_SEG,DATA_SEG
        ASSUME  CS:CGROUP,DS:CGROUP
code_seg        segment   public
code_seg        ends ; BEGIN

DATA_SEG        SEGMENT  PUBLIC
namebuff         db   49
                 db   ?
                 db   50 dup(?)
    buff         equ  20000
    databuff     db   buff dup(?)
    intro        db   0dh,0ah,'Enter  Pathname for send:','$'
    crlf         db   0dh,0ah,'$'
    handle       dw   ?
    psum         db   100 dup(0)
    summing      db   10,13,10,13,'Summing date block :  $'
    summings     db   ' is $'
    last         db   ?
    size         db   10,13,'File size = $'
    sizes        dw   0
    bytes        db   '    bytes$'
    count        db   1
    rec_nak      dw   0
    save_time    dw   ?
    tell_you     db   0dh,0ah,'Initial 8251 and wait for ACK'
                 db   ' form Receiver  5 secs. ','$'
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
er1             db   '****File not found***** ','$'
er2             db   '****path not found***** ','$'
er3             db   '****Too many open files****','$'
er4             db   '****Wrong attribute or access denied ****','$'
er5             db   '***Invalid handle***','$'
er6             db   '***Invalid access code ***','$'
er7             db   '***Error  or Unknow ***','$'
continue        db   'Send the next block $'
r_nak_string  db '*Send it again  $'
ready           db   '* Send data  successful $'
ACK_ready       db    'Receive ACK ready $'
ACK_fail        db   10,13,'Can not receive ACK $'
abort           db   10,13,'*** Can not contact ***!'
ssum            db   10,13,'Send  summing OK $'
ssEOF           db   10,13,'Send  end of file $'
sssETB          db   10,13,'Send  end of block $'
r_cod           db   'Who do you want to contact with ?$'
se_cod          db   10,13,'Code sending.$',10,13
c_port          equ  0301h
d_port          equ  0300h
EOT             equ  04
EOF             equ  1ah
ACK             equ  06
NAK             equ  21
ETB             equ  23
GOOD            equ  77
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
DATA_SEG        ENDS

CODE_SEG        SEGMENT  PUBLIC
        ;*************
        ;MAIN PROGRAM
        ;*************
                ORG  100H
BEGIN:          mov  ax,cs
                mov  ds,ax
                call clear_screen
                call set_cursor
                mov  dx,offset cs:r_cod
                call print_string
                call r_2char
                call s_code
                call open_file
                call read_file                  ;load file for send in memme
                call initial
                call se_data
                call tell
                call check_ACK                  ;wait for ACK
                call send_f_size                ;send file size
                call check_ACK
                mov  bx,offset cs:databuff
                mov  si,offset cs:psum
send1:          call send_data
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    mov  dl,last                    ;check last byte
                    cmp  dl,EOF
                    jz   s_EOF
                    call send_ETB
                    call send_summing
                    call check_ACK_NAK_EOT
                    cmp  ax,GOOD
                    jnz send1
s_EOF:              call send_EOF
                    call send_summing
                    call check_ACK_NAK_EOT
                    cmp  ax,NAk
                    jz   send1
exit:               mov  ah,4ch
                    int  21h
        ;***********
        ;REC 2CHAR
        ;***********
        r_2char     proc   near
                    mov  ah,1h
                    int  21h
                    mov  dl,al
                    sub  dl,30h
                    cmp  dl,9h
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                jle  digit1
                sub  dl,7h
digit1:
                mov  cl,4h
                shl  dl,cl
                int  21h
                sub  al,30h
                cmp  al,9h
                jle  digit2
                sub  al,7h
digit2:
                and  al,0fh
                add  dl,al
                ret
r_2char         endp

s_code          proc  near
                mov  bl,dl
                mov  dx,0304h
                mov  al,00h
                out  dx,al
                mov  dx,0306h
                mov  al,02h
                out  dx,al
                mov  cx,10
kh:             push cx
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                push bx
                mov  dx,offset cs:se_cod
                call print_string
                pop  bx
                pop  cx
                mov  dx,0302h
                mov  al,bl
                out  dx,al
                loop kh
                mov  dx,0302h
                mov  al,00h
                out  dx,al
                ret
s_code          endp

;########
;SE_data
;########
se_data         proc near
                mov  dx,0304h
                mov  al,00h
                out  dx,al
                mov  dx,0306h
                mov  al,00h
                out  dx,al
                ret
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
se_data     endp
;*********************
;Send Summing of block
;*********************
send_summing proc   near
s_sum:      mov  dx,c_port
            in   al,dx
            test al,01
            jz   s_sum
            mov  dx,d_port
            mov  al,[si]
            out  dx,al
            mov  dx,offset cs:ssum
            call print_string
            ret
send_summing endp
;************************************
; send data and Summing Data until
; equal 256bytes or found end of file
;************************************
send_data   proc  near
set:        mov   cx,256                ;number of bytes in 1 block
            call  delay
            xor   ax,ax
            mov   bp,ax
            mov   [si],ax               ;push zero to si before sum data
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
pop_data:   mov   al,[bx]              ;read data from buffer
            cmp   al,1ah               ;if end of file
            jz    sum1
            push  ax
status:     mov   dx,c_port
            in    al,dx
            test  al,01
            jz    status
            pop   ax
            mov   dx,d_port
            out   dx,al
            add   [si],al              ;sum data
            inc   bx
            inc   bp
            loop  pop_data

 sum1:      cmp   cx,255               ;if  no data in this block
            jz    no_data
            mov   last,al
            mov   dx,offset cs:summing ;display summing
            call  print_string
            xor   dx,dx
            mov   dl,[count]
            call  binidec
            mov   dx,offset  cs:summings
            call  print_string
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                xor   dx,dx
                mov   dl,[si]
                call  binihex
                ret
no_data:        mov   last,al
                ret
send_data       endp

    ;*********************************
    ;Check Control Signal from Receiver
    ;*********************************
    check_ACK_NAK_EOT  proc  near
                call set_time
    test_again: call read_time
                cmp  ax,[seve_time]
                jae  contact_fail
                mov  dx,c_port
                in   al,dx
                test al,02
                jz   test_again
                mov  dx,d_port
                in   al,dx
                cmp  al,ACK
                jz   correct
                cmp  al,NAK
                jz   wrong
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                cmp  al,EOT
                jz   complete
correct:        inc  si
                inc  [count]
                mov  dx,offset cs:continue
                call print_string
                mov  [rec_NAK],0
                ret

wrong:          inc [rec_NAK]
                mov ax,[rec_NAK]
                cmp ax,10
                jz  not_care
                mov ax,bp
                sub bx,ax
                mov dx,offset  cs:r_NAK_string
                call print_string
                mov  ax,NAK
                ret
not_care:       cmp  last,EOF
                jnz  correct
                jz   test_again
complete:       mov  dx,offset cs:ready
                call print_string
                mov  ax,GOOD
                ret
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
contact_fail:   mov  dx,offset  cs:abort
                call print_string
                mov  ah,4ch
                int  21h
check_ACK_NAK_EOT  endp

        ;*************************************
        ;Receive ACK for contract from Receiver
        ;*************************************
        check_ACK    proc   near
                     call   set_time
        c_ACK:       call   read_time
                     cmp    ax,[save_time]
                     jae    cancel
                     mov    dx,c_port
                     in     al,dx
                     test   al,02                ;test RxRDY
                     jz     c_ACK
                    mov    dx,d_port
                    in     al,dx
                    cmp    al,06
                    jnz    c_ACK
                    mov    dx,offset cs:ACK_ready
                    call   print_string
                    ret
        cancel:     mov    dx,offset cs:ACK_fail
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                call  print_string
                mov   ah,4ch
                int   21h
check_ACK       endp
;*****************
;Send size of file
;*****************
send_f_size  proc  near
                mov   cx,[sizes]
s_f_s:          mov   dx,c_port
                in    al,dx
                test  al,01
                jz    s_f_s
                mov   dx,d_port
                mov   al,cl
                out   dx,al
                call  delay
s_f_s1:         mov   dx,c_port
                in    al,dx
                test  al,01
                jz    s_f_s1
                mov   dx,d_port
                mov   al,ch
                out   dx,al
                ret
send_f_size  endp
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;*******
; Delay
;*******
delay       proc  near
            push  bx
            mov   bx,0
delay1:     dec   bx
            jnz   delay1
            pop   bx
            ret
delay       endp
;********************
;Open sending File
;********************
open_file   proc  near
ask:        mov  dx,offset  cs:intro        ;ask for name file
            call print_string

ans:        mov  dx,offset cs:namebuff      ;input  name file
            mov  ah,0ah
            int  21h
            mov  dx,offset cs:crlf
            call print_string

zero:       mov  bl,namebuff+1              ;put zero in name file bu
            mov  bh,0
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                mov  [namebuff+bx+2],0


open1           mov  dx,offset  cs:namebuff+2
                mov  al,0
                mov  ah,3dh                     ;open file
                int  21h
                mov  handle,ax
                jc   error1
                ret
error1:         call error
open_file       endp


;********************
;Read Sending File
;********************
read_file       proc  near
read1:          mov   cx,buff                   ;number of bytes for read
                mov   bx,handle
                mov   dx,offset cs:databuff
                mov   ah,3fh                    ;read file
                int   21h
                jc    error2                    ;if error
                cmp   ax,0                      ;if found end of file
                jz    close1
                mov   cx,ax
                cmp   cx,buff                   ;if file size more than b
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                pushf
                add   [sizes],cx                ;save it before read and
                popf
                jnz   disp                      ;if file size less than
    d1:
                mov   bx,handle
                jmp   read                      ;read again

    disp:       mov   dx,offset cr1size         ;display file
                call  print_string
                mov   dx,[sizes]
                call  binidec
                mov   dx,offset cr1bytes
                call  print_string
    close1:     call  close_file
                ret
    error2:     call  error
    read_file   endp

;************
;close_file
;************
close_file   proc  near
                mov   bx,handle
                mov   ah,3eh
                int   21h
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                jc    error3
                ret

error3:         call  error
close_file      endp

;****************
;* Initial 8251*
;****************
initial     proc  near
            mov   dx,c_port             ;control port
            mov   al,80h                ;reset
            out   dx,al
            mov   al,0ddh               ;mode word
            out   dx,al
            mov   al,15h                ;command  word
            out   dx,al
            ret
initial     endp

;*******************************
;Change Binary to Hexadecimal
;*******************************
binihex     proc  near
            push  bx
            xor   bx,bx
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                mov   bx,dx
                mov   ch,4
rot:            mov   cl,4
                rol   bx,cl
                mov   al,bl
                and   al,0fh
                add   al,30h
                cmp   al,3ah
                jl    printit
                add   al,7h
printit:        mov   dl,al
                mov   ah,2
                int   21h
                dec   ch
                jnz   rot
                mov   dx,offset crlf
                call  print_string
                pop   bx
                ret
binihex         endp
;*************************
;CHANGE BINARY TO DECIMAL
;*************************
binidec         proc  near
                push  si
                mov   ax,dx
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                mov   si,10
                xor   cx,cx
    nonzero:    xor   dx,dx
                div   si
                push  dx
                inc   cx
                or    ax,ax
                jne   nonzero
    w:          pop   dx
                call  writehexd
                loop  w
                pop   si
                ret
    binidec     endp
    writehexd   proc  near
                push  dx
                cmp   dl,10
                jae   hexletter
                add   dl,"0"
                jmp   short wdigit
    hexletter:  add   dl,"0"
    wdigit:     mov   ah,2
                int   21h
                pop   dx
                ret
    writehexd   endp
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;##########
;Tell user
;##########
tell    proc  near
        mov   dx,offset cs:tell_you
        call  print_string
        mov   dx,offset cs:crlf
        call  print_string
        ret
tell    endp

;##################
;Send End of File
;##################
send_EOF  proc  near
eof1:   mov  dx,s_port
        in   al,dx
        test al,01
        jz   etb1
        mov  dx,d_port
        mov  al,EOF
        out  dx,al
        mov  dx,offset cs:seEOF
        call  print_string
        ret
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
send_eof   endp
;#############
; READ_TIME
;#############
read_time  proc near
           push es
           mov  ax,0
           mov  es,ax
           mov  ax,word ptr es:[046ch] ;read data in address 0:46Ch
           pop  es
           ret
read_time  endp
;#################
;Send End of Block
;#################
send_ETB   proc  near
etb1:      mov  dx,c_port
           in   al,dx
           test al,01
           jz   etb1
           mov  dx,d_port
           mov  al,ETB
           out  dx,al
           mov  dx,offset cs:seeETB
           call print_string
           ret
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
send_ETB   endp



;############
; SET_TIME
;############
set_time   proc  near
           push  es
           mov   ax,0
           mov   es,ax
           mov   ax,word ptr es:[046ch] ;read data in address 0:46h
           add   ax,200
           mov   [save_time],ax
           pop   es
           ret
set_time   endp


;################
; PRINT STRING
;################
print_string proc near
           push  cx
           push  bx
           mov   ah,09
           int   21h
           pop   bx
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                pop   cx
                ret
print_string  endp


;##############
; CLEAR_SCREEN
;##############
clear_screen proc  near
                push ax
                push bx
                push cx
                push dx
                xor  ax,ax
                xor  cx,cx
                mov  dh,24
                mov  dl,79
                mov  bh,7
                mov  ah,6
                int  10h
                pop  dx
                pop  cx
                pop  bx
                pop  ax
                ret
clear_screen endp
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;************
; SET_CURSOR
;************
set_cursor proc  near
           push  ax
           push  bx
           push  dx
           xor   dx,dx
           xor   bx,bx
           mov   ah,02       ;set cursor routine
           int   10h         ;goto column 0,row 0
           pop   dx
           pop   bx
           pop   ax
           ret
set_cursor endp

;********************
;WAIT FOR KEY Y OR N
;********************
dicision   proc  near
t1:        mov ah,01
           int 21h
           cmp al,59h
           jz  exit1
           cmp al,4eh
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                jz  exit1
                cmp al,79h
                jz  t2
                cmp al,69h
                jz  t2
                cmp al,59h
                jz  t2
;               mov dx,offset os:fault
                call print_string
                jmp t1
t2:             ret
exit1:          mov ah,4ch
                int 21h
division  endp

;###############
;ERROR ROUTINE
;###############
error     proc  near
                cmp ax,2
                jz  e1
                cmp ax,3
                jz  e2
                cmp ax,4
                jz  e3
                cmp ax,5
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                jz  e4
                cmp ax,6
                jz  e5
                cmp ax,12
                jz  e6
                jmz e7
e1:             mov dx,offset cster1
                jmp e0
e2:             mov dx,offset cster2
                jmp e0
e3:             mov dx,offset cster3
                jmp e0
e4:             mov dx,offset cster4
                jmp e0
e5:             mov dx,offset cster5
                jmp e0
e6:             mov dx,offset cster6
                jmp e0
e7:             mov dx,offset cster7
                jmp e0
e0:             call print_string


                mov ah,4ch
                int 21h
error           endp
```

เอกสารนี้เป็นเอกสารสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
code_seg        ends

                end     begin
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมสำหรับเครื่องลูก(SLAVE)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
CGROUP  GROUP   CODE_SEG,DATA_SEG

        ASSUME  CS:CGROUP,DS:CGROUP

code_seg        segment   public

code_seg        ends


DATA_SEG        SEGMENT  PUBLIC

ums             db       10,13,'Send ACK umsuccessful ,$'

aski_user       db       'DO you want to contact ?(Y/N)$'

ACK_ready       db       10,13,'Send ACK ready.$',10,13

crlf            db       10,13,'$'

er1             db       '**** File not found*******','$'

er2             db       '**** Path not found*******','$'

er3             db       '**** Too  many open file**','$'

er4             db       10,13,'**** Wrong attribute or Access denied **','$',10,

er5             db       '**** Invalid handle ******','$'

er6             db       '**** Invalid access code **','$'

er7             db       '**** Error  or Unknow *****','$'

tell_f_s        db       10,13,'File size = $'

cmt             db       10,13,'Can not connect$',10,13

fould_BTB       db       10,13,'Fould end of block$',10,13

fould_EOF       db       10,13,'Fould end of file $',10,13

t_nak           db       10,13,'Send NAK and receive data again. $'

d_sum           db       'Can not receive sum of data. $'

save_time       dw       ?

fault           db       10,13,'Press only key Y or N. $',10,13

sEOT            db       10,13,'Data in ready and contact complete.',10,10,'$'
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
handle          dw      ?
c_port          equ     0301h
d_port          equ     0300h
num_NAK         dw      0
ccunt           db      1
bytes           db      ' bytes#',10,13
lsst1           db      0
?sum            db      ?
psum            db      100 dup(0)
databuff        db      20000 dup(?)
mf_s            db      ?
lf_s            db      ?
EOT             equ     04
EOP             equ     1ah
ACK             equ     06
NAK             equ     21
file_sizes      dw      0
intro1          db      0dh,0ah,'SAVE data in file name:','$'
namebuff1       db      48
                db      ?
                db      50 dup(?)
ETB             equ     28
fail            equ     24
told            db      0dh,0ah,'Program run at receive_data#',10,13
told1           db      0dh,0ah,'Program run at check_sum#',10,13
told2           db      0dh,0ah,'Program run at send_NAK#',10,13
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
told3           db      0dh,0ah,'Program run at send_EOT$',10,13
wait_int        db      'Waiting for interrupt (IRQ2).....$'
ans_int         db      10,13,'#Interrupt Complete# $'
DATA_SEG        ENDS


;ST_SEG         SEGMENT  stack
;               DB       20 DUP(?)
;ST_SEG         ENDS


CODE_SEG        SEGMENT  PUBLIC
                ORG  100h
BEGIN:          mov  ax,cs
                mov  ds,ax

                in   al,21h             ;Program 8259 int
                and  al,0fbh
                out  21h,al
                push ds
                mov  ax,cs
                mov  ds,ax
                mov  ah,25h             ;set new int vector
                mov  al,0ah
                mov  dx,offset cs:main
                int  21h
                pop  ds
                call clear_screen
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                call  set_cursor
                mov   dx,offset cs:wait_int
                mov   ah,09h
                int   21h

main1:          mov   ah,01
                int   16h                   ;check that press any
                jz    main1

                mov   ah,4ch                ;ret to dos
                int   21h

;*************
;MAIN PROGRAM
;*************
main            proc
                push ax
                push bx
                push cx
                push dx
                push si
                push di
                push es
                push ds
                mov  dx,offset  cs:ems_int
                call print_string
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                call initial
                mov  dx,offset cs:ask1_user
                call print_string
                call division
                call se_data
                call send_ACK
                call rec_file_size
                call send_ack
                mov  di,offset cs:dataBuff
                mov  si,offset cs:pswm
DDATA:          call receive_data
                cmp  ax,fail
                jz   exit
                call check_sum
                cmp  ax,NAK
                jz   s_nak
                cmp  last1,EOF
                jz   s_EOT
                cmp  ax,fail
                jz   out1
                call send_ack
                inc  si
                jmp  DDATA
s_NAK:          inc  [num_NAK]
                call send_NAK
                mov  ax,[num_NAK]
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                cmp  ax,10
                jz   not_care
                jmp  DDATA

not_care:       cmp last1,EOF
                jz  s_EOT
                xor ax,ax
                mov [num_NAK],ax
                mov ax,di
                sdd ax,bp
                mov di,ax
                jmp DDATA

s_EOT:          call send_EOT

out1:           call create_file
                call write_file
                call close_file

EXIT:           mov  al,20h                 ;enable interrupt
                out  20h,al
                pop  ds
                pop  es
                pop  di
                pop  si
                pop  dx
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                pop  cx
                pop  bx
                pop  ax
                lret
  main          endp

;##################
;Select send data
;##################
se_data         proc  near
                mov   dx,0304h
                mov   al,00h
                out   dx,al
                mov   dx,0306h
                mov   al,00h
                out   dx,al
                ret
se_data         endp
;#################
;Select send code
;#################
se_code         proc  near
                mov   dx,0304h
                mov   al,00h
                out   dx,al
                mov   dx,0306h
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                mov   al,02h
                out   dx,al
                mov   dx,0302h
                mov   al,88h
                out   dx,al
                ret
se_code         endp

;#############################################
;Receive size of file from master
;#############################################
rec_file_size  proc near
                call set_time
recf1:          call read_time
                cmp  ax,[leave_time]
                jae  EEXIT
                mov  dx,c_port
                in   al,dx
                test al,02                      ;test RxRDY
                jz   recf1
                mov  dx,d_port
                in   al,dx
                mov  [lf_s],al                  ;receive
recf2:          mov  dx,c_port
                in   al,dx
                test al,02
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                jz   recf2
                mov  dx,d_port
                in   al,dx
                mov  [mf_s],al
                mov  ah,[mf_s]
                mov  al,[lf_s]
                mov  [file_sizes],ax
                call bitodec
                mov  dx,offset cs:bytes
                call print_string
                ret
BEXIT:          mov  dx,offset cs:msg
                call print_string
                mov  ah,4ch
                int  21h
rec_file_size endp
;##################
;Send Acknowledge
;##################
send_ACK        proc  near
                call  set_time
  AACK:         call read_time
                cmp  ax,[save_time]
                jae  wnsu0
                mov  dx,d_port
                in   al,dx
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                test al,01
                jz   AACK
                mov  al,ACK
                mov  dx,d_port
                out  dx,al
                mov  dx,offset cs:ACK_ready
                call print_string
                ret
        unsv01: mov  dx,offset cs:vns
                call print_string
                mov  ah,4ch
                int  21h
 send_ACK       endp

 ;##########################
 ; RECEIVE DATA AND SUM DATA
 ;##########################
 receive_data   proc near
 receive1:      xor  ax,ax
                mov  bp,ax
                call set_time
                mov  dx,offset cs:told
                call print_string
receive_data1:
                call read_time
                cmp  ax,[SAVE_TIME]
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                jae  abort
                mov  dx,c_port
                in   al,dx
                test al,02                  ;test RxRDY
                jz   receive_data1
                mov  dx,d_port
                in   al,dx
                cmp  al,ETB
                jz   EETB
                cmp  al,EOF
                jz   EEOF
                mov  [di],al
                add  [si],al
                inc  di
                inc  bp
                jmp  receive_data1

EETB:           mov  dx,offset cs:fould_ETB
                call print_string
                ret

EEOF:           mov  test1,al
                mov  dx,offset cs:fould_EOF
                call print_string
                ret
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
ABORT:          mov  dx,offset cs:cmt
                call print_string
                mov  ax,fail
                ret
receive_data  endp



;######################
;SEND NEG ACKNOWLEDGE
;######################
send_NAK  proc  near
          push  cx
          mov   dx,offset cs:told2
          call  print_string
nNAK:     mov   dx,s_port
          in    al,dx
          test  al,01
          jz    nNAK
          mov   dx,d_port
          mov   al,NAK
          out   dx,al
          mov   dx,offset  cs:t_nak
          call  print_string
          mov   ax,di
          mov   cx,bp
          sub   ax,cx
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                mov   di,ex
                pop   cx
                ret
send_NAK        endp


;#################
;CHECK BOTH OF SUM
;#################
check_sum  proc near
           call  set_time
           mov   dx,offset os1to1d1
           call  print_string
s_sum1:    call  read_time
           cmp   ax,[save_time]
           jae   s_sum3
           mov   dx,c_port
           in    al,dx
           test  al,02
           jz    s_sum
           mov   dx,d_port
           in    al,dx
           mov   ?sum,al
           mov   al,[si]
           cmp   ?sum,al
           jnz   s_sum2
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
ABORT!          mov  dx,offset cs!cmt
                call print_string
                mov  ax,fail
                ret
receive_data    endp



;######################
;SEND NEG ACKNOWLEDGE
;######################
send_NAK  proc  near
          push  cx
          mov   dx,offset cs:toldZ
          call  print_string
nNAK!     mov   dx,c_port
          in    al,dx
          test  al,01
          jz    nNAK
          mov   dx,d_port
          mov   al,NAK
          out   dx,al
          mov   dx,offset  cs:t_nak
          call  print_string
          mov   ax,di
          mov   cx,bp
          sub   ax,cx
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                mov   di,ax
                pop   cx
                ret
send_NAK        endp


;******************
;CHECK BOTH OP SUM
;******************
check_sum       proc near
                call  set_time
                mov   dx,offset cs:told1
                call  print_string
s_sum1:         call  read_time
                cmp   ax,[save_time]
                jae   s_sum2
                mov   dx,c_port
                in    al,dx
                test  al,02
                jz    s_sum
                mov   dx,d_port
                in    al,dx
                mov   ?sum,al
                mov   al,[si]
                cmp   ?sum,al
                jnz   s_sum2
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            ret


s_sum2:     mov   ax,nak
            ret


s_sum3:     mov   dx,offset cs:d_sum
            call  print_string
            mov   ax,fail
            ret
check_sum endp
;##########################
;SEND END OF TRANSMISSION
;##########################
send_EOT  proc near
          mov dx,offset cs:told3
          call print_string
sEOT1:    mov dx,c_port
          in  al,dx
          test al,01
          jz  sEOT1
          mov dx,d_port
          mov al,EOT
          out dx,al
          mov  dx,offset cs:sEOT
          call print_string
          ret
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
send_EOT  endp
;#################
;CREATE REC FILE
;#################
create_file proc near
sek1:     mov dx,offset cs:intro1
          mov ah,9
          int 21h

ans1:     mov dx,offset cs:namebuff1
          mov ah,0ah
          int 21h

zero1:    mov bl,namebuff1+1
          mov bh,0
          mov [namebuff1+bx+2],0

cre1:     mov dx,offset cs:namebuff1+2
          mov cx,0
          mov ah,3ch
          int 21h
          mov handle,ax
          jc  error1
          ret

error1:   call error
create_file endp
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;**********************
;WRITE REC FILE TO DISK
;**********************
write_file proc near
write1:   mov cx,[file_sizee]
          mov bx,handle
          mov dx,offset cs:databuff
          mov ah,40h
          int 21h
          jc  error2
          ret
error2:   call error
write_file endp
;*****************
; CLOSE REC  FILE
;*****************
close_file proc near
          mov bx,handle
          mov dx,offset cs:databuff
          mov ah,3eh
          int 21h
          jc  error3
          ret
error3:   call error
close_file  endp
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;*************
;INITIAL 8251
;*************
initial   proc  near
          mov   dx,c_port           ;control port
          mov   al,55h              ;reset
          out   dx,al
          mov   al,0ddh             ;mode word
          out   dx,al
          mov   al,15h              ;command  word
          out   dx,al
          ret
initial   endp

;*************************
;CHANGE BINARY TO DECIMAL
;*************************
bitodec   proc  near
          push  si
          mov   si,10
          xor   cx,cx
nonzero:  xor   dx,dx
          div   si
          push  dx
          inc   cx
          or    ax,ax
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                jne   nonzero
    w1          pop   dx
                call  writehex
                loop  w
                pop   si
                ret
    bitodec     endp
    writehex    proc  near
                push  dx
                cmp   dl,10
                jae   hexletter
                add   dl,"0"
                jmp   short wdigit
    hexletter:  add   dl,"0"
    wdigit:     mov   ah,2
                int   21h
                pop   dx
                ret
    writehex    endp



    ;#############
    ; READ_TIME
    ;#############
    read_time   proc near
                push es
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                mov  ax,0
                mov  es,ax
                mov  ax,word ptr es:[046ch] ;read data in address 0:46hh
                pop  es
                ret
read_time  endp

;############
; SET_TIME
;############
set_time   proc  near
                push  es
                mov   ax,0
                mov   es,ax
                mov   ax,word ptr es:[046ch] ;read data in address 0:46hh
                add   ax,200
                mov   [save_time],ax
                pop   es
                ret
set_time   endp

;################
; PRINT STRING
;################
print_string proc near
                push  ax
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                mov   ah,09
                int   21h
                pop   ax
                ret
print_string  endp

;###############
; CLEAR_SCREEN
;###############
clear_screen proc  near
                push ax
                push bx
                push cx
                push dx
                xor  ax,ax
                xor  cx,cx
                mov  dh,24
                mov  dl,79
                mov  bh,7
                mov  ah,6
                int  10h
                pop  dx
                pop  cx
                pop  bx
                pop  ax
                ret
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
clear_screen endp


;############
; SET_CURSOR
;############
set_cursor proc  near
           push  ax
           push  bx
           push  dx
           xor   dx,dx
           xor   bx,bx
           mov   ah,02      ;set cursor routine
           int   10h        ;goto columm 0,row 0
           pop   dx
           pop   bx
           pop   ax
           ret
set_cursor endp


;####################
;WAIT FOR KEY Y OR N
;####################
division   proc  near
t1:        mov ah,01
           int 21h
           cmp al,6eh
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                jz  exit1
                cmp al,4eh
                jz  exit1
                cmp al,79h
                jz  t2
                cmp al,59h
                jz  t2
                cmp al,58h
                jz  t2
                mov dx,offset default
                call print_string
                jmp t1
t2:             ret
exit1:          mov ah,4ch
                int 21h
division        endp

;*************
;ERROR ROUTINE
;*************
error           proc  near
                cmp ax,2
                jz  e1
                cmp ax,3
                jz  e2
                cmp ax,4
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                jz  e3
                cmp ax,5
                jz  e4
                cmp ax,6
                jz  e5
                cmp ax,12
                jz  e6
                jmp e7
e1:             mov dx,offset cater1
                jmp e0
e2:             mov dx,offset cater2
                jmp e0
e3:             mov dx,offset cater3
                jmp e0
e4:             mov dx,offset cater4
                jmp e0
e5:             mov dx,offset cater5
                jmp e0
e6:             mov dx,offset cater6
                jmp e0
e7:             mov dx,offset cater7
                jmp e0
e0:             call print_string
                mov ah,4ch
                int 21h
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
error     endp

code_seg          ends

          end     begin
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

โครงงานนี้มีอาจสำเร็จได้ตามจุดประสงค์ ถ้าขาดการให้คำแนะนำจากอาจารย์ที่ปรึกษา อ.พลพคุง ผดุงกุล และอาจารย์ท่านอื่นอีกหลายท่าน ตลอดจนความร่วมมือในการคิดแก้ไขปัญหาจาก เพื่อน ๆ ภาควิชาอิเลคทรอนิคส์และภาคคอมพิวเตอร์ จึงขอขอบคุณมา ณ.ที่นี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หนังสืออ้างอิง

JAMES W.COFFRON,"Z-80 APPLICATION ",COPY RIGHT,SYBEX INC, PP.209-238,1987

RUSSEL RECTOR AND GEOGE ALEXY OSBOURNE ,"THE 8086 BOOK", MCGRAW-HILL,1984

RAY DUNCAN,"ADVANCED MS DOS VERSION 1.1-3.2" MICROSOFT PRESS,1986

PETER ABEL,"ASSEMBLER FOR THE IBM PC AND PC-XT" PRENTICE HALL,1984

รศ ยืน ภู่สุวรรณ และ น.ต.ดร.ไพศาล สงวนหมู่,"การสื่อสารข้อมูล และ ไมโครคอมพิวเตอร์เนตเวอร์ค."บริษัท ซีเอ็ดยูเคชั่น จำกัด ,พิมพ์ครั้งที่ 9,2529

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้